World Congress on Neural Networks
1996 International Neural Network Society
pp. 1227-1231

Automated Medical Diagnosis based on Decision Theory and Learning from Cases

Magnus Stensmo
Computer Science Division
University of California
Berkeley, CA 94720, U.S.A.
magnus@cs.berkeley.edu

Terrence J. Sejnowski
Computational Neurobiology Lab
The Salk Institute
10010 North Torrey Pines Road
La Jolla, CA 92037, U.S.A.
terry@salk.edu

Abstract

Medical diagnosis is an important but complicated task whose automation would be very useful. A scarce resource could be made less so and diagnosis could be made more accurate and efficient in both monetary terms and in reduced suffering or pain for a patient. We have used decision and probability theory to construct such systems from a database of typical cases. This simplifies the task of knowledge extraction from physicians, who often do not know how they have come to a certain diagnosis. Probability models are constructed using mixture models that are efficiently learned by the Expectation-Maximization algorithm. Problems with missing data are then solved, both for missing data in the case database and during diagnosis (missing data are then observations not yet made). Decision theory is used to find the most informative next question to ask, observation to make, or test to do in order to minimize the total cost for the diagnosis. It is also used to decide when to stop requesting more information. To automatically find good utility values for the decision theoretic model, temporal-difference reinforcement learning is used to increase the system's accuracy. Results are presented on a case database for heart disease.

1 Introduction

Medical diagnosis is an important task that should be performed as accurately and efficiently as is possible. All doctors are unfortunately not equally skilled in every subspeciality and they are in many places a scarce resource. A system for automated medical diagnosis would enhance medical care and reduce costs.

Rule-based expert systems are well known examples of such automated diagnosis systems. They can however be very complicated to design since a physician often cannot express his reasoning in simple if-then rules, and they cannot reason with uncertainties. It would be desirable to learn models instead of manually constructing them. The learning could be done from a database of previous cases. By doing this we can overcome the knowledge acquisition problem.

Decision theory which is normative, or prescriptive, can tell us how to behave optimally in a situation [French, 1988]. Optimal here means to maximize the value of the expected future outcome. This has been formalized as the *maximum expected utility principle* by [von Neumann and Morgenstern, 1947].

Decision theory can be used to make optimal choices based on probabilities and utilities. *Probability theory* tells us how probable different future states are, and how to reason with and represent uncertainty information. *Utility theory* values these possible states so that they can be compared to each other. In evidence gathering during a diagnosis session, it can be used to determine which new possible evidence will be most efficient to acquire next. It will also tell us when further evidence will no longer improve the accuracy of the diagnosis.

2 Probability and Decision Theory Models

Our automated diagnosis system is based on hypotheses and deductions according to the following steps:

1. Observations are made. This means that one or several observation variables are set to one of its possible values.

2. The system constructs a *differential diagnosis*, *i.e.*, estimates probabilities for the different possible outcomes. The system uses a probability model to calculate the conditional probability for each of the outcomes given the current observations.

3. The next observation that would be most useful is recommended. In this step each possible next variable is considered. The expected value of the system prediction with this variable observed is then calculated. From this we subtract the current maximum value before making the additional observation. Then the cost of the observation is subtracted to reach to the net *value of information* [Howard, 1966] for this variable. The variable with the maximum of all of these is then the best next observation to make. Note that we only look ahead one step (called a *myopic* approximation [Gorry and Barnett, 1967]). This is in principle suboptimal, but the reinforcement learning procedure described below will rectify this.

4. The steps 1–3 above are repeated until nothing more can be recommended. This happens when none of the net value of information values in step 3 is positive.

We build separate probability and decision theory models. The probability model is used to predict the probabilities for the different diseases that can occur. By modeling the joint probabilities these predictions are available no matter how many or few of the input variables are available at a certain instant. Diagnosis is thus a missing data problem due to the question-and-answer cycle above. Joint probabilities are modeled using *mixture models* [McLachlan and Basford, 1988]. Such models can be efficiently trained using the *Expectation-Maximization* (EM) *algorithm* [Dempster *et al.*, 1977], which has the additional benefit that missing variable values in the training data also can be handled correctly. Most databases are incomplete in this way.

The utilities are values assigned to different states so that their usefulness can be compared. We can then chose our actions to maximize the expected future outcome. We represent utilities as preferences when a certain disease has been classified but the patient in reality has another one [Howard, 1980; Heckerman *et al.*, 1992]. For each pair of diseases there is a value between 0 and 1, where a 0 means very bad and a 1 means very good. This is a $d \times d$ matrix for d diseases. To find these values is an additional knowledge acquisition problem which is non-trivial.

Methods have been devised to convert people's perceived risk to monetary values [Howard, 1980]. They are asked to answer questions such as: "How much would you have to be paid to accept a one in a millionth chance of instant painless death?" The answers are recorded for various low levels of risk. It has been empirically found that people are relatively consistent in this, and that this is linear for low levels of probability. Howard defined the unit *micromort* (mmt) to mean *one in 1 millionth chance of instant painless death* to simplify this. [Heckerman *et al.*, 1992] found that one subject valued 1 micromort to $20 (in 1988 US dollars) linearly to within a factor of two. We use this to convert utilities in probability units ([0,1]) to values in dollars and vice versa.

Previous systems ask an expert to supply the utility values, which can be very complicated, or use some simple approximation. [Heckerman *et al.*, 1992] use a 1 for misclassification when both diseases are malign or both are benign, and 0 otherwise (see Figure 3, left). They claim that it worked in their system but, as we will see below, this crude approximation will reduce the accuracy. We show how to adapt and learn utilities to find better ones. Utilities are adapted using a type of *reinforcement learning*, more specifically the method of *temporal differences* (TD) [Sutton, 1988]. This method is capable of adjusting the utility values correctly even though we only get an error signal after each full sequence of questions in a diagnosis session.

More details of the various system components can be found in [Stensmo, 1995; Stensmo and Sejnowski, 1995a; Stensmo and Sejnowski, 1995b].

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 fourteen variables as shown in Figure 1. Continuous variables were converted into a *1-of-N* binary code based on their distributions among the cases in the database. Nominal and categorical variables were coded with one unit per value. In total 55 binary variables coded the 14 original variables.

To find the parameter values for the mixture model that is used for probability estimation, the EM algorithm was run until convergence. 98 binomial mixture components gave the best classification results using just the mixture model

	Observation	Description	Values	Cost	Cost $
1	age	Age in years	continuous	0	0
2	sex	Sex of subject	male/female	0	0
3	cp	Chest pain	four types	0.00002	20
4	trestbps	Resting blood pressure	continuous	0.00004	40
5	chol	Serum cholesterol	continuous	0.0001	200
6	fbs	Fasting blood sugar	<, or > 120 mg/dl	0.0001	200
7	restecg	Resting electrocardiographic result	five values	0.0001	200
8	thalach	Maximum heart rate achieved	continuous	0.0001	200
9	exang	Exercise induced angina	yes/no	0.0001	200
10	oldpeak	ST depression induced by exercise relative to rest	continuous	0.0001	200
11	slope	Slope of peak exercise ST segment	up/flat/down	0.0001	200
12	ca	Number major vessels colored by flouroscopy	0-3	0.0001	200
13	thal	Defect type	normal/fixed/ reversible	0.0001	200
	Disorder	**Description**	**Values**		
14	num	Heart disease	Not present/ four types		

Figure 1: The Cleveland Heart Disease database.
The database consists of 303 cases. Costs are somewhat arbitrarily assigned and are given in both probability units [0,1] and converted to corresponding dollar values based on $20 per micromort (one in 1 millionth chance of instant painless death).

to predict the probabilities of the diseases 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. Note that all of the observation variables were set to their correct values for each case, and that utility values were not used in this case.

To evaluate how well the system does, we go through every case in the database and answer the questions that come up according to the correct values for the case. When the system is done with the diagnosis sequence, the result is compared to the actual disease that was recorded in the database. The number of questions that were answered for each case is also recorded. After all of the cases have been processed in this way, the average number of questions needed, the standard deviation for this, and the number of errors are calculated.

The system's answer can be interpreted in two ways: One is to look at the *most probable* disease that is present. This is the disease that has the highest conditional probability given whatever questions were answered for this particular case. The other interpretation is to look at which disease that has the *highest expected utility* using the current utility matrix. These two answers can be quite different. If the system has several best answers, one is selected randomly. This can give rise to some random fluctuations.

Observation costs were assigned to the different variables according to Figure 1. Using the full utility/decision model and the 0/1-approximation for the utility matrix (left part of Figure 3), there were 89 errors using the probability prediction, and 130 errors with the corresponding utility prediction. Over the whole data set an average of 4.42 questions were used with a standard deviation of 2.02. Asking about 4 questions instead of 13 is much quicker but unfortunately less accurate. This is without adapting the utilities.

With TD learning, we were able to decrease the number of errors to 47 for probability measurement and 48 for utility prediction with 20 repeated presentations of all of the cases. The TD algorithm is controlled by two parameters, a learning rate α, and a discount parameter λ that controls how predictions at different states in a sequence relate to each other. We varied λ from 0 to 1 in increments of 0.1, and varied α over several orders of magnitude to find the reported results. The resulting average number of questions were 6.33 with a standard deviation of 1.90. The initial utility

Utility model	Error: Probability	Utility	Number of questions	Standard deviation
Probability model only	46	—	13	—
0/1 approximation	89	130	4.42	2.02
After TD learning	47	48	6.33	1.90

Figure 2: Results on the Cleveland Heart Disease Database.
The three methods are described in the text. The first method does not use a utility matrix. The 0/1 approximation use the matrix in Figure 3, left. The utility matrix that was the result of the TD learning is shown in Figure 3, right.

Initial					After 12 iterations				
1	0	0	0	0	0.9495	0.0870	0.0609	0.0496	0.0505
0	1	1	1	1	0.0294	0.7351	0.1961	0.1840	0.5029
0	1	1	1	1	0.0030	0.1075	0.7321	0.2100	0.4987
0	1	1	1	1	0.0118	0.0644	0.1509	0.7885	0.4816
0	1	1	1	1	0.0004	0.0520	0.1060	0.1304	0.8494

Figure 3: Misclassification utility matrices.
Left: Initial utility matrix. *Right:* After 12 iterations with discount-rate parameter λ=0.1, and learning rate α=0.005 for the TD algorithm.

matrix is shown in the left side of Figure 3. The utility matrix after 12 iterations is shown on the right, with α=0.005 and λ=0.1. After 12 iterations the results stay the same up to 30 iterations which is where we stopped, except for some minor fluctuations in the average number of questions.

The results are summarized in Figure 2. As can be seen, the price paid for increased robustness is an increase in the average number of questions from 4.42 to 6.33, but note that we can then 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. Note that most people intuitively think that half of the questions should be enough. There is however no logical reason for that, and you do not know when to stop asking questions if you just pick some random observations. The results on the same database reported in [Stensmo and Sejnowski, 1995b] are slightly different due to the use of a slightly different TD algorithm: the diagonal of the utility matrix was forced to be one, all costs were set to zero, and the minimum allowed gain was 0.00001 instead of 0.

4 Summary and Discussion

We have designed a system for automated medical diagnosis that is based on probability, utility and decision theory. This allows for principled handling of missing data, observation selection and incorporation of costs into the process. All of the system parameters can in addition be learned from a database of cases.

This method can be extended in several directions. Here are some interesting openings that we are currently pursuing:

- Utilities are considered to be non-linear [Grimmett and Stirzaker, 1992; Russell and Norvig, 1995] but are, probably by convenience, often modeled linearly (as with our misclassification utility matrix) in decision theory, since manual specification and interpretation of the utility values then is quite straight-forward. We believe that non-linear functions for utilities could be very useful. This can also be implemented through TD learning using a multilayer utility prediction 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, as in Q-learning [Watkins and Dayan, 1992; Tsitsiklis, 1994]. This would require one to learn which question to ask in each situation instead of the utility values but would depart from the normal decision theoretic formulation using the maximum expected utility principle.

Acknowledgements

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

References

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

French, Simon (1988). *Decision Theory: An Introduction to the Mathematics of Rationality*. Ellis Horwood, Chichester, UK.

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

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

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

Howard, Ron A. (1966). Information value theory. *IEEE Transactions on Systems Science and Cybernetics*, **SSC-2**, 22–26.

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

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

Russell, Stuart J. and Norvig, Peter (1995). *Artificial Intelligence: A Modern Approach*. Prentice Hall, Englewood Cliffs, NJ.

Stensmo, Magnus (1995). *Adaptive Automated Diagnosis*. PhD thesis, Kungliga Tekniska Högskolan (Royal Institute of Technology), Stockholm, Sweden.

Stensmo, Magnus and Sejnowski, Terrence J. (1995a). A mixture model system for medical and machine diagnosis. In Tesauro, G., Touretzky, D. S. and Leen, T. K., editors, *Advances in Neural Information Processing Systems*, volume 7, pp 1077–1084. MIT Press, Cambridge, MA.

Stensmo, Magnus and Sejnowski, Terrence J. (1995b). Using temporal-difference reinforcement learning to improve decision-theoretic utilities for diagnosis. In *Proc. of the 2nd Joint Symposium on Neural Computation, University of California, San Diego and California Institute of Technology*, pp 9–16. Institute for Neural Computation, La Jolla, CA.

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

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

Tsitsiklis, John N. (1994). Asynchronous stochastic approximation and Q-learning. *Machine Learning*, **16**, 185–202.

von Neumann, John and Morgenstern, Oscar (1947). *Theory of Games and Economic Behavior*. Princeton University Press, Princeton, NJ.

Watkins, Christopher J. C. H. and Dayan, Peter (1992). Q-learning. *Machine Learning*, **8**, 279–292.