Decision Tree Induction with CBR

B. Radhika Selvamani and Deepak Khemani

A.I. & D.B. Lab, Dept. of Computer Science & Engineering,
I.I.T. Madras, India
bradhika@cs.iitm.ernet.in, khemani@iitm.ac.in

Abstract. This paper describes an application of CBR with decision tree induction in a manufacturing setting to analyze the cause for defects reoccurring in the domain. Abstraction of domain knowledge is made possible by integrating CBR with decision trees. The CID approach augments the recall and reuse done by CBR with statistical analysis that is focused towards the discovery of connections between specific defects and their possible causes. We show that this discovery also gives a pointer towards a corresponding corrective action.

1 Introduction

In most domains we can acquire the initial seed cases from the experts, documents or databases in the domain. However such a case base may fail to respond to a query due to: (1) Lack of adequate coverage (2) Presence of noisy cases (3) Occurrence of a novel case. The problem of case coverage may be solved over time when the experts update the case base periodically. Instance-based learning (IBL) is a carefully focused case-based learning approach that contributes evaluated algorithms for selecting good cases for classification, reducing storage requirements, tolerating noise and learning attribute relevances. Handling noisy instances has been widely explored in the IBL framework [1,2]. Occurrence of a novel case can be viewed as a gap in the case base. That is, a query case may find a similar case in the case base, but the retrieved case may be incomplete for solving the given problem. The case needs to be completed with the appropriate solution before it contributes to the case base. Traditional process of reminding as in CBR cannot address the problem at hand. The domain expert may complete an incomplete case after analysis. Other machine learning techniques like case adaptation, decision trees [7], data mining [4] etc., can be combined with CBR [3,10] to handle the problem. One such situation is that in which diagnosis of a frequently occurring fault has to be done. The fault may not have a corrective action identified, or the corrective action in current practice may not be working anymore. In the absence of a well-defined underlying theory, the first step in diagnosis would be the identification of possible parameters that could be causally linked to the fault. This task can be done by a statistical analysis of the past data, and it has been proposed as the construction of a Cause Induction in Discrimination Tree (CID Tree) [8]. We apply the technique to a real world problem from the manufacturing domain[1]. The paper is organized as follows. The earlier work relating induction with CBR has been

[1] This research work has been supported by a grant from Tube Products Investments, Carborundum Universal Ltd., Madras, India.

discussed. The domain where the algorithm has been applied and the experimental set up are explained. Finally the results obtained and the conclusions are given.

2 Domain

The domain is manufacturing steel strips. The raw material is in the form of a coil that passes through various process departments as follows. The pickling department first treats the coil with acid. The coils are then cut into strips of required length and width in the slitting department. The strips are reduced to the required thickness by drawing the steel sheets through a number of passes in a cold roll mill (CRM). Then the strips are subjected to annealing. Tests are done on the strips to check if they satisfy the required physical and mechanical properties. The parameters affecting the final product are the chemical properties of the raw coil used, the preprocessing parameters, the load and pressure parameters during drawing and the annealing parameters. The final steel sheets may have various defects like scratches, edge cracks, dent punch etc. When a defect is found, the strip is remade with modified parameters.

A CBR system was installed to capture the case history in the domain. All the cases manufactured are added to the case base. The case base evolves with each instance of manufacturing. Whenever a defect is seen, the case base is consulted for remedies. If no case is found, the defective case is added to the case base after appropriate analysis on the defect. The aim of the current work is to help the experts analyze the defects when the CBR system lacks the required information explicitly.

2.1 Case Structure

The cases are sought for a solution whenever there occurs a defect in the products. The problem space of the case is made up of the design attributes, the process parameters and the defect description. The solution space consists of possible remedies, which are suggested changes to the process parameters. The solution hence depends on the context provided by the problem space. A case is incomplete when it describes a defective with no corresponding solution. A case without a solution has no utility and hence considered a gap. The aim is to provide the experts with a set of suggestions to complete these cases and thus eliminate the gaps in the system

3 CBR with Cause Induction

CBR with cause induction algorithm CID Tree [8,9] was applied in the domain. The CIDTree method has two phases. The construction of the discrimination tree based on defect clusters constitutes the first phase and the induction process the second phase. During the first phase, the cases are labeled as good or bad based on the presence or absence of the defect to be analyzed. The labeled cases are then used in building up a discrimination tree (We used c4.5 induction algorithm for the experiments). Investigating the possible causes for the defect forms the second phase of the method. The aim of the analysis is to select a pair of nodes under the same parent, such that each has high importance with respect to the alternative classes (Good, Bad). One can observe that if there were a single identifiable cause for the defect in the case records then it would show up at the root of a three-node induction tree. The fact that in practice the induction trees are layered implies that this is not the case. In this situation, the CID algorithm inspects the tree to locate the most likely cause or causes for the given defect. The data is presumably noisy as well. The relevance score for each node P in causing the defect is calculated as follows. $Score_p=(Dp^2*Ns^2)/(Np^2*Ds^2)$

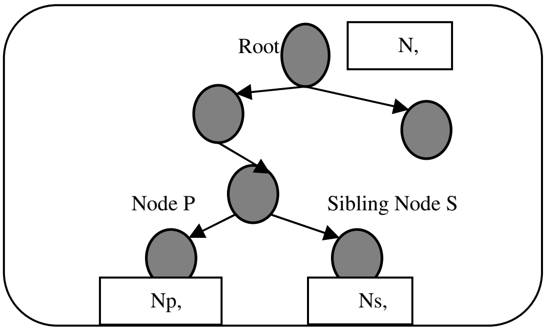

Fig. 1. Distribution of Cases at the Nodes

Np,Dp are the number of good and bad cases under the node P.

Ns,Ds are the number of good and bad cases under the sibling node S.

N, D are the numer of good and bad cases present at the root.

Note that relevance score depends only upon the number of cases under the two nodes. It is high when the node P has relatively high number of defective cases compared to S. If P has more than one sibling, their numbers are clubbed into the parameters Ns and Ds.

3.1 Relevance of the Node

A node represents a pattern observed among the instances, which is the path of the node from the root (Fig.1). A decision tree holds all patterns that provide better classification accuracy within an optimum height. The number of defective/good instances classified under a node can be used to measure the support and confidence of the pattern causing a defective/good case. Let P be a node representing a pattern, the alternative pattern formed by changing the last decision is the sibling node S. We look at the statements (P => Defect), which is to be read as node P is the cause of the defect and (P=>Good) read as P reduces defects.

Support(P => Defect) : Fraction of defects under node P over all cases = $D_p/(N+D)$. (1)
Confidence(P=>Defect) : Fraction of defects among those under the node P=$D_p/(N_p+D_p)$ (2)
Support (P => Good) : Fraction of good cases under node P over all cases = $N_n/(N+D)$. (3)
Confidence(P=>Good): Fract. of good cases among those under the node P = $N_p/(N_p+D_p)$. (4)

Maximize support and confidence (P => Defect):

$$D_p^2 / (N+D) * (N_p+D_p) \quad . \tag{1x 2}$$

Minimize support and confidence (P => Good):

$$N_p^2 / (N+D) * (N_p+D_p) \quad . \tag{3 x 4}$$

Minimize support and confidence (S => Defect):

$$D_s^2 / (N+D) * (N_s+D_s) \quad . \tag{5}$$

Maximize support and confidence (S => Good):

$$N_s^2 / (N+D) * (N_s+D_s) \quad . \qquad (6)$$

Relevance of Node P in causing the defect

$$D_p^2 N_s^2 / D_s^2 N_p^2 \ . \qquad (1x2 \text{ x } 6 / 3x4 \text{ x } 5)$$

Score(P=>Defect): ($D_p^2 N_s^2$ + residue) / ($D_s^2 N_p^2$+residue)

Score(P=>Good) : 1/Score

We have avoided division by zero by adding a residue to the formula.

Earlier work to convert decision trees to rules focused on extracting more comprehensive structures from the tree preserving the accuracy of classification [7]. We are extracting the patterns those are more relevant with respect to the defect used to label the instances. In our earlier work [8,9] we had tried out the algorithm with defects in a refractory blocks manufacturing setting [5,6]. The induction tree classifying the defects in the case base was instrumental in the diagnosis process. The correlations suggested by the trees were in agreement with the experts' intuitions in creating trials during failures. These are rules, which extracted the domain knowledge from the collected cases. The experiments, done on the steel strips manufacturing domain, are described in the following section.

4 Experiments

Data was obtained for 130 parameters from the domain. We have 1168 records from the domain and have done the analysis for three different types of defects namely scratch, edge cracks and dent punch. There were 231 records with scratches, 16 records with edge cracks and 51 with dent punch. The attributes Si (silicon) and others1 (other chemicals) are chemical characteristics of the raw material, the attributes A, B, C are the different vendors (names changed) of raw materials, CRMPasses is the number of times the sheet is passed through the CRM mills to reduce the thickness. The cases were labeled as D-defective case and N-Good case, based on the occurrence of the particular defect in the case to be analyzed.

4.1 CBR C4.5 and CID

The primary goal of the CBR implementation is to suggest process changes when there occurs a defect. When the CBR has cases, which do not provide any solution there is a gap in the system. The system retrieves cases based on the similarity of the problem, but lacks information to solve the problem. The case completion requires acquiring a solution for the problem. Note that all cases have defect description as part of the problem space, which is used to label the cases for induction There is enough information available for building the decision tree though cases are incomplete in the case base. The decision tree obtained using C4.5 is in Fig 2. Now to obtain a solution for a particular defect, we obtain the prominent patterns from the c4.5 using CID scoring. The results are shown in the Table1. This information when added to the cases, which have the particular defect, fills the gap in the system.

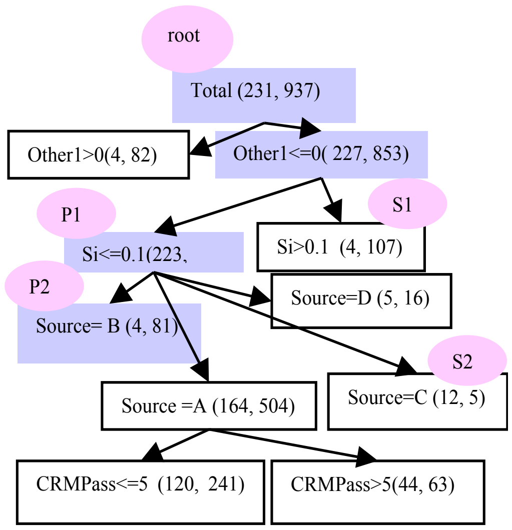

Fig. 2. Analysis of scratch in strips manufacturing

Table 1. The Scores Obtained by CID

Attribute	D	N	Scor	1/Sc
Others	231.3	936.7	0	0
Other1<=0	227.4	853.6	33.6	0
Other1 > 0	3.8	83.2	0	33.6
Si<= 0.1	223.4	746.3	62.0	0
Si > 0.1	4.1	107.3	0	62.0
Source =A	164.3	504.8	1.8	0.6
Source = B	3.8	81.2	0	49.1
Source = C	11.4	4.6	73.5	0
Source =D	4.8	15.8	1.1	1
CRMPass<=5	120.5	242.0	8.9	0.1
CRMPass>5	43.8	262.7	0.1	8.9

4.2 Results

The objective is to learn the causes for certain defects from the cases collected in the case base to complete the cases without solution. The incomplete cases may be filled up with their respective solutions based on the results obtained from the algorithm.
Scratches

- If the raw material for the sheet had silicon < 0.1 % then there were more scratches. The experts agreed upon this. (Shown as P1 and S1 in Fig. 2).
- The algorithm also identified that raw materials purchased from sourceB had less scratches and that from C had more (shown as P2 and S2 in Fig. 2).

Cracked Edges

- Similarly %carbon being less was found to be the cause for cracked edges.

Dent Punch

- Analysis on dent punch showed that pass2thickness being less (mechanical property) caused the defect. But dentpunch is known to occurr due to use of defective raw material (chemical property). The nonconforming results were attributed to the scratch data being misclassified as dentpunch.

In the above experiment the experts could suggest a change in source raw material for better results in cases with scratches. Better classification of the defects scratch and dentpunch may avoid erroneous cases.

5 Conclusions

Case Based Reasoning is built on the foundation of accumulating experience and re-using it to solve problems. Traditionally this reuse is direct, in the sense that the most similar case or a set of k most similar cases are retrieved and used directly. But combined with other machine learning techniques a case base can yield more information by looking at some level of abstraction. Machine learning has been used in case acquisition and in indexing [11]. In this paper we demonstrate that a decision tree learnt to separate defective cases from non-defective cases can be used to identify attributes whose value choices could be the causes of defects. The few experiments we have done with limited data corroborate the findings. As more data accrues, we hope that in future we will develop techniques that would help shop floor personnel choose effective corrective actions when faced with previously unsolved defect problems. Closing the feedback loop in CBR [12] is very important in managing the flow in a domain using CBR. Our future work is towards generating trial cases when the case base lacks enough domain expertise. After the remedy in the trial case is tested on the shop floor, the case can be stored permanently in the case base.

References

1. Aha, D.W., Kibler, D., Albert, M. K. Instance Based Learning Algorithms, Machine Learning, 6:37-66, 1991.
2. Cost S. Salzberg. A Weighted Nearest Neighbor Algorithm for Learning with Symbolic Features, Machine Learning, 10:57-78, 1993.
3. Janet Kolodner. *Case-Based Reasoning*, Morgan Kaufmann Publishers, 1993.
4. Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques,* Morgan Kaufmann Publishers, 2001.
5. Khemani, D., Radhika Selvamani, B., Ananda Rabi Dhar, Michael S.M. InfoFrax: CBR in Fused Cast Refractory Manufacture. *In proceedings on the European Conference on Case Based reasoning*, Lecture Notes in Computer Science 2416 Springer, 560 – 574, 2002.
6. Michael, S. M., and Khemani D. Knowledge Management in Manufacturing Technology. *In Proceedings of the International Conference on Enterprise Information System*, Ciudad Real, Spain, Vol. I, 506-512, 2002.
7. Quinlan, J.R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann Publishers, 1993.
8. Radhika Selvamani, B., Deepak Khemani. Investigating Cause for Failures in Discrimination Tree from Multiple Views. In the proceedings of the *International Conference of Knowledge Based Computer Systems*, Mumbai December 11-14, 2002.
9. Radhika Selvamani, B., Deepak Khemani. Managing Experience for Process Improvement in Manufacturing. In the proceedings of the *International Conference of Case Based Reasoning ICCBR-2003.* Trondheim Norway, June 23-26, 2003.
10. Watson, I., Applying Case Based Reasoning: Techniques for Enterprise Systems, Morgan Kaufmann Publishers, 1997.
11. Patterson, D., Anand, S S., Dubitzky, W., Hughes, J G. : Towards Automated Case Knowledge Discovery in the M2 Case-Based Reasoning System. In Knowledge and Information Systems: An International Journal. Springer Verlag, Vol. 1 61-82, 1999.
12. Price, C.J., Pegler, I.S. Ratcliffe, M.B. and McManus, A. From troubleshooting to process design: closing the manufacturing loop. In proc. *2nd International Conference, ICCBR-97,* Providence, Rhode Island, USA, Lecture Notes in Computer Science 1266 Springer, 1997.