Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

On Learning Compressed Diagnosis Classifiers [⋆]

Gregory Provan [*]

[*] *Department of Computer Science, University College Cork, Cork, Ireland (Tel: 353-21-490-1816; e-mail: g.provan@ cs.ucc.ie).*

Abstract:
We address the problem of embedding a model-based diagnostic system representation within a processor with limited memory (as is typical of most real-world aerospace systems). Given a Boolean diagnostic model f in which we have a probability distribution over fault likelihoods, we describe a method for approximately generating an embedded representation of f by learning a decision tree that encodes only the probabilistically most-likely diagnoses. If the set of possible diagnoses follows a power-law distribution, we show that we can create decision trees that contain the vast majority of the probability mass of the full decision tree, but require significantly less memory than the full decision tree.

1. INTRODUCTION

One of the main issues in on-board diagnosis is the development of diagnostic models. Rule-based systems (Rowland and Jain [1993]) are compact and efficient to evaluate, but their coverage and accuracy do not have any associated guarantees. Model-based systems (Isermann [2005], Reiter [1987]) provide coverage [1] and accuracy guarantees, but typically the associated models require significant space and are complex to evaluate.

A range of other methods have been proposed, such as automatic test-pattern generation (Sheppard and Simpson [1994]), fault trees (Rowland and Jain [1993]), model compilation (Bryant [1986], Darwiche [1998], de Kleer [1986]), and fault detection and isolation (FDI) (Isermann [2005]). All of these methodologies face similarly high time- or space-complexity for isolating multiple simultaneous faults. For example, isolating multiple simultaneous faults is NP-complete for a propositional Horn system (Bylander et al. [1991]); further, the complexity of analogous diagnostic tasks, like single-fault test-set generation, is also NP-complete (Ibarra and Sahni [1975]), and the size of the test set can become prohibitive for large circuits (Sheppard and Simpson [1994]).

One alternative is to induce diagnosis classifiers. However, building diagnosis classifiers for large systems may impose large space requirements: if we want to classify all multiple-fault events in an n-component Boolean system, the classifier must have 2^n leaves.

To address the high time-complexity of computing diagnoses on-line, or the space complexity of storing all system faults (or an analogous set of tests), in this article we study building *compressed* classifiers, i.e., classifiers with size smaller than that of a standard classifier. In particular, we generate an approximate classifier that captures the most-likely diagnoses, and ignores diagnoses that are extremely unlikely. We show that significant compression is possible using this approach, given the properties of typical diagnostic systems.

To generate the set L of learning instances, we simulate from f the probabilistically most-likely diagnoses and associated sensor readings. We then induce from L a decision tree (Quinlan [1986]). Decision trees are popular representations of Boolean functions, and form the basic inference engine in well-known machine learning programs such as C4.5 (Quinlan [1986, 1993]). Decision trees are an important representation for Boolean functions because (a) they can represent all Boolean functions, and (b) many useful operations on Boolean functions can be performed efficiently in time polynomial in the size of the decision tree representation.

Our contributions are as follows:

- We show that, given a probability distribution on prior component failure likelihood, the posterior diagnosis distribution follows a power-law, and hence the majority of the probability mass is captured by a relatively small number of diagnoses.
- We describe an algorithm for inducing a decision tree diagnostic classifier that contains the majority of the probability mass of the full decision tree, but requires significantly less memory than the full decision tree.

2. RELATED WORK

This work makes contributions to model-based diagnosis, test generation, and machine learning. In the area of model-based diagnosis, we extend the state-of-the-art by showing how we can sub-sample from a model to generate an approximate model, which is represented as an embeddable diagnostic classifier. This approach is different to an approximate version of a compiled model, e.g., (Bryant [1986], Darwiche [1998]), in that the combination of stochastic sampling and decision tree representation

[⋆] This work was supported by SFI grant 04/IN3/I524.

[1] Coverage refers to the percentage of all diagnostic queries that can be correctly isolated.

enable our approach to model a wider range of systems than can BDDs and DNNFs.

In the area of test generation, the primary difference is that, rather than generate a set of tests, we integrate the tests into a decision tree, which can be efficiently evaluated within an embedded setting.

In the area of machine learning, our approach is novel in that it introduces a novel approach to decision tree induction, where we generate an approximate (compressed) classifier by sub-sampling from a known model. Most approaches assume that we start with data, rather then a model, the generation of data from which we can control.

3. PROBLEM FRAMEWORK

This section introduces the problem that we are solving, as well as the notation necessary for defining this problem.

3.1 Diagnosis Model

We now introduce the notion of system model and diagnosis that we use to specify a discrete-valued model-based diagnosis (MBD) representation (Reiter [1987]). The MBD models we use are applicable to arbitrary discrete-valued systems, but in the following we assume that we are defining the MBD problem for circuit models. In future work we plan to define models for more general systems.

Central to MBD, a *model* of an artifact is represented as a Boolean propositional function f over a set X of variables. Distinguishing two subsets of these variables as *health* (assumable) and *observable* [2] variables gives us a diagnostic system.

Definition 1. (Diagnostic System). A diagnostic system Ψ is defined as the triple $\Psi = \langle f, \mathcal{H}, \mathcal{O} \rangle$, where f is a propositional theory over a set X of variables, $\mathcal{H} \subseteq X$ is the set of assumables, and $\mathcal{O} \subseteq X$ is the set of observables.

Throughout this paper we will assume that $\mathcal{O} \cap \mathcal{H} = \emptyset$ and $f \not\models \perp$.

Given a system theory f and an observation α, a diagnostic query identifies if α is anomalous with respect to f, in which case we must isolate the diagnoses. MBD computes diagnoses in terms of assignments to the assumable variables $\mathcal{H}$. By convention, $h_i = 0$ denotes normal functionality, and $h_i = 1$ denotes a fault.

Definition 2. (Health Assignment). Given a diagnostic system $\Psi = \langle f, \mathcal{H}, \mathcal{O} \rangle$, an assignment $\lambda_{\mathcal{H}}$ to all variables in $\mathcal{H}$ is defined as a health assignment.

Analogous to a health assignment, a *complete assignment* λ_X is an instantiation to all the variables in X. The MBD notion of diagnosis covers multiple simultaneous faults, as denoted below.

Definition 3. (Diagnosis). Given a diagnostic system $\Psi = \langle f, \mathcal{H}, \mathcal{O} \rangle$, and an observation α over some variables in $\mathcal{O}$, a health assignment is a diagnosis, ω, if $f \wedge \alpha \wedge \omega \not\models \perp$.

[2] In the MBD literature the assumable variables are also referred to as "component", "failure-mode", or "health" variables. Observable variables are also called "measurable", or "control" variables.

The cardinality of a diagnosis ω is the number of faulty components in ω. We denote a diagnosis of cardinality k using ϑ_k, and the set of k-cardinality diagnoses using $\boldsymbol{\vartheta_k}$. The space of all diagnoses is given as follows:

Definition 4. (Diagnosis Space). Given a diagnostic system $\Psi = \langle f, \mathcal{H}, \mathcal{O} \rangle$, a diagnosis space Ω is the set of all instantiations of the health variables $\mathcal{H}$ consistent with f.

3.2 Objectives

Our objective is to generate a space-efficient, or compressed, diagnosis classifier Δ. A complete classifier, $\Delta(\Omega, \mathcal{O})$, stores a representation of the complete diagnosis space Ω, such that for any observation α of observables $\mathcal{O}$, the "optimal" diagnosis can be returned. By selecting a subset of diagnoses, $\Omega' \subset \Omega$, and inducing a classifier from learning instances generated by sampling from Ω', we can create a compressed classifier $\Delta'(\Omega', \mathcal{O})$ which will be more memory-efficient than $\Delta(\Omega, \mathcal{O})$.

In particular, our objective is to generate a classifier Δ such that (a) $\Delta \cup \alpha \models \overline{\omega}$, (b) this new inference process is more efficient than using Δ, and (c) Δ' is much smaller than Δ, $|\Delta'| \ll |\Delta|$.

An alternative method for creating a compressed classifier is by sub-sampling from $\Delta(\Omega, \mathcal{O})$, using an approach like importance sampling (Bucklew [2004]). This approach may lead to a more accurate classifier than the threshold-based pruning method, but it will also lead to a relatively larger classifier Δ, since rare event still must be captured by Δ.

The key issue will be the loss of accuracy of the resulting compressed classifier Δ'. In the threshold-based pruning method, the error arises from three sources: (1) diagnosis-space pruning; (2) sampling; and (3) classifier induction. Pruning is the largest source of error, given appropriate sampling and induction techniques. We will show that if we use a domain in which the diagnosis space follows a power-law distribution, then the pruning error is also small.

3.3 Example

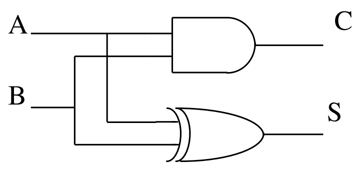
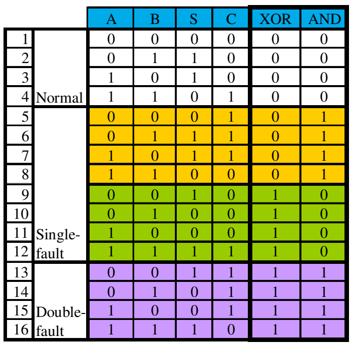


Fig. 1. Diagram of half-adder circuit.

Figure 1 shows an example of a simple circuit, a half-adder. This circuit has two inputs, $I = \{A, B\}$, two outputs, $O = \{S, C\}$, and two components, an XOR and an AND gate. The standard boolean function for this circuit is defined over variables (A, B, S, C).

We define fault-modes for the XOR and AND gates, with boolean domains $\{OK, failed\}$, together with clauses specifying the circuit behaviour given both *OK* and *failed* component modes.

Figure 2 shows the truth table for the half-adder fault model. The first four rows of the table show the normal instances, i.e., the truth table for a standard (non-fault)



		A	B	S	C	XOR	AND
1		0	0	0	0	0	0
2		0	1	1	0	0	0
3		1	0	1	0	0	0
4	Normal	1	1	0	1	0	0
5		0	0	0	1	0	1
6		0	1	1	1	0	1
7		1	0	1	1	0	1
8		1	1	0	0	0	1
9		0	0	1	0	1	0
10		0	1	0	0	1	0
11	Single-	1	0	0	0	1	0
12	fault	1	1	1	1	1	0
13		0	0	1	1	1	1
14		0	1	0	1	1	1
15	Double-	1	0	0	1	1	1
16	fault	1	1	1	0	1	1

Fig. 2. Truth table of half-adder fault model.

model. The next four rows show the instances for AND-faults, the following four for XOR-faults, and the final four rows show the instances when we have both AND- and XOR-faults simultaneously (double faults).

4. ANALYSIS OF DIAGNOSIS SPACE

This section analyses the distribution of the diagnosis space, under standard assumptions of high-reliability complex systems. In particular, we show that the diagnosis CDF is power-law distributed. As a consequence, we can take advantage of the shape of this distribution to generate compressed classifiers that are space-efficient while losing little fault coverage.

4.1 Approximate Diagnosis Distributions

We now specify the probability distributions over the diagnosis space. We make the following assumptions:[3]

- All components fail independently of each other.
- Component c_i has prior failure probability p_i.
- For every component $c_i, i = 1, .., f$, $p_i \ll (1 - p_i)$.

Let $\pi = \{p_1, ..., p_f\}$ be the set of prior fault probabilities for the components. Given our assumptions above, the probability for any diagnosis is given by

$$Pr(\omega) = \prod_{i:c_i=failed} p_i \prod_{i:c_i=OK} (1 - p_i).$$

Our diagnosis distribution is thus given by $Pr(\Omega) = \{Pr(\omega)|\omega \in \Omega\}$.

We introduce a threshold ϕ such that we prune all diagnoses such that $Pr(\omega_i|\omega_i \in \Omega) < \phi$. If we prune a subset of the space of diagnoses given by $\Omega_{<\phi}$, then the pruning error of a threshold-based (compressed) classifier is given by $Pr(\Omega_{<\phi})/Pr(\Omega)$.

We can define a probability density function (PDF) for k-cardinality diagnoses as follows:

$$f(\boldsymbol{\vartheta_i}) = \frac{Pr(\boldsymbol{\vartheta_i})}{Pr(\Omega)}.$$

Further, we can define the cumulative density function (CDF) as follows:

$$F(\boldsymbol{\vartheta_i} \leq k) = \sum_{i \leq k} f(\boldsymbol{\vartheta_i}) = \sum_{i \leq k} \frac{Pr(\boldsymbol{\vartheta_k})}{Pr(\Omega)}.$$

If we define the subset of k-cardinality diagnoses which are above a threshold ϕ using $\boldsymbol{\vartheta_k^\phi}$, then we can obtain threshold-based PDFs and CDFs simply by substituting $\boldsymbol{\vartheta_k^\phi}$ for $\boldsymbol{\vartheta_k}$ in the previous equations.

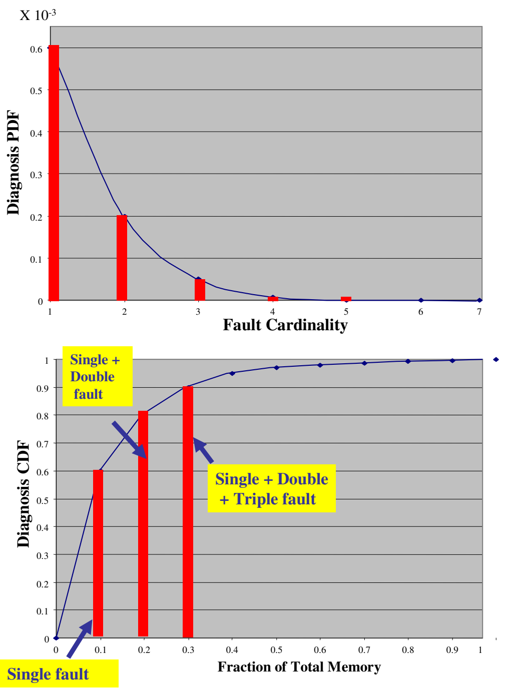


Fig. 3. PDF and CDF of diagnosis distribution.

4.2 Power-Law Diagnosis Space Distributions

Power-law distributions have been observed in many natural systems, such as physical (biological or man-made) and sociological systems (Newman [2003]).

Definition 5. (Power Law). A probability density function (PDF) for a random variable X follows a power-law if $Pr(X > x) \sim x^{-\beta}$, as $x \to \infty$, $0 < \beta < 2$.

Two of the most important power-law distributions are the Zipf and Pareto distribution.

Definition 6. (Zipf distribution). A discrete random variable Z follows a Zipf distribution if the r^{th} largest value has probability density function (PDF) $p(r; \beta) = Kr^{-\beta}$, for $\beta, r > 0$, and constant K.

The simplest continuous power-law distribution is the Pareto distribution, which has PDF and cumulative distribution function (CDF), respectively:

Definition 7. (Pareto distribution).

$$p(x; \alpha, k) = \alpha k^\alpha x^{-\alpha-1}, \text{ for } \alpha, k > 0, \; x \geq k.$$
$$F(x; \alpha, \eta) = Pr[X \leq x] = 1 - \left(\frac{\eta}{x}\right)^\alpha,$$

where η represents the smallest value the random variable X can take.

[3] In the following we assume that a component can be either *OK* or *failed*, but can generalise this the multiple possible normal and failure states in a straightforward fashion.

The closely related exponential distribution has also been used for modeling many natural systems (Newman [2005]).

Example 1. Figure 3 shows a PDF and CDF of a typical diagnosis problem, such as that described in Section 3.3. Using a probabilistic preference function with $Pr(z = OK) = 0.9$, the PDF and CDF are clearly Pareto-distributed. In the CDF the x-axis denotes the fraction of total memory required by a compilation containing diagnoses up to cardinality k, i.e., containing up to k broken components.

Consider a probabilistic ordering over the diagnosis space. Given a skewed failure-mode distribution π, we assume that $\exists\beta$ such that every component h_i has $Pr(h_i = OK) \geq \beta Pr(h_i = \mathit{failed})$. Such a skewed distribution is normal in diagnostics, and implies that normal behavior is β times more likely than faulty behavior, $\beta \geq 1$. By our skewness assumption, we know that $Pr(\vartheta_i) \geq \beta Pr(\vartheta_{i+1})$.

Theorem 1. If we assume that $\beta Pr(h_i = \mathit{failed}) = Pr(h_i = OK)$ for all failure modes h_i, then we obtain a heavy-tailed (Zipf) distribution for the diagnosis distribution.

Proof Sketch: We assume that $\beta Pr(h_i = \mathit{failed}) = Pr(h_i = OK)$, $\forall h_i$. We further assume that each diagnosis consists of n failure-mode variables, for which $Pr(h_i = OK) = \eta_i$, and hence $Pr(h_i = \mathit{failed}) = \eta_i/\beta$. Hence, the probability of a k-fault diagnosis ϑ_k is given by $Pr(\vartheta_k) = \beta^{-k} \prod_{i=1}^{n} \eta_i$. According to the Zipf distribution, if we take the ratio of the r^{th} and $(r+1)^{th}$ ranked variables, we should obtain

$$\frac{p(r)}{p(r+1)} = \frac{Kr^{-\gamma}}{Kr^{-(\gamma+1)}} = r.$$

Now, if we take the ratio of the diagnoses with r- and $(r+1)$-faults, we obtain

$$\frac{Pr(\vartheta_r)}{Pr(\vartheta_{r+1})} = \frac{\beta^{-r}\prod_{i=1}^{n}\eta_i}{\beta^{-(r+1)}\prod_{i=1}^{n}\eta_i} = \beta.$$

This rank distribution over the diagnoses is distributed according to Zipf's law, with each ϑ_{k+1} being β times less likely than ϑ_k. $\Box$

The significance of a diagnosis distribution being power-law is that the majority of the probability mass is concentrated in the low-cardinality diagnoses, and this characteristic increases as the prior fault probabilities decrease, i.e., the higher the reliability of a system (the more skewed the distribution), the more the probability mass is concentrated in the low-cardinality diagnoses. It has been shown that only $n+1$ assignments can cover the vast majority of the probability mass of models for skewed distributions (D'Ambrosio [1993]).

Theorem 2. (Skewness-Based Sampling Size). : Given a joint distribution over n binary-valued variables such that all distributions are skewed with a larger mass of at least $(n+1)/n$, then the largest $(n+1)$ terms in the joint distribution across the variables contain a total mass of greater than $2/e$.

This result indicates that our threshold-based sampling approach will incur low error due to pruning, as long as we retain at least the probabilistically-largest $O(n)$ diagnoses.

5. COMPRESSED DECISION-TREE CLASSIFIERS

This section describes our threshold-based pruning method for decision tree induction.

5.1 Decision Trees

A decision tree T is a binary tree where the leaves are labeled either 0 or 1, and each internal node is labeled with a variable. Given an assignment $\lambda_X \in \{0,1\}^n$, $T(\alpha)$ is evaluated by starting at the root and iteratively applying the following rule, until a leaf is reached: let the variable at the current node be x_i; if the value of at position i is 1 then branch right; otherwise branch left. If the leaf reached is labeled 0 (resp. 1) then $T(\lambda_X)$= 0 (resp. 1). The size of a decision tree is its number of nodes.

We can measure the "quality" of a decision tree using two parameters, one for accuracy and one for complexity (or size). We can measure the complexity of a decision tree T simply in terms of the size of the tree $|T|$. For this work we ignore the structure of the tree.

5.2 Approximation via Decision Trees

The optimal version of the decision tree minimisation problem is: given a boolean function $f : \{0,1\}^n \rightarrow \{0,1\}$, and an assignment of $[0,1]$ weights to the variables of f, can we replace f with an *optimally* smaller boolean function f' representing the set $\{s \in \{0,1\}^n : v(f(s)) \geq \phi\}$, where $\phi \in [0,1]$ is a lower bound for diagnoses, and f' is defined as a decision tree.

We know that this problem is hard, since the specific version of it in which $\phi = 0$. i.e, finding an equivalent representation of f as a decision tree of minimal size, is hard (Zantema and Bodlaender [2000]). Polynomial-time solvability of this problem would imply that satisfiability of CNF formulas can be decided in polynomial time, which is impossible unless P=NP. Further, approximating this problem within a fixed distance δ from optimal is also hard (Sieling [2003]).

As a consequence, we solve a simpler approximation problem, which makes no claims of optimality of the generated Decision Tree T. We measure the "quality" of our approximation using the diagnosis space CDF, which plots the cumulative probability mass (which specifies the coverage of the space of possible weighted diagnoses) versus the relative memory (which specifies the memory ratio of a subset of the diagnosis space).

We solve the problem of computing a decision tree T from a boolean function f such that, for small δ, T has coverage of $1-\delta$ with memory ratio $< 1-\delta$.

5.3 Machine Learning Approach

We propose an approach that uses three main steps to generate an approximate decision tree:

(1) Health assignment sampling (to create health assignment set Λ_ϕ);
(2) Generating *consistent* complete assignments from Λ_ϕ to form the training set L;

(3) Inducing a decision tree from the training set L

We now describe each of these steps.

Generation of Partial Assignments through Sampling We enumerate Λ_ϕ, those health assignments with probability over a threshold ϕ, and then compute complete assignments of each member of Λ_ϕ. In essence, we generate a health assignment lattice in breadth-first fashion, using the assignment probability to guide the search.

We start with the assignment with the highest probability, λ_{max}, i.e., the assignment where each variable instantiation has its higher probability. We then successively create sets of assignments Λ^i with Hamming distance $i = 1, 2, 3, ...$ from λ_{max}, such that $Pr(\lambda') \geq \phi$ for every $Pr(\lambda') \in \Lambda^i$. We denote the j^{th} element of Λ using Λ_j.

We assume a *flip* operation, where we flip the variable assignment that decreases $Pr(\lambda)$ the least to create λ'; if $Pr(\lambda') \geq \phi$, we add this to the set Q of thresholded assignments.We continue this process recursively until no new assignments can be added to Q.

```
Proc Assignment-Gen $(f, \pi)$
$\Lambda^0 \leftarrow \lambda_{max}$; $Q \leftarrow \Lambda^0$;
Do $i = 1$ to $n$;
    Do $j = 1$ to $|\Lambda^i|$;
        Do until there are no unflipped variables;
            $\lambda' = flip(\Lambda^{i-1}_j)$;
            If $Pr(\lambda') > \phi$, then $\Lambda^i \leftarrow \Lambda^i \cup \lambda'$;
    End;
    $Q \leftarrow Q \cup \Lambda^i$;
End;
Return $Q$;
```

It is easy to show that the queue Q that we output from this algorithm will contain a ranked list of all those assignments that have probability above ϕ. Given Q, we then analyse each element for satisfiability, and return just the subset Q' of *consistent* assignments of Q, which consists of the models (complete assignments) of f such that, for every model λ_X, $Pr(\lambda_X) > \phi$, $\forall \lambda_X \in Q'$.

The complexity of this algorithm is specified as follows. In the worst case, if we set $\phi = 0$, then we have to enumerate all 2^n assignments and test their satisfiability. However, if $\phi > 0$, then we must enumerate only $O(n)$ assignments to ensure that the pruning error is less than $1 - 2/e$ (Theorem 2).

When you relax the degree of skewness to ensure that the larger mass p is at least *loglogn* times the smaller mass, i.e., $p \geq loglogn(1 - p)$, then only $O(n)$ assignments can cover the vast majority of the probability mass of models. We make this latter assumption.

Procedure *Assignment-Gen* helps to identify the positive instances for learning a decision tree. However, we also need to include negative instances, Hence, the three types of assignments from f that are necessary for learning cover:

- satisfying assignments with $Pr(\lambda) \geq \phi$: $\mathcal{S}^{\geq\phi}$
- satisfying assignments with $Pr(\lambda) < \phi$: $\mathcal{S}^{<\phi}$
- unsatisfying assignments: $\bar{\mathcal{S}}$

Since there are 2^n possible assignments, we generate only those assignments above or equal to the threshold ϕ, and then sample from the remaining assignments. In the worst-case this is also $O(2^n)$, but we choose such that $|\mathcal{S}^{\geq\phi}|$ is $O(n)$.

We categorise two sets of instances that can be generated from f: (1) positive: $I = \mathcal{S}^{\geq\phi}$; and (2) negative: $\bar{I} = \mathcal{S}^{<\phi} \cup \bar{\mathcal{S}}$. We will generate all assignments in I, and then choose a distribution to sample a set Q of instances from $\bar{I}$ such that, for some $0 < \varpi \leq 1$, $\frac{v(Q)}{v(\mathcal{S})} \geq \varpi$. In other words, we sample until we generate a set Q of assignments such that $v(Q) \geq \varpi v(\mathcal{S})$.

5.4 Example: Inducing a Compressed Decision Tree Diagnosis Classifier

In the half-adder example, given a diagnosis ω, an instance consists of an assignment to $\{A, B, S, C, P, \omega\}$, such as $\{1, 1, 1, 1, (XOR = \textit{failed})\}$, indicating that the XOR gate is faulty for this input-output setting.

Inducing a complete decision tree entails using all 16 consistent instantiations shown in Figure 2. In the following, we show how we sub-sample from these 16 instantiations to generate an approximate tree.

Step 1: Generating Threshold-Based Health Assignments

The step enumerates the health assignments whose probability exceeds ϕ.

		XOR	AND	Pr(l 1)	Pr(l 2)	Pr(l 1)> .001	Pr(l 1)> .05	Pr(l 2) > .3	Pr(l 2)> .05
1		0	0			🟨	🟨	🟦	🟦
2		0	0			🟨	🟨	🟦	🟦
3		0	0			🟨	🟨	🟦	🟦
4	Normal	0	0	0.93	0.54	🟨	🟨	🟦	🟦
5		0	1			🟨	🟨		🟦
6		0	1			🟨	🟨		🟦
7		0	1			🟨	🟨		🟦
8		0	1	0.05	0.06	🟨	🟨	X	🟦
9		1	0			🟨	🟨	🟦	🟦
10		1	0			🟨	🟨	🟦	🟦
11	Single-	1	0			🟨	🟨	🟦	🟦
12	fault	1	0	0.02	0.36	🟨	🟨	🟦	🟦
13		1	1			🟨			
14		1	1			🟨			
15	Double-	1	1			🟨			
16	fault	1	1	0	0.04	🟨	X	X	X

Fig. 4. Truth table of half-adder fault model, with probabilities assigned to each line in the truth table.

Figure 4 shows an example of health assignment probabilities for two distributions. Distribution π_1 is such that Pr(AND-mode)= (0.98, 0.02), and Pr(XOR-mode)=(0.95,0.05); distribution π_2 is such that Pr(AND-mode) = (0.6, 0.4), and Pr(XOR-mode) = (0.9, 0.1). Hence, under π_2, the assignment to the normal mode is Pr(diagnosis=OK)= $0.6 \times 0.9 = 0.54$.

If we have a threshold $\phi = 0.001$, then the column labeled $\pi_1(\lambda) > 0.001$ shows that the normal, single-fault and double-fault mode-instantiations all exceed this threshold. In this case, we would generate full instantiations from all these cases. If we set $\phi = 0.05$, then the double-fault instantiations do not exceed this threshold, and we exclude these cases from the next step of generating full instantiations for positive learning instances.

Step 2: **Generating Complete Instantiations**

For every health assignment exceeding the threshold ϕ, we compute the assignments for all other variables $x \in X \setminus \mathcal{H}$. These labeled instances thus make up the full training set.

In our half-adder example, Figure 4 indicates the training instances for four possible combinations of probability and threshold: $(\pi_1, .001)$, $(\pi_1, .05)$, $(\pi_2, 0.3)$, $(\pi_2, 0.05)$. Of the 16 possible instances (see column 1), the $\times$ indicates the subset of these instance that are ruled out by a (π, ϕ) setting.

The complexity of this step is bounded by the number of input-output combinations that are physically plausible. In the worst case, given a system with k inputs and outputs, the system has the capability to be set to 2^k distinguishable states; however, in reality, physical and practical constraints will limit the state set to a small fraction of 2^k. In this example, we have $2^4 = 16$ possible engine states, but only 2 of those states are physically realisable. Almost all real-world systems operate similarly.

Consonant with this notion of physical plausibility, we assume that a subset of the space if the number of input-output combinations $\mathcal{N}$ is physically plausible, and hence restrict the size of $\mathcal{N} = O(k^2)$. This restriction of the space of possible input state assignments significantly reduces the number of instantiations that are tested for consistency. We also sample from the other assignments to obtain the set Q of instances to complete the training set.

Step 3: **Generating the Decision Tree**

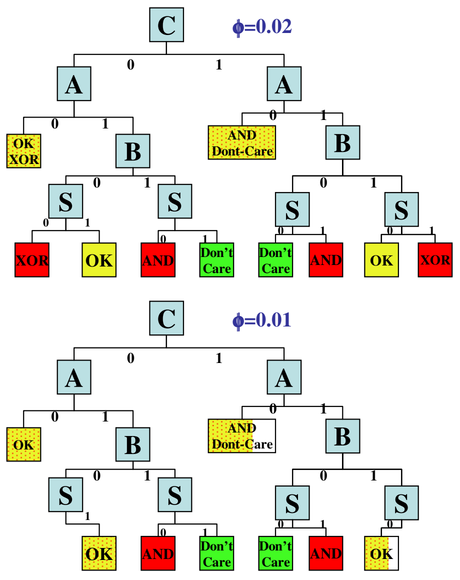


Fig. 5. Decision trees for half-adder fault model, with probabilities π_1 and thresholds $\phi = .002, 0.001$.

Figure 5 shows decision trees generated for the half-adder fault model, with probabilities π_1 and thresholds $\phi = .002, 0.001$. Note that reducing ϕ from 0.002 to 0.001 ends up in removing the node for diagnosing *XOR*. If we introduce the tree-induction option for avoiding overfitting, the tree will shrink further, from 19 nodes for the case of $\phi = .002$ to 11 nodes, with a corresponding reduction in the ability to isolate faults.

6. SUMMARY AND CONCLUSIONS

This article has proposed an approach for generating compressed diagnostic classifiers which trades off memory for diagnostic coverage. We have proven that, if we have a probability distribution π over a set of n component failure-modes, the resulting distribution over system diagnoses, $Pr(\Omega)$, is power-law distributed. If the distribution π is skewed, then we have shown that the probabilistically-largest $O(n)$ diagnoses, $\Omega_{>\phi}$, contribute the vast majority of the probability mass within a power-law distribution. Consequently, building a classifier T based on sampling from $\Omega_{>\phi}$ to generate the positive learning instances results in T having high diagnostic coverage but significantly smaller memory than a complete classifier.

REFERENCES

R. E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, c-35(8):677–691, August 1986.

J.A. Bucklew. *Introduction to Rare Event Simulation*. Springer, 2004.

T. Bylander, D. Allemang, M.C. Tanner, and J. Josephson. The computational complexity of abduction. *Artificial Intelligence*, 49:25–60, 1991.

Bruce D'Ambrosio. Incremental probabilistic inference. In *Proc. Conf. on Uncertainty in AI*, 1993.

Adnan Darwiche. Model-based diagnosis using structured system descriptions. *J. Artificial Intelligence Research*, 8:165–222, 1998.

J. de Kleer. An assumption-based TMS. *Artif. Intell.*, 28 (2):127–162, 1986.

O.H. Ibarra and S.K. Sahni. Polynomially Complete Fault Detection Problems. *IEEE Trans. on Computers*, C-24 (3):242–249, 1975.

R. Isermann. Model-based fault-detection and diagnosis–status and applications. *Annual Reviews in Control*, 29 (1):71–85, 2005.

M. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167– 256, 2003.

M E J Newman. Power laws, pareto distributions and zipf's law. *Contemporary Physics*, 46:323, 2005.

J. Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

J. Ross Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

R. Reiter. A Theory of Diagnosis from First Principles. *Artificial Intelligence*, 32:57–96, 1987.

JG Rowland and LC Jain. Knowledge based systems for instrumentation diagnosis, system configuration and circuit and system design. *Engineering Applications of Artificial Intelligence*, 6(5):437–46, 1993.

J.W. Sheppard and W.R. Simpson. *System Test and Diagnosis*. Springer, 1994.

Detlef Sieling. Minimization of decision trees is hard to approximate. In *IEEE Conf. on Computational Complexity*, pages 84–92, 2003.

Hans Zantema and Hans L. Bodlaender. Finding small equivalent decision trees is hard. *Intl. J. of Foundations of Computer Science*, 11(2):343–354, 2000.