HOW NOT

Rahul Krishna, Tim Menzies
Computer Science, North Carolina State University, USA
{i.m.ralk, tim.menzies}gmail.com

***Abstract*—**
***Index Terms*—Defect prediction, configuration, prediction, planning, case-based reasoning.**

I. Introduction

II. Contrast Trees

Contrast trees are decision trees that can be used to learn decision rules from a large data set. The use of decision trees for this purpose makes it easy to visualize the data and also makes interpretation of the recommended policies fairly straight forward. In addition to this, contrast trees generate ranges of decisions that allow for better feasibility.

A. How does a contrast tree work?

Contrast trees works by building a simple decision tree based on entropy, as in the classification and regression trees. Contrast tree uses the WHERE clustering algorithm, which in-turn uses FASTMAP [] to compute clusters that contain instances that are spatially close. Each cluster is assigned a unique cluster id. Then, a simple decision tree is built using a standard classification and regression tree (CART). The tree is built with the following tunable features:

- *Entropy* is used as criteria for generating optimum cuts while building the decision trees. These were shown to be consistent in giving high performance across most data.
- *Minimum samples per leaf* defines the minimum number of instances in a leaf of the decision tree.
- *Max depth* defines the number of levels of recursive splits that are allowed in the tree.

Decision trees can be used to generate decision rule, which when applied to solution sets must theoretically improve performance. For every instance in a "worst" set W, we find the nearest branch with better performance, "B", that "W". All the changes that need to be made in order to achieve "B" constitute the contrast set, "C". Note that the branch variables use decision ranges instead of single point solutions, making implementation of the rules in real life feasible.

B. Why would it work?

In theory, contrast set is an ideal tool for generating decision rules. Some of the key advantages are listed below:

- Firstly, they allow the user to exert a fine grain control over the parameters that needs to be changed, while providing a way to visualize the recommended changes.
- The changes suggested by the contrast sets are inherently local in nature, making the changes practical and potentially easy to implement.
- The contrast tress require a worst case time complexity of $O(n)$, which is a function in linear time.

C. Why did it not work?

D. What worked?

Contrast tree is the second generation of our efforts to use contrast set learners for planning case based reasoning. W2 represents the first generation of our work. W2 is a CBR planner that reflected over the delta of raw attributes []. However, it frequently suffered from an optimization failure. When its plans were applied, performance improved in only $\frac{1}{3}$rd of test cases.

Generation two of that work resulted in IDEA. and HOW use the same recursive clustering method but IDEA made the mistake of summarizing those recursive divisions into a tree structure. Initial results with IDEA were weak, but somewhat positive [35]. We tried unsuccessfully for months to extend and improve the IDEA prototype. The results were most discouraging: those tree-based summaries often suffered from the same optimization failure problem seen with W2. As part of the IDEA work, we built HOW as a straw- man ; i.e. a simpler approach that was to provide a baseline result, above which IDEA was meant to do better. However, HOWs results were so good that we threw away months of work on tree-based planning with IDEA. Now, we strongly recommend HOW over IDEA (and W2) since, as shown by the following results, HOWs plans never lead to performance getting worse. Also, when HOW did improve the expected values of the performance, those performance improvements were an order of magnitude larger than those seen with IDEA.

III. Future Work