

A Preliminary Evaluation of Machine Learning in Algorithm Selection for Search Problems

Correction

Lars Kotthoff

University of St Andrews

`larsko@cs.st-andrews.ac.uk`

Abstract

Some of the results in the original “A Preliminary Evaluation of Machine Learning in Algorithm Selection for Search Problems” paper were flawed because of a problem with the way SATzilla is compiled. This addendum details the cause of the problem and presents the corrected results.

Problem description

The SATzilla 2009 distribution¹ contains a Makefile with targets for the individual binaries for handcrafted, random and industrial instances. The compilation processes differ in the command line options given to the compiler; in particular a special preprocessor definition is changed for each version. This preprocessor definition affects, among other things, which solvers are eligible for being selected to solve a particular instance and which features are computed for each instance.

The `make` tool however does not consider changes in command line options when determining whether to recompile an object file. That is, building all three versions of SATzilla consecutively, i.e.

```
make release-random
make release-crafted
make release-industrial
```

produces three SATzilla binaries with different names that indicate which kind of problem instances it should be used for, but with the same content. All three binaries will only be suitable for random instances, because that version was compiled first and a change in command line options for the subsequent binaries failed to trigger a recompilation.

The instructions in the README file say to run `make clean-all` before building the respective target. If SATzilla is compiled this way, the problem does not occur. The `make clean-all` target deletes any previously compiled binaries however.

We were unaware of the recompilation issue and, as the README does not mention that running `make clean-all` is required for the correctness of the compiled solver, used

Copyright © 2011, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

¹<http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/SATzilla2009F.tar.gz>

the version that was only suitable for handcrafted instances – this one was compiled first – for *all* instances, thus achieving poor performance for the other two instance distributions.

Terminology

We modified SATzilla to not run a presolver and not run the predicted best solver, as we are not interested in actually solving the problem. It should therefore be noted that we are comparing against *a system derived from SATzilla* and not SATzilla proper. In particular, the model that SATzilla learns does not focus on easy to solve problem instances, as those are solved by the presolver.

Preprocessing of SATzilla training data

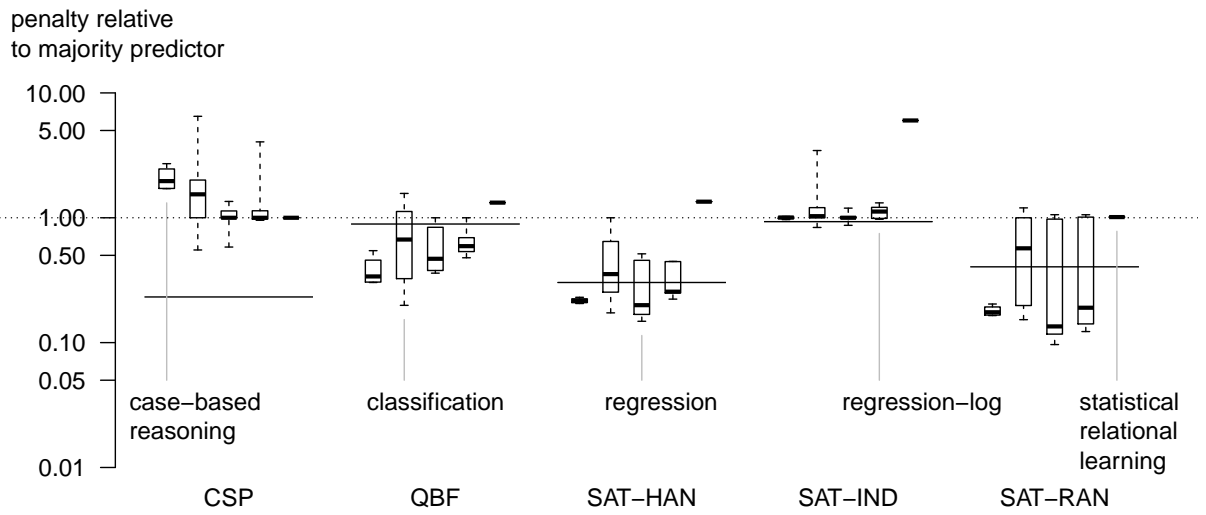
In addition to correcting the problem above, we decided to preprocess some of the solver timeouts in the SATzilla training data². For some solvers on some instances, the reported timeout value is 999999 seconds and thus the penalty for choosing such a solver on such an instance was disproportionately high. We adjusted all timeout values to 3600 seconds. This reduces the highest penalty incurred by the Machine Learning algorithms considerably and gives a fairer comparison.

We furthermore decided to remain closer to what SATzilla does in practice at the expense of some fairness in the comparison. The SATzilla system includes a step where the time required to compute the features for a given problem instance is predicted and, if it is larger than a threshold, no features are computed but a fallback solver is used. The other Machine Learning techniques used here do not include such a step. In the results presented below, our system derived from SATzilla does use this step.

Corrected results

We present the corrected graph for Figure 1 below. The results for the CSP and QBF data sets are unchanged. The performance of almost all Machine Learning algorithms is now significantly better on the SAT data sets because of the adjusted timeout values. SATzilla performs better than the majority predictor in each case.

²<http://www.cs.ubc.ca/labs/beta/Projects/SATzilla/SATzilla2009Data.tar.gz>



The new results do change the evaluation in the original paper, but the general conclusions with respect to the recommended Machine Learning techniques remain the same. We are working on an extended version of the paper that takes the new results into account for the entire evaluation.

Acknowledgments

We thank Lin Xu, Holger Hoos and Kevin Leyton-Brown for pointing out the problem and helping with its investigation and correction.