An Augmented MetiTarski Dataset for Real Quantifier Elimination using Machine Learning*

John Hester[1], Briland Hitaj[2], Grant Passmore[1], Sam Owre[2], Natarajan Shankar[2], and Eric Yeh[2]

[1] Imandra Inc., Austin, TX 78704, USA
{john,grant}@imandra.ai
[2] SRI International, Menlo Park, CA 94025, USA
{briland.hitaj,natarajan.shankar,sam.owre,eric.yeh}@sri.com

Abstract. We contribute a new dataset composed of more than 41K MetiTarski challenges that can be used to investigate applications of machine learning (ML) in determining efficient variable orderings in Cylindrical Algebraic Decomposition. The proposed dataset aims to address inadvertent bias issues present in prior benchmarks, paving the way to development of robust, easy-to-generalize ML models.

Keywords: MetiTarski Variable Ordering · Machine Learning Datasets · Generalizability · Bias.

1 Introduction

Cylindrical Algebraic Decomposition (CAD) is a key proof technique for formal verification of cyber-physical systems such as aircraft collision avoidance systems, autonomous vehicles and medical robotics. While CAD is a complete decision procedure, it is computationally expensive with worst-case exponential complexity. Prior work has demonstrated that machine learning (ML) may be successfully applied to determining efficient variable orderings [2,5]. Much of this work has been driven by CAD problems extracted from applications of the MetiTarski theorem prover [1,7].

However, the original MetiTarski benchmark data is highly imbalanced, inadvertently introducing preferences towards certain variable orders, thus hindering the ability of resulting ML models to generalize to new data [4,6]. Data augmentation can address bias issues while substantially improving the robustness of trained models [3]. In this vein, we make use of inherent symmetries present in the data to create a new *balanced* MetiTarski dataset composed of more than 41K additional challenges. We make the new dataset together with our models publicly available.[3]

* This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR00112290064. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Government or DARPA.

[3] https://github.com/SRI-CSL/augmented-metitarski

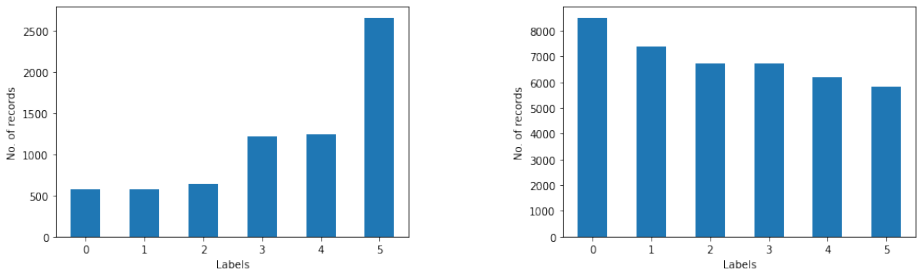
2 The Proposed MetiTarski Dataset Setup

In this section, we provide details about the datasets that we have used in our experiments, and we introduce a new *augmented* dataset designed to remove bias. In addition to the datasets, we provide details on the features considered based on England et al. [2] and Huang et al. [5] respective works, followed by a discussion of our labeling strategy. We conclude the section with a discussion of the machine learning models considered together with their respective hyperparameter setup.

2.1 MetiTarski Datasets

Dataset 1 (Original): The first dataset is predominantly gathered from MetiTarski, by logging MetiTarski's RCF subproblem queries during its proof search [7,2]. The dataset contains 6,895 polynomial systems, together with data on the performance of a CAD-based decision procedure on different orderings of their variables. Every problem in this data set has 3 variables (x_1, x_2, x_3) and thus 6 possible variable orderings.

Dataset 2 (Augmented): We noted that the original MetiTarski dataset was highly imbalanced. While class 0, corresponding to the (x_1, x_2, x_3) variable order contained 580 records, class 5, corresponding to (x_3, x_2, x_1) contained 2,657 records, nearly 4-times more, Figure 1a. We note that this may *bias* ML models towards certain label/s, thus preventing the models from learning relevant information and hindering their generalizability to new, previously unseen data.



(a) Original Dataset (b) Augmented Dataset

Fig. 1: Data distribution per label on both the original MetiTarski dataset and the second, balanced one.

We recognize that variables in the formula and ordering can be swapped without changing the time and cost needed to perform the computation. For instance, swapping variables x_1 with x_2 in the formulas and in the ordering leads to a CAD with the same time and cell cost. While this may seem apparent to a human or a machine reasoner that already has the ability to recognize this

symmetry, for current machine learning systems it needs to be made explicit in the training data. This procedure resulted in a new augmented composed of 41,370 polynomial systems (cf. Figure 1b).

2.2 Feature Engineering and Labeling

We process each set of polynomials extracting features enlisted in [2,5], including the number of polynomials, maximum total degree of polynomials, maximum degree of each x_i, and proportion of each x_i appearing in polynomials and monomials. In addition to the feature set, we assign to each polynomial problem a label ranging from 0...5, where each label translates to one of the 6-possible variable orderings. At present, the label for each polynomial problem corresponds to the variable ordering that takes the least amount of time.

2.3 Models

To evaluate our approach, we used 5-ML classifiers: 1) Support Vector Machines (SVM), 2) k-Nearest Neighbours (k-NN), 3) Decision Trees (DT), 4) Random Forests (RF), and 5) Multi-Layer Perceptrons (MLP). England et al. [2] relies on SVMs, k-NNs, DTs, and MLPs. By following a similar approach, we ensure that our strategy is comparable to the state-of-the-art and thus can be used as a foundation for future adoption of more complex ML strategies, such as Transformers [9] or Graph Neural Networks (GNNs) [8]. We used the `scikit-learn`[4]-based implementations of the ML algorithms. Similar to the works of England et al. [2] and Huang et al. [5], we employed a grid-search strategy with 5-fold cross-validation to identify the right parameter setup for each of the models.

3 Evaluation

In this section, we proceed with the evaluation of the performance of the selected machine learning models (cf. Section 2.3) on identifying the preferred (best) variable order for a given input problem. We transform the problem of determining the best variable order into a multi-class classification problem. In these kind of problems the training set is composed of (x, y) tuples of data, where x is an input sample and y is the corresponding label for that sample. The goal of the learning process translates into the task of finding a function f, such that $f(x) = y$.

In our setting, the input data corresponds to the series of 11-features (cf. Section 2.2) whereas y is one of the 6-labels from $[0, \ldots, 5]$ belonging to a preferred variable order from $[(x_1, x_2, x_3), \ldots, (x_3, x_2, x_1)]$. The features were scaled by subtracting the mean and then scaling to unit variance[5].

[4] `https://scikit-learn.org/stable/`

[5] `https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html`

Training: For each of the datasets, we used 80% of the data for training and kept the remaining 20% as part of the testing set. For Dataset 1 (original), this corresponded to $5,516$ samples for training and $1,379$ for testing, whereas for Dataset 2 (augmented), $33,095$ samples were used during training and the remaining $8,274$ samples for testing. Tables 1 and 2 provide accuracy of each model obtained on the respective training set.

Table 1: Performance of models trained on Dataset 1 when evaluated on Testing set 1 and entire Dataset 2.

Model Trained on Dataset 1	**Performance on Training Set 1**	**Performance on Testing Set 1**	**Performance on Dataset 2 (all)**
SVM	69.38%	58.88%	28.9%
k-NN	75.27%	57.36%	32.21%
DT	75%	55.69%	31.44%
RF	76.39%	58.23%	34.15%
MLP	58.81%	53%	33.64%

Testing: We test each model on 20% of the respective datasets: $1,379$ samples for Dataset 1, and $8,274$ samples for Dataset 2.

Table 2: Performance of models trained on Dataset 2 evaluated on Testing set 2 and Dataset 1. **Perf. on Dataset 1 (all)** also shows (in parentheses) results from training on a "reduced" subset of Dataset 2 obtained by random sampling.

Model Trained on Dataset 2	**Performance on Training Set 2**	**Performance on Testing Set 2**	**Performance on Dataset 1 (all)**
SVM	62.1%	57.39%	60.43% (53.89%)
k-NN	69.2%	54.9%	66.28% (54.19%)
DT	68.03%	55.04%	64.16% (52.27%)
RF	70.47%	55.07%	66.96% (55.66%)
MLP	50.51%	49.62%	48.19% (48.22%)

The performance of models trained on Dataset 1 varied from 53% for the MLP model up to 58.88% for the SVM (Table 1) these results being in-line with state-of-the-art work by England et al. [2] and Huang et al. [5]. Likewise, the models trained on the augmented Dataset 2 exhibit similar performance with the MLP architecture performing poorly with 49.62% accuracy and SVM obtaining up to 57.39% accuracy on the testing set (Table 2).

Evaluation on respective datasets: Model performance is quite promising and substantially better than random choice ($\approx 16.67\%$). However, given the bias of the original dataset (cf. Section 2.1), it is interesting to investigate the performance of models trained on Dataset 1 (the original MetiTarski dataset) and the newly produced Dataset 2, the latter a superset of Dataset 1. As illustrated in Table 1, there is a significant drop in classification accuracy for all models trained

on Dataset 1, with more than 25% drop in some cases. Models trained on the "debiased" Dataset 2 retain good performance when evaluated on Dataset 1. We believe this is due to the training data being balanced and the model potentially seeing some of these samples during training, increasing its decision confidence. Interestingly, we also see good performance for models trained on a "reduced" version of Dataset 2 for which we select one random permutation per problem.

4 Conclusion

We have re-examined a classical dataset for ML-driven CAD variable ordering and observed issues of bias. To address this, we have applied symmetry-based data augmentation to create a debiased version and have shown this improves generalizability. We believe this phenomenon is quite general, and that debiasing with formula symmetries should be a standard tool for applications of ML in computer algebra, program verification, and other fields manipulating mathematical formulas. While this approach generalizes naturally to more variables, there is a bottleneck of exponential growth in the number of distinct orderings that must be considered as the dimension grows (e.g., for 6 variables we have 6! combinations). We intend to investigate variants where, instead of considering a full ordering up front, we consider partial solutions, e.g., the first k variables to project, etc. Nevertheless, an enormous number of RCF verification problems encountered in practice take place over $\mathbb{R}^3$, so even the classical focus on the 3 variable case is well motivated. We plan to extend this work into general-purpose tools and apply it to many problem domains (e.g., Gröbner bases, BDDs, SAT).

References

1. Akbarpour, B., Paulson, L.C.: MetiTarski: An automatic theorem prover for real-valued special functions. Journal of Automated Reasoning **44**, 175–205 (2010)
2. England, M., Florescu, D.: Comparing machine learning models to choose the variable ordering for CAD. In: CICM. pp. 93–108. Springer (2019)
3. Geirhos, R., Rubisch, P., Michaelis, C., Bethge, M., Wichmann, F.A., Brendel, W.: Imagenet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In: International Conference on Learning Representations (2019), `https://openreview.net/forum?id=Bygh9j09KX`
4. Geirhos, R., Temme, C.R., Rauber, J., Schütt, H.H., Bethge, M., Wichmann, F.A.: Generalisation in humans and deep neural networks. Adv.Neur.Inf.Proc. **31** (2018)
5. Huang, Z., England, M., Wilson, D.J., Bridge, J., Davenport, J.H., Paulson, L.C.: Using machine learning to improve CAD. Maths. in C.S. **13**(4), 461–488 (2019)
6. Kawaguchi, K., Kaelbling, L.P., Bengio, Y.: Generalization in deep learning. arXiv preprint arXiv:1710.05468 (2017)
7. Passmore, G.O., Paulson, L.C., Moura, L.d.: Real algebraic strategies for MetiTarski proofs. In: CICM. pp. 358–370. Springer (2012)
8. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks **20**(1), 61–80 (2009)
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)