# Clustering-Based Active Learning for CPSGrader

**Garvit Juniwal**
UC Berkeley
garvitjuniwal@eecs.berkeley.edu

**Alexandre Donzé**
UC Berkeley
donze@eecs.berkeley.edu

**Sakshi Jain**
UC Berkeley
sakshi.jain@eecs.berkeley.edu

**Sanjit A. Seshia**
UC Berkeley
sseshia@eecs.berkeley.edu


L@S 2015, Mar 14-18, 2015, Vancouver, BC, Canada
ACM 978-1-4503-3411-2/15/03
http://dx.doi.org/10.1145/2724660.2728702

## Abstract
In this work, we propose and evaluate an active learning algorithm in context of CPSGrader, an automatic grading and feedback generation tool for laboratory-based courses in the area of cyber-physical systems. CPSGrader detects the presence of certain classes of mistakes using *test benches* that are generated in part via machine learning from solutions that have the fault and those that do not (positive and negative examples). We develop a clustering-based active learning technique that selects from a large database of unlabeled solutions, a small number of reference solutions for the expert to label that will be used as training data. The goal is to achieve better accuracy of fault identification with fewer reference solutions as compared to random selection. We demonstrate the effectiveness of our algorithm using data obtained from an on-campus laboratory-based course at UC Berkeley.

## Author Keywords
Auto-Grading; MOOC; Embedded/Cyber-Physical Systems; Active Learning; Dynamic Time Warping; Density-Based Spatial Clustering of Applications with Noise

**Figure 1:** Cal Climber laboratory platform.

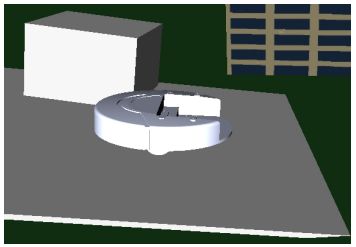

**Figure 2:** Cal Climber in the LabVIEW Robotics Environment Simulator.

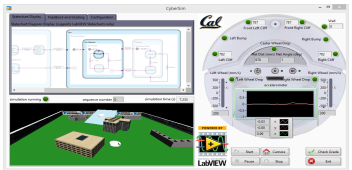

**Figure 3:** Simulator with auto-grading functionality used in EECS 149.1x.


## ACM Classification Keywords

K.3.2 [Computers and Education]: Computer and Information Science Education; I.2.6 [Artificial Intelligence]: Learning.


## Introduction and Background

CPSGrader [4, 1] is a virtual lab automatic grading system designed for courses in cyber-physical systems and robotics. An example of such a course is the *Introduction to Embedded Systems* at UC Berkeley and its online counterpart on edX [5], in which students program the Cal Climber (Fig. 1) robot to perform certain navigation tasks like obstacle avoidance and hill climbing. Students can prototype their controller to work within a simulated environment based on the LabVIEW Robotics Environment Simulator by National Instruments (Fig. 2, Fig. 3.) The dynamical model for the virtual lab is complex and CPSGrader employs simulation-based verification, the only scalable approach. Correctness and the presence of certain classes of mistakes are both checked using *test benches* formalized in *Signal Temporal Logic* (STL) [6], and these test benches are generated in part via machine learning from solutions that have the fault (positive examples) and those that do not (negative examples). In machine learning terminology, this can be thought of as the *training* phase. The resulting test bench then becomes the *classifier* that determines whether a student solution is correct, and, if not, which fault is present. CPSGrader was used successfully in the edX course EECS149.1x offered in 2014.

An issue with this approach is the availability of "positive" and "negative" reference controllers. An instructor has to manually look at the simulation video to decide whether a particular controller is good or bad and then it can be used for training the test bench. In essence, labeling of controllers is an expensive manual process. In this work, we formulate the problem of obtaining labeled data as an active learning problem. We give a clustering-based active learning methodology that takes as input a large set of unlabeled controllers and chooses the controllers that an instructor should label. We believe clustering is useful because amongst many student solutions, the total number solution strategies are still few. We show that selection of training examples based on clustering leads to higher accuracy of classification as compared to random selection of the same number of training examples, and therefore it can lead to reduced overhead for instructors in providing labeled data.

## Approach

Test benches in CPSGrader are generated by machine learning from positive and negative labeled reference solutions (*training data*) [4]. We use this as a black box *training* module TRAIN. A synthesized test bench then is used by a *classification* module CLASSIFY that can label new solutions as being faulty or not.

*Active Learning* [10] is a form of machine learning where the learning algorithm is able to interactively query the user to get the correct labels for new data points. Our active learning technique is described in Algorithm 1. The algorithm works iteratively by using clustering to select the controllers to be added to the training data and using the training module TRAIN at each step. TRAIN first generates a classifier based on some sets of training controllers. Depending on the results of the classifier, the controllers labeled as 0 and 1 are separately clustered by a clustering module CLUSTER. Using the clusters formed, the selection

module SELECT chooses new controllers to get correct labels for. Balance of the number of positive and negative examples is important for TRAIN to work well, hence, SELECT attempts to choose few representatives from each cluster accounting for the balance at the same time. All the selected controllers that were incorrectly labeled by the classifier are now added to the training set and the classifier is trained again. This continues until no fresh training data is added.

We use *dynamic time warping* (DTW) distance [2] as a measure of dissimilarity between controller behavior and density-based spatial clustering for applications with noise (DBSCAN) [9] as the clustering technique. We choose multi-dimensional DTW distance as a measure of dissimilarity between controller behavior since DTW can capture similarities in timed sequence even if matching patterns occur at shifted/scaled positions in time. We use DBSCAN as the clustering technique since it can work with pair-wise distances between data points (instead of requiring full knowledge of feature vectors) and also does not require the number of clusters to the specified in advance. More details of this approach can be found in [3].

## Preliminary Results

The experimental evaluation is done using a collection of solutions implemented by 50 groups of students in the Fall 2013 instance of EECS 149 at UC Berkeley. We evaluate the iterative synthesis algorithm (referred to as ISYN in the rest of the section) by comparing it against the technique RANDOM where we randomly choose the training set and show that ISYN can obtain higher overall accuracy, with a smaller size of training set used.

**Algorithm 1:** ITERATIVESYNTHESIS

**Input**: A dataset of student solutions $\mathcal{D}$, a true labeling oracle $\mathcal{O}$, and a test template $\gamma$ with untuned parameters corresponding to some fault

**Output**: A classifier $\Gamma$ for $\mathcal{D}$

$\mathcal{C}^+, \mathcal{C}^- \leftarrow \emptyset$
**repeat**
  $\Gamma \leftarrow \text{TRAIN}(\gamma, \mathcal{C}^+, \mathcal{C}^-)$
  $\mathcal{D}_0, \mathcal{D}_1 \leftarrow \text{CLASSIFY}(\Gamma, \mathcal{D})$
  $\theta_0, \theta_1 \leftarrow \text{CLUSTER}(\mathcal{D}_0), \text{CLUSTER}(\mathcal{D}_1)$
  $\mathcal{R}_0, \mathcal{R}_1 \leftarrow \text{SELECT}(\theta_0), \text{SELECT}(\theta_1)$
  $\mathcal{C}^+_\Delta = \{C \text{ s.t. } (C \in \mathcal{R}_0 \wedge \mathcal{O}(C) = \text{with\_fault})\}$
  $\mathcal{C}^+ = \mathcal{C}^+ \cup \mathcal{C}^+_\Delta$
  $\mathcal{C}^-_\Delta = \{C \text{ s.t. } (C \in \mathcal{R}_1 \wedge \mathcal{O}(C) = \text{without\_fault})\}$
  $\mathcal{C}^- = \mathcal{C}^- \cup \mathcal{C}^-_\Delta$
**until** $\mathcal{C}^+_\Delta$ or $\mathcal{C}^-_\Delta$ is empty
**return** $(\Gamma(\mathbf{p}), \rho)$

For each fault, we train the test bench using both ISYN and RANDOM and then test the accuracy of the obtained test bench on a disjoint set of solutions. To simplify the comparison, we set the upper bound on the number of training instances used in both techniques as 30. In some cases, ISYN may terminate with less than 30 examples in the training set if the clustering algorithm is not able to find enough number of clusters. To compare accuracy we note the True Positive Rate (TPR), True Negative Rate (TNR), and F-score for both techniques in Table 1. The F-score is individually higher in case of ISYN than RANDOM for all the faults except avoid_left, thus leading to the conclusion that ISYN leads to better accuracy of classification than RANDOM for equal or lesser size of training set. For the fault avoid_right, we see that ISYN performs significantly better than RANDOM for positive examples but worse for negative examples. The reason is that ISYN ends up selecting only 13 (30 being the upper bound) training examples because CLUSTER cannot find enough number of clusters.

**Related Work** DTW has been previously used for classification of temporal sequences of video, audio, and graphics data [8], using an algorithm similar to k-nearest neighbours [7]. Active learning is a popular methodology for cases where obtaining training data is expensive [10]. We have not seen past work that applies clustering based strategies for active learning.

## References


[1] Donzé, A., Juniwal, G., Jensen, J. C., and Seshia, S. A. CPSGrader website. http://www.cpsgrader.org.

[2] Giorgino, T. Computing and visualizing dynamic time warping alignments in R: The DTW package. *Journal of Statistical Software 31*, 7 (8 2009), 1–24.

[3] Juniwal, G. CPSGrader: Auto-grading and feedback generation for cyber-physical systems education. Master's thesis, EECS Department, University of California, Berkeley, Dec 2014.

[4] Juniwal, G., Donzé, A., Jensen, J. C., and Seshia, S. A. CPSGrader: Synthesizing temporal logic testers for auto-grading an embedded systems laboratory. In *Proceedings of the 14th International Conference on Embedded Software (EMSOFT)* (October 2014).

[5] Lee, E. A., Seshia, S. A., and Jensen, J. C. Eecs149.1x cyber-physical systems. https://www.edx.org/course/uc-berkeleyx/uc-berkeleyx-eecs149-1x-cyber-physical-1629.

[6] Maler, O., and Nickovic, D. Monitoring temporal properties of continuous signals. In *FORMATS/FTRTFT* (2004), 152–166.

[7] Mitsa, T. *Temporal Data Mining (Chapter 3)*, 1st ed. Chapman & Hall/CRC, 2010.

[8] Ratanamahatana, C. A., and Keogh, E. Making time-series classification more accurate using learned constraints. SIAM (2004).

[9] Sander, J., Ester, M., Kriegel, H.-P., and Xu, X. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery 2*, 2 (1998), 169–194.

[10] Settles, B. Active learning literature survey. *University of Wisconsin, Madison 52* (2010), 55–66.

| Test Bench | Training Set Size | | TPR | | TNR | | F-score | |
|---|---|---|---|---|---|---|---|---|
| | RANDOM | ISYN | RANDOM | ISYN | RANDOM | ISYN | RANDOM | ISYN |
| avoid_front | 23/133 + 7/74 | 15/133 + 15/74 | 133/133 | 133/133 | 67/74 | 70/74 | 0.97 | 0.99 |
| avoid_left | 23/164 + 7/45 | 15/164 + 15/45 | 164/164 | 164/164 | 42/45 | 39/45 | 0.99 | 0.98 |
| circle | 1/7 + 29/200 | 3/7 + 11/200 | 0/7 | 6/7 | 200/200 | 193/200 | 0.00 | 0.60 |
| hill_climb($\beta = B$) | 26/427 + 4/63 | 14/427 + 16/63 | 427/427 | 427/427 | 55/63 | 60/63 | 0.99 | 1.00 |
| hill_climb($\beta = M$) | 28/442 + 2/48 | 29/442 + 1/48 | 442/442 | 442/442 | 11/48 | 18/48 | 0.96 | 0.97 |
| avoid_right | 24/169 + 6/40 | 10/169 + 3/40 | 70/169 | 169/169 | 40/40 | 26/40 | 0.59 | 0.96 |

**Table 1:** Comparison of ISYN and RANDOM . Training Set Size denotes the (number of positive examples selected in the training set)/(total number of positive examples in data set) + (number of negative examples selected in the training set)/(total number of negative examples in the data set). TPR is the true positive rate of the trained classifier. TNR is the true negative rate.