# AC 2012-4934: AUTOMATIC HANDWRITTEN STATICS SOLUTION CLASSIFICATION AND ITS APPLICATIONS IN PREDICTING STUDENT PERFORMANCE

**Mr. Han-lung Lin, University of California, Riverside**

Han-lung Lin has received his master's degree at the University of Electro-communications in Japan. He is currently a Ph.D. student in computer science at University of California, Riverside.

**Dr. Thomas Stahovich, University of California, Riverside**

Thomas Stahovich received a B.S. in mechanical engineering from the University of California, Berkeley in 1988. He received a S.M. and Ph.D. in mechanical engineering from the Massachusetts Institute of Technology in 1990 and 1995, respectively. He is currently Chair and professor in the mechanical engineering Department at the University of California, Riverside.

**James Herold, University of California, Riverside**

James Herold earned his B.S. in computer science at California Polytechnic State University, Pomona in 2004. He is currently a Ph.D. student in computer science at the University of California, Riverside.

# Automatic Handwritten Statics Solution Classification and its Applications in Predicting Student Performance


## 1. Abstract

In previous research, we found that the spatial and temporal organization of students' solutions to engineering Statics problems correlates with the correctness of their work. Our technique utilized Livescribe™ Smartpens to capture digital records of students' handwritten solutions. Analysis of the temporal and spatial organization of a solution can, for example, enable an instructor to inexpensively identify students who may be struggling in the course and need extra assistance. This requires, however, that the pen strokes be labeled according to the type of content they represent: free body diagrams, equations, or cross-outs of incorrect work (the ink of the Smartpens cannot be erased). For our initial analyses, we manually labeled the pen strokes, but it is too time consuming to do this for the entire data set. This led to our current work on a technique for automatically determining the content type of the pen stoke in a solution. Our approach correctly classified 93.30% of pen strokes in one experiment. We have used these labeled sketches to detect potentially poor-performing students by comparing the equation drawing time and total length of strokes for each student to the mean from all students.


## 2. Introduction

In previous research, we found that the spatial and temporal organization of students' solutions to engineering Statics problems correlates with the correctness of their work. Our technique utilized Livescribe™ Smartpens to capture handwritten student solutions as pen strokes with time-stamped coordinates. Analysis of the temporal and spatial organization of a solution can, for example, enable an instructor to inexpensively identify students who may be struggling in the course and need extra assistance. This requires, however, that the pen strokes be labeled according to the type of content they represent: free body diagrams, equations, or cross-outs of incorrect work (the ink of the Smartpens cannot be erased). For our initial analyses, we manually labeled the pen strokes, but it is too time consuming to do this for the entire data set.

In this paper, we present a four-stage classification algorithm for automatically labeling pen strokes. First, we identify common symbols and letters using a shape recognizer. Second, we classify pen strokes using a machine learning technique. Third, we group pen strokes into larger stroke groups based on the distance and elapsed time between the strokes. Then, in the last stage of our classification algorithm, we correct intra-grouping classification errors.

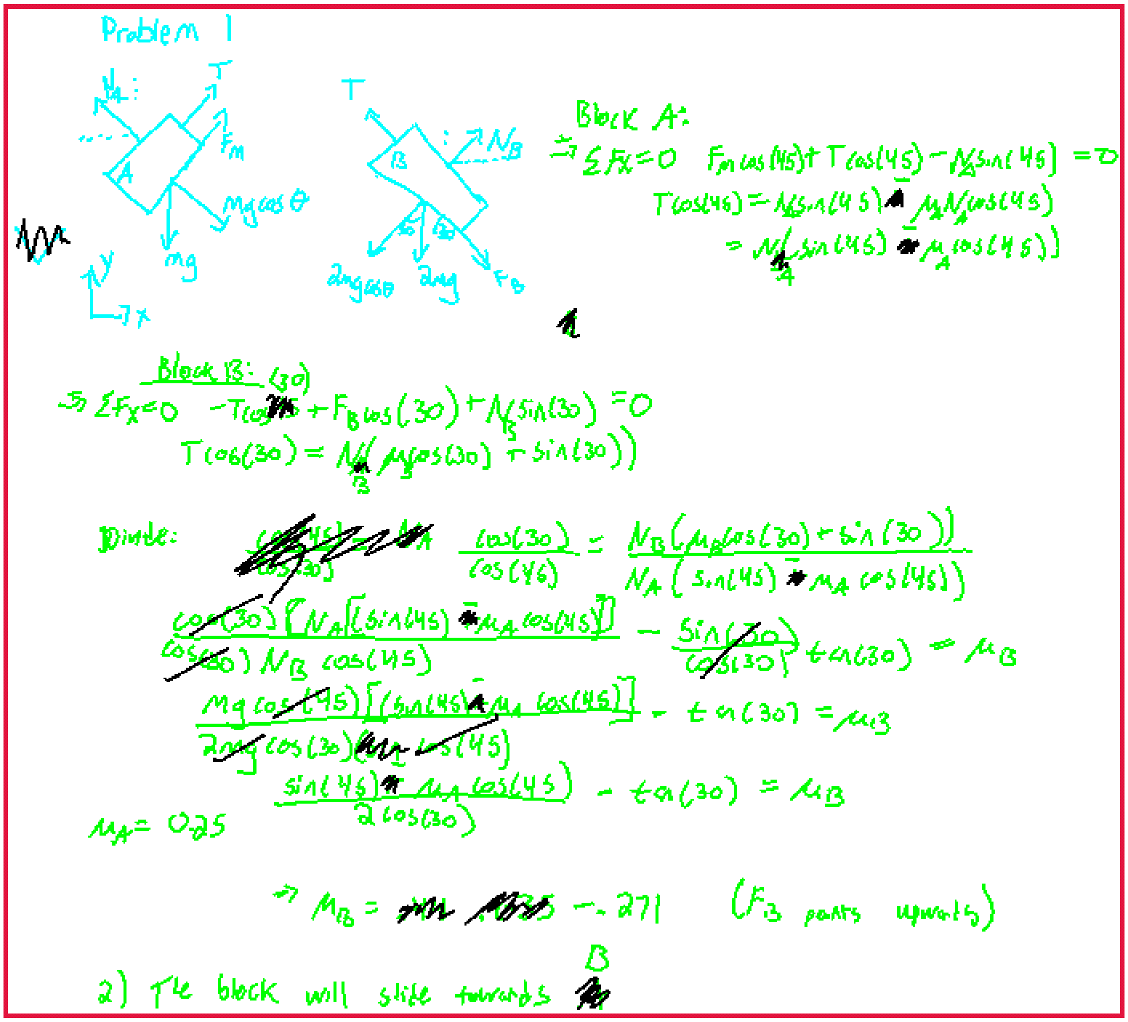

Figure 1: A typical handwritten statics solution comprising equations (green strokes), free body diagrams (cyan strokes), and cross-outs (black strokes).

A number of techniques have been developed to classify strokes. Peterson *et al.*[3], Patel *et al.* [4], and Bhat *et al.* [5] each use a feature-based technique to classify pen strokes. They all characterize each pen stroke using several features. Patel *et al.* used a set of features describing the temporal and spatial organization of the work while Bhat used the zero-order entropy as a feature to identify shape and text strokes. Bishop *et al*.[6] trained and evaluated a classification algorithm using a Hidden Markov Model. Wang *et al*.[7] extend Bishop's approach by integrating a neural network. Gennari *et al*. [8] segmented pen strokes and then used properties of the pen stroke segments to interpret hand-drawn diagrams. Such approaches have typically been tested and developed using neatly-written pen strokes data, and can be less robust when applied to real world data. In our research, we extend the technique presented by Peterson *et al.*[3] by adding 13 new domain-dependent features to characterize statics solutions.

We have evaluated our technique on 810 manually labeled problem solutions containing nearly 300,000 strokes drawn by 90 students. Our approach correctly classified 93.30% of the pen strokes. As we show, this accuracy is sufficient for automatically predicting student performance.

The labeled sketches were used to detect potentially poor-performing students. To account for inherent differences between problems, we averaged equation drawing time across all students. Each student's equation drawing time was then compared to this mean, enabling automatic identification of students who performed poorly. Low effort on equations is an indicator of poor performance. Additionally, we identified strokes which were crossed-out and repeated our analysis excluding those strokes. Since students solved their problems with a pen, they were unable to erase strokes and were instructed to cross out undesired work with horizontal strokes. Removing the crossed-out work from our analysis allowed for more accurate identification of poor student performance. Using this method, we are able to efficiently and automatically identify students who might perform poorly on the final exam. These results demonstrate the feasibility of future systems which can provide rapid feedback and targeted support to poorly performing students and which can inform instructors of the need to adapt their teaching strategies to match students' deficiencies.

The next section presents a detailed description of our classification algorithm. Section 4 introduces the dataset used to train and test our approach. Results are presented in Section 5 and future work and conclusions in Section 6.

## 3. Approach

Our classification algorithm comprises four stages: (1) recognizing letters, mathematical symbols, arrows, and boxes (2) classifying single strokes into one of three semantic classes, (3) grouping classified strokes (4) and correcting errors within each group.

### 3.1 Character Recognition

We have found that features computed using automatic character recognition are the most important features for semantically classifying strokes. We have used an image-based recognizer[2] and several domain-specific, single-character recognizers. The image-based recognizer was chosen because of its ability to recognize multi-stroke characters and its tolerance for over-stroking, which is common in handwritten solutions. We also developed four special-purpose single-character recognizers which identify "plus signs", "equal signs", "answer boxes" and "arrows".

### 3.2 Single-Stroke Classification

This stage of our algorithm maps each of the recognized characters from the previous step to one of three semantic classes: free body diagram, equation, and cross-out. We deployed the feature-based classification algorithm presented by Peterson *et al.*[3] and extended it by adding a number of features which leverage domain-specific heuristics. These features both greatly boost recognition accuracy and provide novel and valuable insight into students' solution processes. These features characterize various aspects of students' handwritten solution style, such as typical stroke size, location on the page, presence of mathematical symbols, and total ink used.

The homework solutions presented here typically contain few cross-outs. Machine learning approaches such as the one presented by Peterson *et al.*[3] have difficulty accurately classifying such rarely occurring cases. For that reason, we have trained an Adaboosted C4.5 decision tree using 13 features to identify whether a stroke is a cross-out or not. The 13 features characterize the underlying ink density and straightness of each stroke. Cross-out strokes typically have high underlying ink density and students frequently cross-out strokes with a single straight line. Strokes are first processed by the cross-out recognizer. Strokes not positively identified as a cross-out are then classified using our extension of Peterson *et al.*'s method.

### 3.3 Stroke Grouping

We have found that there are two types of errors that can be made in classifying strokes: actual errors and contextual errors. Actual errors are straightforward, incorrect classifications, such as part of a beam in a free body diagram that was mislabeled as an equation stroke. Contextual errors are more subtle and depend on the situation in which a stroke appears. Consider the letter "F", which appears frequently in both free body diagrams and equations, depending on whether the letter is used as a force label or as a variable in an equation. In both cases, the geometry of the letter will be the same; it is the context that determines the semantic class of the stroke.

In the third phase of our technique, classified strokes that are both temporally and spatially close are grouped together. These groupings provide context for each stroke which is used later to correct errors.

Our stroke grouping algorithm comprises three steps. In the first step, stroke pairs that both occur within three seconds of each other and are within a specified Euclidean distance are grouped together. We have used a Euclidean distance that is twice the average width of all characters in a sketch. In the second step, stroke pairs whose bounding boxes overlap horizontally with each other are grouped together. Lastly, groups containing too few strokes are merged with the spatially nearest group.

### 3.4 Error Correction

Intuition tells us that strokes which are spatially and temporally close to one another typically correspond to the same semantic class. Students typically solve an equation one step after the next and draw free body diagrams within one region of a page. It follows then, that the strokes within each group resulting from the previous step likely belong to the same semantic class. For that reason, classification errors within each stroke grouping are corrected using a majority vote and two simple heuristics. If the percentage of arrow strokes within a group is above a threshold, all strokes within the group are classified as free body diagram strokes as free body diagrams typically contain a large number of arrows. If the percentage of mathematical symbol strokes within a group is above a threshold, all strokes within the group are classified as equation strokes, as equations typically contain more mathematical symbols than free body diagrams. Lastly, if neither of the previous two thresholds is satisfied, all strokes are classified as the majority class occurring within that group.

## 4. Data set

In the winter quarter of 2010, we conducted a study in which 132 students enrolled in an undergraduate mechanical engineering course on statics were given LiveScribe™ digital pens which they used to complete their homework, quizzes, and exams. These pens serve the purpose of a traditional ink pen, but additionally digitize the ink. This provides a digital record of the students' coursework in the form time stamped (x,y) coordinates of every pen stroke. In total, 6,562 sketches were collected from 12 exam, 30 homework, and 7 quiz problems. We manually labeled 5 exam and 8 homework problems resulting in a total of 293 exam and 810 homework sketches. Our image-based recognizer was trained on a separate set of 374 symbols comprising digits, letters, and several mathematical symbols.

## 5. Results

In the Section 5.1, we present the accuracy of our stroke classification algorithm. Next, in Section 5.2 we present results demonstrating how the automatically classified strokes may be used to accurately predict student performance.

### 5.1 Stroke Classification Accuracy

We trained our stroke classifier using sketches from the midterm and tested its accuracy on homework data. We compare the performance of our labeling technique to that of Peterson *et al.*[3]. Our method achieved an overall recognition accuracy of 93.30% while Peterson's method has an accuracy of 78.56%. Table 2 shows the per-class accuracy of both Peterson's method and ours.

| | FBD | Equation | Cross out | Overall |
|---|---|---|---|---|
| Frequency of each type | 29.23% | 69.75% | 1.02% | 100.00% |
| Peterson's method | 45.55% | 92.71% | 57.34% | 78.56% |
| Our method | 92.64% | 94.50% | 64.26% | 93.30% |

Table 2: Per-class stroke classification accuracy for Peterson’s and our technique.

5.2 Student Performance Prediction Accuracy

In this section, we use the automatically classified pen strokes from homework assignments to predict a student’s final exam performance. Our intuition is that students who spend less than some threshold of time or effort working on their homework assignments are likely to perform poorly on the final exam.

We begin by considering thresholds based on the total amount of ink drawn in an assignment, which is measured by summing the path length of every pen stroke within the assignment. This value is then normalized by the largest amount of ink found in any single assignment. We then group students into bins according to the normalized ink total. The first bin represents students whose normalized ink length is less than 10%, the second represents student between 10% and 20%, and so forth. Figure 2 shows the average final exam grade for each bin. We repeated this analysis disregarding strokes that had been crossed out. Figure 3 shows those results. This figure demonstrates that the students who wrote the least on homework do indeed perform worst.

This first analysis considers only the total amount of ink drawn and does not consider the semantic content of the ink. In our second analysis, we examine the fraction of ink used for equations. We normalize the ink fraction using the mean and standard deviation computed across all students, producing a t-statistic. We plot the average final exam grade of students as a function of the normalized ink length in Figure 4. Figure 5 presents a similar analysis that considers equation time rather than the amount of ink. Figures 4 and 5 show that a student with a normalized equation ink length or equation time greater than 0.9 (i.e., 0.9 standard deviations greater than the mean) is likely to perform poorly on the final exam.

These findings illustrate an early application of our automatic stroke labeling technique. Without considering the meaning of a student’s writing, we are able to automatically identify students who may need additional support. This provides strong evidence that students' writing style is a reliable indicator of their performance and additionally, demonstrates the value of automatic analysis of students' digital coursework.

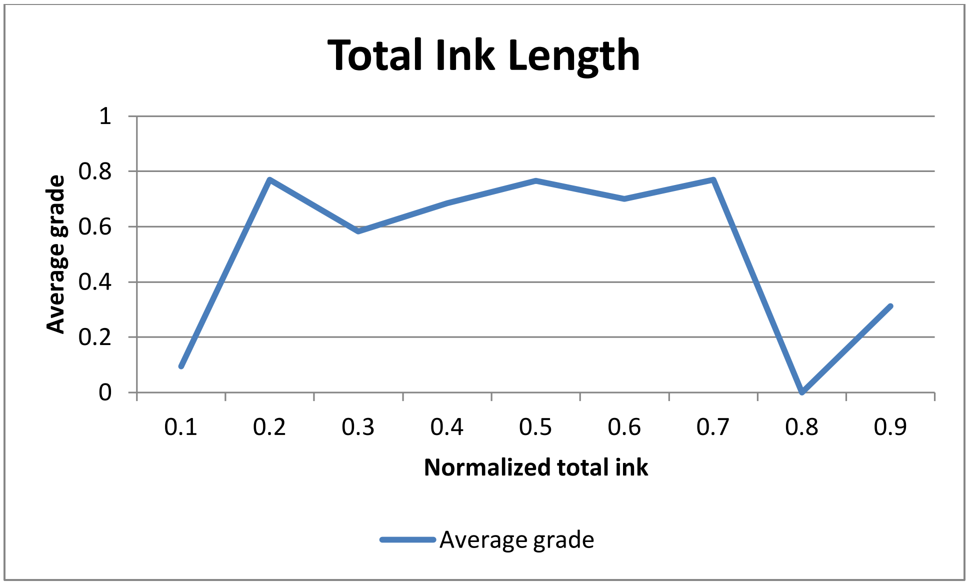


Figure 2: Average grade as a function of the normalized total ink length.

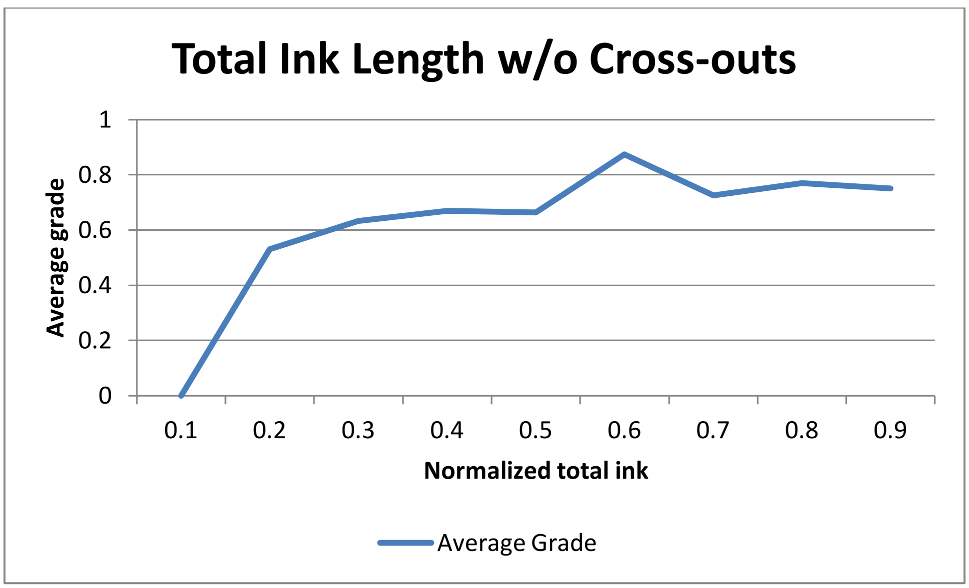


Figure 3: Average grade as a function of the normalized total ink length. Cross-out pen stokes are disregarded.

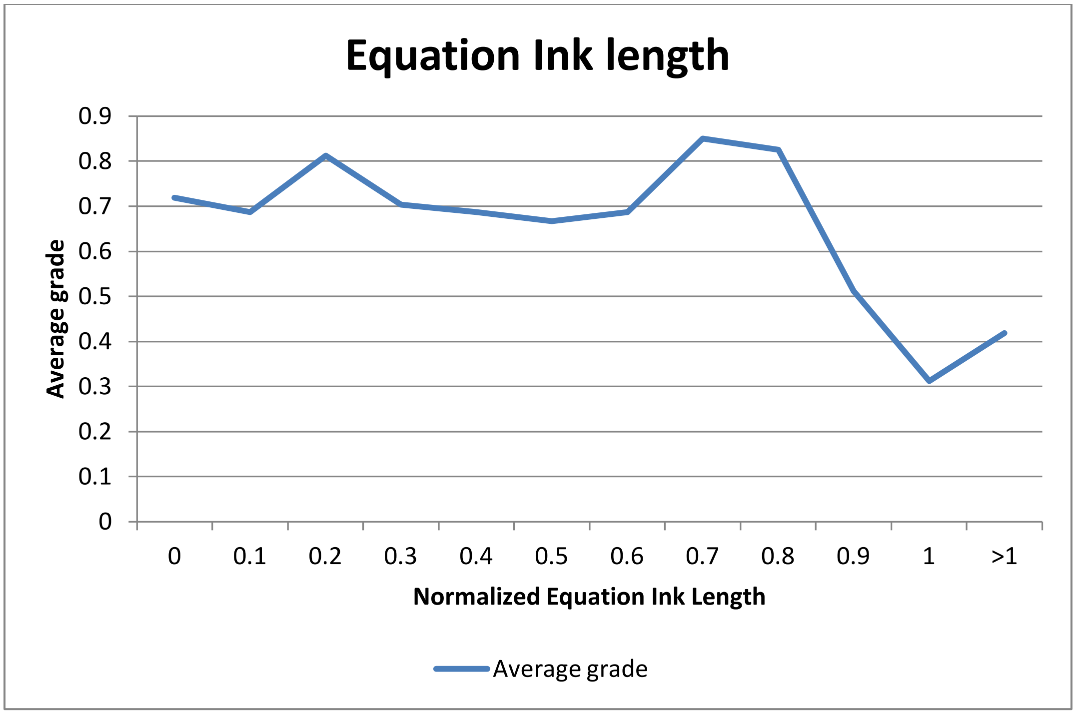


Figure 4: Average grade as a function of normalized equation ink length (t-statistic). Cross-out pen strokes are disregarded.

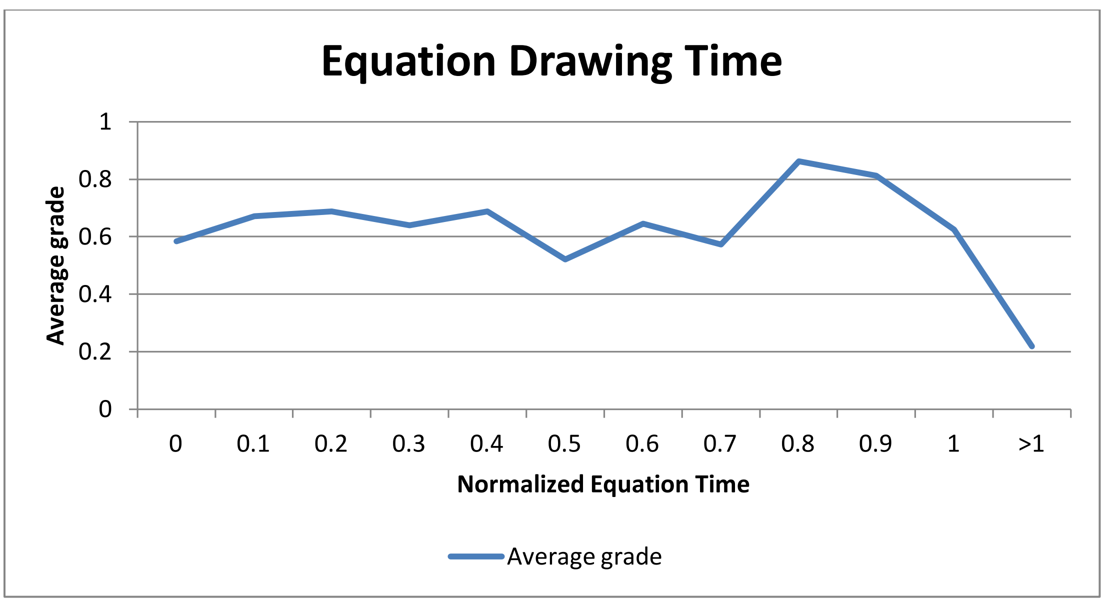


Figure 5: Average grade as a function of normalized drawing time (t-statistic). Cross-out pen strokes are disregarded.

## 6. Conclusions and Future work

In this paper, we have presented an automatic stroke labeling technique and demonstrated an early application. Our technique builds upon prior stroke grouping work by introducing domain-specific heuristics and machine learning techniques. Our technique comprises four steps: recognizing arrows, boxes, and mathematical symbols; classifying single strokes into one of three semantic classes; grouping classified strokes; and lastly, correcting contextual errors within groups. The end result is a semantic class label for every stroke in a handwritten solution.

Our technique is a key enabling technology for large-scale, real-time educational informatics software. As a preliminary demonstration, we have shown in this paper that our technique enables automatic identification of students who might be struggling with their coursework. To do this, we used our classification technique to semantically label a large dataset of students' homework. From those labels, features were computed which characterized the amount of time and effort students spent on their homework and on equations relative to the other students in the class. Using these features, we were able to predict whether or not a student would perform poorly on the final exam.

These results have important implications for future educational systems. With our automatic classification technique, software can monitor the amount of effort students spend on various solution activities. Using this data, the system can determine which students may be at risk of performing badly on the final exam, for example. This will in turn enable software to send targeted instructional materials to struggling students. Additionally, this may help an instructor to adapt lecture materials based on the classes' needs.

## References


[1] Jacob O. Wobbrock , Andrew D. Wilson , Yang Li, Gestures without libraries, toolkits or training: a $1 recognizer for user interface prototypes, Proceedings of the 20th annual ACM symposium on User interface software and technology, October 07-10, 2007, Newport, Rhode Island, USA

[2] Levent Burak Kara, Thomas F. Stahovich, An Image-Based, Trainable Symbol Recognizer for Hand-drawn Sketches Computers & Graphics 29(4): 501-517 2005

[3] Eric Peterson, Thomas Stahovich, Eric Doi, Christine Alvarado, Grouping Strokes into Shapes in Hand-Drawn Diagrams *Proc. of the 24th AAAI Conference on Artificial Intelligence (AAAI-10),* 2010, pp. 974-979

[4] Patel, R.; Plimmer, B.; Grundy, J.; and Ihaka, R. 2007. Ink features for diagram recognition. In Proc. of SBIM.

[5] Bhat, A., and Hammond, T. 2009. Using entropy to identify shape and text in hand-drawn diagrams. In IJCAI.

[6] Bishop, C. M.; Svensen, M.; and Hinton, G. E. 2004. Distinguishing text from graphics in on-line handwritten ink. In Proc. of the Int. Workshop on FHR, 142–147.

[7] Wang, X.; Biswas, M.; and Raghupathy, S. 2007. Addressing class distribution issues of the drawing vs. writing classification in an ink stroke sequence. In Proc. of SBIM.

[8] Gennari, L.; Kara, L. B.; and Stahovich, T. F. 2005. Combining geometry and domain knowledge to interpret handdrawn diagrams.

[9] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; and Witten, I. H. 2009. The WEKA data mining software: An update. *SIGKDD Explor. Newsl.* 11(1).