UNIVERSITY OF CALIFORNIA
RIVERSIDE

Estimating Student Competence in Engineering Statics from a Lexical Analysis of Handwritten Equations

A Dissertation submitted in partial satisfaction
of the requirements for the degree of

Doctor of Philosophy

in

Computer Science

by

Hanlung Lin

March 2015

Dissertation Committee:

Professor Thomas Stahovich, Chairperson
Professor Tao Jiang
Professor Christian Shelton
Professor Vagelis Hristidis

Acknowledgments

I would like to begin by acknowledging the great mentorship and direction of my advisor, Dr. Stahovich, who provided useful guidance not only in research but also in life. Bucket list burger is now in my bucket list. I greatly appreciate all the time, dedication, and wisdom that you have shared with me. I enjoyed the research done with you and am proud to say that I worked with you.

Next, I have to acknowledge the great love and support I have received from my family. Without you I would not be where I am today. Especially during the last year of my Ph.D. life, it was the toughest time in my life. Baby Audrey came to the world, TA duties, wrapping up my research, and seeking a job. Lots of things happened at the same time, but you always believed in me and had my back. I would like to thank my little brother for taking care of Mom and Dad while I was on the other side of the Pacific Ocean. I thank my wife Yu-Chen and my baby Audrey, for loving me and having patience as I spent many nights working on my research.

I would like to share my deep appreciation to all friends. Thanks to Mary Ann, Kimberly, and Stephanie for inviting us to your home, providing a lot of useful information and playing with Audrey. Thanks to Gary, Li-Ting, Yu-Ting, Irene, Dason for helping us find a home and buy a car, and all the other help. Thanks to Ei-Wen, Yan and many others for helping me seek a job.

Lastly, I want to thank everyone in the Smart Tools lab for all the help you have given. Without you, I could not have accomplished my oral exam, proposal defense and dissertation. Thank you for enriching my American life. It was very nice to work with you: Eric, Josiah, Jack, Tim, Jim, Reyna, Levi, Nicholas, Chun-Han, Kevin, Justin, Matthew, and Fatemah. Special thanks go to Teresa for helping to edit this dissertation. Hope we can still hangout and have fun in the future.

To my family.

ABSTRACT OF THE DISSERTATION

Estimating Student Competence in Engineering Statics from a Lexical Analysis of Handwritten Equations

by

Hanlung Lin

Doctor of Philosophy, Graduate Program in Computer Science
University of California, Riverside, March 2015
Professor Thomas Stahovich, Chairperson

We present a technique that examines handwritten equations from a student's solution to an engineering statics problem and estimates the correctness of the work. The solution is recorded with a smartpen that digitizes the writing with time stamps. Our technique first separates equation pen strokes from other content, such as diagrams. Then the equation pen strokes are grouped first into individual equations, then into individual characters. A character recognizer is used to recognize each character, and then a Hidden Markov Model is used to correct recognition errors. The equation text is characterized by a set of features. Some of these describe the frequency of various symbols and symbol combinations, such as a letter following a mathematical operator. One feature describes the frequency with which units of measure (e.g., "kg") appear, while another describes inter-character pauses. This set of features is used to construct SVM regression models to predict the correctness of the work. We tested our approach on a corpus of solutions to exam problems from an undergraduate statics course. The models predicted the grade assigned by the instructor with an average coefficient of determination of 36%. We also combined our features with an existing set of features that describe the temporal and spatial organization of a handwritten solution to a statics problem. Using both sets of features, the SVM regression models predicted the grade on the exam problems with an average coefficient of determination of 56%. This is a surprising result given that none of the features consider the semantic content of the writing, or even the correctness of a student's final answer.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The goal of educational data mining is to apply data mining techniques to understand learning and improve teaching (Romero and Ventura [3]). Traditionally, large-scale education research studies have relied on summative assessments, such as measuring changes from pre-test to post-test (e.g., Garcia [4]). Because capturing and analyzing students' ordinary learning activities (for example by using video recording (Blanc et al. [5])) can be impractical for large scale studies, instruments such as surveys (e.g., Jordan [6] and Lee [7]) are often used for formative assessment. Current efforts in educational data mining, examine readily available digital data sources, such as data from course management systems like iLearn, Loncapa, and Moodle (e.g., Kruger, Merceron, and Wolf [8]; Romero et al. [9])and log files from intelligent tutoring systems (e.g., Beal and Cohen [10]; Shanabrook et al. [11]; Mostow, Gonzalez-Brenes, and Tan [12]). However, these are only a small part of learning.

Our goal is enable data mining for formative assessment of students' ordinary handwritten work. Handwritten homework is an essential component of undergraduate instruction in STEM fields as it allows students to put into practice what is taught in lecture. It also enables instructors to assess a student's understanding of the material. Despite advances in computational systems, computer-based homework has not replaced handwritten assignments. Such systems often grade only the final answer and thus do not directly examine a student's problem-solving processes. However, the alternative — manually grading homework — can be intractable. For example, grading every problem in a homework assignment for an engineering course could take 30 to 45 minutes per student. Thus, a course with 150 students could require more than 100 hours per week just to grade homework. As a work around, some instructors grade only a subset of the problems or assign a grade based solely on submission of the assignment. However, this reduces the feedback that students receive and feedback is critical for learning.

We aim to build an automatic grading system to help instructors and teaching assistants (TAs) do this tedious work. However, it is a challenging problem. As shown in Figure 1.1, a typical handwritten statics solution comprising equations (green strokes), free body diagrams (cyan strokes), and cross-outs (black strokes). It needs a perfect recognizer to identify shapes in the three semantic classes to enable accurate grading. Our goal is to provide an inexpensive means of estimating the correctness of a student's work. In this way, our techniques may be useful for creating an automated early-warning system that monitors a student's coursework and identifies when the student may be having difficulty.

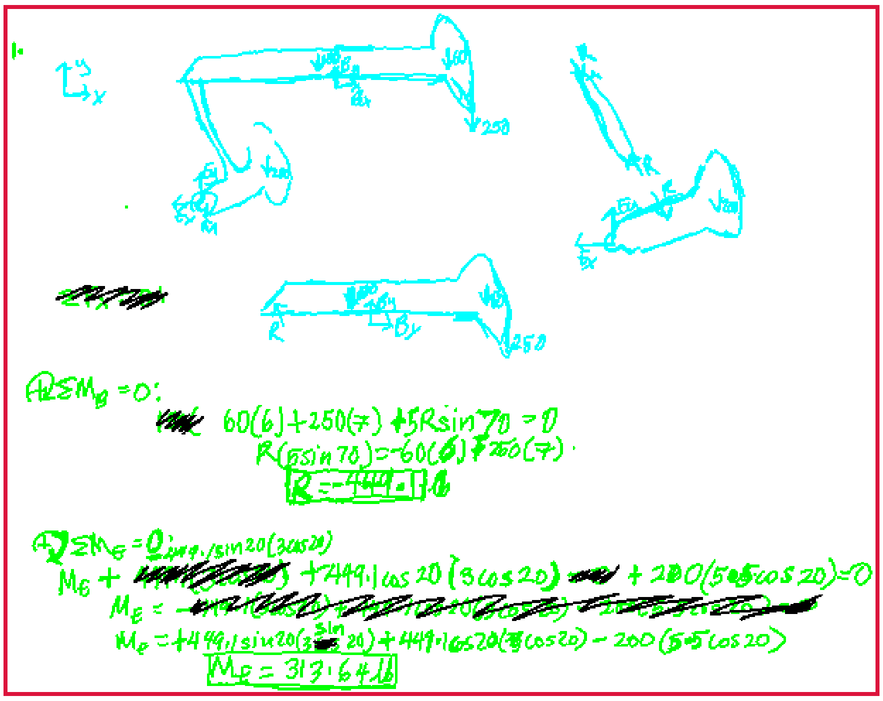


Figure 1.1: A solution to a statics problem. Cyan = free body diagram, green = equation, black = cross-out.

1.1 Approach

Figure 1.2 is a typical statics problem. It asks a student to calculate forces. Figure 1.3 is what a student's solution looks like. Figure 1.4 demonstrates what our system does. It predicts the grade that the student gets on the problem. This graph shows the relationship between the grade our system predicted and the grade assigned by a grader. There is a very strong correlation between the two.

Figure 1.5 shows an overview of our system. In this research, we use Livescribe smartpens to capture students' handwritten homework and exam solutions. These devices have an integrated camera and are used with dot-patterned paper. They serve the same function as a traditional ink pen and also record the work as time-stamped pen strokes, thus enabling both temporal and spatial analysis of the writing. Students use the digital pen to write their homework assignments and exams. We then collect digitized pen stroke data from the pen. Our techniques first extract features from this data and then build regression models, for example, to predict grades.

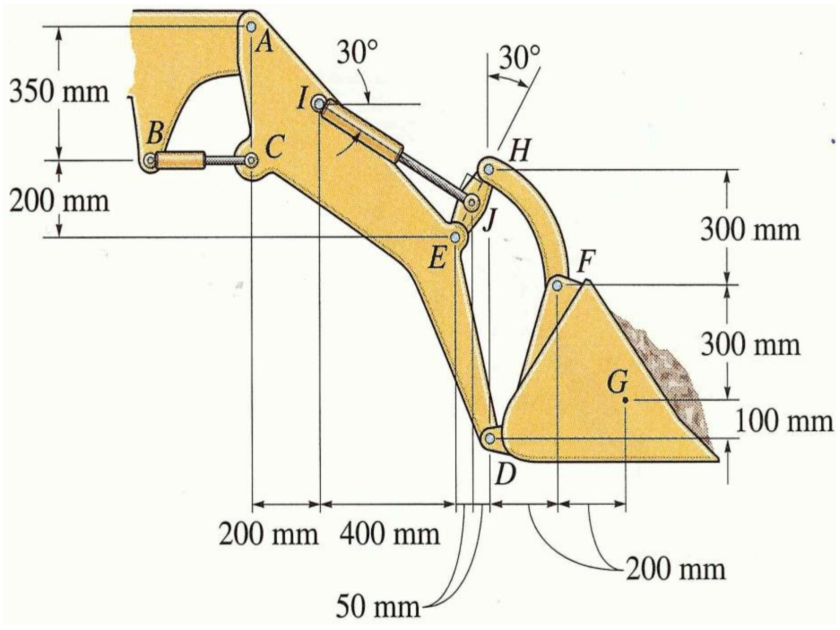
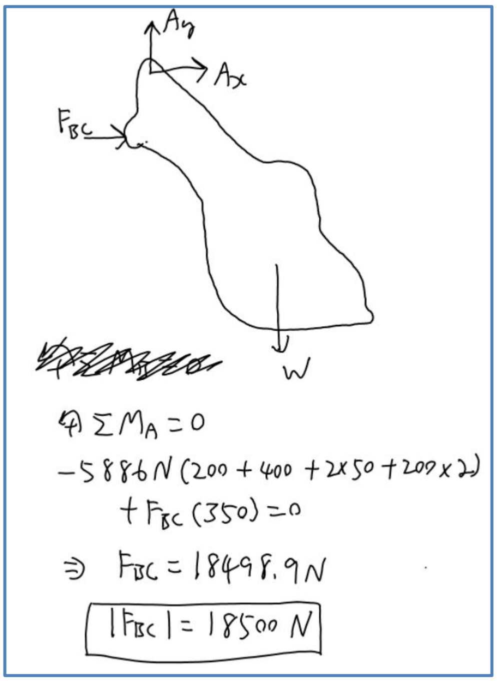


Figure 1.2: A sample statics problem.



$\circlearrowleft \sum M_A = 0$

$-5886N(200 + 400 + 2\times 50 + 200\times 2)$
$+ F_{BC}(350) = 0$

$\Rightarrow F_{BC} = 18498.9N$

$\boxed{|F_{BC}| = 18500\ N}$

Figure 1.3: A sample solution made by student.

Figure 1.6 is another student's handwritten solution. To extract features from the solution, our technique begins with identifying the semantic type of a stroke and separating it into one of three classes: equation, free body diagram, and cross-out (Figure 1.7). We present a two-stage approach to automatically classify pen strokes in this manner. We first use a special-purpose classifier to identify them. We then use a second classifier, which builds on Peterson's technique, to separate the remaining pen strokes into equations and diagrams.

Next, our algorithm groups strokes into equation groups and character groups (Figure 1.8 and Figure 1.9). After separating the strokes by their semantic type, grouping becomes much easier. We only focus on the equation part which allows us to develop a model with less features and higher accuracy. Our equation grouper only needs three features and our character grouper needs only two features. We then apply shape recognition again with the image-based recognizer and Hidden Markov Model for refining the results (Figure 1.10).

Once the equations have been recognized, we compute a variety of features from the text. One feature describes temporal properties of the writing. The remainder describe the frequency of occurrence of various patterns of characters. This set of features is used to construct SVM regression models to predict the correctness of the work.

We evaluated the performance of our stroke classifier, equation and character groupers using our homework dataset. We used our exam dataset to investigate the

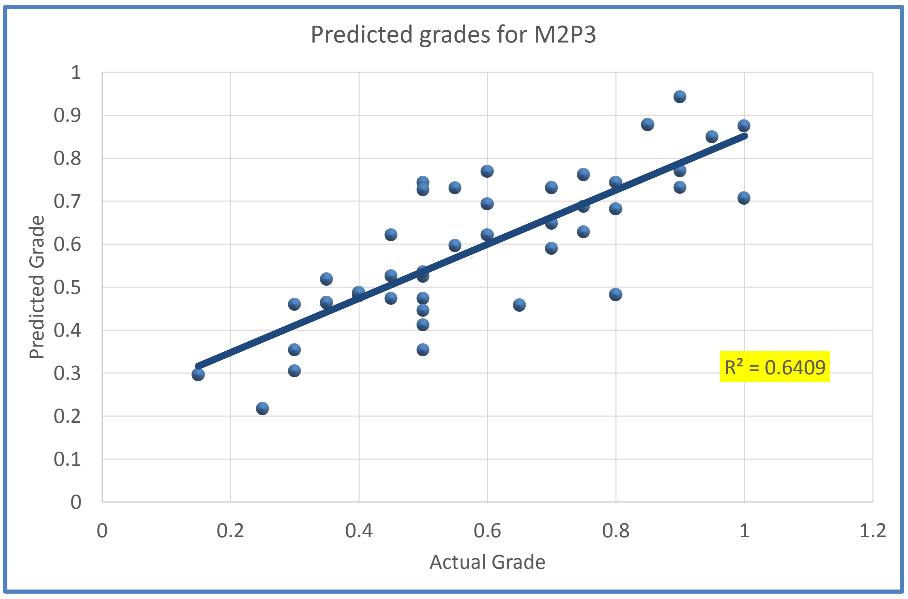
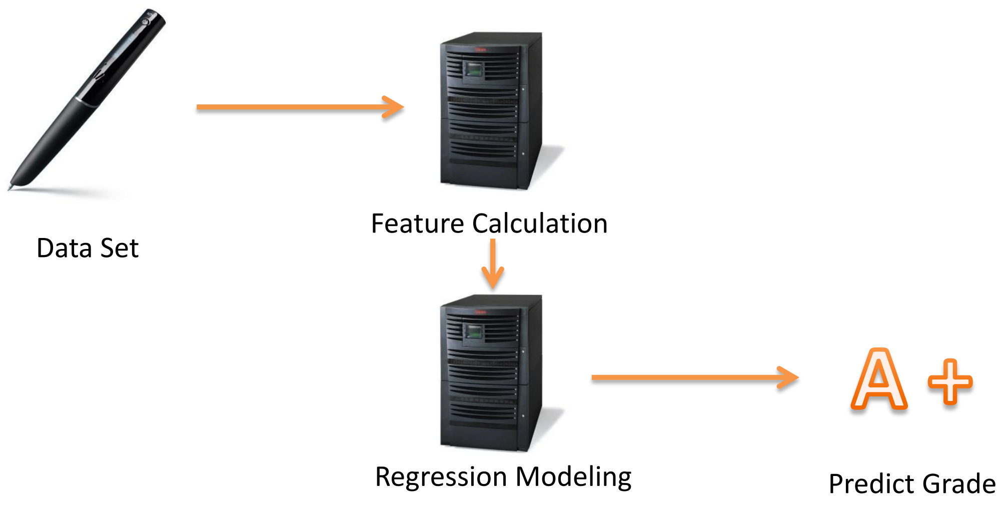


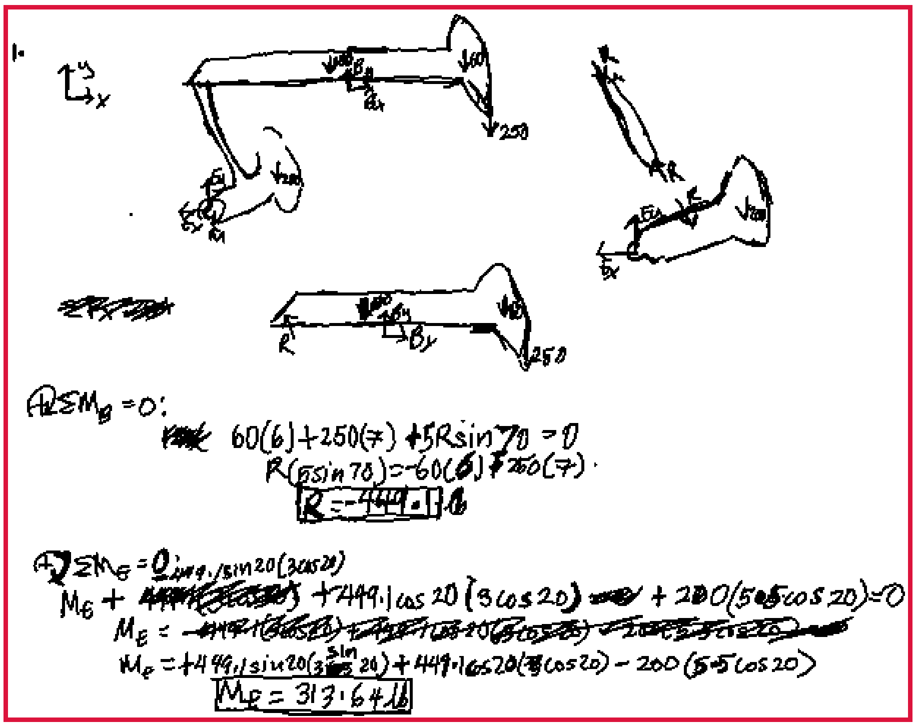
Figure 1.4: The relationship between the grade our system predicted and the grade a grader assigned.



You al

Figure 1.5: The overview of our system. Our technique first extracts features from raw data, then builds a regression model. Finally, predicts grades.

ability of our features to predict student competence. Full descriptions of these datasets can be found in Chapter 6.



$\circlearrowleft + \Sigma M_B = 0$:

$60(6) + 250(7) + 5R\sin 70 = 0$

$R(5\sin 70) = -60(6) - 250(7)$

$\boxed{R = -449.1}$ lb

$\circlearrowleft + \Sigma M_E = 0$; $449.1 \sin 20 (3\cos 20)$

$M_E +$ [illegible] $+ 449.1\cos 20\,(3\cos 20) + 200(5.5\cos 20) = 0$

$M_E =$ [illegible]

$M_E = +449.1\sin 20(3\sin 20) + 449.1\cos 20(3\cos 20) - 200(5.5\cos 20)$

$\boxed{M_E = 313.64}$ lb

Figure 1.6: Another solution to a statics problem.

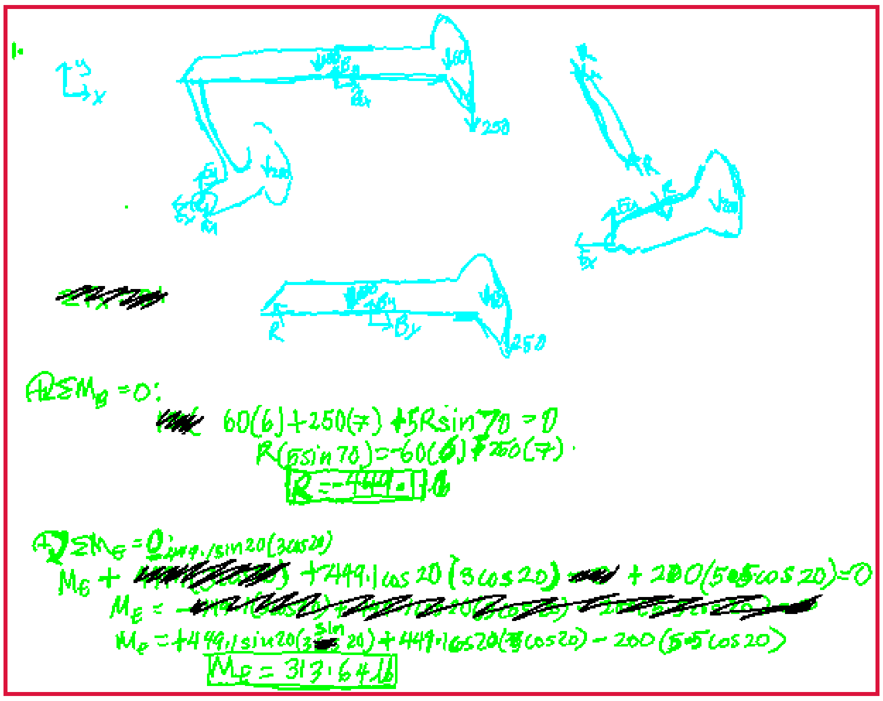

$+\circlearrowleft \Sigma M_B = 0$:
$60(6) + 250(7) + 5R\sin 70 = 0$
$R(5\sin 70) = -60(6) - 250(7)$
$R = -449.1\,lb$
$+\circlearrowleft \Sigma M_E = 0$
$M_E = +449.1\sin 20(3\sin 20) + 449.1\cos 20(3\cos 20) - 200(5+5\cos 20)$
$M_E = 313.64\,lb$
250

Figure 1.7: Separate free body diagrams, equations, and cross-outs. Cyan = free body diagram.Green = equation. Black = cross-out

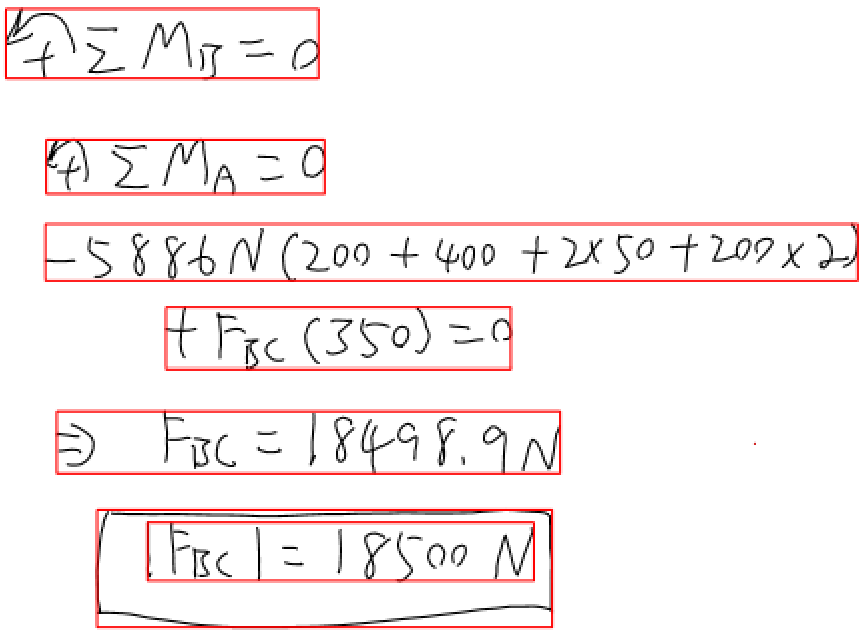

$\circlearrowleft + \sum M_B = 0$
$\circlearrowleft + \sum M_A = 0$
$-5886\,N(200 + 400 + 2 \times 50 + 200 \times 2)$
$+ F_{BC}(350) = 0$
$\Rightarrow F_{BC} = 18498.9\,N$
$|F_{BC}| = 18500\,N$

Figure 1.8: Grouping equations.

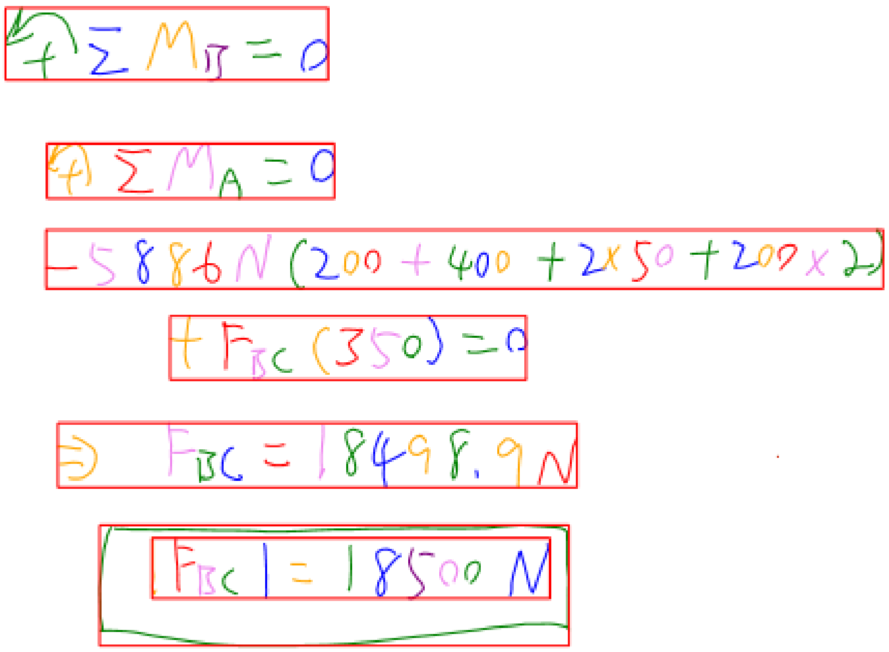


Figure 1.9: Grouping characters.

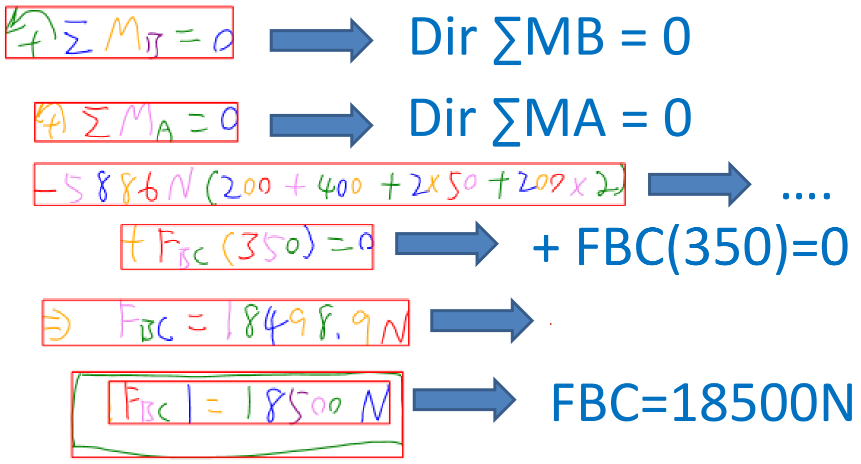


Figure 1.10: Character recognition.

1.2 Contributions

In this work we examine stroke separating, grouping in handwritten sketches and estimating student competence from a lexical analysis of handwritten equations. This dissertation makes contributions in sketch understanding and student modeling.

1.2.1 Sketch Understanding

1. We present an algorithm for stroke classification. Our method is more accurate than the previous methods we tested.

2. We have developed an efficient and accurate technique for grouping handwritten pen strokes into equation groups and character groups that is as accurate as, or more accurate than, other methods we tested. These grouping techniques are typically more efficient than previous approaches.

1.2.2 Educational Informatics / Student Modeling

1. We present the first method which uses lexical properties of handwritten equations on predicting student's performance.

2. We demonstrated that an analysis of the lexical properties of handwritten equations can be used to inexpensively evaluate their correctness.

3. We identified characteristics of high-performing students.

4. This work has several applications. For example, our techniques may provide the basis of an automated student feedback system. It may be useful for creating an automated early-warning system that monitors a student's coursework and

identifies when the student may be having difficulty. In addition, our work may have application to online education, especially for massive open online courses (MOOCs). For MOOCs to become an effective part of our higher education system, new assessment methods are needed. Our technique provides one means for automated assessment of students' handwritten problem-solving.

1.3 Outline

This dissertation is organized as follows: Chapter 2 places our work in the context of related work. Chapter 3 details our approach to stroke classification. Chapter 4 describes our techniques for grouping strokes into equations and characters. Chapter 5 presents the features we extracted from sketches. Chapter 6 describes data sets we used in this dissertation. Chapter 7 presents results of the methods described in Chapter 3,4, and 5. Chapter 8 is the discussion of the methods. Finally, Chapter 9 presents our conclusions.

Chapter 2

Related Work

An important application in biomedical studies is virtual screening, a very efficient computer-aided process in rational drug design. Historically, drugs were discovered through identifying the active ingredient from traditional remedies or by serendipitous discovery. However, the speed of accumulating knowledge for the use of medicine drugs by these conventional methods prohibits us from tackling various critical diseases in a reasonable time and within reasonable cost today. To increase the chance of discovering effective medicine drugs, virtual screening is a widely used technique to search possible drug candidates from known chemical compound libraries. The steps of conducting virtual screening experiments involve the identification of screening hits, medicinal chemistry and optimization of those hits to increase the affinity, selectivity (to reduce the potential of side effects), efficacy/potency, metabolic stability, and oral bioavailability. Once a compound that fulfills all of these requirements has been identified, the process of drug development prior to clinical trials will begin.

A number of techniques have been developed to classify strokes. Peterson et al.

[13], Patel et al. [14], and Bhat et al. [15] each use a feature-based technique to classify pen strokes. They all characterize each pen stroke using several features. Patel et al. [14] used a set of features describing the temporal and spatial organization of the work while Bhat et al. [15] used the zero-order entropy as a feature to identify shape and text strokes. Bishop et al. [16] trained and evaluated a classification algorithm using a Hidden Markov Model. Wang et al. [17] extend Bishop's approach by integrating a neural network. Gennari et al. [18] 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. [13] by adding ten new domain-dependent features to characterize statics solutions

Grouping strokes into equations or distinct objects is one of the most difficult problems in sketch understanding. Some systems require that users complete an equation before beginning another [[19], [20]]. Stahovich et al. [1] present a two-stage clustering algorithm that first classifies pen strokes into different classes of objects, and then groups strokes with like classifications into clusters representing individual objects. Blagojevic et al. [21] presented a feature-based technique which is similar to Stahovich's. They use more than 100 features on classifying strokes. Some other systems require that the user explicitly specify which strokes belong to which object by pausing between each strokes. These techniques are possible when using writing equations or characters on a PC while it is almost impossible to apply the techniques on students' solutions which are written on paper. Josiah developed a feature-based technique to group pen strokes into characters.

Research in educational data mining has seen a dramatic increase in the past few years (Romero and Ventura [3]). Much of the data used in this work is extracted from log files of intelligent tutoring systems (Beal and Cohen [10]; Shanabrook et al. [11]; Mostow, Gonz‘alez-Brenes, and Tan [12]) and learning management systems such as Moodle or Blackboard (Kruger, Merceron, and Wolf [8]; Romero et al. [9]). This work relies on a variety of data mining techniques including clustering (Stevens, Johnson, and Soller [22]; Trivedi et al. [23]), model prediction (Mostow, Gonz‘alez-Brenes, and Tan [12]; Li et al. [24]), and sequence analysis (Beal and Cohen [10]; Shanabrook et al. [11]; Kruger, Merceron, and Wolf [8]; Romero et al. [9]). Our work differs from this in that we record and mine data from learning activities in natural environments, rather than in artificial online environments. The work of Oviatt et al. [25] suggests that natural work environments are critical to student performance.

Researchers have used video recordings to examine student problem solving (Blanc [5]). However, video analysis is labor-intensive. Our pen stroke data is better suited for automated analysis.

Recently, researchers have begun to use smartpens to examine the relationship between homework activities and academic performance. For example, Rawson and Stahovich [26] examined the relationship between homework effort and course performance. Effort was represented by a set of features describing the amount of writing (in pixels) and the distribution of the writing activity over the assignment period. These features were used to construct models predicting course grade. Herold, Stahovich, and Rawson [27] used a similar approach that considered effort on an assignment as a whole and on individual problems.

Herold, Zundel, and Stahovich [28] represent homework activity as sequences of actions, including diagram drawing, equation writing, and breaks. Differential data mining techniques were used to differentiate the activity sequences of students who achieved a high exam grade from those who achieved a low grade. All of these studies examined homework activity (effort) to predict future achievement in the course. By contrast, our work examines the lexical properties of equations to predict their correctness.

Herold and Stahovich [29] used smartpens as assessment tool to examine how self-explanation affects the order in which students solve assigned homework problems. The study found that students who generated self-explanations of their work were more likely to finish each problem before starting the next than were students who did not generate self-explanations.

Van Arsdale and Stahovich [30] developed a technique for estimating the correctness of a student's handwritten solution to a statics problem. They computed 10 features describing the spatial and temporal organization of the solution process and used them to construct stepwise regression models predicting the grade students achieved on the work. Our work is complementary in that we consider lexical properties of equations rather than the organization of the solution process.

Cheng and Rojas-Anaya [31] examined pauses that occurred as students copied equations, and found that the number of long pauses was indicative of competence. A pause is the interval of time from a pen up event to the next pen down event. They defined a long pause as one longer than twice the median pause occurring while the student wrote his or her name. In our work, we consider pauses between characters rather than pen strokes. To apply our technique to large datasets collected in the classroom, it was necessary for us to develop techniques to automatically identify equation groups. Similarly, as we consider inter-character pauses, it was necessary to develop a character grouper.

There is a long history of research on information extraction (IE) techniques (Hobbs et al. [32]) for extracting relations from machine readable documents. This is distinguishable from attempting to understand the entire content of such documents. Older techniques typically relied on domain dependent attributes and were rule-based (Miller et al. [33]) or used machine learning (Culotta [34]). More recently, researchers have focused on open IE techniques that are more extensible than prior methods because they do not require domain knowledge (e.g., (Banko et al. [35]; Soderland et al. [36])). Recently, Herold and Stahovich [29] used an open IE technique to determine if students' self-explanation of their work contained concepts that experts used to explain their work.

Our work differs from IE techniques in that we do not extract relations or concepts from the text. Instead, we examine how the frequency of patterns of symbols correlate with the correctness of the work. In this sense, our work is similar to that of Rhodes et al. [37] which characterized the words in students' self-explanations with "term frequency inverse document frequency" (Robertson and Jones [38]) and used this to predict student grades on homework assignments.

Chapter 3

Stroke Classification

Before the equations can be interpreted, they must be distinguished from diagrams and cross-outs. (Because the digital pens use ink which cannot be erased, students must cross out incorrect work.) We explored the use of Peterson's [13] general-purpose pen stroke classifying algorithm for this task. Although it achieved high accuracy, we were able to develop a special-purpose technique that was both more efficient and more accurate for our task.

Our technique is illustrated in Figure 3.1. Because cross-outs are a rare class, we first use a special-purpose classifier to identify them. We then use a second classifier, which builds on Peterson's technique, to separate the remaining pen strokes into equations and diagrams. We have found that using this two-stage classification approach is far more accurate than directly performing three-way classification using a single general-purpose classifier.

To help distinguish free body diagrams from equations, we perform preliminary shape recognition to attempt to identify alpha-numeric characters and arrows. The goal of this first step is to identify likely characters and arrows to help identify regions containing equations and diagrams, respectively. We use the results of the preliminary recognition to compute 10 features which we use to extend the 27 features that Peterson uses.

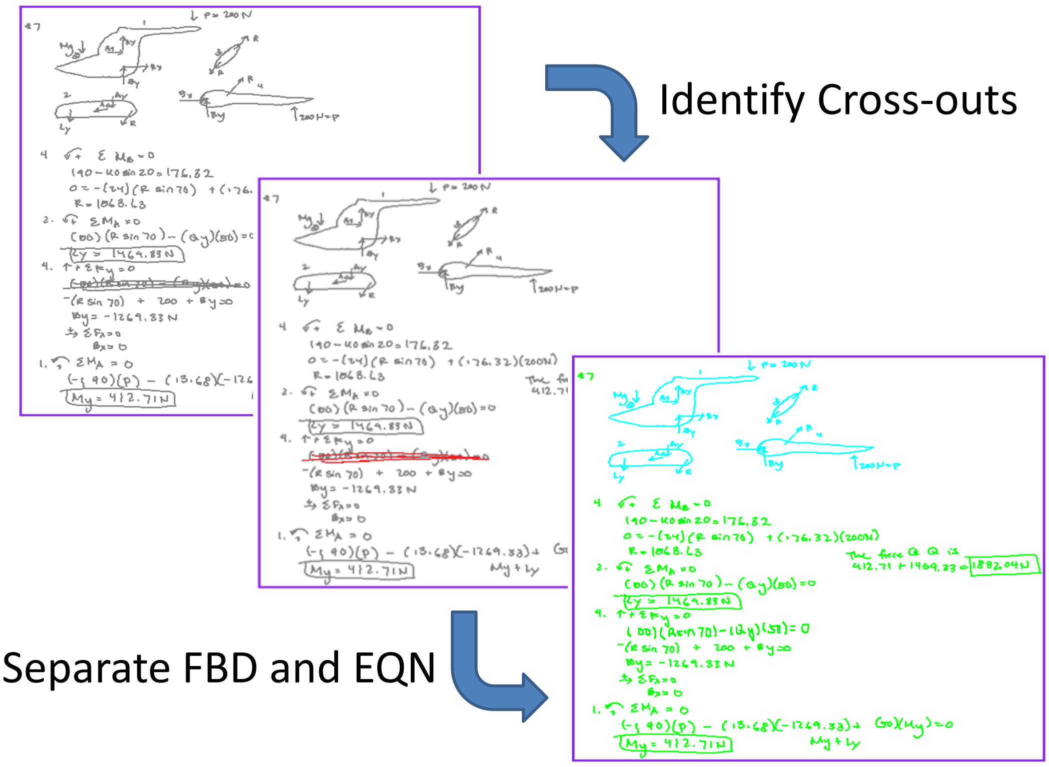


Figure 3.1: Stroke classifier overview.

3.1 Cross-Out Classifier

Handwritten problem solutions typically contain few cross-outs. Creating a classifier to accurately identify rare cases can be challenging. To create an effective classifier to distinguish cross-outs from other objects, we developed 10 special-purpose features. These features are listed in Table 3.1. We used them to train an Adaboosted C4.5 decision tree.

Feature Name	**Description**
SX_{BBW}	Width of the minimum bounding box
SX_{BBH}	Height of the minimum bounding box
SX_{DENS}	Compactness of the stroke
SX_{ST}	Straightness of the current stroke
SX_{POX}	Binary feature – is the stroke part of a cross
SX_{POP}	Binary feature – is the stroke part of a set of parallel lines
SX_{ROCA}	Ratio of the area which the stroke covers other strokes which were drawn earlier
SX_{ROCBA}	Ratio of the area which the stroke covered by other stroke which were drawn later
SX_{UH}	Average height of the underlying strokes
SX_{T2FU}	Time to the first underlying stroke

Table 3.1: Features used by classifier for identifying cross-outs.

The first three features are also used by Peterson's method. These characterize the size and compactness of a stroke. Bounding Box Width (SX_{BBW}) and Height (SX_{BBH}) are properties of the minimum, coordinate-aligned bounding box of the stroke. The Ink Density (SX_{DENS}) feature measures the compactness of the stroke. Cross-out strokes often have high ink density because their purpose is to mask previously drawn strokes. Ink Density is defined as the ratio of stroke length squared to the bounding box area:

$$SX_{DENS}(i) = \frac{L(i)^2}{BBA(i)} \tag{3.1}$$

where $L(i)^2$ is square of the arc length of the ith stroke and $BBA(i)$ is its bounding box area.

As the name suggests, Stroke Straightness (SX_{ST}) is a measure of the straightness of a pen stroke. It is defined as the ratio of the arc length of the stroke and the Euclidean distance between its endpoints:

$$SX_{ST}(i) = \frac{L(i)}{ED(i)} \tag{3.2}$$

where $ED(i)$ is the Euclidean distance between the two endpoints of the ith stroke.

The next two features characterize the geometric relationship between a set of consecutive strokes. The Part of Cross feature (SX_{POX}) indicates whether or not the stroke forms a cross with either the previous or next stroke. To form a cross, both strokes must be approximately straight lines that intersect near their midpoints. More specifically, the strokes must have a straightness of at least 0.7. Additionally, for each stroke, the distance between the intersection point and its midpoint must be less than 10% of its arc length. For example, the two pen strokes in Figure 3.2 do intersect near their midpoints, and thus form a cross. However, the strokes in Figure 3.3 do not intersect near their midpoints and are not considered to form a cross.

The Part of Parallel feature (SX_{POP}) indicates whether the stroke belongs to a set of parallel lines. Such lines must be drawn consecutively and must be relatively straight (straightness ratio of at least 0.7).

As pen strokes are rarely, if ever, perfectly straight, our definition of parallel is approximate. For each point on a stroke, the program calculates the minimum distance to a point on the other stroke. If all of these minimum distances are about the same,

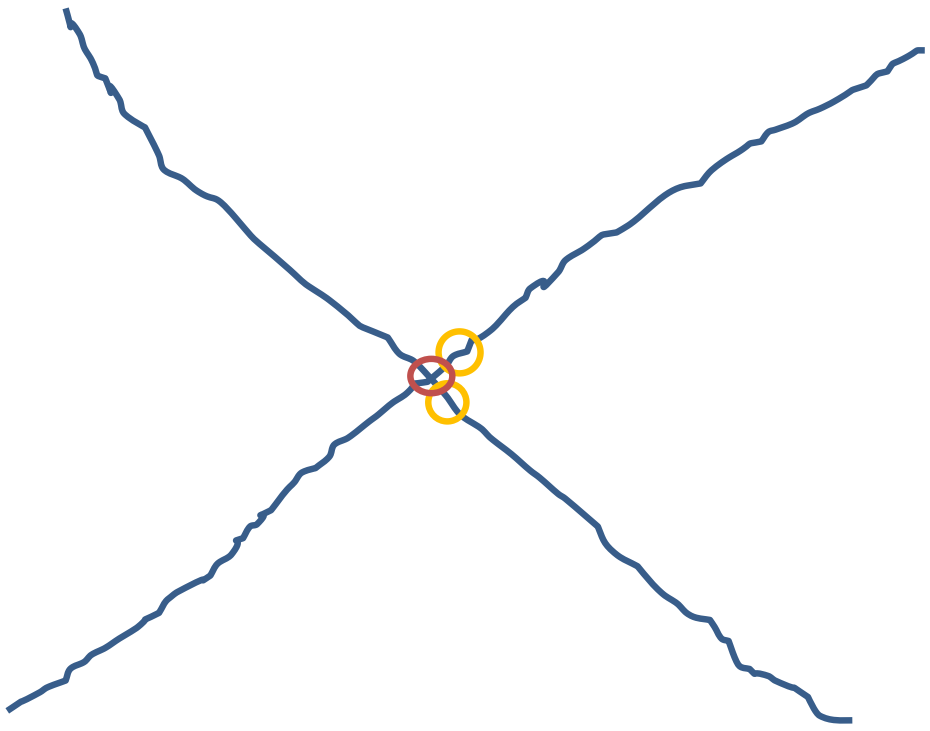


Figure 3.2: Example of strokes that form a cross. The distances between the intersection point (red circle) and the midpoint of each stroke (yellow circles) are less than 10% of the stroke lengths.

the program then considers the two lines to be parallel. Consider, for example, Stroke A and Stroke B in Figure 3.4. The program calculates the average of the minimum point-to-point distances from both A to B and B to A. (The green arrows in Figure 3.4 are the minimum point-to-point distances from A to B.) If all of these distances are between 50% and 150% of the average, the strokes are considered to be parallel.

By their very nature, cross-outs are drawn over other strokes, and typically other strokes are not subsequently drawn over a cross-out. We use two features to characterize this (Figure 3.5. The first is the *Ratio of Covered Area* feature. To compute

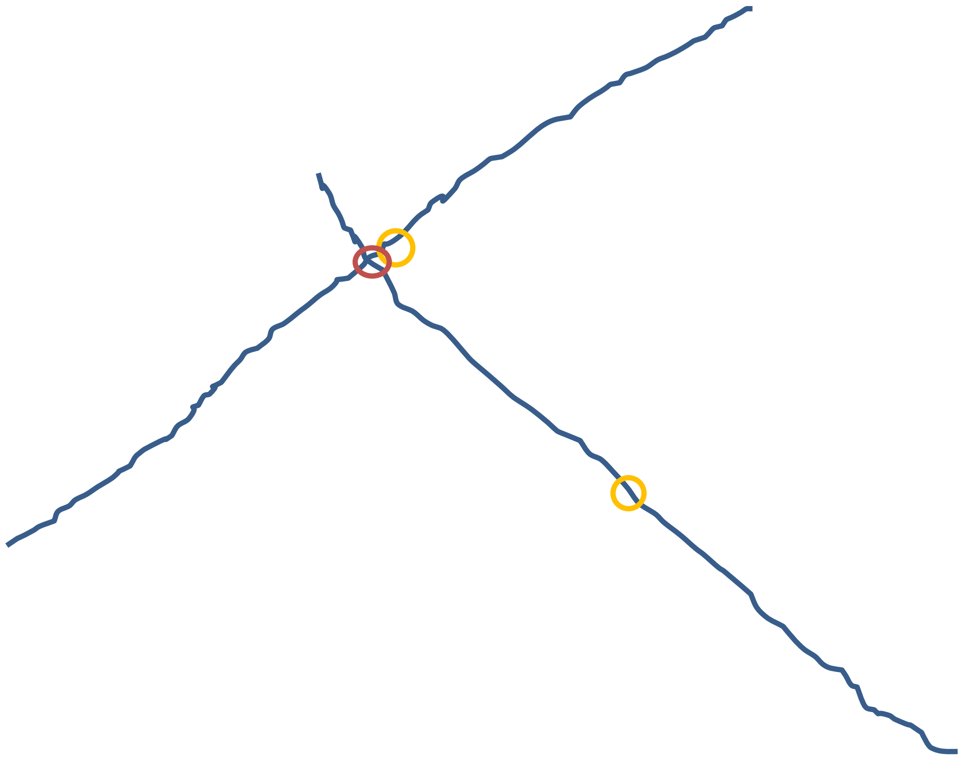


Figure 3.3: Example of strokes that do not form a cross. The distance between the intersection point (red circle) and the centroid (yellow circle) of one of the strokes is greater than 10% of the length of the stroke.

this feature for a candidate cross-out stroke, the program first identifies all strokes that were drawn before the candidate and whose bounding boxes intersect the bounding box of the candidate. This set of strokes comprise the "underlying strokes." *Ratio of Covered Area* is the fraction of the candidate's bounding box that intersects with the bounding box of the underlying strokes.

The second feature is the *Ratio of the Area Covered by Other Strokes* feature. This feature is similar to the first by considering the "overlying strokes." This is the set of strokes that were drawn after the candidate and whose bounding boxes intersect

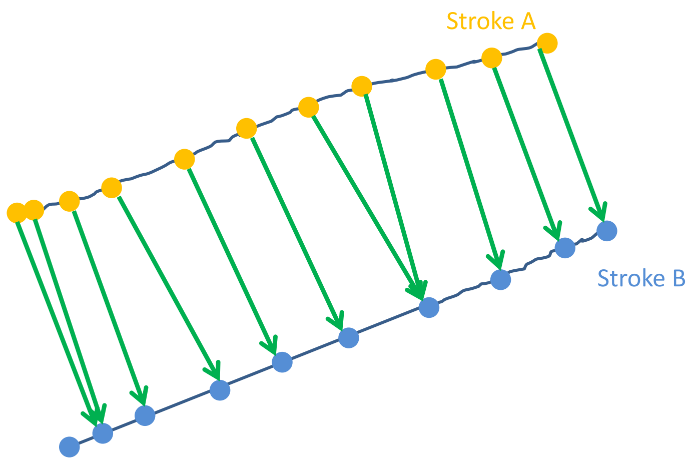


Figure 3.4: Identification of parallel strokes. Yellow circles are sample points in stroke A. Blue circles are sample points in stroke B. Green arrows are the minimum distances from points on stroke A to points on stroke B.

its bounding box. *Ratio of the Area Covered by Other Strokes* is the fraction of the candidate's bounding box that intersects with the bounding box of the overlying strokes.

It is common for students to cross-out an equation with a single horizontal line as illustrated in Figure 3.6. As the bounding box of a horizontal line often has little area, it can be difficult to detect this kind of cross-out by the two area ratio features. The *Average Height of the Underlying Strokes* feature is designed for this situation. As the name suggests, this feature is the average height of underlying pen strokes. Recall that pen strokes for the characters in an equation tend to be smaller than those in a

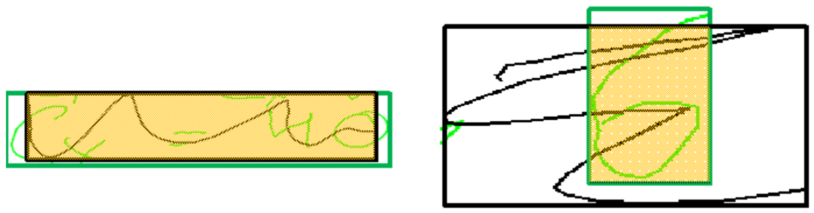


Figure 3.5: The *Ratio of Covered Area* feature (Left) and the *Ratio of the area covered by other strokes feature* (Right). The yellow area indicates overlapped area.

diagram.

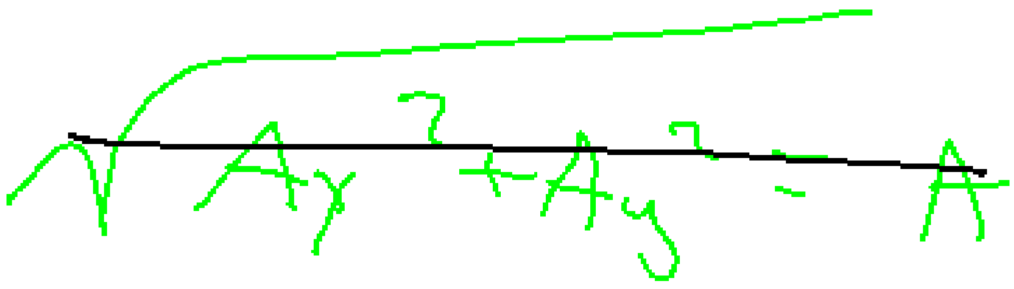


Figure 3.6: Example of a horizontal cross-out line (black). The underlying strokes are in green.

The last feature captures the temporal relationship of a cross-out stroke with its underlying strokes. The *Time to First Underlying* feature is the elapsed time between the first underlying stroke and the candidate cross-out stroke.

3.2 Preliminary Recognition

Our goal at this stage is to compute the probability that a pen stroke is an alpha-numeric character or an arrow. This information will help identify regions containing either equations or diagrams. As shown is Figure 3.7, free body diagrams typically contain arrows which represent forces. Free body diagrams also contain some alphabetic characters which are used as force labels. However, by their very nature, equations typically contain far more alpha-numeric characters than do free body diagrams. We perform the first stage of preliminary recognition with a general-purpose image-based recognizer [39]. In the second stage, we then use two domain-specific recognizers and a special-purpose arrow recognizer.

To perform complete character recognition, it is first necessary to perform grouping so that multi-stroke symbols can be identified. However, the results of stroke-classification are needed to perform grouping. This is a chicken-and-egg problem. Fortunately, it is unnecessary to accurately recognize all of the symbols at this stage.

We use the image-based recognizer because of its tolerance for over-stroking, which is common in free form drawing and writing. This recognizer is also capable of recognizing multi-stroke characters. (While we do not rely on this capability for preliminary recognition, we do use it later for final recognition.) During preliminary recognition, we apply the image-based recognizer to single pen strokes to identify single-stroke characters. Many characters are commonly drawn with a single stroke such as "C" and "O", but some characters require multiple strokes. Thus, at this stage in the processing, we will identify only a subset of the characters.

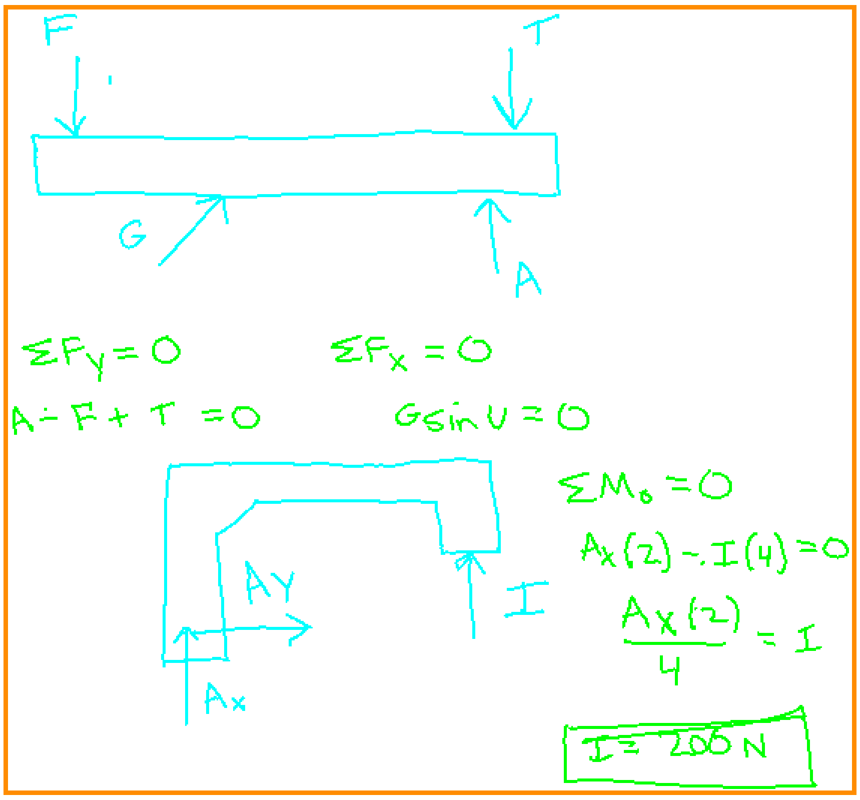


Figure 3.7: Arrows are usually in free body diagrams while characters are usually in equations.

As described below, we built simple heuristic, domain-specific recognizers for two important symbols: "+" and "=". These symbols are formed from straight lines and can be easily identified by their geometric properties. The heuristics are applied to the output of the image-based recognizer.

Arrows can be drawn with a wide variety of aspect ratios. For example, some arrows are long with small heads, while others may be short with large heads. Also arrows can be drawn in arbitrary directions. Because of this wide range of variation, it is inefficient to use template-based recognizers to identify them. Therefore, we use a special-purpose recognizer, described below, to recognize arrows. Our arrow recognizer can handle multi-stroke arrows drawn in any direction.

We compute a number of features from the output of these recognizers. For example, we compute the average size of strokes that have been positively identified as alpha-numeric characters. This gives a good estimate of the expected size of a character in the sketch. Strokes that are much larger than this average are likely to be elements of a free body diagram.

3.2.1 Image-based Recognizer

The image-based recognizer [39] is a trainable recognizer based on a multi-layer recognition scheme. It represents symbols as binary templates and then compares and ranks definition symbols according to their similarity to the unknown symbol using four different classifiers. These include the Hausdorff Distance, modified Hausdorff Distance, Yule Coefficient, and Tanimoto Coefficient. Typically the results of these four classifiers are combined with a voting scheme. To reduce computational cost, we use only the modified Hausdorff distance.

The image-based recognizer is designed to be insensitive to the orientation of the symbol. As we are recognizing letters and numbers which typically appear in a fixed orientation, we do not use the rotation processing portion of the recognizer. This further reduces the computation cost of recognition.

We trained the image-based recognizer with the symbols from Data Set A described in Section 6.1. Because this data set contained very few lower case letters, we excluded them from the training. Our training symbols include all 10 digits, 22 capital letters, eight arrows, and five mathematical symbols ("$-$", "Σ", "/", "(", ")"). For convenience, we refer to all 45 symbol classes, except arrows, as “characters.” We selected 10 training examples for each type of symbol, for a total of 450 training templates. As our goal is only to determine if a stroke is or is not a character, the precise identification of a pen stroke is not important. Thus, we combined eight symbol types into four. For example, “I” and “1” are represented by the same templates. Likewise, the pairs “B” and “8”, “O” (letter) and “0” (number), and “S” and “5” are combined. The eight types of arrows included arrows drawn in the four cardinal directions and the four inter-cardinal directions.

The output of the recognizer is the modified Hausdorff distance from the unknown to each of the 450 training templates. We then sort this in increasing order of distance. If the top two templates (the two with the shortest distances) represent the same symbol, and that symbol is not an arrow, the unknown is assumed to be a character for the purposes of computing the average character size. If the top two templates represent different symbols, the recognition results are uncertain and the unknown is not include in the calculation of the average character size. In either case, if the top-ranked template is not an arrow, the modified Hausdorff distance to that template is used as an indication of the probability that the unknown is a character – the smaller the distance, the greater the probability. If the top-ranked template is an arrow, the probability that symbol is a character is taken to be zero, and the Hausdorff distance is replaced with the value 10,000.

The trained classifier achieved 90% accuracy on 3087 examples of the 45 symbols classes from Data Set A. Here accuracy is defined as assigning the unknown character to the correct one of the 45 symbol types (i.e., both the top and penultimate templates are the correct class). For example, an “A” recognized as an “A” is considered correct. Similarly, an “S” recognized as a “5” is considered correct.

3.2.2 Arrow Recognizer

The image-based recognizer is trained to identify only single-stroke arrows drawn in one of eight directions. Also, the training examples include only a small range of aspect ratios. These limitation prevent the recognizer from correctly identifying many types of arrows. As a remedy we developed a special-purpose arrow recognizer that can identify either one or two-stroke arrows drawn in arbitrary directions with varying aspect ratios.

Arrows can be drawn with a wide variety of aspect ratios. For example, some arrows are long with small heads, while others may be short with large heads. Also arrows can be drawn in arbitrary directions. Because of this wide range of variation, it is inefficient to use template-based recognizers to identify them. Therefore, we use a special-purpose recognizer, described below, to recognize arrows. Our arrow recognizer can handle multi-stroke arrows drawn in any direction.

Table 3.2 describes the types of arrows that exist in Data Set A. 76% of the arrows have straight shafts (Figure 3.8) while only 7% are curved (Figure 3.9). 41% of the arrows are drawn with a single stroke, while 59% are drawn with two. A total of 17% of the arrows are drawn either with an unusual shaft (e.g., a squiggle) or an unusual head (e.g., “ ”). We designed our recognizer to handle both single-stroke and two-stroke arrows with strait shafts. This comprises 76% of the arrows in Data Set A.

To recognize a single-stroke arrow, the program first orients the candidate shape so that it is pointing upward. Then the image is divided at roughly the widest point (see below for more precise description) to from the top and bottom of the image. A recognizer attempts to identify the top as an arrow head and the bottom as a straight shaft. If they are both identified as such, and they have the appropriate relative arrangement, they are classified as an arrow.

To recognize a stroke as a portion of a two-stroke arrow, the program attempts to identify it as an arrow head. If it is identified as such, the program attempts to recognize the previously drawn stroke as a shaft. If it is a shaft, and the relative arrangement of the shaft and head are appropriate, the two strokes are classified as an arrow. If this previously drawn stroke does not form an arrow, then the subsequently drawn stroke is examined in the same way.

	Straight Arrow	**Curved Arrow**	**Others**
Single stroke	31%	3%	7%
2 strokes	45%	4%	10%
3 strokes	0.30%	0%	0%
Total	76%	7%	17%

Table 3.2: Distributions of each type of arrow.

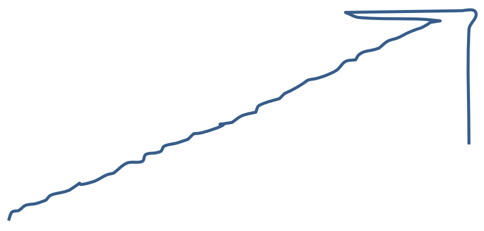


Figure 3.8: Example of a straight arrow.

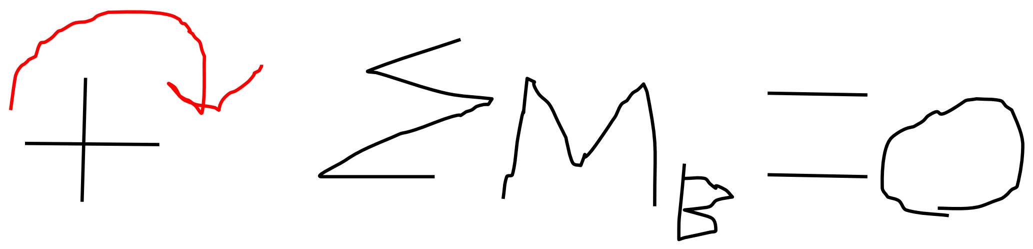


Figure 3.9: Curved arrow (red) in moment equation.

Orienting a Candidate Single-Stroke Arrow

To orient a candidate single-stroke arrow, the program first uses a simple approach to find corners on the pen stroke (Figure 3.10b). A corner is defined as a sample point on the pen stroke that forms an acute angle with the previous and subsequent samples points. More precisely, if the three points form an angle less than 90^0 and greater than 15^0, the point is a corner point. The distance between all pairs of corner points is computed. The image is then rotated so that the two most distant corner points lie on a vertical line (Figure 3.10c). Next, the image is split vertically into two equal halves. The density of samples points in each half is computed, and the image is rotated so that the most dense half is at the top (Figure 3.10d).

Once the image has been oriented, the widest points are identified (Figure 3.11e). These points are the ones that lie on the vertical sides of a coordinate aligned bounding box. The image is split vertically at the lowest of these points (Figure 3.11f). The portion of the image above the split is considered a candidate arrow head. If it is recognized as an arrowhead, then the portion of the image below the split is considered a candidate shaft. Otherwise, the entire stroke is considered a candidate arrowhead.

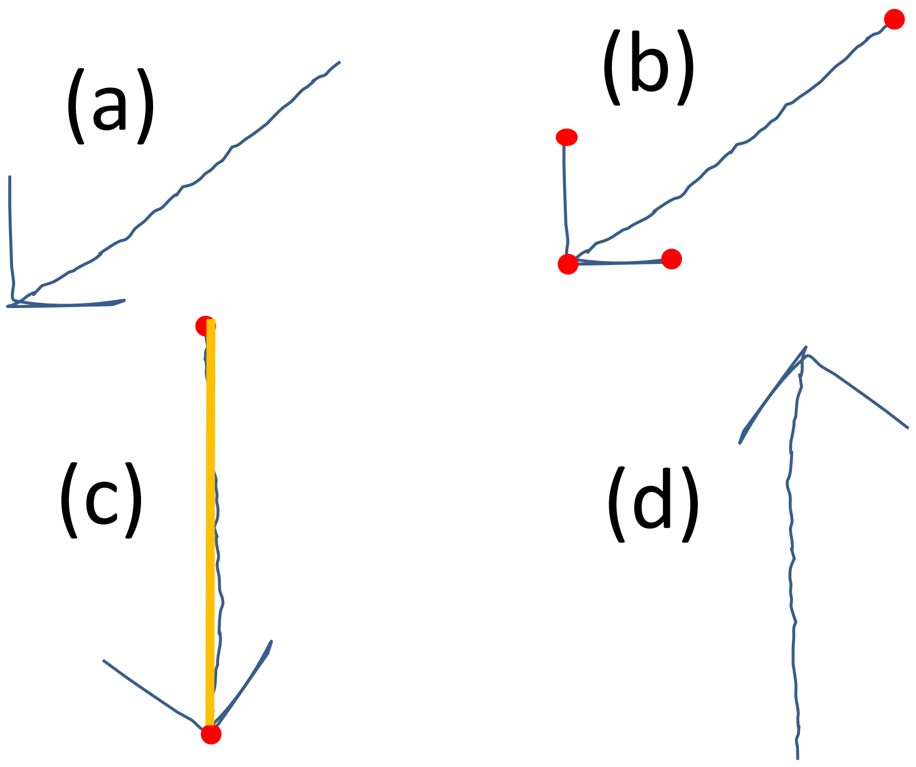


Figure 3.10: Steps in orienting a Candidate Single-Stroke Arrow. Red circles are the corners. Yellow line is the longest distance between two corners.

Recognizing Arrowheads

Our arrowhead recognizer computes a boundary for the candidate stroke that is somewhat related to a convex hull. While the convex hull of a pen stroke may contain both points belonging to the stroke and points which do not lie on the stroke, our boundary contains only points that belong to the stroke. We approximate the boundary with least squares lines and examine those lines to determine if they form either a triangle-shaped arrowhead or a "V"-shaped arrowhead.

To construct our boundary, we use a line-scanning algorithm to determine which sample points on the pen stroke can be viewed from each of the four cardinal

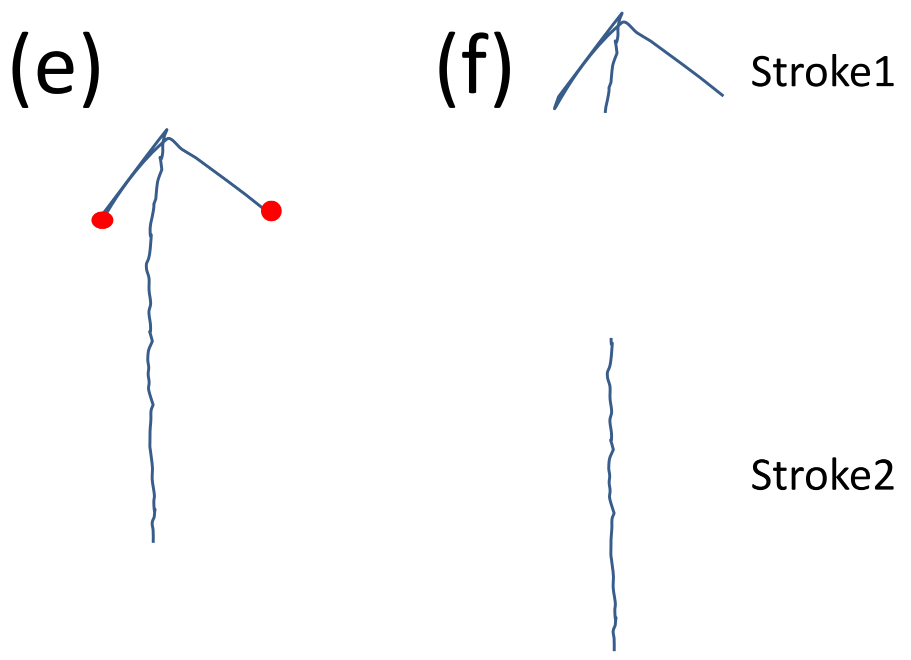


Figure 3.11: Steps in orienting a Candidate Single-Stroke Arrow. Red circles are the widest points.

directions (Figure 3.12 parts (b), (d), (f), and (h)). For example, Point "A" in Figure 3.12 can be viewed from the right, while points "B," "C," and "D" cannot. We then compute four additional views which are the intersections of the views from adjacent cardinal directions. For example, part (a) of Figure 3.12 shows the points that are viewable from both the left and the top.

The program then constructs a single least-squares line for each of the eight views. Consider the viewable points in Figure 3.14. These points are grouped into sets comprising consecutive sample points. In this case, there is one large set of consecutive points, and many smaller sets. If the largest set contains at least 80% of the viewable points, a least squares line is constructed from the set. Otherwise, no least squares line is constructed for this view. This threshold is used so that a line will not be constructed for a curved shape. For example, for the circle in Figure 3.15, there are two large sets of consecutive viewable points, with neighbor containing more than 80%.

If least squares lines are computed for fewer than three views, the candidate is rejected as an arrowhead. Otherwise, the three lines with the least error of fit are identified. If the angles between these three lines are all between 15^o and 165^o, the candidate is classified as an arrowhead.

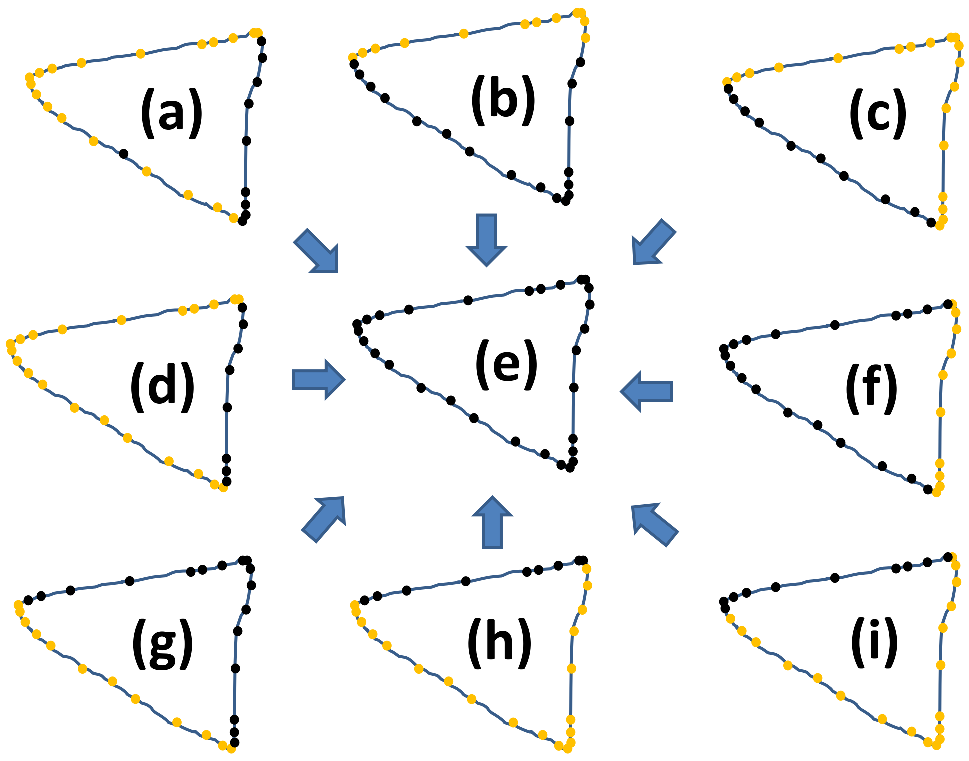


Figure 3.12: Points(Yellow) can be seen from the eight directions.

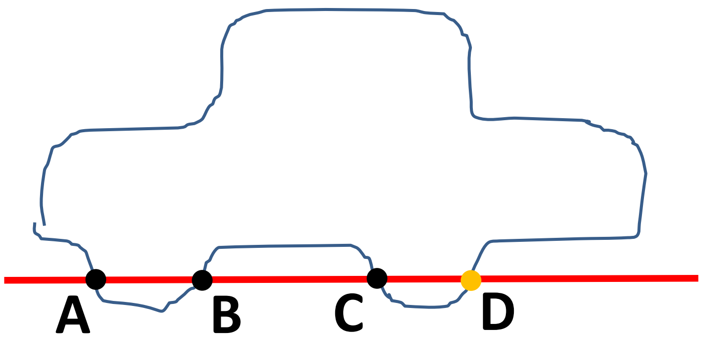


Figure 3.13: Points(Yellow circle) can be seen from the right.

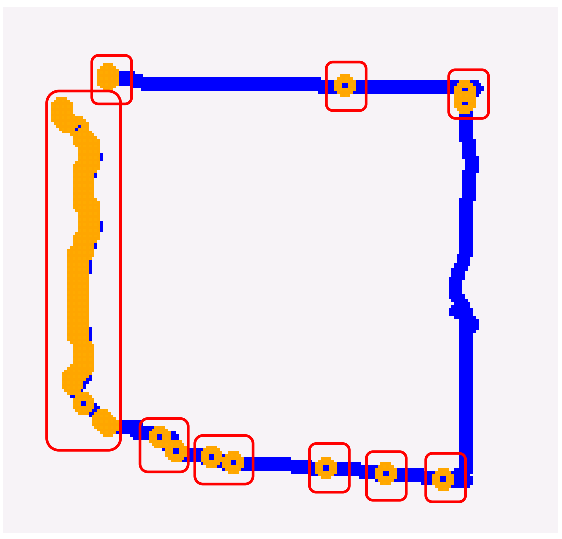


Figure 3.14: Point groups on a rectangle(Red rectangles).

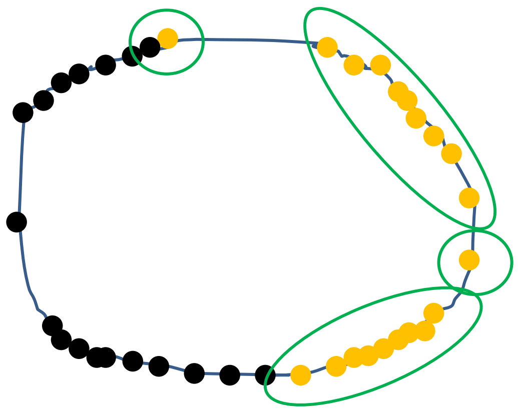


Figure 3.15: Points groups on a circle. Yellow circles can be seen from right direction. Green circles are the point groups.

Recognizing Arrow Shafts

For a pen stroke to be the shaft of an arrow, the stroke must form a straight line. To determine if a stroke is straight, we compute the straightness ratio. This is computed as the ratio of the arc length of the stroke to the distance between the endpoints. If this ratio is greater than 0.7, the stroke is classified as a valid shaft.

Arrowhead and Shaft Alignment

If one of the ends of a shaft lies within the bounding box of an arrowhead, the two satisfy the geometric requirements to be an arrow. However, they are classified as such only if the two belong to the same pen stroke or are two consecutive strokes.

3.2.3 Heuristic Recognizers for Two-Stroke Math Symbols

Two of the most frequent mathematical symbols, “+” and “=”, cannot be recognized by the image-based recognizer because we apply it only to individual strokes. Thus, we identify these two symbols as a post-process.

If two consecutively-drawn strokes that are recognized as a “1” and a “-”, they are examined further to determine if they form a “+”. To do so, the two must intersect, and the intersection point must be roughly in the middle of each stroke. The latter condition is satisfied if the distance from the intersection point to each end of the stroke is at least 10% of the stroke’s length. Finally, the heights and widths of the bounding boxes of the two strokes must be no larger than twice the average height of the characters in the sketch. (Note that this average can be computed once the image-based recognizer has been applied.) If all of these conditions are satisfied, the pair of strokes is classified as a “+”.

An “=” is identified in an analogous fashion. In equal is identified as a pair of consecutively drawn pen strokes that were both recognized as a “-” by the image-based recognizer. The two strokes must not intersect each other and both the height and width of the bounding box of the two must be less than twice the average width of a character.

3.3 Stroke Classification

The task of the stroke classifier is to assign each pen stroke to one of two semantic classes: free body diagram or equation. (Recall that cross-outs are identified by a separate classifier.) The stroke classifier is an AdaBoosted decision tree implemented in WEKA [40] with the following settings: AdaBoostM1 with 10 iterations, a seed of 1, no resampling, and a weight threshold of 100. The base classifier is a J48 decision tree (an implementation of C4.5) with the following settings: pruned, confidence value of 0.25, and minimum number of instances in a leaf of 2.

Our classifier employs a set of 10 features, several of which are computed from the results of the preliminary recognition process. It also uses the 27 features from Stahovich et al.'s [1] feature-based classification algorithm. As described in Section 7.1.2, combining these two feature sets results in high accuracy. However, our set of 10 features alone outperforms their 27 alone.

3.3.1 Our Stroke Classification Features

Feature Name	**Description**
L_{NDT}	Normalized Drawing Time
L_{PC}	Modified Hausdorff Distance
L_{LS}	Binary feature: Is long Stroke?
L_{NCR}	Number of characters in the range
L_{NLR}	Number of long strokes in the range
L_{NAR}	Number of arrows in the range
L_{NU}	Number of underlying strokes
L_{UD}	Density of underlying strokes
L_{D2N}	Direction to the next stroke
L_{NS}	Number of segments

Table 3.3: Our 10 features for single-stroke classification.

Our 10 features for distinguishing free-body diagram and equation strokes are listed in Table 3.3.

We have found that a stroke that was drawn earlier in the sketch is more likely to be a free body diagram. Therefore, we compute the normalized time, T_{NDT}, for each stroke:

$$T_{NDT}(i) = \frac{T_S(i) - T_S(0)}{T_S(n) - T_S(0)} \tag{3.3}$$

where $T_S(i)$ is the start draw time of the i^{th} stroke in a sketch, 0 is the first stroke, and n is the last stroke. The normalized drawing time ranges from 0 to 1.0, with the former corresponding to the first stroke drawn and the latter corresponding to the last one drawn.

Five features are computed from the results of preliminary recognition. The feature L_{PC} is inversely related to the probability that a stroke is a character. If the stroke was identified as a character by the image-based recognizer during preliminary recognition, L_{PC} is assigned the value of the Hausdorff distance between the stroke and the best-matching template. However, if the stroke was identified as an arrow by the image-based recognizer, L_{PC} is set to 10,000.

The binary feature L_{LS} indicates if the stroke is a *long* stroke. A long stroke is defined as one that is at least three times taller or five times wider than the average character size. Long strokes are more likely to belong to free body diagrams than equations. We obtained these thresholds using exhaustive search of a small search space. We considered five values for the thresholds for these features ranging from one to five in steps of one.

The features L_{NCR} and L_{NLR} are the number of nearby characters and long strokes, respectively. Here, two strokes are near each other if the minimum point-to-point distance between them is less than twice the average character height. In the example in Figure 3.16, there are six character strokes near the letter “a”. The circle in the figure provides an estimate of the “nearby” region. (The actual region is the locus of points that are no father than twice the average character height from a point on the “a”.)

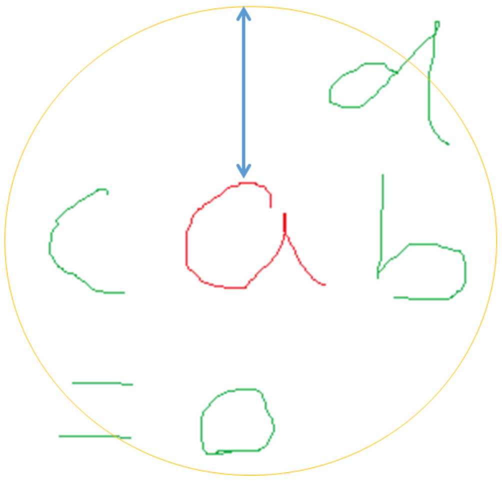


Figure 3.16: Calculating the number of characters near the red stroke. The strokes in green are within the threshold distance of twice the average character height. The circle is a simple approximation of the region containing “nearby” strokes.

Arrows occur frequently in free body diagrams as forces. Thus, the proximity of a large number of arrows may be indicative of a free body diagram. The feature L_{NAR} is the number of arrows that intersect the bounding box of the stroke.

The strokes in an equation typically do not overlap each other, while those in a free body diagram often do. For example, Figure 3.17 shows a pen stroke from a free body diagram which has numerous underlying pen strokes. L_{NU} is the number of underlying strokes while L_{UD} is the underlying stroke density:

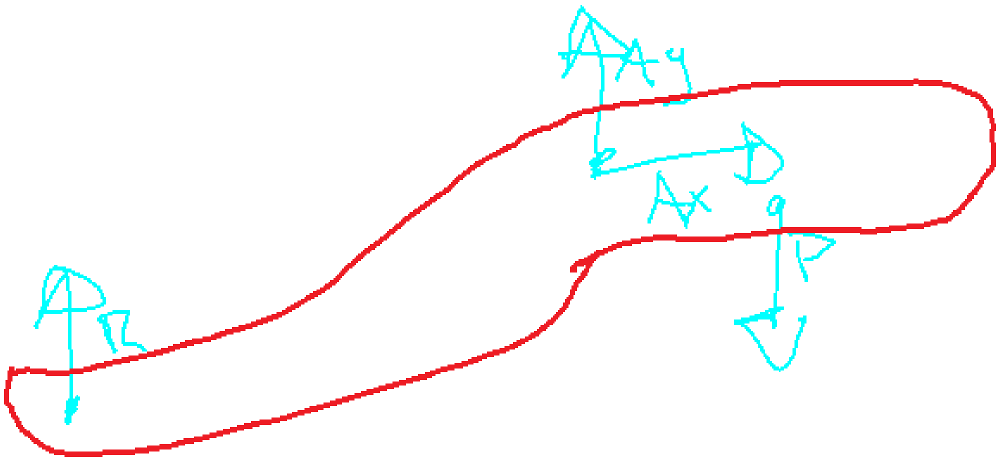


Figure 3.17: The strokes in cyan are the underlying strokes of the red stroke.

$$L_{UD} = \frac{L_U^2}{A_{BB}} \tag{3.4}$$

where L_U^2 is the square of the sum of the arc lengths of the underlying strokes and A_{BB} is the area of the stroke in question.

Equation strokes (in English) are typically drawn from left to right, while free body diagrams can be drawn in any fashion. The feature L_{D2N} is useful for capturing this distinction. The feature is defined as the angle from the centroid of the current stroke to the centroid of the next stroke.

This last feature, L_{NS} is the number of segment points (corners) in the pen stroke calculated with Herold and Stahovich's [41] Classyseg technique. Figure 3.19 shows examples segmentation. For example, the letter "E" has six segment points while a circle has none.

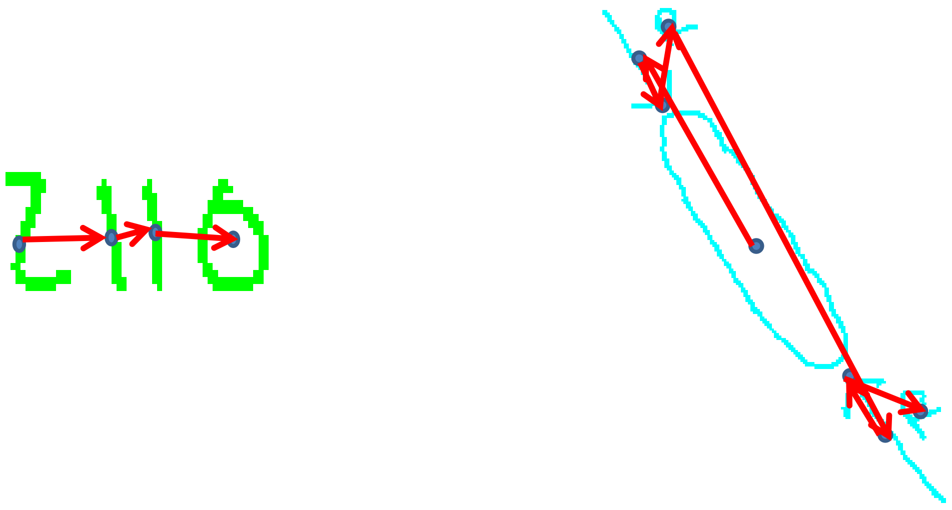


Figure 3.18: (Left) Red arrows show direction from the centroid of one pen stroke to the next in an equation. (Right) Red arrows show direction from the centroid of one pen stroke to the next in a free body diagram.

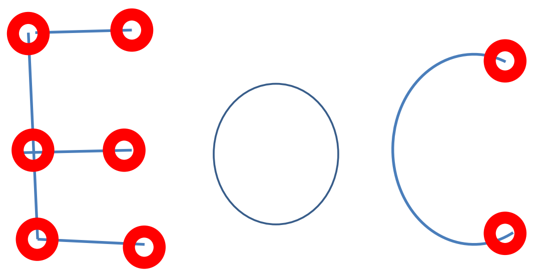


Figure 3.19: Example of segments found in a stroke.

3.3.2 Stahovich's Features

Our stoke classifier also uses the 27 features from the stroke classifier in [13]. These features are listed in the Table 3.4. (See [13, 1] for more details about these features.) The first four features describe the size of a stroke. For example, P_{AL} is the arc length of the stroke. These features are normalized by their average values in the sketch. The next two features describe the position of a stroke. The next eight features describe the shape of a stroke. For example, P_{EPR} measures the degree to which the stroke forms a closed path. It is defined as the Euclidean distance between the endpoints of the stroke divided by the arc length. The next four features describe temporal properties of a stroke. The remaining features characterize the geometric and temporal relationship between the stroke and other strokes in the sketch.

Category	**Feature Name**	**Description**
Size	P_{BBW}	Width of the minimum bounding box
	P_{BBH}	Height of the minimum bounding box
	P_{BBA}	Area of the minimum bounding box
	P_{AL}	Total length of the stroke
Location	P_{D2LR}	Minimum distance between the stroke and the closer of the left or right edge of the canvas
	P_{D2TB}	Minimum distance between the stroke and the closer of the top or bottom edge of the canvas
Shape	P_{EPR}	Degree to which the stroke forms a closed path
	P_{SE}	Binary version of End Point Ratio
	P_{SI}	Number of times the stroke intersects itself
	P_{SOC}	Sum of all curvature values along the stroke
	P_{SOCABS}	Sum of all curvature values along the stroke
	P_{SOC2}	Sum of all curvature values along the stroke
	P_{SOCR}	Sum of all curvature values along the stroke
	P_{ID}	Compactness of the stroke
Drawing Kinematics	P_{APS}	Average Speed while drawing
	P_{MaxPS}	Maximum instantaneous speed for the pen
	P_{MinPS}	Minimum instantaneous speed for the pen
	$P_{DMaxMin}$	Maximum pen speed minus minimum pen speed
	P_{T2D}	Time from pen down to pen up
Geometric Relations	P_{ILL}	Count of endpoint-to-endpoint intersections with other strokes
	P_{IXX}	Count of midpoint-to-midpoint intersections with other strokes
	P_{IXL}	Count of midpoint-to-endpoint intersections with other strokes
	P_{ILX}	Count of endpoint-to-midpoint intersections with other strokes
	P_{CP}	Binary feature - does the stroke help form a closed path
	P_{IP}	Binary feature to indicate whether the stroke is enclosed inside a closed path
Temporal Relations	P_{T2P}	Time between the end of the previous stroke and the beginning of the current stroke
	P_{T2N}	Time between the end of the current stroke and the beginning of the next stroke

Table 3.4: Stahovich's features for single-stroke classification.

Chapter 4

Interpreting the Text

4.1 Grouping Equations

Once the equation pen strokes have been identified, it is necessary to group them into individual equation groups (Figure 1.1). We explored the use of Stahovich, Peterson, and Lin's (2014) general-purpose pen stroke grouping algorithm for this task. Although the algorithm achieved high accuracy, we were able to develop a special-purpose technique that was both more efficient and more accurate for our task. This technique employs three features and is trained with J48 tree implemented in WEKA. To group pen strokes into equations, the classifier is applied to every pair of strokes to determine if the pair belongs to the same equation. Pairs of grouped strokes chain together to form equation groups.

4.1.1 Pairwise Classifier

Our approach uses pairwise classifiers to identify pairs of strokes that belong to the same equation. It considers three features and has two classes (*Join* and *NoJoin*). The three features describe properties of the bounding boxes of a pair of pen strokes. G_{YOR} describes the vertical overlap of the bounding boxes of two strokes defined as:

$$G_{YOR} = max(\frac{y_O}{y_A}, \frac{y_O}{y_B}) \tag{4.1}$$

where y_A and y_B are the heights of the two bounding boxes, and y_O is their vertical overlap as shown in Figure 4.1. G_{MMD} is related to the Manhattan distance defined as:

$$G_{MMD} = x_D - y_O \tag{4.2}$$

where x_D is the horizontal distance between two bounding boxes. If the bounding boxes overlap horizontally, $x_D = 0$. G_{2BOR} is the ratio of the area of the intersection of the bounding boxes to the area of their union. However, before computing this ratio, the bounding boxes are expanded if they are too small. If the height of a bounding box is less than the median bounding box height, the box is expanded to that height. The width is adjusted analogously. To emphasize the relationship between two strokes in an equation, we double the width of the bounding box (Figure 4.2). Strokes in an equation are usually written horizontally. So, here we only double the width of the bounding box. The medians are computed separately for each problem solution. We then train the model using WEKA's (Hall et al. [40]) J48 tree technique with default parameter values.

4.1.2 Chain stroke pairs

Our equation classifier uses two join classes (Join or NoJoin) to identify pairs of strokes that belong together. After these pairs have been identified, we groups joined pairs that have a stroke in common, to form a small stroke group. In the same way, if two groups have strokes in common, we chain the two groups to form a larger group. For example, the strokes groups are chained, as illustrated in Figure 4.3. Here, the classifier identifies that group AB and CD should be in the same group, while all other groups should not. The chainer then forms two groups. The first contains groups A and B because the group A and B have a character "=" in common. The second contains groups C and D because the group C and D have a character "T" in common. The new strokes group A' and B' don't have strokes in common, so we don't chain the two groups. This chaining technique is applied to all grouping algorithms in this thesis.

4.1.3 Merging the rest of the small groups

Sometimes subscripts are not properly grouped with an equation (Figure 4.4. As a remedy, small equation groups containing less than five strokes are merged with the nearest equation group if the distance between that group and the nearest equation group is less than a medium character height.

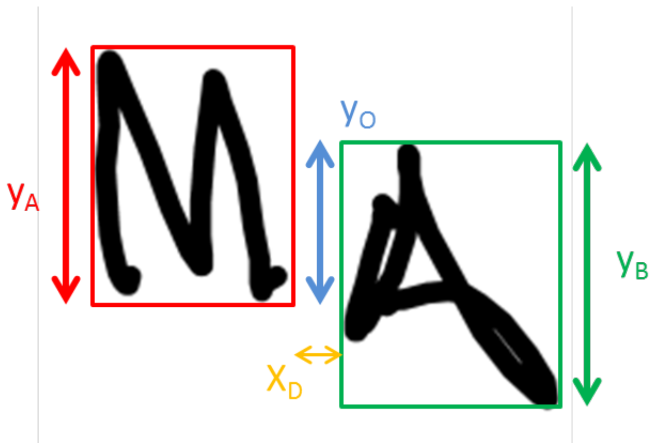


Figure 4.1: y_A,y_B, y_O, and x_D.

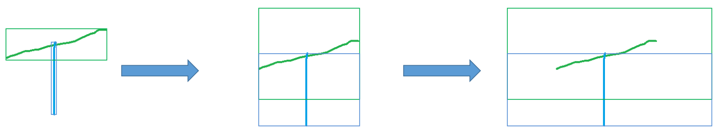


Figure 4.2: Expanded bounding box.

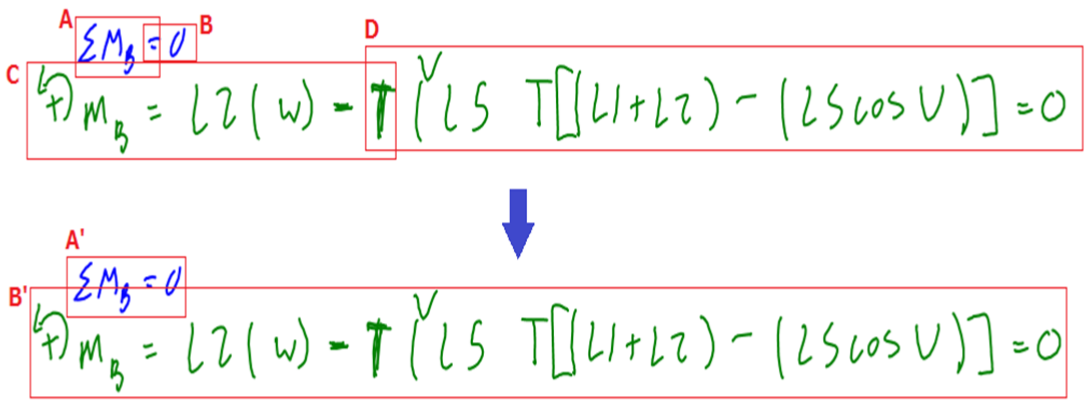


Figure 4.3: The chainer forms two groups which have a stroke in common.

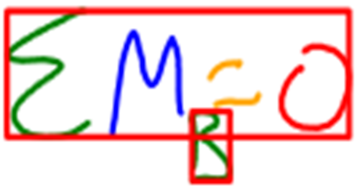


Figure 4.4: Subscripts on equations.

4.2 Grouping Characters

Once the pen strokes have been grouped into equations, it is necessary to group the strokes into individual characters so they can be recognized. We do this using a variation on the equation grouper. Here we use only two features, G_{BOR} and G_{XOR}. G_{BOR} is similar to G_{2BOR} but does not double the width of the bounding box. G_{XOR} is similar to G_{YOR} but considers horizontal overlap of the bounding boxes defined as:

$$G_{XOR} = max(\frac{xO}{xA}, \frac{xO}{xB}) \tag{4.3}$$

where xA and xB are the widths of the bounding boxes of the two strokes, and xO is their horizontal overlap. As before, we then trained the model using WEKA's J48 tree technique. The features are computed for every pair of strokes in an equation group. Grouped pairs can chain together to form larger characters.

4.3 HMM

After the individual characters in a solution have been identified, we use Kara and Stahovich's image-based recognizer [39] to recognize them. While this recognizer has high accuracy, recognition errors are still problematic. Some errors are due to variations in writing style. Others result from ambiguity. For example, a lowercase "t" can be confused with a "+" and a zero can be confused with the letter "o". We correct this using a HMM-based approach from (Lee et al. [2]).

A HMM is characterized by the three-tuple $\lambda(A_{ij}, B_j(k), \pi_i)$. A_{ij} is a matrix describing the probability of transitioning from state i to state j, $B_j(k)$ is a matrix describing the probability of observing symbol k in state j, and π_i is a vector describing the probability that the initial state is state i (Rabiner [42]). Our HMM has 50 states including letters, numerical digits, '(', ')', '+', '-', '*', '/', '=', 'sigma', 'theta', 'phi', "co", "si", "cos", and "sin". The last four states are designed to facilitate identifying "sin" and "cos". Our 50 observation symbols match the 50 states. As in (Lee et al. [2]), we compute the A_{ij} matrix based on a simple grammar for legal equations described in Table 4.1. We assume all legal state transitions defined by the grammar are equally likely, with the total probability of such transition summing to 99%. Conversely, we assumed that all illegal transitions from a state are equally likely, with the total probability of such transitions summing to 1%. However, while Lee et al. [2] use heuristics and recognizer accuracy data to create the $B_j(k)$ matrix, we compute our matrix using maximum-likelihood estimation (Rabiner [42]).

During error correction, we treat the output of the image-based recognizer as the observations and the true identity of the characters as the hidden states. We use the Viterbi algorithm (Rabiner [42]) to compute the most likely sequence of hidden states to produce the observations. This sequence is then used as the interpretation of the equation.

S_i	S_{i+1}
=	**digit**, **alpha**, '('
'('	**digit**, **alpha**,'θ','ϕ'
')'	**op**, '='
op	**digit**, **alpha**, '('
digit	**digit**, **op**
alpha_cs	**digit**, **op**, '=', ')'
'c'	**op**, '=', ')',"co"
's'	**op**, '=', ')',"si"
"co"	"cos"
"si"	"sin"
"cos","sin"	'(', 'θ','ϕ'
'Σ'	'f','m'
'θ','ϕ'	**op**, **alpha**,')'

Table 4.1: Equation grammar: Legal transitions. **digit** = digits '0' – '9'; **alpha**= letters; **alpha cs** = letters except 'c' and 's'; op = '+', '-','*', and '/'.

Chapter 5

Extracting Features from Text

Once the equations have been recognized, we compute a variety of features from the text. One feature describes temporal properties of the writing. The remainder describe the frequency of occurrence of various patterns of characters. The complete set of features is listed in Table 5.1.

Features	Description
F_P	No. of long pauses
F_E	No. of equations
F_D	No. of digits
F_L	No. of letters
F_M	No. of mathematical symbols
F_Σ	No. of Σ
F_C	No. of characters
$F_{D/L}$	Ratio of F_D to F_L
$F_{D/M}$	Ratio of F_D to F_M
$F_{L/M}$	Ratio of F_L to F_M
F_U	No. of units
F_{DD}	No. of pattern DD
F_{DM}	No. of pattern DM
F_{DL}	No. of pattern DL
F_{LD}	No. of pattern LD
F_{LM}	No. of pattern LM
F_{LL}	No. of pattern LL
F_{MD}	No. of pattern MD
F_{MM}	No. of pattern MM
F_{ML}	No. of pattern ML
$F_{=D}$	No. of pattern $=D$
F_{DMD}	No. of pattern DMD
F_{DML}	No. of pattern DML
F_{LMD}	No. of pattern LMD
F_{LML}	No. of pattern LML

Table 5.1: Features for characterizing equations. D = digit, L = letter, M = mathematical symbol, "units" are units of measure such as "kg" and "lb".

5.1 Features

5.1.1 Pause Count Features

Cheng and Rojas-Anaya [31] propose two different measures based on the temporal chunk signal. There are *Long Pause Count* (LPC) and *Long Pause Duration* (LPD). *Long Pause Count* (LPC) is the number of inter-stroke pauses longer than a threshold.*Long Pause Duration* (LPD) is the average of the ratio of the difference between the pause duration and baseline to the baseline itself:

$$LPD = \frac{\sum_{i=1}^{N} \frac{PD_i - BL}{BL}}{N} \tag{5.1}$$

where N is the number of inter-stroke pauses in the equation, PD_i is the pause between stroke i and $i + 1$, and BL is the baseline (a threshold). Cheng and Rojas-Anaya [31] calculate LPC and LPD for each individual equation while we calculate these two measures for the equations in each problem solution. They use the pause duration in the participant's name as a baseline and claims that two or three multiples of the baseline is the best threshold for their dataset.

In this thesis, we evaluated not only inter-stroke pauses but also inter-character and inter-equation pauses. We used the median, which is computed for each individual problem solution, as a baseline and explored a range of thresholds between half and 30 times the median. We used our exam dataset to investigate the ability of LPC and LPD to predict student competence. We trained the models using WEKA's ([40]) SVM regression technique (SMOreg) with default parameters. Our search process thus considers 21 thresholds. Figure 5.1 shows the coefficient of determination (R^2) of LPCs and LPDs using the 21 thresholds. Table 5.2 shows that for our data, the median pause with inter-character LPC was the most effective threshold for predicting student performance.

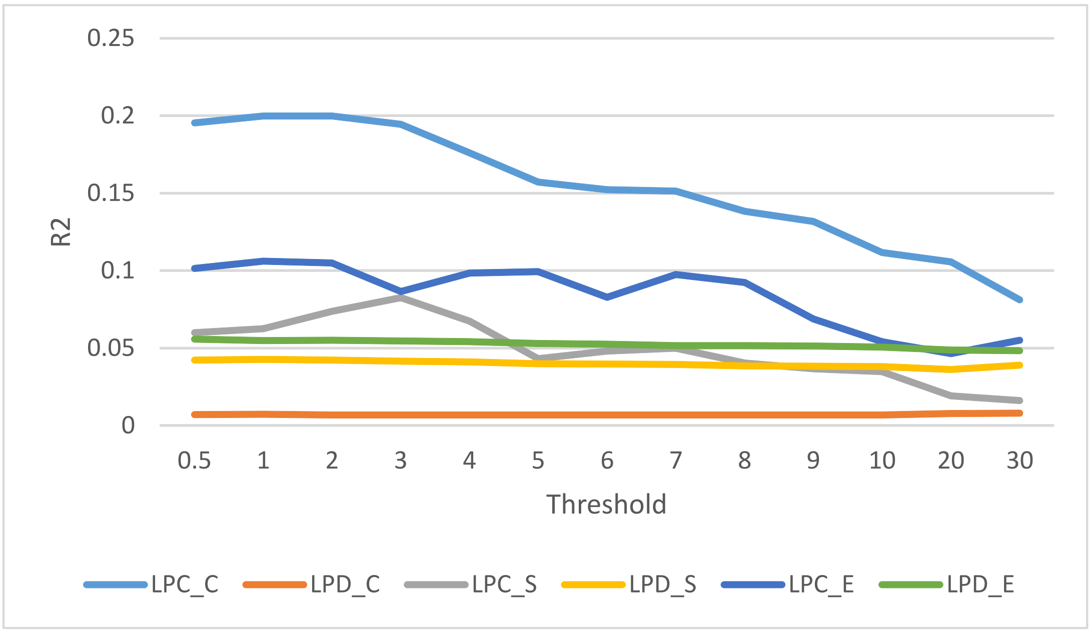


Figure 5.1: Thresholds for LPC and LPD. LPC_C = inter-character LPC. LPD_C = inter-character LPD. LPC_C = inter-stroke LPC. LPD_S = inter-stroke LPD. LPC_E = inter-equation LPC. LPD_E = inter-equation LPD.

As shown in Table 5.2, using one or two times the median of LPC as the threshold was the most effective threshold for predicting student performance. In this thesis, we use the median as the threshold.

The feature F_P is the number of inter-character pauses longer than the median inter-character pause.

Rank	**Type**	**Threshold**	**R2**
1	LPC	1	0.19971768
2	LPC	2	0.199673185
3	LPC	0.5	0.19540232
4	LPC	3	0.194353752
5	LPC	4	0.175949322
6	LPC	5	0.157028346
7	LPC	6	0.152254633
8	LPC	7	0.151372364
9	LPC	8	0.138143552
10	LPC	9	0.131658656

Table 5.2: Top 10 thresholds.

5.1.2 Our Features

Single Item Frequency Features

The feature F_E is the number of equation groups identified by the equation grouper. Recall that an equation group is a string of characters belonging to a single equation and written on the same line. Thus, an equation group may not be a complete equation (Figure 1.1). For example, if an equation wraps to a second line, the grouper will identify each line as an equation group. Similarly, if a fraction is written with a horizontal fraction bar (vinculum), the numerator and denominator will likely each be identified as a separate equation group.

Several features describe the frequency of occurrence of various classes of symbols. F_D is the number of single digits in the solution (i.e., 0 – 9). F_L is the number of letters, including both the English alphabet and the Greek letters 'θ' and 'ϕ'. The latter two letters are often used to represent angles. We include only these two Greek letters (and Σ) because they occur far more frequently in our dataset than other Greek letters. F_M is the number of mathematical symbols in the solution including '(', ')', '+', '-', '*',

'/', and '='. F_Σ is the number of occurrences of the symbol 'Σ', which is typically used in equation prototypes, such as "$\Sigma F_x = 0$." Finally, F_C is the total number of characters in the solution: $F_C = F_D + F_L + F_M + F_\Sigma$. Three features describe the relative frequency of the three most common symbol classes: $F_{D/L} = F_D/F_L$, $F_{D/M} = F_D/F_M$, and $F_{L/M} = F_L/F_M$.

The proper use of units of measure is often important for achieving the correct solution to an engineering or science problem. F_U is the number of units of measure in the solution, including "kg", "g", "kN", "N", "m", "lb", "ft", and "in". To be identified as such, units must be immediately preceded by a digit such as "7 lb".

Binary Pattern Frequencies Features

The last category of features are the frequency of occurrence of binary and tripartite sequences of digits (D), letters (L), and mathematical symbols (M). The features F_{ij} for $i, \in \{D, L, M\}$ are the number of occurrences of binary sequences. For example, F_{DM} is the number of pairs of characters containing a digit followed by a mathematical symbol. The feature $F_{=D}$ considers a special instance of a binary sequence. $F_{=D}$ is the number of equal signs followed by a digit, such as "$= 4$". The occurrence of this pattern may indicate that an equation has been reduced to numbers, rather than variables.

Tripartite Pattern Frequencies Features

The features F_{iMj} for $i, j \in \{D, L, M\}$ are the number of occurrences of tripartite sequences. For example, F_{DML} is the number of character sequences containing a digit followed by a mathematical symbol, followed by a letter.

5.1.3 Van Arsdale's Features

Van Arsdale and Stahovich [30] characterize a solution history in terms of the temporal and spatial organization of the work. More specifically, they consider five types of features describing properties of the temporal organization, the spatial organization, the spatial clustering, the cross-outs, and the pen strokes.

Temporal Organization Features

They use nine features to describe this distribution. The first four describe the amount of time spent on various activities. T_{NF} is the total number of intervals spent on free body diagrams, T_{NE} is the number spent on equations, T_{NB} is the number in which no work was done, and T_{ND} is the number of digressions ($T_{ND} = m$). Taking breaks and digressing may indicate that the student is struggling on a problem. They define the complexity, T_C of the sequence as the length of the compressed string. A random sequence of activities will result in a large value for this feature, while a sequence comprising a few large blocks of activities will result in a small value. We use entropy as a measure of the balance of effort between the free body diagram and equation writing activities:

$$T_{Ent} = -(T_{NF}/N)ln(T_{NF}/N) - (T_{NE}/N)ln(T_{NE}/N) \tag{5.2}$$

where $N = T_{NF} + T_{NE}$. If the sequence is dominated by one or the other of these two types of activities, the entropy is relatively small. However, if an equal amount of time is spent on both, the entropy is maximal.

Two additional features consider the frequency of transitions between free body diagram and equation activity. T_{F2E} is the number of transitions from the former to the latter, while T_{E2F} is the converse. The cross-out, break, and digression intervals are removed from the activity sequence before computing these two features.

Three additional features characterize the size distribution of the periods of inactivity. T_{SBr} is the number of small breaks lasting between 2 and 40 sec., T_{MBr} is the number of medium breaks lasting between 40 and 160 sec., and T_{LBr} is the number of large breaks lasting at least 160 sec.

Spatial Organization Features

They define six features representing the fraction of strokes with various amounts of out-of-order timing. These features are $S_{OO:i}$ where $i \in$ (10-20, 20-30, 30-40, 40-50, 50-60, and 60+). For example, $S_{OO:10-20}$ is the fraction of strokes that are out of order by 10% to 20% of the total solution time, while $S_{OO:60+}$ is the fraction that are out of order by at least 60%. The reference timeline provides a global view of the progression of work. A second type of feature provides a more local view by comparing the time stamp of a stroke to those of nearby strokes that were drawn earlier. Two strokes are considered to be near each other if their expanded bounding boxes intersect. For this calculation, the coordinate-aligned bounding boxes of the strokes are expanded in all directions by 0.8. (We obtained this value using a search process aimed at optimizing the predictive ability of the features.) Each stroke is then characterized by the time delay between it and its earliest nearby neighbor. Analogous to the out-of-order features, six

Features	**Description**
T_{NF}	Number of activity intervals spent on FBD activity.
T_{NE}	Number of activity intervals spent on equation activity.
T_{NB}	Number of activity intervals in which a student had no activity.
T_{ND}	Number of times the student interrupted their work on a problem to work on other problems.
T_{Ent}	Entropy of the discretized activity sequence.
T_C	Complexity of the discretized activity sequence.
T_{F2E}	Number of activity changes from FBDs to equations.
T_{E2F}	Number of activity changes from equations to FBDs.
T_{SBr}	Number of breaks between 2 and 40 seconds in duration.
T_{MBr}	Number of breaks between 40 and 160 seconds in duration.
T_{LBr}	Number of breaks at least 160 seconds in duration.

Table 5.3: Summary of the Temporal Organization features.

features are used to characterize this time delay: $S_{EN:i}$ where $i \in$ (10-20, 20-30, 30-40, 40-50, 50-60, and 60+. For example, $S_{EN:10-20}$ is the fraction of strokes with a delay between 10% and 20% of the total solution time.

Features	**Description**
$S_{OO:10-20}$	Fraction of strokes that differ from their reference time by 10% to 20% of the total problem time.
$S_{OO:20-30}$	Fraction that differ by 20%-30%.
$S_{OO:30-40}$	Fraction that differ by 30%-40%.
$S_{OO:40-50}$	Fraction that differ by 40%-50%.
$S_{OO:50-60}$	Fraction that differ by 50%-60%.
$S_{OO:60+}$	Fraction that differ by over 60%.
$S_{EN:10-20}$	Fraction of strokes that have a delay from neighboring strokes of 10% to 20% of the total problem time.
$S_{EN:20-30}$	Fraction that have a delay of 20%-30%.
$S_{EN:30-40}$	Fraction that have a delay of 30%-40%.
$S_{EN:40-50}$	Fraction that have a delay of 40%-50%.
$S_{EN:50-60}$	Fraction that have a delay of 50%-60%.
$S_{EN:60+}$	Fraction that have a delay over 60%.

Table 5.4: Summary of the Spatial Organization features.

Spatial Cluster Features

They characterize the clusters with seven features. C_{NF} and C_{NE} are the numbers of free body diagram and equation clusters, and C_{EAR} is the ratio of the area of the equation clusters to the total area of all clusters. C_{FR} is the number of times the student returned to a free body diagram cluster to revise it, and C_{FRS} is the number of pen strokes added in that way. C_{ER} and C_{ERS} are the analogous properties of the equation clusters.

Features	**Description**
C_{NF}	Number of FBD pen stroke clusters.
C_{FR}	Number of times a student returned to a previous FBD cluster.
C_{FRS}	Fraction of strokes in a solution that were added during FBD revisits.
C_{NE}	Number of equation pen stroke clusters.
C_{EAR}	Ratio of the net area of the equation clusters to the total area of all clusters.
C_{ER}	Number of times a student returned to a previous equation cluster.
C_{ERS}	Fraction of strokes in a solution that were added during equation revisits.

Table 5.5: Summary of the Spatial Cluster features.

Cross-Out Features

They characterize cross-out gestures with five features. X_T and X_{PS} are the number of typo and problem-solving cross-outs, respectively. The former are cases in which the student writes something and quickly crosses it out, as if correcting a typographical error. The latter are cases in which there is a substantial delay between the time the ink was written and when it was crossed out: these cases are more likely to be corrections of problem-solving errors. X_B is the number of "big" cross-outs that delete 10 or more pen strokes, and thus represent a revision of a substantial amount of work. Finally, X_F and X_E are the number of crossed-out free body diagram and equation strokes, respectively.

Features	**Description**
X_F	Number of FBD strokes that were crossed-out.
X_E	Number of equation strokes that were crossed-out.
X_B	Number of cross-out gestures which removed 10 or more strokes.
X_T	Number of cross-out gestures which occurred within 16 seconds of underlying ink.
X_{PS}	Number of cross-out gestures which occurred after 16 seconds of underlying ink.

Table 5.6: Summary of the Cross-out features.

Basic Pen Stroke Features

They include six features describing the properties of the pen strokes. P_{NF} , P_{NE}, and P_{NX} are the number of free body diagram, equation and cross-out strokes, respectively. Likewise, P_{LF} , P_{LE}, and P_{LX} are the median length of the free body diagram, equation, and cross-out strokes, respectively.

Features	**Description**
P_{NF}	The total number of FBD strokes in the problem solution.
P_{NE}	The total number of equation strokes in the problem solution.
P_{NX}	The total number of cross-out strokes in the problem solution.
P_{LF}	Median length of FBD strokes in the problem solution.
P_{LE}	Median length of equation strokes in the problem solution.
P_{LX}	Median length of cross-out strokes in the problem solution.

Table 5.7: Summary of the Basic Pen Stroke features.

Chapter 6

Dataset

We used LiveScribe digital pens to collect homework and exam solutions from an undergraduate mechanical engineering course in statics from 2010 to 2013. Students enrolled in the course 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.

Figure 6.1: LiveScribe digital pen

6.1 Set A - Homework

In the winter quarter of 2010, we conducted a study with 132 students. In total, 6,562 sketches were collected from 12 exams, 30 homework assignments, and 7 quiz problems. We hand labeled 8 homework problems in a total of 810 homework sketches, of which 69% are equation strokes to provide ground truth data for training and testing our stroke classifier.

6.2 Set B - Homework

In this set, we use solutions to four homework problems from one course offering in 2013 to train our algorithms. This dataset, which we refer to as the homework dataset, comprises a total of 131,304 pen strokes, of which 72% are equation strokes. We hand labeled this dataset to provide ground truth data for training and testing our text interpretation techniques including the groupers and HMM.

6.3 Set C - Exams

We use solutions to six midterm exam problems from a different course offering in 2012 to evaluate the predictive power of our features. This dataset, which we call the exam dataset, comprises a total of 1,069,918 pen strokes, of which 72% are equation strokes. The exam problems were graded by teaching assistants based on rubrics developed by the course instructor. These rubrics assigned credit for the correctness of individual problem-solving steps as well as the overall correctness of the solution. We use our features to predict the assigned grade. Appendix A includes examples of the exam we used in this data set.

6.4 Image-based Recognizer

Our image-based recognizer was trained on a separate set of 450 symbols comprising 10 digits, 22 capital letters, 8 arrows, and 5 mathematical symbols ("$-$", "Σ", "/", "(", ")") from Set A. The eight arrows here mean that arrows point to eight directions. Each symbol (digits, letters, symbols and arrows) has 10 templates in the training set.

Chapter 7

Results

7.1 Stroke Classifier

7.1.1 Cross-out Classifier Accuracy

As previously mentioned, homework solutions typically contain few cross-outs. To evaluate the cross-outs classifier, we first randomly pick up free body diagram strokes and equation strokes with the same amount of cross-outs strokes from the Set A. We then evaluated the cross-outs classifier with the 10 cross-outs stroke feature as described before. Table 7.1 shows the accuracy for our cross-outs classifier and Stahovich's. Our method had an overall accuracy of 92.2% in classifying cross-outs and non-cross-outs.

7.1.2 Stroke Classification Accuracy

We used an Adaboosted C4.5 Decision Tree, implemented in WEKA [40], with its default parameters, and 10 fold cross-validations to train and evaluate the test set.

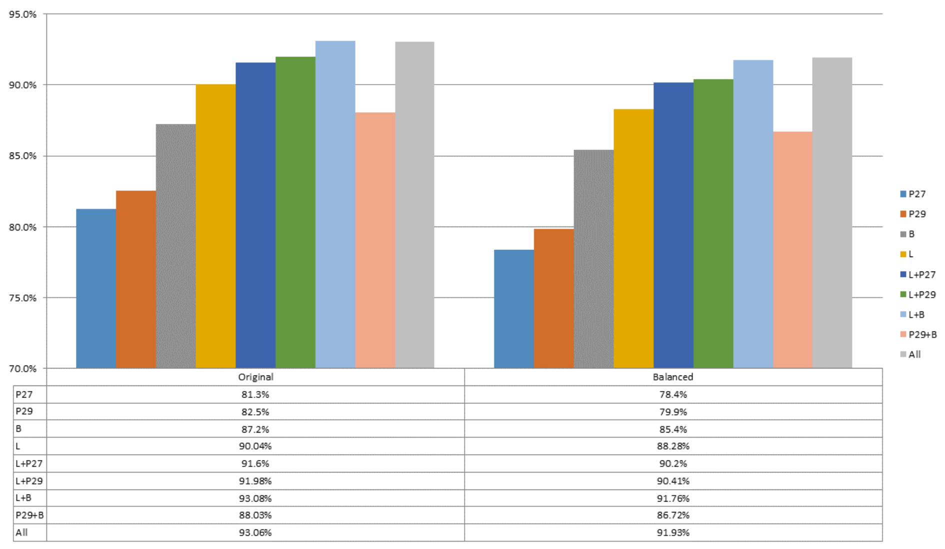
Method	Actual Class	Classified As		Accuracy
		Cross-out	Non Cross-out	
Ours	Cross-out	3154	424	88.15%
	Non Cross-out	342	6812	95.22%
	Overall			92.86%
Peterson's	Cross-out	2658	920	74.30 %
	Non Cross-out	960	6164	86.52 %
	Overall			80.43 %

Table 7.1: Comparison results.

All results in the following section were obtained using user hold-out-cross-validation. The final accuracy is averaged across all users.

We compare the performance of our classification technique to that of Stahovich et al. [1] and Blagojevic el al. [21]. To provide additional insights into the differences between the three-feature sets, we created stroke classifiers which contains all combinations of the three sets of features. Figure 7.1 presents the result of the comparisons. We evaluated all methods using the original data and balanced data. As we can see in the previous section, around 70% of strokes in a static solution are equation strokes. Therefore, we balanced the data so that it contains the same amount of equation strokes and free body diagram strokes. Since free body diagram is the minor class, we randomly picked up the same amount of strokes from equation strokes to generate a balanced dataset. Our approach alone performed better than Stahovich's, and Blagojevic's in both data sets. It also indicates we can combine all three feature-set to achieve high accuracy. However, because of the large number of features, the stroke classifier uses combined features-set requires more training and calculation time.

Table 7.2 contains confusion matrices for FBD versus Equation classification for



	Original	Balanced
P27	81.3%	78.4%
P29	82.5%	79.9%
B	87.2%	85.4%
L	90.04%	88.28%
L+P27	91.6%	90.2%
L+P29	91.98%	90.41%
L+B	93.08%	91.76%
P29+B	88.03%	86.72%
All	93.06%	91.93%

Figure 7.1: Comparison results.

our method, Stahovich's, Blagojevic's and combined. Our method achieved an overall recognition accuracy of 88.28% while Stahovich's method has an accuracy of 79.85%, and Blagojevic's has accuracy of 85.24%. If we combined the three feature sets, it can achieve an accuracy of 91.93%; however, it take more time on training and testing.

7.1.3 Stroke Feature Importance

To determine the most important features, we used the information gain algorithm, implemented in WEKA, to rank the individual discriminating power of each feature as shown in Table 7.3. Six of our features can be found within the top 10 of the list. The first and the third feature are temporal features. duration time between two consecutive strokes in an equation is usually shorter than that in a free body diagram. Table 7.3 shows these two are useful on classifying equation strokes and free body diagram strokes. In the sketch of a statics solution, free body diagram strokes usually have

Method	Actual Class	Classified As		Accuracy
		Equation	FBD	
P29	Equation	5141	1262	80.28%
	FBD	1317	5086	79.43%
	Overall			79.85%
B	Equation	5483	920	85.63%
	FBD	945	5458	85.24%
	Overall			85.43%
L	Equation	5714	689	89.23%
	FBD	812	5591	87.32%
	Overall			88.28%
ALL	Equation	5946	457	92.87%
	FBD	577	5826	90.99%
	Overall			91.93%

Table 7.2: Confusion matrix.

other free body diagram strokes nearby, as well as having many other strokes overlap with its bounding box area. In addition, free body diagrams usually have force arrows, and the direction to the next stroke is usually not consistent. On the other hand, equation strokes usually have other equation strokes nearby. Our features reflect these characteristics ranked in top 10 of the list. The feature *Distance To Top or Bottom Edge* (P_{D2TB}) also seems important in the statics problems because free body diagram strokes are usually drawn at the top of the sketch.

7.1.4 ANOVA

We performed repeated measures analysis of variance (ANOVA) on the accuracy of each method on the original data. Table 7.4 revealed that the difference in the accuracy between the method which combined our features and Blagojevic's (LB) and the method which combined all features (A) is not significant ($p = 0.453$) while the difference in the accuracy between all other methods are significant. The result indicates

Rank	**Features**
1	P_{T2N}
2	L_{NAR}
3	P_{T2P}
4	L_{NCR}
5	L_{NLR}
6	L_{NDT}
7	L_{D2N}
8	P_{D2TB}
9	L_{UD}
10	P_{ILL}
11	P_{BBH}
12	P_{T2D}
13	P_{BBW}
14	P_{SOCR}
15	P_{ILX}
16	P_{IXL}
17	P_{SOCABS}
18	P_{IXX}
19	P_{BBA}
20	P_{SOC2}
21	P_{ID}
22	P_{IP}
23	L_{LS}
24	P_{AL}
25	P_{APS}
26	P_{MinPS}
27	P_{SI}
28	P_{D2LR}
29	P_{SOC}
30	P_{MaxPS}
31	L_{PC}
32	$P_{DMaxMin}$
33	P_{CP}
34	L_{NS}
35	L_{NU}
36	P_{EPR}
37	P_{SE}

Table 7.3: The best features for classification of individual strokes. Features are ranked according to their average merit as determined by WEKA's information-gain- ratio attribute-selection algorithm.

that the effect is less when adding Stahovich's features on classifying strokes.

	S27	S29	B	L	LS27	LS29	LB	BS29	A
S27		0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
S29			0.000	0.000	0.000	0.000	0.000	0.000	0.000
B				0.000	0.000	0.000	0.000	0.000	0.000
L					0.000	0.000	0.000	0.000	0.000
LS27						0.000	0.000	0.000	0.000
LS29							0.000	0.000	0.000
LB								0.000	0.453
BS29									0.000
A									

Table 7.4: Significant values.

7.2 Interpreting the Text

We evaluated the performance of our equation and character groupers using Set B. We trained and tested our algorithms using a problem-holdout approach. In each of the four folds of training and testing, solutions to three of the four problems were used for training, and solutions to the fourth were used for testing. We then averaged the four results. We use three metrics from (Stahovich, Peterson, and Lin [1]) to measure the accuracy of our groupers. Ink-Found is the percentage of a group's ink (by arc length) that was correctly located. Ink-Extra is the percentage of ink that was erroneously added to a group, with the ground truth ink as the basis. Both of these are computed on a per-group basis, and then averaged across all groups. The third measure is the percentage of groups that have no more than a given number (X) of erroneous strokes. $X = 0$ is "all-or-nothing" accuracy. For example, an equation group that contains one extra pen stroke and no other errors would be counted as having "one or fewer errors"

(X = 1).

Before evaluating the performance with these metrics, we mapped the computed groups to the ground truth groups. We exhaustively calculated the ratio of the common strokes in the pair of the computed groups and ground truth groups. The ratio defined as:

$$R_{ij} = \frac{Common_{ij}}{NGS_j} \tag{7.1}$$

where $Common_{ij}$ is the number of strokes exists in both group i in the computed groups and group j in the ground truth groups. NGS_j is the number of strokes in group j in the ground truth groups. We then mapped two groups in each group set from the highest ratio R_{ij} until no more groups could be mapped. Finally, the rest of the groups were added as extra groups or missing groups.

Table 7.5 compares that accuracy of our equation grouper to that of Stahovich's grouper (2014). We trained and tested our algorithms using a problem-holdout approach. In each of the four folds of training and testing, solutions to three of the four problems were used for training, and solutions to the fourth were used for testing. We then averaged the four results. In addition to being more computationally efficient, our approach performs better on all three accuracy measures. For example, our grouper located an average of 93.8% of the ink for each equation group, while Stahovich's located an average of 93.0%. More importantly, our method achieved an all-or-nothing accuracy of 80.5%, while Stahovich's achieved 75.5%. Lee et al. (2012) extended Stahovich's grouper for use in grouping pen strokes into characters. Table 7.6 compares the accuracy of our character grouper to that of Lee's grouper. Our grouper has about the same accuracy as Lee's.

We performed repeated measures analysis of variance (ANOVA) on the four problem set and accuracy of the groupers. Table 7.7 represents that ANOVA revealed that the difference in the accuracy between our equation grouper and combined grouper is significant while the accuracy between Stahovich's grouper and combined grouper is not significant ($p = 0.78$). The result indicates that the effect is less to add Stahovich's features. Table 7.8 reveal that the difference in the accuracy between our character grouper and Lee's grouper is not significant ($p = 0.699$).The results show there is no difference in accuracy between Lee's grouper and ours. However, ours only uses two features while Lee's uses six features.Just as our equation grouper is more efficient than Stahovich's, our character grouper is more efficient than Lee's.

We also evaluated the accuracy of our HMM using the Set B. Here, we used 10-fold cross validation and averaged the results. The image-based recognizer achieved an average recognition accuracy of 85.1%. Using the HMM to correct recognition results improved the average accuracy to 92.6%.

	Ink		Eqns: X Errors or fewer		
	Found	Extra	0	1	2
L	93.8%	6.2%	80.5%	97.1%	99.4%
S	93.0%	7.0%	75.5%	96.8%	99.3%
A	94.0%	6.0%	80.5%	97.1%	99.3%

Table 7.5: Equation grouping accuracy. L = our equation grouper. S = the grouper from [1]. A = the grouper which combined our feature set and Stahovich's

	Ink		Eqns: X Errors or fewer		
	Found	Extra	0	1	2
L	94.6%	5.4%	89.9%	99.3%	99.9%
J	94.6%	5.4%	90.2%	99.4%	99.9%
A	93.9%	6.1%	92.1%	99.5%	99.9%

Table 7.6: Character grouping accuracy. L = our character grouper. J = the method from [2]. A = the grouper which combined our feature set and Lee's

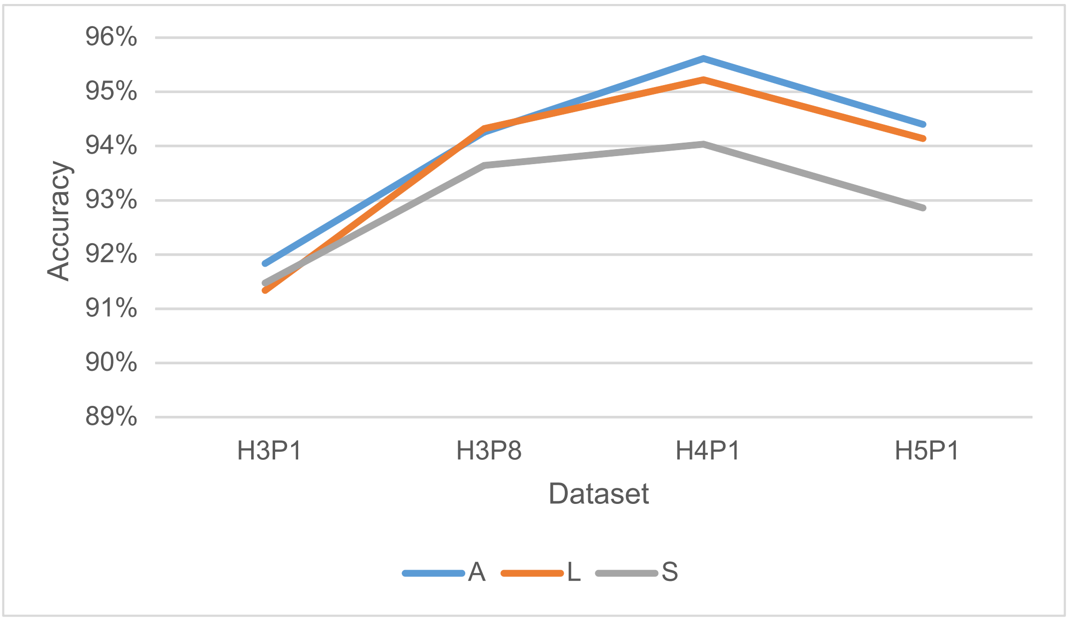


Figure 7.2: Equation grouping accuracy for each problem. L = our character grouper. S = the grouper from [1]. A = the grouper which combined our feature set and Lee's

	A	L	P
A		0.000	0.78
L			0.000
S			

Table 7.7: Equation grouping significant values. L = our equation grouper. S = the grouper from [1]. A = the grouper which combined our feature set and Stahovich's

	A	L	J
A		0.062	0.000
L			0.699
J			

Table 7.8: Character grouping significant values. L = our character grouper. J = the method from Lee's [2]. A = the grouper which combined our feature set and Lee's

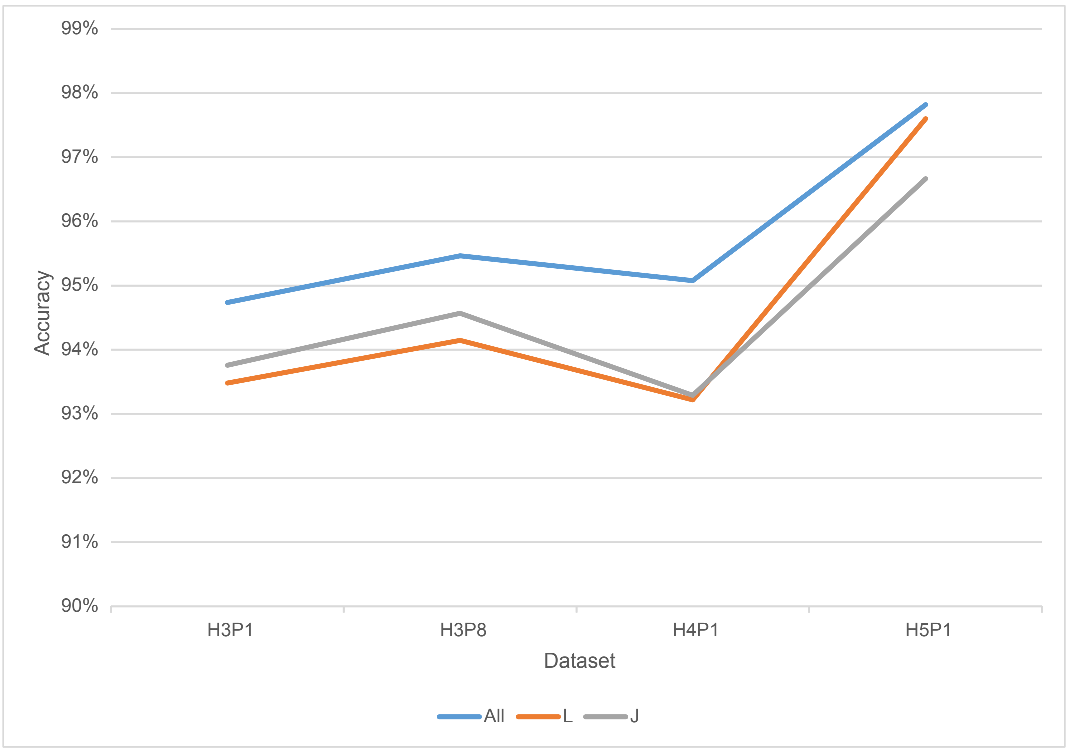


Figure 7.3: Character grouping accuracy for each problem. L = our character grouper. J = the method from [2]. A = the grouper which combined our feature set and Lee's

7.3 Extracting Features from Text

7.3.1 Our Features

We used the Set C to investigate the ability of our features to predict student competence. More specifically, we used the features to train a classifier to predict the grade assigned to each exam solution by the instructor. We again used a problem hold-out approach with the six problems in the dataset. We trained the models using WEKA's (Hall et al. [40]) SVM regression technique (SMOreg) with default parameter values. For this analysis, we trained our groupers and HMM on the complete homework dataset.

We used a beam search process, with a beam width of five, to determine which sets of features are the most effective for predicting the correctness of a student's exam solution. To begin, all possible single-feature classifiers were trained and then evaluated on the test data. The five most accurate classifiers were then expanded to produce a set of two-feature classifiers. The five best of these were then expanded to produce three-feature classifiers, and so on. The results of this analysis are plotted in Figure 7.4.

The best classifiers (Table 7.10) used between three and eight features and achieved a coefficient of determination (R^2) ranging from 0.099 to 0.53. The average value of R^2 across all six problems was 0.36. F_{MM}, F_P , F_E, and F_Σ are the features that occurred most frequently in these models (Table 7.9). They appeared in the best models for half of the six problems. F_{LL}, F_{LM}, F_{LD}, and F_{DMD} are the next most frequently appearing features. They appeared in the best models for a third of the problems.

To gain additional insights into the predictive power of the features, we constructed additional models using the top four most frequently occurring features and the top eight. Both sets of features achieved an average R^2 of 0.21. Thus, the top four features had about two-thirds of the predictive power of the complete set of features.

Our features (Table 7.9) represent a variety of properties of equations. Here, we examine their relative predictive power. We consider four types of features: (L_A) pause count, $\{F_P\}$; (L_B) single item frequencies, $\{F_E, F_D, F_L, F_M, F_\Sigma, F_C, F_{D/L}, F_{D/M}, F_{L/M}, F_U\}$; ($L_C$) binary pattern frequencies, $\{F_{DD}, F_{DM}, F_{DL}, F_{LD}, F_{LM}, F_{LL}, F_{MD}, F_{MM}, F_{ML}, F_{=D}\}$; and ($L_D$) tripartite pattern frequencies, $\{F_{DMD}, F_{DML}, F_{LMD}, F_{LML}\}$. We constructed separate models for each feature type. We again used 10-fold cross validation and beam search to determine the best models for each problem and then averaged the results across the 6 problems (Table 7.12). The item count features (L_B) had the greatest predictive power, with an average $R^2 = 0.29$. While the pause count feature type (L_A) comprises only a single feature, it achieved an average $R^2 = 0.20$. The off-diagonal terms in the table contain the average R^2 values for combinations of two types of features. The item frequency features (L_B) combined with the binary pattern features (L_C) are the most effective pair of feature types, with $R^2 = 0.34$.

7.3.2 Van Arsdale's Features

Van Arsdale [30] uses 41 features on predict student performance. Here we used our dataset (Set C) to evaluate the ability of their features. In the same fashion, we used the features to train a classifier to predict the grade assigned to each exam

Features	**Occurrence**
F_P	3
F_E	3
F_D	1
F_L	1
F_M	1
F_Σ	3
F_C	0
$F_{D/L}$	1
$F_{D/M}$	1
$F_{L/M}$	0
F_U	1
F_{DD}	1
F_{DM}	1
F_{DL}	0
F_{LD}	2
F_{LM}	2
F_{LL}	2
F_{MD}	1
F_{MM}	3
F_{ML}	1
$F_{=D}$	0
F_{DMD}	2
F_{DML}	0
F_{LMD}	1
F_{LML}	0

Table 7.9: Features occurrence using our feature set. D = digit, L = letter, M = mathematical symbol, “units” are units of measure such as “kg” and “lb”.

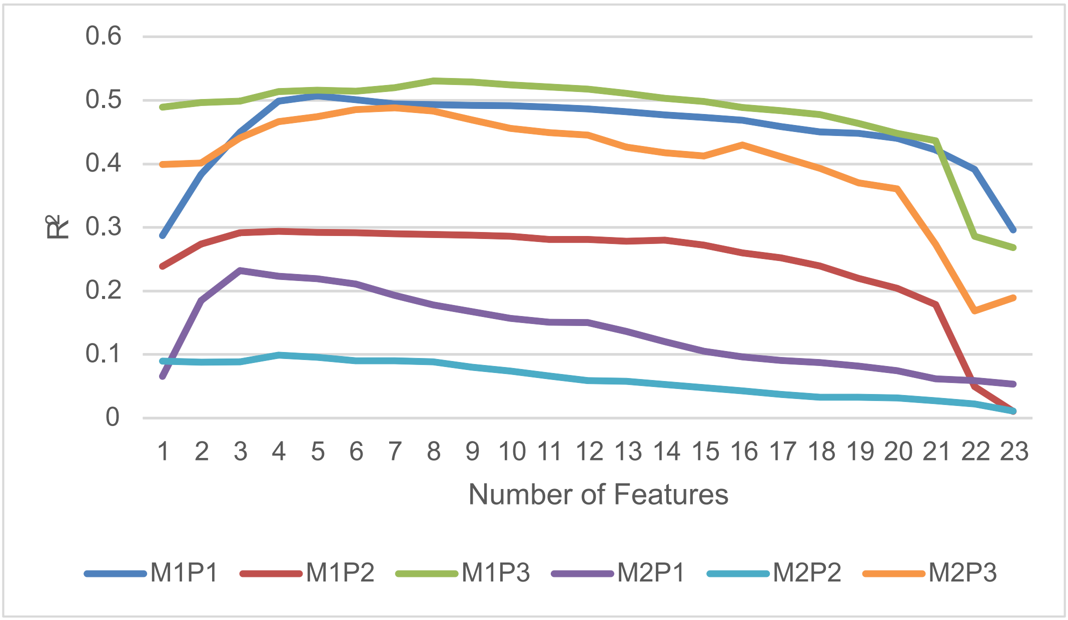


Figure 7.4: Coefficient of determination (R^2) achieved by SVM regression models vs. the number of features used in our feature set.

solution by the instructor. We again used a problem hold-out approach with the six problems in the dataset. We trained the models using WEKA's (Hall et al. [40]) SVM regression technique (SMOreg) with default parameter values.

We used a beam search process as we did in the previous section to determine which sets of features are the most effective for predicting the correctness of a student's exam solution. The results of this analysis are plotted in Figure 7.5.

Data	**Best Model**	R^2
M1P1	F_{LM} $F_{D/M}$ F_{DMD} F_{Σ} $F_{D/L}$	0.51
M1P2	F_P F_{Σ} F_{LL} F_{DM}	0.29
M1P3	F_P F_U F_{MD} F_{DMD} F_D F_{LD} F_{MM} F_E	0.53
M2P1	F_E F_{DD} F_{LD}	0.23
M2P2	F_{MM} F_L F_E F_M	0.099
M2P3	F_{MM} F_{LMN} F_P F_{Σ} F_{LL} F_{ML} F_{LM}	0.49
Ave		0.36

Table 7.10: The best coefficient of determination (R^2) and the corresponding feature set identified by beam search using our feature set.

Problem	R^2 for Top 4	R^2 for Top 8
M1P1	0.16	0.20
M1P2	0.25	0.26
M1P3	0.45	0.44
M2P1	0.01	0.05
M2P2	0.05	0.05
M2P3	0.35	0.28
Ave	0.21	0.21

Table 7.11: Coefficient of determination (R^2) achieved by SVM regression models using the top four and top eight features in our feature set.

	L_A	L_B	L_C	L_D
L_A	0.20	0.31	0.27	0.23
L_B		0.29	0.34	0.33
L_C			0.26	0.27
L_D				0.18

Table 7.12: Average R^2 values of SVM regression models for various types of features in our feature set. $L_A = \{FP\}$, $L_B = \{itemfreq.\}$, $L_C = \{binarypatternfreq.\}$, $L_D = \{tripartitepatternfreq.\}$. Diagonal elements are single types; off-diagonal are pairs of types.

The best classifiers (Table 7.13) used between ten and twenty-one features and achieved a coefficient of determination (R^2) ranging from 0.23 to 0.70. The average value of R^2 across all six problems was 0.51. P_{LX},T_{NE}, and X_E are the features that occurred most frequently in these models (Table 7.14). They appeared in the best models for five out of the six problems. T_{SBr},$S_{OO:20-30}$,$S_{OO:50-60}$,T_{Ent},$S_{EN:40-50}$,T_{NB},X_{PS},C_{ER},and P_{NE} are the next most frequently appearing features. They appeared in the best models for half of the problems.

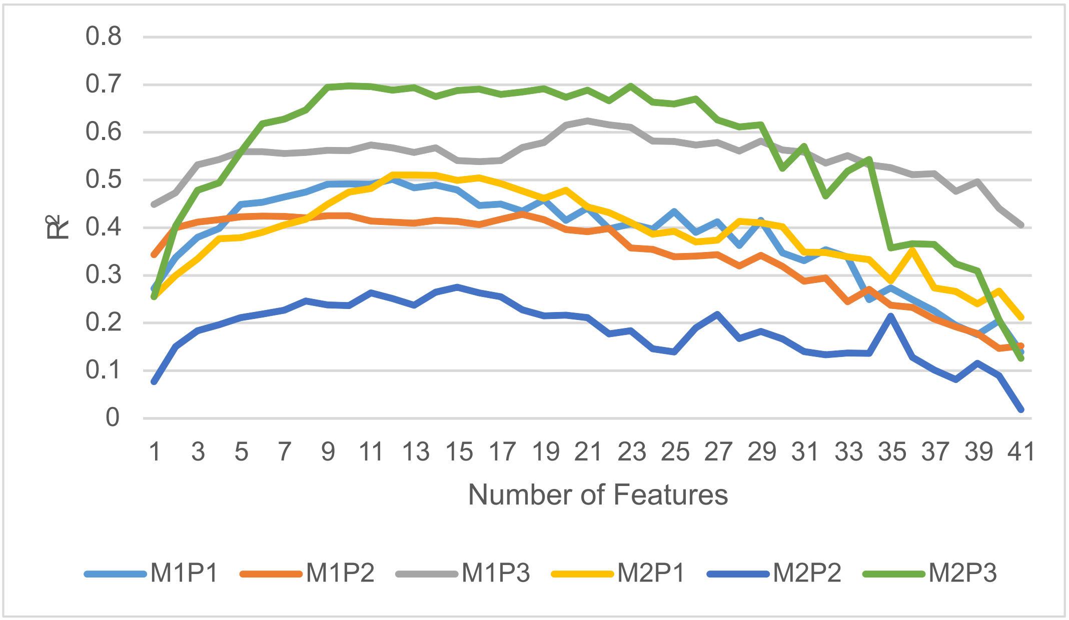


Figure 7.5: Coefficient of determination (R^2) achieved by SVM regression models vs. the number of features used in Van Arsdale's feature set.

7.3.3 Combined Features

Since our work is complementary to that of Van Arsdale's, we combined our features with theirs to see if we could produce more predictive models.

We applied same analysis on the combined features. Figure 7.6 demonstrates the results of the sets of features which are the most effective for predicting the correctness of a student's exam solution.

The best classifiers (Table 7.15) used between thirteen and twenty-two features and achieved a coefficient of determination (R^2) ranging from 0.37 to 0.73. The average value of R^2 across all six problems was 0.56. T_{NB} is the feature that occurred most frequently in these models (Table 7.16). It appeared in the best models for four out of the six problems. F_{LL},F_{ML}, P_{LX},C_{NF},F_L,F_P, $S_{EN:10-20}$, X_E, F_C,F_E, and $F_{D/L}$ are the next most frequently appearing features. They appeared in the best models for half of

Best	**R2**	**n**
M1P1	0.50	12
M1P2	0.43	18
M1P3	0.62	21
M2P1	0.51	13
M2P2	0.27	15
M2P3	0.70	10
Ave	0.51	15

Table 7.13: SVM regression models for 6 problems using Van Arsdale's feature set. Best = R^2 of best models obtained using beam search. n = number of features in the best models.

the problems. More than half of the features which appeared in the best model shown above are from our features set.

We again used 10-fold cross validation and beam search to determine the best models for each problem and then averaged the results across the 6 problems (Table 7.12). The temporal feature (T_T) from Van Arsdale's feature had the greatest predictive power, with an average $R^2 = 0.39$. The temporal feature (T_T) combined with our item count features (L_B) are the most effective pair of feature types, with $R^2 = 0.43$.

Features	**Occurrence**
P_{LX}	5
T_{NE}	5
X_E	5
T_{SBr}	3
$S_{OO:20-30}$	3
$S_{OO:50-60}$	3
T_{Ent}	3
$S_{EN:40-50}$	3
T_{NB}	3
X_{PS}	3
C_{ER}	3
P_{NE}	3
T_{LBr}	2
T_{MBr}	2
T_{ND}	2
T_{F2E}	2
T_{E2F}	2
$S_{OO:10-20}$	2
$S_{OO:60+}$	2
P_{NF}	2
T_C	2
$S_{EN:10-20}$	2
$S_{EN:20-30}$	2
$S_{EN:30-40}$	2
$S_{EN:60+}$	2
T_{NF}	2
X_T	2
X_B	2
C_{NE}	2
P_{LE}	2
C_{ERS}	2
P_{NX}	1
$S_{OO:30-40}$	1
$S_{OO:40-50}$	1
P_{LE}	1
C_{FRS}	1
C_{FR}	1
$S_{EN:50-60}$	1
X_F	1
C_{EAR}	1

Table 7.14: Features occurrence using Van Arsdale's features. (Only listed features which occur more than once.)

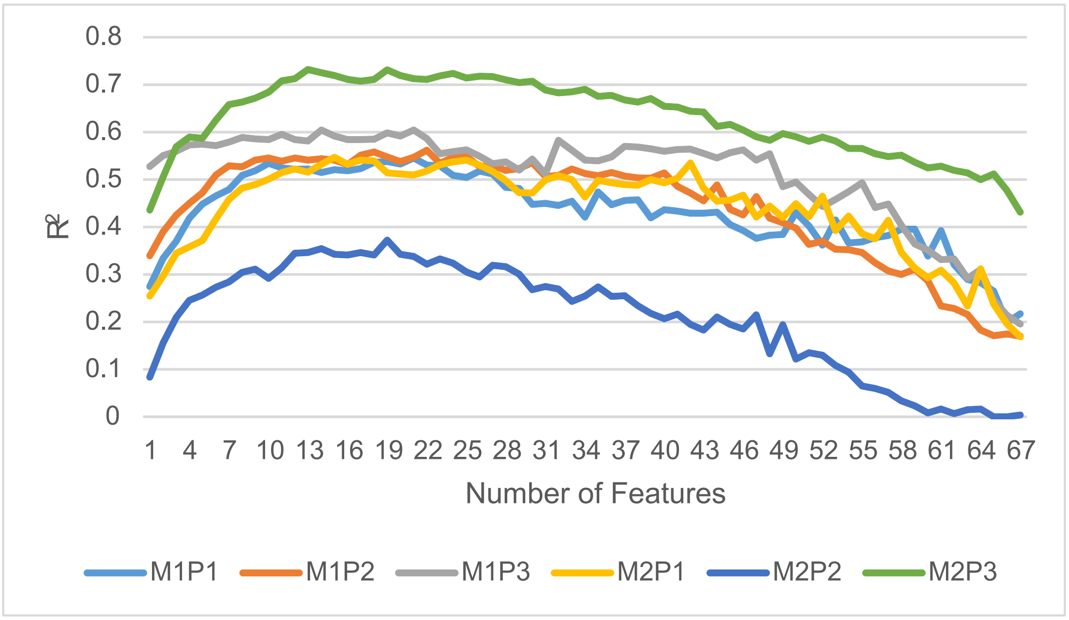


Figure 7.6: Coefficient of determination (R^2) achieved by SVM regression models vs. the number of features used in combined feature set.

Best	**R2**	**n**
M1P1	0.55	21
M1P2	0.56	22
M1P3	0.60	14
M2P1	0.55	15
M2P2	0.37	19
M2P3	0.73	13
Ave	0.56	17

Table 7.15: SVM regression models for 6 problems using combined features. Best = R^2 of best models obtained using beam search. n = number of features in the best models.

Features	**Occurrence**	**Features**	**Occurrence**
T_{NB}	4	F_D	2
F_{LL}	3	P_{NX}	2
F_{ML}	3	X_F	2
P_{LX}	3	P_{LE}	2
C_{NF}	3	C_{ERS}	2
F_L	3	C_{ER}	2
F_P	3	P_{NE}	2
$S_{EN:10-20}$	3	T_{LBr}	1
X_E	3	T_{SBr}	1
F_C	3	T_{ND}	1
F_E	3	F_{LM}	1
$F_{D/L}$	3	F_{MM}	1
T_{MBr}	2	F_{NL}	1
T_{F2E}	2	F_{NM}	1
T_{E2F}	2	F_{NN}	1
F_{MN}	2	P_{NX}	1
$S_{OO:30-40}$	2	$S_{OO:20-30}$	1
$S_{OO:40-50}$	2	$S_{OO:50-60}$	1
T_{Ent}	2	$S_{OO:60+}$	1
$F_{=N}$	2	P_{LF}	1
C_{ERS}	2	$S_{EN:20-30}$	1
P_{NF}	2	F_{NML}	1
F_U	2	F_{NMN}	1
F_M	2	X_{PS}	1
$S_{EN:40-50}$	2	$F_{D/M}$	1
$S_{EN:50-60}$	2	F_{Σ}	1
T_{NF}	2	C_{NE}	1
T_{NE}	2		

Table 7.16: Number of times the feature appeared in a “best” model in combined feature set.

	L_A	L_B	L_C	L_D	T_T	T_S	T_C	T_X	T_P
L_A	0.20	0.31	0.27	0.23	0.39	0.31	0.29	0.25	0.29
L_B		0.29	0.34	0.33	0.433	0.35	0.36	0.32	0.33
L_C			0.26	0.27	0.430	0.35	0.33	0.32	0.33
L_D				0.18	0.38	0.30	0.28	0.24	0.26
T_T					0.36	0.39	0.38	0.36	0.40
T_S						0.18	0.31	0.23	0.34
T_C							0.21	0.26	0.31
T_X								0.03	0.27
T_P									0.22

Table 7.17: Average R^2 values of SVM regression models for various types of features in combined feature set. $L_A = \{FP\}$, $L_B = \{itemfreq.\}$, $L_C = \{binarypatternfreq.\}$, $L_D = \{tripartitepatternfreq.\}$, $T_T = \{temporalorganization\}$, $T_S = \{spatialorganization\}$, $T_C = \{cluster\}$, $T_X = \{cross - out\}$, $T_P = \{penstroke\}$. Diagonal elements are single types; off-diagonal are pairs of types.

Chapter 8

Discussion

The results in Table 7.12 indicate that various types of frequency features are all predictive of correctness. This demonstrates that it is possible get a useful measure of correctness without a full interpretation of the equations. Table 7.12 suggests that the frequency features tend to be more predictive than the pause count feature. However, F_P was selected for the best models for half of the problems (Table 7.9). Thus, this feature is useful in conjunction with the frequency features.

In the combined model, the temporal organization features are the most predictive features. However, only the feature (T_{NB}), which indicates the number of times in which no work was done in the solving process, has ranked in the top 12 features which were selected for the best models for half of the problems. On the other hand, we have eight features selected in the top twelve features - five frequency features, one the pause count feature, and two binary pattern frequency features (Table 7.16). It indicates that the combination of our features and the temporal features from Van Arsdale's features are useful in predicting students performance.

For all but exam problem M2P2, the SVM regression models achieved high correlations. We expected that the predictive power of the features might depend on the difficulty of the exam problems. However, there was only a slight negative correlation between the R^2 values of the best models and the average grades students achieved on the problems. Further analysis is needed to understand the lack of predictive ability for problem M2P2.

The four most predictive features in our features set are F_P , F_{MM}, F_E, and F_Σ. F_P has a moderate positive correlation with grade, with $R^2 = 0.22$. F_{MM} has a smaller positive correlation, with $R^2 = 0.11$. F_E and F_Σ have weak positive correlations with grade, with R^2 values of 0.07 and 0.04, respectively. It appears that high-performing students tend to have more long pauses than low-performing students. Cheng and Rojas-Anaya [31], by contrast, found a negative correlation between long pauses and competence. The difference between our results and theirs is likely due to the nature of the tasks considered. We examined a problem-solving task, while they considered the task of copying equations.

Chapter 9

Conclusion

In this paper, we have presented an automatic stroke labeling technique and demonstrated a technique that uses lexical properties of a student's handwritten equations to evaluate the correctness of the work. Our technique first classifies single strokes into one of the three semantic classes by using machine learning techniques;group classified strokes into equations and characters; and lastly extracts features from the interpreted results.

We characterize a solution with a number of quantitative features describing inter-character pauses and the frequencies of various classes of symbols and binary and tripartite sequences of symbols.We use these features to construct SVM regression models to predict the correctness of the work, i.e., the grade a human expert would assign. We evaluated these techniques on a dataset containing solutions to exam problems from an undergraduate engineering course in statics. Students completed the exam problems using digital pens that recorded the work as time-stamped pen strokes. SVM regression models revealed that, on average, 36% of the variance in student performance on these problems could be explained by our features. This is a surprising result given that our approach does not attempt to interpret the equations or even the final numerical answer.

Our approach for assessing the correctness of a student's work employs simple lexical analysis. This approach is attractive because it does not require recognition of diagrams and equations, nor does it require knowledge of the subject. Consequently, our approach should be readily extensible to other subject areas. In particular, we expect that our techniques will be useful for assessing student learning in a variety of engineering, science, and math domains.

Bibliography

[1] Thomas F. Stahovich, Eric J. Peterson, and Han-Lung Lin. An efficient, classification-based approach for grouping pen strokes into objects. *Computers & Graphics*, 42:14–30, August 2014.

[2] Chia-Keng Lee, Josiah Jordan, Thomas F Stahovich, and James Herold. Newtons Pen II : An Intelligent , Sketch-Based Tutoring System and its Sketch Processing Techniques. In *Sketch-based Interfaces and Modeling*, pages 57–65, 2012.

[3] C. Romero and S. Ventura. Educational Data Mining: A Review of the State of the Art. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 40, 2010.

[4] Reyna De Los Angeles Garcia. *Intelligent Statics Tutoring Systems as Tools for Undergraduate Mechanical Engineering Education.* Master thesis, UNIVERSITY OF CALIFORNIA RIVERSIDE, 2013.

[5] Paul Blanc and Saint Martin. SOLVING A NON-ROUTINE PROBLEM: what helps, what hinders? Paul Blanc Saint Martin's College. *Learning*, 19(June):1–6, 1999.

[6] J W Jordan. *An Intelligent Tutoring System for Static Equilibrium Equations.* University of California, Riverside, 2011.

[7] C K Lee. *Statics Intelligent Tutoring System.* BiblioBazaar, 2012.

[8] André Krüger, Agathe Merceron, and Benjamin Wolf. A Data Model to Ease Analysis and Mining of Educational Data. In *Proc. of the 3rd Int. Conf. on Educational Data Mining (EDM'2010)*, pages 131–140, 2010.

[9] C Romero and JR Romero. Mining rare association rules from e-learning data. *. . . of educational data mining. . . .* , pages 171–180, 2010.

[10] Carole R Beal and Paul R Cohen. Temporal data mining for educational applications. In *Proceedings of 10th Pacific Rim International Conference on Artificial Intelligence*, pages 66–77, 2008.

[11] D H Shanabrook, D G Cooper, B P Woolf, and I Arroyo. Identifying high-level student behavior using sequence-based motif discovery. In *Educational Data Mining 2010 - 3rd International Conference on Educational Data Mining*, pages 191–200, 2010.

[12] Jack Mostow, José González-Brenes, and Bao Hong Tan. Learning Classifiers from a Relational Database of Tutor Logs. In *Educational Data Mining, proceedings of the 4th International Conference on*, pages 149–158, 2011.

[13] E J Peterson, T F Stahovich, E Doi, and C Alvarado. Grouping Strokes into Shapes in Hand-Drawn Diagrams. In *Artificial Intelligence*, pages 974–979, 2010.

[14] Rachel Patel, Beryl Plimmer, John Grundy, and Ross Ihaka. Ink features for diagram recognition. In *Sketch-based Interfaces and Modeling*, page 131, 2007.

[15] Akshay Bhat and Tracy Hammond. Using entropy to distinguish shape versus text in hand-drawn diagrams. In *IJCAI International Joint Conference on Artificial Intelligence*, pages 1395–1400, 2009.

[16] Christopher M. Bishop, Markus Svensén, and Geoffrey E. Hinton. Distinguishing text from graphics in on-line handwritten ink. In *Proceedings - International Workshop on Frontiers in Handwriting Recognition, IWFHR*, pages 142–147, 2004.

[17] Xin Wang, Manoj Biswas, and Sashi Raghupathy. Addressing class distribution issues of the drawing vs writing classification in an ink stroke sequence. *Proceedings of the 4th Eurographics workshop on Sketch-based interfaces and modeling - SBIM '07*, 1:139, 2007.

[18] Leslie Gennari, Levent Burak Kara, Thomas F. Stahovich, and Kenji Shimada. Combining geometry and domain knowledge to interpret hand-drawn diagrams. *Computers and Graphics (Pergamon)*, 29:547–562, 2005.

[19] Josiah Jordan. *Dr. Thomas F. Stahovich, Chairperson Dr. Eamonn Keogh Dr. V. Sundararajan.* Master thesis, UNIVERSITY OF CALIFORNIA RIVERSIDE, 2010.

[20] Levi Scott Lindsey. *Pen-Based Interfaces for Intelligent Statics Tutoring Systems.* Master thesis, UNIVERSITY OF CALIFORNIA RIVERSIDE, 2013.

[21] Rachel Blagojevic, Beryl Plimmer, John Grundy, and Yong Wang. Using data mining for digital ink recognition: Dividing text and shapes in sketched diagrams. *Computers and Graphics (Pergamon)*, 35:976–991, 2011.

[22] Ron Stevens, David F Johnson, and Amy Soller. Probabilities and predictions: modeling the development of scientific problem-solving skills. *Cell biology education*, 4:42–57, 2005.

[23] Shubhendu; Trivedi, Zachary A; Pardos, Gabor N Sarkozy, and Neil T Heffernan. Spectral Clustering in Educational Data Mining. In *Educational Data Mining*, pages 129–138, 2011.

[24] Nan Li, WW Cohen, KR Koedinger, Noboru Matsuda, and Carnegie Mellon. A Machine Learning Approach for Automatic Student Model Discovery. In *Educational Data Mining*, pages 31–40, 2011.

[25] Sharon Oviatt, Alex Arthur, and Julia Cohen. Quiet interfaces that help students think. *Proceedings of the 19th annual ACM symposium on User interface software and technology UIST 06*, pages 191–200, 2006.

[26] Kevin Rawson and Thomas F. Stahovich. Predicting Course Performance from Homework Habits. In *Proceedings of the American Society for Engineering Education*, 2013.

[27] James; Herold, Thomas F; Stahovich, and Kevin Rawson. Using Educational Data Mining to Identify Correlations Between Homework Effort and Performance Using Educational Data Mining to Identify Correlations Between Homework Effort and Performance. In *Proceedings of the American Society for Engineering Education*, 2013.

[28] James Herold, Alex Zundel, and Thomas F. Stahovich. Mining Meaningful Patterns from Students Handwritten Coursework. In *EDM '13*, pages 67–73, 2013.

[29] James; Herold and Thomas Stahovich. CHARACTERIZING STUDENTS HANDWRITTEN SELF- Characterizing students handwritten self-explanations. In *Proceedings of the American Society for Engineering Education*, 2012.

[30] Timothy; Van Arsdale and Thomas F Stahovich. Does neatness count ? What the organization of student work says about understanding Understand correlation. In *Proceedings of the American Society for Engineering Education*, 2012.

[31] PCH Cheng and H Rojas-Anaya. Measuring mathematic formula writing competence: An application of graphical protocol analysis. *... of the Thirtieth Annual Conference of ...*, 2008.

[32] J R Et Al Hobbs. Description of the FASTUS system used for MUC-4. In *Proceedings of Forth Message Understanding Conference*, pages 268–275, 1992.

[33] Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. Algorithms That Learn To Extract Information BBN: Description Of The Sift System As Used For MUC-7. In *Message Understanding Conference (MUC-7)*, pages 1–17, 1998.

[34] Aron Culotta, Andrew McCallum, and Jonathan Betz. Integrating Probabilistic Extraction Models and Data Mining to Discover Relations and Patterns in Text. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, pages 296–303, 2006.

[35] Michele Banko, MJ Cafarella, and Stephen Soderland. Open information extraction for the web. *IJCAI*, pages 2670–2676, 2007.

[36] Stephen Soderland, Brendan Roof, Bo Qin, and Shi Xu. Adapting Open Information Extraction to Domain-Specific Relations. *AI Magazine*, 31:93–102, 2010.

[37] Nicholas M Rhodes, Matthew A Ung, Alexander E Zundel, James Herold, and Thomas F Stahovich. Using a Lexical Analysis of Students Self-Explanation to Predict Course Performance. In *Educational Data Mining*, 2013.

[38] S.E. Robertson and Karen Spärck Jones. Relevance Weighting of Search Terms. *Journal of the American Society for Information Science*, pages 129–146, 1976.

[39] Levent Burak Kara and Thomas F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Computers and Graphics (Pergamon)*, 29:501–517, 2005.

[40] Mark Hall, Hazeltine National, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The WEKA Data Mining Software : An Update. *SIGKDD Explorations*, 11:10–18, 2009.

[41] Jim Herold and Thomas Stahovich. A machine learning approach to automatic stroke segmentation. *Computers & Graphics*, pre-print:1–8, 2013.

[42] L R Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition, 1989.

[43] James Herold, Thomas Stahovich, Han-lung Lin, and Robert C Calfee. AC 2011-2253 : THE EFFECTIVENESS OF PENCASTS AS AN INSTRUC- TIONAL MEDIUM The Effectiveness of Pencasts as an Instructional Medium. In *Proceedings of the American Society for Engineering Education*, 2011.

[44] Jacob O Wobbrock, Andrew D Wilson, and 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 UIST 07*, 85:159, 2007.

[45] Han-lung; Lin, Thomas; Stahovich, and James Herold. AC 2012-4934 : AUTOMATIC HANDWRITTEN STATICS SOLUTION CLAS- SIFICATION AND ITS APPLICATIONS IN PREDICTING STUDENT PER- Automatic Handwritten Statics Solution Classification and its Applications in Predicting Student Performance. In *Proceedings of the American Society for Engineering Education*, 2012.

[46] Cristóbal Romero and Sebastián Ventura. Educational data mining: A review of the state of the art. *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews*, 40:601–618, 2010.

[47] Tevfik Metin Sezgin, Thomas F Stahovich, and Randall Davis. Sketch Based Interfaces: Early Processing for Sketch. *ACM SIGGRAPH 2006 Courses*, page 22, 2006.

Appendix A

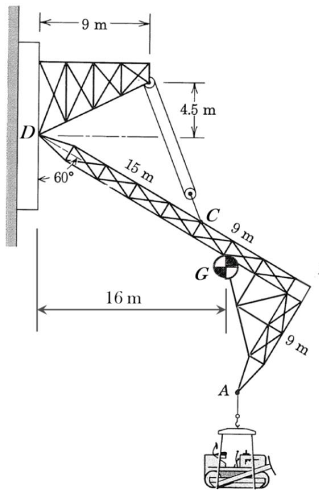
Set C - Exams

Features	**Description**
F_P	No. of long pauses
F_E	No. of equations
F_D	No. of digits
F_L	No. of letters
F_M	No. of mathematical symbols
F_Σ	No. of Σ
F_C	No. of characters
$F_{D/L}$	Ratio of F_D to F_L
$F_{D/M}$	Ratio of F_D to F_M
$F_{L/M}$	Ratio of F_L to F_M
F_U	No. of units

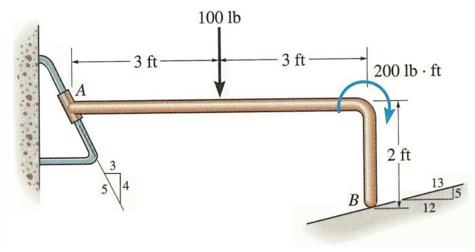
Features	**Description**
F_{DD}	No. of pattern DD
F_{DM}	No. of pattern DM
F_{DL}	No. of pattern DL
F_{LD}	No. of pattern LD
F_{LM}	No. of pattern LM
F_{LL}	No. of pattern LL
F_{MD}	No. of pattern MD
F_{MM}	No. of pattern MM
F_{ML}	No. of pattern ML
$F_{=D}$	No. of pattern $=D$
F_{DMD}	No. of pattern DMD
F_{DML}	No. of pattern DML
F_{LMD}	No. of pattern LMD
F_{LML}	No. of pattern LML

M1P1: The crane is hoisting a 4000 kg bulldozer. The mass center of the 2000 kg boom is located at *G*. The system is in equilibrium in the configuration shown. In your analysis, neglect the width of the boom.

a) Draw a large, clearly-labeled free body diagram.
b) Determine the tension ***T*** in the cable where it attaches at *C.*
c) Determine the magnitude of the force applied to the boom at its hinge *D*.



M1P2: Determine the reactions on the bent rod which is supported by a smooth surface at *B* and by a collar at *A*. The rod is fixed to the collar, which prevents the rod from rotating in the plane, but allows the rod to slide freely over the fixed inclined rod. Your solution must include a large, clearly-labeled free body diagram. Your answers should include both magnitudes and directions. Neglect the weight of the bent rod.



M1P3: Force ***F*** is applied to point *E* which is located 60 mm above the plane containing the centerlines of bearings *A, B, C,* and *D*. Force ***F*** lies in a plane parallel to the *x-z* plane and is inclined 15° from the *z*-axis as shown. 900 N forces are applied to bearings *C* and *D*. Neglect the weight of the structure.

a) Draw a large, clearly-labeled free body diagram.
b) Determine the magnitude of force F required to maintain equilibrium.
c) Determine the radial and axial components of the forces in bearings A and B. Assume that B does not support a thrust force.

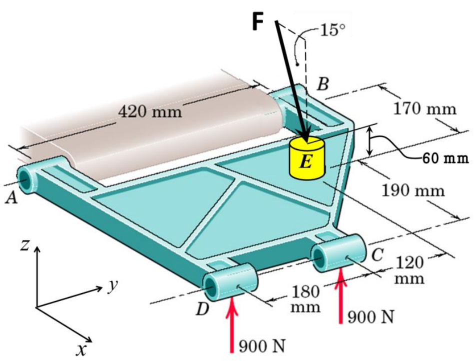
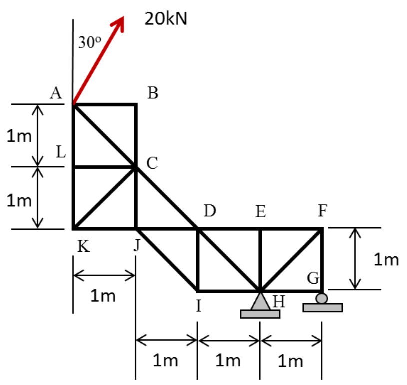


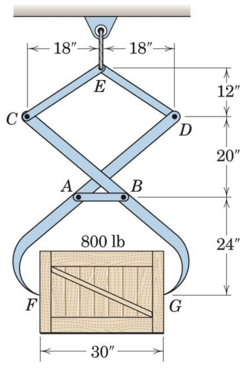
Figure A.1: Problem M1P1, M1P2, and M1P3.

M2P1: Determine the forces in members *CD*, *JI*, and *DI* and indicate if they are in tension or compression.



M2P2: The 800-lb crate is held in equilibrium by the lifting device.

(a) Determine the forces in members *CE* and *ED* and indicate if they are in tension or compression.
(b) Determine the force in member *AB* and indicate if it is in tension or compression.



M2P3: The tractor shovel carries a 600-kg load of soil, having a center of gravity at G. EJ = 100mm. JH = 100mm.

(a) Identify all of the bodies that are two-force members. List them by the points on the bodies, e.g., “DFG”.
(b) Determine the force in hydraulic cylinder *BC* and indicate if it is in tension or compression.
(c) Determine the magnitude of the force acting on the shovel at point *F*.
(d) Determine the force in hydraulic cylinder *IJ* and indicate if it is in tension or compression.

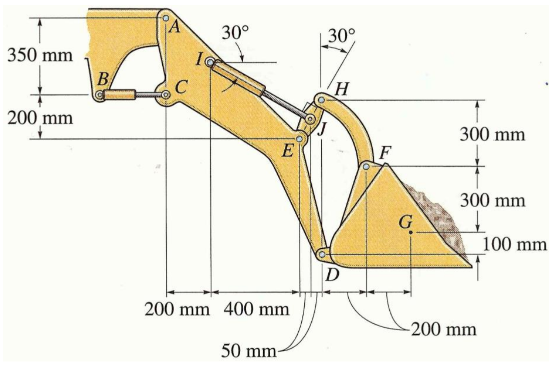


Figure A.2: Problem M2P1, M2P2, and M2P3.