2-2014

Automated Learner Classification Through Interface Event Stream And Summary Statistics Analysis

Edgar E. Troudt
Graduate Center, City University of New York

How does access to this work benefit you? Let us know!
Follow this and additional works at: https://academicworks.cuny.edu(gc_etds
Part of the Computer Sciences Commons, and the Instructional Media Design Commons

Recommended Citation
CUNY Academic Works.
https://academicworks.cuny.edu(gc_etds/118

This Dissertation is brought to you by CUNY Academic Works. It has been accepted for inclusion in All Dissertations, Theses, and Capstone Projects by an authorized administrator of CUNY Academic Works. For more information, please contact deposit@gc.cuny.edu.
AUTOMATED LEARNER CLASSIFICATION THROUGH INTERFACE EVENT STREAM AND SUMMARY STATISTICS ANALYSIS

by

Edgar E. Troudt

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy, The City University of New York

2014
This manuscript has been read and accepted for the
Graduate Faculty in Computer Science in satisfaction of the
dissertation requirement for the degree of Doctor of Philosophy.

Dr. Danny Kopec

Date

Chair of Examining Committee

Dr. Robert M. Haralick

Date

Executive Officer

Dr. Matt Huenerfauth, Queens College

Dr. Paula Whitlock, Brooklyn College

Dr. Christina Schweikert, St. John’s University
Supervisory Committee

THE CITY UNIVERSITY OF NEW YORK
Abstract

AUTOMATED LEARNER CLASSIFICATION THROUGH INTERFACE EVENT STREAM AND SUMMARY STATISTICS ANALYSIS

by

Edgar E. Troudt

Adviser: Professor Danny Kopec

Reading comprehension is predominately measured through multiple choice examinations. Yet, as we will discuss in this thesis, such exams are often criticized for their inaccuracies. With the advent of "big data" and the rise of ITS (Intelligent Tutoring Systems), increasing focus will be placed on finding dynamic, automated ways of measuring students' aptitude and progress.

This work takes the first step towards automated learner classification based on the application of graphic organizers. We address the following specific problem experimentally: How effectively can we measure task comprehension via human translation of written text into a visual representation on a computer? Can an algorithm employ data from user interface (UI) interaction during the problem solving process, to classify the user's abilities? Specifically, from the data we show machine learning predictions of what a human expert would say about the:

1. integrity of the visual representation produced;
2. level of logical problem solving strategy the user applies to the exercise;
3. level of effort the user gives to the exercise.

The core of the experiment is a software system that allows a human subject to read a preselected text and then "draw" a diagram by manipulating icons on a grid-canvas using standard transforms.
Acknowledgments

A doctoral thesis takes perseverance on the part of the author and a strong support network. I wish to thank the many people who had a part in this process:

- Drs. Audant, Gomez, Katz, Peruggi, Schulman and Suss created an environment at Kingsborough Community College where I had the intellectual support and time to build this work. Your constant encouragement to complete the project kept me moving.

- CUNY Language Immersion Program (CLIP) Instructors Sunderlin and Teter, and CLIP Director Milano provided the educational materials and subject pools for the experiment. Their expertise made the experimental process move smoothly.

- Technical support was crucial. Jason Liu provided skilled coding for the experiment’s interface in a professional and timely manner. Kingsborough CIO Asif Hussain provided the technical resources of the College so that conducting the experiment could become a reality.

- Dr. Kopec, my supervisor, and the committee comprising Drs. Huenerfauth, Schweikert and Whitlock, provided invaluable insights into the project’s design and the interpretation of the results.

- And, a special thanks to Gillian, who provided the emotional support to keep moving through the highs and lows of this process.

Without these individuals, this work would not have been possible. I owe each of you a debt of gratitude.
# Table of Contents

Chapter 1 – Background and Literature ......................................................... 1

1.0 This Work ........................................................................................................... 1

1.1 Focal Works in Human-Computer Interaction ............................................. 4

1.2 Background and Motivation: Education ....................................................... 8

1.3 Successful HCI Techniques and Features .................................................. 15

1.4 Preliminary Work Performed by the Investigator ...................................... 22

Chapter 2 – The Study Design ........................................................................ 24

2.0 Focal Questions and Study Contributions .................................................. 24

2.1 Design of Diagram Manipulation System ..................................................... 27

2.2 Automated Observation of Input Interaction Features ............................... 29

2.3 Subjects and Data ............................................................................................ 31

Chapter 3 – Implementation and Analysis Techniques .................................. 33

3.0 Introduction .................................................................................................... 33

3.1 Designing and Implementing the Software .................................................. 33

3.2 Recruiting a Subject Pool and Developing Experimental Content ............... 34

3.3 Software Testing and Deployment ................................................................. 36

3.4 Administration of the Experiment ................................................................. 37

3.5 Subjective and Objective Experimental Data ............................................... 38

3.6 Secure Storage and Data Preparation ......................................................... 40

3.7 Weka Pass 1 – Algorithm Selection ............................................................... 41

3.8 Weka Pass 2 – Feature Segmentation ........................................................... 42

Chapter 4 – Lessons from the Experimental Results .................................... 49

4.0 Organization of the Analysis ......................................................................... 49

4.1 Analysis of Subjective Outcome Assessment: On Task ............................... 50

4.2 Analysis of Subjective Outcome Assessment: Logical ............................... 52

4.3 Analysis of Subjective Outcome Assessment: Key Elements ...................... 55
<table>
<thead>
<tr>
<th>Chapter/Section</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.4</td>
<td>Analysis of Subjective Outcome Assessment: Precision</td>
</tr>
<tr>
<td>4.5</td>
<td>Analysis of Objective Grade Data: Michigan Test</td>
</tr>
<tr>
<td>Chapter 5</td>
<td>Broader Conclusions for Future Work</td>
</tr>
<tr>
<td>5.0</td>
<td>A Holistic View of the Results</td>
</tr>
<tr>
<td>5.1</td>
<td>Movement Features</td>
</tr>
<tr>
<td>5.2</td>
<td>Distance Features</td>
</tr>
<tr>
<td>5.3</td>
<td>Timing Features</td>
</tr>
<tr>
<td>5.4</td>
<td>The Self-Assessment and Multiple Choice Questions</td>
</tr>
<tr>
<td>5.5</td>
<td>Icon-Level Features</td>
</tr>
<tr>
<td>5.6</td>
<td>For Future Experiments</td>
</tr>
<tr>
<td>Appendix A</td>
<td>Database Structure</td>
</tr>
<tr>
<td>Appendix B</td>
<td>Participants’ Subjective Self-Assessment</td>
</tr>
<tr>
<td>Appendix C</td>
<td>Extensions to Game Table for DMS</td>
</tr>
<tr>
<td>Appendix D</td>
<td>Tools to Aide Software Development</td>
</tr>
<tr>
<td>Appendix E</td>
<td>Data Pre-Processing Tools</td>
</tr>
<tr>
<td>Appendix F</td>
<td>Sample Weka ARFF File</td>
</tr>
<tr>
<td>References Cited</td>
<td>References Cited</td>
</tr>
</tbody>
</table>
# List of Tables

Table 2.1: Computed input interaction features that were collected for the use of the machine learning classifier. p. 30
Table 3.1: Rubric for subjective outcome assessments on experimental drawings. p. 39
Table 3.2: Feature distribution across Cases 1 through 4. p. 45
Table 3.3: Varying feature distribution in Case 5 for each of the subjective outcome assessments. p. 46
Table 3.4: Binary thresholds for subjective outcome assessments. p. 47
Table 3.5: Classification accuracy of feature sets for subjective outcome assessments. p. 48
Table 3.6: Classification accuracy of feature sets for objective grade data. p. 48
Table 4.1: The most salient classifications from this experiment. p. 49
Table 4.2: Best and worst feature sets for On Task data. p. 51
Table 4.3: Salient feature selection in Case 5 for On Task data. p. 52
Table 4.4: Categories of features and their predictive qualities for On Task data. p. 52
Table 4.5: Best and worst feature sets for Logical data. p. 53
Table 4.6: Salient feature selection in Case 5 for Logical data. p. 54
Table 4.7: Categories of features and their predictive qualities for Logical data. p. 55
Table 4.8: Best and worst feature sets for Key Elements data. p. 56
Table 4.9: Salient feature selection in Case 5 for Key Elements data. p. 57
Table 4.10: Categories of features and their predictive qualities for Key Elements data. p. 58
Table 4.11: Best and worst feature sets for Precision data. p. 59
Table 4.12: Salient feature selection in Case 5 for Precision data. p. 61
Table 4.13: Categories of features and their predictive qualities for Precision data. p. 62
Table 4.14: Best and worst feature sets for Michigan data. p. 63
Table 4.15: Salient feature selection for Michigan data. p. 64
Table 4.16: Categories of features and their predictive qualities for Michigan data. p. 64
Table 5.1: Size of optimal feature sets by classification category. p. 65
Table 5.2: Predictive nature of feature types by classification category. p. 66
Table 6.1: Explanation of raw data collected in each database table. p. 75
List of Figures

Figure 1.1: Summary of scope of work and literature justification. p. 3
Figure 2.1: Top-level flowchart of study activities. p. 26
Figure 2.2: Flow-chart describing interaction with the Diagram Manipulation System. p. 27
Figure 2.3: The main interfaces of the Diagram Manipulation Suite. p. 29
Figure 3.1: Screenshots of the login and passcode functions of the software. p. 37
Chapter 1 – Background and Literature

1.0 This Work

Reading comprehension is predominately measured through multiple choice examinations. Yet, as we will discuss in this chapter, such exams are often criticized for their inaccuracies. With the advent of “big data” and the rise of ITS (Intelligent Tutoring Systems), increasing focus will be placed on finding dynamic, automated ways of measuring students' aptitude and progress. Repurposing HCI (Human-Computer Interaction) research to derive human intentions and understanding from their interface actions, presents an opportunity for classifying learners. CSE (Computer Science Education) researchers have already shown such kinesthetic and graphical manipulation suites as being useful to the teaching and assessment of CS concepts. This thesis builds upon that work by monitoring students as they manipulate graphics related to their readings.

The experiment conducted in this work takes only the first step towards automated learner classification based on the application of graphic organizers. The study demonstrates that during human translation of written text into a visual representation on a computer, a machine learning algorithm can employ data from user interface (UI) interaction during the problem solving process, to classify the user's abilities. Specifically, from the data we seek to predict what a human expert would say about the:

1. integrity\(^1\) of the visual representation produced;
2. level of logical problem solving strategy the user applies to the exercise;
3. level of effort the user gives to the exercise.

The core of the experiment is a software system that allows a human subject to read a preselected text and then “draw” a diagram by manipulating icons on a grid-canvas using standard transforms. Chapter 2 describes the system that logs the subject's actions. Chapter 4 describes the performance of a machine

---

\(^1\) For the purposes of this study, we define “integrity” as the inverse distance from being a semantically correct representation, as subjectively assessed by a human expert.
learning algorithm in classifying subjective ratings in the three aforementioned categories using features describing the user actions. While such work could eventually be used to indicate the human’s overall comprehension of the passage, it falls beyond the scope of the study.

The investigation is grounded in seminal work by two contemporary researchers: Louis von Ahn (2006, 2006a, 2006b) who has pioneered the art of having computers acquire non-computable knowledge through monitoring human interaction, and Ryan Baker (2007) who repurposed techniques in interface assessment for the understanding of students’ online-learning behavior. The work is grounded in the Human-Computer Interaction (HCI) sub-discipline of Computer Science. It also makes contributions to the design of Intelligent Tutoring Suites (ITS). Figure 1.1 summarizes the project and its grounding in relevant literature.
**Prevailing Research from Education**

1. Graphic organizers can have a positive impact on reading comprehension.
2. Interactive tutoring suites have gained popularity among educators because of positive effects and desire for more distance learning.

**Prevailing Research from Computer Science**

1. Innovative game-like environments can be designed to allow algorithms to derive useful, non-computable data from observing human behavior. (von Ahn)
2. ITS can understand user behaviors based on metrics studying their interaction. (Baker)
3. Robust HCI metrics and methodologies exist for studying users’ intentions when using software. While currently used for software testing, these methods can be repurposed for understanding the user’s intentions.
4. Learning CS through kinesthetic and graphical manipulation is well-studied with positive results.

**Research Shortcomings to be Answered**

1. Despite other teaching tools, multiple choice tests are still the prevailing method of mass-assessing reading comprehension.
2. There is a strong desire for automated classroom metrics that go beyond attendance and tests.

1. Much work on automated grading and feedback to students has focused on English essay text.
2. Baker’s research using HCI metrics applied within an ITS was limited to drawing interfaces only about “off-task behavior”.
3. von Ahn’s research uses games to provide information about an object but not about the player.

**Proposed Research Project**

Develop rudimentary environment to allow for monitoring users’ process of translating text to graphic. Show proof of concept that a relationship exists between user actions and quality of image produced or provide adequate experimental data to refute such a claim.

Demonstrating sound educational practice or the improvement of comprehension is beyond the scope of this study.

**Commercial Product**

Intelligent Tutoring suite that Educational Psychology research has shown demonstrably improves and assesses reading comprehension through having students translate text to graphic.

**Focus of Dissertation**

**Potential Future Work**

Figure 1.1: Summary of scope of work and literature justification.
1.1 Focal Works in Human-Computer Interaction

1.1.0 Extracting Information from Human Behavior

Louis von Ahn (Carnegie Mellon, Computer Science) uses games to provide computers with data that would otherwise not be easily computed. The key to design is having the humans step through a process that yields data on problems easily solved by humans; the added art is designing the process to be engaging. Matchin’ is a game where two players are shown pairs of pictures and asked to determine which is more attractive, points are scored when players agree; in the background, the computer is able to develop a hierarchy of beauty amongst the pictures (Thompson, 2007). Peekaboom also uses images, but through the process of one human providing clues to the other, the human is segmenting the elements of the image that are crucial to conveying the underlying meaning (von Ahn, 2006a). Verbosity is a Charades-type game, where one player is given an object noun, and the other has to guess it by crafting true or false descriptive questions; behind the scenes, the computer generates a database of “commonsense knowledge” related to the object (von Ahn, 2006, 2006b).

1.1.1 Off-Task Behavior Detection

Baker (2007) developed a user-log-based framework for detecting off-task and “gaming” behavior of students engaged with Intelligent Tutoring Systems (ITS), specifically the Cognitive Tutor. Off-task behavior includes anything that is irrelevant to the learning process, such as sleeping and having side conversations. “Gaming” behavior involves exploiting the system, such as abusing help functions and entering nonsensical answers to skip questions; though specifically not included is when students do this for their learning benefit, such as skipping easy questions to spend more time with challenging ones. The study was of high-school students in a mathematics classroom. It involved classroom observation of
student behavior, pre- and post- skills tests, collection of usage data from software logs, and attitudinal surveys.

Although Baker is not the only investigator to apply statistics to rate students in an e-Learning environment, the work embodies the notion of repurposing HCI methods for evaluating the user-participant as opposed to using these to assess the quality of the interface. Baker’s experimental design was elegant. And, based on both our own and the author’s sampling of the literature was the only work that made significant use of comparing individuals to the class as a whole, rather than evaluating individuals standalone. The works reviewed include those by Gude, Jackson and Shaw (2000), Junior and Filgueiras (2005), Lanzilotti, Costable and Ardito (2006), Matera, Costable, Garzotto and Paolini (2002), and Singley and Lam (2005) presented in the following sections.

1.1.2 HCI Evaluation Frameworks

The field of Human-Computer Interaction (HCI) has devised numerous methodologies for assessing interface usability. HCI evaluation generally falls into two forms: user-based methods, where a population is studied making use of the software, and inspection methods where experts directly evaluate software quality. While the work of this experiment is not concerned with the software development process nor the assessment of interface quality, as with Baker’s work many HCI methods that give insight into the meaning users’ actions may be repurposed to derive conclusion about the user based on their interactions in our work.
User-based methods often record user event streams\(^2\) to make conclusions. Hilbert and Redmiles (2000) built a comprehensive framework for categorizing and comparing HCI methodologies for extracting usability-related meaning from user events and use it to summarize prevailing techniques.

Evaluation with real users is often costly and requires numerous expert observers to watch over and note the progress of the users. Often to reduce costs, quality assurance (QA) specialists are used, although the disadvantage is that QA testers do not necessarily mimic the original populations.

Ardito, Costabile, De Angeli and Lanzilotti (2006) proposed that quality assurance teams evaluate interfaces using Abstract Tasks (ATs) – this notion breaks the interface into a set of discrete tasks each of which is independently reported on (including the specific focus of action and its outcome), allowing for a more rigorous review of the entire application. Ardito et al.’s proposal builds upon the work of Matera et al. (2002). The process for developing ATs is (1) observe expert quality assurance inspectors using the application, (2) define discrete sets of tasks based on the expert’s interaction, (3) allow less qualified evaluators to evaluate using the framework. ATs are particularly relevant to our work - Matera et al. (2002) and Lanzilotti et al. (2006) have shown that these are the sophisticated approach to multimedia and learning environment evaluations, respectively.

We close the gap between methods, using real test subjects but mitigating the need for numerous evaluators through the system recording features related to behavior; furthermore we embrace the AT methodology by breaking the experiment into discrete steps and noting what the user is working on, relevant to the behavioral features. Our experiment (as defined in Section 2.1) follows the AT design method, with a first round during which all features are evaluated, time for refining these features, then demonstrating that a model can be generated for using the features on another population.

Fisher and Sanderson (1996) defined a continuum of frequencies for events related to software usage.

\(^2\) User event streams are a series of user-to-interface actions, such as a mouse-click.
“High-frequency events” like mouse clicks and eye movements are generally used to study interface design, whereas "low-frequency events" like meetings and steps in a project are generally used to study competency at cooperative work. HCI researchers often break down high-frequency events into single letters, and describe event streams as long strings. The work of this project does the reverse, using the high-frequency events in understanding the user’s competency. Following Baker’s lead in comparing users against one another, we compare the user actions patterns across participants. (Further discussion of the events in our experiment’s stream is in Section 2.2; comparison is discussed in Section 2.4.)

The HCI subfield of predictive user modeling attempts to calculate user intentions on the fly through asking probing questions, often with appropriately timed notification boxes (Iqbal & Bailey, 2008; Horvitz & Apacible, 2003). Such a technique would be invaluable to determining when one is “confused” in an eventual commercial tutoring suite, however is beyond the scope of the current experiment. The work of Junior et al. (2005), on User Modeling with Personas suggests that by classifying users into distinct roles and mindsets, modeling their behavior becomes simpler. This concept can be applied to our assessment of the features. When users are grouped into classes based on their comprehension levels, it may be simpler to identify patterns. The reverse is also true, certain patterns of behavior may allow for easy classification of comprehension. (The implications of this to this experiment will be discussed in Section 2.4.)

1.1.3 Research Gaps

The works of von Ahn and Baker have the unified approach of using human actions with the program interface to derive data that would be otherwise incalculable for the computer. Von Ahn’s work uses game-like environments to learn about preexisting photos, phrases and scanned documents, but never has applied it to drawing conclusion about the human player. Both researchers make use of HCI theory, but both are novel in that its application is not the traditional evaluation and improvement of software interfaces. The investigator of this project continues in the direction of their work.
While Baker’s (2007) work took novel steps forward in making inferences about behavior from statistics, it is limited in both its input and derived conclusions. Baker primarily makes use of what Hilbert and Redmiles (2000) term *summary statistics*, and ignores the much larger HCI data-set of *online event streams*. Limiting the input to comparative statistics (and focusing heavily on features of the UI used as opposed to concrete actions) limits the conclusions to the simplistic conclusions of the user being on- or off-task. As will be discussed in Section 1.2.2 (Algorithms for Education Statistics), it would be far more useful to ITS if more sophisticated features, already used in HCI, could be applied to derive meaning about the quality of the users’ educational output. And, unlike Baker’s approach, which serves to inform after the run of a program, our approach seeks to be a framework that is more compatible with online, real-time analysis.

Following Baker’s experimental design, in the next section, we seek to define and ground an educational problem on which to focus our study. Following that, von Ahn et al.’s work will influence the interface design in Section 2.1.

### 1.2 Background and Motivation: Education

#### 1.2.0 Reading: A Dire Educational Challenge

Call it Generation Y, the Internet- or Digital-Generation, or just the Millennials, but by any name they crave connectivity, surround themselves in media and information, and embrace technology. The term refers to the group of young Americans born between roughly 1981 and 1993 (Deloitte, 2006). A

---

3 *Summary statistics* are aggregate data about the overall usage of a software suite (e.g., task times, duration and frequency of feature use, range of functions used, percentage of tasks completed (Sweeney, Maguire & Shackel, 1993)) provided upon the completion of use.

4 *Online event streams* are recorded sequences of actions in the user interface (UI).
byproduct of the embracement of technology is the widespread adoption of non-textual media for communication. It can be as simple as the emoticon attached to text-based messages to convey the feelings the text cannot. This embracement also appears in the photos that end up on Facebook, the videos that end up on YouTube, podcasts (audio blogs), and video blogs.

Perhaps as a result of their involvement with these other modalities of communication, formal language is the Millennial Generation’s point of weakness; this has become especially obvious as they have advanced to college level. Consider the combination of the pervasiveness of “leetspeak,” the online dialect of the English language, with the commonness of non-textual communication options – formal language skills are likely to suffer. “Leetspeak” phrases such as “lol” and “pwned” have matured to the point where in many online interactions, these take preference over the proper-English counterpart (Blashki & Nichol, 2005). “Pwned” was so overused that it was even added to the official “Banished Word List” of Lake Superior State University (2007) in Michigan. According to the US Department of Education’s National Assessment of Educational Progress (NAEP), overall performance in reading and mathematics among high-school graduates has declined to the lowest levels since 1992. With respect to the reading ability of the current generation, 35% are considered proficient (with the ability to make significant inference from text), and 73% are at a basic skill level. In mathematics, 23% are considered proficient and 61% basic. No gain has been made on the performance gap between White and Black students since 1992 (Grigg, Donahue & Dion, 2007). Sadly, since these statistics count only those who graduate, the full pervasiveness of the problem is undocumented.

Closer to home, Kingsborough Community College of the City University of New York echoes these trends. Of the student body that entered in 2006, 31.5% required some form of remedial English reading, 55.6% required some form of remedial English writing, and 54.9% required some form of remedial Math. Only slightly over half of those in remediation end up passing a proficiency exam after their coursework. The College services a predominantly immigrant population, just over half of its students are born outside of the United States (Fox, 2007).
1.2.1 Algorithms for Education Statistics

Classroom patterns have always been important to education. Singley et al. (2005), point out that after the advent of the “No Child Left Behind” legislation, the concept of statistical approaches to recognizing these patterns has entered the classroom. Singley and others have different data-mining and delivery methods, but all are derived from longitudinal performance data and demographic details. IBM, Chancery, and Microsoft have developed systems that bring such data to the instructor in report format. The goal is to have teachers use the data to refine pedagogy, as opposed to just for the assignment of grades (Singley & Lam, 2005).

Singley and Lam’s “Classroom Sentinel” (2005), and comparable suites, bring the alert data directly to the instructor. The desire is for an agile response in his or her teaching, as opposed to leaving it to administrators to get an aggregate report and merely deal with the worst cases. The data mined goes beyond short-term grades including long term grades, attendance and behavior issues. But the literature does not indicate an attempt to go beyond the summary statistics and look at actual problem solving behavior and understanding.

1.2.2 Teaching and Assessing Comprehension

Reading comprehension assessment often centers on multiple choice assessments, despite the promise of graphics and kinesthetic learning in reading education. There has been algorithmic work toward automating assessment, but it is primarily focused on higher-level composition as opposed to lower-level comprehension.

Advances have been made in teaching comprehension using kinesthetic and visual learning. Some exercises force students to engage in the RAP (read, ask questions, paraphrase) process (Katims & Harris, 1997). One such technique is to have students draw flowcharts illustrating the events of a story
with text-filled boxes connected in sequence; the kinesthetic version does this on index cards which can be rearranged at will (Douglas, 2009). For content areas, the recommendation is to bring in various forms of knowledge diagramming, including concept maps (Douglas, 2009a).

Gude et al. (2000) designed a reading comprehension study for these various non-traditional interventions, including graphic organizers (Venn diagrams, concept webs, timelines and story pyramids). The study lasted 16 weeks with students of various elementary-school ages in three schools across the state of Illinois. In two of the three schools the various graphic organizers were cited as having improved recall of details of textual passages. Among first graders just learning to read, these organizers increased recall of details, understanding of event sequences and the ability to make comparisons between objects within a reading. These young students expressed extreme satisfaction with and desire to continue such exercises.

Mathematica Policy Research, Inc. designed a more widespread and rigorous national study (James-Burdumy, Myers, Deke, Mansfield, 2006) for the US Department of Education that looked at the effects of teaching reading comprehension strategies to fifth graders in 100 schools nationwide from 2006 through 2008. Like Gude, the authors also note that “graphic organizers” are a strategy in education literature for teaching reading.

Despite the promise of graphics, multiple choice tests have, somewhat disappointingly, continued to be the standard language assessment technique. Both the Gude (2000) and Mathematica studies base their results on multiple-choice test performance (James-Burdumy, 2009). The current standard used by American Colleges for assessment are the ACT and TOEFL (Chodorow & Leacock, 2000), both of which evaluate comprehension at least partially through multiple choice questions. CUNY’s own entrance exam also assesses in this manner. Even leading commercial reading software packages use multiple choice for ultimate assessment, graphics are only as a teaching tool. Some examples:
• ABCTeach.com is a website for generating elementary-education level classroom worksheets. Among the “reading comprehension” interventions are forms for writing book reports and multiple choice exams (ABCTeach, 2009).

• Mindplay publishes RAPS360, an educational suite geared towards phonics and reading comprehension. Interventions such as eye-tracking and pronunciation exercises tackle the technical issues that hinder reading. But, ultimately reading is followed by multiple choice diagnostic tests. Two other products from the company – “Fluent Reading Trainer” and “My Reading Coach” – have similar assessments for students at other levels of comprehension (MindPlay 2009).

• Merit Software focuses on adult literacy and GED preparation software. Their suite illustrates stories to assist in comprehension but ultimately relies on multiple choice and fill in questions to assess (Merit Software, 2009).

• Blackboard, the leading platform for online education by annual revenue, allows an instructor to build exams that roughly comprise long/short English textual answers and various forms of multiple choice answers, including matching and ordering (Blackboard 2009).

As with the research on educational statistics presented in Section 1.2.1, some work in Computational Linguistics and Natural Language Processing has been focused on automating the assessment process, instead of making use of multiple choice questions.

Chodorow and Byrd et al.’s research centers on extracting meaning from natural language texts statistically (Chodorow, Byrd & Heidorn, 1985; Byrd, Calzolari, Chodorow, Klavans, Neff & Rizk, 1987). This methodology has evolved to form the basis for detecting syntactic (Chodorow & Leacock, 2000) and semantic (Burstein, Kukich, Wolff, Lu, Chodorow, Braden-Harder & Harris, 1998) quality. Pendergast (2006) described a similar, but much more rudimentary, algorithm to assess quality of participation in online discussion forums. Bayesian analysis has provided automated extraction of an essay’s thesis statement (Burstein, Marcu, Andreyev & Chodorow, 2001). This goes beyond simply providing a
statistically-based score, eventually allowing for providing crucial feedback – the lifeblood of educating students in proper composition.

Such techniques are both feasible (given how ripe English text is for statistical analysis and mining) and desirable (as demonstrated by research in alternative educational statistics). However, this line of work is predicated on working with a student with at least a US high-school level of English composition ability and thus is only valuable at the higher end of English literacy. Evaluating reading comprehension, especially prior to possessing writing capabilities is far more difficult. At Kingsborough, as at most other Colleges, English remediation does not fully tackle writing until reading has been perfected. Thus, with a growing non-native speaking population, the more rudimentary skill of reading comprehension is not sufficiently addressed by these methods.

1.2.3 Kinesthetic Learning from a CS Perspective

Kinesthetic learning has a long tradition of use in educational applications, especially those designed by Computer Scientists.

Papert, a Computer Scientist and protégé of educational psychologist Piaget, advocates active discovery and kinesthetic learning\(^5\) over passive absorption of knowledge, warning against computers that only push educational content and calling the classroom an “artificial and inefficient learning environment society was forced to invent” to make up for its exploratory deficiencies (Papert, 1980). His seminal example is the LOGO language which allows for the drawing of pictures through the manipulation of a simple identifiable object – a “turtle” (Papert, 1980).

Kinesthetic manipulation is popular for educational suites designed by Computer Scientists. College-level courses in areas such as Chemistry with SmartTutor (Harrow, Eckhardt, Kopec, Kobraek & Whitlock, 2007; 10.4314/CJCE.v11i3p102).

\(^5\) Kinesthetic learning is accomplished through motion and object manipulation (Papert, 1980).
Kopec, Whitlock & Kogen, 2004), and Object Oriented Programming (McLaren, Bollen, Walker, Harrer, Sewall, 2005) make use of this approach. Two more classic examples: In 1994, Resnick created StarLogo, extending Papert’s language to allow for the building of models of real life phenomena such as “bird flocks, traffic jams, ant colonies, and market economies” (StarLogo, 2009). Wilensky and Stroup (1999) followed with N-LOGO, a Java-based, network-adapted version of StarLogo, and HubNet that allows the N-LOGO simulations to be controlled by handheld devices, such as Texas Instruments TI-83 graphing calculators. Classroom math problems involving graphing are reduced to kinesthetic exercises.

McLaren et al.’s work in Object Oriented Programming education investigated learning through collaborative graphical manipulation. The group learning experiment performed had students in dyads collaboratively solve a graphically-based problem. Specifically, eight dyads comprised of sixteen Computer Science students were presented with parts of a car and were asked to relate them using classification (instance of) and composition (part of) links (McLaren et al., 2005).

Resnick (2007) ties many of the aforementioned ideas together in his paper “All I Really Need to Know (About Creative Thinking) I Learned (By Studying How Children Learn) in Kindergarten.” He notes that modern education is turning kindergarten from play to learning by rote, much like the rest of school. His thesis is that the opposite should happen, kindergarten-style learning should be infused through the rest of school. He summarizes the thought process of kindergarten children: “imagine what they want to do, create a project based on their ideas, play with their creations, share their ideas and creations with others, reflect on their experiences” which leads them back to imagining bigger and better ideas. This does not happen in most classrooms for older students.

In all, the use of kinesthetic exercises has been generally accepted within and beyond the Computer Science community as a method of assisting in comprehension. This pedagogical approach will form the basis for our experiment, as described in the following section.
1.2.4 Educational Focus and Approach

Given the prevalence of problems with reading and the availability of a College-age population with which to work, the investigator chooses to make this the educational focus of the experiment. Furthermore, the success of kinesthetic approaches is synergistic with the propensity of Millennials to gravitate toward interactive rich-media technologies. Finally, we wish to follow the trends in reading comprehension education on graphics and the desire for rich statistics that go beyond the mundane multiple choice test scores. Thus the investigation in this work is built around an interactive, visual software system that could be used to enhance reading. The system engages an important educational deficiency with a sound educational approach.

Studying the students’ interaction with said system would have indirect implications for the area of educational-statistics data-mining. We look beyond the simplistic attendance and disciplinary-file statistics that are overused, into real features collected about problem solving behavior. While Singley and Lam (2005) advocate the use of daily teacher alerts, alerts based on our data could eventually be real-time instead. The work is timely as well, with Community Colleges, among others, reporting growing class sizes and shrinking budgets, choosing to put more resources behind online courses (Selingo, 2009).

It should be noted that the educational outcomes of this project are not the primary focus. This is a study grounded in HCI, and the primary contributions will be made there. It is our desire, however, to ground the work in a socially-relevant problem.

1.3 Successful HCI Techniques and Features

As discussed in Sections 1.1 and 1.2, this experiment will put established HCI techniques and features into the non-traditional role of understanding student-users’ behaviors. This section reviews the techniques for inclusion in the study design.
1.3.1 Techniques

Hilbert’s and Redmiles’ (2000) framework made several observations which are followed in this investigation:

- Effective user-based analysis involves both formative and summative evaluation – as a guide to interface revision and the latter as an analysis of that revision. *In following this design for our experiment, there is a formative data collection phase that allows for the gathering of mass quantities of data. After data analysis, a second, summative phase will be used to focus on subsets of data to determine effectiveness.*

- A critical component of UI monitoring is balancing the level of abstraction so that it is abstract enough so that the evaluators are not deluged with data, but sufficiently detailed for drawing conclusions for the investigation. *Thus in our in our experiment we define a series of distinct UI actions, and then base features on these. The literature supports the notion of monitoring at this level, a layer above the individual clicks and drags. This is especially true since we are concerned with the problem solving behavior, rather than specific interface quirks. Details are described Section 2.2.*

1.3.2 Baker’s Off-Task Detection

In Baker’s (2007) study, described in Section 1.1.1, time-on-task was found to be a good single-variable feature. Whereas many previous studies used extremely-short response time as an indication of an inconsiderate, gaming response, Baker argues that taking too long is a good measure of off-task behavior. In this model, the mean time-on-task for all students across all questions is computed; any student with more than 3.8 standard deviations longer than that time is considered delinquent. While proving effective for detecting off-task behavior, this single-feature scheme could also erroneously consider delinquent those students who are immersed in serious thought or in problem-related discussion with peers or the teacher.
Multivariate detection schemes were also investigated. Baker tried various combinations of statistics and kept those features that produced results similar to those of the classroom observers. He examined behaviors such as quick-paced performance in proximity to slow-paced performance, time on task, frequency of help requests, and making errors on tasks at which students are known to be strong.

From the surveys, Baker postulated the causes of students gaming and acting off-task while using ITS systems. Disliking computers or the subject at hand caused either behavior. Passive aggressiveness was linked with off-task behavior. Lack of self-drive and disliking the tutoring system were reasons to game the system. Surprisingly, students did not vacillate between the two behaviors; overall they were quite consistent with their chosen form of avoiding learning. Baker suggested that a rewards system is a better solution to this problem than a punitive system that notifies the user that it is aware of the delinquencies. A sample reward is offering more challenging problems to students who stay on-task (Baker, 2007).

1.3.3 Other HCI Features

Singley and Lam’s (2005) work in data-mining for monitoring student performance yielded some useful features:

- **Short-Term Trends**: The notion of “inconsistent achiever”, when a student performs differently than their norm, flags the student for further scrutiny.
- **Long-Term Trends**: The student is flagged if behavior nearly consistently improves or degrades over time. For good analysis of behavior, the investigators suggest that statisticians “work backwards” using short-term performance data to detect problems and “work forwards” using long-term predictors to prep the algorithm for the type of student. *The use of long-term trends poses an immediate problem to our work: the student will not make extensive use of the system. However by making use of College grade data and quality observations, we can simulate the long-term predictors and the short-term performance data. Section 2.3 discusses the selection of this objective data.*
Hirschman, Light, Breck and Burger’s (1999) experiments involving the evaluation of visual diagrams (called DeepRead, described in the next section) suggest the basic assessment features of “recall” and “precision”. Such features would be useful in revealing patterns the short- and long-term trends of student performance. These features also form the basis of the pilot study described in Section 1.4.

Ardito et al. (2006) discussed a systematic method of evaluating e-Learning systems. In their review of the literature, those authors assert that the number of studies addressing e-Learning system interface usability is small with vague statistical features and is often highly subjective. The authors suggest that interface evaluators in this area return to three basic features:

- **Effectiveness** is the number of problems discovered by an inspector related to the severity of the problem.
- **Efficiency** is time expended on task.
- **Satisfaction** is a self-reported, subjective measure of the end result.

The efficiency feature echoes Baker’s (2007) choice of “time-on-task”, and will also be adopted in our investigation. The notion of effectiveness can be simulated by attempting to uncover how many times a student undoes his or her own actions. (In this project we term this as “re-manipulation”.) Satisfaction is similar to Baker’s use of surveys; students in our experiment express their satisfaction with their performance on a Likert scale.

MIKE (Olsen & Halversen, 1988) is a system that allows the investigator to ask about many summary statistics, most importantly time on task, mouse travel, frequency of command use, command correlation (which commands are used together), cancelled and undone actions, and physical device swapping (between keyboard and mouse). Time-on-task is an oft repeated statistic throughout the literature. From this work we draw the summary statistics of distance travelled and undone actions for our experiment.

### 1.3.4 Useful Results from Relevant Experiments

Baecker, DiGiano and Marcus (1997) studied alternative forms of presenting computer algorithms. Anecdotally, visual animations of algorithms appear to improve students’ comprehension of program
processes. The authors developed the classroom video *Sorting out Sorting* and other sorting animations. Among the salient features of their illustration are: focusing on showing only the data that is crucial at each algorithmic step; simultaneous comparisons of similar algorithms; adopting consistent visual conventions; adding a music track to convey the “feeling of what is going on”; and, narrations in sync with motion. The claim is that their thirty minutes of video covers as much material as a thirty page textbook chapter. Baecker et al. also showed that neatly formatting code improves students’ ability to read, using the *SEE Visual Compiler*, a print preprocessing system.

*LogoMedia* is Baecker et al.’s software development environment – allowing for the attachment of MIDI-based sounds and basic visualizations to running software. In its most sophisticated use, programmers can assign different sounding instruments to variables and monitor the changes to those variables by hearing the instrument play at different pitches. (For example, an infinite loop might have a saxophone play down the scale until the loop becomes stuck at a value, then the saxophone would repeatedly output the same note.)

Baecker et al. claimed that auditory representations of code assist in debugging. *LogoMedia* was tested on a sample group of programmers. The programmers spent two hours learning the software, two hours using it to write their own code, and two hours using it to debug unknown code – during the last two, the subjects were asked to “talk aloud” about their thought processes. In all, the test group used the auditory flags in more than half of their test runs. Subjects were generally creative, using sounds, such as explosions and clicking, that melded well with the meaning of a particular code section. Invariably the subjects’ vocabulary would shift to describe problems by the sound it made.

The common conclusion across Baecker et al.’s research is that engaging a reader with more than one form of representation (through visual and aural augmentation) improves the process of reading, debugging and understanding written code. Although not exactly the manipulation of physical objects, the argument bears a striking resemblance to those examples of kinesthetic learning from Section 1.2.3. Thus, we expect similar results when applied to the comprehension of English text.
A lesson may be borrowed from Hirschman et al.’s (1999) simple experimental design in testing DeepRead, an answer system and platform for testing NLP algorithms. The system was provided passages; its task was to select the sentence from the reading that best answered related short-answer reading comprehension questions. That work made use of simple, commercially-available reading banks that have been reviewed for quality, ranked by difficulty, and marked up with comprehension questions allows the experiment to have a standardized assessment that controls for skew across different tests. Standardization in assessment allows the investigators to avoid outside biases and the various sophisticated assessment techniques falling outside of Computer Science in the domain of Educational Psychology. Thus the focus remains on the systems’ algorithms. In the experiment described in this thesis, we do not use a commercial reading bank. Instead, the investigator engages an instructor with knowledge of the participants’ reading levels to provide passages that are at an appropriate difficulty level.

McLaren et al.’s work on collaborative graphical manipulation (described in Section 1.2.3) had each teams’ students were assigned two workstations on opposite sides of a desk; communication was facilitated only through chat-messages and visual manipulation within the application. One student was responsible for composition, the other for classification. It is this simplicity of interface that will guide the design of the application (Section 2.1). Furthermore, the separation of tasks allowed better conclusions to be drawn; this separation is reflected in our study design, which separates drawing experiments from multiple choice questions so that a single task is presented at a time to a subject. (See Section 2.0 for more details.)

At the start, half the teams received an electronic whiteboard where the car parts were organized; the other half received the same part-set but randomly organized. While the initial organization of the parts had little effect on the results, the teams that were successful in building the proper relationships were also methodical in their reordering of the parts. These successful groups also had more visual
communication than textual. Groups with poor solutions spent too much time discussing future steps without concrete actions (McLaren et al., 2005).

A framework for assessing students’ collaborative work in such an environment was laid out. The factors that were examined are: task coherence (appropriateness of actions), task coordination (ability to agree upon a strategy), task selection (ability to create logical subgoals to achieve a master step), conceptual understanding (understanding the underlying working of the problem), visual organization (ability to arrange shapes logically to complete the task) (McLaren et al., 2005). Many of McLaren et al.’s features have an analogue in Baker’s (2007) experiments. Conceptual understanding is a combination of Baker’s counting of help-requests and time-on-task. Task coherence is related to Baker’s count of errors. These features are also important to our study.

Carberry, Elzer and Demir (2006), built an informal taxonomy of modes of communication in documents. The primary three methods are: (1) text, (2) information graphics (a depiction of attributes of or relations between entities), and (3) pictorial graphics (snapshots of a scene). Their work is based upon a review of 100 information graphics randomly selected from articles in mainstream media sources (Carberry et al., 2006). The inspiration for Carberry et al.’s work was that of Kerpedjiev and Roth (2000), who study the reverse process – automated generation of graphs from data and communicative goals. Kerpedjiev and Roth’s (2000) generation framework is firmly grounded in computer graphics and cognitive science.

Carberry et al. and Kerpedjiev and Roth’s works provided effectiveness (simplicity of drawing) and cognitive economy (simplicity of understanding) as possible features about information graphics. These features will be repurposed in our study not to classify existing mainstream drawings, but to assess those produced by the amateur subject. These features bear striking resemblance to those of “effectiveness” and “efficiency” that Ardito et al. (2006) applied to HCI evaluation.
1.4 Preliminary Work Performed by the Investigator

Several preliminary steps have been taken in preparation for the design of this study:

- In January of 2007, the investigator, Edgar Troudt, joined the Scholarship of Teaching and Learning (SoTL) research group at Kingsborough Community College. The group began reading papers related to best practices in classroom pedagogy. Dr. Connie Schroeder, of University of Wisconsin at Milwaukee, conducted a two-day training on research tools and methodology for studying pedagogy.

- In February of 2007, IRB approval was secured for “SoTL Study of Visualization on Learning” (Kingsborough Community College, proposal #203), the precursor to this project. In connection with the study, Troudt underwent and passed the CITI human subjects training.

- Troudt posed the questions: (1) What is the impact of visualization projects on students' understanding and retention of case studies?, (2) Do visualization exercises increase a student's recall and understanding of case study materials? These primary assessment features were based on the work of Hirschman et al. (1999). During the Spring 2007 semester (spanning March-June 2007), and the Fall 2007 semester (spanning September-December 2007), students at various reading levels in a reading-intensive case-studies course were asked to develop pencil and ink diagrams organizing data presented in four of eight case studies in a meaningful manner.

- As part of the course, students were taught about Concept Mapping; though they were free to choose or ignore this format for their diagrams.

- Students were debriefed via survey and informal discussion. The investigator was interested in understanding both the students' preferences on the construct of the engagement, the types of visual representations, and whether students perceived this task as useful.

This preliminary work should be seen as a pilot of the process as opposed to a generator of empirical study data. Students submitted diagrams in a variety of formats, from the highly structured concept map to the freeform drawing with and without textual labels. Freeform drawings were the most popular; students felt more comfortable performing these.
Students’ general reactions to the work were that the process of drawing reinforced the reading, forcing them to carefully review the materials in the case study to organize their thoughts. Attitudes indicated that students were accepting of performing such work. Though, the overwhelming majority of the students wanted a structured description describing how the final deliverable should look.

The pilot study gave the investigator the indication that: students were willing to be engaged in this activity; expectations for output should be made clear with only a limited set of options; and, visual scene descriptions have low overhead and high comfort among community college students.
Chapter 2 – The Study Design

2.0 Focal Questions and Study Contributions

The works of Baker (2007), von Ahn et al. (2006, 2006a, 2006b), and McLaren et al. (2005) demonstrate that a wealth of information about the user can be derived from observing human interactions within a simple interface so long as there is a well-defined activity. Further, there is a strong desire from Computer Scientists and educators to assess comprehension skills and provide useful, non-test based, features to classroom instructors to guide interventions.

At its core, the study presents a human subject with a text passage and a software-based Diagram Manipulation System (DMS) allowing him or her to create a visual representation of the passage. The study demonstrates that during human translation of written text into a visual representation on a computer, a machine learning algorithm can employ data from user interface (UI) interaction during the problem solving process, to classify the user's abilities. Specifically, from the data we seek to predict what a human expert would say about the:

1. integrity of the visual representation produced;
2. level of logical problem solving strategy the user applies to the exercise;
3. level of effort the user gives to the exercise.

To that end, the following questions guide our conclusions in Chapter 4:

4. Can users be divided into classes based on diagram-integrity identified solely using a combination of trends in input interaction features?\(^6\)
5. What computer-observable actions translate into viable information about the integrity of the produced diagram and the strategy and effort used to produce it?

\(^6\) The Investigators define “trends in features” as similar patterns across two or more data points. We define a “class of users” as a group of users that have similar experimental outcomes (such as the group of users that produces “high quality diagrams”, or scores high on comprehension tests).
This study makes numerous contributions to Computer Science and Human Computer Interaction:

1. A process by which other researchers’ input interaction features may be tested, including a novel game-like engagement has been designed and successfully run with a small group. (Chapter 2)

2. A set of measurable features covering actions for student readers using a drawing suite are developed. Some of these features are novel, others are extensions of those identified in the literature. (Chapter 2)

3. A rubric for scoring drawings by the integrity, logic and effort is presented. The rubric includes several subjective categories. (Chapter 3)

4. Our method for collecting, securing, and pre-processing HCI data is presented. This effort includes several utility programs that were developed during the course of this thesis. (Chapter 3)

5. Input interaction features are identified by their usefulness to the behavior prediction process. (Chapter 4) The value of the features are presented with possible lessons for future research. (Chapter 5)

After the basic setup, the study process followed three phases: (1) data collection via experiment; (2) machine learning of user classifications; and (3) analysis. To prove our hypothesis we relied on machine learning algorithms in phase 2 (Chapter 3) to analyze the robust set of features from phase 1 that recorded subjects’ actions in the DMS. Figure 2.1 describes the experiment’s process.
During the **Experimental Design** phase the Diagram Manipulation System – the software system that allows for the manipulation of visual representations – was developed; section 2.1 discusses this in detail. Ancillary to the software design was the selection of reading passages and input interaction features. Section 2.2 discusses the selection of automated observational features. Section 2.3 discusses the selection of the subjects and the collection of additional objective and subjective data needed to draw conclusions discussed in Section 2.4. Texts and icon banks used are described in Chapter 3.

The core experiment – the subjects' interaction with the DMS – occurred in Phase 1. The human subject was asked to read a preselected text then “draw” a diagram using icons, a grid-canvas, and standard transforms. Some of the drawing activity was interleaved with a no-interaction control experiment, where subjects will merely answer the comprehension questions attendant to a passage. This system provides the observed input interaction features for the study. Figure 2.2 describes the operations of the DMS.
Each student interacted with six passages in total, three as control experiments and three as drawing experiments. Each passage was accompanied with multiple-choice comprehension questions. The order of the stories, as well as which stories are in the control group was randomized per student. The former was to mitigate the effects of practice and fatigue on particular stories. The latter was to prevent possible differences in the difficulty levels of stories from skewing the results.

### 2.1 Design of Diagram Manipulation System

At the core of our experiment is the diagram manipulation system (DMS). In order to discretize the number of possible diagrams and actions, we restricted the abilities of the user. No free form drawing was allowed; instead users dragged their choice of icon from a bank into a square of the canvas’ grid. The grid is a static eight-by-eight and the choices of icons were pre-selected, some of which did not match the meaning of the passage. Basic transforms were allowed to be performed on the icons once placed: rotation (in 90-degree increments), movement, and sizing. All icons were black-and-white, so color was not a confounding factor. Figure 2.3 depicts the primary interface.

Choosing an existing application on which to base our software was prudent. Doing so avoided the great overhead involved in designing and implementing a DMS from scratch. And, since the focus of the work
is on the recording and study of user actions and not on developing a commercial product, it is not necessary to have a novel interface. The investigator examined numerous concept mapping and whiteboard collaboration suites. Many existing systems were either closed-source or rich in features that would only be a distraction to students in this study. We look to the simplicity of von Ahn’s game interfaces (von Ahn et al., 2006, 2006a) and to McLaren’s experimental style (McLaren et al., 2005), which limited students to a canvas and a chat-box.

Game Table was an ideal candidate on which to base the software. It is a gamers’ mapping program designed to allow users to draw and share icon based maps. The software allows for placement of and basic transformations on “pogs”, implemented as draggable icons. The full source code is made available by the project team, making modification both legal and feasible. The software is coded in Java, lending itself well to being integrated into a networked lab environment (Sourceforge, 2014).

The process for development of the software for our study took the following course:

1. Developed a centralized database for tracking users, logging actions, storing reading texts and associated drawing icons, and storing students’ representations. (Database design is presented in Appendix A.)

2. Stripped the Game Table application of functionality beyond the requirements of the study. Added additional features, such as transforms on icons. Added an on-screen version of the passage texts with a scrollbar, so that reading effort could be monitored.

3. Developed a server-based control application that enforced the study’s flow, including the management of the game to be played, and the central logging of the actions taken.
Note: Left, is the drawing interface. Right, is the multiple choice question interface. Our software is a modified version of the open-source Game Table suite (Sourceforge, 2014). Figure 2.3: The main interfaces of the Diagram Manipulation Suite.

2.2 Automated Observation of Input Interaction Features

In order to build an effective algorithm, we needed to record (Data Collection and Experiment phase) and identify (Machine Learning of User Classifications phase) patterns of behavior in problem solving.

As discussed in Chapter 1, the features chosen for this experiment go beyond the summary statistics made use of by Baker (2007). We reviewed actions at occurrence at the more rudimentary level of clicks and drags, but still at a high enough abstraction so as to be able to see patterns in meaning.

The drawings produced during the experiment were subjectively graded (see Section 2.3). These subjective scores (see Chapter 3), plus objective grade data, were correlated with the logs of the captured user-actions to determine which features are useful to our experiment. Table 2.1 details these features.
<table>
<thead>
<tr>
<th>Name of Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Summary of Move Counts per Experiment</strong></td>
<td></td>
</tr>
<tr>
<td>1 MOVCmoveType</td>
<td>MOVE COUNT: number of icon manipulations involving movement</td>
</tr>
<tr>
<td>2 MOVCremanipulations</td>
<td>MOVE COUNT: number of re-manipulations, defined as moving an icon that already existed on the canvas but was not the most recent icon manipulated.</td>
</tr>
<tr>
<td>3 MOVCotherTypeMoves</td>
<td>MOVE COUNT: number of non-move manipulations</td>
</tr>
<tr>
<td>4 MOVCtotManipulations</td>
<td>MOVE COUNT: number of manipulations of any type</td>
</tr>
<tr>
<td>5 NUMIcons</td>
<td>NUMBER: icons used in depiction</td>
</tr>
<tr>
<td><strong>Summary of Move Distances per Experiment</strong></td>
<td></td>
</tr>
<tr>
<td>6 DISTtotalIcon</td>
<td>DISTANCE: the total distance icons have been dragged across canvas</td>
</tr>
<tr>
<td>7 DISTactualIcon</td>
<td>DISTANCE: the sums of the distances between first and last spots for each icon</td>
</tr>
<tr>
<td>8 DINSTDiffBtwnTotalandActual</td>
<td>DISTANCE: difference between the total (6) &amp; actual (7)</td>
</tr>
<tr>
<td><strong>Summary of Timing per Experiment</strong></td>
<td></td>
</tr>
<tr>
<td>9 TIMETotManipIcon</td>
<td>TIME: total time spent manipulating the icons</td>
</tr>
<tr>
<td>10 TIMETotConsideringIcon</td>
<td>TIME: total time spent &quot;considering&quot; between icon manipulations</td>
</tr>
<tr>
<td>11 TIMEexperimentDRAW</td>
<td>TIME: total time spent in the drawing portion of the experiment</td>
</tr>
<tr>
<td>12 TIMEexperimentQUES</td>
<td>TIME: total time spent in the questions portion of the experiment</td>
</tr>
<tr>
<td>13 TIMEexperimentBOTH</td>
<td>TIME: total time spent between both the drawing and question portions</td>
</tr>
<tr>
<td>14 TIMEcontemplation</td>
<td>TIME: experimental &quot;contemplation&quot;, time with canvas before first manipulation is made</td>
</tr>
<tr>
<td>15 TIMEreflection</td>
<td>TIME: experimental &quot;reflection&quot;, time with canvas after last manipulation, before submission</td>
</tr>
<tr>
<td>16 NUMtextScroll</td>
<td>NUMBER: count of instances of scrolling the passage</td>
</tr>
<tr>
<td><strong>Individual Icon Averages Per Experiment</strong></td>
<td></td>
</tr>
<tr>
<td>17 AVGMOVCMoveTypeMoves</td>
<td>AVERAGE COUNT: of move segments per icon comprising the drawing</td>
</tr>
<tr>
<td>18 AVGMOVCREmanipulations</td>
<td>AVERAGE COUNT: of re-manipulations per icon comprising the drawing</td>
</tr>
<tr>
<td>19 AVGMOCotherTypeMoves</td>
<td>AVERAGE COUNT: of non-move icon manipulation</td>
</tr>
<tr>
<td>20 AVGDISTtotalIcon</td>
<td>AVERAGE DISTANCE: of total movement of an icon</td>
</tr>
<tr>
<td>21 AVGDISTactualIcon</td>
<td>AVERAGE DISTANCE: of actual movement of an icon</td>
</tr>
<tr>
<td>22 AVGTIMEManipIcon</td>
<td>AVERAGE TIME: manipulating each icon</td>
</tr>
<tr>
<td>23 AVGTIMEConsideringIcon</td>
<td>AVERAGE TIME: &quot;considering&quot; each icon comprising the drawing</td>
</tr>
<tr>
<td>24 OTHERavgSelfPercep</td>
<td>OTHER: the student’s self-assessment of his/her performance across all experiments on a Likert scale</td>
</tr>
</tbody>
</table>

Table 2.1: Computed input interaction features that were collected for the use of the machine learning classifier.

All of the above features are an extension of the works described in Chapter 1. Many of the above are extensions of Baker’s (2007) work. Baker uses time-on-task extensively, and draws conclusions about
carelessness based on total time without examining the details of contemplation time between moves. Instead, we demonstrate in Chapter 5 that this added feature will strengthens the confidence in our conclusion. Additionally, Baker spoke of students submitting incorrect answers as being either careless or off-task. This thesis investigates this area more thoroughly by going beyond his summary correctness statistic and examining the timings and actions undertaken by the subject to correct (or not to correct) that answer.

McLaren et al.’s (2005) framework is also strongly reflected here. Task coherence and conceptual understanding can be measured by the re-manipulation numbers. Visual organization is determined by the subjective Likert-scale assessments of a human expert (see next section). Unlike in McLaren et al.’s work, where the feature describing coordination is somewhat subjectively measured, this thesis’ work moves to objectively measure this through both the distance, contemplation and re-manipulation features. The effectiveness and cognitive economy features of Carberry et al. (2006) and Kerpedijiev and Roth (2000) can be recreated with the collected statistics. Cognitive economy again will be based on the Likert-scale assessment.

For further technical details on data collection, see Appendix A. There, the structure of the raw data captured by the database, from which many of these features are derived, is detailed.

2.3 Subjects and Data

The study subjects are CUNY Language Immersion Program (CLIP) students at Kingsborough Community College. As a Community College member of the City University of New York system, Kingsborough serves students with limited English abilities (often recent immigrants). CLIP is a non-credit, intensive study of English run by the Division of Continuing Education. The investigator has previously worked with students in this group, and is familiar with their abilities.

Objective academic grade data was collected from each study subject. These consist of:
1. Score from the Michigan standardized reading competency test.

2. Numeric comprehension scores from multiple choice questions presented after each passage.

Subjective data was also collected:

1. Human assessments of produced sketches in several subjective outcome categories (as detailed in Section 3.5);

2. Subjects’ self-assessment of overall correctness of diagrams (Likert scale);

IRB approval for the study has been obtained at Kingsborough Community College [Protocol # 393073-1]. Appendix B contains the participants’ subjective self-assessment form.

Chapter 3 details the collection and preparation of the data for the analysis.
Chapter 3 – Implementation and Analysis Techniques

3.0 Introduction

This chapter describes the implementation details for the experiment outlined in the previous chapter. Management of the experiment comprises three parts:

- **Setup**: The experiment was setup over a period of one year (June 2010 through May 2011); this is described in Sections 3.1 through 3.3.

- **Data collection and preparation**: This included the experiment, collection of objective comparison features and subjective scoring. This spanned July 2011 through December 2011. The process is described in Sections 3.4 through 3.6.

- **Data analysis**: This spanned calendar years 2012 and 2013. It is described in Sections 3.7 and 3.8.

3.1 Designing and Implementing the Software

Transforming the Game Table suite into the experimental platform for this project was a cumbersome task. As was noted in the previous chapter, Game Table is a software suite originally designed for building role-playing game style maps by placing icons on a grid. The suite is open-source and programmed in Java. The development commenced in the Summer of 2010 for a period of one year. To assist in the development, the investigator employed a graduate of Brooklyn College’s Undergraduate CIS program as a programmer.

The first round of development focused on having the software save the input interaction features locally. The process began by collaboratively understanding the codebase of Game Table. This allowed the team to determine the subroutines to which the software “hooks” could be added to collect features on user actions. The interface was easily edited using IBM’s open-source Eclipse IDE (available from http://www.eclipse.org/). The team designed a series of Java objects that could hold the interaction features and eventually save these to a local file. Because the lab in which the software was to be used
made use of Windows PCs, much of the compiling, packaging and testing took place on a Windows computer. The Cygwin interface (available from http://www.cygwin.org/) was installed to give access to the bash shell and UNIX-style scripting. (See Appendix D for manifests and makefiles.)

Then the team focused on the interface. Game Table was then stripped of menus and additional interactive features that were irrelevant to the experiment. User prompts were added to guide each subject through the experiment. The interface was locked down to an eight-by-eight grid with only the transforms required.

Unfortunately, the task of revising the software to operate in a networked environment was rather difficult. Neither member of the team was familiar with serializing objects for transportation over a network. Many of the original objects designed to store features had to be dramatically redesigned to enable the use of simple Java data-structures that could be easily transmitted. This phase culminated with the development of an input interaction feature server that could centrally log all information gleaned from subjects’ interfaces.

3.2 Recruiting a Subject Pool and Developing Experimental Content

Nearly as important as the design of the software was the selection of an appropriate subject pool and developing the content for the experiments. This required working collaboratively with the CUNY Language Immersion Program (CLIP), for which the investigator had previously co-taught courses. CLIP’s Director, Frank Milano, allowed for collaboration with CLIP Instructor Liza Sunderlin, a fifteen year veteran of the program. Ms. Sunderlin was eager to add the experiment to her class’ activities and to assist in the content design.
The finding of passages was the first step. The goal was search for six descriptive scenes, each comprising approximately three to five paragraphs at the reading level of her students.

Passages were selected that were conducive to drawing a visual scene. Each passage was trimmed, removing some (but not all) non-descriptive statements and changing individual words that were unlikely to be in the accessible vocabulary for the subject pool. For each passage, the team then developed multiple choice reading comprehension questions and appropriate visual icons were selected. Half of the stories had three attendant comprehension questions, while the other half had four. Each multiple choice question offered exactly five answers from which to choose the best answer.

Project icons were taken from ClipArtEtc (Florida Center for Instructional Technology, 2013). The site allows a limited number of icons to be used without special permission for non-commercial, educational projects.
3.3 Software Testing and Deployment

As with most software development projects, the testing phase was crucial. The software first underwent several rounds of non-networked testing to ensure that it would correctly log user actions. We implemented a master text log feature to recount the experimental actions in detail; we then ran many instances of diagram manipulation and compared the actions to the log, manually verifying numbers, such as "icon distance moved" as a check on the integrity of the data saved.

In the second round, the software was tested on the network at Kingsborough Community College. The software was tested at different times of the day, which proved to be a worthy strategy. This second round exposed flaws in the design, most notably delays in the user interface due to network latency. These delays were caused due to daytime student traffic across the network and would not have been detected without the software being tested in the operating environment. MySQL Workbench (available from http://www.mysql.com/products/workbench/) is a free, open-source database management tool, that proved to be a crucial for this phase of testing. The Workbench allowed for rapid configuration of test accounts and real-time monitoring of the database.

With the software completed and tested, we required heavy cooperation from Kingsborough Community College’s Department of Information Technology Services to appropriately deploy it. To protect the experimental data from being intercepted by unauthorized third-parties on the Internet, Kingsborough established an on-campus server that could securely receive data from a campus lab. The server was configured with a MySQL database and loaded with our proprietary Java program for receiving and storing data from the individual experiment clients. The College then identified a lab that could be used for the experiments and published the software to a local drive available only in this lab.
3.4 Administration of the Experiment

The experiment was successfully run with 62 student-participants. It was run with three CLIP cohorts with two instructors. Each session began with the investigator demonstrating how to use the software, explaining the experiment and providing the appropriate IRB consent forms. The features, including how to add icons to the canvas and how to perform the various transforms were clearly shown to all users. Prior to starting the experiment, participants were aware of the sequence of their upcoming tasks – (1) reading, (2) drawing and (3) answering multiple-choice questions. Each student was provided with a numeric login and passcode. The login was used in lieu of a name to protect subjects’ identities (see Figure 3.1).

![Figure 3.1: Screenshots of the login and passcode functions of the software.](image)

The first run of the experiment was with approximately twenty of Professor Sunderlin’s students. This smaller run was done in case of any need to revise the experiment protocol. The run proved to be successful. The investigator then worked with the Director of the CUNY Language Immersion Program to secure permission to work with a second of Professor Sunderlin’s classes and a class of a second instructor in the program. The second and third experimental trials were run on the same day several months later.

Although the potential for problems in administering the experiment was great, no major adverse events occurred. On the day of the first session, several lab workstations were found to be out-of-order ranging from major system problems to minor issues such as broken mice or missing mousepads. (The latter, while seemingly minor, would have a significant detrimental effect on an experiment that monitors mouse-actions so closely.) To accommodate the loss of workstations, the investigator secured a second lab and manually installed the software on workstations. However, due to student absences, this became
unnecessary. It was also anticipated that there would be late students that would miss the initial demonstration or that students might not like the exercise. Fortunately, neither of these situations occurred during the experiment. In general, students appeared quite eager to complete the exercise.

3.5 Subjective and Objective Experimental Data

Objective comparison data came from in-experiment multiple choice questions and scores on the standardized Michigan Test for English proficiency. (CLIP students already undergo Michigan testing as part of the program.) Multiple choice questions were answered inside the application and scored immediately when stored in the database. The two instructors provided data on standardized test scores for the students that consented to participating, this score data was immediately coded with the anonymous four-digit IDs, thus making the data immediately de-identified.

Subjective grading was a larger undertaking, requiring two passes in examining the corpus of produced drawings. In the first pass, the three original categories – cognitive economy, on-task behavior and precision (see Chapter 2 and Table 3.1, on the following page) – were graded on a five-point Likert scale. However, analysis showed that this was too large of a grading spread given the number of data-points in our experiments to yield any useful classifications. Also, the first round of evaluation allowed the investigator to recognize that some drawings had more logic in composition than others, some looked more like a cogent real-life scene, and some did a better job of placing key elements from the story.

A second pass was necessary to narrow the grading to a three-point Likert scale (the tertiles originally proposed), and to add the subjective categories realized in the first pass. Further, to ensure consistency in grades across drawings, the following safeguards were put into place:

- A detailed grading rubric was developed (see Table 3.1) to ensure the utmost consistency in grades across drawings.
- Drawings were graded one category at a time; each category was graded in a single sitting.
- Drawings were graded story-by-story.
**Table 3.1: Rubric for subjective outcome assessments on experimental drawings.**

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>On-Task</strong></td>
<td>the level of effort put into creating the diagram.</td>
</tr>
<tr>
<td>3</td>
<td>demonstrates special, thoughtful, attention paid to details.</td>
</tr>
<tr>
<td>2</td>
<td>demonstrates a basic read of the story with effort made to choose and/or arrange icons.</td>
</tr>
<tr>
<td>1</td>
<td>took minimal effort, student was disinterested in spending effort completing the exercise.</td>
</tr>
<tr>
<td><strong>Cognitive Economy</strong></td>
<td>how easily the drawing is understood.</td>
</tr>
<tr>
<td>3</td>
<td>at first glance, drawing makes sense without question.</td>
</tr>
<tr>
<td>2</td>
<td>causes a few &quot;why?&quot; questions to the viewer, but mostly makes sense.</td>
</tr>
<tr>
<td>1</td>
<td>makes little sense even after having moments to reflect upon it.</td>
</tr>
<tr>
<td><strong>Logical</strong></td>
<td>the level of relative sizing and visual organization of the icons</td>
</tr>
<tr>
<td>3</td>
<td>chosen icons are perfectly (or nearly so) organized to depict the scene at hand.</td>
</tr>
<tr>
<td>2</td>
<td>chosen icons demonstrate some/decent organization with some questions about why items were placed where they were.</td>
</tr>
<tr>
<td>1</td>
<td>chosen icons demonstrate poor or lack any scene-like organization.</td>
</tr>
<tr>
<td><strong>Key Elements</strong></td>
<td>does the sketch have the most prominent few elements from the passage.</td>
</tr>
<tr>
<td>(These elements were identified for each passage.)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>yes, all.</td>
</tr>
<tr>
<td>2</td>
<td>yes, most.</td>
</tr>
<tr>
<td>1</td>
<td>no.</td>
</tr>
<tr>
<td><strong>Like Scene</strong></td>
<td>does the sketch displayed reasonably appear to be a real-life room.</td>
</tr>
<tr>
<td>3</td>
<td>diagram looks like a room.</td>
</tr>
<tr>
<td>2</td>
<td>diagram looks a room but there is at least one non-trivial error.</td>
</tr>
<tr>
<td>1</td>
<td>does not look like a room.</td>
</tr>
<tr>
<td><strong>Precision</strong></td>
<td>how accurately the drawing portrays the passage.</td>
</tr>
<tr>
<td>3</td>
<td>diagram accurately reflects the story and the chosen icons are correct.</td>
</tr>
<tr>
<td>2</td>
<td>spirit of the story comes across in the diagram, but there are technical errors.</td>
</tr>
<tr>
<td>1</td>
<td>diagram misses large elements of the story or depicts an unrelated scene.</td>
</tr>
</tbody>
</table>
3.6 Secure Storage and Data Preparation

Before data analysis could occur in Weka, the data had to be securely moved from the MySQL database, pre-processed, and stored. Two programs were useful for the initial step: MySQL Workbench again allowed for the extraction of the data from the server. TrueCrypt allowed for secure encryption of study data (TrueCrypt, 2014). Among its features is the ability for a user to create a portable file that acts as an encrypted volume. Data extracted from the SQL server, and other experimental files were stored with AES encryption. Data was stored in raw-text, CSV (comma separated value) files. At this stage the data were still not directly usable for any numeric analysis. It was a log of experiment times, icons dragged, interface interaction, transforms performed and multiple choice answers.

The next step was to digitize the paper surveys from the experiments and the objective scores collected from instructors. This was accomplished in spreadsheets developed in Microsoft Excel. Data was coded by the anonymous participant ID number and then immediately de-identified. To associate the paper-survey’s self-assessment scores with actual drawing experiments required developing a Java program (See AssociateSelfAssessment.java in Appendix E.). Data underwent a minor cleanup operation using find and replace functions to make it compliant with other databases. (For example, MySQL used power-of-ten notation, base-E-exponent, to represent numbers. These numbers were converted to raw decimals.) Data was loaded into a Microsoft Access database in order to manage the multiple tables and to run structured queries. An extraction program was written to recover drawings from the database (stored as textual strings) and convert these to viewable JPEG images. (See ExtractPictures.java in Appendix E.)

Next a program was developed to pre-process the raw interaction data, distilling it into the compound features specified in Chapter 2. The program was implemented in Java and used Windows ODBC drivers to retrieve data from the Microsoft Access database. (See ComputeMoveStats.java in Appendix E). Specifically, this computed:

1. TOTAL time between moves.
2. TOTAL distance moved.
3. TOTAL time making moves.
4. AVERAGE time between moves.
5. AVERAGE distance moved.
6. AVERAGE time making moves.
7. TOTAL number of changing move types, defined as making a type of move, then choosing a new type of move.
8. TOTAL number of each types of move.
9. AVERAGE number of move segments per icon.
10. TOTAL number of remanipulations – defined as manipulating an icon that has been previously moved.

The last cleanup task involved removing demonstration data from the database. Having such data in the database was an unavoidable consequence of the experiment for two reasons: (1) It was necessary to demonstrate the software live in front of the participants; (2) By using a live version of the software directly prior to the experiment, the operability of the software, the network and the server database could be verified. This ensured that no participant would be interrupted and that no experimental data would be lost. Several delete queries in Microsoft Access accomplished this final task. First the test participant IDs were deleted, then all moves and features without a matching participant were removed.

3.7 Weka Pass 1 – Algorithm Selection

Weka, developed at University of Waikato, is an open-source suite that packages a wide array of standard artificial intelligence algorithms with a simple data and analysis interface. Weka was chosen for the analysis because of its robust set of machine learning algorithms and its simple graphical user interfaces (Weka Experimenter and Weka Explorer) (Waikato, 2010).
The extensive preprocessing effort made the data ready to be exported to attribute-relation file format (ARFF files). These are the standard ASCII text files used to describe attributes that are loaded into the Weka suite for classification (Waikato, 2008). A separate ARFF file had to be generated for every subjective outcome assessment described in Table 3.1. Appendix F shows the format of a sample file.

The investigator performed a baseline pass at the data to ascertain what strategy and algorithms were most suited for the analysis. Several regression and classification algorithms were run. Classification algorithms were chosen in the categories of Bayesian, Linear Classification, Decision Tree, and Lazy Learning. These algorithms were run against the full feature set with ten-fold cross-validation. Weka partitions the data into ten sets, nine of which are used for training and the remaining one is used for validation. The algorithm is repeated ten times, with each subset used once as the validator. Resulting scores of correctness are based on the average of the ten runs.

The first pass was performed with a brute force approach. It demonstrated that a variant of the NaïveBayes algorithm (NaïveBayesUpdatable with kernel estimator activated) produced the most correct results for these data. This is a simple classification algorithm that creates probability rules through the multiplication of features. Redundant data can cause issues with the classification (Witten, Frank & Hall, 2011). For example, in our experiment, “experiment time” encapsulates “contemplation time” and “reflection time”. Thus, the removal of individual input interaction features by a feature selection process generally improved the overall classification. Activating the kernel estimator causes Weka to first determine the underlying distribution of the set.

3.8 Weka Pass 2 – Feature Segmentation

The first pass yielded a standard function (NaïveBayesUpdatable, with the kernel estimator activated) to be used in our analysis. Next the focus was on which of the features were most helpful in the classification. Recall that Table 2.1 lists the complete set of collected features.
The investigator devised four cases of potential feature segmentation described below.

**Case 1:** one set, all features

MOVC\text{moveType}, MOVC\text{remanipulations}, MOVC\text{otherTypeMoves}, MOVC\text{totManipulations},
NUM\text{Icons}, DIST\text{totalIcon}, DIST\text{actualIcon}, DIST\text{DiffBtwnTotalandActual}, TIMETotManip\text{Icon},
TIMETotConsidering\text{Icon}, TIME\text{experimentDRAW}, TIME\text{experimentQUES},
TIME\text{experimentBOTHTIMEcontemplation}, TIME\text{reflection}, NUM\text{textScroll},
AVGMOVC\text{MoveTypeMoves}, AVGMOVC\text{Remanipulations}, AVGMOVC\text{OtherTypeMoves},
AVGDIST\text{TotalIcon}, AVGDIST\text{ActualIcon}, AVGTIME\text{Manip\text{Icon}}, AVGTIME\text{Considering\text{Icon}},
OTHER\text{avgSelfPercep}

**Case 2:** three mutually exclusive sets, logical manual groupings by type:

**set 2.1:** move counts and distances (8)

MOVC\text{moveType}, MOVC\text{remanipulations}, MOVC\text{otherTypeMoves}, MOVC\text{totManipulations},
NUM\text{Icons}, DIST\text{totalIcon}, DIST\text{actualIcon}, DIST\text{DiffBtwnTotalandActual}

**set 2.2:** timing and interface (8)

TIMETotManip\text{Icon}, TIMETotConsidering\text{Icon}, TIME\text{experimentDRAW}, TIME\text{experimentQUES},
TIME\text{experimentBOTHTIMEcontemplation}, TIME\text{reflection}, NUM\text{textScroll}

**set 2.3:** averages at the icon level (7)

AVGMOVC\text{MoveTypeMoves}, AVGMOVC\text{Remanipulations}, AVGMOVC\text{OtherTypeMoves},
AVGDIST\text{TotalIcon}, AVGDIST\text{ActualIcon}, AVGTIME\text{Manip\text{Icon}}, AVGTIME\text{Considering\text{Icon}}

**Case 3:** three mutually exclusive sets, logical manual groupings by type:

**set 3.1:** experiment-level and icon-level move counts (8)

MOVC\text{moveType}, MOVC\text{remanipulations}, MOVC\text{otherTypeMoves}, MOVC\text{totManipulations},
AVGMOVC\text{MoveTypeMoves}, AVGMOVC\text{Remanipulations}, AVGMOVC\text{OtherTypeMoves},
NUM\text{textScroll}.

**set 3.2:** distances (5)
DISTtotalIcon, DISTactualIcon, DISTDiffBtwnTotalandActual, AVGDISTTotalIcon, AVGDISTActualIcon

set 3.3: timing data (9)
TIMETotManipIcon, TIMETotConsideringIcon, TIMEexperimentDRAW, TIMEexperimentQUES, TIMEexperimentBOTH, TIMEcontemplation, TIMEREflection, AVGTIMEManipIcon, AVGTIMEConsideringIcon

**Case 4:** three overlapping sets, determined by Weka’s automated feature selection tool for the composite scores:

set 4.1: top six features by a best-first algorithm
MOVCmoveType, MOVCotherTypeMoves, MOVCtotManipulations, DISTtotalIcon, DISTactualIcon, TIMETotManipIcon

set 4.2: top six features by a wrapper for NaïveBayesUpdatable

set 4.3: top ten features by a wrapper for NaïveBayesUpdatable

**Case 5:** a feature set that is different for each subjective outcome assessment, determined by Weka’s automated feature selection tool.

Case 4 used Weka’s attribute selection function on a composite of all of the subjective outcome assessments. This created a single set of features that was subsequently used to classify each individual subjective outcome assessment. In Case 4.1, the CfsSubsetEval algorithm with the BestFirst Search method was used to score the unsegmented set of features by their influence on the set. Cases 4.2 and 4.3 used a wrapper class that iteratively executed NaïveBayesUpdatable with different feature subsets. Each feature selection was run in Weka Explorer with ten-fold cross-validation.

Case 5 repeated the wrapper class selection for each subjective outcome assessment. This created different feature sets for each assessment. The sets were created using a wrapper class that iteratively executed NaïveBayesUpdatable with different feature subsets. Features that had a usefulness score
higher than 40% were included in the set. (That is, if the feature had an impact in four or more of the ten folds, it was deemed useful.)

Table 3.2 summarizes the distribution of the features across Cases 1-4. Table 3.3 summarizes the feature distribution for Case 5.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Segmented Case</th>
<th>All</th>
<th>Logical 1</th>
<th>Logical 2</th>
<th>Automatic FS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>2.1</td>
<td>2.2</td>
<td>2.3</td>
</tr>
<tr>
<td>1 MOV CmoveType</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 MOV Cremanipulations</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 MOV CotherTypeMoves</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>4 MOV CtotManipulations</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5 NUMIcons</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>6 DISTtotalIcon</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>7 DISTactualIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>8 DISTDiffBtwntotalandActual</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>9 TEMTTotManipIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>10 TEMPLotConsideringIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>11 TEMTexperimentDRAW</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>12 TEMTexperimentQUES</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>13 TEMTexperimentBOTH</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>14 TEMTcontemplation</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>15 TEMTreflection</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>16 NUMtextScroll</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>17 AVGMOV CmoveTypeMoves</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>18 AVGMOVCremanipulations</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>19 AVGMOV CotherTypeMoves</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>20 AVGDISTtotalIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>21 AVGDISTactualIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>22 AVGTIMEManipIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>23 AVGTIMEConsideringIcon</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>24 OTHERavgSelfPercep</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3.2: Feature distribution across Cases 1 through 4.
Next, each subjective outcome assessment was reduced from tertiles to a binary set. That is, the graded tertiles were noted as strictly above a meaningful threshold or not. Given the relatively small number of participants (n=62) and number of usable drawings (n=178) reduction was necessary for the AI algorithms to have sufficient data on which to base classifications. Table 3.4 describes these thresholds and the rationale behind the selection. (Recall that Table 3.1 describes the subjective outcome assessments in detail.)
<table>
<thead>
<tr>
<th>Subjective Outcome Assessment</th>
<th>Threshold (&gt;)</th>
<th>Justification</th>
</tr>
</thead>
<tbody>
<tr>
<td>On-Task</td>
<td>1</td>
<td>Separates reasonable effort from drawings with little to no effort.</td>
</tr>
<tr>
<td>Cognitive Economy</td>
<td>2</td>
<td>Separates drawings that are easily understood from drawings that need some revision.</td>
</tr>
<tr>
<td>Logical</td>
<td>2</td>
<td>Separates the nearly perfect drawings with icon placements that seem purposeful to one another. The viewer has no questions about icon placement.</td>
</tr>
<tr>
<td>Key Element</td>
<td>1</td>
<td>Separates drawings with most or all salient elements from the passage.</td>
</tr>
<tr>
<td>Like Scene</td>
<td>2</td>
<td>Separates scenes that look like a room from those that have glaring issues.</td>
</tr>
<tr>
<td>Precision</td>
<td>1</td>
<td>Separates diagrams that convey the meaning of the story from those that are wholly unrelated.</td>
</tr>
</tbody>
</table>

Table 3.4: Binary thresholds for subjective outcome assessments.

The results of the classification are described in Tables 3.5 and 3.6. The first of these tables (3.3) lists the rate of correctly classified instances when the feature sets from Tables 3.2 and 3.3 are tested at the thresholds noted in Table 3.4. The bolded numbers are where the classifier correctness is maximized. The parenthetical numbers to the right of the names indicate the number of included features. Each classification was run in Weka Explorer with ten-fold cross-validation. The input files were sorted ascending by the participants’ unique ID, and Weka was set to preserve order in building the folds. The effect of this is that folds were generally comprised of approximately six participants’ data. Thus the cross-validation was rarely using one participant’s own data across several drawings as model-builder and validator, keeping reported classification successes legitimately based on the underlying feature set.

Table 3.6 repeats this for the objective grade data. Each table also includes the lower baseline, which provides the percentage of data points that are members of the larger half of the binary split. Since these are binary sets, a classification correctness rate above 50% indicates better-than-random odds. Despite this, the lower baseline is provided as a second benchmark – an AI algorithm that exceeds this mark would outperform a simple skew algorithm that classifies all elements as members of the larger of the binary partitions.
Table 3.5: Classification accuracy of feature sets for subjective outcome assessments.

<table>
<thead>
<tr>
<th>Subjective Outcome Assessment Category</th>
<th>Case</th>
<th>Feature Set Description</th>
<th>Tertile-to-Binary Threshold</th>
<th>On Task</th>
<th>Cog. Econ.</th>
<th>Logical</th>
<th>Key Elem.</th>
<th>Like Scene</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>All (24)</td>
<td>65.22%</td>
<td>54.66%</td>
<td>60.87%</td>
<td>65.22%</td>
<td>58.39%</td>
<td>66.46%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.1</td>
<td>Experiment Move Counts and Distances (8)</td>
<td>68.99%</td>
<td>59.01%</td>
<td>63.35%</td>
<td>65.22%</td>
<td>60.87%</td>
<td>69.57%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>Experiment Timing and Interface (8)</td>
<td>60.87%</td>
<td>53.42%</td>
<td>57.14%</td>
<td>71.43%</td>
<td>54.66%</td>
<td>62.11%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2.3</td>
<td>Icon-Level Averages (7)</td>
<td>67.70%</td>
<td>49.69%</td>
<td>59.01%</td>
<td>69.57%</td>
<td>60.25%</td>
<td>62.73%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.1</td>
<td>Experiment- &amp; Icon-Level Moves (8)</td>
<td>67.70%</td>
<td>54.66%</td>
<td>59.01%</td>
<td>67.70%</td>
<td>59.01%</td>
<td>68.32%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>Experiment- &amp; Icon-Level Distances (5)</td>
<td>66.46%</td>
<td>54.66%</td>
<td>63.35%</td>
<td>70.19%</td>
<td>60.87%</td>
<td>65.84%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>3.3</td>
<td>Experiment- and Icon-Level Timing (9)</td>
<td>62.73%</td>
<td>53.42%</td>
<td>59.63%</td>
<td>68.94%</td>
<td>59.01%</td>
<td>62.73%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.1</td>
<td>Aggregate Top-6 by Best-First Algorithm (6)</td>
<td>67.70%</td>
<td>59.01%</td>
<td>63.98%</td>
<td>64.60%</td>
<td>62.11%</td>
<td>71.43%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.2</td>
<td>Aggregate Top-6 by Naïve Bayes Wrapper (6)</td>
<td>70.19%</td>
<td>63.35%</td>
<td>60.87%</td>
<td>73.91%</td>
<td>57.76%</td>
<td>68.32%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>4.3</td>
<td>Aggregate Top-10 by Naïve Bayes Wrapper (10)</td>
<td>65.22%</td>
<td>60.25%</td>
<td>65.22%</td>
<td>75.78%</td>
<td>60.25%</td>
<td>68.94%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Rated &gt; 40% Utility by Naïve Bayes Wrapper (V)</td>
<td>74.53%</td>
<td>61.49%</td>
<td>68.94%</td>
<td>73.91%</td>
<td>60.87%</td>
<td>69.57%</td>
<td></td>
</tr>
</tbody>
</table>

Table 3.6: Classification accuracy of feature sets for objective grade data.

<table>
<thead>
<tr>
<th>Objective Grade Assessment Category</th>
<th>Case</th>
<th>Category Description</th>
<th>Multiple Ch. %</th>
<th>Michigan</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>All (24)</td>
<td>52.17%</td>
<td>60.14%</td>
</tr>
<tr>
<td></td>
<td>2.1</td>
<td>Experiment Move Counts and Distances (8)</td>
<td>50.93%</td>
<td>57.97%</td>
</tr>
<tr>
<td></td>
<td>2.2</td>
<td>Experiment Timing and Interface (8)</td>
<td>55.90%</td>
<td>63.04%</td>
</tr>
<tr>
<td></td>
<td>2.3</td>
<td>Icon-Level Averages (7)</td>
<td>57.14%</td>
<td>61.59%</td>
</tr>
<tr>
<td></td>
<td>3.1</td>
<td>Experiment- &amp; Icon-Level Moves (8)</td>
<td>50.93%</td>
<td>60.14%</td>
</tr>
<tr>
<td></td>
<td>3.2</td>
<td>Experiment- &amp; Icon-Level Distances (5)</td>
<td>52.17%</td>
<td>60.14%</td>
</tr>
<tr>
<td></td>
<td>3.3</td>
<td>Experiment- and Icon-Level Timing (9)</td>
<td>54.04%</td>
<td>63.04%</td>
</tr>
<tr>
<td></td>
<td>4.1</td>
<td>Aggregate Top-6 by Best-First Algorithm (6)</td>
<td>51.55%</td>
<td>60.14%</td>
</tr>
<tr>
<td></td>
<td>4.2</td>
<td>Aggregate Top-6 by Naïve Bayes Wrapper (6)</td>
<td>50.31%</td>
<td>61.59%</td>
</tr>
<tr>
<td></td>
<td>4.3</td>
<td>Aggregate Top-10 by Naïve Bayes Wrapper (10)</td>
<td>47.21%</td>
<td>60.87%</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>Rated &gt; 40% Utility by Naïve Bayes Wrapper (V)</td>
<td>59.63%</td>
<td>68.12%</td>
</tr>
</tbody>
</table>

The most significant of these results, and their implications, are discussed in Chapter 4.
Chapter 4 – Lessons from the Experimental Results

4.0 Organization of the Analysis

This chapter reviews the results of the classification process and draws conclusions about the usefulness of different types of input interaction features (Chapter 2) toward making predictions of about the subjective outcome assessments (Section 3.5).

In the previous chapter, Tables 3.5 and 3.6 describe the entirety of the results. In this chapter, we focus on the results that have the most useful correct classification rates. These categories – specifically: On Task, Logical, Key Element, Precision and Michigan Test – will provide insights for future research and are noted in Table 4.1.

<table>
<thead>
<tr>
<th>Subjective Outcome Assessment Category</th>
<th>On Task</th>
<th>Logical</th>
<th>Key Element</th>
<th>Precision</th>
<th>Michigan</th>
</tr>
</thead>
<tbody>
<tr>
<td>Threshold</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>60%</td>
</tr>
<tr>
<td>Case Category Description</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 All (24)</td>
<td>65.22%</td>
<td>60.87%</td>
<td>65.22%</td>
<td>66.46%</td>
<td>60.14%</td>
</tr>
<tr>
<td>2.1 Experiment Move Counts and Distances (8)</td>
<td>68.99%</td>
<td>63.35%</td>
<td>65.22%</td>
<td>69.57%</td>
<td>57.97%</td>
</tr>
<tr>
<td>2.2 Experiment Timing and Interface (8)</td>
<td>60.87%</td>
<td>57.14%</td>
<td>71.43%</td>
<td>62.11%</td>
<td>63.04%</td>
</tr>
<tr>
<td>2.3 Icon-Level Averages (7)</td>
<td>67.70%</td>
<td>59.01%</td>
<td>69.57%</td>
<td>62.73%</td>
<td>61.59%</td>
</tr>
<tr>
<td>3.1 Experiment- &amp; Icon-Level Moves (8)</td>
<td>67.70%</td>
<td>59.01%</td>
<td>67.70%</td>
<td>68.32%</td>
<td>60.14%</td>
</tr>
<tr>
<td>3.2 Experiment- &amp; Icon-Level Distances (5)</td>
<td>66.46%</td>
<td>63.35%</td>
<td>70.19%</td>
<td>65.84%</td>
<td>60.14%</td>
</tr>
<tr>
<td>3.3 Experiment- and Icon-Level Timing (9)</td>
<td>62.73%</td>
<td>59.01%</td>
<td>68.94%</td>
<td>62.73%</td>
<td>63.04%</td>
</tr>
<tr>
<td>4.1 Aggregate Top-6 by Best-First Algorithm (6)</td>
<td>67.70%</td>
<td>63.35%</td>
<td>64.60%</td>
<td>71.43%</td>
<td>60.14%</td>
</tr>
<tr>
<td>4.2 Aggregate Top-6 by Naïve Bayes Wrapper (6)</td>
<td>70.19%</td>
<td>60.87%</td>
<td>73.91%</td>
<td>68.32%</td>
<td>61.59%</td>
</tr>
<tr>
<td>4.3 Aggregate Top-10 by Naïve Bayes Wrapper (10)</td>
<td>65.22%</td>
<td>65.22%</td>
<td>75.78%</td>
<td>68.94%</td>
<td>60.14%</td>
</tr>
<tr>
<td>5 Rated &gt; 40% Utility by Naïve Bayes Wrapper (V)</td>
<td>74.53%</td>
<td>68.94%</td>
<td>73.91%</td>
<td>69.57%</td>
<td>68.12%</td>
</tr>
<tr>
<td># of features for Case 5</td>
<td>2</td>
<td>5</td>
<td>2</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>Lower Baseline</td>
<td>68.90%</td>
<td>61.49%</td>
<td>73.91%</td>
<td>51.55%</td>
<td>64.49%</td>
</tr>
<tr>
<td>Maximum Classification Correctness</td>
<td>74.53%</td>
<td>68.94%</td>
<td>75.78%</td>
<td>71.43%</td>
<td>68.12%</td>
</tr>
</tbody>
</table>

Table 4.1: The most salient classifications from this experiment.

Sections 4.1 through 4.5 explain the significance of the features to each of these categories. We will discuss how useful different types of features are for classifying data into each of the evaluated categories. Recall that Table 2.1 divided features into two primary types:
Experiment-level: These features comprise the sum total of all actions in one drawing experiment. For example, “actual icon” measures the distance an icon was moved between its first and last position (ignoring intermediate placements). Thus, $DIST_{ActualIcon}$ is the sum of the movements of every icon on the canvas.

Icon-level: These features comprise averages for the individual icons that comprise an experiment. For example, $AVGDIST_{ActualIcon}$ would take the average of the distance each icon moved from first to final position.

Feature subsets are primarily scored by their Correct Classification Rate (CCR), that is the percentage of data points that were correctly labeled given the subset of features that was the input to the NaiveBayes classifier. The conclusion of each section presents a summary table rating individual and categories of features by their level of prediction – how well these features improved CCR. These ratings are:

- predictive (the majority of elements contributed to a best-case classification);
- trivial (classification performed with these features performs neither near the top nor near the bottom);
- detractor (a feature set comprising these elements led to a worst-case classification).

Finally, a connector denotes the categories of features that comprised the best feature sets.

Then, Chapter 5 draws broader conclusions from this experiment.

### 4.1 Analysis of Subjective Outcome Assessment: On Task

The On Task assessment measures the level of effort put into creating the diagram. The classifier was tasked with separating drawings with reasonable effort (those that were thoughtful and those that demonstrated at least a basic effort to represent the story) from ones with minimal effort. Table 4.2 shows the best and worst cases for the classification algorithm.
Table 4.2: Best and worst feature sets for On Task data.

<table>
<thead>
<tr>
<th>Feature Count</th>
<th>Correct Classification Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>2</td>
</tr>
<tr>
<td>2 MOVCremanipulations</td>
<td>X</td>
</tr>
<tr>
<td>3 MOVCotherTypeMoves</td>
<td></td>
</tr>
<tr>
<td>5 NUMIcons</td>
<td>X</td>
</tr>
<tr>
<td>8 DISTDiffBtwnTotalandActual</td>
<td></td>
</tr>
<tr>
<td>9 TIMETotManipIcon</td>
<td>X</td>
</tr>
<tr>
<td>10 TIMETotConsideringIcon</td>
<td>X</td>
</tr>
<tr>
<td>11 TIMEexperimentDRAW</td>
<td>X</td>
</tr>
<tr>
<td>12 TIMEexperimentQUES</td>
<td>X</td>
</tr>
<tr>
<td>13 TIMEexperimentBOTH</td>
<td>X</td>
</tr>
<tr>
<td>14 TIMEcontemplation</td>
<td>X</td>
</tr>
<tr>
<td>15 TIMEreflection</td>
<td>X</td>
</tr>
<tr>
<td>16 NUMtextScroll</td>
<td></td>
</tr>
<tr>
<td>18 AVGMOVCREmanipulations</td>
<td>X</td>
</tr>
<tr>
<td>22 AVGTIMEManipIcon</td>
<td>X</td>
</tr>
<tr>
<td>23 AVGTIMEConsideringIcon</td>
<td>X</td>
</tr>
<tr>
<td>24 OTHERavgSelfPercep</td>
<td>X</td>
</tr>
</tbody>
</table>

Case 2.2 and 3.3, which both have the worst performance, rely entirely on timing data. But while Case 2.2 relies entirely on the experimental-level timing data, when some icon-level timing data is added in Case 3.3, there is a 3% increase in performance (60.87% to 62.73%). These two worst cases have a larger set of features than the best cases.

The best cases do not make much use of the timing data; these data seem harmful to the classification. As evidence of its lack of use, TIMEreflection scored a 30% in the Case 5 feature selection. When this feature is added to the set, the correctness rate decreases from 74.53% to 70.19%. Only the TIMEcontemplation is useful from the experiment-level timing features. Sets that made use of distance features scored mid-range, though these features did not enter into the calculations of the best sets. It is particularly interesting to note that in the second best solution, DISTDiffBtwnTotalandActual (experiment-level) was crucial, and in the best, the AVGMOVCREmanipulations (icon-level) was crucial. It is
understandable that one could potentially be swapped for the other as these two numbers do have a meaningful relationship: As the number of remanipulations increases per icon, more non-final movement per icon is occurring. Thus there is a high likelihood of the difference between actual and total distance moved increasing as well. (Recall that total distance is the sum of all of the interim movements, whereas actual distance only computes the distance between the first and last moves.) NUMIcons is present in both of the best feature sets and absent in the worst; it scores a full 100% from Weka’s feature selection algorithm (as noted in Table 4.3).

<table>
<thead>
<tr>
<th>Number of folds (%)</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 (100 %)</td>
<td>5 NUMIcons</td>
</tr>
<tr>
<td>3 ( 30 %)</td>
<td>15 TIMEreflection*</td>
</tr>
<tr>
<td>7 ( 70 %)</td>
<td>18 AVGMOVCREmanipulations</td>
</tr>
</tbody>
</table>

* TIMEreflection is not included, as it falls below the 40% usefulness threshold.

Table 4.3: Salient feature selection in Case 5 for On Task data.

<table>
<thead>
<tr>
<th>Feature Category</th>
<th>Level</th>
<th>Predictive Features*</th>
<th>Pred.</th>
<th>Triv.</th>
<th>Detr.</th>
<th>Conn.***</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement Counts</td>
<td>Experiment</td>
<td>NUMIcons, Remanipulations, OtherTypeMoves,</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>Remanipulations</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distances</td>
<td>Experiment</td>
<td>DiffBtwTotalandActual</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timing</td>
<td>Experiment</td>
<td>Contemplation</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Self-Report</td>
<td>Experiment</td>
<td>SelfPercep</td>
<td></td>
<td></td>
<td>X**</td>
<td></td>
</tr>
</tbody>
</table>

* Bolded text indicates features or categories of particular strength (or weakness). The solid connector indicates the categories comprising the best feature set.

** Icon-level timing is an element of the second-worst case, but its addition improves the CCR from the worst-case, thus it is considered predictive.

*** The connection column notes which feature categories were most predictive.

Table 4.4: Categories of features and their predictive qualities for On Task data.

4.2 Analysis of Subjective Outcome Assessment: Logical

The Logical assessment measures the correctness in the relative sizing and visual organization of the icons. The classifier was tasked with segmenting the nearly perfect drawings with icon placements that seem purposeful to one another from sketches that have questionable to no reasonable organization. For
the top drawings, the viewer has no obvious questions about icon placement. Table 4.5 details the best and worst classification cases.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Segmented Case</th>
<th>Best</th>
<th>Worst</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4.3</td>
<td>5</td>
</tr>
<tr>
<td>1 MOVComoveType</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 MOVCremanipulations</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3 MOVCootherTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 MOVClotManipulations</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>5 NUMIcons</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6 DISTactualIcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 DISTDiffBtwnTotalandActual</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 TIMETotManipIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9 TIMETotConsideringIcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10 TIMEexperimentDRAW</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11 TIMEexperimentQUES</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 TIMEexperimentBOTH</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>13 TIMEcontemplation</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>14 TIMEreflection</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>15 NUMtextScroll</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16 AVGMOVCMoveTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17 AVGMOVCRemanipulations</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18 AVGMOVCOtherTypeMoves</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>19 AVGDISTTotalIcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20 AVGDISTActualIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21 AVGDISTActualIcon</td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>22 AVGTIMEManiplcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23 AVGTIMEConsideringIcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24 OTHERavgSelfPercep</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Feature Count</strong></td>
<td>10</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td><strong>Correct Classification Rate</strong></td>
<td>68.94%</td>
<td>65.22%</td>
<td>57.14%</td>
</tr>
</tbody>
</table>

Table 4.5: Best and worst feature sets for Logical data.

For the Logical data, the three worst sets demonstrate that the indiscriminate use of all members of a category of features leads to poor classification performance. Movement features play the strongest role in the classification. Much like in the case of On Task, the use of movement features, coupled with select features from other categories, leads to the best case. Using all movement features alone, degrades performance. The best case makes heavy use of experiment-level move counts, yet when the bad Case
3.1 adds icon-level move counts to this, it dramatically drops accuracy. The experiment-level counts of MOV CaitManipulations and MOVCo therTypeMoves were particularly helpful to the best feature set.

Timing data plays a role in the classification. The two best sets use some combination of the TIMEexperimentques or TIMEexperimentBOTH features, with the TIMEcontentpation feature. But experiment-level timing data alone does not provide enough information, as illustrated by the worst case, Case 2.2.

Most distance features are of little use. The AVGDISTActualIcon is used in both of the best feature sets, though the best-case and second-best case pair it with DISTDiffBtwnTotalandActual and DISTactualIcon, respectively. The first choice provides more data about interim icon movements, while the latter is somewhat redundant.

Table 4.6 reinforces the power of TIMEcontentpation, AVGDISTActualIcon and MOV CaitManipulations. These features appear in both of the top-two feature sets, and score a 50% usefulness in feature selection. NUMIcons, which was crucial in On Task classification, appears in the best-case feature set for Logical.

<table>
<thead>
<tr>
<th>Number of folds (%)</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 (50 %)</td>
<td>4 MOV CaitManipulations</td>
</tr>
<tr>
<td>4 (40 %)</td>
<td>7 DIST actualIcon</td>
</tr>
<tr>
<td>4 (40 %)</td>
<td>13 TIME experiment BOTH</td>
</tr>
<tr>
<td>5 (50 %)</td>
<td>14 TIME contentpation</td>
</tr>
<tr>
<td>5 (50 %)</td>
<td>21 AVGDIST actual Icon</td>
</tr>
</tbody>
</table>

Table 4.6: Salient feature selection in Case 5 for Logical data.

Table 4.7 summarizes the usefulness of the categories of features.
<table>
<thead>
<tr>
<th>Feature Category</th>
<th>Level</th>
<th>Predictive Features</th>
<th>Pred.*</th>
<th>Triv.</th>
<th>Detr.</th>
<th>Conn.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement Counts</td>
<td>Experiment</td>
<td>TotManipulations, NUMIcons</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>–</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distances</td>
<td>Experiment</td>
<td>–</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>ActualIcon</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Timing</td>
<td>Experiment</td>
<td>Contemplation, ExperimentQUES, ExperimentBOTH</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>ConsideringIcon</td>
<td></td>
<td>X**</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Self-Report</td>
<td>Experiment</td>
<td>SelfPercep</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* A category is only considered predictive if multiple elements of that category were members of a best feature set.

** Icon-level timing is considered “trivial” in this case because when these are added to Case 3.1, it yields Case 2.3 with no loss of CCR. Case 3.3, which also includes these features, ranks near the performance median.

Table 4.7: Categories of features and their predictive qualities for Logical data.

The performance of the Logical data classification is notable. In its best case, it exceeds the lower-baseline of 61.49% by a healthy 12%. However, it is the lowest performer of the four salient subjective outcome assessments. This indicates that it is a suitable area for future research.

4.3 Analysis of Subjective Outcome Assessment: Key Elements

Key Elements measures whether the sketch has the most prominent few elements from the passage.

The classifier was tasked with separating sketches that have all or most of the prominent elements of the passage from the remainder. Key Elements is a prime case in our analysis. The best-case feature set for Key Elements achieves the highest CCR of any of the subjective outcome assessments. Table 4.8 details the best and worst classification cases.
<table>
<thead>
<tr>
<th>Feature</th>
<th>Segmented Case</th>
<th>Best Cases</th>
<th>Worst Cases*</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4.3</td>
<td>4.2</td>
<td>5</td>
</tr>
<tr>
<td>1 MOVCremoveType</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>2 MOVCremanipulations</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>3 MOVCootherTypeMoves</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>4 MOVCTotManipulations</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>5 NUMIcons</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>6 DISTtotalIcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7 DISTactualIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 DISTSdiffBtwntTotalandActual</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>9 TIMETotManiplcon</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12 TIMEexperimentQUES</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>14 TIMEcontemplation</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>21 AVGDISTActualIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23 AVGTIMEConsideringIcon</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>24 OTHERavgSelfPercep</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Feature Count</th>
<th>10</th>
<th>6</th>
<th>2</th>
<th>6</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct Classification Rate</td>
<td>75.78%</td>
<td>73.91%</td>
<td>73.91%</td>
<td>64.60%</td>
<td>65.22%</td>
</tr>
</tbody>
</table>

*The full set of features was also a “worst case”. As it does not further the discussion of predictive and detractive features, we choose to omit it.

**Table 4.8: Best and worst feature sets for Key Elements data.**

The two worst-case feature sets rely primarily on experiment-level move count and distance data. The experimental-level move count information is not harmful to classification in and of itself, as the best case relies on a subset of it combined with features from other categories. The second-worst case, Case 2.1, adds MOVCremanipulations, NUMIcons and DISTSdiffBtwntTotalandActual to Case 4.1 causing a slight uptick in CCR (0.9%, from 64.60% to 65.22%).

Case 5 has astoundingly solid performance with only two features – NUMIcons and AVGTIMEConsideringIcon. NUMIcons is significant for many reasons:

- As previously noted, it improved performance for the second-worst case.
- It is a member of all three of the best feature sets.
- It was one of two features crucial to the classification of On Task data. (It was also useful, to a lesser degree, for classifying the Logical data.)
• The feature selection algorithm ranked it at a 100% usefulness score in both On Task and Key Elements.

The pairing of these two features suggests that a large amount of the calculation of Key Elements hinges upon whether subjects carefully considered their work. Part of this consideration is whether the subject used enough icons to represent the key parts of the passage, yet did not litter the canvas with too many icons (as several of the unfocused sketches did). Table 4.9 details the salient features for Case 5.

<table>
<thead>
<tr>
<th>Number of folds (%)</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>10(100 %)</td>
<td>5 NUMIcons</td>
</tr>
<tr>
<td>4( 40 %)</td>
<td>23 AVGTIMEConsideringIcon</td>
</tr>
</tbody>
</table>

Table 4.9: Salient feature selection in Case 5 for Key Elements data.

Case 4.3, the best case, can be seen as adding features to the pair in Case 5 to refine the classification. The best case uses a trio – TIMEcontemplation, AVGTIMEConsideringIcon, AVGDISTActualIcon – that indicate how much time subjects are spending thinking about adjustments to their drawing, and how small or large those adjustments end up being upon completion.

Additional features comprise Case 4.3, the best-case subset:

• DISTDiffBtwnTotalandActual seems to be the only useful features of the experiment-level distance.

• Experimental-level data for manipulations, remanipulations and the number of other moves are added. (Feature selection ignores move type moves, which is effectively included in the combination of manipulations and other moves.) These features measure the higher-order functions of the software, features that are necessary for drawings that seek to accurately include the most salient story elements.

• TIMEexperimentQUES is included. The classifier was not able to build a sufficient model for the objective category of multiple choice scores. But it makes logical sense that a subject who spends the time to answer multiple choice questions (which draw from different minutia of the
story) is a subject who has scanned the passage repeatedly and has made note of the key elements.

- The self-assessment feature is also included.

Case 4.2, the second best case, has the same performance as Case 5 with triple the feature set membership. This case notably adds the self-report measure of the subject’s confidence and some higher-order movement counts; it removes AVGTIMEConsideringIcon. Clearly the subject’s self-perception combined with their demonstrated use of higher-order functions of the software reveals a similar amount of information about their deliberative nature that the icon-level consideration timing did.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement Counts</td>
<td>Experiment</td>
<td>NUMIcons, Remanipulations, OtherTypeMoves, TotManipulations</td>
<td>X*</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>–</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Distances</td>
<td>Experiment</td>
<td>DiffBtwTotal&amp;Actual</td>
<td></td>
<td></td>
<td>X*</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>ActualIcon</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Timing</td>
<td>Experiment</td>
<td>Contemplation, ExperimentQUES</td>
<td>X</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>ConsideringIcon</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Self-Report</td>
<td>Experiment</td>
<td>SelfPercep</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

* Both Movement Counts and Distances appear in the worst two cases. Movement Counts is also present in the best two cases whereas the majority of Distances are absent.

Table 4.10: Categories of features and their predictive qualities for Key Elements data.

4.4 Analysis of Subjective Outcome Assessment: Precision

The Precision assessment details how accurately the drawing portrays the passage. The classifier was tasked with separating diagrams that convey the meaning of the story, even with some technical errors, from those that are wholly unrelated. Table 4.11 details the best and worst feature sets for classification.
<table>
<thead>
<tr>
<th>Feature Count</th>
<th>Correct Classification Rate</th>
<th>6</th>
<th>8</th>
<th>8</th>
<th>8</th>
<th>7</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feature</td>
<td>Segmented Case</td>
<td>Best Cases</td>
<td>Worst Cases</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------</td>
<td>------------</td>
<td>-------------</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>MOVCmoveType</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MOVCremanipulations</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MOVCotherTypeMoves</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MOVCtotManipulations</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>NUMicons</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DISTtotalIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>DISTactualIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>DISTDiffBtwTotalandActual</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>TIMETotManipIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>TIMETotConsideringIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>TIMEexperimentDRAW</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>TIMEexperimentQUES</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>TIMEexperimentBOTH</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>TIMEcontemplation</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>TIMEreflection</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>NUMtextScroll</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>AVGMOVCMoveTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>AVGMOVCremanipulations</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>AVGMOVCotherTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>AVGDISTTotalIcon</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>AVGDISTactualIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>AVGTIMEManipIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>AVGTIMEConsideringIcon</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>OTHERavgSelfPercep</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.11: Best and worst feature sets for Precision data.

Precision has a strong performance in the classification process. The best-case CCR of 71.43% is a robust 38.5% over the lower-baseline measure. There are several strikingly different characteristics of the Precision classification. First, all three of the best-cases and two of the worst-cases tend to concentrate on the experiment-level features. The best sets concentrate on movement and distance. Two of the worst sets concentrate on timing (the third comprises all the icon-level features). The third-best and third-worst each add a pair of features from another category. And, sets that mix more types of features tend toward the median classification performance. This stands in stark contrast to previous
classifications, where mixed sets outperformed the homogeneous sets. Moving from the second-best to the best feature set, we shed two elements to comprise only six. In the cases of the other subjective outcome assessments that measured diagram quality (Logical and Key Element), the reverse was true: larger, ten-element sets performed best.

The combination of fewer features and more homogeneity signifies that there is less complexity to classifying which sketches were constructed with more Precision. The top two feature sets for the Precision classification completely rely on the more simplistic experiment-level summary data. The third-best feature set adds only one icon-level feature – AVGDISTActualIcon – which it prefers over the lower-complexity MOVCMoveType. Reinforcing the aversion to icon-level features, this substitution is rated as a 40% usefulness (see Table 4.12) and coupled with 2.6% drop in CCR. Further, the second worst feature set draws all icon-level features.

It turns out that gauging Precision is a simpler task than the previous assessments. From the best features, it is clear that “overthinking” precision by looking at individual icon-level transforms degrades classification. And, the composite feature, DISTDiffBtwnTotalandActual, which was useful in Logical and Key Elements, is replaced by its elemental parts: DISTTotalIcon and DISTActualIcon. The speed at which subjects contemplate and reflect on icon placements and the experiment itself, also does not correlate well. Of the timing elements, only the summary item TIMETotManipIcon matters. This again is different because in every other assessment it was a member of the worst case feature sets. Here it scores a 70% usefulness score in the Case 5 feature selection. Further evidence of the detriment of timing elements is that two of the three worst-case feature sets have all of the experiment-level timing features as members. Table 4.12 describes the feature selection for Case 5.
<table>
<thead>
<tr>
<th>Number of folds (%)</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>8( 80 %)</td>
<td>3 MOVCotherTypeMoves</td>
</tr>
<tr>
<td>10(100 %)</td>
<td>4 MOVClotManipulations</td>
</tr>
<tr>
<td>5( 50 %)</td>
<td>5 NUMIcons</td>
</tr>
<tr>
<td>4( 40 %)</td>
<td>6 DISTtotalIcon</td>
</tr>
<tr>
<td>6( 60 %)</td>
<td>7 DISTactualIcon</td>
</tr>
<tr>
<td>7( 70 %)</td>
<td>9 TIMETotManipIcon</td>
</tr>
<tr>
<td>4( 40 %)</td>
<td>21 AVGDISTActualIcon</td>
</tr>
<tr>
<td>4( 40 %)</td>
<td>24 avgSelfPercep</td>
</tr>
</tbody>
</table>

Table 4.12: Salient feature selection in Case 5 for Precision data.

Thus, measuring Precision is done with features that are the closest to the task at hand, not extraneous features or the intermediate corrections. It matters how often and how far icons are moved, and how long the subject takes to make those movements. The previous sections demonstrated that when a subject reflects on a movement or contemplates the next, it helps us to determine if they are thinking about the Logical layout of the sketch and including the right icons to satisfy the Key Elements. But it is the time spent on each icon placement, from mouse-up to mouse-down (TotManipIcon), that indicates whether a sketch precisely represents the scene. Users are also valued by their use of otherTypeMoves, which are needed for a precise sketch (but not necessarily for one that is Logical or has the salient elements).

In summary: The top four features by usefulness from Case 5 feature selection are MOVClotManipulations, MOVCotherTypeMoves, TIMETotManipIcon, DISTactualIcon. Three of these four appear in the top-three feature sets. Taken together, these features focus only on the core task of sketching. The classifier ignores thinking, startup and close out times. And, only the number of actions are of concern, not the number of corrections. The resulting correction distances (DISTactualIcon), and not the intermediate movements or re-manipulations, matter to the classifier. This is a highly end-result focused algorithm.
Table 4.13: Categories of features and their predictive qualities for Precision data.

4.5 Analysis of Objective Grade Data: Michigan Test

The Michigan test is a standard test of English proficiency. (More information about the examination may be found at http://www.cambridgetest.org/exams.) The classifier was tasked with separating passing from failing test scores. Table 4.14 presents the best and worst feature sets for the classifier.
Table 4.14: Best and worst feature sets for Michigan data.

<table>
<thead>
<tr>
<th>Feature Count</th>
<th>Feature</th>
<th>Segmented Case</th>
<th>Best *</th>
<th>Worst **</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>MOVEmoveType</td>
<td>X</td>
<td>5.0</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>MOVECremanipulations</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MOVCotherTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MOVCtotManipulations</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>NUMIcons</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>DISTtotalIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>DISTactualIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>DISTDiffBtwnTotalandActual</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>TIMEreflection</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>AVGTIMEConsideringIcon</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Correct Classification Rate

<table>
<thead>
<tr>
<th>Feature Count</th>
<th>Feature</th>
<th>Segmented Case</th>
<th>Best *</th>
<th>Worst **</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>MOVEmoveType</td>
<td>X</td>
<td>5.0</td>
<td>2.1</td>
</tr>
<tr>
<td>2</td>
<td>MOVECremanipulations</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MOVCotherTypeMoves</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MOVCtotManipulations</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

* The spread between the best feature set (CCR = 68.12%) and the second best (CCR = 63.04%) was too large for us to draw parallels across multiple best sets. Further, the second best set has a CCR below the lower-baseline of 64.49%.

** Similarly, all feature sets other than Case 2.1 were tightly clustered between with CCRs in the range of [60.14%,63.04%].

The classification process for the Michigan data did not yield many viable feature sets. All but the best case were below the lower baseline. (As was noted in Chapter 3, this does not mean that the algorithms are malfunctioning. It merely means that an algorithm that skews to one side would outperform these cases.) Case 2.1, comprising only experimental-level movement and distance features, performed particularly poorly.

On the other hand, our best feature set, Case 5, is rather elegant. It comprises three features: NUMIcons (used in On Task, Key Elements and Logical), TIMEreflection, AVGTIMEConsideringIcon (used in Key Elements and Logical). This trio tells the story of how thoughtful a subject was, with the raw number of icons used, and the time taken to think before each icon and after the whole experiment. It makes sense that such a set could be somewhat predictive of how the students would perform on the Michigan test. A student who is thoughtful about what comprises a drawing is likely also to be thoughtful about each question on a standardized comprehension examination. Table 4.15 further reinforces this by assigning a
100% weight to the reflection time and smaller, but significant, weights to the icon-count and icon-contemplation time.

<table>
<thead>
<tr>
<th>Number of folds (%)</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>6 (60%)</td>
<td>5 NUMIcons</td>
</tr>
<tr>
<td>10 (100%)</td>
<td>15 TIMEreflection</td>
</tr>
<tr>
<td>5 (50%)</td>
<td>23 AVGTIMEConsideringIcon</td>
</tr>
</tbody>
</table>

Table 4.15: Salient feature selection for Michigan data.

<table>
<thead>
<tr>
<th>Feature Category</th>
<th>Level</th>
<th>Predictive Features</th>
<th>Pred.</th>
<th>Triv.</th>
<th>Detr.</th>
<th>Conn.*</th>
</tr>
</thead>
<tbody>
<tr>
<td>Movement Counts</td>
<td>Experiment</td>
<td>NUMIcons</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>–</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Distances</td>
<td>Experiment</td>
<td>–</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>–</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Timing</td>
<td>Experiment</td>
<td>Reflection</td>
<td></td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Icon</td>
<td>ConsideringIcon</td>
<td></td>
<td></td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Self-Report</td>
<td>Experiment</td>
<td>–</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

* The best feature set does not draw more than one feature from any feature category. Thus we are unable to report on predictive categories for the best-case feature set.

Table 4.16: Categories of features and their predictive qualities for Michigan data.

Table 4.16, above, summarizes the usefulness of features.

Chapter 5 organizes the separate conclusions presented in this chapter and suggests how these conclusions may be applied to future work.
Chapter 5 – Broader Conclusions for Future Work

5.0 A Holistic View of the Results

This experiment allowed us to examine the categories of features that guided the classification process. In Sections 4.1 – 4.4, we discussed the composition of optimal feature sets for the subjective outcome assessment categories – On Task (a derivation from Baker (2007), Logical (a derivation from McLaren et al. (2005) and Carberry et al. (2006)), Key Element, and Precision (these last two derived from Hirschman et al. (1999)). And in section 4.5 we described that for a set of Objective Grade Data, specifically the Michigan Test.

<table>
<thead>
<tr>
<th>Subjective Outcome Assessment</th>
<th># (%) of features in optimal subset</th>
</tr>
</thead>
<tbody>
<tr>
<td>On Task</td>
<td>2 (8.3%)</td>
</tr>
<tr>
<td>Logical</td>
<td>10 (41.6%)</td>
</tr>
<tr>
<td>Key Element</td>
<td>10 (41.6%)</td>
</tr>
<tr>
<td>Precision</td>
<td>6 (25%)</td>
</tr>
<tr>
<td>Objective Grade Data</td>
<td></td>
</tr>
<tr>
<td>Michigan</td>
<td>3 (12.5%)</td>
</tr>
</tbody>
</table>

Table 5.1: Size of optimal feature sets by classification category.

Table 5.1 notes the number of features in the optimal subsets and the corresponding portion of the total feature set (24). The sheer number of different features that were needed to achieve the best-case classification of Logical and Key Element – and Precision to a lesser degree – demonstrate that the features presented in this experiment were appropriate for the task presented to subjects.
This section reviews which individual features and categories of features are most useful to the classification process and how other researchers might make use of this information in derivative works. Table 5.2, above, aggregates the summaries of assessment-by-assessment analysis from Chapter 4. While we seek to draw generalities, the reader should be reminded that the previous chapter presented some finer details that are masked by this summary table.

### 5.1 Movement Features

Features from the experiment-level movement counts were part of every subjective outcome assessment’s classification, and every best-case set. In four of the five cases presented, multiple elements combined allowed movement features to have the most significant effect on the classification. But in each of those successful cases, these features had to be properly paired with ones from other
feature categories. Used indiscriminately as a single block, movement features have negative effects. Case in point: two of the worst-case feature sets comprised all movement features.

Recall Hilbert & Redmiles’ (2000) framework, described in Chapter 1, that discussed levels of abstraction of interface actions. Our levels of abstraction for features are how far removed a measurement is intellectually from the task at hand. Movement features come at two levels of abstraction.

Examine two of the simplest, lowest-level movement features, most closely related to the task at hand:

- **NUMIcons**, the simplest feature to record for this type of experiment, was essential in all cases but Precision. And, Michigan only made use of NUMIcons. A subject that uses a reasonable number of icons is at the very least paying attention to the passage and the task at hand. Thus the use in On Task and Michigan makes sense, where a subject who is accustomed to paying attention to the passage will likely be the same person to stay with an exercise and earn higher scores on a standardized examination.

- **TotManipulation**, which is the second simplest feature, counts any type of icon manipulations. It was useful in classifying the categories that dealt with the appearance of the diagram (Logical, Key Elements, Precision). And, Logical only uses this and the previous movement feature. It stands to reason that a subject that has a similar amount of basic movement actions as others would score commensurate with their peers. Some reasonable number of actions would be necessary, but not sufficient, to develop a reasonable sketch. (Sufficiency comes from features described in the next two sections.)

Future experiments involving HCI for ITS should determine what actions are closest to the core of completing the exercise and choose an abundance of these features.

At one higher level of abstraction, the more sophisticated movement features demonstrate a user’s proclivity toward self-correction and use of the higher-order functions of the software. (Recall that Baker’s (2007) work used judicious use of the help feature to ascertain when a subject recognizes their mistakes,
and their willingness to embrace higher-order functions. Our features are a successful evolution, in that they are less human-intensive and intrusive to the experiment process.) Examine the two higher-order features:

- **Remanipulation**: Recall from Chapter 1 that Remanipulation is a novel feature that examines self-correction. (It is our extension of Olsen and Halversen’s (1988) tracking of user “undoing” in their work.) This, and the next feature is useful in the classification of On Task and Key Elements. It stands to reason that when judging On Task behavior, a subject that engages in self-correction and avails themselves of advanced functions is more likely to be paying attention to task. In fact, this was so crucial to On Task that the feature was used both at the experiment-level and in the examination of individual icon-level averages. Similarly, this attention to detail translates well into a sketch containing the Key Elements.

- **OtherTypeMoves** demonstrates the use of non-movement features of the interface. The above categories use it coupled with remanipulation. Precision uses this feature, coupled with self-correction distances, to get a sense for the subject-turned-artist, who finely refines icon placements and finishes the sketch with non-move tweaks. We discuss this more in the next section.

Future interfaces should include more features that are directly applicable to the task at hand. In this style of experiment, more emphasis should be placed on the movements of icons themselves. Perhaps more granular analysis of the directionality of movement and the changing of the clustering-density of icons in the drawing could provide useful classification insights for future rounds of experiments. The latter, for example, might provide enough information to strengthen the Cognitive Economy assessment.
5.2 Distance Features

Distance features sit one higher level of abstraction from the rudimentary movement features of the previous section. These record actions in slightly more detail than simple counts of moves. At the experiment-level, their use in classification is mixed. As a whole, these act as a detractor for the Michigan and Key Element assessments, but a predictor for Logical and Precision. The only icon-level feature of use is ActualIcon for Logical and Key Elements. Otherwise this block of features is trivial or degrades classification.

What had the most substantive effect were the distance features that expressed the difference between total icon distance (where the icon had been dragged in the interim) and actual icon distance (the difference from start to finish). There are three distinct cases of their use:

- On Task uses the composite **difference between total and actual distances**. Key Elements does this as well, adding icon-level **average actual icon distances**. The composite score adds additional information to the remanipulation data. The algorithm now not only knows that one is correcting their work, but also how dramatic those changes are. A large difference with low remanipulation count would mean that icons were moved far from their final resting point. When determining whether someone is giving the proper attention to the exercise and the salient elements of a sketch, this is a reasonable tool.

- Then there is the icon-level **average actual icon distance**. Whereas the composite difference data shows the overall extent of all changes, that number does not let the system discern whether one icon was moved large distances (much depth) or many icons were moved short distances (much breadth). On Task adds icon-level average remanipulation counts, which is a low-grade indicator of the breadth of changes. Key Elements adds average actual icon distances to more accurately gauge the depth of changes. Combined, the pairing gives the Key Elements classifier a rough approximation of the self-corrections. Logical uses only this average actual icon distance; this makes sense as the assessment is only concerned with how icons are moved in relation to one another.
Precision, which was discussed in Section 5.1, pairs movement features with the separate TotalIcon and ActualIcon experimental-level distances. Despite its more simplistic classification (discussed in Section 4.4), Precision is the most sophisticated sketch assessment. Thus the classification sheds the remanipulation and distance difference summary features and opts for the full picture of the movements from the raw icon distances. And, as will be noted in the next section, Precision is the only assessment to use the raw movement-time feature.

The lesson of this section for future ITS-related experiments is to provide features that measure the frequency of self-correction, and to supplement these features with others that measure the depth to which the user makes corrections. Providing this data at different raw and composite levels can provide insight into the quality of the produced work.

### 5.3 Timing Features

Although the experimental-level and icon-level feature groups are detractors as a whole, individual timing features played a role in each assessment category. As will be evident, our features have gone far beyond Baker’s (2007) simple time-on-task features to include timings associated with different actions in the experiment.

The value of the time spent by subjects in carefully considering work came across in all assessments except Precision:

- On Task uses experiment-level contemplation and reflection timing. These are low-level measurements of how much time a subject spends before and after the experiment, indicating if they are rushing to “click through” things. This is what Baker (2007) terms "gaming the system" and is our answer to time-on-task in Ardito et al. (2006), and Olsen & Halversen (1988). On Task seems not to need as much detail of time spent considering intermediate steps as the categories
of the next point. Instead, as previously mentioned, it gathers a rough estimate of the intermediate steps with the icon-level remanipulation feature.

- Key Elements and Logical use experiment-level contemplation, time-on-questions, and icon-level consideration time. (Logical also adds total experiment time, which also provides the sketching time in addition to the aforementioned question time.) Developing a sketch that makes meaningful placements of icons and gathers the salient elements of the story requires thoughtfulness on the part of the subject. It is not enough to simply measure their click-through speed with experiment contemplation time. Their pace on each elemental movement (consideration) is also significant.

Precision was alone in focusing on the raw time spent manipulating icons (mouse-up to mouse-down). This is consistent with the classifier’s selections of the most detailed data in previous categories.

### 5.4 The Self-Assessment and Multiple Choice Questions

As we discussed in our opening chapter, multiple choice questions may be unreliable predictors of comprehension. Our input interaction features are also unable to support respectable classification results for the multiple choice questions. But, as the timing section demonstrated, the process of answering the questions is not without merit: The time that a subject takes on these questions can be used as evidence of their ability to perform key tasks. Such timing data draw parallels to long-hand tests where the final answer does not matter as much as the series of steps the student employed. But it should be noted that our experiment was not wholly unable to gauge multiple choice performance: it possessed a moderate ability to predict the Michigan scores.

The self-report satisfaction measure, that we derived from Ardito et al.’s (2006) work, is another interesting feature. It appears in one of the top feature sets in each of the categories of On Task, Key Elements, Logical and Precision. (Though it is not in the best feature set of Key Elements and Logical.)
This implies that subjects have a sense of when they did well, when they appropriately selected some of the key components of the sketch, and when they arranged icons sensibly. However, the subjects might not be fully aware of their deficiencies in understanding the passage perfectly and constructing a precise sketch.

This disconnect is significant for Intelligent Tutoring Systems. Often we expect struggling students to summon help from a teacher or a tutorial feature. But if students ultimately do not know what they do not know, they cannot always request that help. As noted in Chapter 1, Iqbal and Bailey (2008) and Horvitz and Apacible (2003) asked questions and provided notification boxes to calculate the user’s intention on the fly. Baker’s (2007) work also relied on students asking for help. The features recorded in our experiment seem to be able to do this without such an interface intrusion and could eventually be used to engage automatic student support routines. Further, our computerized assessments of Precision seem to surpass the students’ self-reporting. Such automated interventions can make a difference in educational task performance, and ultimately in an education.

5.5 Icon-Level Features

On the surface, Table 5.2 seems to indicate that icon-level features have little use. As an undivided block, these features are at best trivial and at worst harmful to classification. But, revisiting Table 3.3, individual features from those at the icon-level stand out:

- The icon-level **remanipulations** assisted the On Task and Cognitive Economy assessments.
- **Actual icon distance** appeared in Cognitive Economy (with 80% usefulness), Logical, Like Scene and Precision. It was particularly useful in Logical and Key Elements.
- **Icon consideration time** appeared in Cognitive Economy, Key Element and Michigan.
- **Move counts of other moves** appeared in Like Scene.
The subjective outcome assessment of Cognitive Economy drew three of the six features from the icon-level pool and two of five features for Like Scene. Neither of these categories were examined in detail in Chapter 4 because the CCR for these two categories were too low. A future iteration of the experiment may include more types of icon-level features that may provide much needed data to these categories.

5.6 For Future Experiments

This experiment should be considered the first step toward more research in HCI with ESL students. We conclude with several caveats for future researchers:

- This experiment should be repeated with a larger subject pool. The first attempt at classification in tertiles revealed some individual features that were useful in detecting just the lowest performers or just the highest performers. Yet, there was not enough data to achieve strong results across the tertiles for most categories; the small amount of test data was further confounded by uneven distribution into the tertiles. The reduction to binary measures allowed us to demonstrate the efficacy of our feature categories but at a cost of identifying a features niche for a very low- or very high-performing student.

- As noted in our review of the literature, Singley and Lam (2005) looked to use educational task statistics to provide teachers with monitoring updates about their students. A variant of this experiment could be performed where an instructor is alerted of struggling subject and intervenes with explanations. Performance can be measured against non-intervention control experiments, or the data features for the experiment can be examined pre- and post-intervention to detect potential improvement. Further, we might leverage long-term trends as Singley and Lam suggested, by loading objective grade data into the system pre-experiment and using it as part of the monitoring criteria.

- The next generation of experiment software could include real-time prompts for when the features indicate that the user clearly understood some of the story, and included some of the key elements, but could not complete the diagram. The ITS could provide hints and or ask these
students to think again before submitting. These hints could be subjectively rated by the user or provided alongside a control experiment for objective comparison.

- That our analysis separated movement (and other) features into the lower and higher levels of abstraction reinforces the continued strength of Hilbert and Redmiles’ (2000) framework. Future attempts at classification may benefit from restructuring the groupings of feature types. Instead of primarily homogeneous groups by type (movement, distance, timing), cross-cutting groups can be formed by higher-order and lower-order functions of the interface.

- This experiment controlled for educational level with the selection of the classes and the selection of the passages. The selection of future study participants should also control for experience in using computers. Variability in age and comfort with computers (and dexterity with the use of a mouse) may have contributed to some of the features registering little utility. With icon-level experiment timing, for example, only the pauses to consider icons were useful, other averages may have been too wildly variable from user-to-user. (Speed with software may be as much a function of a subject’s comfort with use of a computer as it is of understanding the materials.) Icon level movements around the canvas, as well, might include additional stopping points if one is not a comfortable mouse user.

As society relies more on intelligent tutoring systems and big data for its interventions, work in this area will become ever more crucial. The positive indications from our analysis are encouraging for future researchers to perform experimentation of this kind, perhaps embracing newer AI techniques and tools, and including mobile technologies, gaming and HCI. Such advances will facilitate easier design and implementation of future experiments.
Appendix A – Database Structure

These features are used to create the calculated features for the study.

dataIconSet associates a "set" of icons with a passage to be used in experiment.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>passageID</td>
<td>INT</td>
<td>passage</td>
</tr>
<tr>
<td>iconID</td>
<td>INT</td>
<td>icon</td>
</tr>
</tbody>
</table>

dataIconBank complete set of icons for all experiments.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>iconID</td>
<td>INT</td>
<td>unique icon identifier</td>
</tr>
<tr>
<td>filename</td>
<td>VARCHAR</td>
<td>icon file on disk</td>
</tr>
<tr>
<td>shortdesc</td>
<td>VARCHAR</td>
<td>human-readable icon description</td>
</tr>
</tbody>
</table>

dataPassages passages for all experiments.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>passageID</td>
<td>INT</td>
<td>unique passage number</td>
</tr>
<tr>
<td>text</td>
<td>LONGTEXT</td>
<td>text of passage</td>
</tr>
</tbody>
</table>

experimentIcons icons comprising a single experiment's drawing.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>drawingID</td>
<td>INT</td>
<td>experiment referenced</td>
</tr>
<tr>
<td>iconID</td>
<td>INT</td>
<td>icon placed on experiment canvas</td>
</tr>
<tr>
<td>locationX</td>
<td>INT</td>
<td>X-coordinate on canvas</td>
</tr>
<tr>
<td>locationY</td>
<td>INT</td>
<td>Y-coordinate on canvas</td>
</tr>
</tbody>
</table>

experimentDrawings an original or single experiment's subject-generated drawing.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>drawingID</td>
<td>INT</td>
<td>experiment number</td>
</tr>
<tr>
<td>participantID</td>
<td>INT</td>
<td>unique participant identifier</td>
</tr>
<tr>
<td>filename</td>
<td>VARCHAR</td>
<td>drawing file on disk</td>
</tr>
<tr>
<td>passageID</td>
<td>INT</td>
<td>passage on which experiment is based</td>
</tr>
</tbody>
</table>

experimentParticipants subjects of all experiments

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>participantID</td>
<td>INT</td>
<td>subject's unique id</td>
</tr>
<tr>
<td>numberOfExperiments</td>
<td>INT</td>
<td># of experiments participated in, successful or not.</td>
</tr>
</tbody>
</table>

logPerMove individual moves within a drawing experiment.

<table>
<thead>
<tr>
<th>COLUMN_NAME</th>
<th>TYPE_NAME</th>
<th>REMARKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>moveSeq</td>
<td>INT</td>
<td>unique move ID</td>
</tr>
<tr>
<td>COLUMN_NAME</td>
<td>TYPE_NAME</td>
<td>REMARKS</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>iconID</td>
<td>INT</td>
<td>icon being manipulated</td>
</tr>
<tr>
<td>experimentID</td>
<td>INT</td>
<td>current experiment</td>
</tr>
<tr>
<td>startLocX</td>
<td>INT</td>
<td>starting X-coordinate of icon</td>
</tr>
<tr>
<td>startLocY</td>
<td>INT</td>
<td>starting Y-coordinate of icon</td>
</tr>
<tr>
<td>endLocX</td>
<td>INT</td>
<td>ending X-coordinate of icon</td>
</tr>
<tr>
<td>endLocY</td>
<td>INT</td>
<td>ending Y-coordinate of icon</td>
</tr>
<tr>
<td>timeStart</td>
<td>DATETIME</td>
<td>clock time at start of move</td>
</tr>
<tr>
<td>timeEnd</td>
<td>DATETIME</td>
<td>clock time at end of move</td>
</tr>
<tr>
<td>logPerExperiment</td>
<td></td>
<td>log of individual drawing experiments.</td>
</tr>
</tbody>
</table>

Table 6.1: Explanation of raw data collected in each database table.
Appendix B – Participants’ Subjective Self-Assessment

This form was provided to subjects before their use of the software.

Student Code # ___________________
(Please do not write your name on this form.)

1. How correctly (accurately) do you think your diagram explains the story?
Write in the drawing number and check one box.

<table>
<thead>
<tr>
<th>Drawing #</th>
<th>Perfectly Accurate</th>
<th>Accurately</th>
<th>Somewhat Accurately</th>
<th>Somewhat Inaccurately</th>
<th>Not At All</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2. How well did you understand this system?
Perfectly A lot Somewhat Not Very Well Not at All
[ ] [ ] [ ] [ ] [ ]

3. How well did you understand the stories?
Perfectly A lot Somewhat Not Very Well Not at All
[ ] [ ] [ ] [ ] [ ]

4. How much did you enjoy this exercise?
A lot Somewhat Not Very Well Not at All
[ ] [ ] [ ] [ ]
Appendix C – Extensions to Game Table for DMS

Client-Side Code

- DBObjec.java: Parent class of all client data objects, handles query construction and database & network communication.
- ExperimentLog.java: logs information about an experiment completed, including some client-determined features.
- LogActivities.java: logs non-drawing actions (scroll-bar movement).
- LogAnswer.java: logs selected multiple choice answers and the correctness.
- LogMoves.java: logs individual actions (icon placement, movement and transforms).
- Participant.java: handles gathering of login data from database and updating login activity.
- QueryConstants.java: Provides configuration information for the location of the experiment server.

Server-Side Code

- ExperimentLog.java: implemented for redundancy, writes action information to a local text file.
- ExperimentDataServer.java: the data logging server that stores information to a remote SQL database.
import java.text.SimpleDateFormat;
import java.net.*;
import java.sql.*;
import java.io.*;
import java.awt.image.BufferedImage;
import javax.imageio.ImageIO;

public class DBObject {
    SimpleDateFormat format = new SimpleDateFormat("yyyy-MM-dd hh:mm:ss");
    String query = ""; // query string to send to server
    BufferedImage image = null;

    protected String escapeString( String unesc ) {
        // generic code for escaping a string so that it is DB friendly goes here.
    } // escStr

    // writes a query, expects a response
    public Serializable writeWithResponse( int qtype ) {
        // Since ResultSets are not serializable (transmittable over the network),
        // we'll need a return vehicle to return a single Integer to Experiment Log,
        // a list of Integers to Participant, and a list of Strings to Participant.
        try {
            // creates and connects socket.
            Socket netOut = new Socket( QueryConstants.RELAYSVR,
                                          QueryConstants.RELAYPORT );

            ObjectOutputStream dos = new ObjectOutputStream( netOut.getOutputStream() );
            ObjectInputStream  dis = new ObjectInputStream( netOut.getInputStream() );

            // prevent unauthorized clients with a codeword.
            dos.writeUTF( QueryConstants.RELAYPASS );
            dos.flush();

            // send the query across the network
            dos.writeObject( query ); // modified to send a string object
            dos.writeInt( qtype );
            dos.flush();

            // SEND IMAGE QUERY: sending image as raw byte array, preceeded by size
            if ( qtype == QueryConstants.INSERT_RT_LOGID ) {
                if ( image != null ) {
                    // compose jpeg
                    ByteArrayOutputStream baos = new ByteArrayOutputStream();
                    ImageIO.write(image, "jpeg", baos);

                    // write size to server
                    dos.writeInt( baos.size() );
                    if (QueryConstants.DEBUGMD) System.err.println(
                        "Writing image of size:" + baos.size() );
                    dos.flush();

                    // write bytes to server
                    baos.writeTo( dos );
                    dos.flush();
                } // if
            } // if image
            else {
                dos.writeInt( 0 );
                dos.flush();
            } // else
        } // try
        Serializable ast = (Serializable) dis.readObject();
    } // writeWithResponse
}

79
netOut.close();
image = null; // needed to zero out image
return ast;
} // yrt
  catch (Exception e) {
    if (QueryConstants.DEBUGMD) System.err.println("DBObject ERROR in writeWithResponse: " + e.toString());
    return null;
  }
} // wWR

// writes a query, expects no response
public void writeNoResponse () {
try {
  // creates and connects socket.
  if (QueryConstants.DEBUGMD) System.err.println("DBObject OPENING CONNECTION.");
  Socket netOut = new Socket( QueryConstants.RELAYSVR, QueryConstants.RELAYPORT );
  ObjectOutputStream dos = new ObjectOutputStream( netOut.getOutputStream() );
  ObjectInputStream dis = new ObjectInputStream( netOut.getInputStream() );
  // prevent unauthorized clients with a codeword.
  dos.writeUTF( QueryConstants.RELAYPASS );
  dos.flush();
  // send the query across the network
  dos.writeObject( query );
  dos.flush();
  dos.writeInt( QueryConstants.SELECT_RT_NULL );
  dos.flush();
  dis.readInt(); // delay required to prevent premature
  // dropping of connection and thus query
  if (QueryConstants.DEBUGMD) System.err.println("DBObject CONNECTION CLOSED.");
  } // ytr
  catch (Exception e) {
    if (QueryConstants.DEBUGMD) System.err.println("DBObject ERROR in writeNoResponse()" + e.toString());
  } // hctac
} // DBO
Appendix D – Tools to Aide Software Development

make.sh

# a program to package the client and server software
jar cvfm DrawReadExperiment.jar experiment-manifest -C DrawReadExperiment/.
nar cvfm DrawReadServer.jar server-manifest -C DrawReadServer/.

server-manifest: JAR manifest for server program

Main-Class: troudtserver.ExperimentDataServer
Manifest-Version: 1.0
Class-Path: .:troudtserver:mysql-connector-java.5.1.13-bin.jar

create-tables.sql: SQL script used to create the database structure

delimiter $$
CREATE TABLE `experimentlogs` (  
`logID` int(11) NOT NULL AUTO_INCREMENT,  
`participantID` int(11) DEFAULT NULL,  
`experimentID` int(11) DEFAULT NULL,  
`startTimePart1` datetime DEFAULT NULL,  
`endTimePart1` datetime DEFAULT NULL,  
`startTimePart2` datetime DEFAULT NULL,  
`endTimePart2` datetime DEFAULT NULL,  
`image` longblob,  
`totalPogs` int(11) DEFAULT '0',  
`manipulation` int(11) DEFAULT '0',  
`iconManipulation` int(11) DEFAULT '0',  
`remanipulation` int(11) DEFAULT '0',  
`xDistance` int(11) DEFAULT '0',  
`yDistance` int(11) DEFAULT '0',  
`comment` varchar(50) DEFAULT NULL,  
PRIMARY KEY (`logID`)  ) ENGINE=MyISAM AUTO_INCREMENT=58 DEFAULT CHARSET=latin1$$
delimiter $$
CREATE TABLE `logactivities` (  
`logID` int(11) NOT NULL,  
`activityIndex` int(11) DEFAULT NULL,  
`type` varchar(45) DEFAULT NULL,  
`target` varchar(45) DEFAULT NULL,  
`startTime` datetime DEFAULT NULL,  
`endTime` datetime DEFAULT NULL,  
`actionValue` int(11) DEFAULT '0'  ) ENGINE=MyISAM DEFAULT CHARSET=latin1$$
delimiter $$
CREATE TABLE `loganswers` (  
`logID` int(11) NOT NULL,  
`question` varchar(500) DEFAULT NULL,  
`answer` varchar(500) DEFAULT NULL,  
`isCorrect` bit(1) DEFAULT NULL  ) ENGINE=MyISAM DEFAULT CHARSET=latin1$$
delimiter $$
CREATE TABLE `logmoves` (  
`logID` int(11) NOT NULL,  
`x` int(11) DEFAULT NULL,  
`y` int(11) DEFAULT NULL,  
`oldX` int(11) DEFAULT NULL,  
`oldY` int(11) DEFAULT NULL,  
`newX` int(11) DEFAULT NULL,  
`newY` int(11) DEFAULT NULL,  
`xDistance` int(11) DEFAULT NULL,  
`yDistance` int(11) DEFAULT NULL,  
`moveTime` datetime DEFAULT NULL,  
`isCorrect` bit(1) DEFAULT NULL  ) ENGINE=MyISAM DEFAULT CHARSET=latin1$$
CREATE TABLE `movetables` (  
  `moveIndex` int(11) DEFAULT NULL,  
  `iconID` int(11) DEFAULT NULL,  
  `startTime` datetime DEFAULT NULL,  
  `endTime` datetime DEFAULT NULL,  
  `iconName` varchar(50) DEFAULT NULL,  
  `locationX` int(11) NOT NULL,  
  `locationY` int(11) NOT NULL,  
  `size` int(11) NOT NULL,  
  `angle` float DEFAULT NULL,  
  `comment` varchar(20) DEFAULT NULL  
) ENGINE=MyISAM DEFAULT CHARSET=latin1$$

delimiter $$

CREATE TABLE `participants` (  
  `participantID` int(11) NOT NULL AUTO_INCREMENT,  
  `numOfExperiments` int(11) DEFAULT '0',  
  `userID` varchar(10) DEFAULT NULL,  
  `userPassword` varchar(45) DEFAULT NULL,  
  `firstLogin` datetime DEFAULT NULL,  
  `lastLogin` datetime DEFAULT NULL,  
  `totalLogin` int(11) DEFAULT '0',  
  PRIMARY KEY (`participantID`)  
) ENGINE=MyISAM AUTO_INCREMENT=2 DEFAULT CHARSET=latin1$$

clear-testpoints.sql: SQL script used to clear test data before actual experiments

DELETE FROM etroudt.experimentlogs;
DELETE FROM etroudt.logactivities;
DELETE FROM etroudt.loganswers;
DELETE FROM etroudt.logmoves;
UPDATE etroudt.participants SET firstLogin=null, lastLogin=null, totalLogin=0,
numOfExperiments=0;
Appendix E – Data Pre-Processing Tools

// AssociateSelfAssessment.java:
// uses fields "Draw 1 SA" -- "Draw 7 SA" to consolidate self-assessment scores.
// Does the following:
// 1. Find each experiment ID in order, associated with a particular user. (No need to print.)
// 2. Average Self Assessment of non-zero features per user
// 3. Raw Self Assessment Score associated with an experiment ID

import java.sql.*;

public class AssociateSelfAssessment {
    public static void main ( String args[] ) {
        try {
            // code for connecting to Access database here
            Connection con = DriverManager.getConnection( database ,"","");

            // TASK 1 -- a user's experiments
            Statement s = con.createStatement();
            s.execute("SELECT userID,logID FROM experimentlogs inner join participants on experimentlogs.participantID = participants.participantID order by userID,logID");

            // create and clear array
            int expIDs[][] = new int[9011][8];  // 9010 userIDs, 7 experiments [0][0] is empty
            for ( int i = 0 ; i < 9011 ; i++ )
                for ( int j = 0 ; j < 8 ; j++ )
                    expIDs[i][j] = 0;
            ResultSet rs = s.getResultSet();
            if (rs != null) {
                int userID = 0;
                int expN   = 0;
                while ( rs.next() ) {
                    int nUID = rs.getInt( "userID" );
                    if ( nUID != userID ) {
                        userID = nUID;
                        expN = 0;
                    } // fi
                    expN++;
                    expIDs[userID][expN] = rs.getInt( "logID" );
                } // elihw
            } // fi

            /*  For debugging purposes only.
            for ( int i = 0 ; i < 9011 ; i++ )
                for ( int j = 0 ; j < 7 ; j++ )
        } // elihw
    } // fi
*/
if ( expIDs[i][j] != 0 )
    System.err.println( Integer.toString(i) + "," + Integer.toString(j) + "," + Integer.toString(expIDs[i][j]) );
*/

// TASKS 2 & 3 -- the self-assessment numbers
s = con.createStatement();
s.execute("SELECT * FROM participant-outsidedata");
rs = s.getResultSet();
if (rs != null)
    while ( rs.next() ) {
        int[] scores = new int[8];
        int userID   = rs.getInt ( "UserID" );
        scores[1] = rs.getInt ( "Draw 1 SA" );
        scores[2] = rs.getInt ( "Draw 2 SA" );
        scores[3] = rs.getInt ( "Draw 3 SA" );
        scores[4] = rs.getInt ( "Draw 4 SA" );
        scores[5] = rs.getInt ( "Draw 5 SA" );
        scores[6] = rs.getInt ( "Draw 6 SA" );
        scores[7] = rs.getInt ( "Draw 7 SA" );

        int runningAVG = 0;
        int nonZeroSCR = 0;
        for ( int i = 1 ; i <= 7 ; i++ ) {
            runningAVG += scores[i];
            if ( scores[i] != 0 ) {
                nonZeroSCR++;
                // experiment ID and self assessment score
                if ( expIDs[userID][i] != 0 ) {
                    System.err.println( Integer.toString(expIDs[userID][i]) + "," + Integer.toString(scores[i]) );
                }
            }
        }

        if ( nonZeroSCR != 0 ) {
            runningAVG /= nonZeroSCR;
            // userID and average self-assessment score
            System.out.println(Integer.toString(userID) + "," + Integer.toString(runningAVG));
        }
    }
} // yrt

catch ( Exception exc ) {
    System.out.println( "Exception had: " + exc );
}
ExtractPictures.java: Extracts photos from SQL DB as strings, converts to JPG pictures and names by experiment ID and story ID.

```java
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.*;
import javax.imageio.*;
import java.io.*;

public class ExtractPictures {
    public static void main(String[] args) {
        try {
            // database connection code goes here

            // select number of images
            String query = "SELECT experimentID, participantID, logID, " +
                            "image FROM 'etroudt'. 'experimentlogs'";
            Statement stmt = conn.createStatement(ResultSet.TYPE_SCROLL_INSENSITIVE,
                                                          ResultSet.CONCUR_UPDATABLE);
            ResultSet rst = stmt.executeQuery(query);

            System.out.println("EXECUTED QUERIES.");
            if (rst == null) System.out.println("RST is NULL.");
            System.out.println(Integer.toString(rst.getMetaData().getColumnCount()));

            // foreach image in DB
            while (rst.next()) {
                // save non-null images
                if (rst.getString(4) != null) {
                    int experimentID = rst.getInt(1);
                    int participantID = rst.getInt(2);
                    int userID = rst.getInt(3);
                }
            }
        } finally {
```
byte[] img = rst.getBytes(4);
String filename = "EX" + Integer.toString(experimentID) + "_PT" + Integer.toString(participantID) + "_US" + Integer.toString(userID) + ".jpg";

FileOutputStream fospic = new FileOutputStream( filename );
fospic.write(img);
fospic.close();
}

// elihw
System.out.println("SAVED ALL FILES.");

// close database
conn.close();
// yrt

} // niam
} // ssalc

/******************************************
ComputeMoveStats.java:
Computes statistics on subject move data
that is not possible through typical DB
functions.

Specifically, this computes:
1. TOTAL time between moves.
2. TOTAL distance moved.
3. TOTAL time making moves.
4. AVERAGE time between moves.
5. AVERAGE distance moved.
6. AVERAGE time making moves.
7. TOTAL number of changing move types --
   defined as making a type of move, then
   choosing a new type of move.
8. TOTAL number of each types of move.
9. AVERAGE number of move segments per icon.
10. TOTAL number of remanipulations --
    defined as manipulating an icon that's
been moved before.

Usage:
java CMS > aggregatedt.txt 2> icontd.txt

import java.sql.*;
public class ComputeMoveStats {
  static final int METRICS =  // metric number goes here
  static final int MAXICONS = // icon count goes here
  public static void main(String[] args) {
    try {
      // code for connecting to Access database goes here

      // selection
      Statement s = con.createStatement();
      s.execute("SELECT * FROM logmovesrestricted");
      ResultSet rs = s.getResultSet();
      if (rs != null) {
        // Data Headings
        // output 1
        System.out.print( "logID," );
        // output 2
        System.out.print( "moveTypeMoves,remanipulations,totalIconDistance,actualIconDistance,otherTypeMoves," );
        System.out.print( "TOTtimeManipIcon,TOTtimeConsideringIcon," ); // from iconsD
        // output 3
        System.out.print( "avgMoveTypeMoves,avgRemanipulations," +"avgTotalIconDist,avgActualIconDist,avgOtherTypeMoves," );
        System.out.print( "AVGtimeManipIcon,AVGtimeConsideringIcon," ); // from iconsD
        // output 4
        System.out.println( "numIcons" );
        // aux output
        System.err.print( "logID,iconNum,movtypeMoves,remanipulations,totalIconDistance," +"actualIconDistance,otherTypeMoves," );
        System.err.println( "timeMovingIcon,timeConsideringIcon" );

      // trackers
      int logID = -1; int iconName = -1;
double[][] icons = new double[MAXICONS][METRICS]; // not most algorithmically efficient, but simple.
    // [icon #][0] = "move" moves
    // [icon #][1] = remanipulation
    // [icon #][2] = total distance manipulated
    // [icon #][3] = actual distance manipulated ** computed after all other stats
    // [icon #][4] = firstX
    // [icon #][5] = firstY
    // [icon #][6] = lastX
    // [icon #][7] = lastY
    // [icon #][8] = "other" moves

long[][] iconsD = new long[MAXICONS][3];
    // [icon #][0] = end time of last move
    // [icon #][1] = time moving icon
    // [icon #][2] = time considering icon (between contiguous moves)

while ( rs.next() ) {
    // db values can only be read once from left-to-right
    int dLOGID   = rs.getInt( "logID" );
    int dICONID  = rs.getInt( "iconID" );
    Timestamp dSTARTTIME = rs.getTimestamp( "startTime" );
    Timestamp dENDTIME = rs.getTimestamp( "endTime" );
    int dLOCATIONX = rs.getInt( "locationX" );
    int dLOCATIONY = rs.getInt( "locationY" );

    // check for new EXPERIMENT
    if ( logID != dLOGID ) {
        // summarize
        if ( logID != -1 ) {
            // OUTPUT 1.
            System.out.print( Integer.toString( logID ) + "," );

            int numIcons = 0;
            double[] TOTALS = new double[METRICS+2];
            for ( int i = 0 ; i < MAXICONS ; i++ ) {
                // only icons existing in this experiment
                if ( icons[i][8] != 0 ) {
                    // determine actual distance of movements for each icon

                    // print raw icon data to secondary stream
                    System.err.print( Integer.toString(logID) + "," + Integer.toString(i) );
                    for ( int j = 0 ; j < METRICS ; j++ ) {
                        // keep running total
                        TOTALS[j] += icons[i][j];
                    }
                }
            }
            for ( int j = 0 ; j < METRICS+2 ; j++ ) {
                TOTALS[j] /= numIcons;
                System.out.println( Integer.toString(logID) + "," + Integer.toString(TOTALS[j]) );
            }
        }
    }
}
// stats not needed
if ( j > 7 || j < 4 )
// continue printing raw icon data to 2nd stream
    System.err.print(""," +
        Double.toString(icons[i][j]));
} // rof(j)
TOTALS[METRICS] += (double) icons[i][1];
TOTALS[METRICS+1] += (double) icons[i][2];
System.err.println(""," + Long.toString(icons[i][1]) + "," +
    Long.toString(icons[i][2]));
numIcons++;
} // fi
} // for(i)

// OUTPUT 2:
// print totals -- number of icons, total moves, total remanipulations,
// total distance, etc..
for ( int i = 0 ; i < METRICS+2 ; i++ ) {
    // stats not needed
    if ( i > 7 || i < 4 )
        System.out.print( Double.toString( TOTALS[i] ) + "," );
} // rof(i)

// OUTPUT 3:
// print averages -- avg # moves per icon, avg # remanipulations per icon,
// avg icon distance, etc..
for ( int i = 0 ; i < METRICS+2 ; i++ ) {
    if ( i > 7 || i < 4 ) if ( numIcons != 0 ) {
        System.out.print( Double.toString ( TOTALS[i] / numIcons ) + "," );
    } else {
        System.out.print( "0," );
    }
} // rof(i)

// OUTPUT 4:
System.out.println( Integer.toString(numIcons) );
} // fi on logID

// reset
logID = dLOGID;
iconName = -1;
for ( int j = 0 ; j < MAXICONS ; j++ ) {
    iconsD[j][0] = 0;
    iconsD[j][1] = 0;
    iconsD[j][2] = 0;
    for ( int i = 0 ; i < METRICS ; i++ ) {
        icons[j][i] = 0.0;
    } // i
} // end for(j)
} // experiment fi

// delay in retrieval removes issue identified with driver
String dCOMMENT = rs.getString( "comment" );

// check for new ICON
if ( iconName != dICONID ) {
  // new icon
  if ( iconName != -1 ) {
    // not start of experiment
    iconsD[dICONID][2] += dSTARTTIME.getTime() - iconsD[iconName][0];
  }
  iconName = dICONID;
  iconsD[iconName][0] = dENDTIME.getTime();
  if( dCOMMENT.equals("Added") ) {
    // never seen before
    icons[iconName][4] = dLOCATIONX;
    icons[iconName][5] = dLOCATIONY;
    icons[iconName][6] = dLOCATIONX;
    icons[iconName][7] = dLOCATIONY;
  } // fi
  else {
    // seen before -- remanipulation
    icons[iconName][1] += 1.0;
  } // else
} // fi new icon
else {
  // not new icon
  // time between last move, ignoring first move/remanip
  iconsD[iconName][2] += dSTARTTIME.getTime() - iconsD[iconName][0];
}

// calculate times, irrespective of move type
iconsD[iconName][1] += dENDTIME.getTime() - dSTARTTIME.getTime(); // manip time

// keep track of time of last move
iconsD[iconName][0] = dENDTIME.getTime();

if ( dCOMMENT.equals("Moved") ) {
  // moved, so compute distance
  icons[iconName][0] += 1.0;
  icons[iconName][2] = Math.sqrt(
      Math.pow((double)dLOCATIONX - icons[iconName][6],2.0)
    + Math.pow((double)dLOCATIONY - icons[iconName][7],2.0)
  ); // Pythagorian thm
  icons[iconName][6] = dLOCATIONX;
  icons[iconName][7] = dLOCATIONY;
} // fi
else { // "added", "rotated", etc..
  icons[iconName][8] += 1.0;
} // fi
} // elihw
} // fi RS == null
s.close();
con.close();
}
catch (Exception e) {
    System.out.println("Error: " + e);
}
}
Appendix F – Sample Weka ARFF File

% 1. Title: Box 1 -- Statistics by Drawing
% 2. Source: Dissertation Project
@RELATION StatsByDrawingCognitiveEconScore
@ATTRIBUTE DrawingID NUMERIC
@ATTRIBUTE StoryID NUMERIC
@ATTRIBUTE OnTaskScore NUMERIC
@ATTRIBUTE LogicalScore NUMERIC
@ATTRIBUTE PrecisionScore NUMERIC
@ATTRIBUTE LikeASceneScore NUMERIC
@ATTRIBUTE KeyElementScore NUMERIC
@ATTRIBUTE MoveTypeMoves NUMERIC
@ATTRIBUTE Remanipulations NUMERIC
@ATTRIBUTE TotalIconDistance NUMERIC
@ATTRIBUTE ActualIconDistance NUMERIC
@ATTRIBUTE OtherTypeMoves NUMERIC
@ATTRIBUTE TOTtimeManipIcons NUMERIC
@ATTRIBUTE TOTtimeConsideringIcons NUMERIC
@ATTRIBUTE AVGMoveTypeMoves NUMERIC
@ATTRIBUTE AVGRemanipulations NUMERIC
@ATTRIBUTE AVGTotalIconDist NUMERIC
@ATTRIBUTE AVGActualIconDist NUMERIC
@ATTRIBUTE AVGOtherTypeMoves NUMERIC
@ATTRIBUTE AVGtimeManipIcons NUMERIC
@ATTRIBUTE AVGtimeConsideringIcons NUMERIC
@ATTRIBUTE NumIcons NUMERIC
@ATTRIBUTE DiffBtwnTotalandActualDistance NUMERIC
@ATTRIBUTE NumberOfManipulations NUMERIC
@ATTRIBUTE ExperimentTimeDRAW NUMERIC
@ATTRIBUTE ExperimentTimeQUES NUMERIC
@ATTRIBUTE ExperimentTimeBOTH NUMERIC
@ATTRIBUTE ContemplationTimeEXP NUMERIC
@ATTRIBUTE reflectionTimeEXP NUMERIC
@ATTRIBUTE TextScroll NUMERIC
@ATTRIBUTE MCCorrect NUMERIC
@ATTRIBUTE CognitiveEconScore {"low","high"} 

@DATA
...[data removed]...
References Cited


