ABSTRACT

USING MACHINE LEARNING TECHNIQUES FOR ANALYZING EDUCATIONAL DIALOGUES AND STUDENT RESPONDES

Douglas Krieghbaum
Department of Computer Science
Northern Illinois University, 2014
Reva Freedman, Ph.D., Director

Can sentence structure and complexity be used to identify authors in dialogues between students and tutors? Are there relationships between how an individual structures their sentences and their learning curve?

This thesis uses machine learning techniques and statistical analysis in two separate educational experiments. In the first experiment we attempt to find relationships between students' written essay responses to physics questions and their learning of the physics data. To find these relationships, we used multiple types of sentence data such as noun phrases, verb phrases, and other aspects of student writings.

In the second experiment we attempt to find the same relationships as in the above physics experiment, but also attempt to do author identification and to find the relationships (if any) between the teachers' linguistics and effectiveness.

Along with the aspects used in the physics experiment, we also used additional aspects like the Flesch Reading Ease test, and the percentage of domain words. The processes we used to find these features include the C4.5 decision tree

algorithm (WEKA’s implementation J48), the cluster algorithm KMeans (WEKA’s implementation SimpleKMeans), and a statistical method, Student’s t.

NORTHERN ILLINOIS UNIVERSITY

DEKALB, ILLINOIS

MAY 2014

USING MACHINE LEARNING TECHNIQUES FOR ANALYZING EDUCATIONAL DIALOGUES AND STUDENT RESPONSES

BY

DOUGLAS MATTHEW KRIEGHBAUM

A THESIS SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE

MASTER OF SCIENCE

DEPARTMENT OF COMPUTER SCIENCE

Thesis Director:
Reva Freedman, Ph.D.

ACKNOWLEDGEMENTS

“Had it not been for you, I should have remained what I was when we first met, a prejudiced, narrow-minded being, with contracted sympathies and false knowledge, wasting my life on obsolete trifles, and utterly insensible to the privilege of living in this wondrous age of change and progress.” (Benjamin Disraeli)

Since I began the journey that is this thesis, I have taken a much larger notice of acknowledgment pages. This is due to the fact that I have come to truly realize how much others impact a project, even in a relatively smaller development such as a Master’s thesis. The number of individuals and how they helped me through this process continued to grow throughout this thesis’s entirety. Without any of these individuals this thesis would not have been nearly as successful.

The most obvious and important is my family, especially my parents, without whom I could not have stayed sane during the many hours spent going over details. Their never ending support of my chosen path allowed me to grow as a student, researcher and most importantly as a person. I literally cannot say enough to thank them for their help and generosity.

I would also like to single out my brother-in-law and friend, John Hiltenbrand. All his tidbits of intuitive and in-depth information helped me to move around seemingly unmovable mental road blocks. His always helpful advice allowed me to

attempt new ideas or undertake existing ideas in new ways. In addition to this, his patient listening gave me a dummy to talk through any issues that arose and also to bounce off my many ideas.

I would like to thank Dr. Diane Litman for the use of the ITSPOKE physics data. I would also like to thank Dr. Allen Rovick and Dr. Joel Michael for the CIRCSIM-Tutor data. Without the use of either of these three Doctors' data, this thesis would not be possible.

I would also like to take this opportunity to acknowledge and thank my thesis committee members, Dr. Minmei Hou and Dr. Jie Zhou for their support, intuitive comments, and helpful criticisms. They not only helped to improve my thesis in content and accuracy, but also helped me to understand these concepts in more depth.

Finally, my thesis advisor, Dr. Reva Freedman, whom I cannot thank enough. She not only helped me in my decision for the thesis, she consistently offered pointers when the next direction was difficult to see. Her seemingly unending sage advice allowed me to go further into detail my understanding of the many concepts present within this thesis, which in turn allowed me to always go one step further in all processes. For all her hard work in proofing my ideas, coding, my presentation and the thesis itself. Without all the time she sacrificed, the hard work she added, and the wisdom she bestowed, this thesis would have been far less substantial then it is.

DEDICATION

To all those that come before me, family, friends, colleagues, and teachers,
without which this would not be possible.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

PREFACE

A man enters the office of a professional investigator with an electronic copy of a novel by an up and coming, but otherwise unknown, author. This man has a suspicion that the author is not an unknown amateur author, but is in fact a well-known and experienced wordsmith. The investigator that he is meeting to consult with has a system that may possibly help this man determine if his suspicions are correct.

In another scenario, a university director is hoping to limit the acceptance of prospective students to those that he knows will show an aptitude for learning so as to keep their learning statistics at an all-time high. This director has heard hypotheses that students who construct more complex sentences in their writings are also more likely to possess deeper learning skills.

In yet a third scenario it was conveyed to a concerned computer science professor that certain aspects of teaching improve education effectiveness. Some of these aspects were as simple as asking more questions or more complex as dominating her dialogue with domain words. She begins to wonder if she added some of these key aspects into her daily presentations, would her students gain more knowledge.

Down in the student chamber, a brave and intuitive masters student begins to speculate that the three previous scenarios could be combined into a single experiment. He considers the value of using machine learning to combine all these key pieces to answer these issues to assist in better educating his fellow students and his knowledgeable professors.   Can it be completed?

CHAPTER 1

INTRODUCTION

In this thesis we plan to show that the conspiracies in the preface are not just hypothetical scenarios, but are in fact problems that can be dissected, and possibly answered, with the help of machine learning techniques. We will show the use of machine learning algorithms to explore patterns in various linguistic features and then also demonstrate any relationships between these hidden patterns. To approach the many facets of these inquiries we have chosen to use multiple techniques and systems including the Stanford Parser, to break down all sentences to their basic parts, and the machine learning package WEKA, for its multiple machine learning algorithms. We used two separate experiments to attempt to answer multiple questions in regards to author identification, sentence complexity, educational effectiveness and learning with the help of machine learning techniques.

The two separate experiments that are the focus of this thesis use data from previously completed experiments from two different sources. The first experiment to be discussed utilizes data from physics essay problems answered by a large set of student volunteers engaging an intelligent tutoring system. This portion of the composition largely concentrates on the use of a decision tree algorithm and statistical analysis to search for relationships within the complexities of student writings and the educational determiners therein. The second experiment within this thesis uses data in the form of dialogues between tutors and students. The main focus of this section attempts to search for sentence complexity markers for use in attempting to identify the authors, or speakers. This portion also attempts to use this information to find relationships between the educational techniques in speech and their corresponding educational effectiveness. In both sections, machine learning algorithms are employed to help process the data from its raw form to its completed result.

CHAPTER 2

BACKGROUND

Theoretical Background

Psychologists believe that interactivity and deep learning are two of the features that make teaching effective (Chi, 2009; Graesser, McNamara, and VanLehn, 2005). Seeing as we cannot measure these features directly, we look at linguistic measures that are available. For example we can measure such linguistic characteristics as domain vocabulary usage and the number of questions asked in a session, as aspects of deep learning.

Much of an individual's learning style and understanding ability could be inherent in the way that individuals structure their sentences. Between both sets of data there are some questions that we attempt to answer. Some of these questions we endeavor to answer in both data sets. Some other questions will be geared only towards a specific data set. The following are a sample of the research questions that we will attempt to answer in this treatise.

Research Questions for Physics Data Set

Question 1) Is there a relationship between various linguistic features and physics knowledge?

Question 2) Since students wrote multiple versions of each essay response with tutoring in between, was there a significant difference in essay complexity between initial and final essays?

Question 3) Did essay locale (first vs. last essay) determine essay complexity?

Question 4) Does experiment type determine complexity?

Research Questions for Biology Data Set

Question 1) Can basic linguistic complexity be used for author identification?

Question 2) If so, which linguistic features contribute more strongly to author identification?

Question 3) Do more successful students . . .
use more domain words?
have longer sentences
have larger percentage/averages of SBARs?
have higher tree heights?
students utilize more words?
ask more questions?

Question 4) Do teachers that pose more questions elicit more understanding from their students?

Question 5) Is there a relationship between a teacher's linguistics and their teaching ability?

Question 6) Is there a measurable linguistic difference between tutors?

Question 7) If there is a measurable linguistic difference, which features are distinct?

Questions for both sets of data

Question 1) Does linguistic complexity determine learning?

Question 2) Do better students use more complicated data structures?

Question 3) Can machine learning techniques can help answer any of these questions?

NLP Background

In this thesis, the main scientific technique used under the broader umbrella of computer science is natural language processing, or NLP, and to continue with this paper, we must first establish the concept of natural language processing.

Natural language processing is the "computer understanding, analysis, manipulation, and/or generation of natural language. This can refer to anything from fairly simple string-manipulation tasks like stemming, or building concordances of natural language texts, to higher-level AI-like tasks like processing user queries in natural language" (reference.com, n.d.). More specifically it is "a field of computer science, artificial intelligence, and linguistics concerned with the interactions between computers and human (natural) languages. As such, NLP is related to the area of human–computer interaction. Many challenges in NLP involve natural language understanding, that is, enabling computers to derive meaning from human or natural language input, and others involve natural language generation" (wikipedia.org, n.d.).

In other words, as a wise mentor once described to me, “NLP is not the language structuring you learned in 4th grade.” [1]

The use of NLP is what gives us our ability to dissect the sentences into detailed usable structures. An example of a typical sentence processed in a NLP format is shown below with the sentence “The intelligent developer wrote the code.”

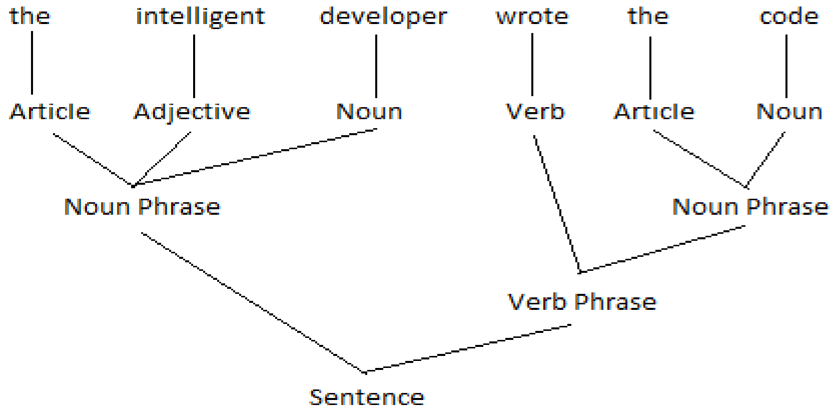


Figure 1: NLP Sentence Structure Example: "The intelligent developer wrote the code”

As you can see in the inverted tree in Figure 1, a normal sentence can be broken down to its most basic features within NLP. Though not restricted to language parsing (computer programming languages are broken down similarly), NLP is very helpful in finding these individual key components in the English language.

[1] As heard in conversation with Dr. Reva Freedman

A normal sentence is broken into multiple sub-phrases, for instance noun phrases and verb phrases. A noun phrase is a section of a sentence formed by a noun (e.g. noun or pronoun) and all of its articles (determiners such as ‘the’) and modifiers (such as adjectives). One such example (as in the Figure 1 above) is the noun phrase “the intelligent developer” which, in this case, has an article (or determiner) ‘the’, an adjective ‘intelligent’, and a noun ‘developer’.

Correspondingly, a verb phrase is a section of a sentence that is formed by a verb and all of its modifiers and auxiliaries, which are not part of the subject. The verb phrase in this example has a verb ‘wrote’ and another noun phrase as its children. The key point in a verb phrase is that modifiers and auxiliaries are only included in the verb phrase if it is not already a main part of a noun phrase.

Verb phrases and noun phrases are normal in every complete sentence, but to assist in measuring complexity another phrase structure should be explained. Subordinate clauses, or SBARs as used in the parser, are dependent clauses that require more information for the reader to complete the idea. Subordinate clauses usually begin with a subordinating word and include a relative pronoun. Tables 1 and 2 lists examples of subordinating elements and relative pronouns. Since subordinate clauses are more complex than normal sentences, these are good ways in helping to measure sentence complexity.

Table 1: Subordinating Word Examples

Although	Because	Even though
Since	That	Though
Until	Whether	While

Table 2: Relative Pronouns Examples

That	What	Which
Whichever	Who	Whoever
Whom	Whomever	Whose

The sub-phrases can have multiple child phrases, including nested children (e.g. noun phrase within a noun phrase). The phrasing is not limited to just noun phrases, verb phrases, and subordinate clauses, other such phrases are adjective phrases, adverb phrases, questions (SQs), prepositional phrase, conjunction phrases, fragments and others. All levels in the sentence tree though always terminate to a part of speech such as a sentence terminator (e.g. a period or a question mark).

Stanford Parser

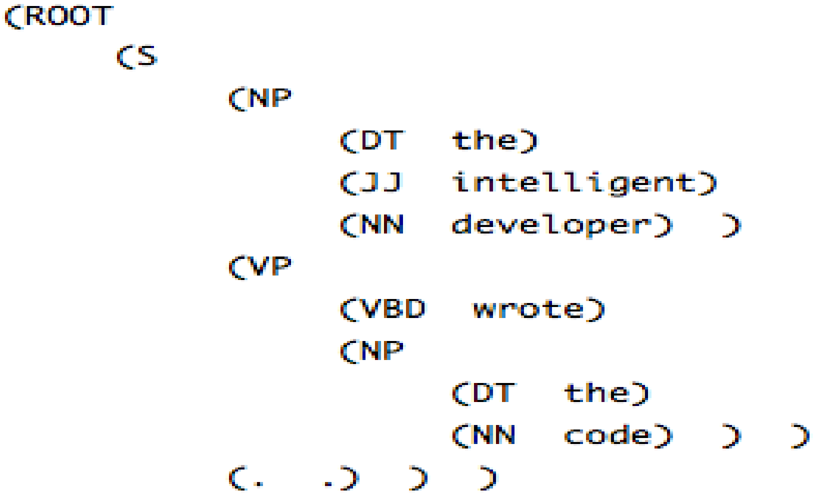
Among the natural language processing implementations available is the very powerful natural language parser, the Stanford Parser. The Stanford Parser has been developed by the Stanford Natural Language Processing Group at Stanford University and is one of the most accurate and most recognized natural language parser in the linguistics community. The Stanford Parse is a program that parses natural language into grammatical structures of sentences. In other words, it attempts to break down sentences into their basic parts, from the sentence phrasing to the word structure or parts of speech of each word. The Stanford Parser is a Java-based program that is available publically and has been proven to be relatively reliable in parsing sentences.

This natural language processor is the program we have used for this thesis to compute the sentence complexities in these experiments. As with the human NLP, one of the functions of the Stanford Parser is the ability to parse a sentence into a tree structure. The Stanford Parser also tags each word in the sentence with its part of speech. For their tagging the Stanford Natural Language Processing Group uses the Penn Treebank Project's tags for each word (Santorini, 1995). Table 3 below shows their word tags.

Table 3: Stanford Parser's Parts of Speech Tag List

Tag	Tag Description
CC	Coordinating conjunction
CD	Cardinal number
DT	Determiner
EX	Existential there
FW	Foreign word
IN	Preposition or subordinating conjunction
JJ	Adjective
JJR	Adjective, comparative
JJS	Adjective, superlative
LS	List item marker
MD	Modal
NN	Noun, singular or mass
NNS	Noun, plural
NNP	Proper noun, singular
NNPS	Proper noun, plural
PDT	Predeterminer
POS	Possessive ending
PRP	Personal pronoun
PRP$	Possessive pronoun
RB	Adverb
RBR	Adverb, comparative
RBS	Adverb, superlative
RP	Particle
SYM	Symbol
TO	to
UH	Interjection
VB	Verb, base form
VBD	Verb, past tense
VBG	Verb, gerund or present participle
VBN	Verb, past participle
VBP	Verb, non-3rd person singular present
VBZ	Verb, 3rd person singular present
WDT	Wh-determiner
WP	Wh-pronoun
WP$	Possessive wh-pronoun
WRB	Wh-adverb

Using the same sentence as the basic Natural Language Processing (NLP) example from Figure 1. Figure 2 shows the output of the sentence from the Stanford Parser’s probabilistic lexicalized parser. Some of the sentence complexity levels, or sentence phrases, given by the Stanford Parser are: Root (the entire sentence), S (Simple Sentence), NP (Noun Phrase), VP (Verb Phrase), ADJP (Adjective Phrase), ADVP (Adverb Phrase), PP (Prepositional Phrase), FRAG (Fragmented Sentence), and CONJ (Conjunction Phrase).



```
(ROOT
     (S
          (NP
               (DT   the)
               (JJ   intelligent)
               (NN   developer)   )
          (VP
               (VBD   wrote)
               (NP
                    (DT   the)
                    (NN   code)   )   )
          (.   .)   )   )
```

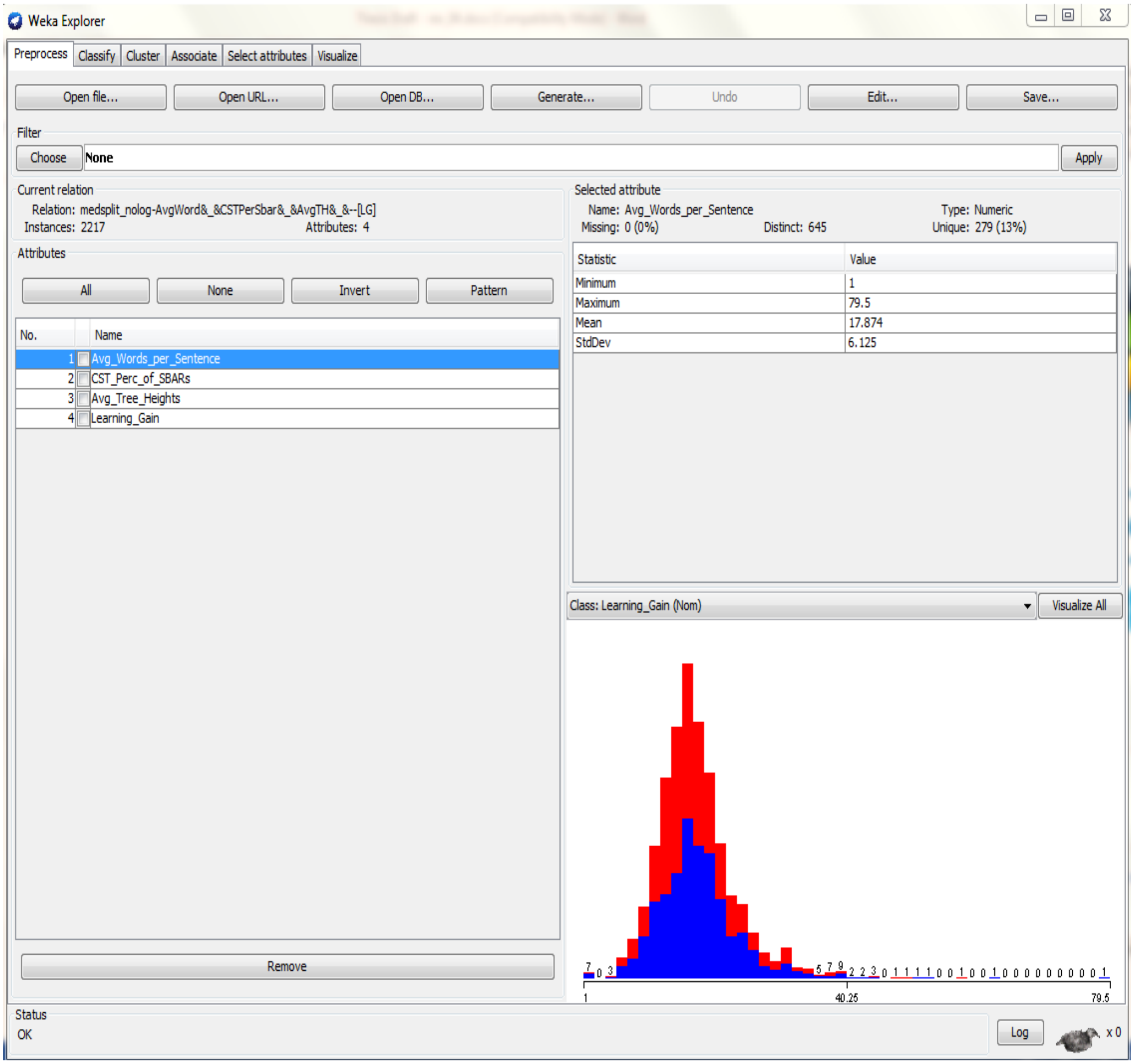
Figure 2: Stanford Parser Parsed Example:
"The intelligent developer wrote the code"

Comparing Figure 1 and Figure 2, we can see that the Stanford Parser parses the example sentence in a similar but functionally equivalent way. The main difference is the parts-of-speech-tag variation and the structure of the printout, though still in a tree form. Even though the Stanford Parser will parse the parts of speech to specific types, such as multiple types of noun like plural nouns or mass nouns, in this paper we grouped all like types together. For example there are six types of verbs: base, past tense, present participle, past participle, non-third person singular, and third-person present. For our experiments we grouped all of these together as the 'verb' group. The same process was used for all similar part of speech groups.

WEKA

To more readily use machine learning algorithms, we will be using a data mining package called Waikato Environment for Knowledge Analysis (Hall et al., 2009). In particular we will be using WEKA 3: Data Mining Software in Java. WEKA contains a collection of many data mining algorithms for various types of data-mining tasks. WEKA “is a comprehensive tool bench for machine learning and data mining. Its main strengths lie in the classification area, where all current [machine learning] approaches — and quite a few older ones — have been implemented within a clean, object-oriented Java class hierarchy. Regression, Association Rules, and clustering algorithms have also been implemented.” (Bouckaert et al., 2013) Some of the specific algorithms that WEKA implements are J48 (C4.5), K-Means, Bayesian Logistic Regression, KStar and many others. The ones we will be concentrating on in this thesis are J48 and K-Means.

WEKA offers both a command line access and a graphical user interface to their collection. Though in this thesis we used mainly the command line interface, the GUI interface provides various graphing options and clarifies formatted textual output. Figures 3 and 4 show the output of one of the runs of WEKA’s implementation of C4.5 (which WEKA has named J48) on our physics data set.



Filter

None

Current relation

Relation: medsplit_nolog-AvgWord&_&CSTPerSbar&_&AvgTH&_&--[LG]

Instances: 2217 Attributes: 4

Attributes

No.		Name
1	□	Avg_Words_per_Sentence
2	□	CST_Perc_of_SBARs
3	□	Avg_Tree_Heights
4	□	Learning_Gain

Selected attribute

Name: Avg_Words_per_Sentence Type: Numeric

Missing: 0 (0%) Distinct: 645 Unique: 279 (13%)

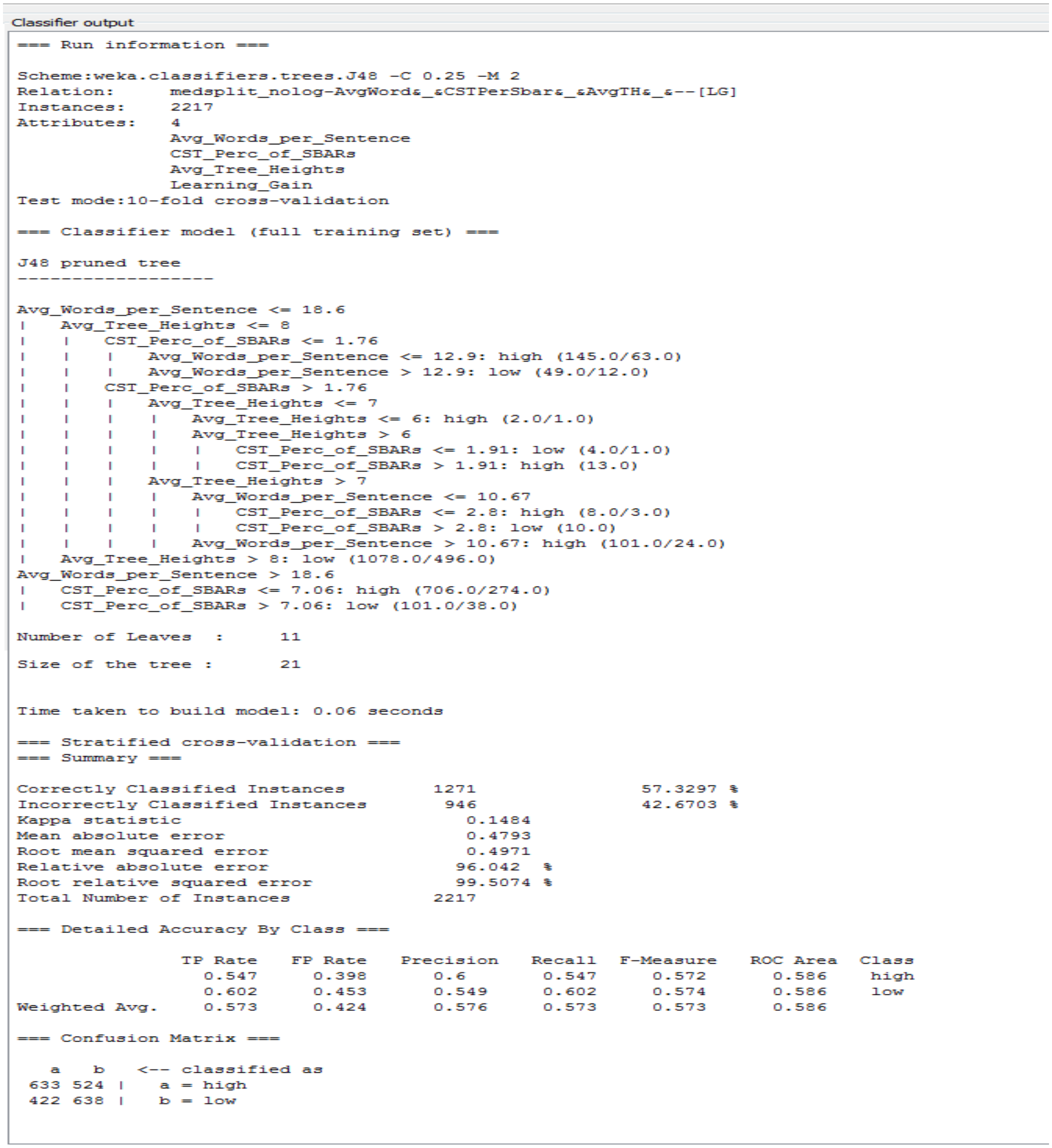
Statistic	Value
Minimum	1
Maximum	79.5
Mean	17.874
StdDev	6.125

Class: Learning_Gain (Nom)

Status

OK

Figure 3: WEKA’s Pre-Processed GUI Example



Classifier output

```
=== Run information ===

Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     medsplit_nolog-AvgWords_sCSTPerSbars_sAvgTHs_s--[LG]
Instances:    2217
Attributes:   4
              Avg_Words_per_Sentence
              CST_Perc_of_SBARs
              Avg_Tree_Heights
              Learning_Gain
Test mode:10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

Avg_Words_per_Sentence <= 18.6
|   Avg_Tree_Heights <= 8
|   |   CST_Perc_of_SBARs <= 1.76
|   |   |   Avg_Words_per_Sentence <= 12.9: high (145.0/63.0)
|   |   |   Avg_Words_per_Sentence > 12.9: low (49.0/12.0)
|   |   CST_Perc_of_SBARs > 1.76
|   |   |   Avg_Tree_Heights <= 7
|   |   |   |   Avg_Tree_Heights <= 6: high (2.0/1.0)
|   |   |   |   Avg_Tree_Heights > 6
|   |   |   |   |   CST_Perc_of_SBARs <= 1.91: low (4.0/1.0)
|   |   |   |   |   CST_Perc_of_SBARs > 1.91: high (13.0)
|   |   |   Avg_Tree_Heights > 7
|   |   |   |   Avg_Words_per_Sentence <= 10.67
|   |   |   |   |   CST_Perc_of_SBARs <= 2.8: high (8.0/3.0)
|   |   |   |   |   CST_Perc_of_SBARs > 2.8: low (10.0)
|   |   |   |   Avg_Words_per_Sentence > 10.67: high (101.0/24.0)
|   Avg_Tree_Heights > 8: low (1078.0/496.0)
Avg_Words_per_Sentence > 18.6
|   CST_Perc_of_SBARs <= 7.06: high (706.0/274.0)
|   CST_Perc_of_SBARs > 7.06: low (101.0/38.0)

Number of Leaves  :     11

Size of the tree :      21


Time taken to build model: 0.06 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        1271               57.3297 %
Incorrectly Classified Instances       946               42.6703 %
Kappa statistic                          0.1484
Mean absolute error                      0.4793
Root mean squared error                  0.4971
Relative absolute error                 96.042  %
Root relative squared error             99.5074 %
Total Number of Instances             2217

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.547     0.398      0.6       0.547     0.572      0.586    high
                 0.602     0.453      0.549     0.602     0.574      0.586    low
Weighted Avg.    0.573     0.424      0.576     0.573     0.573      0.586

=== Confusion Matrix ===

   a   b   <-- classified as
 633 524 |   a = high
 422 638 |   b = low
```

Figure 4: WEKA's J48 Example: Tutor Identification

J48 (C4.5) Algorithm

One of the machine learning algorithms employed in this study is the C4.5 algorithm as implemented in WEKA as J48. C4.5 is a statistical classifier developed by Ross Quinlan (1992) as a decision tree algorithm. It is actually an extension of his earlier algorithm ID3. Before a description of these algorithms can be discussed, a cursory explanation of decision trees should be given.

First used in 1964, decision trees are a potent tool for classifying and predicting data points from different sets of records. The art of a decision tree can be classified as a decision making process where a node is represented by a Boolean test of a category. For each node a decision is made as to the next branch to move down, until it reaches a terminal node, indicating the final decision/prediction. Figure 5 shows a graphical representation of a simple decision tree.

The C4.5/ID3/J48 algorithms allow a set of data with multiple categories, all of the same structure, to be used to determine a decision tree and predict the outcome from this decision tree. These categories are attribute/value pairs with data to be used to attempt to correctly calculate the value of a final attribute/value pair. This final category is normally a limited value set such as {'rainy', 'cloudy', 'sunny'} or {'true', 'false'}. This last thought is one of the major benefits of using J48. Where other similar decision tree algorithms restrict branches to only a true/false type decision, J48 does not have this limit. J48 allows the use of a multitude of decision options.   Figure 5 shows a rendition of a decision tree for deciding whether or not to play baseball on a particular day depending on different weather characteristics.

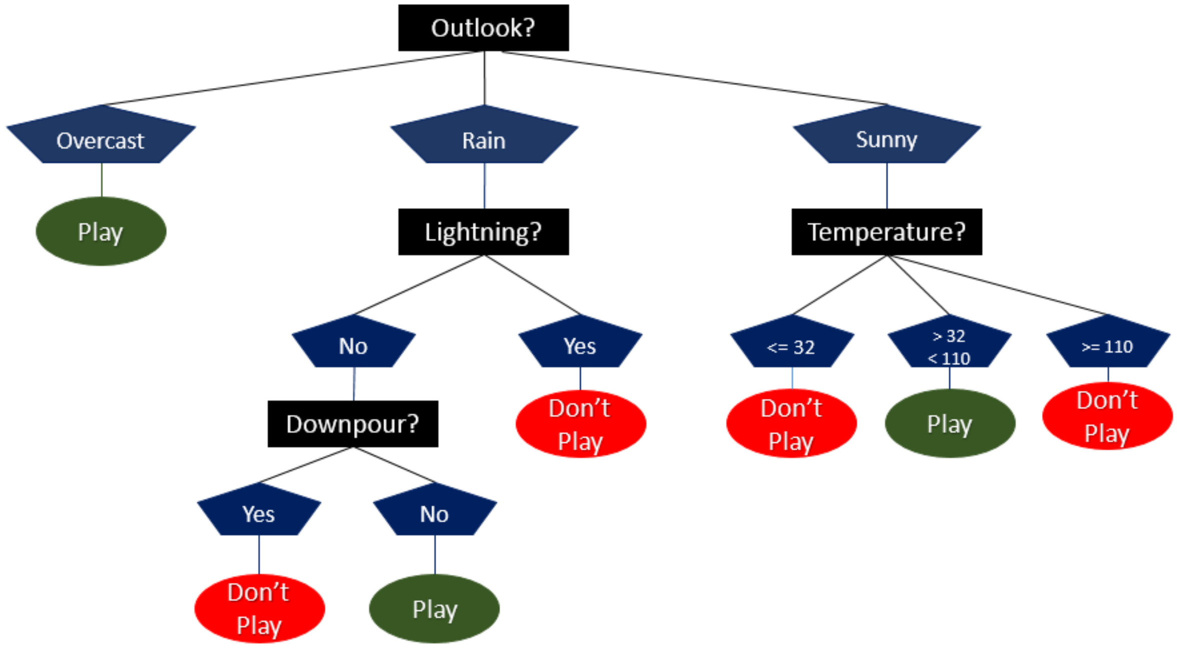


Figure 5: Decision Tree: Whether or Not to Play Baseball

In Figure 5 there is a decision tree that has a height of four levels. That is, at its deepest point, from the root of 'outlook' through the nodes, 'rain', 'lightning', 'overcast', and 'downpour' children, there are four levels. To use this tree we start at the top and answer the question "what is the outlook?" If it is overcast, the decision is over and we play. If the outlook answer instead is sunny, we check the temperature question. If the temperature is within a safe range we play, otherwise we don't play. If however it is raining, we then check if it is lightning. If there is lightning we terminate and don't play. If there is no lightning we trek further down the tree to see if there is a downpour and decide to play based on that node's result.

Referring back to Figure 4 and the WEKA J48 output example we can examine in detail the data it provides. The first thing WEKA's implementation of C4.5 (which they call J48) shows after their headers is the pruned decision tree that it creates. Based on the previous decision tree explanation, one can see that this tree has six levels and J48 is using all three of the input fields. It even uses one of them in a nested form (CST_Perc_of_SBars is a child of Avg_Tree_Heights, which is a child of Avg_Tree_Heights, which is a child of CST_Perc_of_SBars). Following the tree is the information we are most interested in, the summary of the results with accuracy ratings. In this example the prediction accuracy of the J48 algorithm was 57%. J48 gives more than just an accuracy for the correctly classified instances in its reports of the computations. The other statistics that J48 reports are Kappa Statistics, Mean Absolute Error, Root Mean Squared Error, Relative Absolute Error, and Root Relative Square Error. These are defined in the following paragraph and Table 4.

Kappa Statistics is a measure of the degree to which two judges agree in their respective sorting of specified items into distinct categories. Mean Absolute Error on the other hand is the weighted average used to measure how close the predictions are to their actual conclusions. Root Mean Squared Error is a method of measuring the average magnitude of an error. Relative Absolute Error is a technique of measuring how good the measurement is in relation to the object measurement. Finally Root Relative Squared Error is "relative to what it would have been if a simple predictor had been used. More specifically, this simple predictor is just the average of the actual values. Thus, the relative squared error takes the total squared error and normalizes it by dividing by the total squared error of the simple predictor. By taking the square root of the relative squared error one reduces the error to the same dimensions as the quantity being predicted." (gepsoft.com, n.d.)

Table 4: Statistic Definitions

Statistic	Definition
Kappa Statistics	Kappa Statistics is a measure to the degree in which two judges agree in their respective sorting of the items into the distinct categories.
Mean Absolute Error	Mean Absolute Error is the weighted average used to measure how close the predictions are to their actual conclusions.
Root Mean Squared Error	is a method of measuring the average magnitude of an error.
Relative Absolute Error	Relative Absolute Error is a technique of measuring how good the measurement is in relation to the object measurement.
Root Relative Error	Root Relative Squared Error is "relative to what it would have been if a simple predictor had been used. More specifically, this simple predictor is just the average of the actual values. Thus, the relative squared error takes the total squared error and normalizes it by dividing by the total squared error of the simple predictor. By taking the square root of the relative squared error one reduces the error to the same dimensions as the quantity being predicted." (gepsoft.com, n.d.)

Aside from the previous five statistical measures that WEKA reports, there are several more details that are reported by J48. This other pertinent information follows the error statistics in the “Detailed Accuracy by Class” and are defined in the Table 5 with an example in Table 6.

Table 5: Accuracy Definitions

Information	**Definition**
TP Rate	True Positive Rate, the rate at which the instances were correctly categorized as a positive.
FP Rate	False Positive Rate, the rate at which the instances were incorrectly categorized as a positive.
TN Rate	True Negative Rate, the rate at which the instances were correctly categorized as a negative.
FN Rate	False Negative Rate, the rate at which the instances were incorrectly categorized as a negative.
Precision	Also referred to as positive predictive value, the percentage, or fraction, of the relevant instances correct.
Recall	Also referred to as sensitivity, the percentage, or fraction, of positive predictions caught.
F-Measure	This measure combines both precision and recall with a harmonic mean (complement of the arithmetic mean).
ROC Area	Relative Operating Characteristic, a more complex measure of the degree of discrimination.

Table 6: Information Retrieval Example

If there were 1000 cases to predict a positive case or negative case, the following shows the previous definitions in action. TN = True Negative; FN = False Negative

	Actual Positive Case	Actual Negative Case
Predicted Positive Case	TP Rate: 400	FP Rate: 100
Predicted Negative Case	FN Rate: 200	TN Rate: 300

Recall = Percent of positives caught = $\frac{True\ Positives(TP)}{True\ Positives+False\ Negatives\ (FN)} = \frac{400}{400+200}$ = 66%

Precision = Percent of positive predictions correct = $\frac{True\ Postitives\ (TP)}{True\ Positives+False\ Positives\ (FP)} = \frac{400}{400+100}$ = 80%

All of this information gives a summary of how the algorithm did in terms of accuracy in guessing the correct outcome. In the previous example the three input features could only guess the single output element slightly more accurately than a coin flip, specifically 50% for a coin flip versus 57% for this example.

Clustering (K-Means)

Decision tree algorithms are very useful in attempting to predict specific outcomes, but as an independent option we need another type of machine learning technique to show relationships in a different manner. One such option is the machine learning technique of clustering. Clustering is a technique that divides the categories, not into a branch and node set, but into similar groupings (Steinbach, Ertoz, & Kumar, 2004). The idea behind clustering is to see if there are some underlying related qualities that cause the data to be grouped together (Witten et al., 2011).

The clustering algorithm that we used is the K-Means implementation in WEKA, named Simple-K-Means. At the outset of K-Means, a specified, usually random, number of central points are used as initial focal points for the clusters. Each data item is then given a value representing its distance from the closest focal point. The next step in the K-Means algorithm is to calculate the mean for all the current cluster groups. These new values, gathered by the means, from which the algorithm gets its name, are the new focal points for each individual cluster group. The process of assigning the data to each group is iterated again with these new focal points. When the same data are assigned to the same groups in consecutive iterations, then the process is complete and the clustered groups are in their final position. This process gives a great way of showing relationships without having to assign a specific output to a prediction. Figure 6 shows an example of clustering. The large black circle represents the scope of the data, and the inner circles represent the cluster group of each of the shapes.

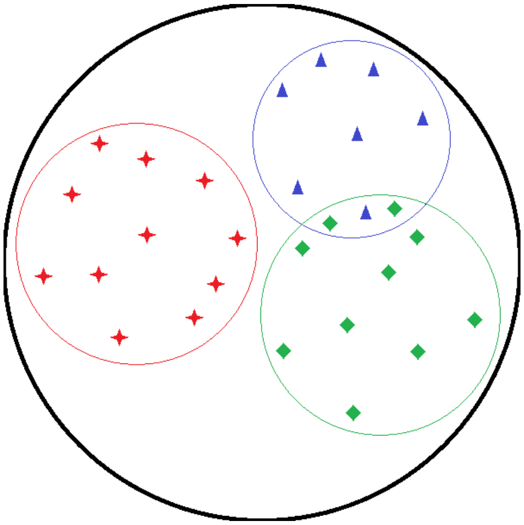


Figure 6: Visual Cluster Example

Statistics and the T-Test

While the machine learning algorithms J48 and K-Means are very useful in showing specific accuracy or a significant cluster of the data, conventional statistical methods are still an effective way of showing some significant comparisons. The two-tailed paired t-test is excellent for showing some type of variance between two sets of data, without caring which way the difference leans. One example of using a two-tailed paired t-test is in the case of pharmaceutical studies where a group is comparing a blue pill to a red pill with the same group of subjects. A two-tailed paired t-test shows only that there was or was not a significant difference between the two pills. Specifically, it is useful in showing that there is a significant difference in either direction of a set of data. On the other hand, the two-tailed unpaired t-test, while still only concerned about variances in either direction, is used with separate sets of data.

In contrast to the two-tailed t-test, a one-tailed t-test only shows a significant difference in the specified direction one proposes. In a one-tailed t-test, if the direction of the statistics is in the opposite direction than hypothesized, the result would not be significant. Or more simply, one-tail answers the question, “is one better than two”, and two-tail answers the question, “are they different”. The benefit to utilizing the t-Test formula is that it can use discrete values as input but it computes an output in a continuous form (e.g. a float).

We used the two-tailed paired t-test in the physics experiments to show the differences in the students’ data between their initial attempts and their final attempts. For the biology experiments we used the two-tailed unpaired t-test because the number of sessions each tutor participated in was not the same. Dr. Michael tutored a total of 23 discussions while Dr. Rovick only presided in a total of 17.

Related Work

There have been multiple research attempts that utilize many of the same ideas that this paper explores. None of these associated attempts are implemented, to our knowledge, in the way in which we have approached it, nor with the types of techniques we are attempting. There has been much related research in the fields of author identification, effective teaching practices, and machine learning.

Stylometrics, or the study of linguistics style, is one of the main techniques used for author identification and has a long history dating back to 1439 and the proof that the Donation of Constantine was a forgery. There are several author identification programs and methodologies in use today. The earliest case of modern techniques of author identification in use was to help determine the authors of specific Federalist papers (Mosteller & Wallace, 1984). Mosteller and Wallace used the frequency of words to determine the authors. For example, one author of the Federalist Papers used “whilst”, while the other used the word “while” for the same context. Another well-known author identification project was searching for the author *of Primary Colors* (Liptak, 2000). For this project there were three focuses of attributes. The first and primary, as with Foster and Wallace, was vocabulary; the second was punctuation, and the third was the unquantifiable attribute of “points of anxiety”. The points of anxiety refer to the issues that the author was concerned with.

The most extensive author identification package so far is JGAAP, built by Patrick Juola, which is a Java based system that allows the user to choose among multiple algorithms (JGAAP, n.d.). As Juola points out though: “. . . methods for authorship attribution are neither reliable . . . nor well-understood” (Juola, 2006). These systems, and other similar systems, only concentrate on evaluating writing styles and are used specifically for author identification. These systems are not designed with education in mind. They also do not use the results for purposes other than just authenticating authors. Another difference that these previous systems have is that they also rely heavily on words, such as function words, while this paper relies heavily on lexical tags of sentences and types of words, not the words themselves.

More recently, Patrick Juola has initiated some of the most recent and intensified research in this area. In a more recent paper, he tests multiple author identification software against a common set of corpora. The best of which, in a series of different types of known authorship questions, had a precision and recall of 0.753 (Juola, 2013). In Juola's own author identification analysis program much of its process involves inspecting the authors' individual words (Juola, 2004; Zhao, 2007). His program examines the most common bigrams and/or n-grams, the top 100 used words, frequently appearing clusters of words, and the distribution of word lengths (Sostek, 2013). Even Juola's system, thought by some to be in the forefront of the field, does not guarantee an identification, but only a highly educated probability.

Educational effectiveness is a technique that most successful educators strive to improve in their daily teaching routines. Chi quotes Benjamin Franklin as saying "Tell me and I forget. Teach me and I remember. Involve me and I learn." One of the proposed inquiries into deep learning can be referred to as the interaction between teacher and student, specifically the amount of 'action', or participation of the student. The participation of the student is a valid way of measuring the amount the student is involved. One of the interactive characteristics that Chi provides in her paper is the proposed idea of "dialoging substantively on the same topic, and not ignoring a partner's contributions" (Chi, 2009). In general an effective way of educating is forming a dialogue with a student and involving them in their own learning. Quantifying effectiveness and interactivity are two points both Chi and this thesis attempt to address.

The Support Vector Machine is a machine learning technology that is very similar to the experiments conducted in this research. Using the support vector machine mechanism, Bullington, Endres and Rahman have experimented with classifying open-ended questions (2007). To train and test their hypotheses, they cleaned their data set by removing punctuation, removing ‘stop words’ (e.g. a, and, the, etc.), and doing word stemming, which is the process of reducing a word to its root form (e.g. slyly to sly). In the process of training the SVM for open-ended questions, the authors were hoping to train the system to look for keywords and phrases for the system to use in its assignment of question type. This research reiterates the importance of questions in gathering information about individuals.

Earlier researchers have used machine learning to analyze the biology corpus that we used (Freedman et al., 2001; Freedman et al., 1998; Kim et al., 2006; Kim et al., 2000), but their work involved higher level discourse phenomena, as opposed to ours, which used word, phrase, and sentence level phenomena. Similarly, other researchers have studied the physics corpus. For example, Lipshultz et al. (2011) used machine learning to study high level discourse phenomena, while Rose et al. (2003) and Ai and Litman (2006) used statistical methods.

CHAPTER 3

PHYSICS

Physics Background

This chapter of the thesis describes a continuing attempt to identify relationships between the linguistic complexities of students' writing regarding answers to physics word problems and their ability to solve said problems. It will attempt to dissect and use many different written complexity markers and how the markers relate to individual students' understanding of the problems being answered.

In this paper, complexity is measured using simple linguistic elements and the more complex measures calculated from them. Simple linguistic measures include the percent of nouns, verbs, adjectives and other parts of speech in the discourse compared to the total number of words used. Measures of complexity include average sentence length, average tree height, percent of subordinate clauses and percent of certain types of verb phrases. The metrics for measuring the students' understanding of the problems are the use of pretest and posttest scores and normalized learning gain. The normalized learning gain used in this process is defined as:

$$\frac{posttest - pretest}{1 - pretest}$$

This study uses a set of 2217 files consisting of answers from the physics questions of the 91 recorded students involved in the study. In addition to these files the pretest and posttest scores for these students are included. We used the Stanford Parser to parse the files according to their parts of speech and sentence structure. We then used the C4.5 algorithm, as implemented in WEKA as J48, to test our hypotheses. We follow these results with statistics by using a two-tailed t-test to find any statistical differences between data.

Data Collection

The data used in this study were originally collected for testing ITSPOKE, a spoken dialogue intelligent tutoring system (ITS) that uses the facilities of the text-based Why2-Atlas physics ITS. In the ITSPOKE system, a student is given a qualitative problem in elementary college physics. The student responds with an essay answer, then is coached using tutorial dialogue to improve the answer until it is judged acceptable. In general, students revised their essays by adding a missing concept or revising an incorrect one. Figure 7 shows a sample problem.

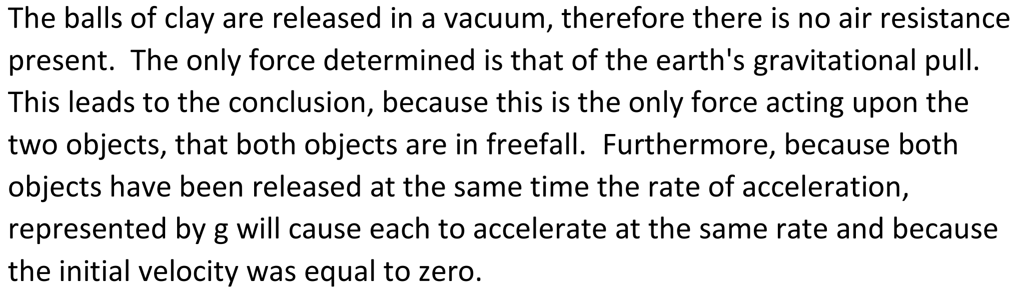
The data included essays from three experiments. In each experiment, students who had never studied college physics worked through approximately five problems each with ITSPOKE, with a pretest before the first problem and a posttest after the last. The first experiment, conducted in 2002-2003, used human tutors. The second experiment, conducted in Spring 2003, used a synthesized voice. The third experiment, conducted in Fall 2005, had two branches, one with a synthesized voice and one using text built from prerecorded voice snippets. Students completed a pretest measuring their knowledge of physics before the first problem and a posttest after the last problem.

For this study we only used students who completed the entire experiment, including the posttest. A few students were dropped for technical reasons, e.g., because the voice quality of the recordings was not sufficiently good. There were 91 students who did a total of 495 problems. (There were 11 different problems.) The students wrote a total of 2217 essays, or about 4.5 essays per problem. There were a total of 14524 sentences, or about 6.5 sentences per essay.

Figure 7 shows a sample problem, and Figure 8 shows an excerpt from a student essay written in response to this problem.

> Suppose that you released 3 identical balls of clay in a vacuum at exactly the same instant. Now you stick two of the balls together, forming one ball that is twice as heavy as the remaining, untouched clay ball. Both balls are released in a vacuum at exactly the same instant. Which ball hits the ground first?

Figure 7: Sample ITSPOKE: Physics Problem



The balls of clay are released in a vacuum, therefore there is no air resistance present. The only force determined is that of the earth's gravitational pull. This leads to the conclusion, because this is the only force acting upon the two objects, that both objects are in freefall. Furthermore, because both objects have been released at the same time the rate of acceleration, represented by g will cause each to accelerate at the same rate and because the initial velocity was equal to zero.

Figure 8: Excerpt from Student Essay [from Essay 105-3-3]

Data Preparation

To reduce the frequency of erroneous parses, we engaged in several forms of data cleaning. We deleted extraneous punctuation and unprintable characters. In some cases, the Stanford parser handled single letters at the end of a sentence incorrectly, for example, considering ‘a.’ at the end of a sentence as an abbreviation alone and not sentence-ending punctuation as well. As a result, sentences ending in an equation such as $f = m * a$ or $m = f / a$ would be concatenated to the following sentence. For this reason, we doubled single-letter variables in an equation, for example, a became aa.

Next, contractions were de-abbreviated. For example, the Stanford parser treats the word “don’t” as two words, “do” and “n’t”. Since words like this would increase the “do” count, but not the “not” count, this would in essence give “not” a count of about half its actual value. To fix this issue, all instances of “n’t” were changed to “not” and equivalently for other contractions.

Finally, we spell-corrected the corpus. Spelling correction reduced the unique word count from the 2217 essays (247192 words) from about 2000 words to 1471. In one extreme case, Table 7 contains 27 alternate spellings for acceleration, totaling 130 instances.

Table 7: Alternate Spellings for 'Acceleration'

accceleration	accelaration	accelaration	accelaraton
accelation	acceleartion	acceleceration	acceleraction
acceleraion	acceleratino	accelerationg	acceleratoin
acceleraton	accelercation	acceletation	accelleration
accelration	accelreation	acceration	accerlation
accerleration	accleration	acclerationn	acelaration
acieration	excellerarion	excelleration	

Feature Identification

We started by identifying 17 basic features divided into nine categories.

1. Experiment type

The data were collected from three experiments, comprising four cases. As described above, the first had a human tutor and the second had a synthesized voice. The third experiment had two arms, one with a synthesized voice and one that built responses from prerecorded snippets of human voices. Experiment type is a feature that could lead to differentiation in learning (Ai & Litman, 2006).

2. Essay locale

Students wrote between one and 16 essays per problem. Because of this large variation, it would be misleading to look at the absolute essay number. Instead, we labeled essays as the student’s first, middle or last attempt. If there was only one essay, we arbitrarily labeled it as an initial essay.

3. Average POS counts per essay – Noun, Adjective, Adverb, Preposition

We counted several categories of basic parts of speech, including nouns, verbs, adverbs and prepositions. Since some essays were longer than others, all of our counts were normalized by dividing by the number of words in the essay.

4. Other constituent counts – noun phrase(NP), adjective phrase(AdjP), adverb phrase (AdvP), prepositional phrase (PP)

We also counted the number of noun phrases, adjective phrases, adverb phrases, and prepositional phrases.

5. Average words per sentence

We used several measures of writing complexity. As in the Flesch readability formula (wikipedia.org, n.d.), we used the number of words per sentence as a simple measure of writing complexity. This number was calculated at the essay level, i.e., the total number of words in the essay divided by the number of sentences.

6. Average sentence tree height

Since the height of the parse tree is a rough measure of the amount of subordination in a sentence, we used the average height of the parse trees in a student essay as a second measure of essay complexity.

7. Average subordinate clauses per essay

As an additional measure of complexity, we used the average number of SBARs per essay, which modeled the number of subordinate clauses used by the student. The Stanford parser generates an SBAR whenever a subordinating conjunction such as “that” is used. This number was divided by the number of words in the essay.

8. Average “non-consecutive” VPs per essay

Nested verb phrases also increase complexity. However, the Stanford parser adds an additional VP node for each auxiliary verb, so that a form like “the ball *will* have hit the ground” contains three VPs. Since that seemed to overweight the difficulty level of generating verb forms with multiple auxiliaries, we only counted VPs whose parent was not a VP. Again, this number was divided by the number of words in the essay.

9. Student educational data – pretest, posttest, learning gain

Pretest and posttest scores were available at the student level, i.e., the student took the pretest before their first problem and the posttest after their last. The pretest and posttest scores are expressed as the percent of correct answers. Normalized learning gain was defined in the conventional manner as the student's improvement with respect to questions missed on the pretest, i.e., (posttest - pretest)/(1 - pretest). The normalized learning gain has a value between -1 and 1.

Compound Features

With the exception of experiment type, all of the basic 17 features are numeric. In addition to treating each feature as a number, for each input feature *n* we also created a variant $\log_2 n$ to reduce the influence of large values, such as extremely long sentences. For each of the two versions, we also created a discrete version in which we assigned the data to 10 equal-width bins. In this thesis we show the results from the numeric version; the others were not significantly different.

We created two versions of each output feature. One version used a median split and the other a quartile split. To give the best possible results, in this thesis we show the results from the median split. On average, the results from the quartile split were about 20 percentage points less.

CHAPTER 4

BIOLOGY

Biology Background

Michelene Chi, director of the Learning Sciences Institute of Arizona State University, has done extensive research showing that deep learning is one of the most important features in student learning (Chi, 2009). Kurt VanLehn, co-founder of two of the most successful NSF-funded centers for the study of the learning sciences, has written extensively on the importance of interactivity (2011). Danielle McNamara and Art Graesser, respectively the current and former directors of another such center at the University of Memphis, have also written extensively on this topic as well as on the importance of deep learning (Graesser, McNamara & VanLehn, 2005; Graesser et al., 2001).

Teaching styles and learning styles vary greatly between different geographical areas, between educational subjects, and even between like-minded teachers and students. Some teach with a grandiose vocabulary, some ask many questions of their students, and some are short and to the point. Everyone has their own way of explaining themselves, i.e., everyone has their own variation on a teaching style. For this portion of the thesis we built upon the elements from the physics experiment and added more elements in an attempt to gather more accurate results. One such element is the ability to gain the identity of a specific professor within a dialogue.

Author ID

Identifying a professor is a beginning step in helping to differentiate teaching styles in a virtual environment. From here we can explore relationships between these styles and their effectiveness with a student or in the classroom. Although earlier researchers used statistical approaches based on vocabulary choice and frequency for author identification, we use data mining algorithms, starting with the algorithmic work of Quinlan (1992). Aside from the C4.5 algorithm and the statistical t-test we used in the previous portion of our experiment, in this part we will also use the unsupervised algorithm K-Means. Whereas many others have attempted author identification using mainly the vocabulary of the writer, we are attempting to identify the speaker beyond the use of just vocabulary. Still using the parts of speech, grammatical phrasing, and other sentence breakdowns as in the physics experiments, we also use other key writing and speaking elements to find author identification and other relationships. Some of the other key aspects we use are readability tests, such as the Flesch Reading Ease, the Coleman-Liau Index, and the Automated Readability Index (ARI) and the use of domain words.

Readability Formulae

Readability tests are metrics for evaluating the readability of a set of texts, such as a full document or something as small as a sentence. They are designed to indicate the comprehension level needed to understand a passage written in English. Most everyone always writes in their own style, using the same types of words and the same types of sentence patterns. Due to this fact, the readability tests are another excellent way in using speech and writing styles as another element in our experiments.

There are multiple valid and usable readability tests which anyone can use for computing the readability of a document. For these experiments we chose three separate readability formulae to get multiple readability scores for each individual passage. The three we chose were the Flesch Reading Ease Test, the Coleman-Liau Index, and the Automated Readability Index (ARI). Each of these three readability assessments has their own, mostly unique, formula for dictating their results. The Flesch Reading Ease formula is:

$$206.835 - (1.015 * \frac{Total\ Words}{Total\ Sentences}) - (84.6 * \frac{Total\ Syllables}{Total\ Words})$$

and gives a result in the range of 0 – 100, where a lower number represents a higher comprehension level required for the reader to understand the passage. For example, the Flesch Reading Ease score of this paragraph is 34.0 which is best understood by university graduates. The formula for the Coleman-Liau Index is:

$$5.89 * \frac{Total\ Characters}{Total\ Words} - 0.29 * \frac{Total\ Sentences}{Total\ Words} - 15.8$$

and returns a result of the grade level needed to understand the passage. A variation of the Flesch Reading Ease formula, Coleman uses the number of characters in a segment per 100 words instead of the number of syllables. The Coleman-Liau Index of this section is 15.11 which corresponds directly to the American grade level needed to understand this passage, in this case an upper undergraduate student. Similar to the Coleman-Liau Index, the Automated Readability Index also uses characters, but has a slightly different formula. The ARI formula is:

$$4.71 * \frac{Total\ Characters}{Total\ Words} + 0.5 * \frac{Total\ Words}{Total\ Sentences} - 21.43$$

Like the Coleman-Liau Index, the Automated Readability Index returns a value that approximates the American grade level needed to comprehend the text. As such the ARI of this section is 13.44, indicating that a student of at least an undergraduate level would be needed to understand this paragraph.

Data Collection

The data to be used in this thesis is derived from a corpus collected by the Circsim-Tutor project. The Circsim-Tutor project (Evens and Michael, 2006; Freedman et al, 2001; Freedman et al., 2004; Circsim-Tutor, n.d.) built a language-based intelligent tutoring system for first-year medical students to learn about reflex control of blood pressure. This project was one of the first attempting to characterize human tutoring language and behavior and to adapt it for computer use (Kim, Freedman, Glass, & Evens 2006). The data from this project consists of transcripts of tutoring sessions between two of the founders, Dr. Joel Michael and the late Dr. Allen Rovick of Rush Medical College, and their students, on a one-on-one basis.

Some of these transcripts were collected in a face-to-face session; others are of the teacher and student communicating using a computer in an instant-messaging style. This data has communications between the two professors and over thirty students, totaling over 100,000 words, or the equivalent of a 400-page book when printed. The information housed within these transcripts has also been annotated with the results (correct / incorrect) for each step of the problems that the students are attempting to solve. Using these results, we attempt to correlate the linguistic features with their teaching effectiveness.

Data Preparation

As with the physics data, before any of the experiments could be conducted, a great deal of data cleanup and preparation had to be executed. This also reduced the frequency of erroneous parses by the Stanford Parser. Like the physics data, there were many incorrect spellings, issues with contractions, formulae, and many other types of linguistic objects to be fixed. To combat the erroneous parses we closely followed the same process we used in cleaning the physics data. Due to the fact that much of the biology data contains different wordage and was written as dialogues, there was some more specific cleaning that also needed to be done to the files.

In view of the fact that many of the biology files were either transcriptions of a dialogue or were actual typed dialogues between the tutor and student, there were a great many abbreviations used. Many of these abbreviations were valid English words, such as IS, so these, as with the physics equations, had to be addressed. Most of the other abbreviations were medical terms associated with the topics covered in these dialogues. Figure 9 shows a sample of a dialogue before cleaning and Figure 10 shows the same dialogue sample after cleaning. Table 8 we show an excerpt of the abbreviations used by the tutors and students and the subsequent change needed to guarantee correct parsing. By shifting these to the complete term, it also increased the accuracy of the unique word count for each of these individual words.

K24-tu-042-01: Good thinking (but be careful about this notion of "backed up").
K24-tu-042-02: Why did you predict that CC and TPR were both 0?
K24-st-043-01: CC does not change b/c there is no change in the sympathetic innervation on the heart w/ the change in the pacemaker.
K24-st-043-02: TPR does not change b/c of the same reason.
K24-tu-044-01: Another way of saying this is that both CC and TPR are determined by the reflex and the reflex hasn't happened yet in DR.

Figure 9: Sample from CIRCSIM Tutor Transcript - Original [K24]

Table 8: Biology Acronym Expansion

Original	Post-Processed	Original	Post-Processed
TPR	Total Peripheral Resistance	CO	Cardiac Output
RAP	Right Arterial Pressure	LAP	Left Arterial Pressure
MAP	Mean Arterial Pressure	SV	Stroke Volume
CVP	Central Venous Pressure	DR	Direct Response
CNS / CNX	Central Nervous System	RR	Reflex Response
ANS	Autonomic Nervous System	RA	Right Atrium
CBV	Central Blood Volume	SS	Steady State
IS	Inotropic State	LV	Left Ventricle
RV	Right Ventricle	BV	Blood Volume
CV	Cardiac Volume	BP	Baroreceptor Pressure
EDV	End-Diastolic Volume	EDP	End-Diastolic Pressure
CC	Cardiac Contractility	HR	Heart Rate

K24-tu-042-01: Good thinking (but be careful about this notion of "backed up").
K24-tu-042-02: Why did you predict that Cardiac Contractility and Total Peripheral Resistance were both no change?
K24-st-043-01: Cardiac Contractility does not change because there is no change in the sympathetic innervation on the heart with the change in the pacemaker.
K24-st-043-02: Total Peripheral Resistance does not change because of the same reason.
K24-tu-044-01: Another way of saying this is that both Cardiac Contractility and Total Peripheral Resistance are determined by the reflex and the reflex hasn't happened yet in Direct Response.

Figure 10: Sample from CIRCSIM Tutor Transcript - Cleaned [K24]

Finally, as with the physics data, we spell-corrected the corpus. Spelling correction, abbreviation expansion, and other corrections reduced the unique word count in the 51 separate dialogues of roughly 100,000 words from about 3000 words to 2645. Because these are dialogues we were able to split the statistics by speaker role into tutor versus student. The tutors spoke/wrote 58,582 words with a total of 2092 unique words, while the students totaled 31,705 words with a total of only 1731 unique words. To further divide the tutor numbers we can break it down to each tutor. Dr. Michael participated in 23 interactions, speaking/writing a total of 29,844 words with 1438 unique words. Dr. Rovick participated in 17 interactions and spoke/wrote a total of 20,819 words with 1,353 unique words.

Feature Identification

The biology data gave us a different set of features to use for our experiments. Though we were able to use much of the same in terms of parts of speech and sentence structure there were other features we added into the biology experiment.

The features were divided into the following categories.

1. Session Type

The data from the biology experiment was split into two separate types of data. The first case was transcriptions of the student and tutor’s spoken dialogue. The second case was an instant messaging set up with everything they typed being recorded into a file.

2. Speaker

Since this was a dialogue, either the tutor or the student could be speaking/writing the passage recorded.

3. Tutor / Student Name

There were multiple tutors and multiple students so there needed to be a way of distinguishing them. Knowing these names helped in two separate instances, one being as a way to show effectiveness and the other to identify the authors. To protect the students, the students were only referred to by their initials.

4. Tutor/Student Speech Ratio

One of the possible ways of distinguishing tutors and possibly showing teaching effectiveness was to show how much of a conversation was dominated by the tutor. We divided the total number of words the tutor used against the total words of the entire dialogue.

5. Readability Formula

This included the three separate readability formulae used in this portion of the experiment. As described above they included the Flesch Reading Ease, The Cole-Liau Index, and the Automated Readability Index or ARI.

6. Domain Word Usage

We counted the use of biology specific domain words used by either tutor or student within a dialogue. This allowed us to use the normalized percentage per sentence or normalized average per 100 words use of the domain words per individual. We obtain the domain words by the project medical words accessible at medicalwords.sourceforge.net.

7. *Unique Word*

We kept track of the use of unique word per individual to be able to differentiate the vocabulary usage by each individual and to use the normalized percentage per sentence and normalized average per 100 words of unique words.

8. *Turn Statistics*

With the data coming from dialogues, we were able to keep track of the number of turns each individual had. A turn is whenever a new individual begins to speak. We were able to collect data such as the average number of sentences per turn and the average number of words per turn.

9. *Student Educational Data – Student's Final Grade*

At the student level there were grades available for all the students. These grades were figured based on answers they provided during the original experiment. There were seven questions regarding medical knowledge they discussed in the course of the dialogues. The grades were simply the number correct over the total possible and stored as a percentage.

Features 10 through 15 corresponded to the same features described in the physics chapter.

10. *Average POS Counts per Dialogue – Noun, Adjective, Adverb, Preposition*

11. *Other Constituent Counts – Noun Phrase(NP), Adjective Phrase(AdjP), Adverb Phrase (AdvP), Prepositional Phrase (PP)*

12. *Average Words per Sentence*

13. *Average Sentence Tree Height*

14. *Average Subordinate Clauses per Essay*

15. *Average "Non-Consecutive" VPs per Essay*

Compound Features

With the exception of session type, speaker, and tutor/student name all of the other features were numeric. As with the physics data we added an additional input feature for each element, the variant $\log_2 n$ to reduce the influences of larger values. We also used a binning technique to separate the data into 10 equal-width bins.

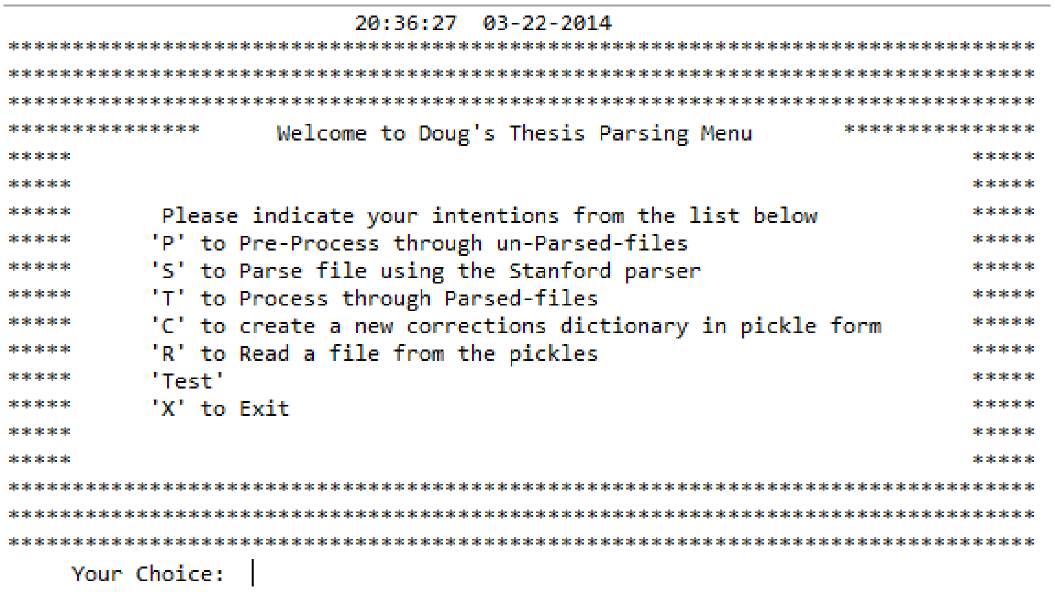
In addition to the binning technique we created multiple versions of each output feature when the numeric elements were used. The two versions created used a median split, where all data was given a high or low tag, and a quartile split, where the data was split into four quarters. Most of the numeric results are shown using the median split.

CHAPTER 5

METHODOLOGY

Python Coding Introduction

The majority of the experiments were processed using Python programming code for everything from the data cleanup, processing the cleaned data into the format and file structure that the Stanford Parser could understand, processing the results of the parser, getting these results into a format that WEKA could use, and finally reading the final results. Most of these phases required multiple iterations to get the data to a usable format. This chapter describes the Python code used to generate the final usable data, retrieve data returned from the Stanford Parser, and input and retrieve data from WEKA. Figure 11 shows the basic menu options that allowed us to run multiple iterations of cleaning and running of the data.



```
                           20:36:27  03-22-2014
********************************************************************************
********************************************************************************
********************************************************************************
***************      Welcome to Doug's Thesis Parsing Menu       ***************
*****                                                                      *****
*****                                                                      *****
*****       Please indicate your intentions from the list below            *****
*****      'P' to Pre-Process through un-Parsed-files                      *****
*****      'S' to Parse file using the Stanford parser                     *****
*****      'T' to Process through Parsed-files                             *****
*****      'C' to create a new corrections dictionary in pickle form       *****
*****      'R' to Read a file from the pickles                             *****
*****      'Test'                                                          *****
*****      'X' to Exit                                                     *****
*****                                                                      *****
*****                                                                      *****
********************************************************************************
********************************************************************************
********************************************************************************
     Your Choice:  |
```

Figure 11: Main Program Screen Shot - Main Menu

Preprocessing, the Art of Cleaning Files

Unfortunately data in its raw form is rarely in a completely computer legible format. For both the physics portion and the biology portion a generous amount of file cleanup was required before more processing could be completed. Due to the magnitude of both sets of data, knowing all of the needed changes on a first pass was impossible. For this reason we needed to make multiple passes on the cleanup adding needed changes each time. To help us find the oddball elements, one of the first things we did was to create a program that would process each file, and gather all the unique words and their number of occurrences. This data was then written to a text file for further analysis.   Once this was created, we could then use another method to go through and find words that were incorrectly spelled, and in some cases completely incorrectly used. After creating a list of all misspelled and correctly spelled words, we were able to write a program to go through and change all the misspelled words to their correctly spelled cousins. We did all of this correcting because we wanted all data to be consistent to guarantee a reliable set of statistics. Also for our purposes we were not concerned with the actual words themselves, but their role as parts of speech. The fact that many of the physics students misspelled ‘acceleration’ was not pertinent to us, just the idea that, for our purposes, that ‘acceleration’ was a noun (not a verb).

For the majority of the cleanup the main focus and methodology utilized regular expressions and substitutions. As stated in the physics chapter, we needed to change certain letters and words so that the Stanford Parser would correctly parse the words to their corresponding part of speech. So with the use of a program mainly using regular expressions we were able to correct a vast majority of the misspelled words, change contractions to their full-paired words, remove punctuation that was unnecessary for the parsing or statistics, expand abbreviations, and generate other substitutions. In Table 9 we show examples of the majority of the changes made to the files for both the physics and the biology data.

Table 9: File Cleaning Examples

Original	Cleaned
acceleration.the	acceleration. the
amt	amount
carin	car in
Find truck.,	find truck
‘m	Am
n’t	Not
a.	Aa
b/c	because

Preparing for the Parser

The Stanford Parser is written in the Java programming language and is open for anyone willing to download it to use. Luckily, Python allows the use of implementing the Java Virtual Machine in a few different ways within its API. In our programs we use two of these processes to access the Java run methods. In the case of the Stanford Parser, the use of Python’s system call was enough to run the basic code. Though not the most efficient due to the fact that for each call, the JVM had to be started, loaded, and then run, it was adequate for our needs. Once all the cleanup was completed, we made sure all the files were in a consistent format so that there was no chance of the parser parsing one differently than another due to a minor format difference. The code of this portion of the program was very simple but powerful. The free software allows one to feed a text file with the sentences to be parsed and can provide an output to another text file with the parsed information. With this in mind we were able to feed the now cleaned files to the parser and get an equal number of files back in the parsed arrangement. The form we used was the parser's tree form, seeing as it is easier to extrapolate all the needed data from each tree.

Tree Climbing and Preparing for WEKA

Using the Stanford Parser showed that there were still some random word issues within the files. Some of these issues were such that we needed to correct them so that we could gather accurate data. An example of this was where some words had a dot in the middle (see Table 9). Within our post-parser processing we needed to handle these abnormal renditions. These extra functions included removing punctuation within a word, so as not to alter counts, gathering a set of unique words, placing everything into separate lists, applying many different statistical methods, and collecting all statistical elements.

The Stanford Parser's Lexicalized Tree Parser was used to gather data from the parser. The Stanford Parser's Lexicalized Tree Parser returned a sentence tree with all the sentence phrases and parts of speech in one file. The main function for processing the parsed files was a tree-traversing method that not only gathered the individual words and their corresponding parts of speech, but also collected all the sentence phrases. From this point we were able to create multiple statistical collectors to gather the individual figures we needed.

One of the main duties of this step of the program, aside from gathering all the pertinent data, is to get the data ready to be processed by WEKA. Rather than use WEKA's way of converting a standard CSV into their required ARFF file, we wrote a program to write the ARFF file itself. This way we could more accurately dictate the key aspects needed within the ARFF file. The reasoning behind this is that due to all the types of data we are using and the many types of binning available to the numeric types, we needed to be able to easily change between them without worrying about the conversion.

Once we were able to get all the data into data structures that allowed us to pick and choose any combination of data aspects, we needed a way to choose key combinations. At first there was nothing to indicate which combination of features would give us the most accurate evaluation. To combat this uncertainty, we created a way to iterate through all the different combinations. At first we reduced the number of elements to choose from down to 17. These included percentage of nouns, percentage of verbs, percentage of noun phrases, average verb phrases per 100 words, and many others. To be able to iterate through all 2^{16} * 17 possible combinations (16 possible inputs and one output) we wrote a multiple-part program. This process took every possible permutation, created the ARFF file for each combination, fed it into J48, and retrieved the accuracy rating. Since the original search space was huge, the reasoning behind this was to guide J48 away from local maxima in the search space.

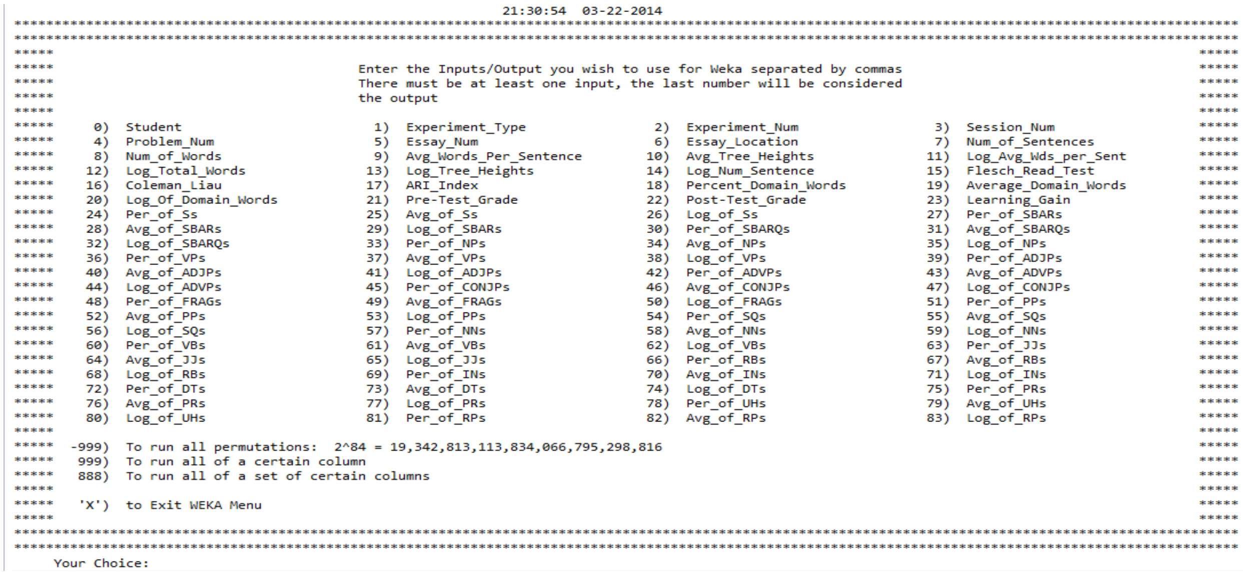
To iterate through each combination, a multi-phase function was created. It loops down from the total number of each grouping possible computed by the formula:

$$2^{(Number\ of\ Elements-1)}-1$$

In our first attempt at using this process there were 17 columns or data elements so we had $2^{(17-1)}$ -1 or 65,535 possible combinations. Starting at 65,535 we sent each number, down to 1, to a function that took this number and figured out its bit field and returned a list corresponding to the element numbers to be added for this iteration. For example 65,535 would correspond to the bit field of 1111 1111 1111 1111, which in this case meant that all data elements were to be added to this ARFF file for WEKA processing. The next number sent, 65,534, would correspond to 1111 1111 1111 1110, which would correspond to sending all but the 16th element for this iteration.

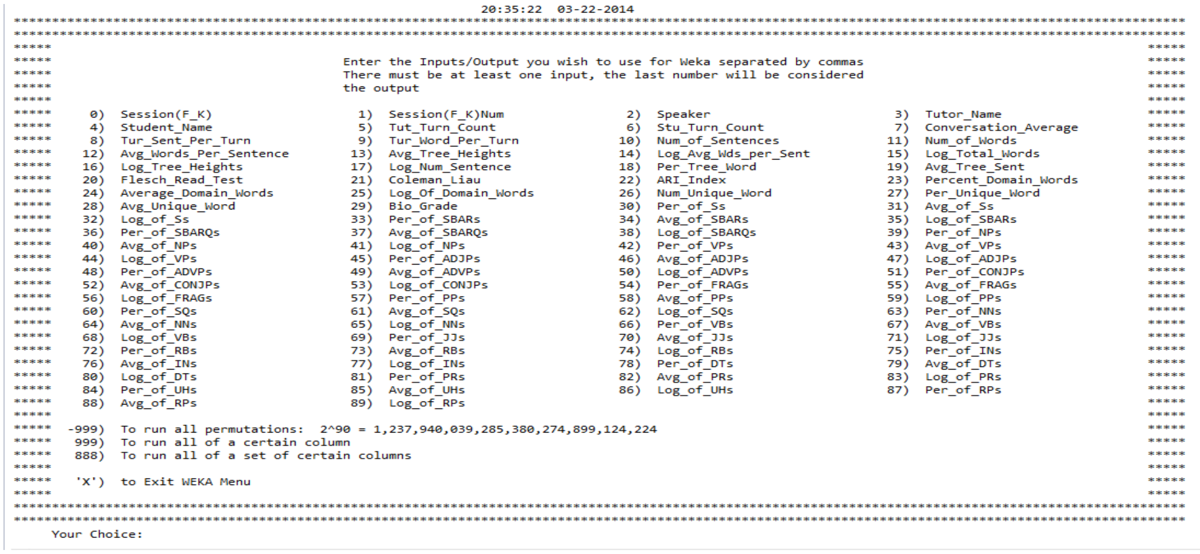
One should notice that with the previous examples, there are technically only 16 columns when in fact we are using 17. This is because one column is considered the output or the column that WEKA will attempt to guess. Due to this fact, the output column is not included in these permutations. To work with this issue, another function is needed to modify the columns so that this permutation works even when the ‘output column’ is in the middle of the list. This separate function takes the created bit field and moves the current columns to correspond to the needed columns. For example if we were using column 13, after the bit field is computed, it is sent to a function to move all columns from column 13 up one. This will allow the functions following to access the correct data element.

For most of the experiments, there are many more elements then just 17 that we could use for inputs and output in WEKA format. Initially we could have used over 125 separate entities. The physics and biology experiments had data points that were unique to that particular experiment. Due to this, we implemented a user interface that would only display the features pertaining to the immediate experiment. Figures 12 and 13 show the Selection Screen-shots. Figure 12 shows the physics data set, and Figure 13 displays the biology data set. These screens allowed us to choose specifically any features we hypothesized would give an accurate result.



```
                                                             21:30:54  03-22-2014
*********************************************************************************************************************************************************
*********************************************************************************************************************************************************
*****                                                                                                                                               *****
*****                                      Enter the Inputs/Output you wish to use for Weka separated by commas                                     *****
*****                                      There must be at least one input, the last number will be considered                                     *****
*****                                      the output                                                                                               *****
*****                                                                                                                                               *****
*****      0)  Student                     1)  Experiment_Type                2)  Experiment_Num                 3)  Session_Num                    *****
*****      4)  Problem_Num                 5)  Essay_Num                      6)  Essay_Location                 7)  Num_of_Sentences               *****
*****      8)  Num_of_Words                9)  Avg_Words_Per_Sentence        10)  Avg_Tree_Heights              11)  Log_Avg_Wds_per_Sent           *****
*****     12)  Log_Total_Words            13)  Log_Tree_Heights              14)  Log_Num_Sentence              15)  Flesch_Read_Test               *****
*****     16)  Coleman_Liau               17)  ARI_Index                     18)  Percent_Domain_Words          19)  Average_Domain_Words           *****
*****     20)  Log_Of_Domain_Words        21)  Pre-Test_Grade                22)  Post-Test_Grade               23)  Learning_Gain                  *****
*****     24)  Per_of_Ss                  25)  Avg_of_Ss                     26)  Log_of_Ss                     27)  Per_of_SBARs                   *****
*****     28)  Avg_of_SBARs               29)  Log_of_SBARs                  30)  Per_of_SBARQs                 31)  Avg_of_SBARQs                  *****
*****     32)  Log_of_SBARQs              33)  Per_of_NPs                    34)  Avg_of_NPs                    35)  Log_of_NPs                     *****
*****     36)  Per_of_VPs                 37)  Avg_of_VPs                    38)  Log_of_VPs                    39)  Per_of_ADJPs                   *****
*****     40)  Avg_of_ADJPs               41)  Log_of_ADJPs                  42)  Per_of_ADVPs                  43)  Avg_of_ADVPs                   *****
*****     44)  Log_of_ADVPs               45)  Per_of_CONJPs                 46)  Avg_of_CONJPs                 47)  Log_of_CONJPs                  *****
*****     48)  Per_of_FRAGs               49)  Avg_of_FRAGs                  50)  Log_of_FRAGs                  51)  Per_of_PPs                     *****
*****     52)  Avg_of_PPs                 53)  Log_of_PPs                    54)  Per_of_SQs                    55)  Avg_of_SQs                     *****
*****     56)  Log_of_SQs                 57)  Per_of_NNs                    58)  Avg_of_NNs                    59)  Log_of_NNs                     *****
*****     60)  Per_of_VBs                 61)  Avg_of_VBs                    62)  Log_of_VBs                    63)  Per_of_JJs                     *****
*****     64)  Avg_of_JJs                 65)  Log_of_JJs                    66)  Per_of_RBs                    67)  Avg_of_RBs                     *****
*****     68)  Log_of_RBs                 69)  Per_of_INs                    70)  Avg_of_INs                    71)  Log_of_INs                     *****
*****     72)  Per_of_DTs                 73)  Avg_of_DTs                    74)  Log_of_DTs                    75)  Per_of_PRs                     *****
*****     76)  Avg_of_PRs                 77)  Log_of_PRs                    78)  Per_of_UHs                    79)  Avg_of_UHs                     *****
*****     80)  Log_of_UHs                 81)  Per_of_RPs                    82)  Avg_of_RPs                    83)  Log_of_RPs                     *****
*****                                                                                                                                               *****
*****   -999)  To run all permutations:  2^84 = 19,342,813,113,834,066,795,298,816                                                                  *****
*****    999)  To run all of a certain column                                                                                                       *****
*****    888)  To run all of a set of certain columns                                                                                               *****
*****                                                                                                                                               *****
*****    'X')  to Exit WEKA Menu                                                                                                                    *****
*****                                                                                                                                               *****
*********************************************************************************************************************************************************
*********************************************************************************************************************************************************
      Your Choice:
```

Figure 12: WEKA Selection Screen-Shot - Physics Data



```
                                                            20:35:22  03-22-2014
********************************************************************************************************************************************************
********************************************************************************************************************************************************
*****                                                                                                                                              *****
*****                                      Enter the Inputs/Output you wish to use for Weka separated by commas                                     *****
*****                                      There must be at least one input, the last number will be considered                                     *****
*****                                      the output                                                                                               *****
*****                                                                                                                                              *****
*****     0)  Session(F_K)                  1)  Session(F_K)Num               2)  Speaker                       3)  Tutor_Name                     *****
*****     4)  Student_Name                  5)  Tut_Turn_Count                6)  Stu_Turn_Count                7)  Conversation_Average           *****
*****     8)  Tur_Sent_Per_Turn             9)  Tur_Word_Per_Turn            10)  Num_of_Sentences             11)  Num_of_Words                   *****
*****    12)  Avg_Words_Per_Sentence       13)  Avg_Tree_Heights             14)  Log_Avg_Wds_per_Sent         15)  Log_Total_Words                *****
*****    16)  Log_Tree_Heights             17)  Log_Num_Sentence             18)  Per_Tree_Word                19)  Avg_Tree_Sent                  *****
*****    20)  Flesch_Read_Test             21)  Coleman_Liau                 22)  ARI_Index                    23)  Percent_Domain_Words           *****
*****    24)  Average_Domain_Words         25)  Log_Of_Domain_Words          26)  Num_Unique_Word              27)  Per_Unique_Word                *****
*****    28)  Avg_Unique_Word              29)  Bio_Grade                    30)  Per_of_Ss                    31)  Avg_of_Ss                      *****
*****    32)  Log_of_Ss                    33)  Per_of_SBARs                 34)  Avg_of_SBARs                 35)  Log_of_SBARs                   *****
*****    36)  Per_of_SBARQs                37)  Avg_of_SBARQs                38)  Log_of_SBARQs                39)  Per_of_NPs                     *****
*****    40)  Avg_of_NPs                   41)  Log_of_NPs                   42)  Per_of_VPs                   43)  Avg_of_VPs                     *****
*****    44)  Log_of_VPs                   45)  Per_of_ADJPs                 46)  Avg_of_ADJPs                 47)  Log_of_ADJPs                   *****
*****    48)  Per_of_ADVPs                 49)  Avg_of_ADVPs                 50)  Log_of_ADVPs                 51)  Per_of_CONJPs                  *****
*****    52)  Avg_of_CONJPs                53)  Log_of_CONJPs                54)  Per_of_FRAGs                 55)  Avg_of_FRAGs                   *****
*****    56)  Log_of_FRAGs                 57)  Per_of_PPs                   58)  Avg_of_PPs                   59)  Log_of_PPs                     *****
*****    60)  Per_of_SQs                   61)  Avg_of_SQs                   62)  Log_of_SQs                   63)  Per_of_NNs                     *****
*****    64)  Avg_of_NNs                   65)  Log_of_NNs                   66)  Per_of_VBs                   67)  Avg_of_VBs                     *****
*****    68)  Log_of_VBs                   69)  Per_of_JJs                   70)  Avg_of_JJs                   71)  Log_of_JJs                     *****
*****    72)  Per_of_RBs                   73)  Avg_of_RBs                   74)  Log_of_RBs                   75)  Per_of_INs                     *****
*****    76)  Avg_of_INs                   77)  Log_of_INs                   78)  Per_of_DTs                   79)  Avg_of_DTs                     *****
*****    80)  Log_of_DTs                   81)  Per_of_PRs                   82)  Avg_of_PRs                   83)  Log_of_PRs                     *****
*****    84)  Per_of_UHs                   85)  Avg_of_UHs                   86)  Log_of_UHs                   87)  Per_of_RPs                     *****
*****    88)  Avg_of_RPs                   89)  Log_of_RPs                                                                                         *****
*****                                                                                                                                              *****
*****  -999)  To run all permutations:  2^90 = 1,237,940,039,285,380,274,899,124,224                                                               *****
*****   999)  To run all of a certain column                                                                                                       *****
*****   888)  To run all of a set of certain columns                                                                                               *****
*****                                                                                                                                              *****
*****   'X')  to Exit WEKA Menu                                                                                                                    *****
*****                                                                                                                                              *****
********************************************************************************************************************************************************
********************************************************************************************************************************************************
     Your Choice:
```

Figure 13: WEKA Selection Screen Shot - Biology Data

Originally we ran the initial entire set of 2^{17} elements, as we had narrowed down our choices to those specific 17 features. Since that time we have added many more elements and even removed a few. To allow us to be able to pick and choose the inputs and output we needed, there exists the option to choose either a single run or an option to run a set of any number of inputs with all the combinations of those inputs to a single output. This gave us the freedom to run any number of experiments for either physics or biology. If we needed to look at a specific run or to see the actual tree J48 created, we just had to run the GUI version of WEKA.

CHAPTER 6

RESULTS

For each of the physics and biology experiments we attempted to answer a number of questions which addressed linguistic complexity and student learning. For each research trial we attempted to answer domain specific questions and general questions that would be addressed in both physics and biology.

Physics Results

The physics experiments were geared towards answering questions that involved how students' learned through the ITSPOKE experiment. Our attempts to answer some of these questions involved both the statistical two-tailed paired t-Test and/or the C4.5 Algorithm implemented by WEKA. The following section details the questions and the results of these questions.

J48 Decision Tree Results

Does Linguistic Complexity Determine Learning?

Originally we used WEKA to dictate the pertinent input fields by giving it all of the 16 fields at once with post-test score as its output. This gave J48 the assignment of pruning the decision tree down to what is believed to be the most accurate fields. Table 10 shows the results of this experiment using a median split.

Table 10: J48 Directed Accuracy

Input Features		Output	Accuracy
Experiment Type	Essay Locale	Post Test Score	66.12%
Average Words/Sentence;	Average Tree Height		
Percent SBARs	Percent Noun Phrases		
Percent Verb Phrases	Percent Adjective Phrases		
Percent Adverb Phrases	Percent Nouns		
Percent Verbs	Percent Prepositions		

. Since it was only 66% accurate, we wondered if a hill climbing issue was preventing J48 from finding a better solution. In a hill climbing problem, the machine learning algorithm finds a ‘peak’ or high accuracy and without any more guidance ‘believes’ that it has found the highest peak. To circumvent this possible problem, we fed all seventeen fields into the custom permutation algorithm to allow J48 to be fed every possible combination of features. Though not a large increase, the permutation function did find a slightly higher accuracy and kappa rating. The function found a solution that used a few less input features which is shown in Table 11 with the accuracy of the output by median split.

Table 11: J48 Guided Accuracy

Input Features		Output	Accuracy
Experiment Type Average Tree Height Percent Noun Phrases Percent Adjective Phrases Percent Nouns	Average Words/Sentence Percent SBARs Percent Verb Phrases Percent Adverb Phrases Percent Verbs	Post Test Score	68.82%

Does Essay Locale (First vs Last) Determine Essay Complexity

We wanted to know whether essay locale determined essay complexity. To study this question, we asked to what extent initial or final essay locale could be used to predict whether a given measure of essay complexity was greater or less than the median value. Table 12 shows the results for this experiment with the output separated by the median split.

Table 12: Results for Physics Essay Locale

Input	Output	Accuracy
Essay locale	Avg tree height	55.53%
Essay locale	Avg sentence length	53.22%
Essay locale	Avg SBAR count	53.59%

From Table 12 above one can see that using essay locale as an input and these three averages as three separate outputs, the accuracy for this question does not appear significant.

Does Experiment Type Determine Complexity

We wanted know whether we could identify any of the causes of complexity. We tested all 2^{17} combinations of the 17 basic features. In Table 13 we show sample results from this experiment. As the reader can see, we attempt to see whether experiment type or essay locale can predict whether the percent of SBARs is greater or less than the median value. The percentages of accuracy are very similar to the previous experiments.

Table 13: Results for Physics Experiment Type

Inputs	Output	Accuracy
Experiment type	Percent of SBARs	57.60 %
Essay locale	Percent of SBARs	53.59 %

Is There a Relationship between Linguistic Features and Physics Knowledge?

Our previous results were at the essay level. Here, we wanted to look for relationships at the student level. In particular, we were looking for relationships between measures of complexity at the student level and measures of knowledge or learning, such as pretest score, posttest score, and learning gain.

We computed the student average sentence length for initial essays by dividing the total number of words written by the student in any of their initial essays by the total number of sentences in those essays, and similarly for their final essays.

We computed student average SBAR percent the same way, dividing total SBARs in any of the student's initial (or final) essays by the number of words in those essays. As Table 14 displays, being able to determine learning based on complexity from J48 is at most 56% accurate.

Table 14: Results for Linguistic Complexity to Educational Data

Input	**Output**	**Accuracy**
Student average sentence length computed over all initial essays	Pretest score	50.40%
	Posttest score	51.01%
	Learning gain	50.20%
Input	**Output**	**Accuracy**
Student average SBAR % computed over all initial essays	Pretest score	50.40%
	Posttest score	53.44%
	Learning gain	50.20%
Student average sentence length computed over all final essays	**Output**	**Accuracy**
	Posttest score	52.41%
	Learning gain	56.29%
Student average SBAR % computed over all final essays	**Output**	**Accuracy**
	Posttest score	53.43%
	Learning gain	50.05%

Table 15 shows the results for the J48 attempting to find the relationship between educational data of physics and two of the complexity markers, average tree height and percentage of subordinate clauses (SBARs). As shown, there is no relationship.

Table 15: Results for Educational Data to Linguistics

Input	Output	Accuracy
Post-test Score	Average Tree Height	55.90%
Post-test Score	Percent of SBARs	52.59%
Average Tree Height & Percent of SBARs	Post-test Score	53.42%

Two-Tailed Paired T-Test

Is There a Relationship Between Initial and Final Essays

One of our earlier questions asked whether there was a significant difference in sentence complexity between first and last essays for the same problem.

We first used a two-tailed paired t-test to determine whether final student essays were significantly longer than the corresponding initial essays. After deleting problems where students only wrote one essay, there were 482 essays. The average lengths were significantly different, averaging 53 words for the initial essays and 129 for the final essays. The value t = -22.38 (df = 481) is significant at the p < .001 level.

Next we used the two-tailed paired t-test to determine whether final student essays contained a larger percentage of SBARs than the corresponding initial essays. We obtained t = 2.97 (df = 481), which is significant at the p < .001 level. Thus students did write more complex essays after tutoring for each problem.

Although we did not run this experiment for all the measures of sentence complexity, in other experiments the three measures performed equivalently.

Biology Results

The biology experiment was a two-fold experiment that took what we were attempting to discover in the physics experiment, add additional features, and answer additional questions. Overall the results were mixed, vectored towards the idea that most of our data exhibited some type of relationship, but not one that could be easily identified. The following headings ask the original questions we attempted to answer and what we found in our journey to answer the question.

Can Basic Linguistic Complexity be Used for Author Identification?

There have been many attempts to accomplish author identification in years past, though as stated previously, a vast majority of them use individual words, clusters of words, or some other form of word usage, for example using the author's use of 'while' versus 'whilst' in the crusade to prove who authored the Federalist Papers. With the use of computers and machine learning we can expand to more complicated ways of researching how someone speaks or writes. In our experiments we attempted to use lexical elements such as the many different parts of speech and the ways in which people structure their sentences to point to the author of a speech or essay. By using this format we found some success in our author identification process. With the help of the iteration technique described in the methodology section we were able to run over 120,000 separate tests at a time.

Answer Using J48

Table 16 shows the results of the top scoring runs. The input features used are shown along with the accuracy percentage and the kappa statistic for that run. Figure 14 and Figure 15 show the decisions trees that J48 created for the two most accurate feature sets.

Table 16: Accuracy of Identification of Tutors

List of Input Features	Accuracy	Kappa
Avg. Adverb Phrases Avg. Questions	87.5%	0.74
Percent of Noun Phrases Percent of Conjunction Phrases Percent of Verb Phrases Percent of Verbs Percent of Adjectives Percent of Pronouns	85.0%	0.69
Percent of Noun Phrases Percent of Verb Phrases Percent of Verbs Percent of Conjunction Phrases	82.5%	0.65
Avg. Questions Avg. Adverbs Avg. Pronouns Log. Interjections	80.0%	0.59
Avg. SBARs Avg. Questions Avg. Adjectives Log. Interjections	80.0%	0.58
Avg. Questions Avg. Verbs Avg. Adjective Phrases Avg. Pronouns Log Interjections Avg. Prepositional Phrases	77.5%	0.53

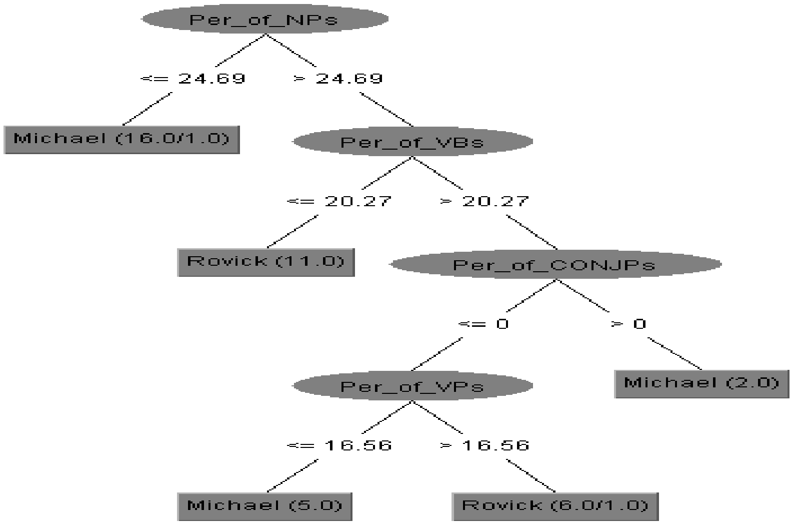


Figure 14: J48's Decision Tree – 2nd Best Run – 85%

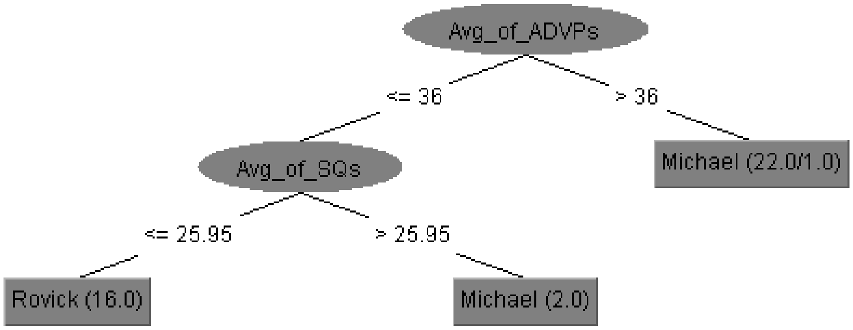
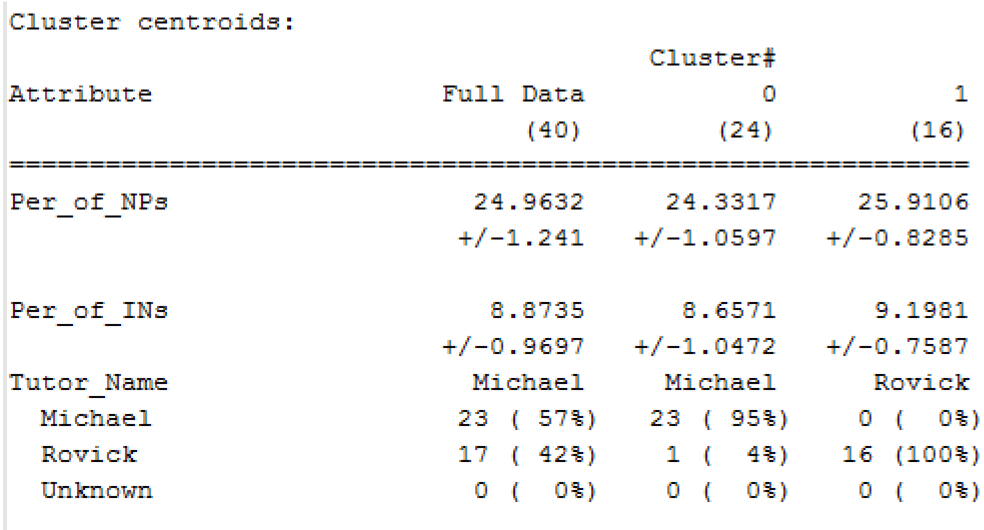


Figure 15: J48 Decision Tree - Best run – 87.5%

In some cases we were able accurately predict the correct tutor over 80% of the time. In the two most accurate runs the most prevalent features that J48 was inclined to utilize for inputs were: average number of adverb phrases, average number of questions, percentage of noun phrases, and percentage of verb phrases.

Answer Using Clustering

The accuracy of clustering is more difficult to determine due to the fact that clustering gives a subjective output (Japkowiczs & Shaw, 2011). The following figures show how a visualization of the clusters created by running K-Means with the same dataset as our top J48 run. Many times in clustering, there are overlaps in the clusters because there are always some outliers. That is, some points that are part of one cluster end up in another cluster usually due to an instance outside the norm for that cluster. Figure 16 shows the textual output of the K-Means execution for using the top two features from J48, percentage of noun phrase and percentage of prepositions, to identify the tutor. Included in the textual output are the standard deviations and means that the algorithm used to divide the clusters.



```
Cluster centroids:
                                        Cluster#
Attribute                   Full Data          0           1
                                 (40)       (24)        (16)
============================================================
Per_of_NPs                    24.9632    24.3317     25.9106
                             +/-1.241  +/-1.0597   +/-0.8285

Per_of_INs                     8.8735     8.6571      9.1981
                            +/-0.9697  +/-1.0472   +/-0.7587
Tutor_Name                    Michael    Michael      Rovick
  Michael                   23 ( 57%)  23 ( 95%)    0 (  0%)
  Rovick                    17 ( 42%)   1 (  4%)   16 (100%)
  Unknown                    0 (  0%)   0 (  0%)    0 (  0%)
```

Figure 16: K-Means Text Output: Percent of Noun Phrases; *Percent of Prepositions; Tutor Name*

For this example, the K-Means algorithm placed into cluster 1 one all noun phrase percentages around 24 and all percentages of prepositions around 8, which were identified as Dr. Michael. Correspondingly it placed any percentages of noun phrases around 26 and percentages of prepositions around 9, which it identified with Dr. Rovick, into the second cluster. This signifies that it found that those sets of data belonged in the same cluster. Strictly speaking it found a reasonable association between the specified ranges of noun phrases, the specified ranges of preposition percentages, and the specific tutors. Figures 17 and Figure 18 show the visualization of the K-Means clustering.   The circles represent the clusters, with the x's signifying Dr. Michael and the circles indicating Dr. Rovick.

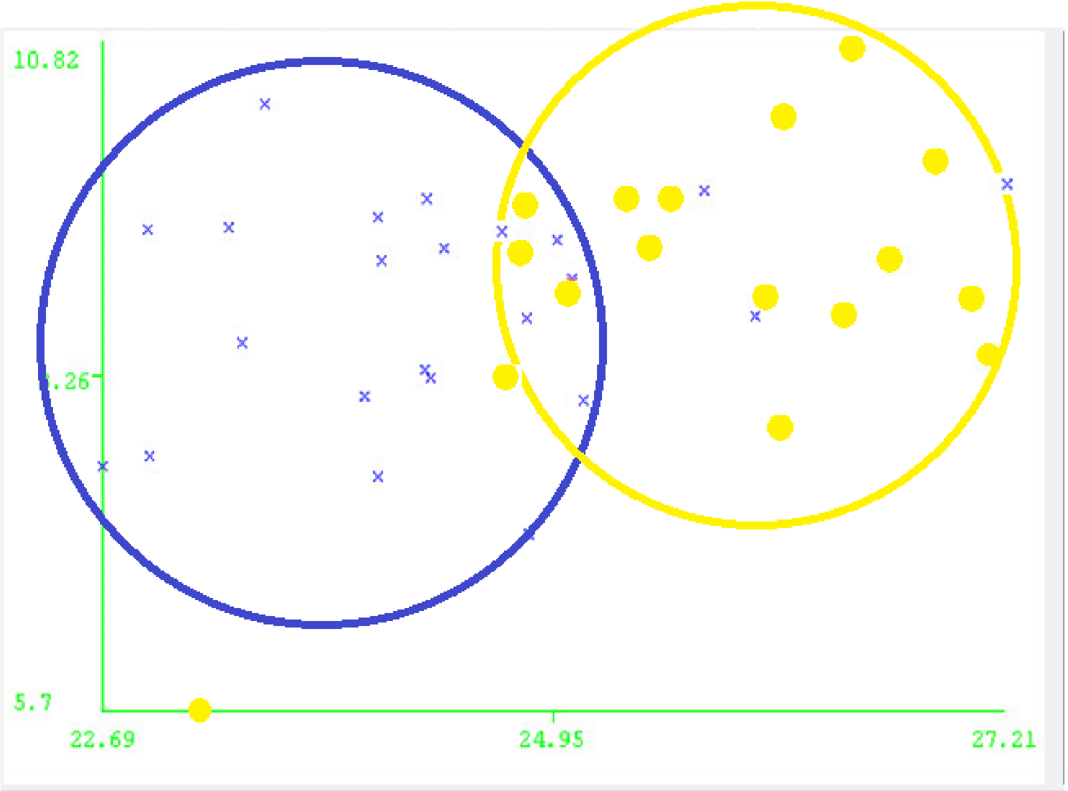


Figure 17: K-Means Cluster: X-Axis: Percent of Noun Phrases; Y-Axis: Percent of Prepositions; Shape: Tutors

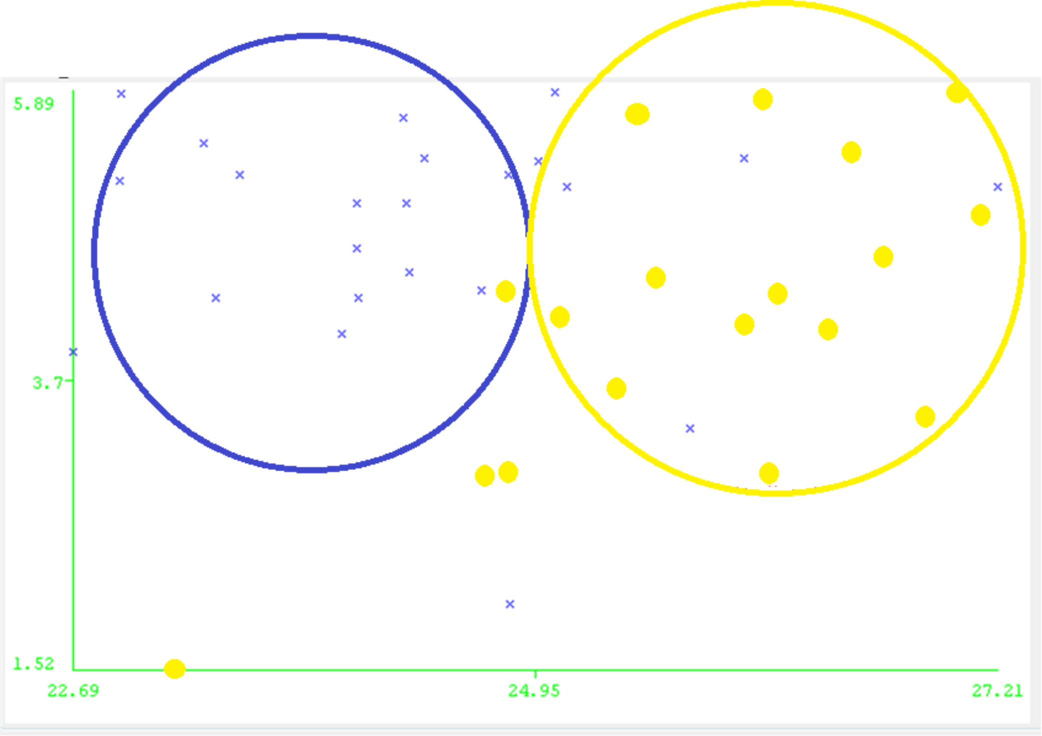


Figure 18: K-Means Cluster: X-Axis: Percent of Noun Phrases; Y-Axis: Percent of SBARs; Shape: Tutors

The two previous examples clearly show that there are relationships between these features and the specified tutor. As displayed in the Figure 18, there will usually be outlying data points outside the designated clusters. Clustering allows for these without majorly affecting a cluster group. In this case, this can easily be explained since no one always writes in the exact same format every time.

Which Linguistic Features are Possibly Better at Identification?

Since we were able to relatively accurately show that some linguistic features can be used to help author identification, we wanted to decipher which elements were the most advantageous. The features that were the most effective were features that were more likely to being included as input elements in the higher scoring runs in J48, and those that showed better cluster groups in K-Means. The most prevalent features from the J48 procedure were average adverb phrases per 100 sentences and average questions per 100 sentences, as they provided the most accurate tree. However, though this was the most accurate there were other features that continued to be present in many other J48 operations, especially the higher accuracy results. These features were percentage of noun phrases, percentage of verb phrases, average subordinate clauses, average words per sentence and the Flesch Reading Test score, and even to a lesser degree the other two reading indices.

Do more successful students . . .

Use More Domain Words?

Table 17 shows the relationship between students' use of domain words and their biology grade. As the reader can see there is no apparent relationship.

Table 17: J48 Accuracy: Student Domain Word & Biology Grade

Input	Output	Accuracy
Percentage Domain Words	Biology Grade	22.86%
Average Domain Words	Biology Grade	25.71%
Log_2 Domain Words	Biology Grade	22.86%
Biology Grade	Percentage Domain Words	45.71%
Biology Grade	Average Domain Words	51.53%
Biology Grade	Log_2 Domain Words	51.53%

Have Longer Sentences?

Table 18 shows the relationship between students' average sentence length and their biology grade. There is no apparent relationship.

Table 18: J48 Accuracy: Student Sentence Length & Biology Grade

Input	Output	Accuracy
Average Words Per Sentence	Biology Grade	22.86%
Log Words Per Sentence	Biology Grade	25.71%
Biology Grade	Average Words Per Sentence	42.86%
Biology Grade	Log Words Per Sentence	42.86%

Have More Complex Sentences?

Table 19 shows the relationship between students' usage of subordinate clauses (SBARs) and their biology grade. There is no apparent relationship.

Table 19: J48 Accuracy: Student SBARs & Biology Grade

Input	Output	Accuracy
Percentage SBARs	Biology Grade	31.43%
Average SBARs	Biology Grade	34.29%
Log_2 SBARs	Biology Grade	22.86%
Biology Grade	Percentage SBARs	51.43%
Biology Grade	Average SBARs	51.43%
Biology Grade	Log_2 SBARs	45.71%

Have Higher Tree Heights?

Table 20 shows the relationship between students' average sentence tree heights and their biology grade. There is no apparent relationship.

Table 20: J48 Accuracy: Student Tree Height & Biology Grade

Input	Output	Accuracy
Average Tree Heights	Biology Grade	17.14%
Percent Tree Heights Per Wd	Biology Grade	28.57%
Biology Grade	Average Tree Heights	45.71%
Biology Grade	Percent Tree Heights Per Wd	51.43%

Utilize More Words?

Table 21 shows the relationship between students' usage of unique words and their biology grade. There is no apparent relationship.

Table 21: J48 Accuracy: Student Unique Word & Biology Grade

Input	Output	Accuracy
Percentage Unique Words	Biology Grade	42.86%
Average Unique Words	Biology Grade	25.71%
Log_2 Unique Words	Biology Grade	17.14%
Biology Grade	Percentage Unique Words	54.29%
Biology Grade	Average Unique Words	51.43%
Biology Grade	Log_2 Unique Words	48.57%

Ask More Questions?

Table 22 shows the relationship between students' usage of questions and their biology grade. There is no apparent relationship.

Table 22: J48 Accuracy: Student Questions & Biology Grade

Input	Output	Accuracy
Percentage Questions	Biology Grade	25.71%
Average Questions	Biology Grade	17.14%
Log_2 Question	Biology Grade	20.00%
Biology Grade	Percentage Questions	60.00%
Biology Grade	Average Questions	51.43%
Biology Grade	Log_2 Question	51.43%

Do Teachers That Pose More Questions Elicit More Understanding From Their Students?

Table 23 shows the relationship between the teachers' usage of questions and the students' biology grade. There is no apparent relationship.

Table 23: J48 Accuracy: Teacher Questions & Student Biology Grade

Input	Output	Accuracy
Percentage Questions	Biology Grade	11.43%
Average Questions	Biology Grade	11.43%
Log_2 Question	Biology Grade	25.10%
Biology Grade	Percentage Questions	42.86%
Biology Grade	Average Questions	45.71%
Biology Grade	Log_2 Question	40.00%

Is There a Relationship Between a Teacher's Linguistics And Their Teaching Effectiveness?

Table 24 shows the relationship between the teachers' usage of subordinate clauses (SBARs) and the students' biology grade. There is no apparent relationship.

Table 24: J48 Accuracy: Teacher SBARs & Student Biology Grade

Input	Output	Accuracy
T - Percentage SBARs	Biology Grade	17.14%
T - Average SBARs	Biology Grade	25.71%
T - Log_2 SBARs	Biology Grade	28.57%
Biology Grade	T - Percentage SBARs	40.00%
Biology Grade	T - Average SBARs	48.57%
Biology Grade	T - Log_2 SBARs	48.84%

Is There a Measurable Difference Between Tutors?

Seeing as we were able to reasonably predict the correct author in our author identification experiment, it can clearly say that there are measurable differences between the two tutors. As it is shown in the J48 decision trees (Figure 14 and Figure 15) by the greater than 80% accuracy rating and the 70% kappa statistic, and in the distinct groupings in the K-Means visualizations (Figure 17 and Figure 18), Dr. Michael and Dr. Rovick had some distinct ways of communicating with their students.

Combined Experimental Results

Do Better Students Use More Complicated Data Structures?

From the datasets we have utilized during these experiments the answer to this question is undetermined. Though many of the accuracies were around the baseline of 50%, more research in this matter should be completed. Different algorithms may shed more light on these relationships.

Does Linguistic Complexity Determine Learning?

This was one of the major questions we were attempting to answer throughout these experiments. The idea that there is a relationship between learning and linguistic complexity could provide a better understanding of student learning.

Throughout the duration of these experiments, there was always a suggestion of a connection between linguistic complexity and understanding, but unfortunately there were no definitive relationships found in any of the machine learning experiments.

In the physics section of our experiments, the best relationship found between any type of known linguistic complexity and educational markers was barely better than a coin flip. This was shown with the J48 accuracy ratings between students' linguistic complexity (using SBARs) and their grades, scarcely reaching 60%.

CHAPTER 7

FINAL THOUGHTS

Conclusion

In this thesis we delved into the notion of author identification utilizing linguistic measures exclusively. We also investigated whether relationships could be found between sentence complexity and student learning. We then attempted to combine these two notions into searching for aspects of teaching effectiveness.

There appeared to be only weak correlations between writing complexity and student learning ability. As shown in the results of the physics chapter, showing specific relationships from the J48 output was barely above a baseline at best. The fact that we could not find a displayable relationship does not destroy the hypothesis that there is not an underlying connection. As proven with the results from the statistical analysis, there are some fundamental associations at work between these features. At this time we cannot indicate what the relationships are, but we know there are relations.

For author identification, we were able to execute a vast number of iterations, numbering around one million, by using code developed in Python and interfacing with WEKA directly. In terms of J48 accuracy percentages, most of these iterations only resulted in decent, but not Earth shattering scores. Many scores, due to poo combinations, were even below the baseline of 50%. There were however, a few excellent outcomes from a select group of features that indicate that doing author identification via sentence structures and linguistic complexity is plausible. Added into these results are the undeniable associations shown in the K-Means clustering. K-Means produced a handful of great cluster groups using multiple aspects of linguistic complexity. With the results of these two machine learning algorithms, we believe that linguistic complexity, as used in a dialogue setting, is not only plausible, but extremely viable.

One of the main topics of this thesis has been the value of employing machine learning techniques. Throughout the entirety of these experiments, much of what we accomplished would not have been possible without the use of computer processors and more specifically machine learning algorithms. One of the final inquiries in this thesis was if machine learning algorithms or techniques would be advantageous in answering the hypotheses proposed in this thesis. The results gained, both positive and not so positive, would not have been achievable without the use of machine learning. These techniques have a great value that could continue to assist researchers in developing answers to these complex issues of author identification, student learning, and effective teaching techniques.

Future Work

This thesis identifies many research opportunities in multiple areas. Since we were able to find some relationships such as with author identification, more research into this area would be beneficial. Gathering more data from dialogues from the same discipline, the medical education field, would allow some of these statistics to become even more precise. Because there are such an overwhelming number of possible combinations of features for use in these experiments, obviously it would be problematic to attempt every single one of them. A continuation of this would be to attempt to try all of the 2^{162} combinations and narrow down the selection pool to a handful of more practical combinations. Due to there being such a tremendous number to sift through, the use of a high-powered multi-processing cluster would be necessary for this type of activity.

Not only would collecting additional data from the same pool be advantageous, finding correlations with other disciplines would also beneficial. Adding in dialogues or written essays from computer science and other fields could show how these disciplines could be interconnected in terms of learning.

In general, more data would be very valuable in both the author identification process and the journey to find relationships between teaching techniques and learning. In the case of the physics data, there were many different written files, but each file was limited. In the case of the biology the opposite was true, there were fewer individual files, but much more in terms of content. More variety in size and quantity would be very helpful in advancing these techniques.

BIBLIOGRAPHY

Ai, Hua and Diane Litman. (2006). Comparing Real-Real, Simulated-Simulated, and Simulated-Real Spoken Dialogue Corpora. In Proceedings of the AAAI Workshop on Statistical and Empirical Approaches for Spoken Dialogue Systems. Boston, MA. Pp. 1-6.

Bullington, Jim, Ira Endres, and Muhammad Asadur Rahman. (2007). Open-Ended Question Classification Using Support Vector Machines. In Proceedings of the Eighteenth Midwest Artificial Intelligence and Cognitive Science Conference. Chicago, IL. Pp. 45-49.

Bouckaert, Remco R., Eibe Frank, Mark Hall, Richard Kirkby, Peter Reutemann, Alex Seewald, and David Scuse. (2013). University of Waikato, Machine Learning Group. WEKA Manual for Version 3-6-10.

Chi, Michelene T.H. (2009). Active-Constructive-Interactive: A Conceptual Framework for Differentiating Learning Activities. *Topics in Cognitive Science* 1:73-105.

Circsim-Tutor Project. Illinois Institute of Technology, Department of Computer Science and Rush College of Medicine, Department of Physiology. http://www.cs.iit.edu/~circsim/. Last accessed April 11, 2014.

De Marneffe, Marie-Catherine, Bill MacCartney, and Christopher D. Manning. (2006). Generating Typed Dependency Parses from Phrase Structure Parses. In Proceedings of LREC-06. Pp. 449-454.

Evens, Martha W. and Joel Michael (2006). *One-on-One Tutoring by Humans and Computers*. Mahwah, NJ: Lawrence Erlbaum Associates.

Freedman, Reva, Nicholas Haggin, Dessislava Nacheva, Tom Leahy, and Richard Stilson. (2004). Using a Domain-Independent Reactive Planner to Implement a Medical Dialogue System. In AAAI Fall Symposium on Dialogue Systems for Health Communication. Pp. 24-31.

Freedman, Reva, Byung-In Cho, Michael Glass, Yujian Zhou, Jung Hee Kim, Bruce Mills, Feng-Jen Yang, and Martha W. Evens. (2001). Adaptive Processing in a Medical Intelligent Tutoring System. In Proceedings of NAACL 2001 Workshop on Adaptation in Dialogue Systems. Pp. 33-40.

Freedman, Reva, Yujian Zhou, Michael Glass, Jung Hee Kim, and Martha W. Evens. (1998). Using Rule Induction to Assist in Rule Construction for a Natural-Language Based Intelligent Tutoring System. In Proceedings of the Twentieth Annual Conference of the Cognitive Science Society. Pp 362-367.

Getsoft.com. Analyzing GeneXproTools Models Statistically http://www.gepsoft.com/gxpt4kb/Chapter10/Section1/SS07.htm. Last accessed on April 13, 2014.

Graesser, Arthor, Danielle McNamara, and Kurt VanLehn. (2005). Scaffolding deep Comprehension Strategies through Point & Query, AutoTutor, and iSTART. *Educational Psychologist* 40: 225-234.

Graesser, A. C., Kurt VanLehn, Carolyn P. Rose, Pamela W. Jordan, and Derek Harter. (2001). Intelligent Tutoring Systems with Conversational Dialogue. *AI Magazine* 22(4):39-52.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. (2009). The WEKA Data Mining Software: An Update. *SIGKDD Explorations* 11(1):10-19.

Japkowicz, Nathalie and Mohak Shah. (2011). *Evaluating Learning Algorithms: A Classification Perspective*. Cambridge, England: Cambridge University Press.

JGAAP (Java Graphical Authorship Attribution Program). EVL Labs. http://evllabs.com/jgaap/w/index.php/Main_Page. Last accessed February 19, 2013.

Juola, Patrick, John Sofko, and Patrick Brennan. (2006). A Prototype for Authorship Attribution Studies. *Literary and Linguistic Computing* 21:169-178.

Juola, Patrick and John Sofko. (2004). Proving and Improving Authorship Attribution Technologies. In Proceedings of Canadian Symposium for Text Analysis.

Juola, Patrick and Efstathios Stamatatos. (2013). Overview of the Author Identification Task at PAN 2013. In Proceedings of PAN 2013.

Kim, Jung Hee, Reva Freedman, Michael Glass, and Martha W. Evens. (2006). Annotation of Tutorial Dialogue Goals for Natural Language Generation. *Discourse Processes* 42(1):37-74.

Kim, Jung Hee, Michael Glass, Reva Freedman, and Martha W. Evens. (2000). Learning the Use of Discourse Markers in Tutorial Dialogue for an Intelligent Tutoring System. In Proceedings of the Twenty-Second Annual Conference of the Cognitive Science Society. Pp. 262-267.

Liptak, Adam. (2000). Paper Chase. *The New York Times*, November 26, 2000. Available at http://www.nytimes.com/books/00/11/26/reviews/001126.26liptakt.html.

Lipshultz, Michael, Diane Litman, Pamela Jordan, and Sandra Katz. (2011). Predicting Changes in Level of Abstraction in Tutor Responses to Students. In Proceedings of the 24th International Florida Artificial Intelligence Research Society Conference. Pp. 525-530.

Mosteller, Frederick and David Wallace. (1984 [1964]). *Applied Bayesian and Classical Inference: The case of the Federalist Papers*, 2/e. New York, NY: Springer-Verlag.

Quinlan, John. Ross. (1992). *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann.

Reference.com. Dictionary.com http://dictionary.reference.com/browse/Natural%20language%20processing. Last accessed April 13, 2014.

Rose, Carolyn P., Diane Litman, Dumisizwe Bhembe, Kate Forbes, Scott Silliman, Ramesh Srivastava, and Kurt VanLehn. (2003). A Comparison of Tutor and Student Behavior in Speech Versus Text Based Tutoring. In Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing. V. 2 pp. 30-37

Santorini, Beatrice. (1995). Part-of-Speech Tagging Guidelines for the Penn Treebank Project. University of Pennsylvania, Natural Language Processing Project. Technical Report MS-CIS-90-47 LINC LAB 178.

Sostek, Anya. (2013). Duquesne Professor Helps ID Rowling as Author of “The Cuckoo’s Calling”. *The Pittsburgh Post-Gazette*, July 16, 2013. Available at http://www.post-gazette.com/news/education/2013/07/16/Duquesne-professor-helps-IDRowling-as-author-of-The-Cuckoo-s-Calling/stories/201307160124.

Steinbach, Michael, Levent Ertoz, and Vipin Kumar. (2004). The Challenges of Clustering High Dimensional Data. In Luc T. Wille (ed), *New Directions in Statistical Physics: Applications in Econophysics, Bioinformatics, and Pattern Recognition*, New York, NY: Springer-Verlag. Pp. 273-309.

VanLehn, K. (2011). The relative effectiveness of human tutoring, intelligent tutoring systems and other tutoring systems. *Educational Psychologist* 46(4):197-221.

Witten, Ian H., Eibe Frank, and Mark A. Hall. (2011). *Data Mining: Practical Machine learning Tools and Techniques, 3/e*. Burlington, MA: Morgan Kaufmann.

Wikipedia.org. The Free Encyclopedia. http://en.wikipedia.org/wiki/Natural_language_processing. Last accessed April 13, 2014.

Zhao, Ying. (2007). Effective Authorship Attribution in Large Document Collections. Ph.D. dissertation, RMIT University, School of Computer Science and Technology.