

THE PENNSYLVANIA STATE UNIVERSITY  
SCHREYER HONORS COLLEGE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NLP ANNOTATION TOOLS FOR MANUAL MARKUP OF CONTENT IN SHORT  
SUMMARIES AND ESSAYS

ALEX DRIBAN  
SPRING 2019

A thesis  
submitted in partial fulfillment  
of the requirements  
for baccalaureate degrees  
in Computer Science and Biochemistry and Molecular Biology  
with honors in Computer Science

Reviewed and approved\* by the following:

Rebecca Passonneau  
Professor of Computer Science and Engineering  
Thesis Supervisor

John Hannan  
Associate Department Head of Computer Science and Engineering  
Honors Adviser

\* Signatures are on file in the Schreyer Honors College.

## ABSTRACT

Automated evaluation of students' reading and writing skills could enable teachers to more efficiently assess student abilities. One important skill is mastery of content: a student's ability to understand reading material and demonstrate their understanding through short summaries or essays about what they have read. Student mastery of content can be evaluated by comparing students' written summaries to those written by a "wise crowd," considered to be a gold standard of content mastery. There are several automated methods for building the wise crowd content model (called a pyramid) and scoring student summaries, such as PyrEval. These methods have been tested against manual methods for accuracy using DUCView, a tool for performing and collecting manual content annotations of summaries. However, PyrEval and DUCView are only suitable for simple summaries. I have developed a new tool called SEAView, using the DUCView source code as a starting point, for content annotation of essays that have a special format. This special format includes a summary in the header of the essay, followed by a body that makes an argument. DUCView was last updated in 2005, and the original Java source code has since been lost. However, the DUCView JAR file has been decompiled using two decompilers to recover the code. I have made modifications to the DUCView source code and created this new tool using the decompiled Java code. The manual annotations created using this tool will be used for developing, training, and testing machine learned models to performed automated annotations. It will also be used to study the relationship between summary and essay content, and for scoring essays of this format according to their content.

## TABLE OF CONTENTS

LIST OF FIGURES .....	iii
LIST OF TABLES .....	iv
ACKNOWLEDGEMENTS .....	v
Chapter 1 Introduction .....	1
1.1 Evaluation of Reading and Writing Skills.....	1
1.1.1 Content Mastery .....	2
1.1.2 Different Genres of Content Mastery .....	2
1.1.3 Capturing Main Ideas: Rubric versus Wise Crowd Scoring .....	3
1.2 Pyramid Method of Summary Content Annotation .....	5
1.2.1 Summary Content Units .....	6
1.2.2 Building the Pyramid .....	8
1.2.3 Using the Pyramid.....	9
1.2.4 Pyramid Scores.....	10
1.3 Pyramid Annotation Tools .....	11
1.3.1 Manual Methods of Summary Content Analysis: DUCView .....	12
1.3.2 Automatic Methods of Summary Content Analysis: PyrEval.....	15
1.4 Argument Annotation.....	16
1.4.1 Next Step in Content Annotation: Summaries versus Arguments .....	16
1.4.2 Elementary Discourse Units.....	18
1.4.3 Argumentative Discourse Units .....	20
1.4.4 Need for a Tool for Manual Content Annotation of Argumentative Writing..	21
Chapter 2 DUCView: A Tool for Summary Content Annotation .....	22
2.1 Decompilation of the DUCView JAR File .....	22
2.2 Analysis of Decompiled DUCView Source Code .....	23
2.3 Control Flow of DUCView.....	25
2.4 Modifications to DUCView .....	28
Chapter 3 SEAVIEW: A New Tool for Argument Content Annotation .....	31
3.1 Design of SEAVIEW .....	31
3.2 Implementation of SEAVIEW Design.....	33
3.3 Using SEAVIEW .....	37
3.4 Analysis of SEAVIEW Design.....	48
Chapter 4 Conclusions and Future Work.....	50
BIBLIOGRAPHY.....	51

## LIST OF FIGURES

Figure 1. Two SCUs annotated from six gold-standard summaries. ....	6
Figure 2. Annotated SCUs. ....	7
Figure 3. Four-level pyramid constructed from four gold-standard summaries. ....	8
Figure 4. Optimal content score of a peer summary. ....	10
Figure 5. Pyramid creation in DUCView version 1.4. ....	12
Figure 6. Peer annotation in DUCView version 1.4. ....	13
Figure 7. Peer annotation score in DUCView version 1.4. ....	14
Figure 8. Evidence relation in RST. ....	18
Figure 9. Sentences containing two clausal EDUs. ....	19
Figure 10. A sentence with a phrasal EDU. ....	20
Figure 11. Pyramid creation view in DUCView version 1.5. ....	26
Figure 12. Peer annotation view in DUCView version 1.5. ....	27
Figure 13. Comparison of version 1.4 and 1.5 score windows. ....	30
Figure 14. Essay annotation schematic. ....	32
Figure 15. SEAView ready screen. ....	37
Figure 16. Pyramid loaded in SEAView. ....	38
Figure 17. Selected SCU in SEAView. ....	39
Figure 18. An error during EDU creation in SEAView. ....	40
Figure 19. An EDU and SCU annotated in SEAView. ....	41
Figure 20. Changed EDU label in SEAView. ....	42
Figure 21. A SCU without a corresponding EDU in SEAView. ....	43
Figure 22. An ordered SEA table in SEAView. ....	44
Figure 23. SCU-EDU alignment window. ....	45
Figure 24. Peer annotation view in SEAView. ....	46

Figure 25. Model essays window in SEAView. ....47

**LIST OF TABLES**

Table 1. Descriptions of DUCView source code files. ....	23
Table 2. DUCView version 1.5 changes. ....	28
Table 3. Essay annotation workflow using DUCView and SEAView. ....	34
Table 4. Primary functions of SEAView source code files. ....	35

## ACKNOWLEDGEMENTS

I would like to thank all the members of the Penn State Natural Language Processing lab for their inspiration and discussions, but especially the following people: Dr. Passonneau, my thesis adviser, for providing me with an enriching and valuable research experience, and her feedback and guidance throughout this project; Saptarashmi for his amazing work on decompiling the DUCView source code into a usable format; and Yanjun for her help with understanding this project and learning about DUCView. I would also like to thank my honors adviser, Dr. Hannan, for his guidance during my undergraduate career. Finally, I am grateful for all the support of my family and friends.

## **Chapter 1**

### **Introduction**

#### **1.1 Evaluation of Reading and Writing Skills**

Development of reading and writing skills is a priority for educators since these skills and fundamental to success in many careers. However, data from the National Center for Education Statistics suggests that most students in the United States fail to demonstrate grade level reading and writing proficiency on national assessments [1], [2]. The reasons for this systematic inadequacy are complex. One possible cause for concern is insufficient preparation of instructors: several national surveys on the quality of writing instruction in high school and middle school classrooms suggest that the majority of teachers feel they did not receive enough training on how to teach writing. In addition, teachers indicated many of their most commonly used writing activities lack analysis or personalization of information. These concerns were most prevalent in teachers who taught subjects other than language arts, although teachers generally agreed that teaching writing should be a shared responsibility among different subjects [3], [4]. These surveys call into question current teaching paradigms and suggest they should be reassessed to improve the quality of writing instruction in the United States.



### **1.1.1 Content Mastery**

Content mastery is an important domain of reading and writing skills for educators to evaluate and provide feedback on for students to improve their abilities. It encompasses a student's ability to understand reading material and demonstrate their understanding through essays about what they have read [5]. In a piece of writing, evaluation of content mastery focusses on the quality of ideas incorporated into the text, rather than details such as grammar and style.

In research conducted in both low- and high-achieving third-grade classrooms, most feedback teachers tended to give students was related to surface-level problems in their students' writing, such as spelling or grammar issues. However, the amount of content-level feedback on the students' drafts significantly predicated the quality of content and organization in students' final work. This suggests that content-level feedback is important for improving the quality of students' writing, but some teachers fail to give adequate feedback on content issues [6].

### **1.1.2 Different Genres of Content Mastery**

Content mastery itself is an abstract concept, and it can be shown in many ways, depending on the subject. Several applications relevant to this paper include summarization and argumentative essays. For instance, students can display content mastery by reading news articles and writing short summaries about the content they read. Students who exhibit content mastery are able to discern the most important details from the text and put them in their own words in a short summary [5]. Thus, those students have displayed their mastery of reading comprehension and summarization. Summarization itself is an important skill for students to

learn because it is required for certain academic and professional assignments, such as writing papers or presentations using complex source materials. Furthermore, it is a useful study aid for some students. Therefore, testing students' summarization skills is a practical exercise [7].

Another relevant means of demonstrating content mastery is through the writing of argumentative essays. Studies have demonstrated that for most children, persuasive essay writing is more difficult than other types of writing because it is more cognitively demanding [8], [9]. This complexity can present challenges in evaluating mastery of argumentative writing. In addition, student performance on national surveys of persuasive writing ability have historically been poor [2], [10]. However, the ability to write effective and logical arguments is important in the context of students' academic, professional, and personal lives. Therefore, evaluating mastery of argumentative writing is also an important avenue for educator intervention.

### **1.1.3 Capturing Main Ideas: Rubric versus Wise Crowd Scoring**

Rubrics are one of the most commonly used methods of assessing student content mastery. They are a standardized method of scoring student responses based on the fulfillment of various categories determined to be important to their mastery of the subject matter. In principle, rubrics reduce subjectivity in the grading process and can save time in individually analyzing student essays. However, in practice, creating high quality rubrics that accurately reflect learning goals is a time-consuming and difficult process for many educators [11]. Two rubric writers, each of whom could be highly qualified for their positions, may present different interpretations of the most important main ideas of their rubrics. Furthermore, scoring using rubrics requires judgment to decide if a student has adequately met the requirements of a given category;

subjective biases may present an obstacle to objective scoring using a rubric [5]. These issues are compounded for teachers who feel unprepared to teach writing, particularly teachers in math and science.

These challenges suggest an opportunity for a new type of scoring and feedback system to be implemented to save educators time and effort. One possible approach to reducing complexity and subjectivity in rubric generation involves the use of a “wise crowd.” A wise crowd is distinguished from an irrational crowd by four key criteria, according to James Surowiecki. These criteria including the following characteristics: diversity of opinions, independence of opinions (people are not influenced by other members of the wise crowd), decentralized knowledge (members of the crowd form opinions based on their own specialized, local knowledge), and aggregation of opinions (the group makes a collective decision based on individual judgments). He argues that wise crowds often make better decisions than any individual in the crowd could make. Surowiecki gives a few examples of wise crowds in real life scenarios, such as Google’s PageRank algorithm, which powers its famous search engine. PageRank works by ranking search results according to the frequencies with which each result is linked to by other sites. Suppose the search result is Page A, and an arbitrary site that linked to A is page B. Then the weight of the link from B to A is also determined by the rank of page B in PageRank. PageRank assumes that important content on the Web is important because the rest of the Web indicates its importance by linking to it. Therefore, the Web is considered the wise crowd that powers Google’s search algorithm, and certainly, the diversity of the users of the Web satisfies Surowiecki’s four criteria [12].

The wise crowd has relevance to scoring and providing feedback on reading and writing tasks because it can be used to determine the most important ideas for a given task. To use the

Google analogy, rather than having one person rank every search result according to importance, the wise crowd of Web users collectively rank the search results, which generally results in a more objectively useful ranking. Similarly, recall the example in Section 1.1.2 of students demonstrating content mastery through short summaries of news articles. A wise crowd could be asked to complete the same task, and those ideas which are most commonly represented in the wise crowd summaries constitute the highest weighted portions of a content rubric. Then, this wise crowd generated rubric can be distributed to other educators, and used with relatively little cost in terms of time or effort. This method, known as the “pyramid method,” is discussed further in Section 1.2. Another benefit afforded by wise crowd rubrics for content is the standardization of scores, which may enhance fairness in grading and promote standardization across curricula [5].

## **1.2 Pyramid Method of Summary Content Annotation**

The pyramid method of summary content annotation is an application of wise crowd rubrics toward grading of summaries. As input, this method takes a series of several gold-standard summaries – summaries that meet a high standard of quality of content. These gold-standard summaries constitute the wise crowd for the pyramid method. Each of the summaries is used to construct a content model that ranks the importance of content, to be used as a rubric for assessing the quality of content in a peer summary – a summary written by a student. This content model is built upon summary content units [13], as discussed in Section 1.2.1.

### 1.2.1 Summary Content Units

Summary content units, or SCUs, are the building blocks of the content model for the pyramid method of summary content annotation. Each SCU is a short unit of text that generally conveys a single idea, derived from semantic similarity between one or more summaries. SCUs may be as short as one word, but never longer than one sentence; the length may vary depending on the annotator. Figure 1, an excerpt from Nenkova and Passonneau [13], depicts the annotation of two SCUs from a single sentence from each of six gold-standard summaries.

*A1. The industrial espionage case involving GM and VW began with the hiring of Jose Ignacio Lopez, an employee of GM subsidiary Adam Opel, by VW as a production director.*

*B3. However, he left GM for VW under circumstances, which along with ensuing events, were described by a German judge as “potentially the biggest-ever case of industrial espionage”.*

*C6. He left GM for VW in March 1993.*

*D6. The issue stems from the alleged recruitment of GM’s eccentric and visionary Basque-born procurement chief Jose Ignacio Lopez de Arriortua and seven of Lopez’s business colleagues.*

*E1. On March 16, 1993, with Japanese car import quotas to Europe expiring in two years, renowned cost-cutter, Agnacio Lopez De Arriortua, left his job as head of purchasing at General Motor’s Opel, Germany, to become Volkswagen’s Purchasing and Production director.*

*F3. In March 1993, Lopez and seven other GM executives moved to VW overnight.*

**Figure 1. Two SCUs annotated from six gold-standard summaries.**

In Figure 1, each of the six sentences is preceded by a letter and a number. The letter indicates from which of the six gold-standard summaries the sentence originated; the number indicates the sentence number within the gold-standard summary. In each of the six sentences, a portion is underlined, and in several of the six, another portion is italicized. The underlined

portions are grouped to form one SCU, while the italicized portions correspond to a second SCU [13].

Figure 2, also from Nenkova and Passonneau [13], shows the two SCUs that resulted from the annotation in Figure 1.

**SCU1** (w=6): *Lopez left GM for VW*  
*A1.* the hiring of Jose Ignacio Lopez, an employee of GM ... by VW  
*B3.* he left GM for VW  
*C6.* He left GM for VW  
*D6.* recruitment of GM's ... Jose Ignacio Lopez  
*E1.* Agnacio Lopez De Arriortua, left his job ... at General Motor's Opel ... to become Volkswagen's ... director  
*F3.* Lopez ... GM ... moved to VW

**SCU2** (w =3) *Lopez changes employers in March 1993*  
*C6.* in March, 1993  
*E1.* On March 16, 1993  
*F3.* In March 1993

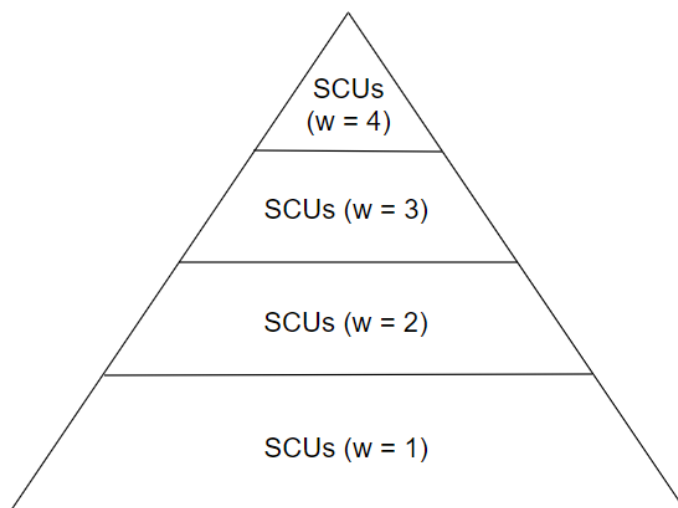
**Figure 2. Annotated SCUs.**

The first line of each SCU gives it a unique index (1, 2...), a weight from 1 to 6, corresponding to the number of gold-standard summaries in which the SCU was found, and a short descriptive label. SCU1, corresponding to the underlined portions of text in Figure 1, occurred in some form in all six gold-standard summaries, and therefore, has a higher weight than SCU2, corresponding to the italicized portions, which was only found in three of the six summaries. Each of the lines below the bolded SCU number, beginning with a letter followed by a number, correspond to the portion of each summary in which the SCU occurred. These lines are referred to as contributors to the SCU to which they belong. Contributors are grouped to form a single SCU based on the similarity of the content of each contributor.

In addition to the two SCUs shown in Figure 2, the remaining unformatted portions of the six sentences shown in Figure 1 will be further annotated to produce more contributors to other SCUs.

### 1.2.2 Building the Pyramid

A pyramid is built by annotating all of the sentences in each of the gold-standard summaries into SCUs. After all sentences have been annotated, the SCUs are sorted by their weights into a pyramid structure, in which the most highly weighted SCUs appear at the top, and the SCUs with the lowest weight appear at the bottom of the pyramid. This relationship is depicted in Figure 3, which shows a four-level pyramid constructed from four gold-standard summaries.



**Figure 3. Four-level pyramid constructed from four gold-standard summaries.**

The term pyramid is also descriptive of the observed Zipfian distribution of the SCUs within the gold-standard summaries. The width of the pyramid levels is indicative of the number of summaries that express each SCU. At the top of the pyramid, which contains the SCUs that were expressed in all of the gold-standard summaries, there are relatively few SCUs. At the bottom of the pyramid, which contains SCUs that were expressed in only one gold-standard summary each [13].

### 1.2.3 Using the Pyramid

The pyramid can serve as a content model for evaluating the quality of ideas in peer summaries, or those written by students, outside of the wise crowd. This process is called peer annotation. Given a pyramid of order  $n$ , describing a pyramid with  $n$  levels, it is possible to determine the optimal content in a peer summary with  $m$  contributors. The peer summary should only include content from any of the lower levels of the pyramid, unless the SCUs in the upper levels of the pyramid have all already been represented in the peer summary. For instance, if  $m$  is less than the number of SCUs in the  $n$ th level of the pyramid, the peer summary is considered optimal if and only if the pyramid contains only SCUs from the  $n$ th level. If  $m$  is greater than the number of SCUs in the  $n$ th level of the pyramid, the peer summary must contain all of the SCUs from the  $n$ th level, and as many as possible in the  $(n - 1)$ th level, and so on down the pyramid, until all  $m$  contributors have been annotated. The score of a peer summary can be calculated in several ways, discussed in Section 1.2.4 [13].



### 1.2.4 Pyramid Scores

Nenkova and Passonneau [13] described the optimal content score of a peer summary in Figure 4.

$$\text{Max} = \sum_{i=j+1}^n i \times |T_i| + j \times \left( X - \sum_{i=j+1}^n |T_i| \right), \text{ where } j = \max_i \left( \sum_{t=i}^n |T_t| \geq X \right).$$

**Figure 4. Optimal content score of a peer summary.**

Suppose the scoring is based on an  $n$  level pyramid, and each level is denoted  $T_1$  through  $T_n$ , where  $T_1$  is the lowest level and  $T_n$  is the highest level. The subscript denotes the weight of the level.  $|T_i|$  indicates the number of SCUs in level  $i$ .  $j$  represents the lowest level from which the peer summary will draw SCUs. Then the formula for the optimal content score of a peer summary with  $X$  SCUs is shown in Figure 4, which simply states the summary will contain all of the SCUs from the levels above  $j$ , and its remaining SCUs from level  $j$ . Its score is the sum of the weights of the SCUs.

Given the optimal score, the “pyramid score” or “quality score” of a peer summary is simply the ratio of the sum of the weights of the SCUs,  $D$ , to Max. This score represents the quality of the SCUs found in the peer summary, or the quality of the content in the summary, according to the gold-standard summaries. A “coverage score” can be derived, roughly indicating the quantity of the ideas in the peer summary, by substituting the average number of SCUs in the gold-standard summaries for  $X$  in the equation in Figure 4. Thus, the coverage score is calculated using the expected number of SCUs based on the gold-standard summaries, rather

than the number of SCUs found in the peer summary. Similarly,  $D$  is divided by the coverage Max to determine the coverage score.

These scores complement each other in reflecting different aspects of a high-quality summary. For instance, a peer summary with only one SCU, of weight  $n$  for an  $n$ -level pyramid, would have a quality score of 1. However, its coverage score would likely be low since the gold-standard summaries likely contain more SCUs. Conversely, a peer summary with many weight 1 SCUs would have a high coverage score, but a low quality score. Neither of these summaries is considered optimal when both scores are considered. The “comprehensive score” is calculated as the harmonic mean of the quality and coverage scores, and reflects both the quality and quantity of ideas in a peer summary. These scores can then be used to evaluate and provide feedback on student summaries, and evaluate both the quality and depth of their written content in summaries. However, it is still important to pair this content selection metric with a rubric for evaluating linguistic qualities of the summary, such as the order the SCUs are presented in the summary [13].

### **1.3 Pyramid Annotation Tools**

Unfortunately, the pyramid annotation process is challenging to complete manually in writing. There are several issues with this process, such as making sure all the content in each of the gold-standard summaries has been annotated, keeping track of the sources of each SCU contributor in the pyramid, and calculating the scores. For these reasons, a tool called DUCView was developed to enable manual pyramid annotation through a virtual interface, as discussed in

Section 1.3.1. Several automated methods of pyramid creation also exist, such as PyrEval, which is analyzed in Section 1.3.1.

### 1.3.1 Manual Methods of Summary Content Analysis: DUCView

Manual methods of pyramid creation require a person to use a tool, such as DUCView, to perform the pyramid annotation. The DUCView pyramid annotation tool was created around 2005 by Sergey Sigelman at Columbia University [14].

DUCView has two main views: pyramid creation and peer annotation. The pyramid interface is shown in Figure 5.

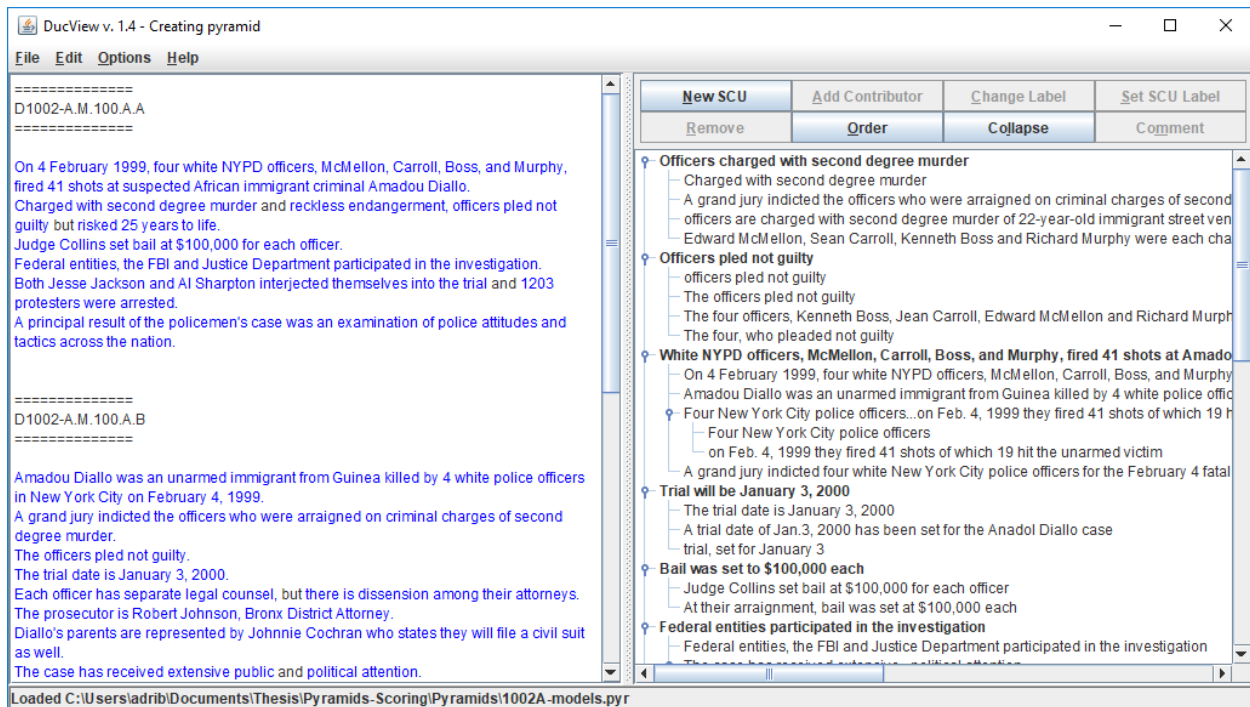


Figure 5. Pyramid creation in DUCView version 1.4.

The pyramid creation interface, shown in Figure 5, has two main text panes on the left and the right. The left text pane loads all of the gold-standard summaries at once, separated by a delimiter. The right text pane shows the pyramid, displayed in a tree format, in which there is an implicit root node that is not displayed, and then each of the next level nodes are SCU labels (in bold). The children of the SCU nodes are SCU contributors, with at most one SCU contributor per gold-standard summary. Users create SCUs and contributors by highlighting text on the left pane and using the buttons above the right pane to indicate the desired action. Finally, the pyramid can be saved in an XML format for further processing or later use.

The second main view involved in using DUCView is the peer annotation view, shown in Figure 6.

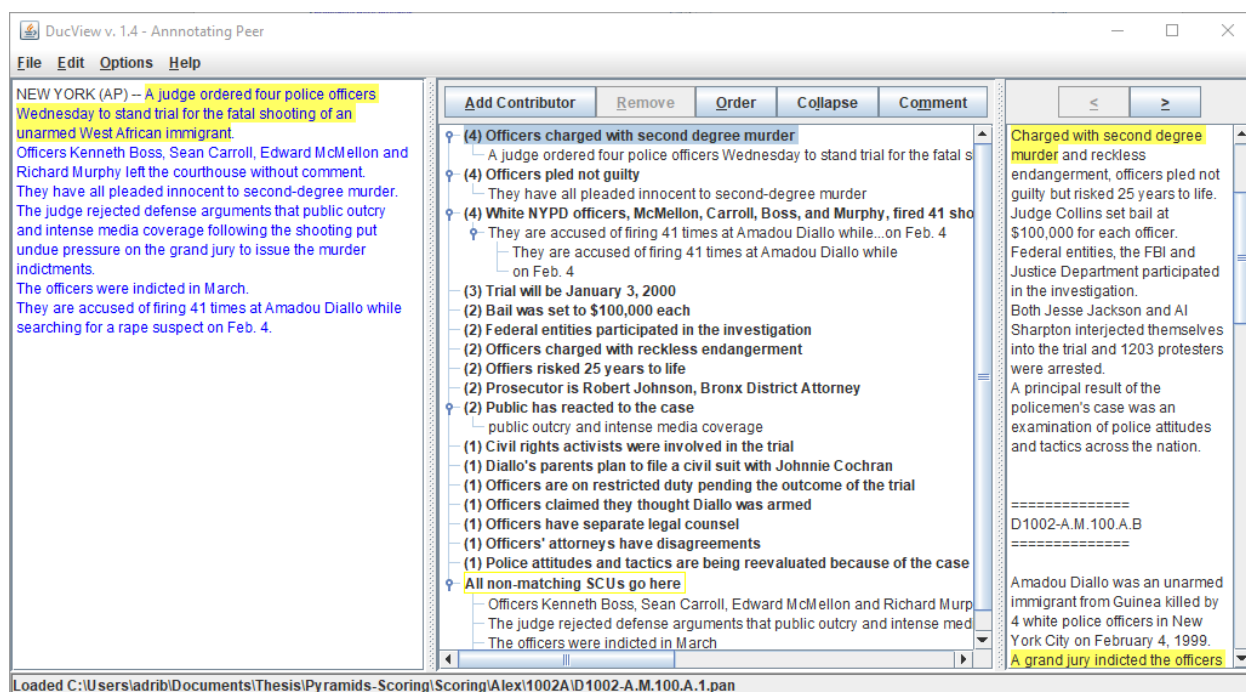
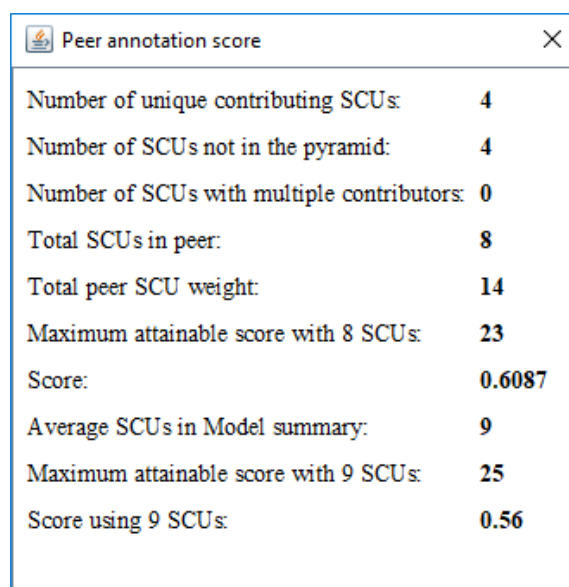


Figure 6. Peer annotation in DUCView version 1.4.

In the peer annotation view, there are three main text panes. The left pane displays the peer summary to be annotated and scored; the center pane displays the pyramid, with SCU contributors now representing contributors from the peer summary; and the right pane displays the gold-standard summaries. Contributors can be added in a similar manner to the pyramid annotation view, and the XML containing the pyramid can also be exported for further processing or later use.

DUCView can automatically calculate the quality score (simply denoted as “Score”) and the coverage score (listed as “Score using X SCUs”) given a completed peer annotation. The score window is shown in Figure 7.



Peer annotation score	
Number of unique contributing SCUs:	4
Number of SCUs not in the pyramid:	4
Number of SCUs with multiple contributors:	0
Total SCUs in peer:	8
Total peer SCU weight:	14
Maximum attainable score with 8 SCUs:	23
Score:	0.6087
Average SCUs in Model summary:	9
Maximum attainable score with 9 SCUs:	25
Score using 9 SCUs:	0.56

**Figure 7. Peer annotation score in DUCView version 1.4.**

DUCView enables annotators to manually create pyramids and peer annotations, which can be used to evaluate content mastery in student summaries. In addition, these manual annotations can be used to train models for automatically producing pyramids and peer

annotations. This application was involved in the creation of PyrEval [15], as discussed in Section 1.3.2.

### **1.3.2 Automatic Methods of Summary Content Analysis: PyrEval**

Even using DUCView, manual creation of pyramids can be time consuming, so a variety of automatic methods for generating pyramids and peer annotations have been created. Other methods of content evaluation exist that do not require the use of a pyramid at all; however, these methods have several disadvantages compared to pyramid-based methods. For instance, the most commonly used tool for evaluating machine generated summaries, called ROUGE, matches substrings between model and target summaries. It is not reliable for scoring single summaries, and does not provide information on the content in the summary, such as which ideas have been included and which ones have been left out [15].

PyrEval is a tool recently developed for automatic creation of pyramids and peer annotations. In contrast to ROUGE, because it is based on a pyramid content model, it is accurate for scoring single summaries, and can provide useful feedback, which makes it more applicable in an educational setting. The PyrEval model has four parameters, which were tuned by minimizing standard deviations for correlations with manual pyramids constructed in DUCView. This tool has been tested on various human and machine datasets, and shown to exhibit good correlation with scores calculated using manual pyramids [15].

## 1.4 Argument Annotation

While the process for content annotation of summaries has been well explored through tools such as DUCView and PyrEval, annotation of argumentative essays is relatively unexplored. The rest of Chapter 1 will discuss argument annotation.

### 1.4.1 Next Step in Content Annotation: Summaries versus Arguments

A system for content annotation of arguments would be of great interest to several communities. For instance, just as DUCView and PyrEval can be used to create standardized, wise crowd content rubrics to provide grading and feedback on summaries, a system for argument annotation would enable grading and feedback on argumentative essays. As discussed in Section 1.1, argumentative writing is a relatively poor style of writing for many children. An argument content annotation tool could help educators teach argumentative writing by providing thorough and timely feedback on student writing.

In addition, an argument content annotation tool could promote advances in the field of argumentation mining. To understand argumentation mining, it is helpful to first understand machine summarization. There are two primary classes of text summarizers: extractive and abstractive summarizers. Extractive summarizers produce summaries that only consist of information from the original text. In contrast, abstractive summarizers can produce summaries with information that may not be found in the original text. Both types share a series of common steps to produce a summary: they first create an intermediate representation of the original text that captures only the most important features, then rank sentences in order of importance based

on the intermediate representation, and finally select the optimal combination of sentences from the source to produce a new, short summary [16].

Argumentation mining is a process analogous to summarization: Lippi and Torroni defined it as “automatically extracting structured arguments from unstructured textual documents,” [17]. However, while summarization only considers details such as what people think about a topic, argumentation mining considers why people think the way they do. Summarization can roughly be accomplished without substantial regard to the logical order of information in the summary, but argumentation relies on its structure of premise, conclusion, and the details that link premise to conclusion. Therefore, it is generally a more challenging process to extract arguments than it is to extract parts of summaries. However, it is an important field to researchers for numerous reasons. For instance, argumentation mining could enhance legal databases by tracing the argumentation of parties in legal texts, it could improve medical decision-making about superior and inferior treatments, and it could be tremendously useful from a commercial perspective by allowing businesses to analyze reasons why internet users like or dislike their products [17], [18]. The aforementioned applications of argumentation mining include just a few of the many potential benefits.

In order to complete argument annotation, it is necessary to build a content model analogous to the pyramid method for summary content annotation. However, because the structure of an argument is relevant to its meaning, SCUs are not suitable for annotation of arguments. SCUs represent an idea, but do not convey a logical relationship between ideas as in an argument. Instead, annotation of arguments relies on aggregations of several logically connected ideas, called “argumentative discourse units.” Argumentative discourse units are discussed in Section 1.4.3. In addition, each argumentative discourse unit is generally made up of



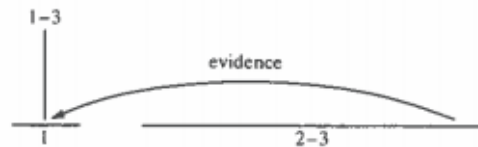
several “elementary discourse units” [19]. Elementary discourse units are discussed in Section 1.4.2.

### 1.4.2 Elementary Discourse Units

Elementary discourse units, or EDUs, arise from Rhetorical Structure Theory (RST), first described in 1988 by William C. Mann and Sandra A. Thompson. RST provides a descriptive framework for text that explains why text feels coherent. Mann and Thompson explained that coherent text feels unified because every part of the text has some reason for its existence, and the relationships among parts of the text can be represented in a hierarchal structure. Their 1988 paper defined a number of relation definitions, such as the simple “Evidence” relation depicted in Figure 8, an excerpt from this paper [20].

1. The program as published for calendar year 1980 really works.
2. In only a few minutes, I entered all the figures from my 1980 tax return and got a result which agreed with my hand calculations to the penny.

The RST diagram in Figure 2 shows Units 2-3 in an **Evidence** relation with Unit 1. They are provided to increase the reader’s belief in the claim expressed in Unit 1.



**Figure 8. Evidence relation in RST.**

In Figure 8, two sentences extracted from a letter are shown, labelled as 1 and 2. From these two sentences, three “Units” were derived. Units 2 and 3, both of which come from

sentence 2, serve as evidence for Unit 1 (from sentence 1), according to the diagram (called a “discourse tree”) shown in the figure. Mann and Thompson defined many more similar relations in their 1988 paper [20].

In Figure 8, the Units referred to by Mann and Thompson correspond to EDUs in the present discussion. Units, or EDUs, are the basic building blocks of a discourse tree in RST. The basic EDU is a clause. Some examples of clause-based EDUs are shown in Figure 1.9, extracted from Carlson and Marcu’s “Discourse Tagging Manual” [21]. In Figure 9, each of the three sentences numbered 9-11 contain two EDUs, each of which is contained within square brackets.

- (9) [Such trappings suggest a glorious past] [**but** give no hint of a troubled present.]<sub>wsj\_1302</sub>
- (10) [**Although** Mr. Freeman is retiring,] [he will continue to work as a consultant for American Express on a project basis.]<sub>wsj\_1317</sub>
- (11) [Share prices in Frankfurt closed narrowly mixed] [**after** Wall Street opened stronger.]<sub>wsj\_0374</sub>

**Figure 9. Sentences containing two clausal EDUs.**

The bolded words, called discourse cues, give hints as to where to break up the sentences to form EDUs. Sometimes, an EDU can consist of a phrasal expression rather than a clause, as shown in Figure 10, also taken from the “Discourse Tagging Manual” [21]. In Figure 10, the second EDU (denoted by the square brackets again), is not a clause, but the strong discourse cue “without” justifies splitting the phrase as an EDU.

(151) [Today, no one gets in or out of the restricted area] [**without** De Beers's stingy approval]<sub>wsj\_1121</sub>

**Figure 10. A sentence with a phrasal EDU.**

Therefore, the main difference between EDUs and SCUs is that EDUs have a grammatical context and are generally a full clausal unit, whereas SCUs have no grammatical context and may be as short as a single word, depending on the style of the annotator. EDUs are formed based on sentence structure, while SCUs are formed based on semantic meaning. In addition, SCUs are often made up of groups of contributors when the input is a group of documents; in contrast, for this annotation task, EDU contributors are not grouped. This is because the EDU contributors are not the final products of the annotation – the ADUs are the end goal of annotation [13], [21].

After all of the EDUs in an essay are annotated, it is possible to combine EDUs where necessary to form argumentative discourse units [19]. These argumentative discourse units can then be used to build a content model for argument annotation, as discussed in Section 1.4.3.

### **1.4.3 Argumentative Discourse Units**

Annotation of argumentative discourse units, or ADUs, makes up the final step of argument annotation. ADUs are the minimal units of analysis in an argumentative essay; while EDUs convey meaning, a single EDU does not convey a part of an argument itself. The EDUs in an argumentative essay must be joined with adjacent EDUs to form a composite ADU. ADUs can then be classified by various metrics, according to the application [19]. For instance, ADUs from several gold-standard essays can be used to build a content model, analogous to an SCU

pyramid, for scoring peer essays. In argumentation mining, ADUs can be classified, put into a graph or tree structure to show the relationships among different ADUs in the original text, and selected by some algorithm to produce an extractive argumentation, just as an extractive summarizer would select relevant sentences to form a new summary [22].

#### **1.4.4 Need for a Tool for Manual Content Annotation of Argumentative Writing**

As discussed throughout Section 1.4, annotation of argumentative writing would have applications in fields such as education and argumentation mining. In order to facilitate annotation of this kind, software tools for EDU and ADU content annotation would be useful. In this paper, I discuss the development of a new tool, called SEAVIEW, for EDU annotation. This tool was created based on DUCVIEW and will primarily be used to annotate content in essays with a specific format. In particular, this special format includes a summary in the header of the essay, followed by a body that makes an argument. Therefore, the first portion of the essays will be annotated for SCUs, using a modified version of DUCVIEW, as discussed in Chapter 2. The second half of the essays, which contains the argumentative portion of the writing, will be annotated for EDUs in SEAVIEW, as discussed in Chapter 3. This annotation will then be passed to ADU annotators to complete the argument annotation, and score the essays. Since the essays contain both summaries and arguments, this tool will also be used to study the relationship between summaries and arguments – namely, to explore the types of relationships that exist between SCUs and EDUs. This will provide new insights into argumentation skills and may facilitate argumentation mining.

## Chapter 2

### DUCView: A Tool for Summary Content Annotation

The DUCView pyramid annotation tool was created around 2005 by Sergey Sigelman at Columbia University [14]. However, the original source code for the tool was lost. For this project, several brief modifications to the tool were required for compatibility, so it was necessary to decompile the classes in the JAR file to recover the original source code. In addition, several modifications were made to the source code based on user feedback collected after using the tool. The process of decompilation and restoration of the original code is described in Section 2.1.

#### 2.1 Decompilation of the DUCView JAR File

Saptarashmi Bandyopadhyay, from the Penn State Natural Language Processing Lab, was able to decompile the DUCView JAR file using two decompilers. Decompilation was first attempted using only JD-CORE [23], part of the Java Decompiler Project – a set of tools for decompilation and analysis of Java 5 byte code. Apache Maven version 3.6.0 [24] was used to handle JD-CORE dependencies, and the decompilation was completed in an Ubuntu 16.04 operating system. The decompilation was successfully completed with JD-CORE, yielding a ZIP file containing Java source code files from the DUCView JAR file. However, attempting to compile the recovered source code revealed 444 compilation time errors. 378 of these errors were access restriction errors due to use of Swing to create the DUCView GUI. These errors were fixed by adding `**/*` as a type access rule in the libraries section of the Java build path for the project in Eclipse. However, 66 errors remained, caused by a variety of issues, such as

scrambled code or multiple instantiation of variables. In order to resolve the remaining errors, the decompilation results from JD-CORE were cross-analyzed using the output from a second Java decompiler, Luyten version 0.5.3 [25], in a Fedora 20 operating system. Cross-analysis of the two decompiler outputs yielded functional, yet messy, source code, due to the lack of code documentation available and the nature of the decompiler outputs [26].

## 2.2 Analysis of Decompiled DUCView Source Code

The decompiled DUCView source code initially contained nine Java source code files.

The purpose of each of the decompiled source code files is described in Table 1.

**Table 1. Descriptions of DUCView source code files.**

File Name	Description
DucView.java	<ul style="list-style-type: none"> <li>• Contains the main function and calls all the other classes except DocumentRenderer.java</li> <li>• Creates the application's GUI using the Java Swing library</li> <li>• Parses and creates XML files, defines the XML DTD</li> <li>• Calculates peer summary scores</li> </ul>
SCU.java	<ul style="list-style-type: none"> <li>• Defines an SCU with an ID, label, and a comment</li> </ul>

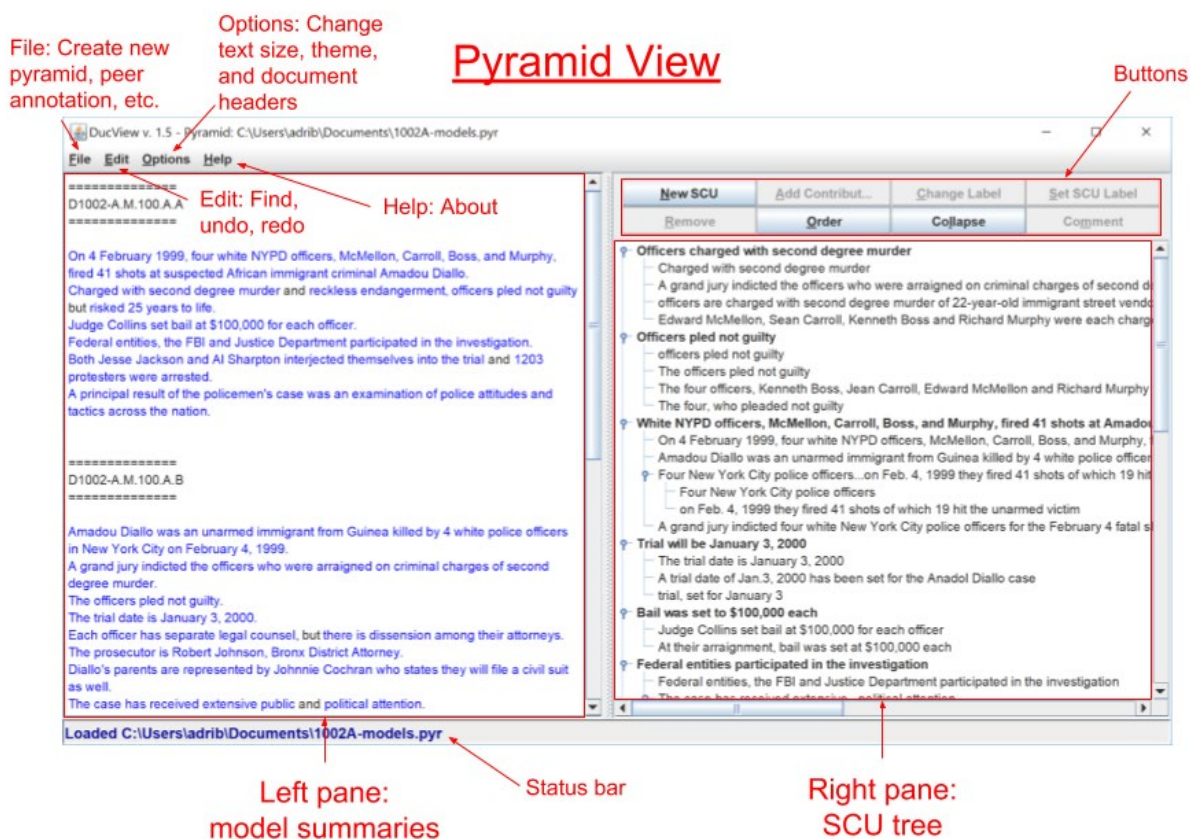
SCUContributor.java	<ul style="list-style-type: none"> <li>• Defines an SCU contributor from a summary</li> <li>• Includes a list of the contributor's SCUContributorParts, which may be non-adjacent in the text</li> </ul>
SCUContributorPart.java	<ul style="list-style-type: none"> <li>• Defines a part of an SCU contributor</li> <li>• Includes the starting and ending indices of the SCU contributor in the summary, as well as the text</li> </ul>
SCUTextPane.java	<ul style="list-style-type: none"> <li>• Defines the text panes used by DUCView to display essay text</li> <li>• Handles interacting with text, such as selecting and highlighting text</li> </ul>
SCUTree.java	<ul style="list-style-type: none"> <li>• Defines a tree for the SCUs in a pyramid or a peer annotation</li> <li>• Includes functions for manipulation of SCUs in the pyramid, such as obtaining, ordering, comparing, selecting, highlighting, dragging, scrolling, and dropping SCUs</li> </ul>
SearchDialog.java	<ul style="list-style-type: none"> <li>• Enables searching of text and SCU labels</li> </ul>
ScoreDialog.java	<ul style="list-style-type: none"> <li>• Displays the HTML of a summary's score, generated in the DucView class</li> </ul>
DocumentRenderer.java	<ul style="list-style-type: none"> <li>• Provides support for printing pyramids</li> </ul>

### 2.3 Control Flow of DUCView

DUCView's control flow is predominantly determined in the `DucView.java` class. This class contains the main function which instantiates the Swing-based GUI. The GUI contains a variety of buttons and menus that are primarily used to interact with DUCView. To create a pyramid, a user must select a text file containing several gold-standard summaries, each of which must be separated by a text delimiter. The delimiter must be specified as a RegEx expression in the Options menu to let DUCView know the number of gold-standard summaries that are contained with the text file. This allows DUCView to accurately calculate the coverage score of a peer summary, and enables some error checking of SCUs, since it defines the bounds of each summary. Therefore, DUCView can detect and prevent users from entering multiple SCU contributors from the same gold-standard summary.

Once the text file containing the gold-standard summaries has been loaded into DUCView, the gold-standard summaries will be displayed on the left, as shown in Figure 11.



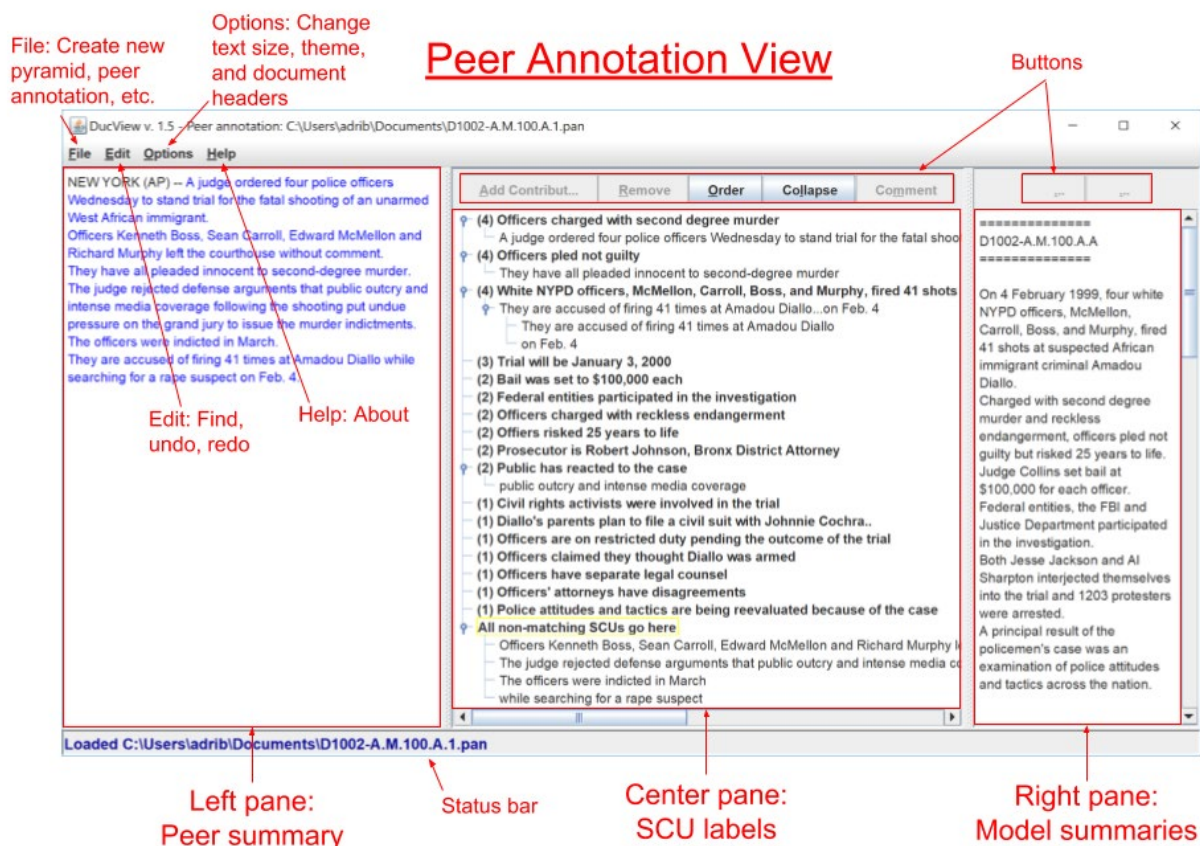


**Figure 11. Pyramid creation view in DucView version 1.5.**

Initially, during pyramid creation, an empty pyramid will appear on the right, to which the user must use the buttons panel to add SCUs and their contributors to complete the pyramid. The panel also contains several other useful buttons. The “Change Label” button enables a user to set the SCU label, which is useful for clarifying the context of shorter SCUs such as dates. The “Set SCU Label” button automatically changes the SCU label to match the currently selected contributor. The “Remove” button is used to remove SCUs or individual contributors. The order button automatically orders the pyramid tree by the SCU weights, but users can also create a custom ordering by dragging SCUs up or down in the tree. The “Expand/Collapse” button enables users to expand or collapse the pyramid tree. Finally, the comment button enables users

to comment SCU labels or contributors to record important notes or other information. Saving the pyramid creates a \*.pyr file, which contains an XML representation of the gold-standard summaries and the pyramid tree, and can be loaded to view the pyramid later.

To start a new peer annotation, the user must have already created a pyramid using several gold-standard summaries. Starting a new peer annotation will change the view in DUCView to the three-pane view shown in Figure 12.



**Figure 12. Peer annotation view in DUCView version 1.5.**

Now the left pane contains the peer summary, the center pane contains SCU labels from the pyramid (along with the SCU weights), and the right pane contains the gold-standard

summaries. Users must categorize the text in the peer summary into matching SCU labels to annotate the peer summary. If text from the peer summary does not match any of the SCU labels in the pyramid, users should add the text to the “All non-matching SCUs go here” label. After annotating the peer summary, users can view the peer score, or save the peer annotation as a \*.pan file in an XML format that includes the gold-standard summaries, the pyramid, the peer summary, and the matching and non-matching SCUs.

## 2.4 Modifications to DUCView

Although DUCView version 1.4 generally performed well for annotation tasks, several modifications to the tool were made based on user feedback to create version 1.5. The source code was also cleaned up and made available online at the Penn State Natural Processing Lab’s GitHub page (<https://github.com/psunlpgroup/DucView-1.5>). The revisions made to the DUCView source code for version 1.5 are described in Table 2.

**Table 2. DUCView version 1.5 changes.**

<b>Task</b>	<b>Description</b>
Update help	Previously, the help section only included original programmer’s email and the year the tool was updated. The help section was updated with PSU NLP Lab information and link to DUCView help site
Enable larger font sizes	Enabled larger font sizes and fixed spacing issues with large fonts on Mac.

Add file loading message and enhance status bar visibility	Enlarged status bar font and changed color for visibility; changed the window title to display the current file.
Export score result in XML	Included the score of a peer annotation in the XML so it can be processed without entering the tool to manually check the score for each *.pan file. Also updated the score window to include the comprehensive score, and renamed several score categories to reflect the proper nomenclature.
Read in a folder of *.txt	Allowed users to load a folder of text files and let DUCView automatically generate delimiters for gold-standard summaries, instead of the user having to concatenate the summaries themselves.
Code documentation	Commented code and created a README.
Remove unused features	Remove print and auto-annotate features for simplicity, since these features were unused by previous annotators.
Add a second delimiter for summary/argument format essays	Added a second RegEx delimiter so that users can distinguish between summary and argument portions of essays when making pyramids.

The version 1.4 and version 1.5 peer annotation score windows in DUCView are shown on the left and right sides of Figure 13, respectively.

Peer annotation score (Version 1.4)		Peer annotation score (Version 1.5)	
Number of unique contributing SCUs:	4	Total SCUs in peer:	2
Number of SCUs not in the pyramid:	4	Number of unique contributing SCUs:	1
Number of SCUs with multiple contributors:	0	Number of SCUs not in the pyramid:	1
Total SCUs in peer:	8	Total peer SCU weight:	4
Total peer SCU weight:	14	Maximum attainable score with 2 SCUs:	8
Maximum attainable score with 8 SCUs:	23	<i>Quality score:</i>	0.5
Score:	0.6087	Average SCUs in Model summary:	8
Average SCUs in Model summary:	9	Maximum attainable score with 8 SCUs:	23
Maximum attainable score with 9 SCUs:	25	<i>Coverage score:</i>	0.1739
Score using 9 SCUs:	0.56	<i>Comprehensive score:</i>	0.2581

**Figure 13. Comparison of version 1.4 and 1.5 score windows**

The DUCView version 1.5 score window primarily differs from the version 1.4 window in that it now includes the comprehensive score and has renamed the “Score” and “Score using X SCUs” categories to the quality score and coverage score respectively. Since these are the score categories with which most users are primarily concerned, the font is italicized. In addition, several minor categories have been moved or removed entirely. All of the items in this window can be exported in an XML format in version 1.5 for processing of scores without having to open DUCView.

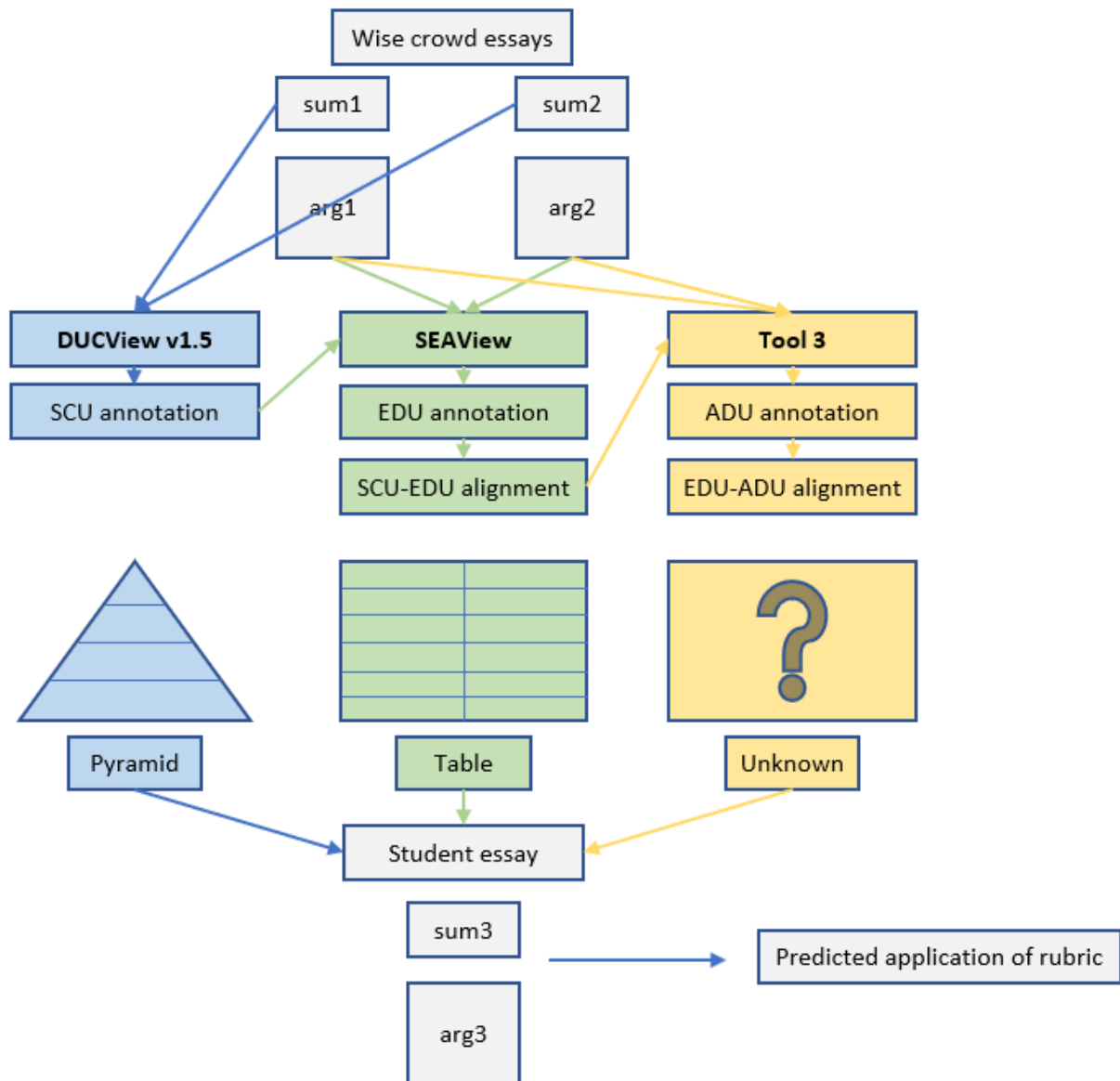
## Chapter 3

### SEAVIEW: A New Tool for Argument Content Annotation

After modifying the DUCView source code with several improvements, and for compatibility with the new annotation task, design of the new tool was started. An introduction for the justification for a new tool for argument content annotation was presented in Section 1.4.4. This new tool, called SEAVIEW, was designed for EDU annotation of essays written in a specific format. The essay format requires writers to read about a topic, and then write a two-part essay. The first section of the essay contains a short summary about the topic, and in the second part of the essay, the writers must make an argument about the topic. Therefore, the task to annotate the content in these types of essays requires annotation of SCUs, EDUs, and ADUs. While DUCView is largely suitable for SCU annotation of these essays, no tool existed previously for the manual EDU or ADU annotation portions of these essays. SEAVIEW enables annotators to complete the first step of the essay annotation process by annotating EDUs in the argument portion of the essays, and aligning those EDUs with SCUs annotated in DUCView.

#### 3.1 Design of SEAVIEW

SEAVIEW was designed to represent one step in a three-step process for annotation of these essays. A schematic detailing the three main steps is shown in Figure 14.



**Figure 14. Essay annotation schematic**

In Stage 1 of Figure 14, the wise crowd or gold-standard essays, are input into DUCView version 1.5 for SCU annotation in order to build a pyramid content model. Each essay, represented by gray rectangles stacked on top of each other, consists of two parts, “sum1” and “arg1,” denoting the summary and argument portions of each gold-standard essay, as well as the

number of the essay. Only the sum portions of the essays are input into DUCView version 1.5, since DUCView can only annotate the summary portions of the essays.

Then, in Stage 2, the arg portions of the gold-standard essays are input into SEAView to create a “SEA (SCU-EDU alignment) table,” which contains all of the EDUs found in the gold-standard essays. In addition, if the EDU from the argument essay portion has an SCU from the DUCView-generated pyramid that has a similar meaning, the SCU is added to the row in the SEA table. An EDU may or may not have a corresponding SCU from the summary. The SEA table can then be used to analyze relationships between the SCUs and the EDUs, such as the effect of SCU weight on its frequency in the SEA table. Finally, in Stage 3, in a tool to be built in the future, a correspondence between EDUs and ADUs is annotated. This correspondence, plus the EDU table and the SCU pyramid, can finally be used in tandem to score student essays.

### **3.2 Implementation of SEAView Design**

The essay annotation workflow using SEAView and DUCView is described in Table 4, based on the design schematic shown in Figure 14.



**Table 3. Essay annotation workflow using DUCView and SEAView.**

Step #	Tool Used	Task Description	Input File Type(s)	Output File Type
1	DUCView	Annotate SCUs in gold-standard essays to create a pyramid	*.txt	*.pyr (pyramid)
2	DUCView	Annotate SCUs in peer essays	*.pyr (for pyramid) *.txt (for peer essay)	*.pan (peer annotation)
3	SEAView	Annotate EDUs and align with SCUs in gold-standard essays to create a table	*.pyr	*.sea (SEA table)
4	SEAView	Annotate EDUs in peer essays	*.sea (for table) *.pan (for peer essay)	*.sep (SEA peer annotation)

The overall goal of this annotation task is to create a table with EDUs and SCUs, and a pyramid with SCUs, from the model and peer summaries. In short, SCUs are annotated in DUCView for the summary portions of both the gold-standard essays and the peer essays, yielding a pyramid and a peer annotation, respectively. Next, EDUs are annotated in the argument portion of the essays, and aligned with SCUs from the pyramid to create a table, called an SCU-EDU Alignment table, or a SEA table. The output of this step is an XML file containing the SEA table, as well as the pyramid and gold-standard essays text, called a \*.sea file. Then,

once SEAVIEW has a loaded SEA table, it can be used to annotate EDUs in peer essays. The input peer essay comes from a \*.pan (peer annotation) file completed in DUCVIEW, which will contain the peer summary, as well as the completed peer annotation. The final result is a SEA Peer annotation, or SEP annotation (\*.sep file). This file contains XML with the pyramid and summary peer annotation, as well as the SEA table and SEP annotation.

With the goal of facilitating this workflow, SEAVIEW was created by extending and refactoring the DUCVIEW source code. Since DUCVIEW already included some useful GUI elements and pyramid tree tools, it provided a helpful starting point for SEAVIEW. However, while DUCVIEW's control flow primarily depended on the DucView.java class, SEAVIEW's control flow was passed primarily to the SEATable.java class. SEAVIEW's classes and their primary functions are listed in Table 4. The source code is also available online at the Penn State Natural Processing Lab's GitHub page (<https://github.com/psunlpgroup/SEAVIEW>).

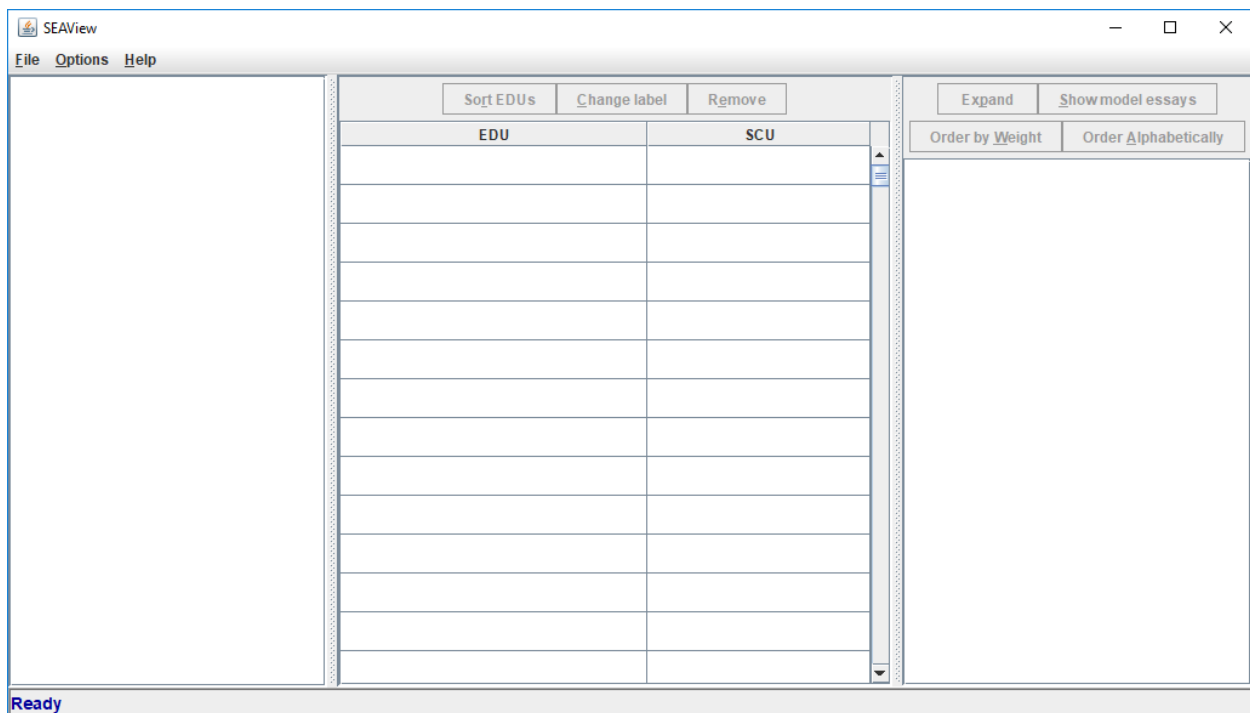
**Table 4. Primary functions of SEAVIEW source code files.**

File Name	Description
SEAVIEW.java	<ul style="list-style-type: none"> <li>• Contains the main function</li> <li>• Creates the application's GUI using the Java Swing library</li> <li>• Parses and creates XML files</li> <li>• Handles loading and saving of files</li> <li>• Error handling</li> </ul>
SEATable.java	<ul style="list-style-type: none"> <li>• The SEA table</li> <li>• Handles interactions with the table, such as sorting and drag and drop</li> </ul>

Unit.java	<ul style="list-style-type: none"> <li>• Defines an SCU/ EDU, with an ID, label, and a comment</li> </ul>
UnitContributor.java	<ul style="list-style-type: none"> <li>• Defines an SCU/EDU contributor - a portion of the text from a summary that makes up an SCU/EDU</li> <li>• Includes a list of the contributor's SCUContributorParts, which may be non-adjacent in the text</li> </ul>
UnitContributorPart.java	<ul style="list-style-type: none"> <li>• Defines a part of an SCU/EDU contributor</li> <li>• Includes the starting and ending indices of the SCU/EDU contributor in the summary, as well as the text</li> </ul>
SEAViewTextPane.java	<ul style="list-style-type: none"> <li>• Defines the left pane of SEAView that contains the essay text</li> <li>• Includes functions for displaying and selecting text</li> </ul>
UnitTree.java	<ul style="list-style-type: none"> <li>• Defines a tree for the SCUs in a pyramid or a peer annotation</li> <li>• Also defines the tree format for EDUs in the EDU table</li> </ul>
EssayAndSummaryNum.java	<ul style="list-style-type: none"> <li>• A helper class for finding a text selection's original essay number and whether it came from a summary</li> </ul>

### 3.3 Using SEAVIEW

While DUCView contains two separate views depending on whether the user is annotating gold-standard summaries or a peer summary, SEAVIEW has simplified the annotation process by using only one main view with three panes. The SEAVIEW ready screen, which is shown when no files have been loaded into the tool, is depicted in Figure 15.

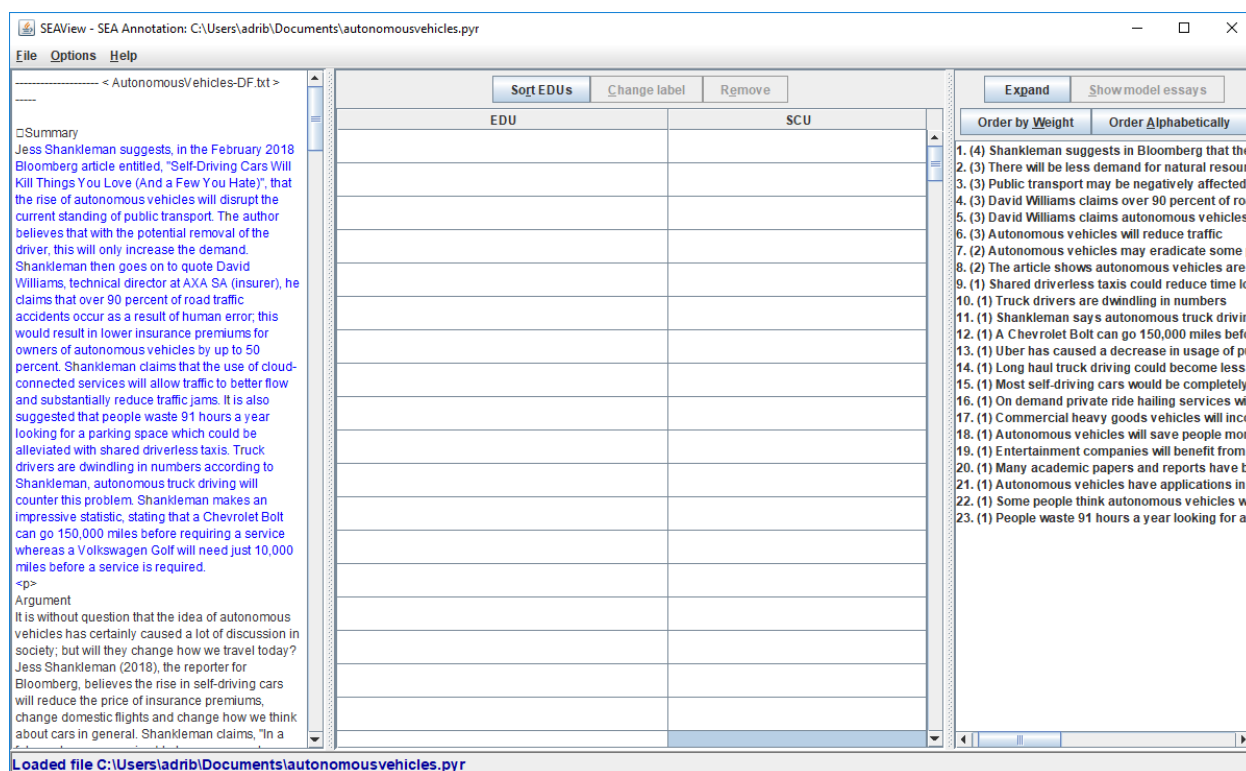


**Figure 15. SEAVIEW ready screen**

The left pane in SEAVIEW is used to load the essay text, the center pane is used to display the SEA table, and the right pane is used to display the pyramid of SCUs that was created in DUCView during Step 1 of Table 3. SEAVIEW also includes a status bar at the bottom, with the text “Ready” in Figure 15. This status bar displays messages such as errors and annotation steps during use. The menu bar along the top, below the SEAVIEW title, contains three submenus. The

File submenu is primarily used to load and save files. The Options submenu contains customization options for font sizes and themes, as well as options for specifying essay delimiters. The Help submenu contains contact information for the Penn State Natural Language Processing Lab.

After the pyramid of SCUs has been created in DUCView, annotation in SEAVIEW begins by loading the pyramid file into SEAVIEW. This represents the beginning of Step 3 in Table 3. Upon loading an essay, SEAVIEW shows the view in Figure 16.



**Figure 16. Pyramid loaded in SEAVIEW.**

In the view shown in Figure 16, the gold-standard essay text is now shown on the left pane. Text that has been annotated is blue to show the user that the text unit has been accounted

for in the model. The pyramid is shown on the left. The pyramid can be expanded to view SCU contributors, but is collapsed by default, since the contributors are excessive information for the EDU annotation task. Each SCU in the pyramid has the following elements: an index, a weight (in parentheses), and a label. The index is a temporary index assigned based on the SCU's order within the pyramid, and may change depending on the way the pyramid is sorted. The pyramid can be sorted by weight (since SCUs of high weight are hypothesized to appear more frequently in the table) or by alphabetical order (to enable users to find SCUs based on their content). In addition, selecting a SCU in the pyramid, or selecting annotated text on the left pane, highlights the text that has been annotated and the pyramid SCU that corresponds to the text, as shown in Figure 17.

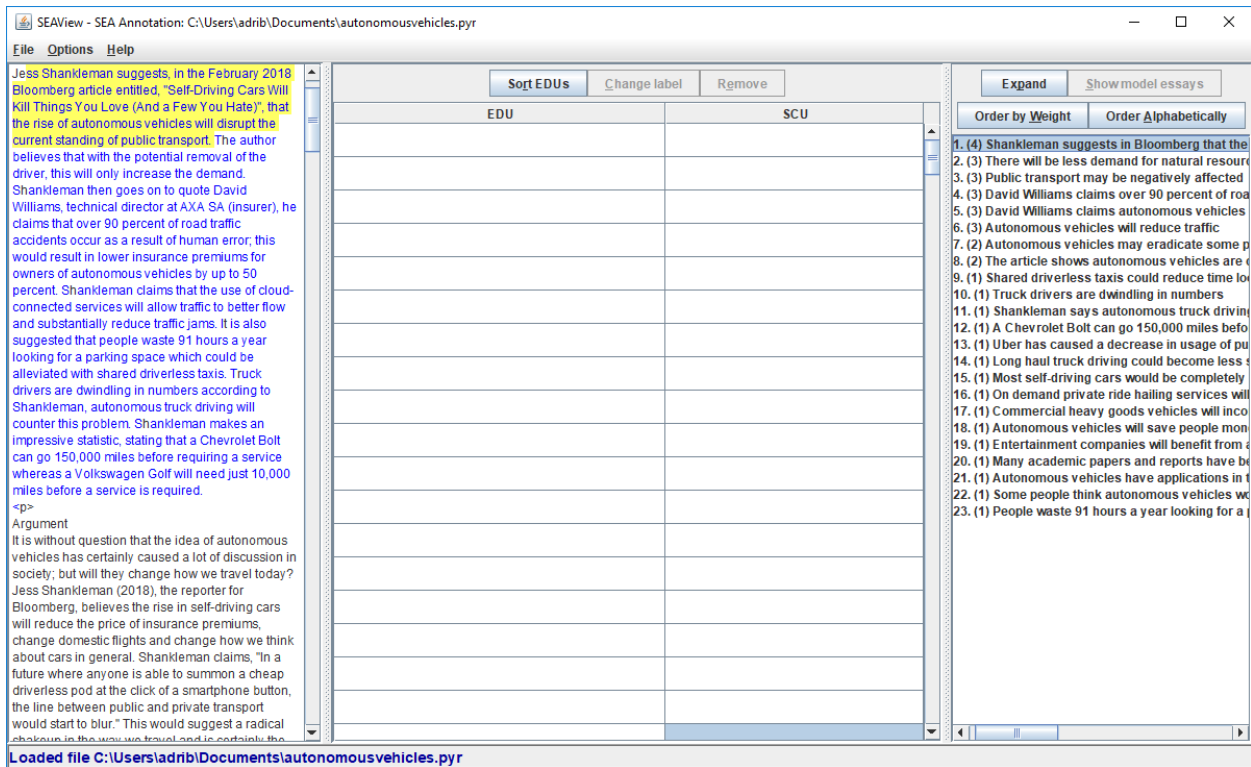
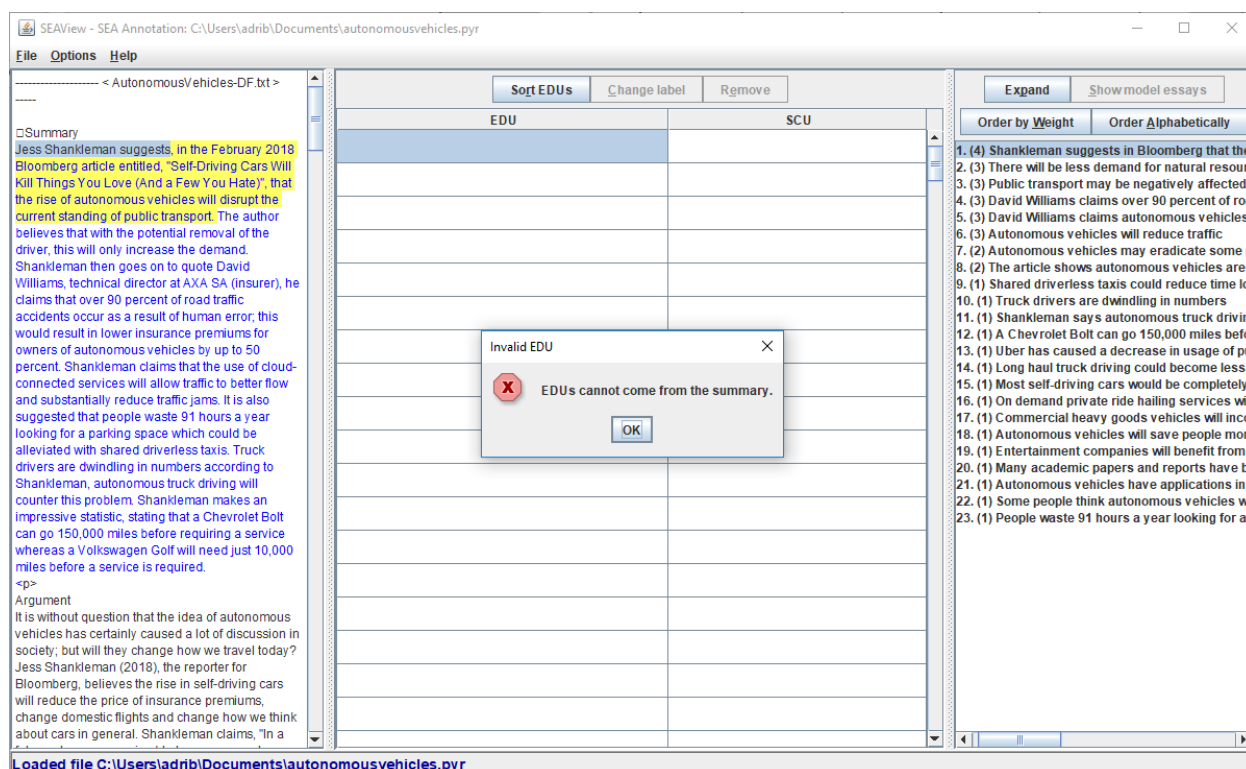


Figure 17. Selected SCU in SEAVIEW.

The final step for users to complete before beginning annotation of the argument text is to specify the regular expressions denoting the document delimiters (if they have not already been specified in DUCView). One delimiter separates the summary from the argument, and the other delimiter separates gold-standard essays from each other. This must be completed prior to annotation so SEAView can perform error checking on EDU selections from the text. An example error is shown in Figure 18.



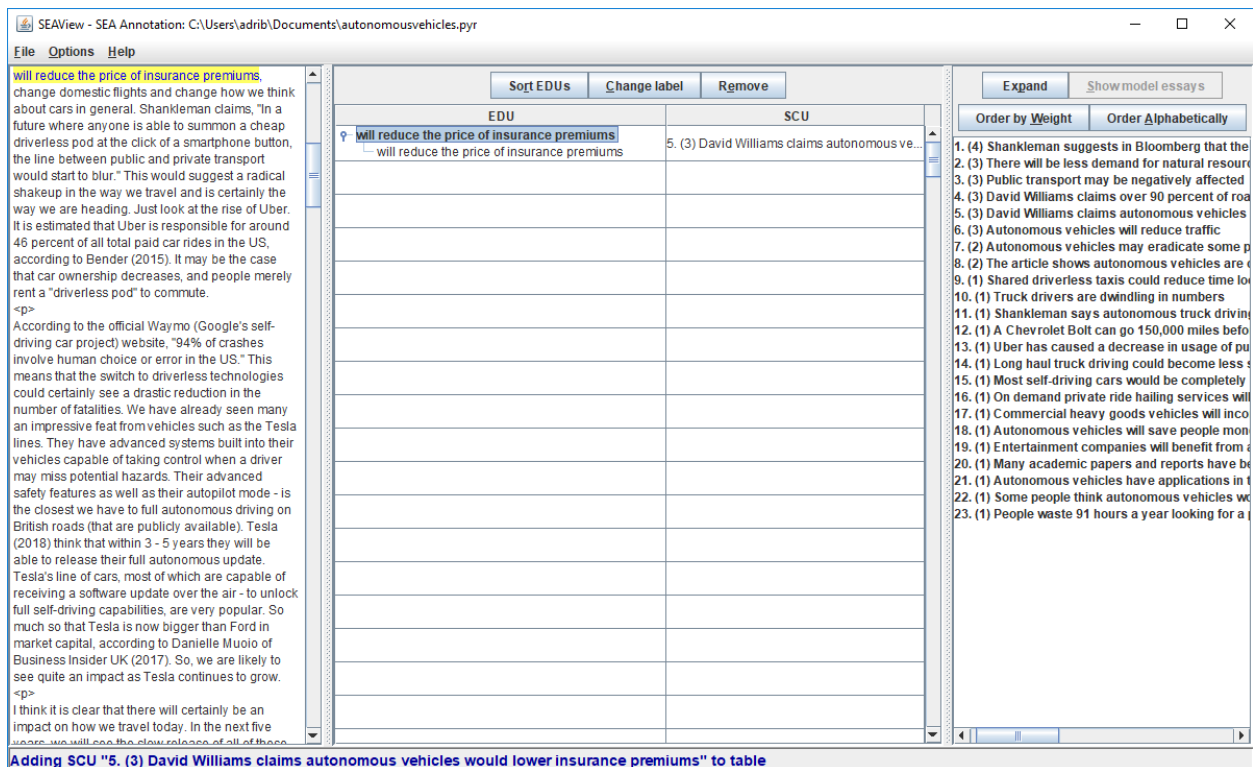
**Figure 18. An error during EDU creation in SEAView.**

In Figure 18, a user has attempted to annotate an EDU from the summary of an essay. EDUs may only come from the argument portion of the essay, so SEAView has displayed the error message “Invalid EDU” to prevent annotation errors. The document delimiters also are

used to prevent users from adding EDU contributors from different essays and in sorting the SEA table.

After specifying the delimiters, SEA annotation may begin. Users may add an EDU to the table by highlighting text on the left pane and dragging the text into the desired table cell.

Similarly, users may add a SCU to the table by highlighting an SCU node in the pyramid on the right pane and dragging the SCU into the table. An annotated EDU and a corresponding SCU are shown in Figure 19.



**Figure 19. An EDU and SCU annotated in SEAVIEW**

In Figure 19, the EDU “will reduce the price of insurance premiums” was first annotated from the essay text, and then the corresponding SCU “David Williams claims autonomous



vehicles would lower insurance premiums” is added to the same table row. This indicates that the EDU and SCU have similar meanings. The table displays the EDU with its contributor below it, while the SCU only appears with its index, weight, and part of its label. The full label is not displayed to reduce the amount of information in the table, since the user can easily reference the SCU in the pyramid by its index. The EDU label is highlighted in Figure 19, which highlights the source text on the left pane. In addition, if the label is selected, a user may remove the EDU from the table or change its label using the buttons above the table. Changing labels is useful since EDUs are not complete sentences, and therefore can lack context. In the EDU “will reduce the price of insurance premiums,” it is unclear what will reduce the price of insurance premiums based on the label. Therefore, a user could change the label to the label shown in Figure 20.

The screenshot shows the SEAVIEW application window. The title bar reads "SEAVIEW - SEA Annotation: C:\Users\adrib\Documents\autonomousvehicles.pyr". The interface is divided into several sections:

- Left Pane:** Contains source text for an EDU. The text discusses autonomous vehicles and insurance premiums, mentioning sources like Shankleman, Bender (2015), and Tesla.
- Table:** A table with two columns: "EDU" and "SCU". The first row contains the EDU "Autonomous vehicles will reduce insurance premiums" and the SCU "5. (3) David Williams claims autonomous v...". Above the table are buttons for "Sort EDUs", "Change label", and "Remove".
- Right Pane:** Contains a list of items, each with a weight and a brief description. The list starts with "1. (4) Shankleman suggests in Bloomberg that..." and ends with "23. (1) People waste 91 hours a year looking fo...".
- Top Bar:** Includes a menu bar ("File", "Options", "Help") and buttons for "Expand" and "Show model essays".
- Bottom Bar:** A status bar that reads "Adding SCU '5. (3) David Williams claims autonomous vehicles would lower insurance premiums' to table".

**Figure 20. Changed EDU label in SEAVIEW.**

In Figure 20, the EDU label has been changed from “will reduce the price of insurance premiums” to “Autonomous vehicles will reduce the price of insurance premiums.” This adds context to the EDU when it is not clear from the initial text.

EDUs may or may not have a corresponding SCU. However, an SCU may not appear in the table without a corresponding EDU. If a user attempts to add an SCU without an EDU, SEAVIEW will highlight the row to show the user that an EDU should be added. An invalid row is depicted in Figure 21.

The screenshot shows the SEAVIEW application window. The main area is a table with two columns: EDU and SCU. The table is sorted by weight. A row is highlighted in blue, and a message box is overlaid on it: "Each SCU must have a corresponding EDU". The table contains the following data:

EDU	SCU
Autonomous vehicles will reduce insurance premiums — will reduce the price of insurance premiums	5. (3) David Williams claims autonomous v...
change domestic flights — change domestic flights	
change how we think about cars in general — change how we think about cars in general	
"94% of crashes involve human choice or error in the — "94% of crashes involve human choice or error in the	4. (3) David Williams claims over 90 perce...
autonomous vehicles will change the way we travel to — autonomous vehicles will change the way we travel to	1. (4) Shankleman suggests in Bloomberg ...
the line between public and private transport would st — the line between public and private transport would s	3. (3) Public transport may be negatively af...
it could not be any clearer — it could not be any clearer	
more than 90% of all road accidents came down to hu — more than 90% of all road accidents came down to h	4. (3) David Williams claims over 90 perce...
Ashley (2017) fears that with the rise of autonomous v — Ashley (2017) fears that with the rise of autonomous	3. (3) Public transport may be negatively af...
Flat-pack furniture giants IKEA have opened SPACE1 — Flat-pack furniture giants IKEA have opened SPACE	
It may be the case that car ownership decreases — it may be the case that car ownership decreases	
people merely rent a "driverless pod" to commute — people merely rent a "driverless pod" to commute	
people using services such as Uber — people using services such as Uber	13. (1) Uber has caused a decrease in usa...
	1. (4) Shankleman suggests in Bloomberg ...

At the bottom of the window, a status bar reads: "Adding SCU "1. (4) Shankleman suggests in Bloomberg that the rise of autonomous vehicles will disrupt the current standing of public transport" to table".

**Figure 21. A SCU without a corresponding EDU in SEAVIEW.**

In Figure 21, several EDUs and SCUs have now been added to the SEA table. Some EDUs, such as the ones in the first and fourth rows of the table, have a corresponding SCU,

while others do not, such as the EDUs in the second and third rows. In the last row of the table, the user is prompted to add an EDU to the row because a SCU has been added without an EDU. Once many EDUs have been added to the SEA table, it may become difficult to understand the table. To organize the table, users can click the “Sort EDUs” button, which orders the table based on the order in which the EDUs occur in the gold-standard summaries. An example of an ordered SEA table is shown in Figure 20.

SEAVIEW - SEA Annotation: C:\Users\adrib\Documents\autonomousvehicles.pyr

File Options Help

public transport services. Without the need to pay a human to drive the cars, the price of the services will decrease and even more people would switch from public transport. Also, as most collisions are due to human error, costs of insurance for self-driving cars could fall by up to 50%. For people in some professions, such as parking attendants and driving instructors, self-driving cars could mean the eradication of their lines of work while jobs such as long haul truck driving could become less stressful. As most self-driving cars would be completely electric, this would mean that there would be far less demand for natural resources such as the crude oil used to make petroleum and diesel.

<p>  
Argument  
In my opinion, it could not be any clearer.  
Autonomous vehicles will change the way that we travel. This can be seen already in cases such as the Erica autonomous bus in Spain. Although still in the early stages of being tested on limited routes, this innovative vehicle has transported over 4,600 passengers. Currently this vehicle has room for 11 passengers and an attendant, to help in emergencies and to advise passengers. The major upside of this vehicle is that it produces less emissions as it is electrically powered with capability to run for 14 hours. It is also fully air-conditioned and can cater for passengers in wheelchairs, despite its small size. This shows such promise for the future considering it is a project in its early stages. However, it does have some major issues. The first of these major issues makes this mode of transport impractical. Currently, it requires 2 days of preparation "as it has to record the route to be driven in detail using GPS" (Solana, 2018). For a bus this shouldn't be too much of a problem as they general go on the same routes, however it is clear that this technology wouldn't be able to be used in taxi

Sort EDUs Change label Remove

EDU	SCU
1.1. Autonomous vehicles will reduce insurance premiums — will reduce the price of insurance premiums	5. (3) David Williams claims autonomous v...
1.2. change domestic flights — change domestic flights	
1.3. change how we think about cars in general — change how we think about cars in general	
1.4. the line between public and private transport would — the line between public and private transport would	3. (3) Public transport may be negatively af...
1.5. It may be the case that car ownership decreases — It may be the case that car ownership decreases	
1.6. people merely rent a "driverless pod" to commute — people merely rent a "driverless pod" to commute	
1.7. "94% of crashes involve human choice or error in — "94% of crashes involve human choice or error in the	4. (3) David Williams claims over 90 perce...
2.1. it could not be any clearer — it could not be any clearer	
2.2. Autonomous vehicles will change the way that we — Autonomous vehicles will change the way that we tra	1. (4) Shankleman suggests in Bloomberg ...
2.3. Flat-pack furniture giants IKEA have opened SPA — Flat-pack furniture giants IKEA have opened SPACE	
2.4. autonomous vehicles will change the way we trav — autonomous vehicles will change the way we travel	1. (4) Shankleman suggests in Bloomberg ...
3.1. more than 90% of all road accidents came down to h — more than 90% of all road accidents came down to h	4. (3) David Williams claims over 90 perce...
4.1. Ashley (2017) fears that with the rise of autonom — Ashley (2017) fears that with the rise of autonom	3. (3) Public transport may be negatively af...
4.2. people using services such as Uber — people using services such as Uber	13. (1) Uber has caused a decrease in usa...

Expand Show model essays

Order by Weight Order Alphabetically

1. (4) Shankleman suggests in Bloomberg that  
2. (3) There will be less demand for natural res  
3. (3) Public transport may be negatively affect  
4. (3) David Williams claims over 90 percent of  
5. (3) David Williams claims autonomous vehic  
6. (3) Autonomous vehicles will reduce traffic  
7. (2) Autonomous vehicles may eradicate son  
8. (2) The article shows autonomous vehicles a  
9. (1) Shared driverless taxis could reduce tim  
10. (1) Truck drivers are dwindling in numbers  
11. (1) Shankleman says autonomous truck dri  
12. (1) A Chevrolet Bolt can go 150,000 miles b  
13. (1) Uber has caused a decrease in usage o  
14. (1) Long haul truck driving could become le  
15. (1) Most self-driving cars would be complet  
16. (1) On demand private ride hailing services  
17. (1) Commercial heavy goods vehicles will i  
18. (1) Autonomous vehicles will save people n  
19. (1) Entertainment companies will benefit fr  
20. (1) Many academic papers and reports hav  
21. (1) Autonomous vehicles have applications  
22. (1) Some people think autonomous vehicet  
23. (1) People waste 91 hours a year looking fo

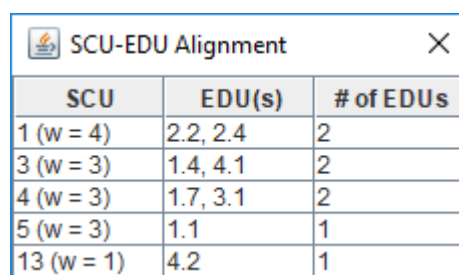
Creating new EDU "Autonomous vehicles will change the way that we travel"

Figure 22. An ordered SEA table in SEAVIEW.

In Figure 22, blue lines group table rows into sections based on the gold-standard essay from which the row's EDUs originated. This provides a visual representation of the number of EDUs from each essay. Each EDU is also assigned an index of the form  $x.y$ , where  $x$  indicates

the essay from which the EDU originated, and  $y$  indicates the order in which the EDU occurred in that essay relative to the other EDUs that have been annotated.

At any point during annotation, the user can also view an SCU-EDU alignment window, shown in Figure 21.

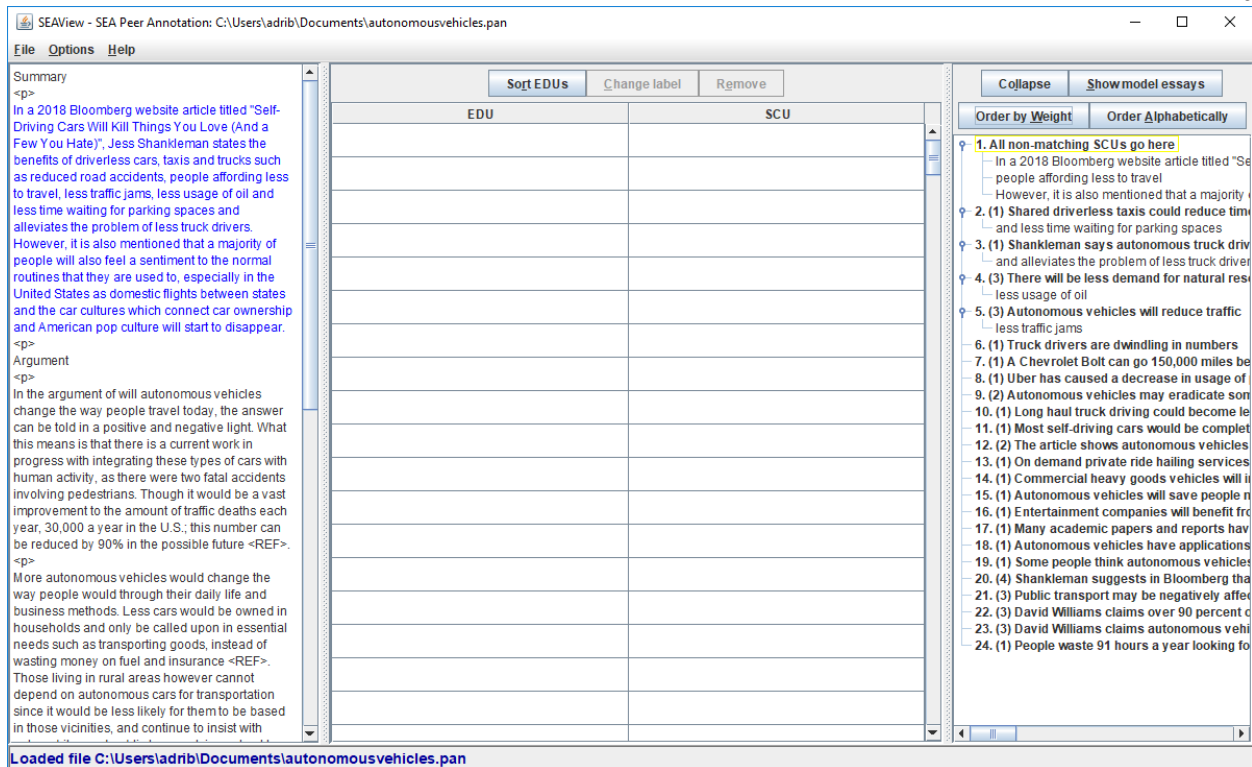


SCU	EDU(s)	# of EDUs
1 (w = 4)	2.2, 2.4	2
3 (w = 3)	1.4, 4.1	2
4 (w = 3)	1.7, 3.1	2
5 (w = 3)	1.1	1
13 (w = 1)	4.2	1

**Figure 23. SCU-EDU alignment window.**

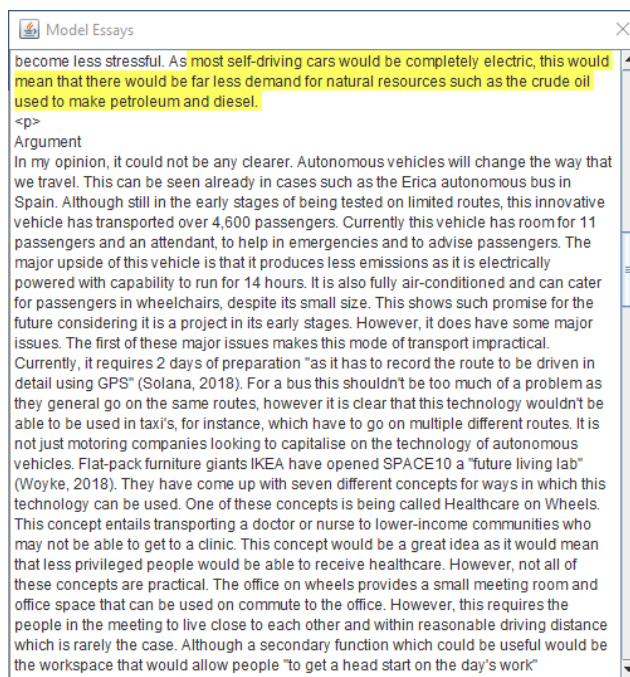
This window displays a list of SCUs in the table, along with the identities and number of EDUs that each SCU appears with in the table. The table can be sorted by the SCU weight or by the number of EDUs by clicking on the column headers. This window is useful for analyzing the distribution of EDUs according to SCU weight.

When the SEA table is complete the composite table is sorted and separated into  $n$  wise crowd SEA tables for  $n$  wise crowd essays. Users can then save the table as a \*.sea file containing the SEA tables, the alignment window shown in Figure 23, the pyramid, and the gold-standard summaries. To complete the essay annotation task using a peer summary (step 4 in Table 3), users can load a \*.pan file created in DUCView. The SEA peer annotation view is shown in Figure 24.



**Figure 24. Peer annotation view in SEAVIEW.**

The peer annotation view is very similar to the gold-standard essay annotation view. The primary differences are that the “Show model essays” button is now enabled and the pyramid now shows peer SCU contributors instead of pyramid SCU contributors. The “Show model essays” button opens a separate window which allows users to view the gold-standard essays that have now been replaced by a peer essay on the left pane. This window is shown in Figure 25.



**Figure 25. Model essays window in SEAVIEW.**

The model essays window shown in Figure 25 is useful for viewing the context of pyramid SCUs, since the SCU labels may not convey the context in which the SCU occurred in the text. Selecting an SCU in the pyramid will highlight the text in the model essays window to enable users to understand the SCU's full context. This window is hidden by default to avoid cluttering the screen when users do not need to view this information.

SEA peer annotation proceeds similarly to SEA wise crowd annotation. Users highlight and drag semantically similar EDUs and SCUs into the SEA table until the peer argument has been completely annotated. The user can also view a SCU-EDU alignment, as in Figure 23, for the SEA peer annotation. After annotating all of the EDUs in the peer summary, the user can save the annotation as a \*.sep file, containing the following elements: SEA peer table, SEA tables from each wise crowd essay, SCU-EDU (peer and wise crowd) alignments, the pyramid, and the DUCView peer annotation.

### 3.4 Analysis of SEAVIEW Design

SEAVIEW's primary design goals include minimizing annotation time and reducing visual clutter; SEAVIEW minimizes annotation time by expediting the process of adding EDUs and SCUs to the SEA table. In DUCVIEW, to create an SCU, a user must highlight text on the left pane, then move the cursor to the right of the window to press a button – either the “New SCU” or “Add Contributor” button, depending on whether the text represents a new SCU or a contributor to an existing SCU. Applying the DUCVIEW approach to SEA table creation, a user would have to select text, select a table cell, and then click a button to add an EDU to the SEA table. In SEAVIEW, users simply highlight text and drag the highlighted text into the SEA table. This reduces the number of clicks and total mouse movement needed to create an EDU or SCU. It also reduces the complexity of EDU creation by allowing users to create new EDUs and add contributors in the same way, rather than requiring separate buttons as in DUCVIEW. While this reduction in mouse movement may appear insignificant, gathering useful data will require annotators to annotate many EDUs and many SCUs across numerous essays. In addition, the argument portions of the essays are longer than the summaries, and EDUs are more numerous than SCUs. Therefore, small time savings in annotation of EDUs and SCUs – the most frequent tasks in SEA table creation – can greatly enhance the user experience of SEAVIEW.

The second design goal of SEAVIEW is to minimize visual clutter. This was accomplished through several design choices. For instance, replacing buttons with drag and drop functionality decreases the complexity of SEAVIEW's appearance compared to DUCVIEW. Non-essential information is displayed in several pop-up windows, such as SCU-EDU alignments and model essays during SEA peer annotation; these windows tailor the user experience according to the level of detail required throughout the process. The table and pyramid also represent this

principle, by allowing users to expand and collapse trees to view or hide contributors. The table also reduces redundancy by primarily identifying SCUs by their indices rather than labels.

Finally, the decision to display the SCU-EDU correspondences in a table format is successful since there is a one-to-one or one-to-zero correspondence between SCUs and EDUs.



## Chapter 4

### Conclusions and Future Work

In conclusion, DUCView has been modified, and SEAView has been created, to complete the first two steps of the argument annotation process. Further development is required to create an ADU annotation tool, which can be used to complete the third step of argument annotation. SEAView can be used to support argument content analysis in student essays, advances in argumentation mining, and automated argument content analysis developments. SEAView can also be used to study the relationships between SCUs and EDUs using this special essay format. For instance, this tool could be used to investigate whether a correlation exists between weight of SCUs in the pyramid and the SCU's frequency in the SEA table. This would suggest more important content units play a larger role in arguments than less important content units. It would therefore be possible that a model of content units, such as a pyramid, would facilitate argumentation mining.

## BIBLIOGRAPHY

- [1] A. Glymph and S. Burg. "The Nation's Report Card: A First Look: 2013 Mathematics and Reading," National Center for Education Statistics, Washington, D.C., Tech. Memo. NCES 2014-451, Nov. 2013.
- [2] A. Glymph. "The Nation's Report Card: Writing 2011," National Center for Education Statistics, Washington, D.C., Tech. Memo. NCES 2012-470, Sep. 2012.
- [3] S. Graham et al., "Teaching writing to middle school students: a national survey," *Reading and Writing*, vol. 27, no. 6, pp. 1015-1042, 2014.
- [4] A. Gillespie et al., "High school teachers' use of writing to support students' learning: a national survey," *Reading and Writing*, vol. 27, no. 6, pp. 1043-1072, 2014.
- [5] R. Passonneau et al., "Wise Crowd Content Assessment and Educational Rubrics," *International Journal of Artificial Intelligence in Education*, vol. 28, no. 1, pp. 29-55, 2018.
- [6] J. Underwood and A. Tregidgo, "Improving Student Writing Through Effective Feedback: Best Practices and Recommendations," *Journal of Teaching Writing*, vol. 22, no. 2, pp. 73-97, 2006.
- [7] M. Kirkland and M. Saunders, "Maximizing Student Performance in Summary Writing: Managing Cognitive Load," *TESOL Quarterly*, vol. 25, no. 1, pp. 105-121, 1991.
- [8] M. Crowhurst, "Teaching and Learning the Writing of Persuasive/Argumentative Discourse," *Canadian Journal of Education*, vol. 15, no. 4, pp. 348-359, 1990.

- [9] D. Prater and W. Padia, "Effects of Mode of Discourse on Writing Performance in Grades Four and Six," *Research in the Teaching of English*, vol. 17, no. 2, pp. 127-134, 1983.
- [10] H. Persky, M. Daane, and Y. Jin, "The Nation's Report Card: Writing 2002," National Center for Education Statistics, Washington, D.C., Tech. Memo. NCES 2003-529, Jul. 2003.
- [11] M. Dornisch and A. McLoughlin, "Limitations of web-based rubric resources: Addressing the challenges," *Practical Assessment, Research & Evaluation*, vol. 11, no. 3, pp. 1-8, 2006.
- [12] J. Surowiecki, *The Wisdom of Crowds*. New York, NY: Doubleday, 2004.
- [13] A. Nenkova, R. Passonneau, and K. McKeown, "The Pyramid Method: Incorporating human content selection variation in summarization evaluation," *ACM Transactions on Speech and Language Processing*, vol. 4, no. 2, 2007.
- [14] S. Sigelman, "DUCView," 2006. [Online] Available: <http://personal.psu.edu/rjp49/DUC2006/2006-pyramid-guidelines.html>.
- [15] Y. Gao, A. Warner, and R. Passonneau, "PyrEval: An automated method for summary content analysis," in *Proceedings of the Eleventh International Conference on Language Resources and Evaluation: LREC 2018, May 7-12, 2018, Miyazaki, Japan*, H. Isahara et al., Eds. Miyazaki: European Language Resources Association. pp. 3234-3239.
- [16] A. Nenkova and K. McKeown, "A Survey of Text Summarization Techniques," in *Mining Text Data*, C. Aggarwal and C. Zhai, Eds. New York: Springer, 2012, pp. 43-76.
- [17] M. Lippi and P. Torroni, "Argumentation Mining: State of the Art and Emerging Trends," *ACM Transactions on Internet Technology*, vol. 16, no. 2, 2016.

- [18] A. Hunter and M. Williams, "Aggregating evidence about the positive and negative effects of treatments," *Artificial Intelligence in Medicine*, vol. 56, no. 3, pp. 173-190, 2012.
- [19] A. Peldszus and M. Stede, "From Argument Diagrams to Argumentation Mining in Texts: A Survey," *International Journal of Cognitive Informatics and Natural Intelligence*, vol. 7, no. 1, pp. 1-31, 2013.
- [20] W. Mann and S. Thompson, "Rhetorical Structure Theory: Toward a functional theory of text organization," *Text*, vol. 8, no. 3, pp. 243-281, 1988.
- [21] L. Carlson and D. Marcu, "Discourse Tagging Manual," Information Sciences Institute, University of Southern California, Tech. Report ISI-TR-545, Sep. 11, 2001.
- [22] A. Peldszus and M. Stede, "Joint prediction in MST-style discourse parsing for argumentation mining," in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Sep. 17-21, 2015, Lisbon, Portugal*, Lisbon: Association for Computational Linguistics. pp. 938-948.
- [23] Java Decompiler Project, "JD-CORE," 2019. [Online] Available: <http://java-decompiler.github.io/>.
- [24] The Apache Software Foundation, "Apache Maven," 2019. [Online] Available: <http://maven.apache.org/index.html>.
- [25] Deathmarine, "Luyten," 2019. [Online] Available: <https://github.com/deathmarine/Luyten>.
- [26] S. Bandyopadhyay, NLP Lab Meeting Presentation, Topic: "Design of a New Tool for Argument Annotation," Pennsylvania State University, University Park, Pennsylvania, Jan. 21, 2019.

## ACADEMIC VITA

Alex Driban

adriban6@gmail.com

### EDUCATION

---

#### The Pennsylvania State University | Schreyer Honors College

- B.S. with Honors in Computer Science May 2019
- B.S. in Biochemistry and Molecular Biology May 2019

### EXPERIENCE

---

#### Penn State Natural Language Processing Lab

Undergraduate Researcher Jan. 2018 – Feb. 2018, Jan. 2019 – May 2019

- Created a new tool for manual markup of content in short summaries and essays
- Collaborated on a dataset for evaluation of PyrEval, an automated method for summary content analysis

#### Penn State Center for Eukaryotic Gene Regulation

Undergraduate Researcher Oct. 2014 – Dec. 2018

- Researched structures of gene regulatory enzymes in Song Tan's lab
- Presented at the Penn State Undergraduate Exhibition

#### Restek Corporation

Operations Technician June 2018 – Aug. 2018

- Evaluated inventory of 4,000+ raw materials to facilitate migration to new software system
- Increased accuracy of digital inventory records

### HONORS AND AWARDS

---

#### Grants and Awards

- Phi Beta Kappa 2019
- Dean's List, *seven semesters* 2014 – 2018
- Erickson Discovery Grant 2015
- The President's Freshman Award 2015

#### Merit Scholarships

- Raytheon Scholarship in Computer Science 2016 – 2018
- Academic Excellence Scholarship 2014 – 2018
- Lum/Pethick/Strauss Engineering Scholarships 2014 – 2018

### ACTIVITIES

---

- *Captain*, Relay for Life 2014 – 2015
- *Player*, Big Ten Chess League 2014 – 2015
- *Volunteer*, Special Olympics 2012 – 2015