LingCues – A Linguistic Cues Software Tool for Research in Text-based Automatic Deception Detection

by

Shayi Zhang

(Under the Direction of Michael A. Covington)

ABSTRACT

LingCues aims to provide a software tool for text-based automatic deception detection (TADD) studies. LingCues' basic function is automatically working out the values of linguistic cues for TADD studies. Linguistic cues are cues that aim to represent particular linguistic properties of a text using numbers. LingCues also allows users to create new linguistic cues and use them in TADD experiments. LingCues does not require any programming background from users. With LingCues' two functions, various linguistic cues may be discovered and new combinations of linguistic cues may be used in future TADD studies.

INDEX WORDS: Deception, honesty, deceptive language, deception detection, natural language processing, linguistic cues, text processing

LingCues – A Linguistic Cues Software Tool For Text-based Automatic Deception Detection Research

by

Shayi Zhang

B.A., Soochow University, China, 2010

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2012

LingCues – A Linguistic Cues Software Tool For Text-based Automatic Deception Detection Research

by

Shayi Zhang

Major Professor:	Michael A. Covington
Committee:	William A. Kretzschmar
	Paula Schwanenflugel

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
August 2012

ACKNOWLEDGEMENTS

First, I would like to thank Michael A. Covington for his encouragement, guidance and assistance in my completing this thesis. His help made this thesis possible. I would also like to thank William A. Kretzschmar and Paula Schwanenflugel for providing advice and serving on my committee. Also, I would like to thank Cati Brown for her patient answers of my questions about her research. Finally, I would like to thank Rada Mihalcea for her sharing with me the datasets she created which enabled me to validate LingCues.

TABLE OF CONTENTS

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

CHAPTER 1

INTRODUCTION

Text-based automatic deception detection (TADD) studies aim to detect deceptive statements or deceptive language in texts by computer analysis of the written words. Text-based deception detection detects deception from text-based documents that consist of written words with punctuation but without pictures or sounds. Text-based automatic deception detection (TADD) is a special kind of text-based deception detection that uses a computer rather than a human being to make judgments.

TADD studies are significant, because the amount of text-based information increases rapidly in the age of computers and the internet. TADD studies are conducted based on the hypothesis that when people are telling lies, they use words in a way different from when they are telling the truth (Pennebaker 2011).

Most TADD studies use linguistic cues as attributes to inspect the differences between true statements and false statements. Linguistic cues are cues that aim to represent particular linguistic properties of a text using numbers. A cue can be a linguistic cue if it is able to indicate a certain linguistic feature of a text, which can be words, phrases or sentences. For example, a cue indicating whether a text-based document uses first person singular pronouns frequently is a linguistic cue. Because texts have no pictures or voices, linguistic cues are important resources of information and, therefore, widely used in text-based studies, including TADD studies. By offering distinct values

for different texts, linguistic cues are able to provide information about the words, phrases, and sentences of these texts (Zhou et al. 2004a).

TADD studies require computers' assistance to calculate the values of linguistic cues. Many previous TADD studies use well-developed software tools to get these values. However, because these software tools are not developed specifically for TADD studies, they may have some linguistic cues useless for deception detection, and they may lack some other useful linguistic cues. Some other TADD studies use self-developed programs or scripts to get linguistic cues' values. However, writing programs or scripts requires a programming background, which some TADD researchers may lack.

My goal in building LingCues was to create a software tool for TADD studies. The main functions of LingCues are providing useful linguistic cues, returning linguistic cues' values and allowing users, even without a programming background, to create and use self-defined linguistic cues for their research. In order to process linguistic cues, LingCues recognizes parts of speech, specific words and a few special linguistic features (section 3.3.2). With these functions, LingCues may help to discover new potential or new combinations of linguistic cues for future TADD studies.

Among the next chapters, chapter two discusses the background of developing LingCues, chapter three is the development of LingCues, chapter four introduces how to use LingCues, and chapter five is the validation and limitations of LingCues.

CHAPTER 2

BACKGROUND

Text-based automatic deception detection (TADD) studies detect deceptive statements or deceptive language in texts. Most TADD studies detect deception using linguistic cues which provide information about certain types of words or other linguistic features (Zhou et al. 2004a).

This chapter discusses the mathematical representation of a linguistic cue, the steps to conduct a TADD study using linguistic cues, and previously used methods of selecting linguistic cues and getting their values.

2.1 Mathematical representation of a linguistic cue

In text-based research, linguistic cues provide information about certain linguistic features (Zhou et al. 2004a). The mathematical representation of a linguistic cue is a fraction equation consisting of a numerator which is the count of a certain type of words, and a denominator which measures the length of a text. For example, Zhou et al. (2004b) creates a linguistic cue "pausality" for deception detection as:

$$\text{Pausality} = \frac{total\ number\ of\ punctuation\ marks}{total\ number\ of\ sentences}$$

In the mathematical equation for "pausality," the numerator is the total number of punctuation marks of a text, and the denominator is the total number of sentences.

Different from "pausality," the denominators of some other linguistic cues measure the length of a text by counting the total number of words. Besides, the denominators in

some other linguistic cues do not measure a text's length, but count words of a particular type, as in the linguistic cue "passiveness" by Brown (2006):

$$\text{Passiveness} = \frac{number\ of\ passive\ verbs}{total\ number\ of\ verbs}$$

There are a few linguistic cues, whose numerator or denominator (or both) is the sum of several counts, or the subtraction of certain counts, as the linguistic cue "Emotiveness" by Zhou (2004b):

$$\text{Emotiveness} = \frac{total\ number\ of\ adjectives + total\ number\ of\ adverbs}{total\ number\ of\ nouns + total\ number\ of\ verbs}$$

2.2 Four steps in a TADD study using linguistic cues

A TADD study using linguistic cues has four major steps: collecting text-based datasets, selecting experimental linguistic cues, getting the values of linguistic cues and analyzing these values.

A TADD study starts with collecting experimental datasets. There are two basic requirements for a good experimental dataset. First, a dataset is required to be text-based. For instance, a text-based dataset may be a number of written documents, transcripts of speeches, or online texts. Second, a good experimental dataset should contain both true and false texts. Some studies collect existing text-based materials as the experimental datasets, such as online articles (Toma and Jeffrey 2010; Rubin and Conroy 2011), or written statements like financial statements (Humpherys et al. 2011) or "person of interest statements" (Fuller, Biros, and Wilson 2009). Some other studies create experimental datasets by inviting participants to tell the truth or lies about given topics

(Newman et al. 2003; Zhou et al. 2004a; Zhou et al. 2004b; Mihalcea and Strapparava 2009).

After collecting experimental datasets, the next step in a TADD study is to decide on good linguistic cues to use. Section 2.2.1 discusses the methods previous TADD studies use.

The third step in a TADD study is to calculate linguistic cues' values. TADD studies rely on computers to determine linguistic cues' values, which saves time and reduces the likelihood of mistakes. Section 2.2.2 introduces the previous methods used to get the values of linguistic cues.

After getting the cues' values, the final step is to analyze these values using machine learning algorithms or certain statistical methods. By using these algorithms or methods, a TADD study may examine the relationships between certain linguistic cues and deceptive language.

Neither the first step nor the last step is discussed in detail, because LingCues does not help much in these two steps. However, both the second step and the third step are essential to LingCues, so the following two subsections discuss them in greater depth.

2.2.1 Selecting linguistic cues

Selecting a group of linguistic cues is significant to a TADD study, because a good group of linguistic cues may lead to a more accurate experimental result. A good group of linguistic cues should include useful linguistic cues for TADD studies and exclude useless linguistic cues (Fuller et al. 2009). Previous TADD studies select linguistic cues using two major methods.

First, many TADD studies rely on well-developed psycholinguistic or linguistic software tools, such as Linguistic Inquiries and Word Count (LIWC) and General Architecture for Text Engineering (GATE). GATE is an infrastructure used in many projects that processes natural language. The two main components in GATE are Language Resources (LR), like corpora, and Processing Resources (PR), like parsers and taggers. GATE is able to count numbers of words in its corpora using LRs and PRs (Cunningham et al. 2011). LIWC is text analysis software that calculates how often a text-based document uses a category of words. LIWC has built-in dictionaries of 72 dimensions of words that can be used as 72 linguistic cues (Newman et al. 2003).

Both GATE and LIWC are popular and powerful research tools, but they are not designed specifically for TADD studies. Many linguistic cues they provide may not be able to detect deception, which might even negatively impact deception detection. Also they may lack some potential linguistic cues for TADD studies.

Among studies using well-developed software tools, some select a sub-list of all the linguistic cues provided by the tool LIWC (Newman et al. 2003; Toma and Hancock 2010; Toma and Hancock 2012; Mihalcea and Strapparava 2009), some select linguistic cues from the tool GATE (Zhou et al. 2004b). Also a few studies combine certain linguistic cues from LIWC and some from GATE into a new group of linguistic cues (Fuller et al. 2009).

The second method used by some TADD studies is to create the new linguistic cues needed, rather than using linguistic cues provided by well-developed software tools (Zhou et al. 2004a; Brown 2006). For example, in Brown's (2006) study of deception in the tobacco industry, she creates the following six linguistic cues that detect corporate

deception: “Adversarial,” “Allness,” “Group,” ”Ambiguity,” “Passiveness,” and “Image.” Brown’s (2006) six linguistic cues are implemented as one group of linguistic cues in LingCues. Section 4.1.2 introduces how to use the six linguistic cues. Additionally, the other group of twelve linguistic cues in LingCues, which are introduced in section 3.2.1, are based on Pennebaker’s (2011) words associated with honesty and deception.

Additionally, some other TADD studies combine the two methods together. These studies use some of the linguistic cues provided by software tools and also create several new linguistic cues for their experiments (Hancock et al. 2010; Rubin and Conroy 2011).

2.2.2 Methods of getting cue values

TADD studies require computers’ assistance in getting the values of linguistic cues. Computers calculate the values of linguistic cues more efficiently than humans, because of the large size of some experimental datasets.

Previous TADD studies seek assistance from computers in various ways. Many studies use the software tools GATE and LIWC (Fuller et al. 2009; Newman et al. 2003; Toma and Hancock 2010; Zhou et al. 2004b; Toma and Hancock 2012; Mihalcea and Strapparava 2009). Both GATE and LIWC provide certain numbers of linguistic cues and return their values for TADD studies. A few studies get linguistic cues’ values using self-developed programs or scripts (Zhou et al. 2004a; Humpherys et al. 2011; Brown 2006). Other studies rely on GATE or LIWC for certain linguistic cues and write programs or scripts to get the values of the new linguistic cues they create (Hancock et al. 2010; Rubin and Conroy 2011), if these new linguistic cues are not provided by GATE or LWNC.

2.3 Limitations of previous methods

One limitation of using well-developed software tools is that these tools may provide many linguistic cues useless for TADD studies, but lack some other potential linguistic cues. Besides, these tools may only provide one mathematical representation for a linguistic feature. However, a linguistic feature may be associated with various mathematical representations and different representations may have their own strengths in different situations.

Writing programs or scripts to get the values of newly created linguistic cues requires programming skills. However, not all TADD researchers have a programming background. Even with a programming background, TADD researchers might spare the time they spent in programming if they had a tool allowing user-defined linguistic cues.

2.4 Goal of building LingCues

An ideal software tool for TADD studies should have the strengths of the previous methods, but avoid their drawbacks. The goal of LingCues is to provide TADD researchers with such a useful tool that offers linguistic cues for TADD studies and also allows users, even without a programming background, to create and use self-defined linguistic cues for their research easily.

CHAPTER 3

THE DEVELOPMENT OF LINGCUES

LingCues is a software tool written in the programming language C# using the development tool Microsoft Visual Studio 2010. LingCues' interface is written using Windows Forms.

Three major parts are very important in the development of LingCues. The first part is the natural language processing techniques which are prerequisites for implementing essential functions of LingCues. The second important part is the implementation of two groups of linguistic cues. The third part is the processing of linguistic cues.

3.1 Natural language processing techniques

In LingCues, the main functions of natural language processing techniques are to recognize and to process text-based documents. Natural language processing is a technology that uses computers to understand human languages. Two natural language processing techniques are implemented in LingCues. One of them is tokenization. Tokenization, the first operation on text-based documents, is identification of the basic units during the computational processing of human languages (Habert et al. 1998). Tagging is another natural language processing technique used in LingCues that annotates each word with a tag indicating the word type (Santorini 1990).

In LingCues, both tokenization and tagging are accomplished by a C# tool named Opportunistically Developed Tagger (ODT) (Covington 2008) that reads text-based documents and recognizes words' part of speech.

3.2 Two groups of linguistic cues in LingCues

LingCues provides two groups of linguistic cues that can be directly used for TADD studies. One group is twelve linguistic cues developed based on Pennebaker's (2011) words associated with deception and honesty. The other group is six linguistic cues detecting corporate deception created by Brown (2006) in her research on tobacco industry deception.

3.2.1 Twelve linguistic cues based on Pennebaker's research

In his book *The Secret Life of Pronouns: What Our Words Say About Us*, Pennebaker (2011) collects many text-based documents and draws the conclusion that some types of words appear more in deceptive statements whereas some others appear more in honest documents (Figure 3.1).

3.2.1.1 Words associated with deception and honesty

In Figure 3.1, Pennebaker (2011) divides words associated with deception and honesty into five groups: self-reference, cognitive complexity, detailed information, social and emotional references, and verbs. To the right of each word type is a horizontal bar that indicates the level of deception or honesty. The three vertical lines from left to right are lines of "Deception," "Mixed," and "Honesty." If the horizontal bar is more left,

its corresponding type of words are more frequent in deceptive texts; if the horizontal bar is more right, the type of words are more frequent in honest texts.

In the development of LingCues, not every word type in Figure 3.1 is designed as a linguistic cue. If a word type is located close the "Mixed" line in Figure 3.1, this type is not selected as a linguistic cue. Only those able to indicate deception or honesty are selected to build linguistic cues.

As a result, twelve linguistic cues are implemented in LingCues as the default group of linguistic cues, among which one linguistic cue is "TimeSpaceNumber" that combines types of words indicating time, space and number together. The first reason for combining the three word types into one linguistic cue is that all three types indicate whether a text-based document has detailed information or not. Second, although "Time," "Space," or "Number" may appear more in honest documents than in deceptive documents, they still appear at extremely low frequencies in many honest documents, close or even equal to zero if counted separately.

The words associated with positive and negative emotions are not included in LingCues' linguistic cues, because of a lack of databases of positive and negative emotion words. Due to the limited time and resources, LingCues lacks many important lexical databases of various types of words. LingCues might be extended in the future with more useful databases, in order to provide more useful linguistic cues for TADD studies.

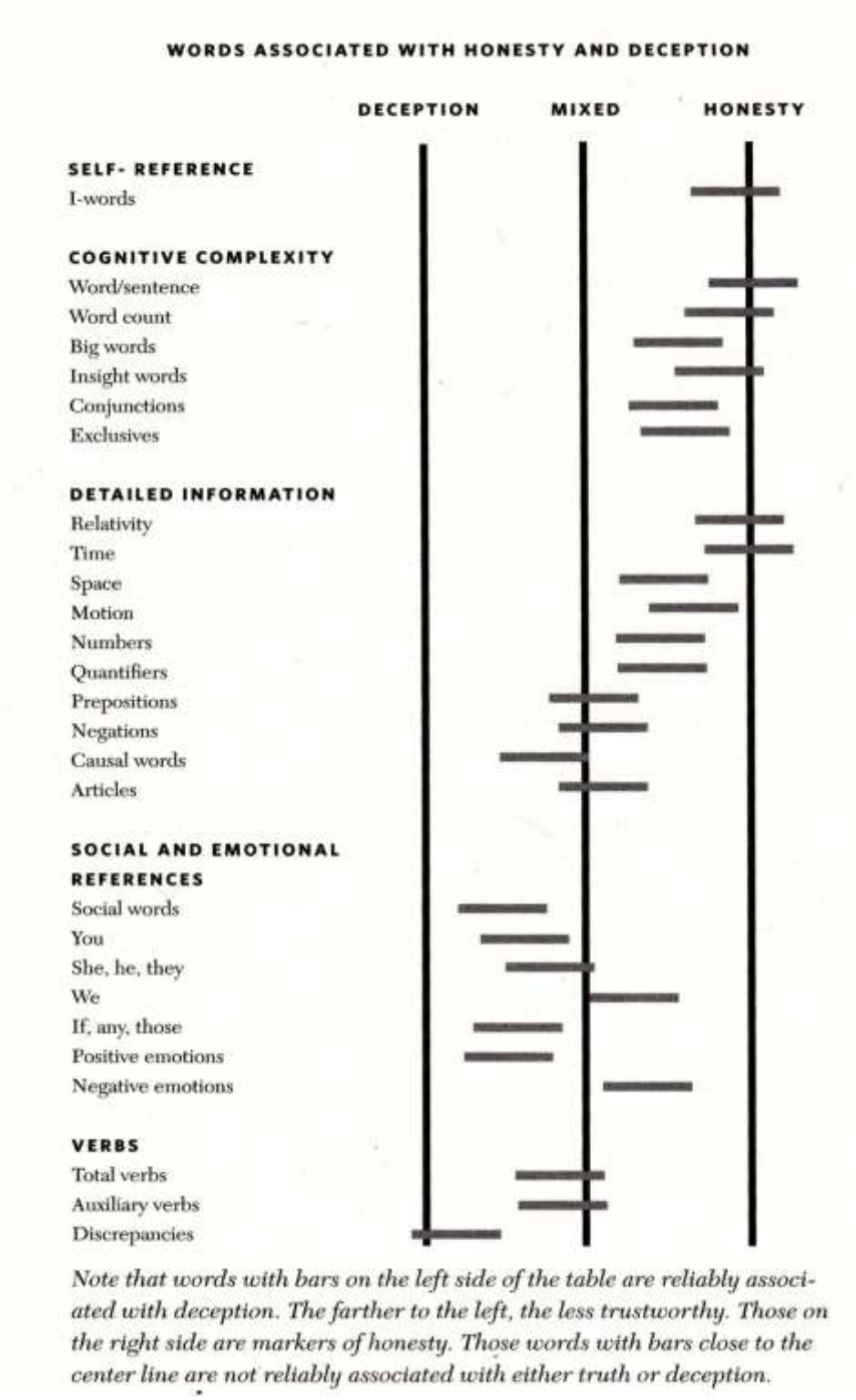

Figure 3.1: Words associated with honesty and deception

(Pennebaker 2011, p. 162)

3.2.1.2 Implementation of the twelve linguistic cues

The twelve linguistic cues are implemented in LingCues according to their mathematical representations. The mathematical representation of a linguistic cue is a fraction equation, in which the numerator is the count of a type of words, and the denominator measures the length of a target text-based document. All the twelve linguistic cues and their mathematical representations are listed in Table 3.1.

Many denominators of the twelve linguistic cues are the total number of sentences. The reason for measuring the length using the total number of sentences rather than the number of words is that the average number of words per sentence itself is a linguistic cue for TADD research (Pennebaker 2011). Pennebaker (2011) suggests that a true story may be more complex than a false story, and that the average number of words per sentence is one of the cues indicating the complexity. Since the number of words per sentence tends to be larger in honest documents than in deceptive documents, honest documents may have more words than deceptive documents. LingCues does not measure the length using the total number of words, because the number may be impacted by a text's complexity. However, the total number of sentences is not impacted by the complexity.

3.2.2 Implementation of Brown's six linguistic cues

Brown's (2006) six linguistic cues for corporate deception are also implemented in LingCues. These six cues are not among the twelve default linguistic cues, but can be imported into LingCues. The six linguistic cues are stored in "CatiCues.txt" (Appendix A). Section 4.1.2 introduces how to import the six linguistic cues into LingCues.

Table 3.1: Mathematical representations of the twelve linguistic cues

Linguistic cue	Type of words	Mathematical representation
I	1st singular personal pronouns	$\frac{number\ of\ 1st\ singluar\ personal\ pronouns}{number\ of\ personal\ pronouns}$
WordPerSent	Average number of words per sentence	$\frac{number\ of\ words}{number\ of\ sentences}$
BigWord	Words with more than 7 letters	$\frac{number\ of\ words\ with\ more\ than\ 7\ letters}{number\ of\ sentences}$
InsightVerb	Verbs that indicate insight	$\frac{verbs\ indicate\ insight}{number\ of\ verbs}$
Coordinating conjunction	Coordinating conjunctions	$\frac{number\ of\ coordinating\ conjunctions}{number\ of\ sentences}$
TimeSpaceNumber	Words that indicate time, space, or number	$\frac{number\ of\ words\ indicate\ time, space\ or\ number}{number\ of\ sentences}$
You	2nd personal pronouns	$\frac{number\ of\ 2nd\ personal\ pronouns}{number\ of\ personal\ pronouns}$
SheHeThey	3rd personal pronouns	$\frac{number\ of\ 3rd\ personal\ pronouns}{number\ of\ personal\ pronouns}$
We	1st plural personal pronouns	$\frac{number\ of\ 1st\ plural\ personal\ pronouns}{number\ of\ personal\ pronouns}$
Quantifier	Quantifiers	$\frac{number\ of\ quantifiers}{number\ of\ sentences}$
IfAnyThose	Number of occurrences of the words *if*, *any*, and *those*	$\frac{number\ of\ occurrences\ of\ the\ words\ if, any\ and\ those}{number\ of\ sentences}$
Discrepancy	Discrepancy	$\frac{number\ of\ discrepancies}{number\ of\ verbs}$

3.3 Processing linguistic cues in LingCues

To process linguistic cues, the internal representation of a linguistic cue is the foundation. Based on this foundation, the cue value equation, the lexical database management, and the cue value calculation system are developed.

Section 3.3.1 introduces the internal representation of a linguistic cue. Section 3.3.2, 3.3.3 and 3.3.4 introduces the cue value equation, the lexical database management and the cue value calculation system respectively.

3.3.1 Internal representation of a linguistic cue

In LingCues, a linguistic cue is represented by the three elemental parts: the linguistic cue's name, the numerator of the linguistic cue's mathematical representation and the denominator. LingCue's internal representation correlates a linguistic cue's name with its mathematical representation, which not only mathematically represents a linguistic cue but also enables computers to retrieve the mathematical equation in order to calculate the cue's value.

The internal representation requires that linguistic cues have distinct names, because LingCues recognizes a linguistic cue by its name. Given a linguistic cue's name, LingCues can retrieve the numerator and denominator of that linguistic cue.

In LingCues, the three elemental parts together represent a linguistic cue. The order of the three parts is fixed. LingCues reads in the three parts one by one, treats the first one as the name, and view the second and third ones as the numerator and the denominator automatically.

3.3.2 Counts in LingCues

LingCues recognizes parts of speech, specific words and a few special linguistic features such as the total number of words, and records their occurrences as counts, which are the basic units that constitute the numerator and the denominator of linguistic

cues' mathematical representations. The general function of a count is to count a certain type of words in a text-based document. The value a count returns is the total number of words belonging to that type. There are three methods used to count words, and consequently three groups of counts in LingCues.

Counts in the first group count words directly from texts, because these counts do not require lexical databases of key words, or any information about words' parts of speech tags. Lexical databases are text files that contain the key words to count. Users of LingCues can import lexical databases and then use the corresponding counts to create new linguistic cues. For example, the two counts "Word" and "Sentence" are counts not requiring lexical databases or words' tags. "Word" is a count that counts how many words are in a text-based document, and "Sentence" counts how many sentences are in a text-based document. In addition to "Word" and "Sentence", LingCues has a count "BigWord" that counts words with more than 7 letters, which also belongs to the first group.

Counts in the second group require lexical databases of key words. A lexical database is a text file that contains keywords to count. Counts of this type count how many times the keywords in lexical databases appear in a text-based document. For instance, "You" is a count in this group that has a database containing all second personal pronouns.

Counts in the third group enumerate words by the words' parts of speech tags. A count of this type counts words with a required tagger. For example, "Determiner" is a count that counts words tagged as determiners.

LingCues recognizes and is able to process all the linguistic cues constituted of counts that belong to the second and third groups. LingCues may not recognize linguistic cues consisted of counts belonging to the first group, if these counts are not built in LingCues. For example, LingCues is unable to recognize a linguistic cue for words starting with the letter *z*, since this linguistic cue requires a count LingCues does not have, which counts words starting with the letter *z*.

3.3.3 Cue value equation

The cue value equation is the visualized mathematical representation of a linguistic cue. In order to help users design linguistic cues, the cue value equation is implemented in LingCues' interface as a fraction equation where users can edit the numerator and the denominator.

Before user input, the cue value equation looks like an empty fraction equation with no numerator or denominator:

$$\text{cue value} = \frac{\qquad}{\qquad}$$

Users can use the cue value equation to design any new linguistic cues. For example, users can use it to design a linguistic cue named "Conjunction" as:

$$\text{cue value} = \frac{\text{Coordinating\_conjunction}}{\text{sentence}}$$

In this equation, "Coordinating_conjunction" is the count that counts words tagged as *coordinating_conjunction*, and "sentence" is the count that counts the total number of sentences.

3.3.4 Lexical database management

Lexical databases in LingCues are managed through lexical database management. A lexical database is a text file containing key words to count. Users of LingCues can delete or add lexical databases using lexical database management. Section 4.1.2 introduces how to add new lexical databases to LingCues.

In a lexical database, every two words need a space to separate them. If a key word is a phrase, the words of this phrase should be connected by @. LingCues automatically treats words connected by @ as a phrase.

3.3.5 Cue value calculation system

Calculating a linguistic cue's value is a recursive process in LingCues. LingCues returns the value of a linguistic cue in four steps.

First of all, with the linguistic cue's name, LingCues searches for its numerator and denominator. Then LingCues looks into counts which are the basic units of the numerator and the denominator.

Second, LingCues works out the values of all the counts. To get the value of a count, LingCues first figures out in what group this count is. LingCues counts directly words of a certain type, if this count is in the first group. If the count is in the second group that has a lexical database of key words, LingCues then counts the lexical database's key words. If the count belongs to the third group that requires knowing the word's part of speech tag, LingCues then counts the words with the required tag.

Third, with the values of the counts, LingCues works out the values of the numerator and the denominator. And finally, LingCues gets a linguistic cue's value by dividing the numerator by the denominator.

CHAPTER 4

HOW TO USE LINGCUES

Chapter 4 introduces the functions of LingCues. This chapter consists of two sections: one section introduces how to get the values of linguistic cues, and the other section introduces how to create new linguistic cues using the Cues Editor in LingCues.

4.1 How to get linguistic cues' values

LingCues' basic function is automatically working out linguistic cues' values. LingCues can work out the values of both the default linguistic cues and the imported linguistic cues. Section 4.1.1 introduces how to use LingCues to get the values of the default linguistic cues. Section 4.1.2 introduces how to import linguistic cues into LingCues and then get their values.

4.1.1 How to get the values of the default linguistic cues

First of all, in order to run the software LingCues, users need to open it by clicking "LingCues.exe." LingCues' main form is shown in Figure 4.1.

In the main form of LingCues, the 12 default linguistic cues provided by LingCues are in the left box "Selected Cues." To choose the linguistic cues, users can check or uncheck the small boxes to the left of these default linguistic cues.

After that users should choose one text file or more text files to analyze. If they want to use only one text file, users can click "File" in the menu and then choose "Open a File"

(Figure 4.2). If they want to use more than one text files, users can click “Open a Directory” and then select the target folder in which all the texts are stored.

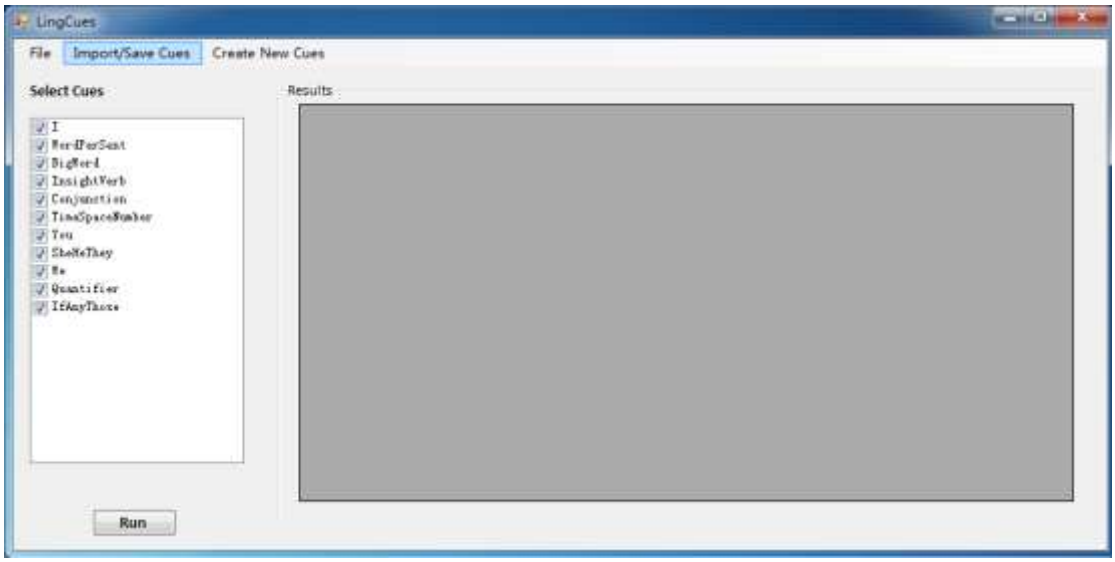

Figure 4.1: LingCues’ main form

Here is an example using LingCues to get the values of the default linguistic cues for 99 true statements about best friends. Mihalcea & Strapparava (2009) created these statements for their deception detection research. In our example, we click “Open a Directory,” because we have more than one text file.

After opening the text file or files, users can click the button “Run” below the “Selected Cues” box to get the values of the selected linguistic cues. LingCues may spend a few seconds to computationally work out the values of the linguistic cues, and then show them in the box “Result” as in Figure 4.3. The values may not show completely in the “Result” box because of its limited size. However, users are able to

view all the values if they move the vertical or horizontal scrollbars. Users can also save these values if they choose the option “Save Results” which is below “Open a Directory.”

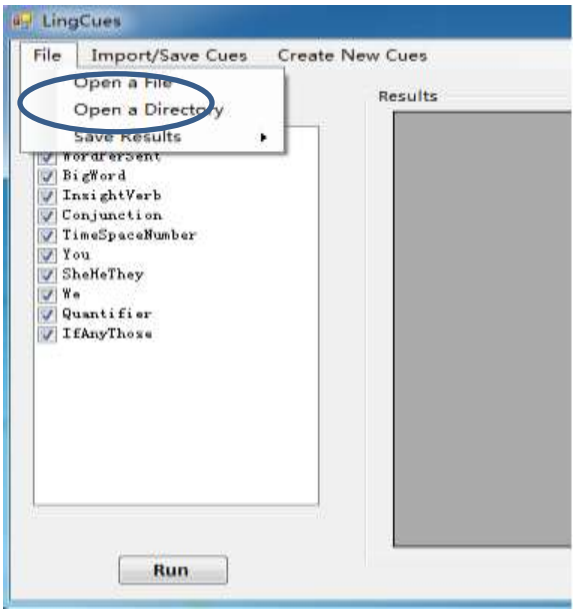

Figure 4.2: The options “Open a File” and “Open a Directory”

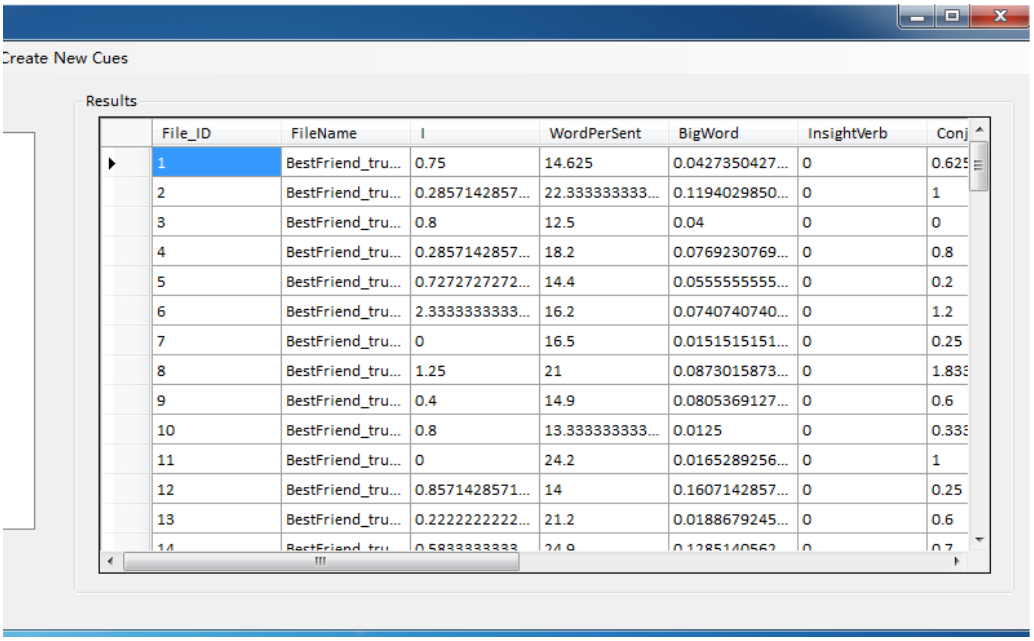

Figure 4.3: The “Result” box (1)

4.1.2 How to import linguistic cues

In addition to the twelve default linguistic cues, LingCues also allows users to use unlimited numbers of linguistic cues by importing text files that contain linguistic cues' information.

The following example shows how to import the six corporate deceptive linguistic cues built by Brown (2005).

After running "LingCues.exe," click "Import/Save Cues" in the menu in order to import linguistic cues into LingCues. Then, choose "Import Cues" under "Import/Save Cues" as shown in Figure 4.4. To import the six linguistic cues, find and select the file "CatiCues.txt" that contains the cues' information. We will see that the six cues show in the "Selected Cues" box, below the default twelve linguistic cues (Figure 4.5).

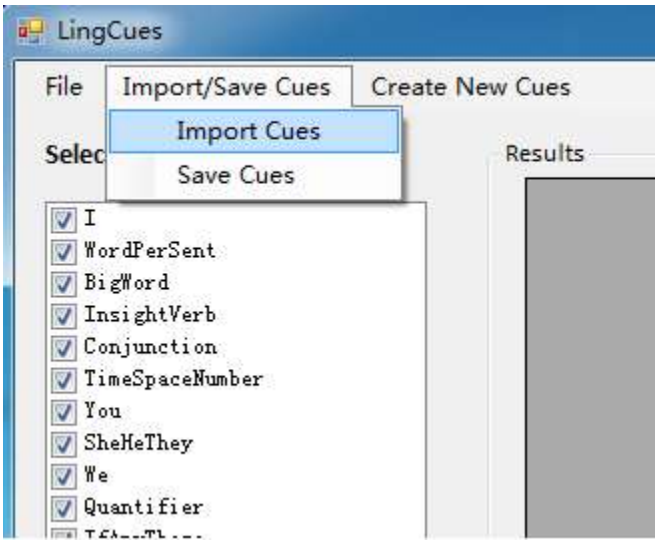

Figure 4.4: The "Import Cues" option in the main form

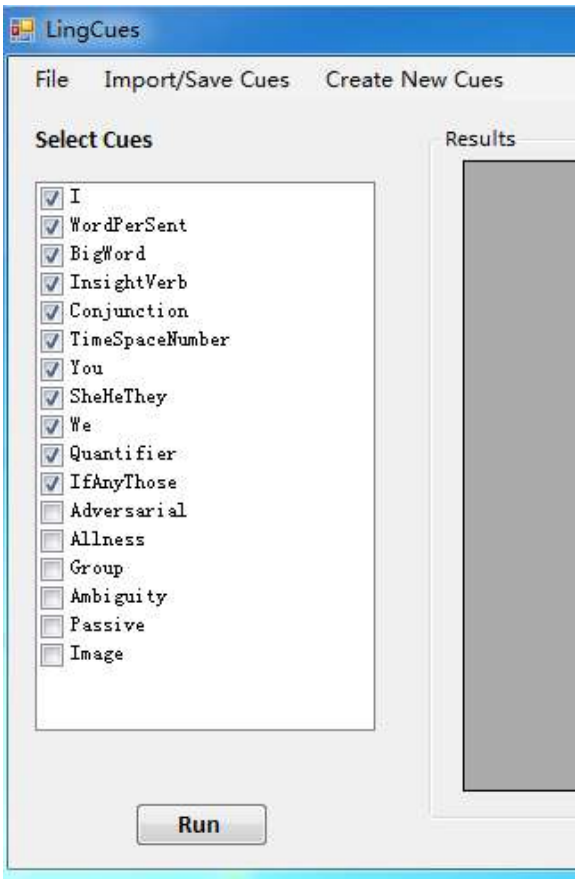

Figure 4.5: The six imported linguistic cues shown below the default ones

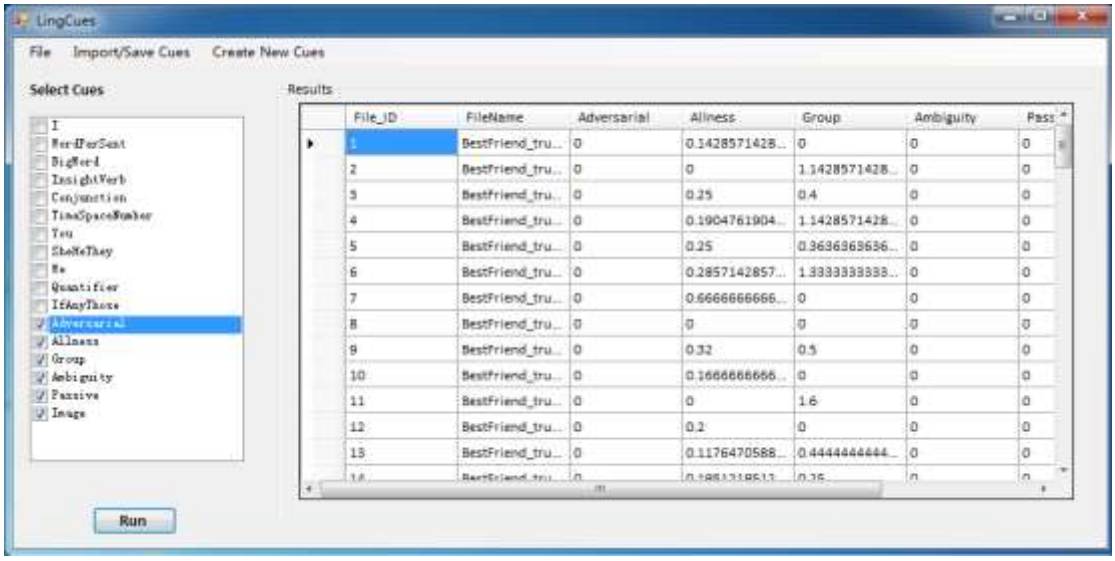

Figure 4.6: The "Result" box (2)

To only use the six newly imported linguistic cues, check the six cues and then uncheck the twelve cues above them. After that, open the text files and click the "Run" button. The values of these six linguistic cues will show in a few seconds in the "Result" box as in Figure 4.6.

LingCues also provides another method of importing linguistic cues which allows only the imported cues shown in the "Selected Cues" box. The advantage of this method is that there is no unnecessary linguistic cue shown in "Selected Cues" box if users only want to use the newly imported linguistic cues. In order to use this method, users should first open the Cues Editor by clicking "Open Cues Editor" (Figure 6.7) under "Create New Cues" in the main menu. The Cues Editor (Figure 6.8) is responsible for creating, editing, and deleting linguistic cues.

The top part of the Cues Editor is a drop-down box named "Linguistic Cues," from which users of LingCues can view all the available linguistic cues. The linguistic cues in the "Linguistic Cues" drop-down box are the same as those in the "Select Cues" box in LingCues' main form.

In Cues Editor, users can click the second top button "Delete All" to firstly remove all the linguistic cues currently available in LingCues. After that, users can click "Import/Save Cues" in the menu and then chose the "Import Cues" option to import a new list of linguistic cues. Users can click "Back to LingCues" button to return to the main form of LingCues. They will find that only the newly imported linguistic cues show in the "Select Cues" box just as in Figure 4.9.

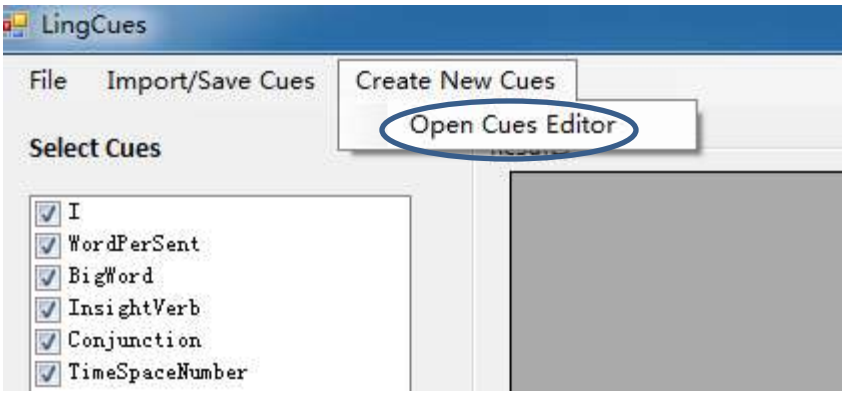

Figure 4.7: The "Open Cues Editor" option

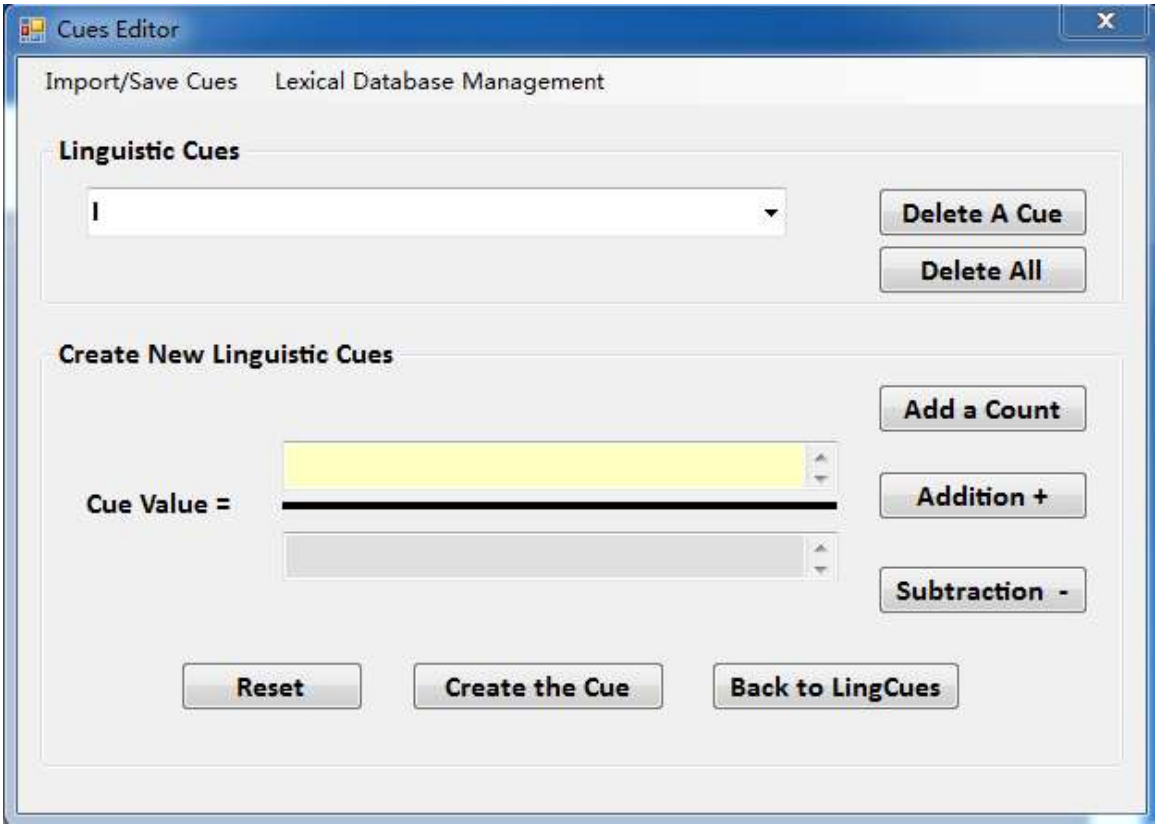

Figure 4.8: Cues Editor

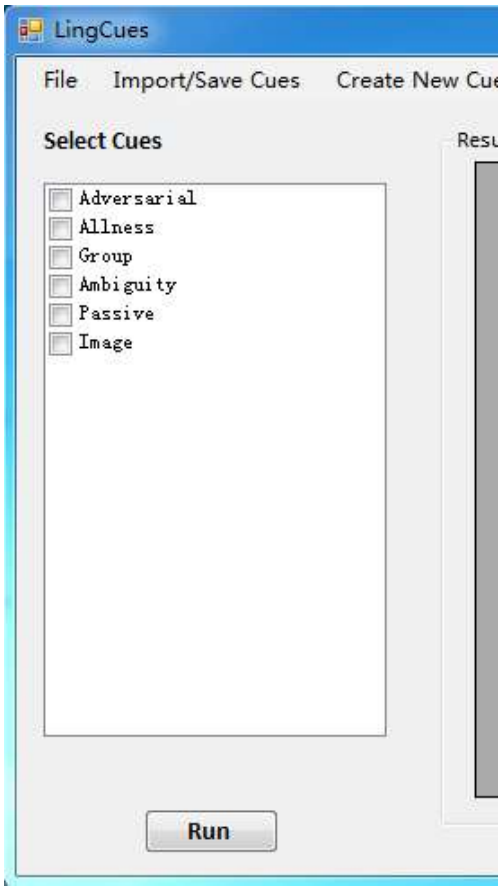

Figure 4.9: The six imported linguistic cues in the box "Select Cues"

4.2 Create new linguistic cues

LingCues allows users to create user-defined linguistic cues easily. This section introduces how to create new linguistic cues using LingCues' Cues Editor. In addition to creating new linguistic cues, users can use LingCues'Cues Editor to delete or edit existing linguistic cues.

Before introducing how to use the Cues Editor, it is significant to mention briefly the three parts that are essential to the Cues Editor: the cue value equation, the group of counts provided by LingCues and the lexical database management. The cue value equation, which is the visualized mathematical representation of a linguistic cue, is the

core part of the Cues Editor. Section 3.3.2 discusses the cue value equation in detail. The available counts provided by LingCues are counts that users can use directly to create new linguistic cues without importing any lexical database. Section 3.3.1.1 introduces the three groups of counts in LingCues. Users are able to add more counts to LingCues by adding corresponding lexical databases that contain key words. Lexical database management is introduced in section 3.3.3.

How to create a new linguistic cue is introduced through the example of creating a new linguistic cue indicating how often a text mentions the English names of the 12 months. The purpose of creating this linguistic cue is only to introduce how to create a new linguistic cue in LingCues. It is not discussed here whether this cue is a good linguistic cue for TADD studies.

4.2.1 Adding a new lexical database

The main function of the lexical database management is importing or deleting text files that contain the key words for certain counts.

The text files which serve as lexical databases are texts that contain lexical words. In such a text file, every two individual words are separated by a space. If there is a particular phrase made up of more than one word, all the words of this phrase should be connected by the mark @. For example, the phrase *a little* should be written as *a@little* in a database text file.

The first step to create a new linguistic cue about the 12 months is to create a lexical database containing the 12 months' English names. We create a new text file and name it "Month.txt," in which *Month* is the name of this file and *.txt* means that the file's format

is text. We write into this text file the 12 months and insert a space between every two words. So the content of the lexical database "Month.txt" is "January February March April May June July August September October November December."

After creating "Month.txt," we can open the Cues Creator by clicking "Create New Cues" then choosing the option "Open Cues Editor" in the menu. In the Cues Editor, in order to import "Month.txt" into LingCues as a lexical database, we click "Lexical Database Management" and then choose "Import Lexical Database(s)" in the menu. Then we choose the option "Open a File" as highlighted in Figure 4.10. If we want to import more than one lexical database, we can use the option "Open a Directory" to select a folder which stores all the files we want to import.

After importing "Month.txt" as a lexical database, we can view it if we click "View All lexical databases" in the menu under the option "Lexical Database Management." In the new form "All Databases," the newly-imported database "Month" is at the bottom of the lexical database list (Figure 4.11). The databases above "Month" are the databases for LingCues' default linguistic cues.

4.2.2 Counts provided by LingCues

Users of LingCues can import lexical databases and then use the corresponding counts to create new linguistic cues. Besides, LingCues provides many counts that can be directly used without importing any lexical database. These counts include the total number of words, the total number of sentences, and many other counts.

Users can view all available counts in a form named "Choose to Add." The "Choose to Add" form shows all the counts, including both counts LingCues provides, and counts

by importing lexical databases. Users can open the form “Choose to Add” by clicking the button “Add a Count” in the Cues Editor. The form “Choose to Add” also provides a reference of the counts LingCues provides (Figure 4.12).

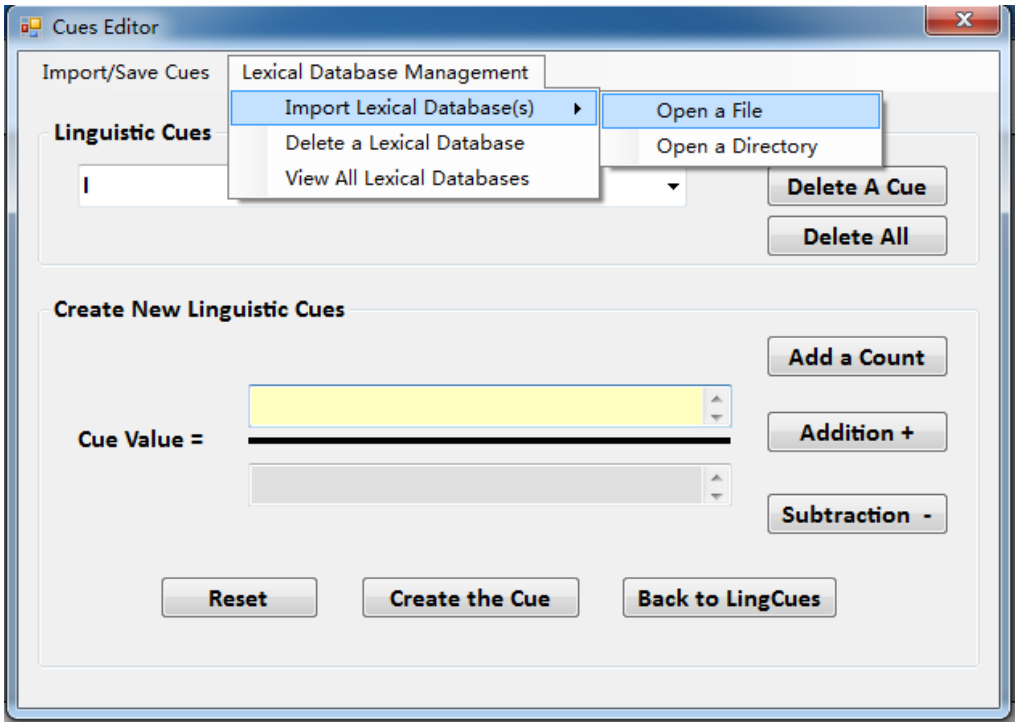

Figure 4.10: “Import Lexical Databases(s)” and “Open a File”

4.2.3 The cue value equation

The cue value equation mathematically represents a linguistic cue, and also enables computers to calculate the cue’s value. Users can easily create a new linguistic cue using the cue value equation by adding counts and operational characters like + or -. After that, the computer is able to calculate the value of a linguistic cue by evaluating its corresponding cue value equation.

It is essential for users to design the cue value equation first. A cue value equation has two parts, a numerator and a denominator. Normally, the numerator or the denominator is a count, or the sum or subtraction of various counts. The numerator counts words of a certain type. The denominator may measure the length of a document by counting the total number of words.

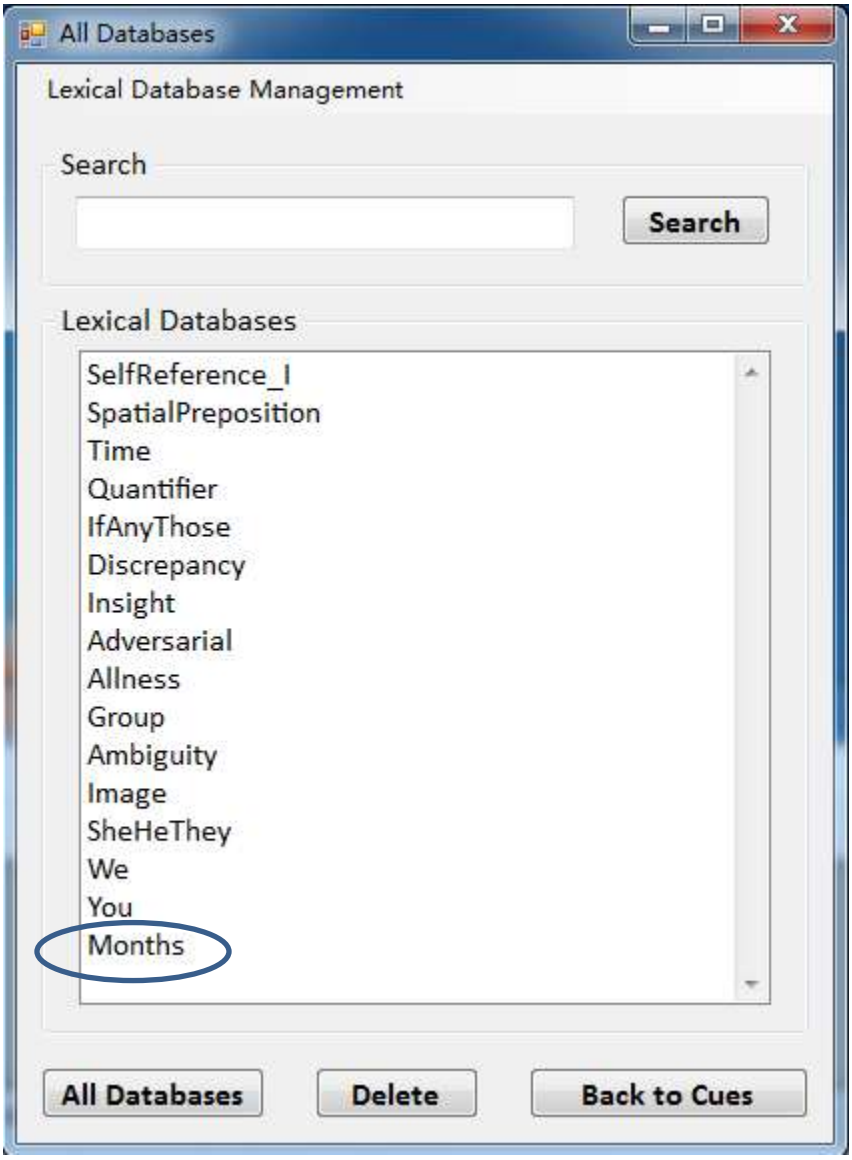

Figure 4.11: The form "All Databases"

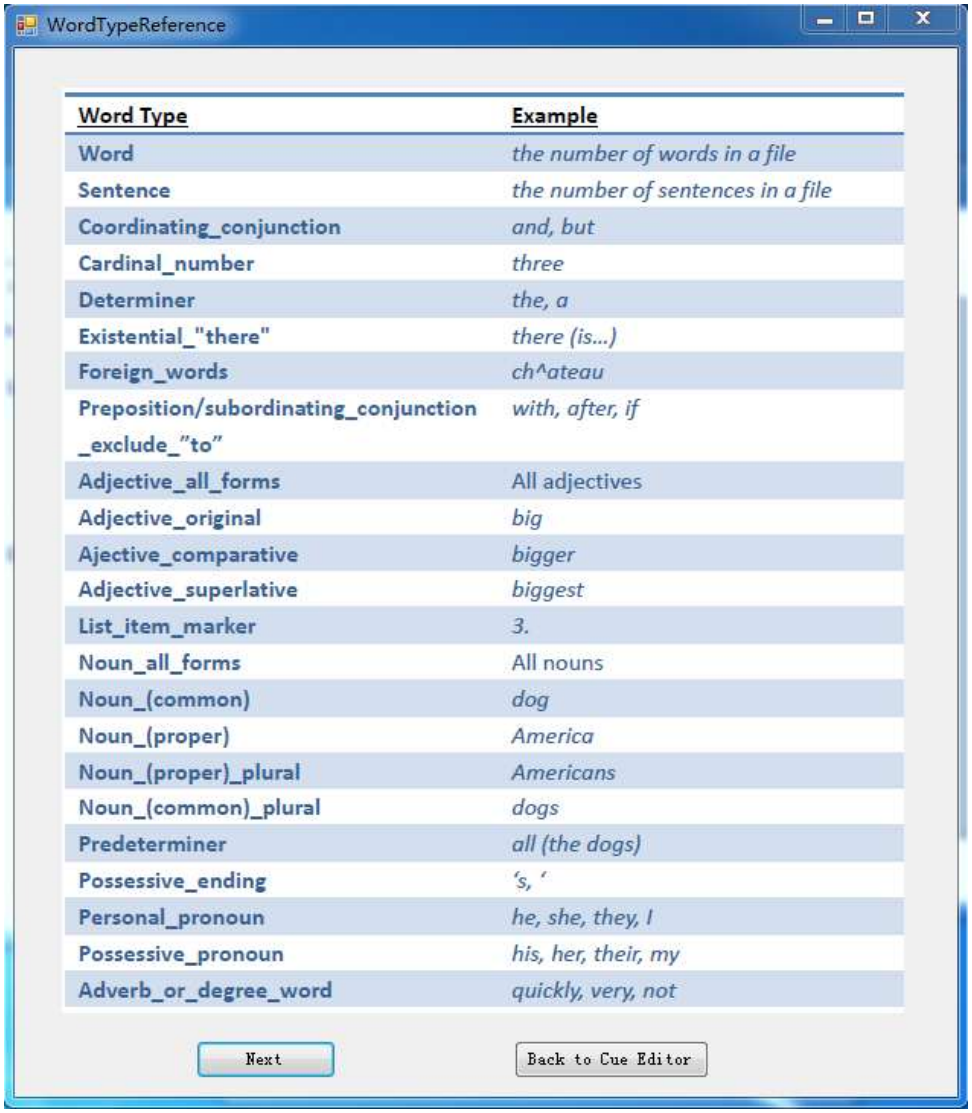

Figure 4.12: Word type reference page 1

In our example, we want to find out how often a document mentions the 12 months. We have already imported the lexical database "Month.txt" that contains the English names of the 12 months. In the cue value equation for the new linguistic cue, we want the count "Month" as the numerator and the total number of words as the denominator. As a result, we design the cue value equation as:

$$\text{Cue Value} = \frac{\text{Month}}{\text{Word}}$$

In the cue value equation above, “Month” is the count of the 12 months, and “Word” is the count of words.

After designing the cue value equation, we can start to create the new linguistic cue. In the Cues Editor’s cue value equation, users can click the position of the numerator or the denominator to choose where to edit. The background color of the position turns from gray to yellow when this position is ready to edit. In our example, we want to use the count “Month” as the numerator, so we click the position of the numerator and then click the button “Add a Count.” We select the count “Month” in the new form “Choose_to_Add.” In the same way, we select the count “Word” as the denominator. The finished cue value equation in the Cues Editor is shown in Figure 4.13.

Then we can click the button “Create the Cue” and we will see a new form that asks for the new cue’s name (Figure 4.14). We name the new linguistic cue as “12_Months.” Upper and lower case letters, numbers, and the underscore mark “_” can be used in a linguistic cue’s name. Besides, LingCues does not allow any space or other characters in a cue’s name.

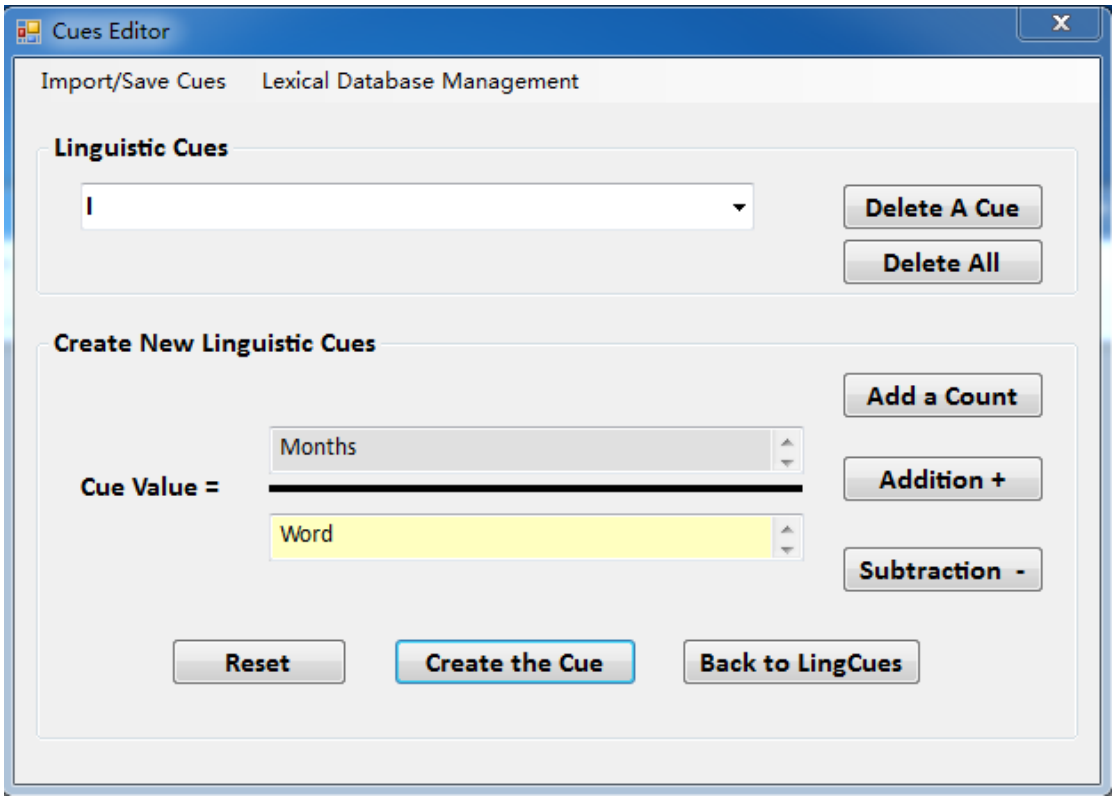

Figure 4.13: The cue value equation in Cues Editor

The new linguistic cue "12_Months" then shows in the drop-down box "Linguistic Cues" (Figure 4.15). The new linguistic cue "12_Month" also shows in the box "Select Cues" of the main form (Figure 4.16).

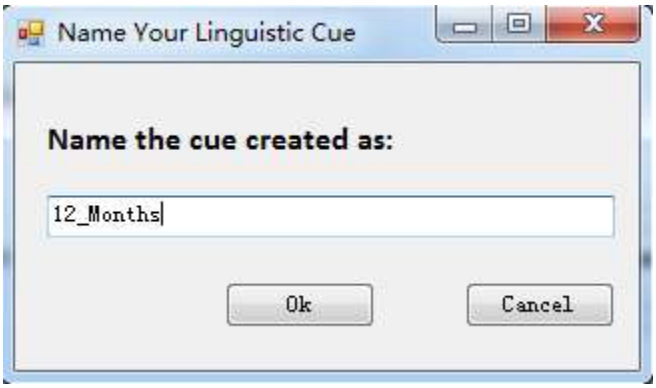

Figure 4.14: The form "Name Your Linguistic Cue"

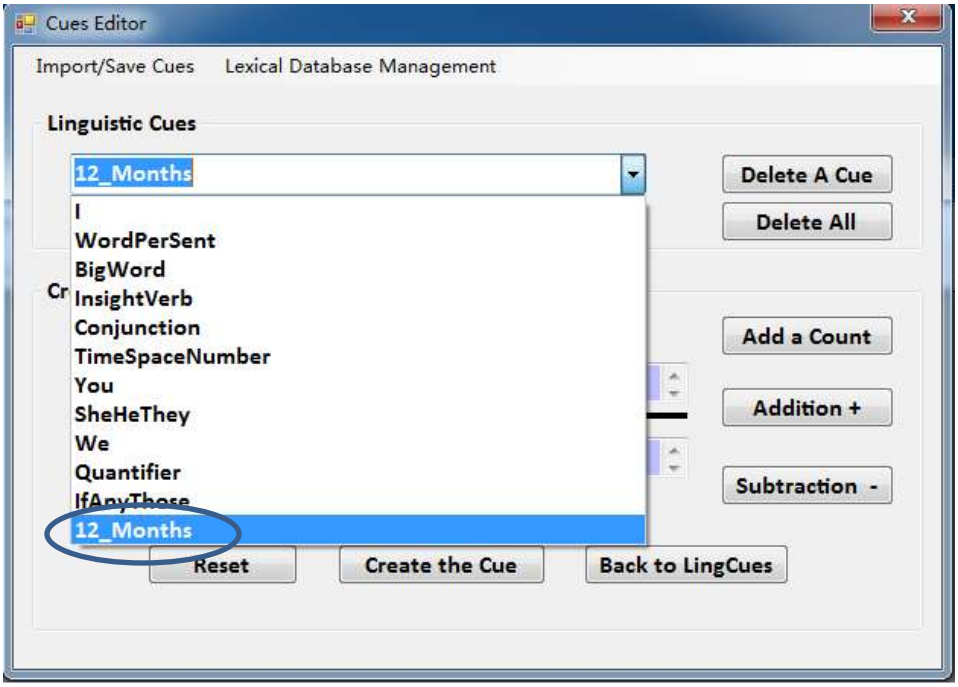

Figure 4.15: The new cue 12_Months in the drop-down box "Linguistic Cues"

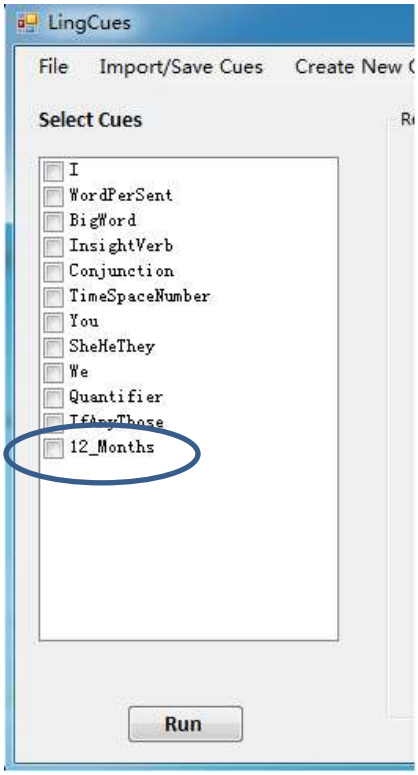

Figure 4.16: The new cue 12_Months in the box "Select Cues"

4.2.4 How to save linguistic cues for future use

Users of LingCues can save linguistic cues for their future use. They can save any combination of linguistic cues, including user-created cues. To save linguistic cues for future use, users just need to click "Import/Save Cues" in the main menu and then choose "Save Cues."

The linguistic cues users save are the linguistic cues listed in the box "Select Cues" in the main form, which also show in the drop-down box "Linguistic Cues" in the Cues Editor. If users want to save more linguistic cues than those already listed in LingCues, they have to import the linguistic cues or create new ones. If they want to remove linguistic cues from the list, they can open the Cues Editor and then delete a linguistic cue by clicking the "Delete A Cue" button (Figure 4.17).

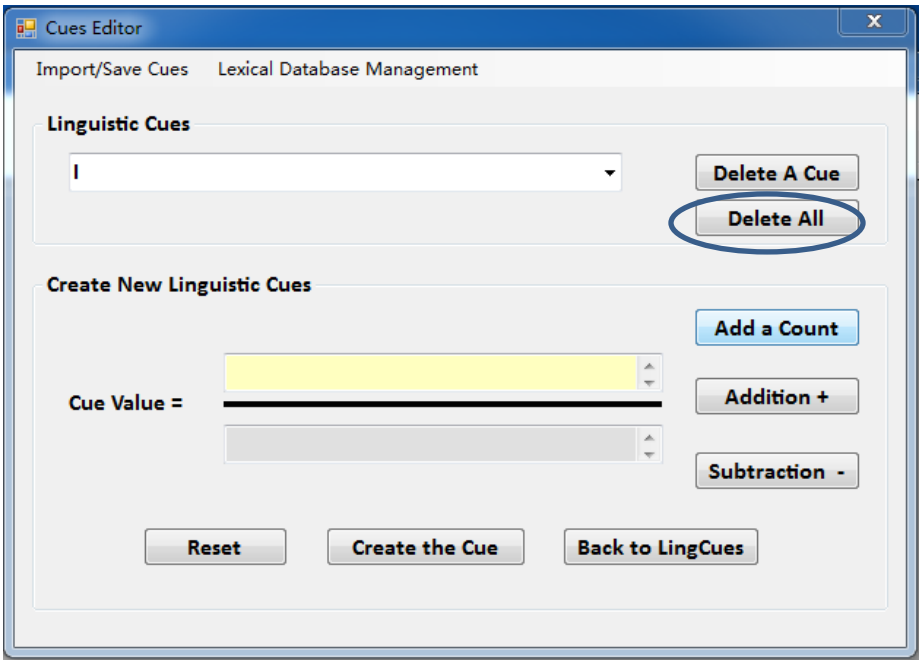

Figure 4.17: The button "Delete A Cue" in Cues Editor

CHAPTER 5

VALIDATION AND LIMITATIONS

In chapter five, section 5.1 examines the validity of LingCues and section 5.2 discusses its limitations.

5.1 Validation

In this section, a simple TADD experiment is conducted to validate LingCues' performance. The experimental datasets are true and false statements about the three topics "abortion," "death penalty," and "best friend." Mihalcea and Strapparava (2009) create these statements as the experimental datasets for their research. This experiment uses the same classfiers as those used in Mihalcea and Strapparava's experiment (2009). Finally the experimental results will be compared with Mihalcea and Strapparava's (2009) results.

In their experiment, Mihalcea and Strapparava (2009) use all the 72 linguistic cues LIWC has as the 72 attributes. They also use LIWC as the tool to get the values of the 72 linguistic cues. The two classifiers they use are ten-folder cross-validation Naïve Bayes and SVM. Their experimental results are in Table 5.1.

Unlike Mihalcea and Strapparava's (2009) experiment, this validating experiment uses LingCues' twelve default linguistic cues as the attributes.

Table 5.1 Mihalcea and Strapparava' results

(Mihalcea and Strapparava 2009, p. 311)

Topic	NB	SVM
ABORTION	70.0%	67.5%
DEATH PENALTY	67.4%	65.9%
BEST FRIEND	75.0%	77.0%
AVERAGE	70.8%	70.1%

5.1.1 Experimental data sets

The three experimental datasets are built by inviting participants to give true and false speeches about the three topics "abortion," "death penalty," and "best friend." The text-based statements in the experimental datasets are the transcripts of these speeches. 100 true and 100 false text-based statements are collected for each topic, with average 85 words for each statement (Mihalcea and Strapparava 2009).

5.1.2 Linguistic cues and classifiers

This expeirment uses LingCues to process the exprimental datasets. LingCues' twelve default linguistic cues associated with deception and honesty are used as the attributes. LingCues reads in all the statements and returns linguistic cues' values for each statement. The twelve default linguistic cues' mathematical representations and implementation are introduced in section 3.2.1.

This experiment uses ten-folder cross-validation Naïve Bayes and SVM as the two classifers to classify true and false statemens.

5.1.3 Experimental results and comparision

Table 5.2 shows the results using LingCues' twelve default linguistic cues. Because the number of true statemens and that of false statements are equal for each dataset, the baseline of a classification is 50%. In this experiment, the two classifiers achieve 66.71% and 72.48% accuracy on average, which are higher than the baseline.

With the LingCues tool, the classification accuracy using the classifer Naïve Bayes is lower than Mihalcea and Strapparava's (2009), but the result using SVM is higher.

Table 5.2 Results using LingCues

Topic	NB	SVM
ABORTION	67.5%	71%
DEATH PENALTY	61.73%	69.90%
BEST FRIEND	70.92%	76.53%
AVERAGE	66.71%	72.48%

LingCues' performance is validated by this experiment. In general the experimental results are very close to Mihalcea and Strapparava's (2009) results. However, this experiment ueses only twelve linguistic cues by LingCues, while Mihalcea and Strapparava use 72 linguistic cues by LIWC. LingCues are more efficient in detecting text-based deception. LIWC are less efficient because many linguistic cues LIWC has might be useless for TADD studies.

5.2 Limitations of LingCues

LingCues is a software tool designed for TADD studies, but it has its own limitations.

Due to limited time and resources, LingCues lacks many important lexical databases of various types of words, such as lexical databases of words associated with positive emotions and negative emotions. By expanding LingCues with more lexical databases, future LingCues releases might be able to provide more useful linguistic cues for TADD research. Besides, in order to ensure LingCues' quality, more datasets are needed to test LingCues' performance.

REFERENCES

Brown, C. G. (2006). *Rating tobacco industry documents for corporate deception and public fraud: a corpus linguistic assessment of intent*, Dissertation, Ph.D., Univerisyt of Georgia.

Covington, M. A. (2008). *Opportunistically Developed Tagger(ODT)*. Unpublished computer program. Institute for Artifical Intelligence, The University of Georgia.

Cunningham, et al. (2011) *Text Processing with GATE (Version 6)*. University of Sheffield Department of Computer Science. 15 April 2011. ISBN 0956599311. Retrieved June 21, 2012 (http://gate.ac.uk/sale/tao/tao.pdf).

Fuller, Christie M., David, P. B., and Rick L. W. (2009). Decision support for determining veracity via linguistic-based cues. *Decision Support Systems*, 46(3), 695-703. Retrieved March 1, 2012 (http://linkinghub.elsevier.com/retrieve/pii/S0167923608001991).

Hancock, J. T., Beaver, D. I., Chung, C. K., Frazee, J., Pennebaker, J. W., Graesser, A., and Cai, Z. (2010). Social language processing: A framework for analyzing the communication of terrorists and authoritarian regimes. *Behavioral Sciences of Terrorism and Political Aggression* 2 (Memory and Terrorism), 108-132. Retrieved April 16, 2012 (http://www.tandfonline.com/doi/abs/10.1080/19434471003597415).

Humpherys, S. L., Kevin, C. M., Mary B. B., Judee K. B., and William F. F. (2011). Identification of fraudulent financial statements using linguistic credibility analysis. *Decision Support Systems*, 50(3), 585-594. Retrieved May 10, 2012 (http://linkinghub.elsevier.com/retrieve/pii/S0167923610001338).

Mihalcea, R., and Carlo, S. (2009). The Lie Detector: Explorations in the Automatic Recognition of Deceptive Language. *ACLShort '09 Proceedings of the ACL-IJCNLP* , 309-312.

Newman, M. L., Pennebaker J. W., Diane, S. Berry, and Jane, M. R. (2003). "Lying words: Predicting deception from linguistic styles." *PERSONALITY AND SOCIAL PSYCHOLOGY BULLETIN*, 29(5), 665-675. Retrieved June 17, 2012 (http://psp.sagepub.com/content/29/5/665.short).

Pennebaker, J. W. (2011). *The secret life of pronouns : what our words say about us*. 1st U.S. New York: Bloomsbury Press.

Rubin, V. L., and Niall J. C. (2011). "Challenges in automated deception detection in computer‐mediated communication." *Proceedings of the American Soc. for Information Science and Tech. Annual Meeting*, 9-12. Retrieved June 16, 2012 (http://onlinelibrary.wiley.com/doi/10.1002/meet.2011.14504801098/full).

Santorini, B. (1990). *Part-of-speech tagging guidelines for the Penn Treebank Project*. Philadelphia, Pa., Dept. of Computer and Information Science, School of Engineering and Applied Science.

Toma, C. L., and Jeffrey, T. H., (2010). "Reading between the lines: linguistic cues to deception in online dating profiles." *Proceedings of the ACM conference on Computer-Supported Cooperative Work*, CSCW 2010, 5-8. Retrieved June 16, 2012 (http://dl.acm.org/citation.cfm?id=1718921).

Toma, C. L., and Jeffrey, T. H. (2012). "What Lies Beneath: The Linguistic Traces of Deception in Online Dating Profiles." *Journal of Communication*, 62(1), 78-97. Retrieved March 24, 2012 (http://doi.wiley.com/10.1111/j.1460-2466.2011.01619.x).

Zhou, L., Burgoon, J. K., Nunamaker, J. F., and Twitchell, D. (2004a). "Automating Linguistics-Based cues for detecting deception in Text-Based asynchronous Computer-Mediated communications." *GROUP DECISION AND NEGOTIATION*, 13, 81-106. Retrieved June 16, 2012 (http://www.springerlink.com/index/v1612l2v814r4000.pdf).

Zhou, L., Burgoon, J. K., Twitchell, D. P., and Qin, T. (2004b). "A comparison of classification methods for predicting deception in computer-mediated communication." *JOURNAL OF MANAGEMENT INFORMATION SYSTEM*, 20(4), 139-166. Retrieved June 16, 2012 (http://mesharpe.metapress.com/index/62m1jqcctguejjcr.pdf).

APPENDIX A

CATICUES.TXT

"CatiCues.txt" is a file that can be directly imported into LingCues as the six linguistic cues created by Brown (2006). These six linguistic cues are briefly introduced in section 3.2.2. After installing LingCues, "CatiCues.txt" is in LingCues's folder named "CueListSample" as a sample list of linguistic cues. Below, the contents of "CatiCues.txt" show how a list of linguistic cues is saved in LingCues.

In "CatiCues.txt", the required lexical databases (start with "$") are listed first, for LingCues to check whether these lexical databases are available. The second part is the internal representations of the linguistic cues. Each representation starts with "#", and consists of a linguistic cue's name, its numerator and denominator. The three parts of a representation is separated by a space. In a numerator or denominator, a mathematical operation is surrounded by "@" for LingCues to recognize it.

```
Sorry, "CatiCues.txt" is not released at this time.
If you are interested in LingCues, please email the author syzhang@uga.edu
```

APPENDIX B

SELECTED SOURCE CODE

The selected source code below is responsible for calculating the value of a linguistic cue when LingCues knows its name. Basically, when LingCues knows the name, it first calculates the values of the counts that form the numerator and denominator of the cue's mathematical equation. Then LingCues calculates the values of the numerator and denominator. Finally, LingCues returns the linguist cue's value using the denominator to divide the numerator.

```
Sorry, the source code is not released at this time.
If you are interested in LingCues, please email the author syzhang@uga.edu
```