Chapter 5
Text Mining Tutorial

Natalie J. Lambert

5.1 Introduction

The world we live in is generating text at an unprecedented rate. Consider how much new text is created by emails, newspapers, blogs, and social media websites every day, and it quickly becomes clear that analysis of group behaviors can become challenging due to the large amount and variety of textual data generated from group members' interactions. Text mining is one strategy for analyzing textual data archives that are too large to read and code by hand, and for identifying patterns within textual data that cannot be easily found using other methods. Text mining as a method can be used to conduct basic exploration of textual data, or can be used in combination with other methods like machine learning to predict group members' future behaviors. This tutorial introduces text mining by outlining two basic methods for data exploration: generation of a concept list and generation of a semantic network. Learning the steps it takes to prepare, import, and analyze textual data for these simple procedures is enough to get started analyzing your own datasets. This tutorial is only a glimpse of the text mining method, however, and new text mining programs and algorithms are continually being developed. Readers interested in learning more about text mining should take formal courses or explore the many text mining packages available in programming languages like R and Python.

Most fundamentally, text mining is a methodology used to extract information, classify data, and identify patterns within textual datasets. It is even more accurate to say that text mining is a collection of methodologies because just as there can be many patterns within any one collection of text, there are many ways to identify these patterns using text mining. Historically, text mining was used to search

N.J. Lambert (✉)
Brian Lamb School of Communication, Purdue University, West Lafayette, IN, USA
e-mail: njlambert@purdue.edu

A. Pilny, M.S. Poole (eds.), *Group Processes*, Computational Social Sciences,
DOI 10.1007/978-3-319-48941-4_5

computer documents in order to identify which documents contained a word or words of interest, and to extract specific information from documents (Fan, Wallace, Rich, & Zhang, 2006). Early electronic card catalogs in libraries utilized text mining to tag and index catalogue holdings (Miner, 2012), and text mining has been used to automatically generate research article abstracts from the content of articles since the 1950s (Luhn, 1958). Text mining is used today by businesses and researchers for a multitude of purposes such as analyzing news stories in order to understand the public's perception of health topics like AIDS (Caputo, Giacchetta, & Langher, 2016), to extract trends in consumer opinions from product reviews posted online (Dasgupta & Sengupta, 2016), and to manage information overload in research fields like biomedical research (Cohen & Hersh, 2005).

There are many situations where other methodologies cannot provide the type of information about a textual dataset that text mining can offer. A researcher with 60 hours of audio recordings of focus group interviews is faced with around 1,800 pages of transcriptions. Hand coding of such data for a factor of interest usually requires multiple readings of the text by several researchers, and such large textual datasets are often a daunting barrier to analysis even when they offer significant benefits like coverage of a greater variety of research subject demographics and backgrounds. Text mining can search through these large datasets for evidence of a factor of interest in seconds as opposed to the many hours it would take to manually search all of the transcriptions.

Another benefit of text mining is its ability to perform data-driven discovery. Data-driven discovery is the process of looking for patterns within datasets without pre-conceived hypotheses regarding what the researcher expects to find. Using the traditional scientific method, the researcher with the large archive of focus group transcriptions would have analyzed the data in order to answer a specific hypothesis such as, "Organizational groups that utilize a cooperative approach to conflict will attain higher productivity ratings than organizational groups that utilize a competitive approach to conflict." The researcher would likely answer this hypothesis by focusing on instances of conflict within the transcriptions, using a method like structural equation modeling to evaluate whether there is a relationship between group conflict style and the groups' productivity. Data-driven discovery conducted using text mining allows the researcher to broaden his or her focus to *anything* within the transcriptions that is significant to the conversation generated during the focus groups. Topic modeling or cluster analysis of a semantic network generated from the transcriptions could reveal a number of frequently-occurring topics like wage gaps or understaffing that a hypothesis-driven approach not focused on these topics would be unlikely to identify. Text mining can also be used in combination with other methods to double-check whether any frequently-occurring themes or words are present within the data that were not recognized by other forms of analysis. Text mining should not, however, be considered in any way superior to traditional research methods—it simply offers a new approach to examining textual data and is especially useful for managing data overload.

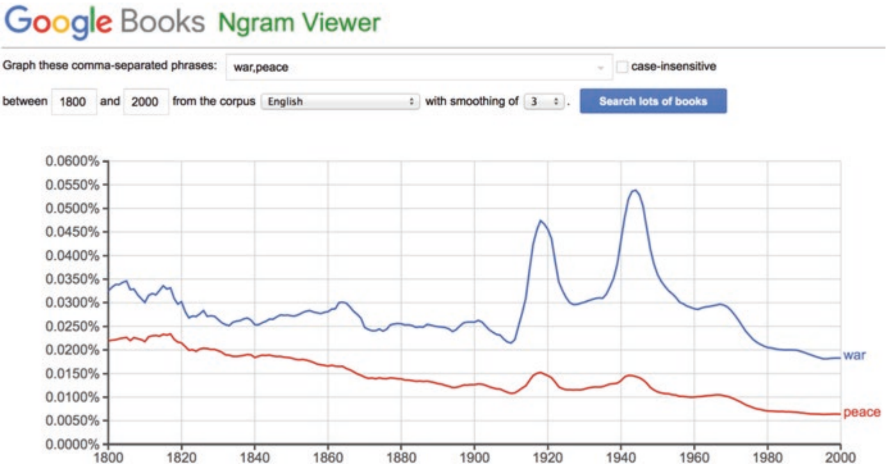
5.2 Overview of Text Mining

There are many analyses that can be performed using text mining, but the way in which the method operates is similar for most text mining procedures. During a text mining procedure, an algorithm built into the software contains a set of instructions for how to examine the text data and what to make note of. For example, during the first phase of analysis, called preprocessing (described in more detail below), the algorithm for the procedure called "stop word removal" tells the software to look word by word through the text data for all the words on a "stop word list," a list of words the researcher wants to exclude from analysis. The software "reads" through the entire dataset one word at a time, comparing each word in the dataset to the words on the stop word list, removing all words from the dataset that match a word found on the list. Another common procedure in text mining is the generation of a concept list, which is an inventory of all of the words in a dataset along with a count of how frequently each word appears in a dataset. The algorithm that creates the concept list also passes through the text word by word, adding new words it encounters to the concept list and adding a count to a word's tally number each time it reencounters the same word in the text. There are many more sophisticated ways that text mining algorithms draw information from a text archive than those just described, but the basic principle is that an algorithm contains a set of instructions for how the software should read and keep track of information found within the text. A full text mining analysis almost always involves running multiple procedures in a particular order in order to extract the information a researcher is interested in from the text.

As the reader likely can imagine, text mining as a method has some very specific assumptions built into it. The biggest assumption is certainly that individual words can have meaning even when they are far removed from their original context. A concept list, for example, counts the total number of times each word in a dataset appears within the text without taking the specific context where each word was used into account. The word "hate" means something very different when someone says "I hate my job" and "I'd hate to lose my job," but a standard concept list cannot tell you that. Data scientists are building algorithms and text mining approaches that can take the context of all words into account (see Lexalytics, 2015), but for scholars new to text mining it is important to remember that words spelled the same but with different meanings can be counted as the same concept. Another common assumption of text mining is that frequently-occurring words within a text archive are more significant than infrequently occurring words. This may indeed be the case, as it is in the tutorial example, or a word could simply occur frequently because it is a commonly used word for a certain language or context. There are also cases where word frequency is completely unimportant for understanding a particular dataset. It is therefore the analyst's responsibility to think through algorithms' built-in assumptions when performing text mining.

A third assumption of many text mining algorithms is that words that occur near each other in a text archive are related in some way. This chapter will demonstrate how to generate a semantic network, which is a group of words existing within a text archive that have been found to share some sort of relationship in common. According to many text mining algorithms, what these words usually have in common is proximity. Text mining tools commonly assign two words to the same meaning group when they both occur within a certain distance of each other within the text. It seems safe to assume that words that occur within the same sentence or paragraph are related, but if we look at the “I hate my job” and “I’d hate to lose my job” example again, it possible to see how words that occur near one another in textual data can be related but also have context-specific meanings that can be overlooked by algorithms only interested in words’ relative positioning.

These examples are not meant to foster mistrust in text mining, but rather for the reader to gain an understanding of what the method can and cannot do. Text mining can provide a researcher with valuable information about his or her data such as which people, organizations, and places feature prominently within it, analyzed through a procedure called entity detection. Text mining can give a data analyst a sense of the emotion being expressed during conversations through a procedure called sentiment analysis. The text mining method can also map out dominant conversations taking place within communication datasets, showing where there is overlap between conversation topics. Or, text mining can be used to identify important phrases or patterns within business reports in order to expose reoccurring problems (Choudhary, Oluikpe, Harding, & Carrillo, 2009) and to detect public health rumors online (Collier et al., 2008). Google Book’s Ngram viewer (http://books.google.com/ngrams) is an example of how simply tracking word frequencies over time can result in a sense of the rise and fall of the public interest in different topics. Figure 5.1 visualizes a comparison of the frequency of the appearance of the words “war” and “peace” over time in Google’s large book archive. Note the rise in the term “war” following the first and second world wars. The graph also indicates that although books contained the word “war” more frequently than “peace,” the appearance of “war” and “peace” followed very similar patterns.



Graph these comma-separated phrases: war,peace □ case-insensitive

between 1800 and 2000 from the corpus English with smoothing of 3.

Fig. 5.1 Google Books Ngram viewer graph of the words “war” and “peace”

The result of text mining analysis is a summary of a pattern identified by the procedure that was run. The form this patterns takes can vary quite a bit, from a simple concept list to a very complex semantic network map or a new document file containing extracted data that fits the parameters the algorithm was designed to find, such as a subconversation. The results of text mining reveal something about what words or sections of a text archive are meaningful either because they occur frequently, are closely related to other words within the text, or because they fit some other parameter set by the researcher. Text mining analysis is not complete, however, until the analyst has reexamined the results back within the context of the data in order to interpret the meaning of the pattern. While the patterns text mining can reveal often seem self-explanatory, a deeper understanding of the data is only gained by assessing why certain words were found to be related and not others, and what this means for the group being studied.

5.3 Text Mining Tutorial

There are many methods that can be used to conduct exploratory text mining. This tutorial covers basic preprocessing steps as well as the generation of a concept list and semantic network. These text mining techniques will be demonstrated using AutoMap (Carley, 2001), a text mining tool developed by the CASOS Group at Carnegie Mellon University. (See Carley, Columbus, Bigrigg, Diesner, and Kunkel (2010) for a tutorial.) There are dozens of text mining tools, each with their individual benefits and suitable for different analyses and types of datasets. Tools like AutoMap that have graphical user interfaces are excellent for beginners interested in exploratory text mining. Once an analyst is comfortable with basic text mining, however, he or she will likely need to learn some programming skills in order to perform advanced procedures customized to his or her particular dataset.

The overall process for conducting text mining is: (1) data collection, (2) data preparation, (3) pre-processing, (4) analysis, and (5) interpretation. This tutorial will take you through each step of the method by describing an analysis conducted for a research project which examined small groups of emergency medical physicians as they drew on their professional expertise during medical consultations in order to develop patient treatment plans (Lammers, Lambert, Abendschein, Reynolds-Tylus, & Varava, 2016).

5.3.1 Data Collection

The sample text corpus used in this tutorial was collected during a study of medical consultations taking place in the emergency department of a hospital. The emergency department was staffed by about two dozen full-time physicians, including doctors, physicians' assistants, nurse practitioners, and medical residents. The team of researchers was permitted to observe physicians' conversations with one another in their shared office space away from patients. The researchers transcribed by hand, as verbatim as was possible, the conversations between physicians related to patients' care in the emergency department. They also noted which physician initiated each conversation and which physicians participated in each conversation. The data collection totaled 90 h of observations, which resulted in 159 pages of field notes and a text corpus of medical consultations containing 19,868 words. The following is a hypothetical example of a typical medical consultation observed by the research team, created in order to preserve participants' privacy:

Doctor: What's going on with room 23?
Resident: He's a 42-year-old man, diabetic. Complaining of pain in abdomen and side. No fever, white count is normal.
Doctor: Possible kidney stone. Any pain medicine prescribed? You can give him morphine.
Resident: Sounds good.

The research team was interested in studying medical consultations because existing research had shown that communication problems between physicians can result in treatment errors, especially during patient handoffs (Maughan, Lei, & Cydulka, 2011). Medical professionals had also called for a better understanding of medical consultations beyond exploratory studies offering models and taxonomies of medical consultations (Kessler et al., 2011). Little was known about what a medical consultation looks like or what topics or problems physicians encounter during consultation, and so that is what the research team set out to learn by collecting and analyzing empirical observations of medical consultations. Their goal was to distinguish between different types or topics of medical consultations in order to better understand how medical professionals enact expertise. Text mining was a useful method for this research project because the data collected by the team was unstructured textual data, meaning the data was in its naturally occurring form and not classified or organized into a database. The researchers knew very little about the data since no one had ever looked at the topics surrounding medical consultations before. An exploratory method that could look for patterns within the textual data was therefore the best fit, and that is what text mining is designed to do.

5.3.2 Data Preparation

After collecting a textual dataset, the next step of the text mining method is to prepare the data for analysis. Data preparation involves removing all data items from a text archive (often called a text corpus) except for the text of interest, and converting the data into a format that the software can import and read. In the case of the example research team, once they decided that they wanted to analyze medical consultations between all the physician role types, they removed the role labels from the text corpus (i.e., Doctor, PA, etc.) so that only the transcribed medical consultations remained. The next step was to copy all the text transcriptions and paste them into Notepad. When using AutoMap and many other text mining tools, the file

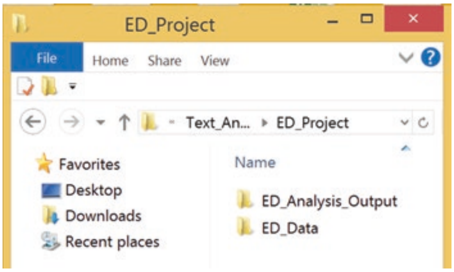


Fig. 5.2 Data and output folder creation

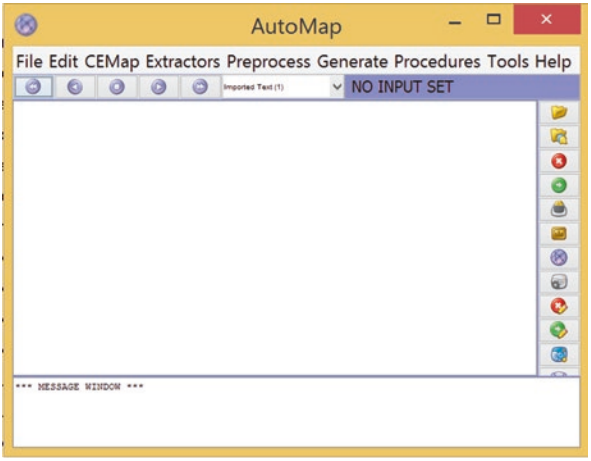
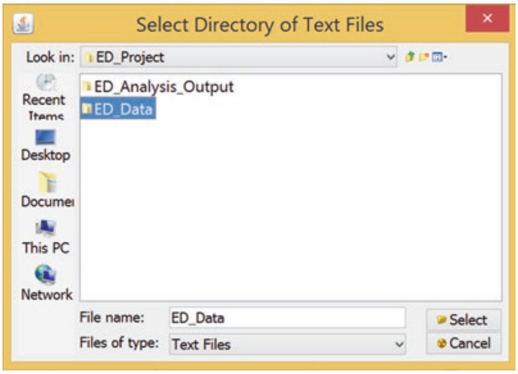

AutoMap
File Edit CEMap Extractors Preprocess Generate Procedures Tools Help
Imported Text (1)
NO INPUT SET
*** MESSAGE WINDOW ***

Fig. 5.3 The AutoMap home screen

extension of the data file must be ".txt", because the data must be contained within in a plain text file in order for the software to be able to read it. Other text editors can be used instead of Notepad as long as they do not preserve file formatting and can generate plain text files. The research team data analyst saved the plain text file containing the medical consultation data within a new folder, and did not place anything else in the folder. If a dataset is comprised of multiple text files, the analyst should place all of the text files he or she wants to analyze simultaneously within this folder. AutoMap will import all files within the folder as one dataset. The research team data analyst next created another empty folder where the data analysis output would be stored. The dataset and analysis output folders can be seen in Fig. 5.2.

After properly formatting and storing the medical consultation data, the data analyst imported the data file into AutoMap. To do this, the analyst began at the AutoMap home screen (Fig. 5.3) and imported the data file by clicking on ***File—Import Text Files***. The next step was to click once to highlight the file folder containing the data, and then click ***Select*** (Fig. 5.4). The data analyst used the preselected settings for text encoding and text direction and pressed ***Enter***.

Fig. 5.4 Importing a text corpus into AutoMap



Select Directory of Text Files

Look in: ED_Project

ED_Analysis_Output
ED_Data

File name: ED_Data

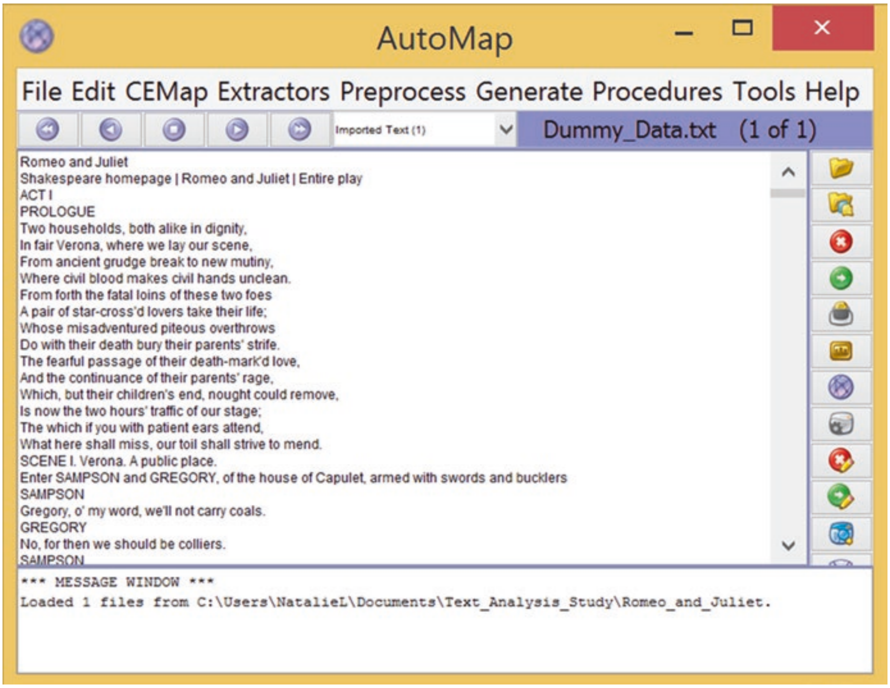
Files of type: Text Files

The text contained within the data file was imported into AutoMap and displayed in the text display pane. Due to privacy agreements with the example study's research subjects, this tutorial cannot show the transcription of the medical consultation dataset. As an alternative, the full script of Shakespeare's *Romeo and Juliet* (Fig. 5.5) has been imported into AutoMap using the previously described steps. This tutorial will use *Romeo and Juliet* as dummy data to demonstrate the next step, data preprocessing, and then return to the medical consultation transcriptions to show the results of a real data analysis.

As can been seen in Fig. 5.5, the all-caps indicators of the act, scene, and characters are included in the imported data file. This was done in order to learn more about the main features of the play, and because the analyst decided in advance that characters were important features of the play and therefore should be included in data analysis. If the analyst was instead interested in analyzing the dialogue of the play and wanted to compare and contrast different characters' dialogue, her or she would have collected each character's lines into separate plain text files and removed all-caps text and any other non-dialogue text from the files. Each file would be analyzed separately and comparisons made of the individual analysis results for each character. Data preparation is a very important part of the text mining method because during this step the analyst must make choices about what selections of a larger text corpus to include in the analysis. Every text corpus contains different characteristics that must be taken into consideration when making decisions about how to best prepare data to answer a specific research question.

5.3.3 Preprocessing

Once the dataset has been formatted and imported into AutoMap, the next step is preprocessing of the data. Preprocessing is a term used to describe the cleaning up and standardizing of textual data prior to analysis. Two common types of preprocessing are stop word removal and stemming. Stop words are any words that would


AutoMap
File Edit CEMap Extractors Preprocess Generate Procedures Tools Help
Imported Text (1)
Dummy_Data.txt (1 of 1)

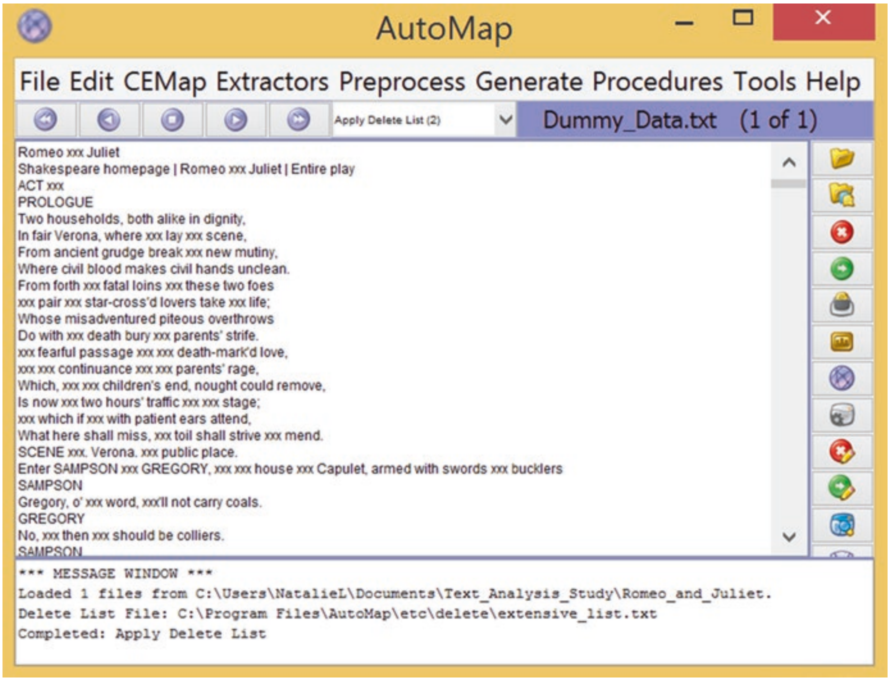
Romeo and Juliet
Shakespeare homepage | Romeo and Juliet | Entire play
ACT I
PROLOGUE
Two households, both alike in dignity,
In fair Verona, where we lay our scene,
From ancient grudge break to new mutiny,
Where civil blood makes civil hands unclean.
From forth the fatal loins of these two foes
A pair of star-cross'd lovers take their life;
Whose misadventured piteous overthrows
Do with their death bury their parents' strife.
The fearful passage of their death-mark'd love,
And the continuance of their parents' rage,
Which, but their children's end, nought could remove,
Is now the two hours' traffic of our stage;
The which if you with patient ears attend,
What here shall miss, our toil shall strive to mend.
SCENE I. Verona. A public place.
Enter SAMPSON and GREGORY, of the house of Capulet, armed with swords and bucklers
SAMPSON
Gregory, o' my word, we'll not carry coals.
GREGORY
No, for then we should be colliers.
SAMPSON

```
*** MESSAGE WINDOW ***
Loaded 1 files from C:\Users\NatalieL\Documents\Text_Analysis_Study\Romeo_and_Juliet.
```

Fig. 5.5 *Romeo and Juliet* imported into AutoMap

interfere in the software's ability to identify meaningful patterns within the data. These are usually high frequency words that do not have a lot of significance for most datasets such as articles, conjunctions, pronouns, number words, contractions, simple verbs, and prepositions. Many text analyzers have built-in stop word lists (also called "delete" lists), but a researcher can also create his or her own by making a list of words that are known to be frequent within a dataset but do not add value to the analysis.

To perform stop word removal on the *Romeo and Juliet* dataset within AutoMap, the data analyst clicked on ***Preprocess—Text Refinement—Apply Delete List***, and clicked ***Confirm***. She used the standard AutoMap delete list (which contains the most frequently-occurring words within the English language: a, an, and, as, at, but, for, he, her, hers, him, his, etc.), although she could have edited the delete list within AutoMap to create a custom list. The next step was to select *Rhetorical* as the type of delete processing because this setting inserts a placeholder, *xxx,* into the data so that the analyst can see which words were removed due to this procedure. Rhetorical delete processing also preserves the distance between words so that two words are not considered closer together after the words on the delete list that exist between them are removed. The analyst clicked ***OK***, and the results of delete list application can be seen in Fig. 5.6. Note that you can see the list of procedures that have been performed on the data so far in the Message Window.


AutoMap
File Edit CEMap Extractors Preprocess Generate Procedures Tools Help
Apply Delete List (2)
Dummy_Data.txt (1 of 1)

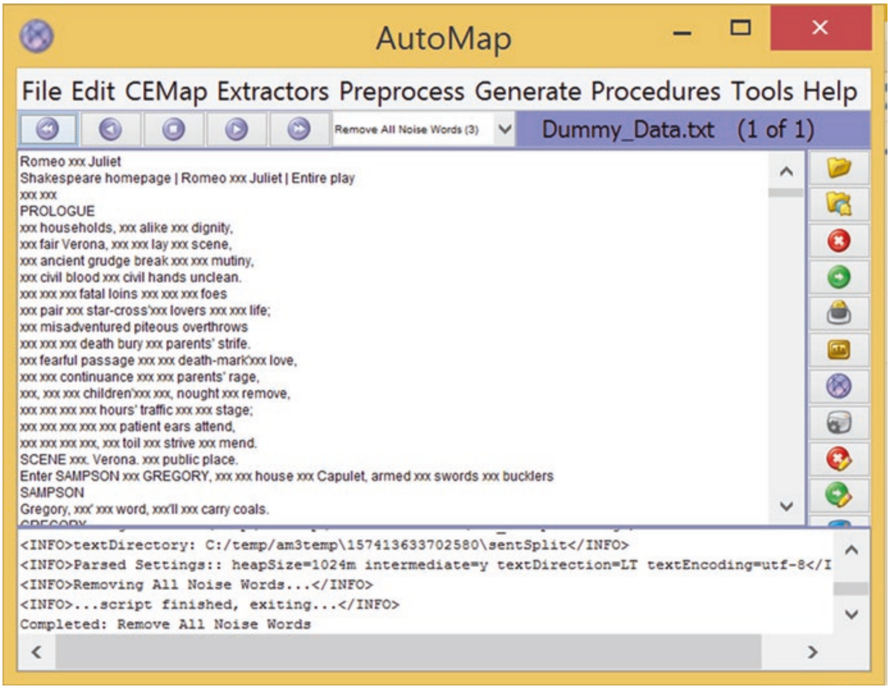
```
Romeo xxx Juliet
Shakespeare homepage | Romeo xxx Juliet | Entire play
ACT xxx
PROLOGUE
Two households, both alike in dignity,
In fair Verona, where xxx lay xxx scene,
From ancient grudge break xxx new mutiny,
Where civil blood makes civil hands unclean.
From forth xxx fatal loins xxx these two foes
xxx pair xxx star-cross'd lovers take xxx life;
Whose misadventured piteous overthrows
Do with xxx death bury xxx parents' strife.
xxx fearful passage xxx xxx death-mark'd love,
xxx xxx continuance xxx xxx parents' rage,
Which, xxx xxx children's end, nought could remove,
Is now xxx two hours' traffic xxx xxx stage;
xxx which if xxx with patient ears attend,
What here shall miss, xxx toil shall strive xxx mend.
SCENE xxx. Verona. xxx public place.
Enter SAMPSON xxx GREGORY, xxx xxx house xxx Capulet, armed with swords xxx bucklers
SAMPSON
Gregory, o' xxx word, xxx'll not carry coals.
GREGORY
No, xxx then xxx should be colliers.
SAMPSON
```

```
*** MESSAGE WINDOW ***
Loaded 1 files from C:\Users\NatalieL\Documents\Text_Analysis_Study\Romeo_and_Juliet.
Delete List File: C:\Program Files\AutoMap\etc\delete\extensive_list.txt
Completed: Apply Delete List
```

Fig. 5.6 Delete list applied to *Romeo and Juliet* dataset

Further preprocessing can be done by using the ***Preprocess—Text Preparation*** functions such as ***Remove Numbers as Words or Remove All Noise Words***. The analyst chose to apply ***Remove All Noise Words*** to the *Romeo and Juliet* dataset because this procedure removes pronouns, verbs, possessives, number words and other words types that researchers often find beneficial to remove from their textual data before analysis. The amount and type of preprocessing that should be performed depends on the dataset and what the researcher wishes to learn from it. For example, in some datasets pronouns could be important indicators of personal identification, and inclusion of all verbs might be important for analysis of storytelling or for identifying time phases. It is up to the researcher to evaluate the benefits and impact of specific preprocessing techniques on a particular dataset. Figure 5.7 shows the dataset after all noise words were removed.

The second preprocessing technique, stemming, involves identifying the root of a word and then standardizing all the various endings that come after a root in order to avoid separate counts of a word that has different forms but the same meaning. For example, the words “live,” “lived,” and “lives” would all be considered unique words by a text analyzer unless the analyst performed preprocessing like stemming that can reconcile these differences within the dataset. After performing stemming, the root of these words, “live”, would take the place of all other forms of the word within the dataset. The analyst applied stemming to the *Romeo and Juliet* dataset in


AutoMap
File Edit CEMap Extractors Preprocess Generate Procedures Tools Help
Remove All Noise Words (3)
Dummy_Data.txt (1 of 1)

```
Romeo xxx Juliet
Shakespeare homepage | Romeo xxx Juliet | Entire play
xxx xxx
PROLOGUE
xxx households, xxx alike xxx dignity,
xxx fair Verona, xxx xxx lay xxx scene,
xxx ancient grudge break xxx xxx mutiny,
xxx civil blood xxx civil hands unclean.
xxx xxx xxx fatal loins xxx xxx xxx foes
xxx pair xxx star-cross'xxx lovers xxx xxx life;
xxx misadventured piteous overthrows
xxx xxx xxx death bury xxx parents' strife.
xxx fearful passage xxx xxx death-mark'xxx love,
xxx xxx continuance xxx xxx parents' rage,
xxx, xxx xxx children'xxx xxx, nought xxx remove,
xxx xxx xxx xxx hours' traffic xxx xxx stage;
xxx xxx xxx xxx xxx patient ears attend,
xxx xxx xxx xxx, xxx toil xxx strive xxx mend.
SCENE xxx. Verona. xxx public place.
Enter SAMPSON xxx GREGORY, xxx xxx house xxx Capulet, armed xxx swords xxx bucklers
SAMPSON
Gregory, xxx' xxx word, xxx'll xxx carry coals.
```

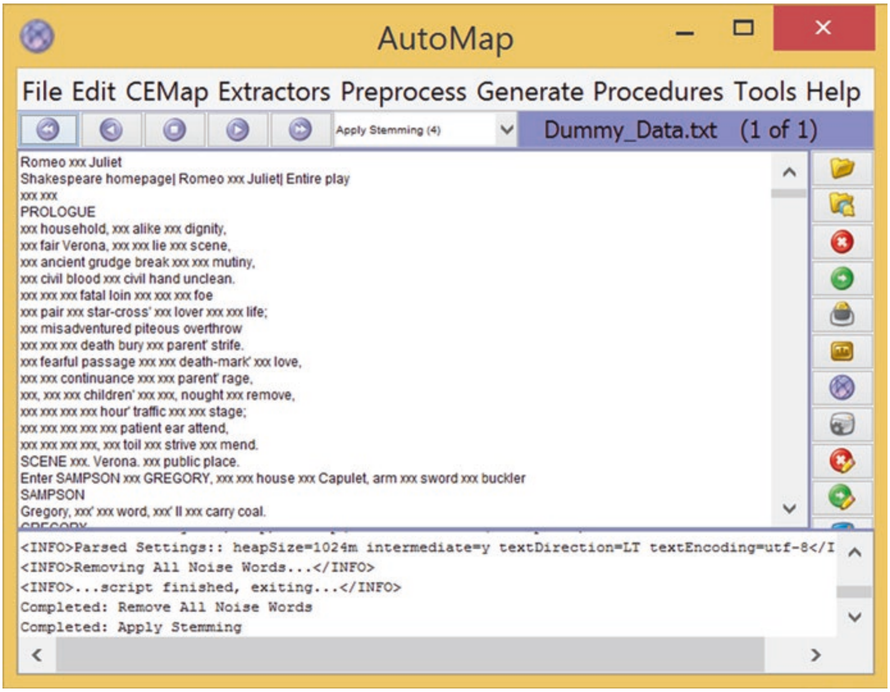
```
<INFO>textDirectory: C:/temp/am3temp\157413633702580\sentSplit</INFO>
<INFO>Parsed Settings:: heapSize=1024m intermediate=y textDirection=LT textEncoding=utf-8</I
<INFO>Removing All Noise Words...</INFO>
<INFO>...script finished, exiting...</INFO>
Completed: Remove All Noise Words
```

Fig. 5.7 *Romeo and Juliet* dataset after all noise words removed

order to demonstrate this procedure. In AutoMap stemming is conducted by clicking on ***Preprocess—Text Refinement—Apply Stemming***. The analyst used the default *K-stemmer*, clicked ***OK***, chose the default option to include capitalized words in stemming, and clicked ***OK*** again. The results (Fig. 5.8) show that verbs have been converted to their root form, so that "lay" became "lie." Plural nouns like "ears" were converted to singular nouns, and all words not in their root form were brought to their root form. It is now much less likely that words with the same meaning will be analyzed separately because of grammar or conjugation factors.

5.3.4 Data Analysis

Text Corpus Statistics. Now that the *Romeo and Juliet* dataset has been preprocessed, the simplest type of exploratory analysis that can be done is generation of a concept list. As mentioned earlier, a concept list is a inventory list of the words that appear within a text corpus along with a count of each word's frequency and other attribute information. The analyst generated a concept list for the *Romeo and Juliet* dataset by clicking on ***Generate—Concept List—Concept List (Per Text)***. AutoMap's request to "Select Directory for Concept Lists" asks the analyst to select


AutoMap
File Edit CEMap Extractors Preprocess Generate Procedures Tools Help
Apply Stemming (4)
Dummy_Data.txt (1 of 1)

```
Romeo xxx Juliet
Shakespeare homepage| Romeo xxx Juliet| Entire play
xxx xxx
PROLOGUE
xxx household, xxx alike xxx dignity,
xxx fair Verona, xxx xxx lie xxx scene,
xxx ancient grudge break xxx xxx mutiny,
xxx civil blood xxx civil hand unclean.
xxx xxx xxx fatal loin xxx xxx xxx foe
xxx pair xxx star-cross' xxx lover xxx xxx life;
xxx misadventured piteous overthrow
xxx xxx xxx death bury xxx parent' strife.
xxx fearful passage xxx xxx death-mark' xxx love,
xxx xxx continuance xxx xxx parent' rage,
xxx, xxx xxx children' xxx xxx, nought xxx remove,
xxx xxx xxx xxx hour' traffic xxx xxx stage;
xxx xxx xxx xxx xxx patient ear attend,
xxx xxx xxx xxx, xxx toil xxx strive xxx mend.
SCENE xxx. Verona. xxx public place.
Enter SAMPSON xxx GREGORY, xxx xxx house xxx Capulet, arm xxx sword xxx buckler
SAMPSON
Gregory, xxx' xxx word, xxx' ll xxx carry coal.
```

```
<INFO>Parsed Settings:: heapSize=1024m intermediate=y textDirection=LT textEncoding=utf-8</I
<INFO>Removing All Noise Words...</INFO>
<INFO>...script finished, exiting...</INFO>
Completed: Remove All Noise Words
Completed: Apply Stemming
```

Fig. 5.8 The *Romeo and Juliet* dataset after stemming was applied

an output folder where he or she wants the results of the analysis to be stored. The analyst should only click once on the output folder to highlight it, then click ***Select***. The next window allows the analyst to specify some concept list generation parameters. For this example the analyst used the default parameters and then clicked ***Confirm***. AutoMap gives the option to open the concept list in its built-in viewer window, but the user can also navigate to the output folder on his or her computer where a new folder, *Concept List1*, has been created to store the concept list. The concept list is created as a Microsoft Excel file, which makes it convenient to sort the list according to the frequency that a word appears in the dataset, or according to any other attribute assigned by the researcher. The analyst opened the file in Excel and then sorted the list by frequency, as seen in Table 5.1. The concept list shows each word within the corpus, a count of how frequently it occurred within the corpus, and a relative frequency score compared to the concept that occurred most often in the corpus.

Even though the concept list is a very simple text mining method, it does reveal some meaningful information about the data, especially for people who have never read *Romeo and Juliet* or seen the play performed. The concept list can be interpreted as evidence that a large amount of the text is devoted to a love story in which two characters, Romeo and Juliet, factor highly. The list indicates that night may be an important time or setting of the play, and that a nurse, friar, and people named Mercutio, Benvolio, and Laurence are important characters. The word "death"

Table 5.1 Concept list generated from the *Romeo and Juliet* dataset

	A	B	C
1	Concept	Frequency	relative_frequency
2	ROMEO	180	1
3	love	155	0.8611111
4	thy	150	0.8333333
5	thee	138	0.76666665
6	JULIET	134	0.74444443
7	Romeo	130	0.7222222
8	CAPULET	119	0.6611111
9	Nurse	114	0.6333333
10	II	91	0.50555557
11	BENVOLIO	74	0.41111112
12	night	73	0.40555555
13	Enter	72	0.4
14	FRIAR	70	0.3888889
15	MERCUTIO	69	0.38333333
16	man	69	0.38333333
17	LAURENCE	65	0.3611111
18	good	65	0.3611111
19	death	64	0.35555556
20	LADY	62	0.34444445

appears in the text relatively frequently, and so the analyst might assume that one or several characters die—this might therefore be a romantic tragedy. The concept list provides only a very basic understanding of the play that is divorced from its prose and plot, but perhaps through this example the reader can now visualize how text mining can aid researchers in extracting meaningful information from text corpuses much larger than a play that would otherwise take weeks to read through and summarize.

The concept list also points out where preprocessing improvements are necessary. The list shows that ROMEO and Romeo were counted separately by AutoMap. This result can be considered useful in that it distinguishes between the play formatting made in all caps and the verbal references to Romeo that appeared during the play, but it could also be considered an error if the analyst’s goal was to count all mentions of Romeo together. “Thy” and “thee” show up at the top of the list because the delete list was created with modern language in mind. These pronouns should be added to the delete list and preprocessing rerun. The word “ll” needs to be investigated since it may be a result of stemming or could be part of an archaic word in the text that could not be properly preprocessed. *As the reader can see, text mining analyses must often be repeated multiple times in order to refine preprocessing to suit the nuances of each dataset.* Conversely, in some cases the analyst may want to do very little preprocessing in order to preserve all variation within the data for analysis. This was the case for the medical consultation dataset. Due to its smaller

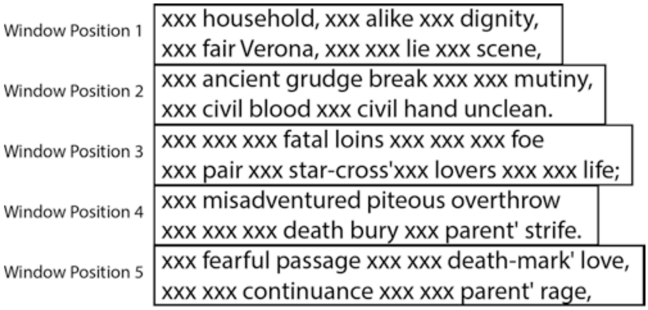

Window Position 1
xxx household, xxx alike xxx dignity,
xxx fair Verona, xxx xxx lie xxx scene,
Window Position 2
xxx ancient grudge break xxx xxx mutiny,
xxx civil blood xxx civil hand unclean.
Window Position 3
xxx xxx xxx fatal loins xxx xxx xxx foe
xxx pair xxx star-cross'xxx lovers xxx xxx life;
Window Position 4
xxx misadventured piteous overthrow
xxx xxx xxx death bury xxx parent' strife.
Window Position 5
xxx fearful passage xxx xxx death-mark' love,
xxx xxx continuance xxx xxx parent' rage,

Fig. 5.9 Illustration created to demonstrate how AutoMap creates "windows" to extract word pairs during semantic network generation

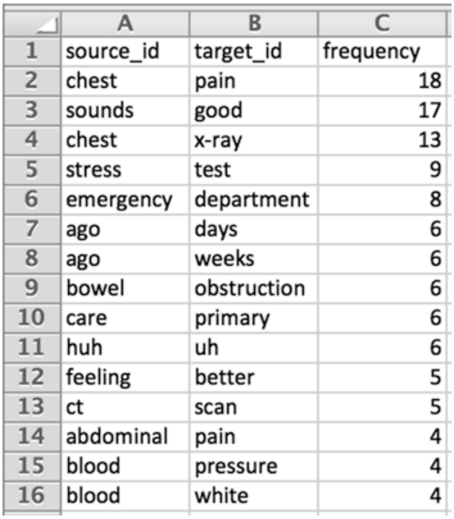
size, stemming made it impossible to detect the nuances of conversations surrounding similar medical consultation topics. As a result the analyst only performed stop word removal and removal of numbers as words when preprocessing the medical consultation dataset.

Semantic Network Analysis. This tutorial now returns to the medical consultation dataset in order to demonstrate how to generate a semantic network from a text corpus. Generation of a concept list using the medical consultation dataset revealed that "pain" was a frequently occurring word within the text corpus as was "goodbye." The research team wanted to know more about the context of these and other frequently occurring words, and so the team's data analyst constructed a co-occurrence semantic network from the data.

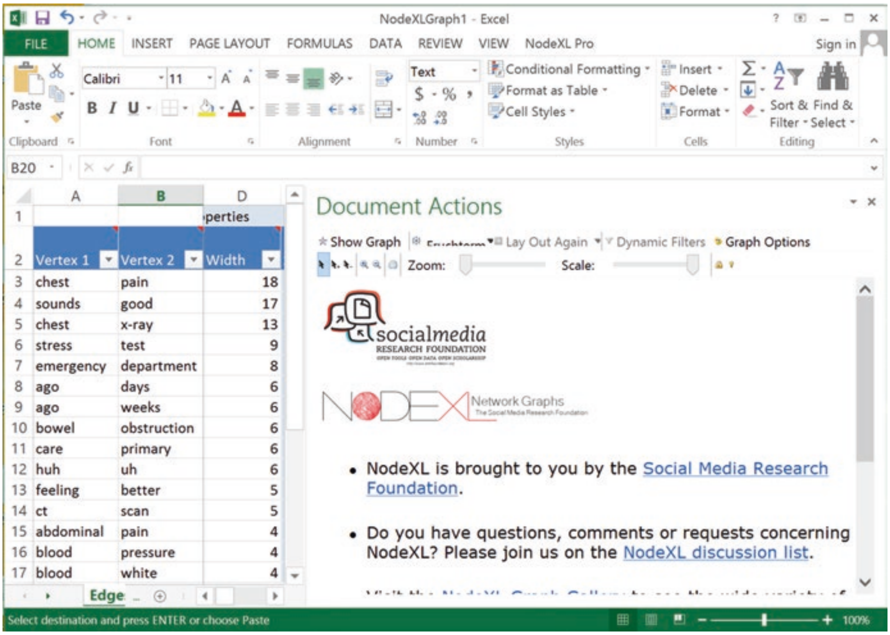
Co-occurrence semantic networks are based upon two key notions: (1) the idea that words that exist close to each other within a textual dataset are likely related in some way, and (2) that the meaning of a text corpus can be analyzed by constructing a network that represents all of the relationships between words in a dataset simultaneously. Take for example the sentence, "The patient complains of pain in his abdomen." Stop word removal would leave us with: "patient complains pain abdomen." Because these words occur near each other (within the same sentence), AutoMap makes note of their proximal relationship. The specific way in which the software does this is as follows. AutoMap creates a "window," the size of which is specified by the analyst (for example, two sentences or a paragraph in size) and then moves the window through the data, looking at the text that fits within the window and keeping track of the words that appear within the same window. (Figure 5.9 is an illustration of how a two-sentence window would move through the *Romeo and Juliet* dataset.) By repeating this procedure throughout the data, AutoMap collects a count of how many times a pair of words like "patient" and "pain" co-occur with one another within the same window. The resulting list of word pairs can be visualized as a network that connects all the pairs to one another so that if "pain" and "patient" co-occur frequently, and "chest" and "pain" co-occur frequently, one branch of the network will look like this: patient—pain—chest. In a network visualization, the lines that connect the words, called edges, can be used to represent how many times the same pair of words co-occurs within the dataset by thickening the width of the line to represent a greater frequency of co-occurrence.

The first step in constructing a co-occurrence semantic network is to click on ***Generate—Semantic Network—Semantic (Co-reference List)***. The analyst again clicked once on the output folder to select it, then clicked ***Next***. The network parameters window allows the analyst to make several decisions about how to generate the network. Directionality refers to whether the edge between two words represents a unidirectional (one-way flow or relationship) or bidirectional relationship (two-way flow or mutual relationship). For the medical consultation network, the analyst chose to setup the network as having bidirectional relationships because the research team wanted to discover the relationships between words within the medical consultation conversations without putting a word order constraint on the network. For their project, "doctor-patient" and "patient-doctor" could be counted as the same word pair because word order would not change which concepts were related topically to one another. Word order had the potential to cause variation in the meaning of these topics, but that was something the analyst was aware she would need to evaluate. Analysis of the network with no word order constraints was her team's best option for a first round of data analysis. Therefore, if two words co-occurred within the same window, the software noted their mutual, proximal relationship. If the team had been interested in identifying frequently-occurring phrases, they would have needed to preserve the order of words within each sentence and would have chosen instead to generate a unidirectional network. The analyst selected the window size as a two sentence window because of the small size of the text corpus, left the other parameters at their default values, and clicked ***Confirm***. This analysis generates a folder, ***SemanticList1***, within the output folder. The output itself is an Excel file containing two columns that represent pairs of words extracted using the described windowing method, along with a column that is a record of how frequently each pair of words occurred within the text corpus (Fig. 5.10).

Fig. 5.10 The semantic word pair list resulting from semantic network generation from the medical consultation dataset



	A	B	C
1	source_id	target_id	frequency
2	chest	pain	18
3	sounds	good	17
4	chest	x-ray	13
5	stress	test	9
6	emergency	department	8
7	ago	days	6
8	ago	weeks	6
9	bowel	obstruction	6
10	care	primary	6
11	huh	uh	6
12	feeling	better	5
13	ct	scan	5
14	abdominal	pain	4
15	blood	pressure	4
16	blood	white	4



	A	B	D
1			perties
2	Vertex 1	Vertex 2	Width
3	chest	pain	18
4	sounds	good	17
5	chest	x-ray	13
6	stress	test	9
7	emergency	department	8
8	ago	days	6
9	ago	weeks	6
10	bowel	obstruction	6
11	care	primary	6
12	huh	uh	6
13	feeling	better	5
14	ct	scan	5
15	abdominal	pain	4
16	blood	pressure	4
17	blood	white	4

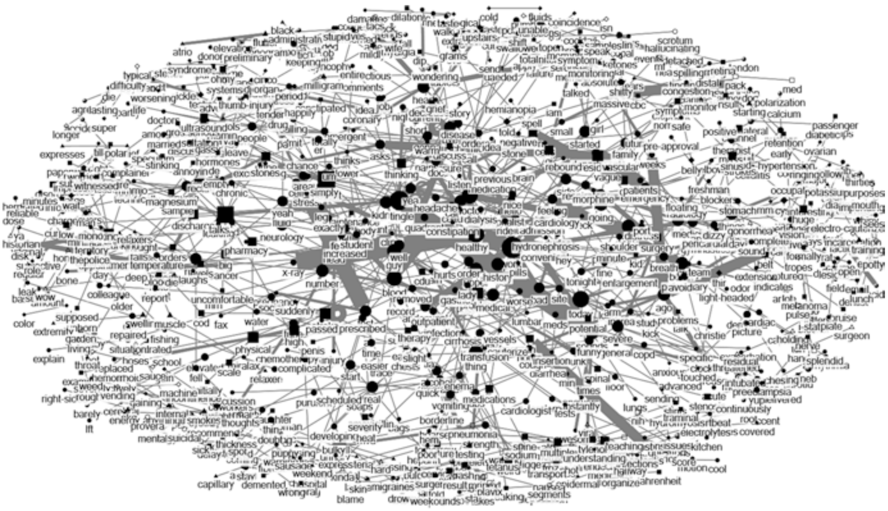
- NodeXL is brought to you by the Social Media Research Foundation.
- Do you have questions, comments or requests concerning NodeXL? Please join us on the NodeXL discussion list.

Select destination and press ENTER or choose Paste

Fig. 5.11 Transfer of the semantic list words pairs and word pair frequency into NodeXL

The next step was to visualize the semantic network using a network visualization tool. The analyst used NodeXL (Smith et al., 2010), which can be downloaded from: https://nodexl.codeplex.com/. An easy way to import the data into NodeXL is to delete the column headers, "source_id," "target_id," and "frequency" from the semantic list, and then copy and paste all the remaining cells in column A of the semantic list into *Vertex 1* under the "Edge" tab in NodeXL. The remaining cells in column B should be pasted under *Vertex 2* (see Fig. 5.11). It is important to make sure that the word pairs match up with one another in the NodeXL spreadsheet the same way they do in the semantic word pair list.

NodeXL's *Width* column is used to display the frequency of each word pair, and it does this visually by adjusting the relative width of the edges linking words in the network map. The analyst copied and pasted the frequency column of the semantic list into the *Width* column in NodeXL. Next, under the "NodeXL" tab at the top of the page, she selected ***AutoFill Columns*** and selected ***Vertex Label*** from the "Vertex" drop down menu and clicked ***AutoFill***. This feature displays the words as labels on the graph. Next, the analyst pressed "Show Graph" in the Document Actions Pane to view the semantic network (Fig. 5.12). An initial network visualization is often uninterpretable because of the many overlapping words and connections. The analyst chose to analyze the underlying structure of the medical consultations network by looking for evidence of subconversations. The procedure

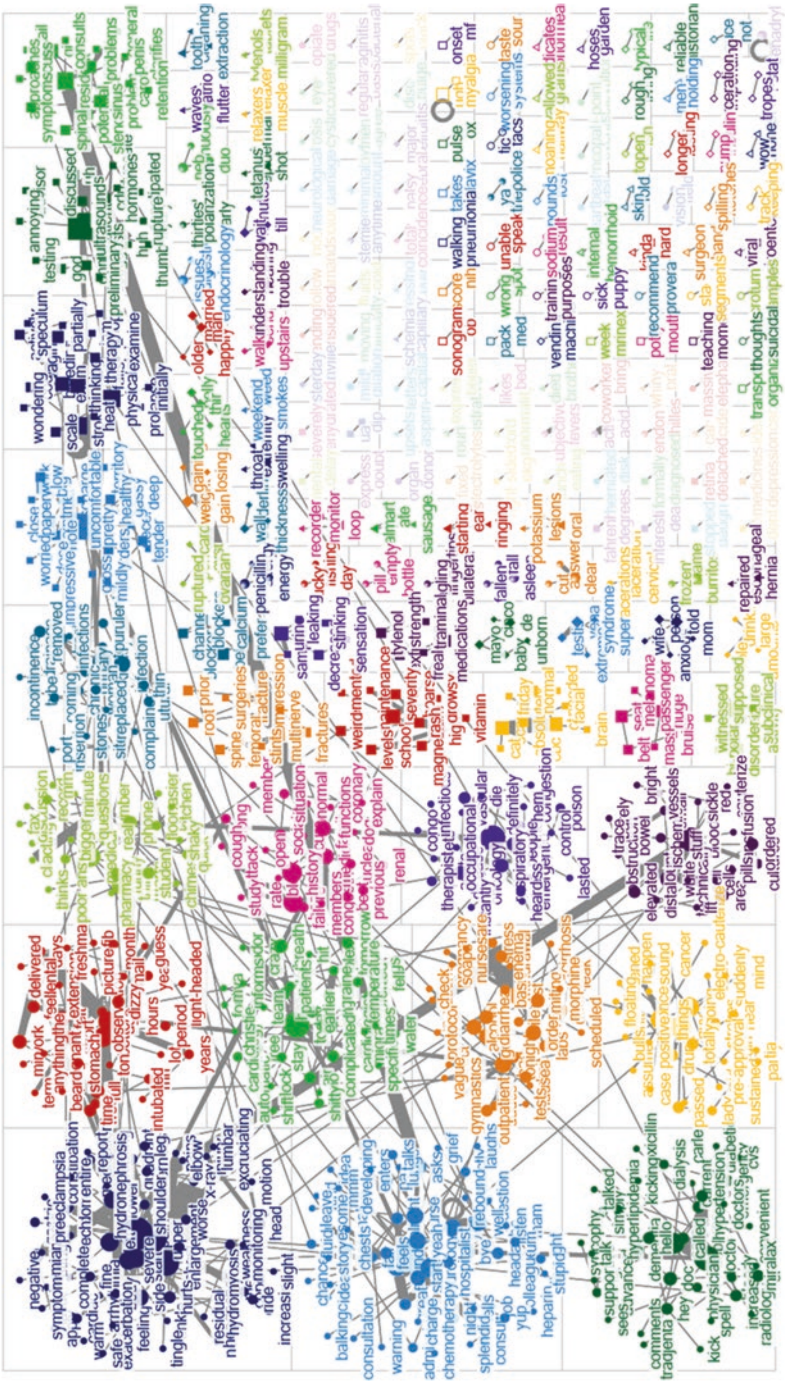


Created with NodeXL Pro (http://nodexl.codeplex.com) from the Social Media Research Foundation (http://www.smrfoundation.org)

Fig. 5.12 The semantic network generated from the medical consultation dataset

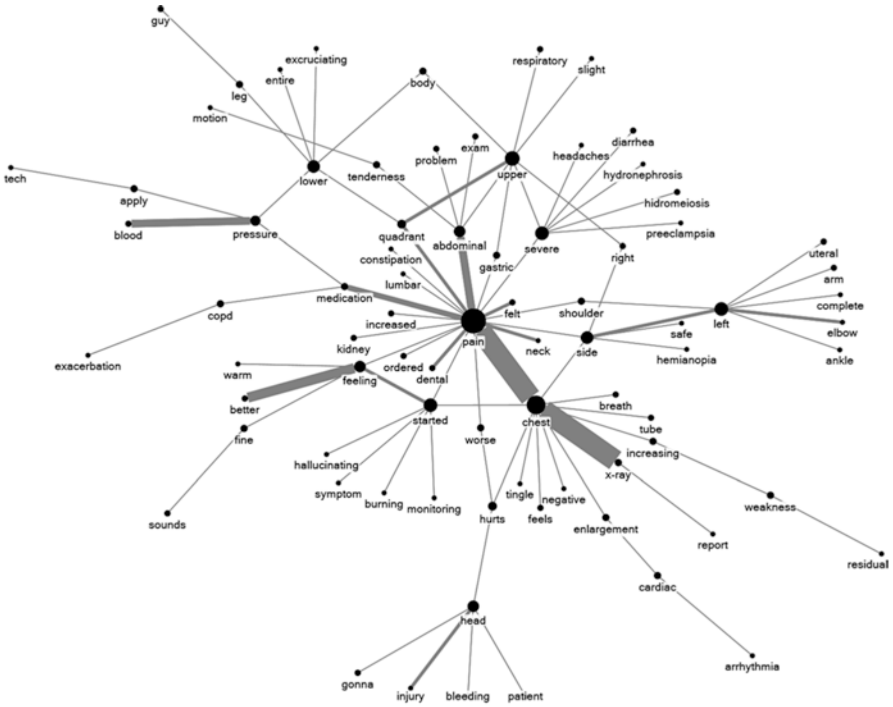
used to do this was cluster analysis, which is run by going to the "NodeXL" tab, clicking on ***Groups—Group by Cluster***, and in this case the analyst chose to group the words using the Clauset-Newman-Moore (2004) cluster algorithm. Under the Document Actions Pane she used the layout drop-down menu to select ***Layout Options***, and chose ***Lay out each of the graph's groups in its own box***. Clicking on "Refresh Graph" visualizes the semantic network clusters (Fig. 5.13).

Each of the groups displayed in the visualization of the cluster analysis have been grouped together by the algorithm because the words within each group co-occur with one another more frequently than they do with other words. Each of the groups extracted from the medical consultation dataset represented a conversation topic that arose during the physicians' medical consultations. The analyst examined the individual groups by clicking on the "Groups" tab on the bottom of the NodeXL worksheet, and then clicked on "G1" in the Groups column to highlight the largest group. She exported this group by clicking on ***Export—Selection to New NodeXL Workbook***. This procedure opened up a new NodeXL workbook containing only this group's data. Switching the layout algorithm to Harel-Koren Fast Multiscale (Koren, 2002) and hitting ***Refresh*** made the network structure easier to view. The analyst also clicked on individual words (represented as circular nodes) to adjust the graph image manually so that there were no overlapping or obstructed words. Figure 5.14 shows the subnetwork generated through this process. Figure 5.15 is the second largest subgroup, which was extracted using the same method performed on Subgroup 1.



Created with NodeXL Pro (http://nodexl.codeplex.com) from the Social Media Research Foundation (http://www.smrfoundation.org)

Fig. 5.13 The medical consultation dataset semantic network grouped by cluster



Created with NodeXL (http://nodexl.codeplex.com)

Fig. 5.14 Subgroup 1: Emergency department physician's medical consultations revolving around pain diagnosis and management

5.3.5 Interpretation

When first undertaking interpretation of the results of a semantic network analysis, it is important to remember that during this method, "word associations in texts were analyzed, and those word associations represent[] the meaning inherent to the data" (Doerfel, 1998, p. 23). The resulting graph, such as those in Figs. 5.12 and 5.13, as well as any other metrics or information gained through the analysis, explain something about the relationships between words in the text. *However, the meaning of these relationships can only be gained through interpretation of the results.* For example, finding that the words "sounds" and "good" co-occur frequently within the medical consultations dataset is a meaningless piece of information unless interpretation is done to connect this result back to the data context, the nature of the text archive, and any theoretical frameworks used to collect or interpret the data.

The analyst's interpretation strategy is usually a function of what analyses were performed on the text corpus. This tutorial's example utilized a cluster analysis, and so interpretation of the results will largely focus on interpreting the semantic graphs

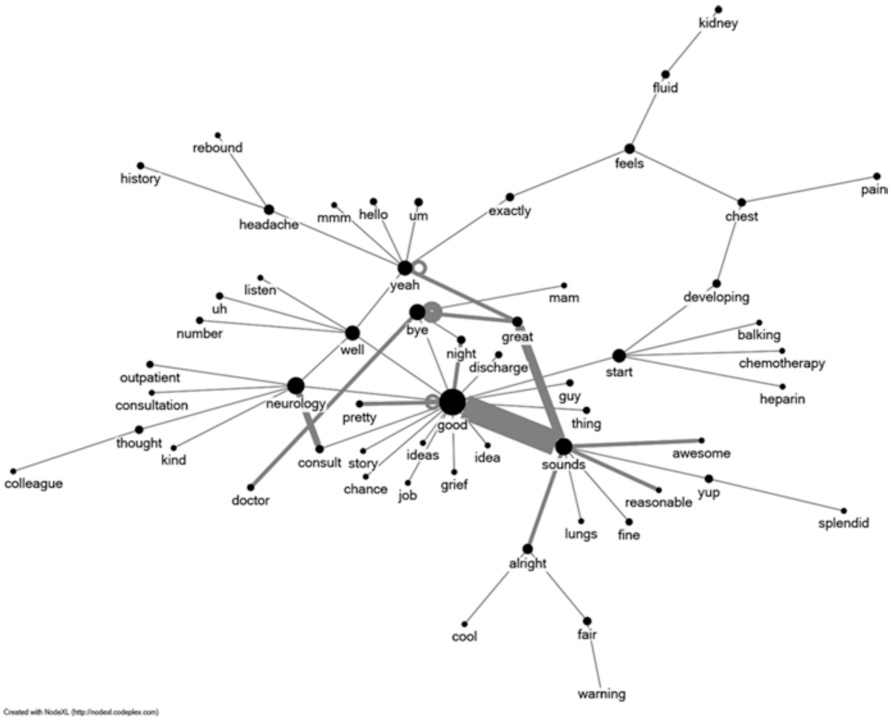


Fig. 5.15 Subgroup 2: Emergency department physician's medical consultations revolving around feedback and affirmation of treatment plans

in terms of what medical consultation conversation topics they indicate. Because so little is known about topics of medical consultations, each conversation cluster should also be evaluated in terms of how these topics manifest within the larger context of the original dataset. As was mentioned earlier, AutoMap linked words when two words existed within the same window frame. While it is likely that words that existed near each other in the text are related in a meaningful way, there is no guarantee that this is the case. Therefore, the prominent, and seemingly meaningful words pairs identified by the network graphs should be searched for within the text corpus to make sure that there are in actuality meaningful relationships between the word pairs.

Some researchers choose to focus on the calculation of graph metrics in order to understand a text corpus, and such metrics should be interpreted in terms of what they explain about the relationships between words or concepts within a dataset. Graph metrics can be calculated at the individual word level (node level metrics) to understand how many connections exist between particular pairs of words. Metrics can also be calculated to understand qualities of the overall graph (graph level metrics). Once again, the simple reporting of a metric like the number of connections between particular words pairs is not enough—the analyst should endeavor to interpret what meaning is indicated by strong or weak connections between word pairs. The researcher may ask: Are there many connections between specific words pairs because they are a common phrase, are they instead two highly-connected concepts, or is there some other reason the words frequently co-occurred? For example, Atteveldt (2008) examined news stories to determine whether words associated with the word “Muslim” changed in news coverage after 9/11. The author found that the word “Muslim” was paired with terrorism-related words in news stories significantly more frequently after 9/11, but that other terror events did not cause an increase in these words’ associations. Atteveldt drew on framing theory when interpreting these word associations, finding that “the associative frame between Muslims and terrorism was created not by local events, but rather by 9/11 as a global event” (2008, p. 88).

Just as there are many ways to conduct text mining, there are many approaches to interpreting the results of a text mining study. Overall, the analyst’s goal during interpretation should be to: (1) identify patterns generated from the results, (2) confirm that these patterns are true representations of the original text corpus, and (3) interpret these patterns to explain what they represent or mean within the context of the dataset; how they answer a hypothesis or research question; how they can be explained using a theoretical framework; or how the patterns form the grounds for new theory development.

Interpretation of the Medical Consultation Semantic Network Analysis. To briefly review, the results of the text mining and subsequent semantic network analysis revealed the most common communication topics that small groups of physicians in an emergency department discussed as they enacted their expertise to coordinate patient care during medical consultations. These communication patterns were extracted by conducting cluster analysis of word associations within the semantic network. Each cluster contained a group of words that frequently co-occurred with each other and therefore had stronger relationships with one another than they had with other words within the text corpus. The final step of this text mining example is to interpret these patterns.

The largest subgroup (Fig. 5.14), showed the research team that a primary topic of medical consultations for their dataset was the diagnosing and managing of pain. This network graph visually represents all medical consultations in the dataset related to pain. The network graph can be read by starting at the center of the image and tracing the connections outward. In this manner, it is possible to see how consultations regarding chest pain led to the ordering of x-rays and the need for subsequent reports. There are many conversation paths radiating out from the pain node that have to do with describing the exact location of a patient’s pain. How pain started and the words patients use to describe the sensation of pain are all parts of this medical consultation topic. From this network, the research team learned about the many ways in which emergency department physicians investigate and treat pain. In terms of the study’s goal of understanding physicians’ expertise, Subgroup 1 in Fig. 5.14 was interpreted as evidence that the diagnosis and treatment of patients’ pain is a primary area of emergency physicians’ professional expertise.

This finding was very interesting to the research team because even though they had read through the transcriptions many times, none of the team members had recognized pain as a concept of interest within the dataset. This study illustrates the fact that even though word co-occurrence and frequency are rather simple ways of tallying the presence of words and the relationships between them, this method can help researchers to gain an entirely new perspective of a textual dataset.

The second most dominant pattern found through the semantic network analysis was Subgroup 2 (Fig. 5.15). This subgroup graph displays all conversations that have the phrases "sounds good" or "sounds great" in common, and like Subgroup 1, the graph shows the variations in conversations surrounding these terms. The many other affirmative phrases within this network like "sounds great," "sounds alright," and "yeah" led the research team to interpret this medical conversation topic as evidence of the use of feedback loops by physicians during medical consultation conversations to confirm or affirm treatment plans. The team went back to the text corpus and examined the contexts in which such phrases took place, and this follow-up examination of the text corpus confirmed that these words were very much used by physicians to communicate mutual understanding during medical consultations. This subgroup was interpreted as evidence that feedback is a very important part of enacting expertise during medical consultations. Looking again at the original text archive, the researchers also found that all physician roles, from medical resident to senior physician, utilized these feedback loops, indicating that feedback is an integral component of medical consultation regardless of a physician's level of medical expertise. Although text mining findings primarily originate from analysis of the textual data itself, it is always advisable to collect several layers of information about the context of a textual dataset because this contextual information can help an analyst achieve a more meaningful interpretation of the text mining results.

5.4 Contributions

In this tutorial, text mining aided a research team working to understand physicians' expertise in several meaningful ways. First, the researchers initially hit a roadblock when analyzing their dataset using traditional qualitative thematic coding. The physicians' language contained a lot of jargon, and as outsiders to the medical world, the researchers had a very difficult time finding topical differences that could help them categorize the consultations. This study is also an example of how text mining is useful for small as well as large datasets when barriers exist to traditional analysis methods. The fact that the research team did not notice that pain was a common medical consultation theme when reading the text corpus is further proof of the value of even the simplest text mining procedures.

This study was also the first step towards building theories to explain how physicians enact expertise and how they communicate to manage patient care. Text mining was valuable in helping the researchers take this first step because it allowed them to conduct data-driven discovery in order to identify meaningful conversation topics without having to first develop hypotheses. So little was known about the content of medical consultations that it would have been difficult to form specific hypotheses. Knowing that they were conducting data-driven discovery, the research team carefully defined the scope of their data (medical consultations) and used text mining to explore their data for significant patterns of medical consultation conversations. The research team also conducted follow-up interviews with the physicians they observed during data collection in order to get the physicians' interpretation of the results. The combined quantitative and qualitative results of this study are helping the researchers to build empirically-driven communication and organizational theory. Text mining is also useful for testing theories by looking for patterns within a text corpus to see whether they support existing theory. Additionally, theory can be used as a framework for gathering textual data or for interpreting the results of text mining. Text mining is a very flexible method well suited to making theoretical advancements, but as was discussed earlier, the many choices the researcher makes during data collection, preprocessing, and analysis determine whether or not a text mining analysis ends up being a good fit for a particular research goal like the development of theory.

There are many more text mining procedures and techniques than the few introduced during this tutorial. After discovering that pain and feedback terms were very relevant words in the medical consultation dataset, the research team could conduct further text mining by using these terms as key words, conducting key word analysis in order to extract all words surrounding the words the previous analyses found to be important within the dataset. This approach would tell the research team more about the specific context surrounding these meaningful terms. The research team might be able to learn more about physicians' expertise by having emergency department physicians rate the individual medical consultations according to the level of expertise they represent and then analyze high and low expertise consultations separately in order to evaluate what really excellent consultations have in common and what features are associated with poorly done medical consultations. In a different study it might make sense to take time into consideration, dividing up a text corpus into time segments and analyzing each segment independently in order to understand how a phenomena of interest evolves or develops over time.

There are an infinite number of ways in which to conduct text mining, and this is both a strength of the method and a barrier to its adoption. There is no guarantee that any meaningful results will come from many hours of data formatting, preprocessing, and analysis because the patterns each text mining procedure looks for can be present or absent from a dataset—the analyst cannot know if there is any merit in running a procedure until the work has been invested in running it. The way in which a dataset has been collected also greatly influences the success of text mining. Text mining is often described as an excellent method for analyzing very large text corpuses, but if the text contained within a very large dataset does not have very much in common, text mining is unlikely to identify any patterns, or if it does, the patterns may be more a function of word prevalence within a certain language or context and not due to the existence of important patterns within the data. For example, text mining may find patterns within a text corpus comprised of 10,000 newspaper articles, but if the researcher did not choose newspaper articles that all focus on a specific issue or social phenomena, or if there are off-topic articles mixed in with the corpus, the results of text mining of this data are unlikely to be interpretable in a meaningful way. Even though text mining is a powerful computational tool, it must be combined with good data collection and preprocessing decisions made by a human being who understands exactly what each algorithm and procedure is doing to the data.

Text mining is a very useful tool for both academic research and practical applications in business, education, and individual contexts. It can be used to help analysts learn more about the exponentially-increasing text archives that are generated while we work, from online commenting and debates, through communication with friends and family, and during every online interaction and email we send. The benefits offered by text mining will increase as this method is utilized by people from many disciplines and fields, especially if those who use text mining continue to share the procedures and techniques they find to be useful. Although text mining has existed since the invention of the computer, it is still in its early stages of development and application by people who are not advanced programmers or software engineers. The potential of text mining will increase for everyone as it is adopted for novel applications by new users like readers of this chapter.

References

van Atteveldt, W. (2008). *Semantic network analysis: Techniques for extracting, representing and querying media content*. Charleston, SC: BookSurge Publishing.

Caputo, A., Giacchetta, A., & Langher, V. (2016). AIDS as social construction: Text mining of AIDS-related information in the Italian press. *AIDS Care*, *28*, 1171–1176.

Carley, K. (2001). AutoMap (version 3.0.10.41) [Computer software]. Pittsburg, PA: CASOS, Carnegie Mellon University. Retrieved from http://www.casos.cs.cmu.edu/projects/automap/index.php

Carley, K. M., Columbus, D., Bigrigg, M., Diesner, J., & Kunkel, F. (2010). AutoMap User's Guide 2010 (CMU-ISR-10). Carnegie Mellon University. Retrieved from http://www.casos.cs.cmu.edu/publications/papers/CMU-ISR-10-121.pdf

Choudhary, A. K., Oluikpe, P. I., Harding, J. A., & Carrillo, P. M. (2009). The needs and benefits of text mining applications on post-project reviews. *Computers in Industry*, *60*(9), 728–740.

Clauset, A., Newman, M. E. J., & Moore, C. (2004). Finding community structure in very large networks. *Physical Review E*, *70*.

Cohen, A. M., & Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, *6*(1), 57–71.

Collier, N., Doan, S., Kawazoe, A., Goodwin, R. M., Conway, M., Tateno, Y., … Shigematsu, M. (2008). BioCaster: Detecting public health rumors with a Web-based text mining system. *Bioinformatics*, *24*(24), 2940–2941.

Dasgupta, S., & Sengupta, K. (2016). Analyzing consumer reviews with text mining approach: A case study on Samsung Galaxy S3. *Paradigm*, *20*(1), 56–68.

Doerfel, M. L. (1998). What constitutes semantic network analysis? A comparison of research and methodologies. *Connections*, *21*(2), 16–26.

Fan, W., Wallace, L., Rich, S., & Zhang, Z. (2006). Tapping the power of text mining. *Communications of the ACM*, *49*(9), 76–82.

Kessler, C. S., Afshar, Y., Sardar, G., Yudkowsky, R., Ankel, F., & Schwartz, A. (2011). A prospective, randomized, controlled study demonstrating a novel, effective model of transfer of care between physicians: The 5 Cs of consultation. *Academic Emergency Medicine*, *19*, 968–974.

Koren, D. H. Y. (2002). A fast multi-scale method for drawing large graphs. *Journal of Graph Algorithms and Applications*, *6*(3), 179–202.

Lammers, J. C., Lambert, N. J., Abendschein, B., Reynolds-Tylus, T., & Varava, K. (2016). Expertise in context: Interaction in the doctors' room of an emergency department. In P. M. Leonardi, & J. W. Treem (Eds.), *Expertise in Organizations* (pp. 145–167). Oxford: Oxford University Press.

Lexalytics. (2015). Dealing with context in text mining [White paper]. Retrieved August 29, 2016, from Lexalytics: https://www.lexalytics.com/content/whitepapers/Lexalytics-WP-Context.pdf

Luhn, H. P. (1958). The automatic creation of literature abstracts. *IBM Journal of Research and Development*, *2*(2), 159–165.

Maughan, B. C., Lei, L., & Cydulka, R. K. (2011). ED handoffs: Observed practices and communication errors. *American Journal of Emergency Medicine*, *29*, 502–511.

Miner, G. (2012). *Practical text mining and statistical analysis for non-structured text data applications*. New York: Academic Press.

Smith, M., Ceni A., Milic-Frayling, N., Shneiderman, B., Mendes Rodrigues, E., Leskovec, J., & Dunne, C. (2010). NodeXL: A free and open network overview, discovery and exploration add-in for Excel 2007/2010/2013/2016. Retrieved from http://nodexl.codeplex.com/