Diedre Brown

INFO 640/Fall 2019 Data Analysis

Final Project

15 December 2019

Audience

The intended audience for this paper would be those interested in technology from a philosophical perspective.

Text Analysis of Lewis Carroll's *Alice in Wonderland*

Introduction

From voice-assistants like Siri and Alexa to credit loan predictors, artificial intelligence (AI) is not only integrated into the functioning of one's daily life, it is also influencing what one's life events may be. As a partial target of AI is to bring convenience to human life (Deng & Jiang 111), I argue that in order to achieve this we must gain a mindful understanding of how all animals learn; develop methodologies which analyze data by encoding for context and identification of biases; and develop technology and devices that, as philosopher Andy Clark states, function as "probabilistic-prediction-devices" like the brain (Clark, n.d.). When Alan Turing first proposed the question, 'Can machines think?' in "Computing machinery and intelligence," he highlighted that the key to developing artificial intelligence is to consider the structure of knowledge from human computational, behavioral, and cognitive perspectives (Turing 433-436; 458-460).

While much of the early "work on AI was inextricably intertwined with neuroscience and psychology" (Hassabis et al, 245), it is only recently that a complement of powerful technology and expanded research in neuroscience has developed to further the concepts of this early work.

Additionally, this detailed understanding of the brain's functions has spawned readily available technology that can more accurately mimic these functions than it could sixty years ago. However, one challenge of creating what some have termed "conscious computation for machines" (Dehaene et al), is this concept of creating intelligent "behavior that relies multiple memory systems" (Hassabis et al., 247). In the human body, these memory systems are multi-sensory (i.e. the nervous system) and mostly processed in and controlled by the brain. And, it only within the last thirty-years that cognitive neuroscientists have been able to identify episodic memory[1], particularly as it relates to experience replay in dream states (Bulkeley 52), as pivotal to the complex structure and function of these multiple memory systems (Pathman et al., 1-4).

Essential to this transfer from the episodic memory of the hippocampus to the permanent memory in the neocortex in the brain is sleep. In sleep, replay of episodic memories allow us to use our experiences as the basis of dreams. Through this replay, we infer logic by drawing ties to what we know and experimenting with hypothetical possibilities. In essence, what makes dreaming significant to humanity, is that it allow us to self-identify context and emotion and use this information in the process of making decisions. However, a challenge of dreams is that they are unique to an individual, and cannot be easily reproduced and shared. One caveat to this is fictional literature, as it is the recantation of an author's imagination or dream-like vision for the narrative. Through this understanding of fictional literature, this paper seeks to determine if descriptive text analysis of Alice's dream in Lewis Carroll's *Alice in Wonderland (AiW)* can illustrate the accurate meaning of events in Alice's dream. Specifically, I will examine the interactions between Alice and major and minor character grouping, respectively, for evidence of an episodic memory (emotion, spatial understanding, or change in time). If I am able to find

[1] Episodic memory allows us to recall specific past events along with their spatial and temporal contexts (Pathman et al, 1).

evidence of episodic memory, then future studies may be able to develop a methodical language or algorithm which can be used to train machines to acquire “human-like” episodic memory through analysis of fictional literature.

Research Design and Methods

The source material for this analysis is the 1916 illustrated edition of Lewis Carroll's *Alice in Wonderland* from Sam L Gabriel Sons & Company found on Project Guttenberg. The text was downloaded directly from Project Guttenberg through R as text object—a tibble of 1,299 rows of characters in one column.

Preliminary Evaluation of the *AiW* Document

Once the text object was transformed into a corpus of one document, preliminary frequency evaluation (

Figure 1 and Figure 2) revealed that the text had a mean of 2.58 word counts (min=1, max=163), contained a lot of italicized text that would need further cleaning for removal of symbols, and that a list of stop words including "Alice" would be necessary. The term "Alice" was counted 163 times and would skew evaluation of the text. To account for this, two corpuses were made—one which included the term "Alice" and one that did not. As a result of the term "Alice", two versions of stop word objects, term document matrixes (TDM), document term matrixes (DTM) were also made. When "Alice" was included in subsequent frequencies, the figures were normalized to account for her presence.

Characters and Dreams

While I initially proposed to evaluate the commonly co-occurring relationships between dream states and animals in *AiW*, this proposed a challenge for the project to be completed on time. Though the animals are significant character's throughout Alice's journey, they posed two problems:

1. No colocation between the animals and an identifiable dream state could be found.

2. As the preliminary frequency distribution illustrated, apart from the White Rabbit, the only other animals to be featured as prominently were humanoid. Similarly, references to animals and humans outside of Wonderland are continuously made.

To account for this, I imposed three rules to the analysis:

1. The analysis will only focus on the Wonderland dream period—from Alice's fall down the rabbit-hole to the moment she wakes up in her sister's lap.
2. No animals or humans outside of Wonderland will be considered. They will be turned into stop words.
3. Two groupings of animals (including humanoids) will exist:
    a. Group A. These are characters that showed up in higher frequency (10 or more counts), and include: The White Rabbit; The King; The Queen; The Duchess; The Mouse; The Caterpillar; and The Cheshire Cat.
    b. Group B characters (less than 10 counts) include: The Cook; The Mad Hatter; The Knave; Two-Five-Seven- "Playing" Guards; The Pigeon; The Duck, Lory, and Eaglet; The Dodo; The Doormouse; The March Hare; The Mock Turtle; The Gryphon; Bill the Lizard; and The Frog-Footman

Topic Modeling with Latent Dirichlet Allocation (LDA)

Topic modeling was used to identify clusters in the story and increase stop word lists. Topic models were taken where k=2, and 6, respectively. The choice to use six topics was based on the mean frequency of words (2.58 counts). I found that three words did not provide enough words to infer context to the topic. I decided to use seven words per topic to account for context and a word to add to the stop word lists. In preliminary testing, I did try to model k=12 topic models in hopes that this would correspond to the twelve chapters of the book. However, I found that by increasing topic models, I only produced more models that overlapped than were distinct.

Sentiment Analysis

While I had initially intended to use sentiment analysis to define the concordance of dream states and animals, the preliminary frequency results and word clouds showed that this would not be possible as keywords were predominantly character names and not dream states. This further supported the decision to revise the project to only consider the dream portion of the text, and to evaluate the interactions between Alice and characters for evidence of characteristics of an episodic memory. Sentiment analysis was then used to examine the overall text for positive-negative polarity and the "eight key emotions" established by Robert Plutchik based on evolution ("American Scientist—Plutchik," 2001).

N-Grams

N-Grams were used to confirm identity of the most frequently occurring characters.

Results and Reflection

While the *Alice in Wonderland* document was an interesting study, it did not provide evidence supporting character evidence that would indicate concepts of episodic memory in text analysis alone. What the analysis did illustrate is that prior knowledge of a text's plot is necessary to employ this type of logic.

Though not the result I was expecting, both frequency and sentiment analysis were fairly accurate in determining the major characters, which I outlined as Group A. This is useful for future studies on any text, as it will assist in isolating key characters.

Topic Modeling (Figure 6-Figure 9)

What was most surprising in the study of the topic models (tm), was that for both the two- and six- form topic models, the tm's that did not include the term 'Alice' were actually more predictive of events throughout the book. Though approximately 207 stop words were used, in future studies, I believe that the accuracy of these models would only increase as more stop words are added. From the frequency and n-gram studies, I would add any words that were less than three in count to the stop words list.

Sentiment Analysis (Figure 10 and Figure 11)

The sentiment analysis mirrored aspects of the topic modeling. In particular, it fairly accurately described most of the major Group A characters. Alice not included in the sentiment analysis because her terms were eliminated in the inner join to the NRC lexicon. While in future sentiment studies, I would endeavor to include Alice (perhaps as a n-gram term), her absence in these studies further supports the findings in the topic models. Additionally, what was also interesting is the "summary" provided by the Plutchik terms of the sentiment analysis. From these groupings, one could assume that *AiW* is a mostly positive tale of full of overwhelming anticipation, joy, and trust. Again, further study using increased stop words and the n-grams for sentiment analysis may lead to more insight of the context surrounding character interactions.

Though this study, as outlined, could not support evidence of episodic memory and was only able to partially describe the text, it did highlight design ideas for future studies of this text for episodic memory. These ideas include:

1. Breaking the analysis apart by chapter to see if this would differ from an overall view of the text.
2. Using frequency and weighted analysis to isolate main characters other than Alice. Perhaps, focusing in on them, whether individually or as a group.
3. Create a custom corpus that could be used as a sentiment lexicon. As characters were eliminated during the inner join with NRC, it is possible that by assigning sentiment to these characters, a more robust analysis may occur.

Appendix A. Figures

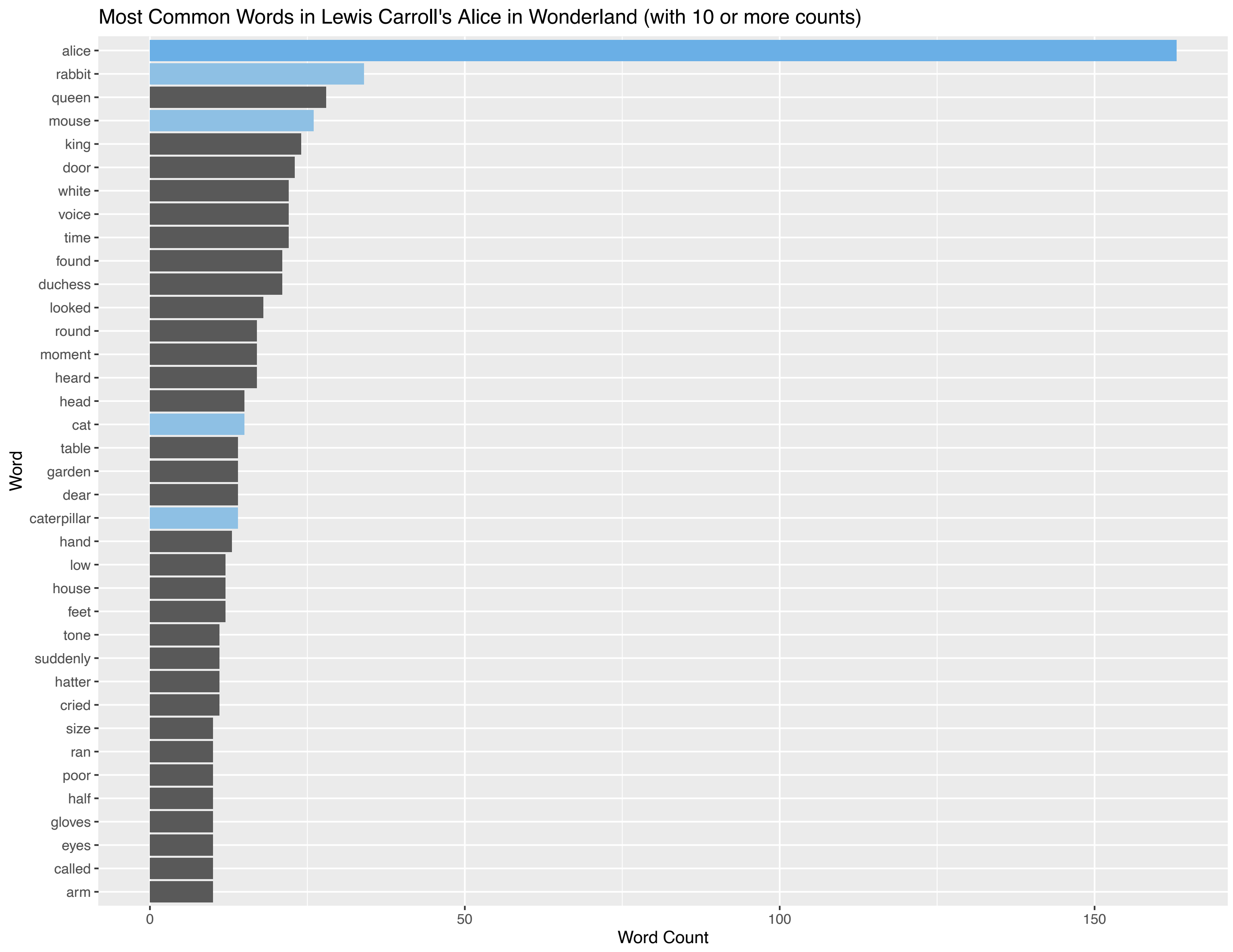

Figure 1. Preliminary Word Frequency Counts (10 or more counts)

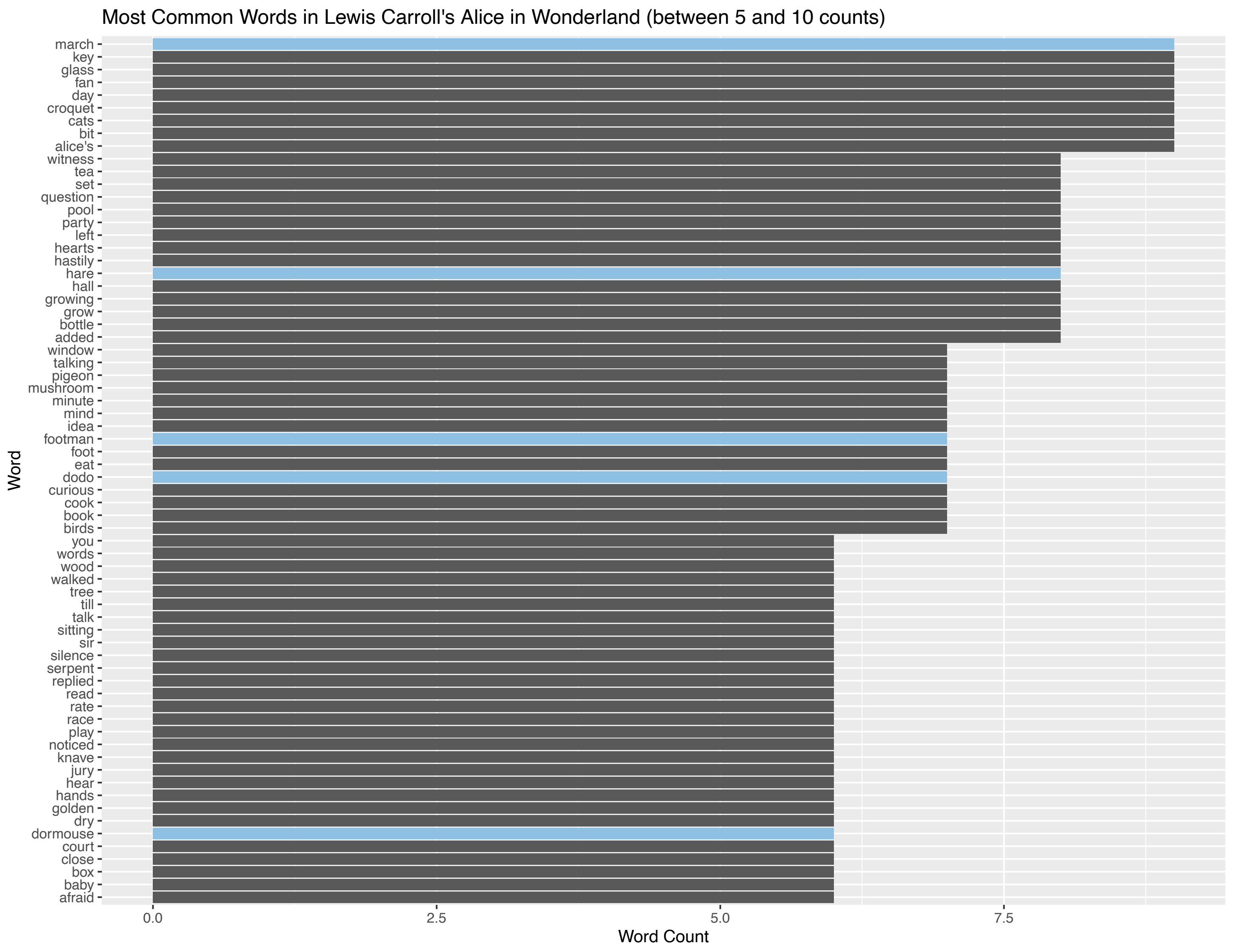

Figure 2. . Preliminary Word Frequency Counts (counts between 5 and 10)

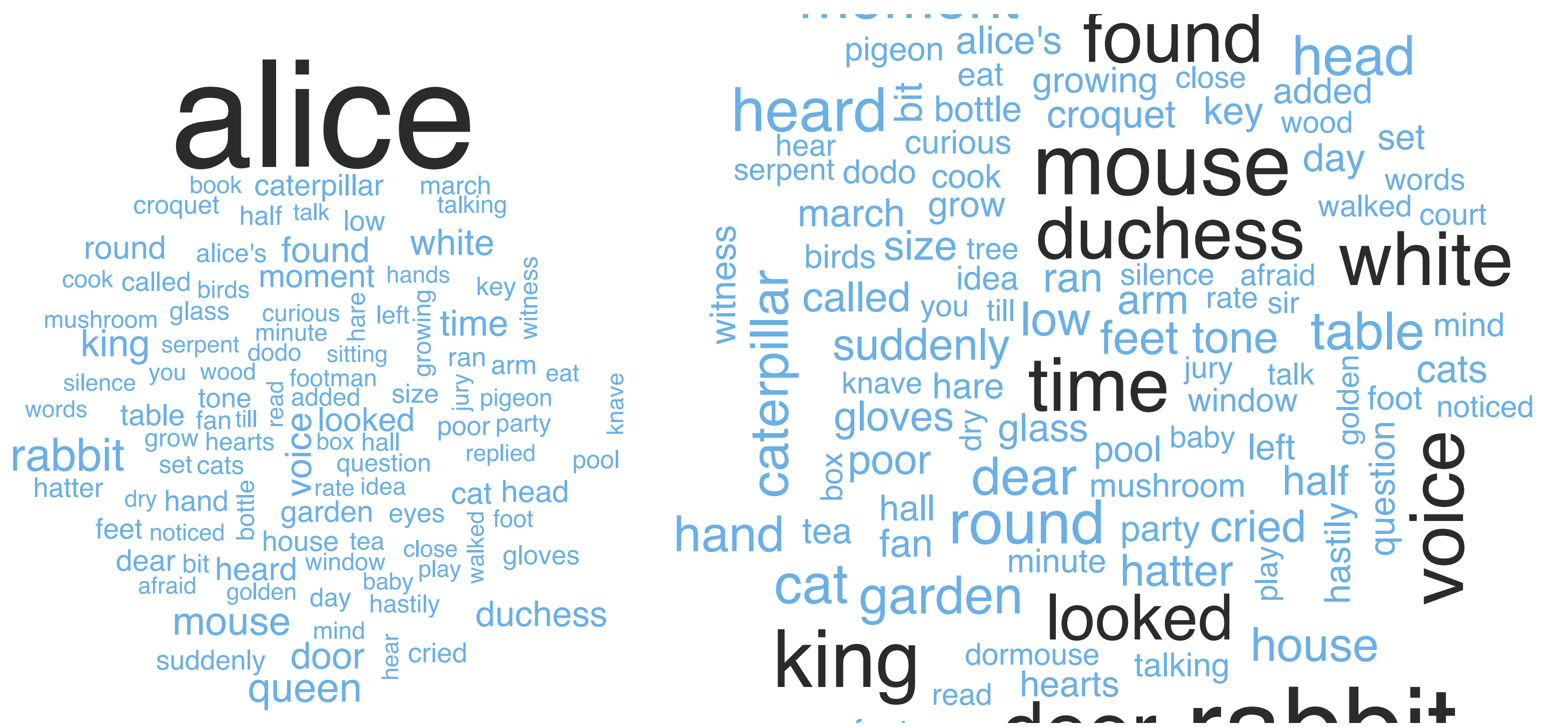

Figure 3. Preliminary Word Clouds of Most Common Terms

Top 50 Most Common Words in Lewis Carroll's *Alice in Wonderland* (with 10 or more word counts)

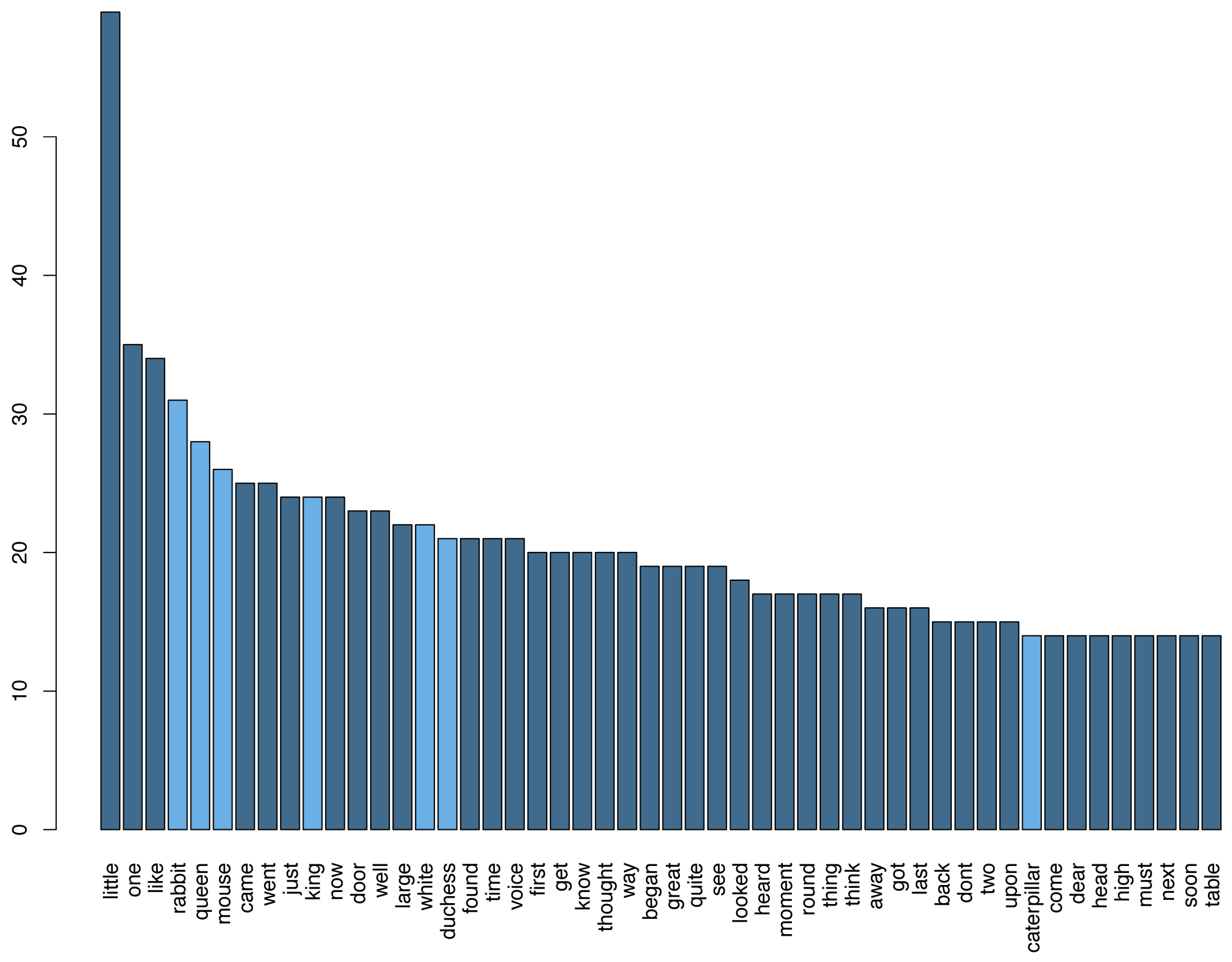

Figure 4. Top 50 Most Common Words in Lewis Carroll's Alice in Wonderland (with 10 or more word counts)

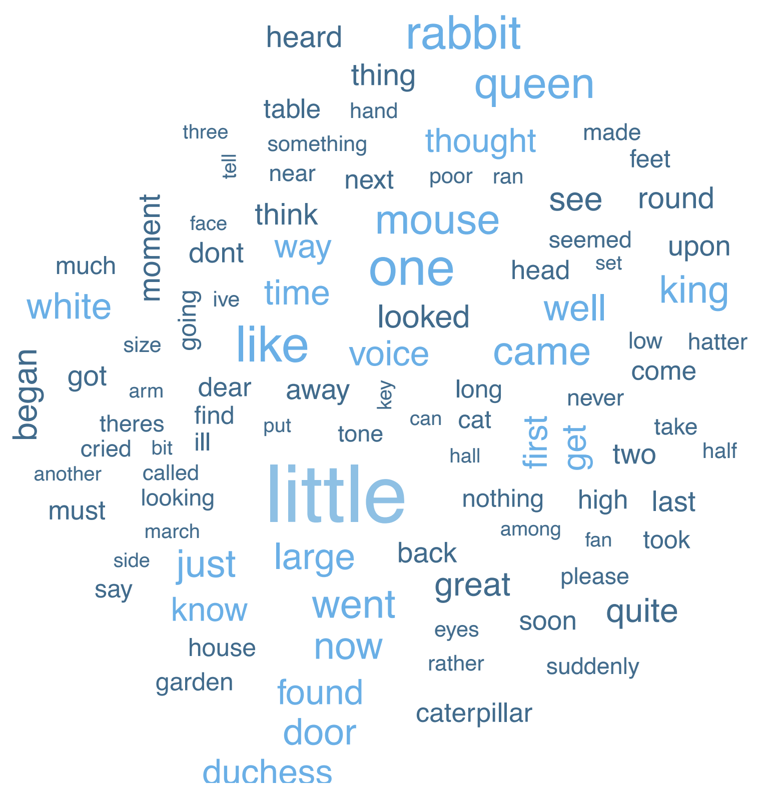

Figure 5. Word Cloud for Top 50 Terms

Tableau 1. Top Terms in Alice in Wonderland ('Alice' included)

Term	Count	tf	idf	tf_idf
Alice	163	0.039267646350277	0	0
rabbit	31	0.00746807998072754	0	0
queen	28	0.00674536256323777	0	0
mouse	26	0.00626355095157793	0	0
king	24	0.00578173933991809	0	0
door	23	0.00554083353408817	0	0
large	22	0.00529992772825825	0	0
white	22	0.00529992772825825	0	0
duchess	21	0.00505902192242833	0	0
found	21	0.00505902192242833	0	0
time	21	0.00505902192242833	0	0
voice	21	0.00505902192242833	0	0
first	20	0.00481811611659841	0	0
get	20	0.00481811611659841	0	0
know	20	0.00481811611659841	0	0
thought	20	0.00481811611659841	0	0
began	19	0.00457721031076849	0	0
great	19	0.00457721031076849	0	0
quite	19	0.00457721031076849	0	0
see	19	0.00457721031076849	0	0
looked	18	0.00433630450493857	0	0
heard	17	0.00409539869910865	0	0
moment	17	0.00409539869910865	0	0
round	17	0.00409539869910865	0	0
thing	17	0.00409539869910865	0	0
think	17	0.00409539869910865	0	0

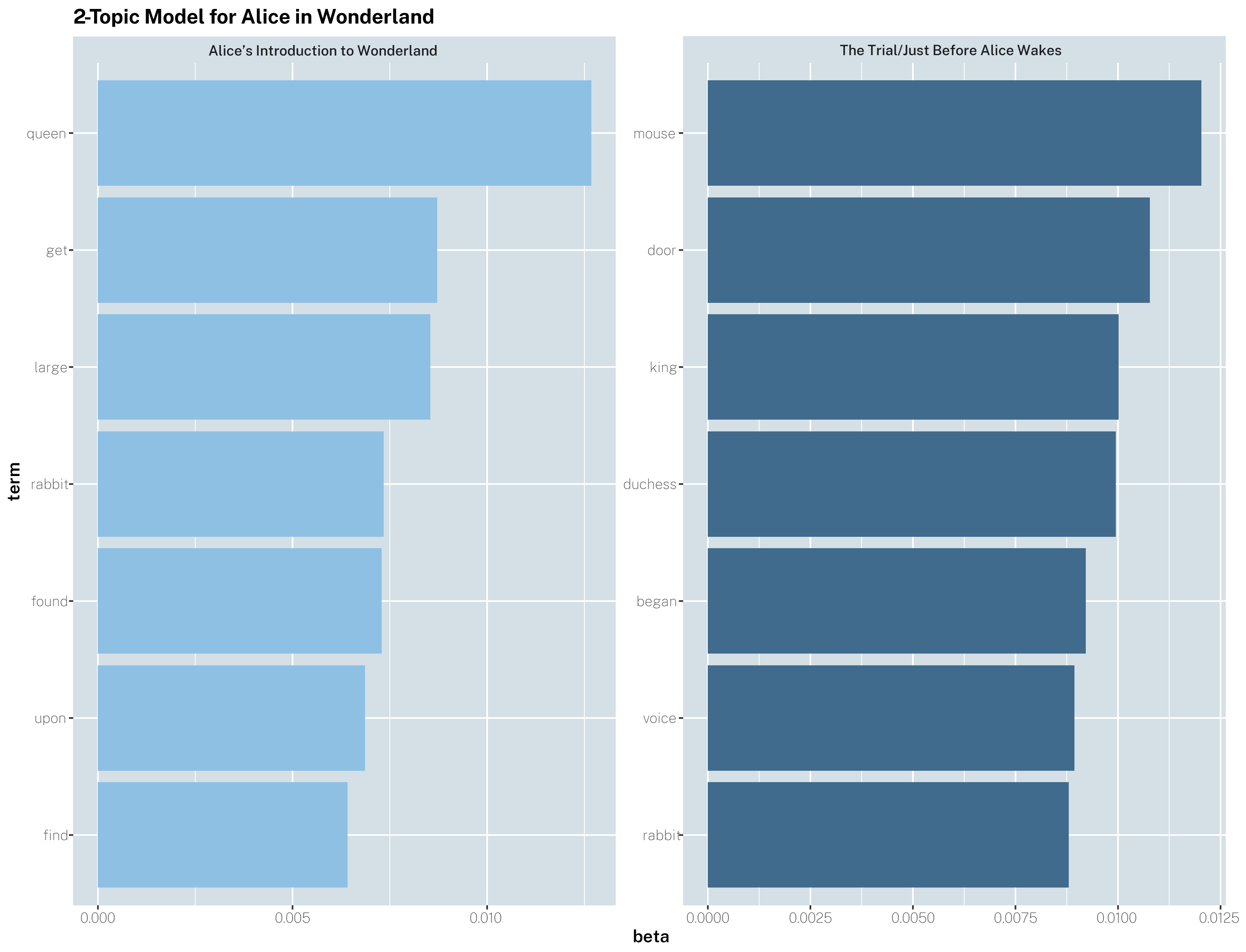

Figure 6. Two Topic Model of Alice in Wonderland

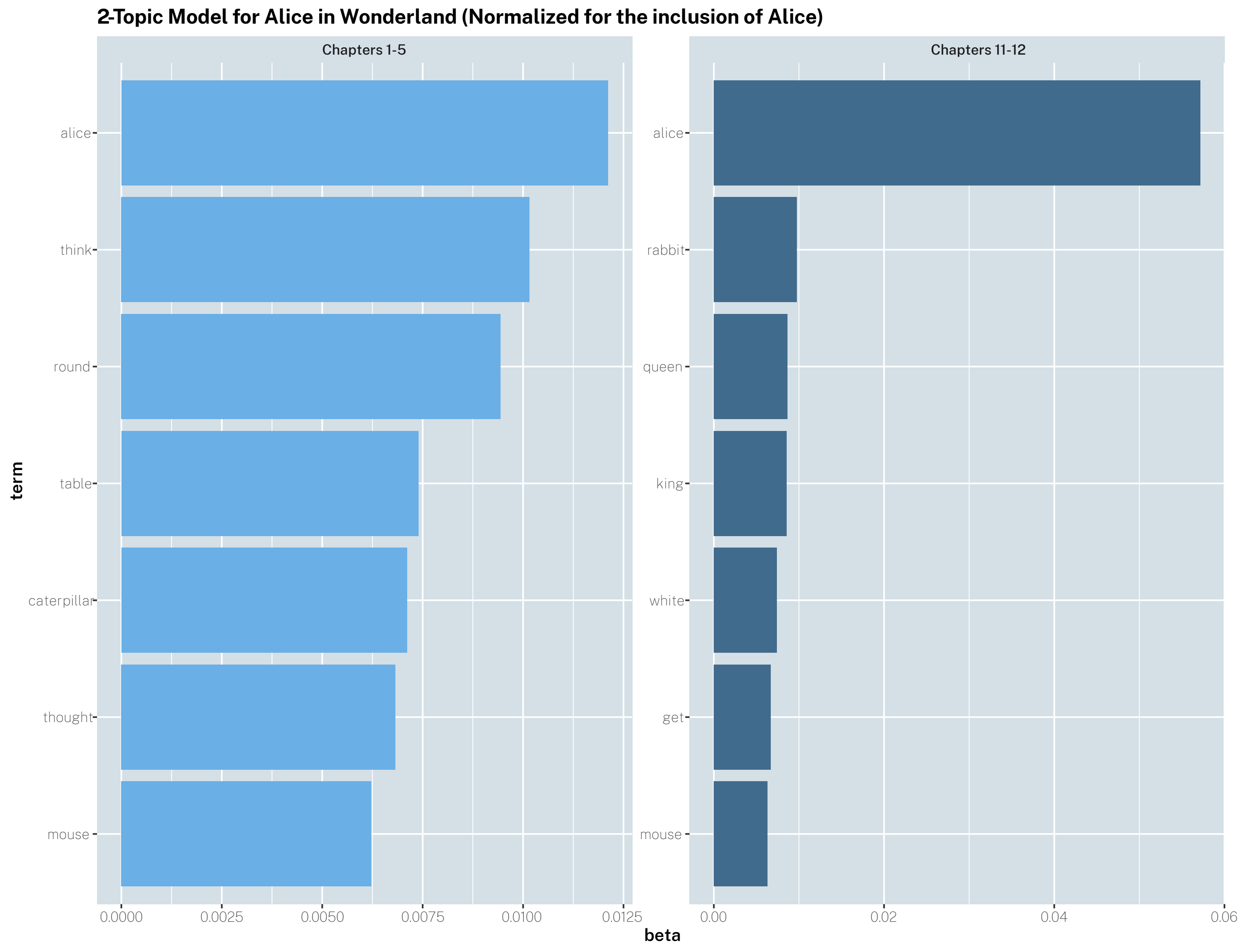

Figure 7. Two Topic Model of Alice in Wonderland (Normalized for inclusion of the term Alice)

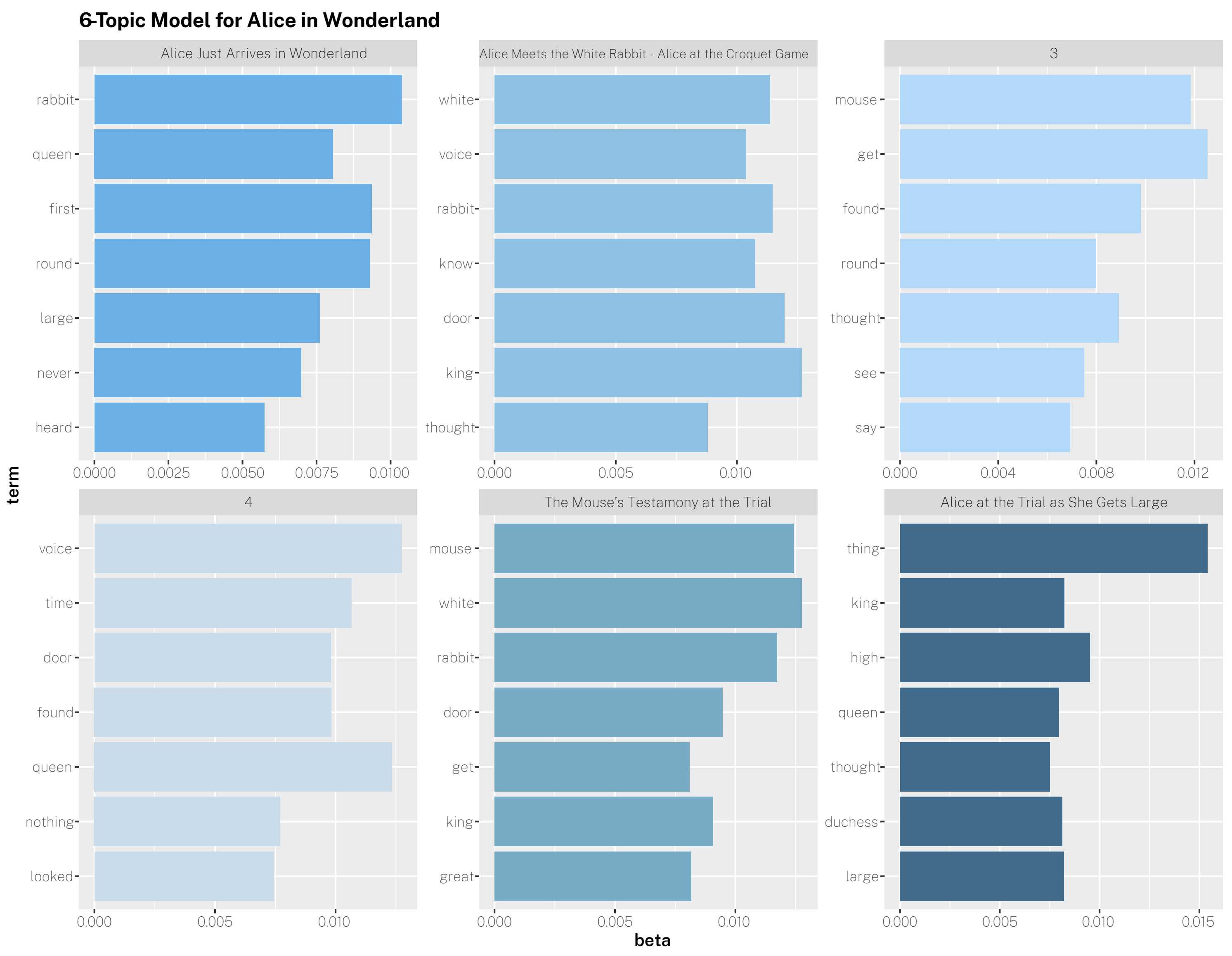

Figure 8. Six Topic Model of Alice in Wonderland

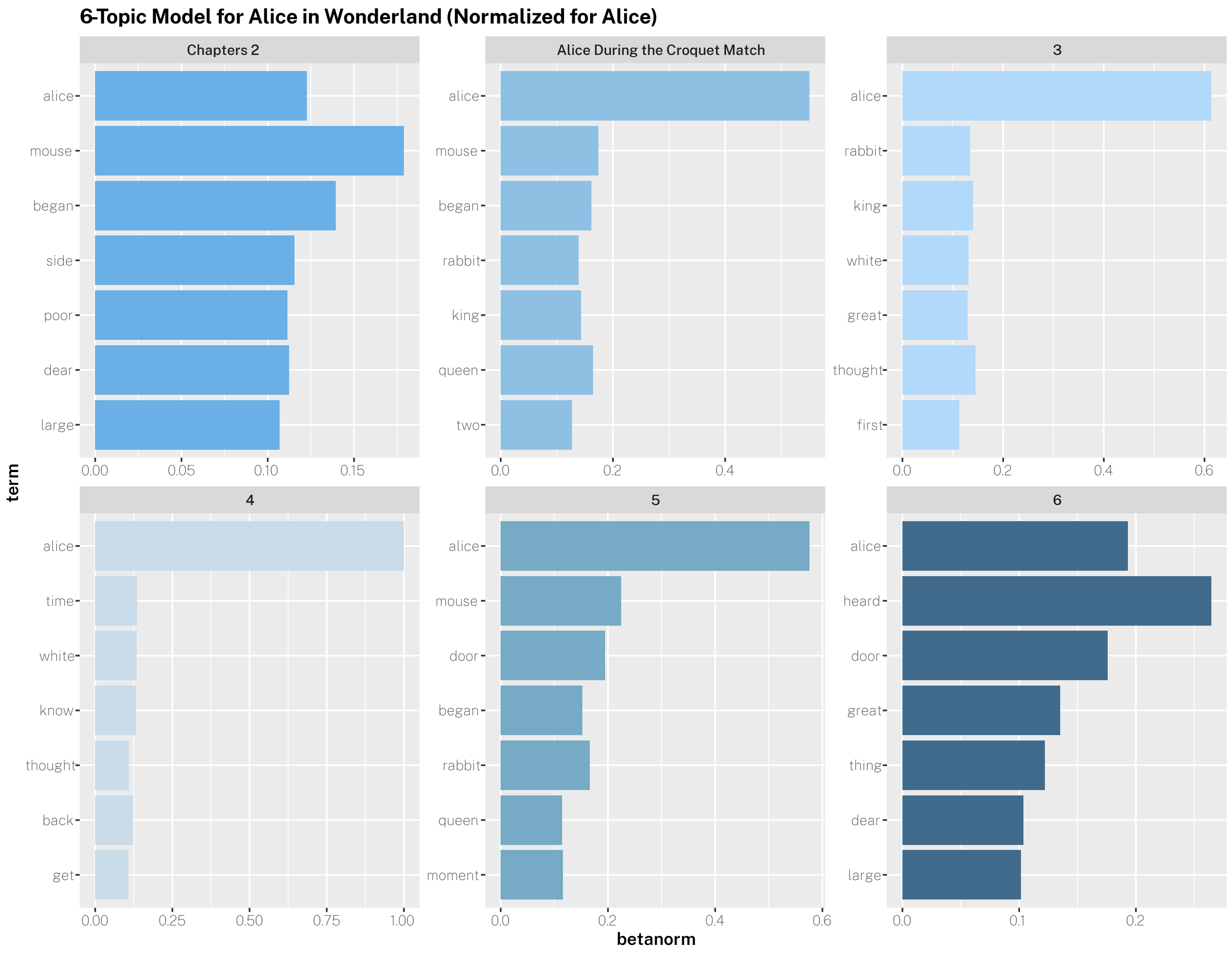

Figure 9. Six Topic Model for Alice in Wonderland (Normalized for Alice)

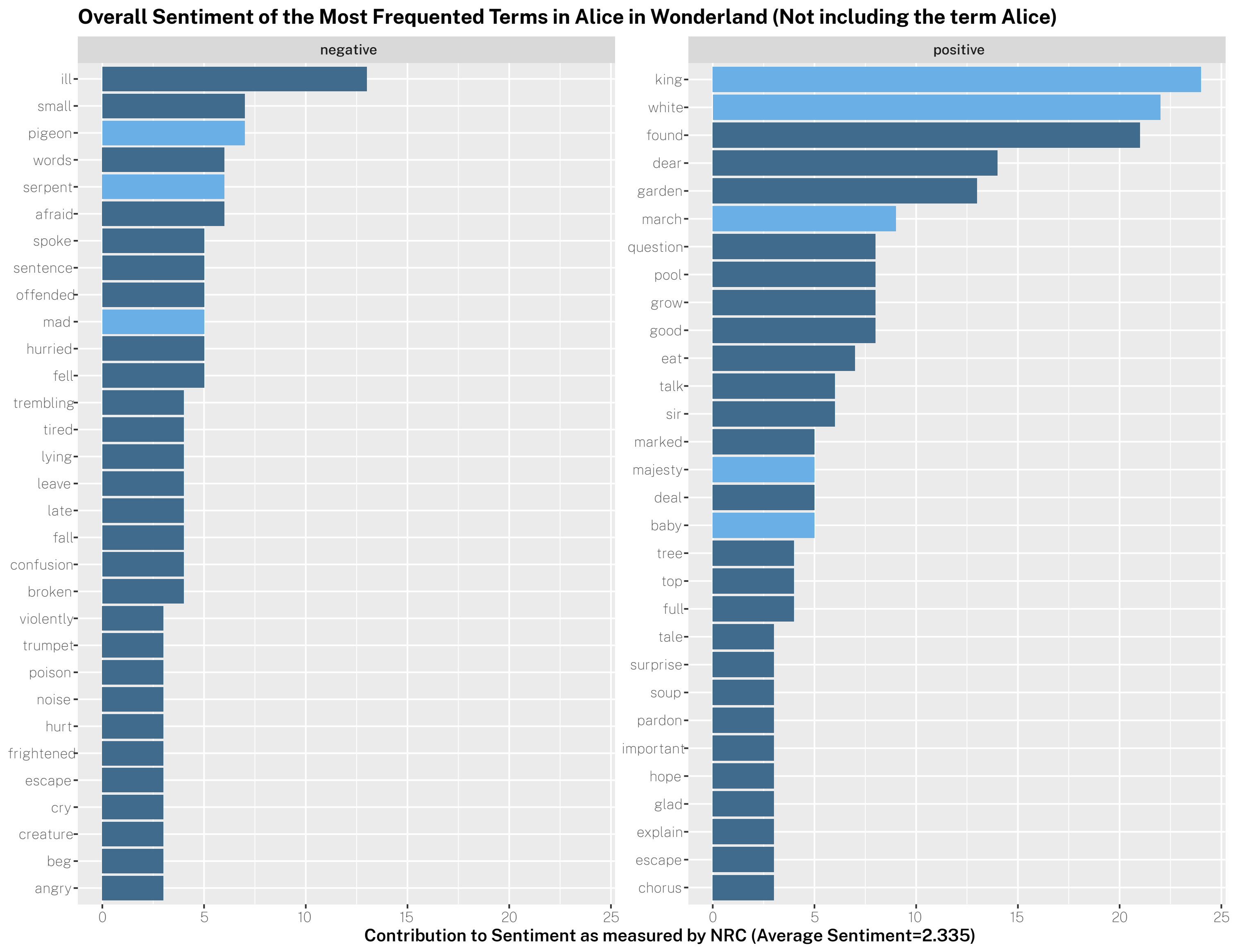

Figure 10. Overall Positive-Negative Sentiment for the Most Frequent Terms in Alice in Wonderland (Group A and Group B characters shown in light blue.)

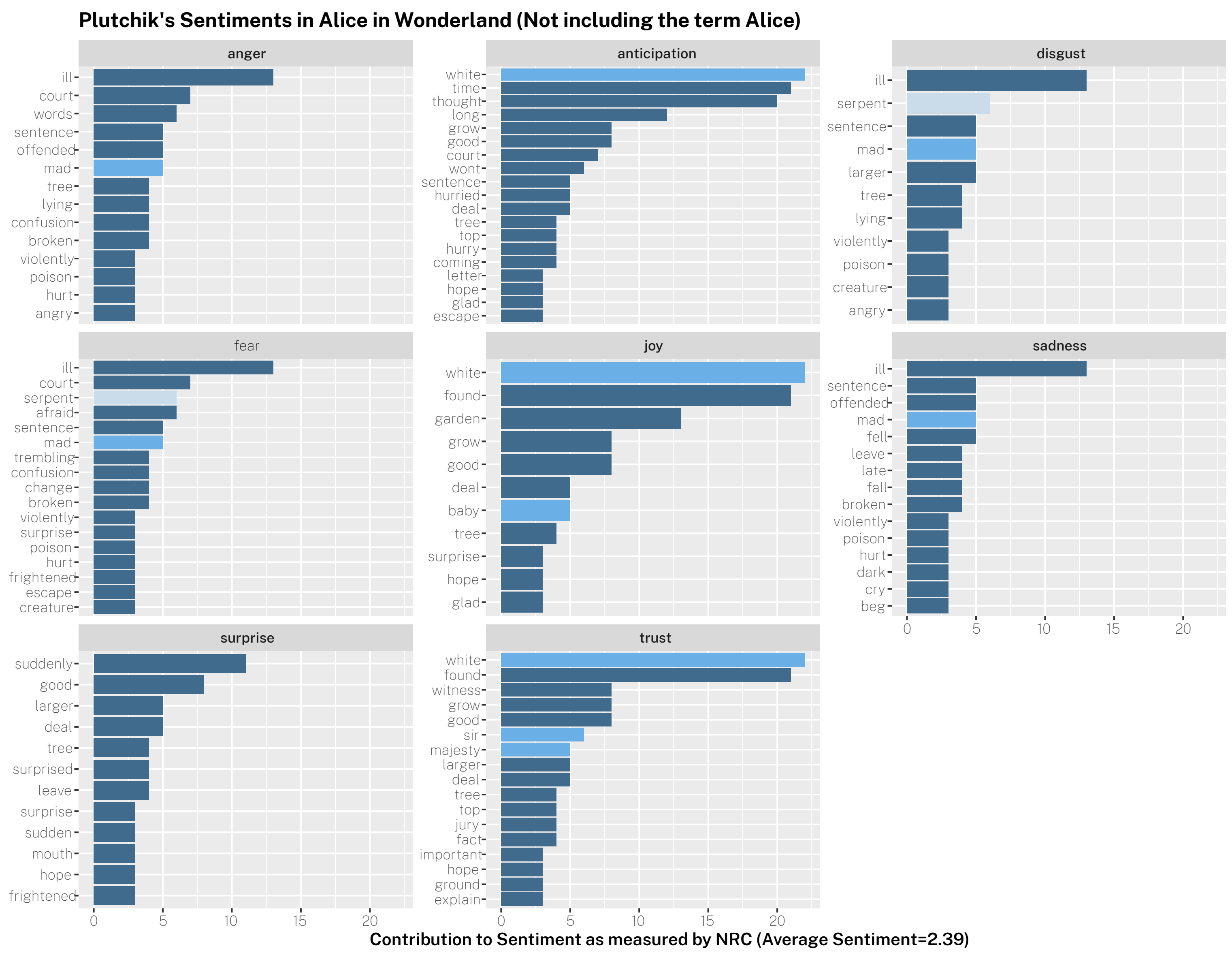

Figure 11. Plutchick's Sentiments of the Most Frequented Terms in Alice in Wonderland (Group A and Group B characters shown in light blue.)

Tableau 2. Bigrams in Alice in Wonderland (count>=10, mean count=2.795)

Count	tf	idf	tf_idf	Bigram
163	0.039267646350277	0	0	NA
31	0.00746807998072754	0	0	NA
28	0.00674536256323777	0	0	NA
26	0.00626355095157793	0	0	NA
24	0.00578173933991809	0	0	NA
23	0.00554083353408817	0	0	NA
22	0.00529992772825825	0	0	large white
21	0.00505902192242833	0	0	duchess found
21	0.00505902192242833	0	0	found time
21	0.00505902192242833	0	0	time voice
20	0.00481811611659841	0	0	first get
20	0.00481811611659841	0	0	get know
20	0.00481811611659841	0	0	know thought
19	0.00457721031076849	0	0	began great
19	0.00457721031076849	0	0	great quite
19	0.00457721031076849	0	0	quite see
18	0.00433630450493857	0	0	NA
17	0.00409539869910865	0	0	heard moment
17	0.00409539869910865	0	0	moment round
17	0.00409539869910865	0	0	round thing
17	0.00409539869910865	0	0	thing think
16	0.00385449289327873	0	0	away got
16	0.00385449289327873	0	0	got last
15	0.00361358708744881	0	0	back dont
15	0.00361358708744881	0	0	dont two
15	0.00361358708744881	0	0	two upon
14	0.00337268128161889	0	0	caterpillar come
14	0.00337268128161889	0	0	come dear
14	0.00337268128161889	0	0	dear head
14	0.00337268128161889	0	0	head high
14	0.00337268128161889	0	0	high must
14	0.00337268128161889	0	0	must next
14	0.00337268128161889	0	0	next soon
14	0.00337268128161889	0	0	soon table
13	0.00313177547578897	0	0	find garden
13	0.00313177547578897	0	0	garden ill

13	0.00313177547578897	0	0	ill much
13	0.00313177547578897	0	0	much nothing
13	0.00313177547578897	0	0	nothing took
12	0.00289086966995905	0	0	cat feet
12	0.00289086966995905	0	0	feet going
12	0.00289086966995905	0	0	going house
12	0.00289086966995905	0	0	house long
12	0.00289086966995905	0	0	long say
11	0.00264996386412913	0	0	cried hatter
11	0.00264996386412913	0	0	hatter looking
11	0.00264996386412913	0	0	looking low
11	0.00264996386412913	0	0	low made
11	0.00264996386412913	0	0	made near
11	0.00264996386412913	0	0	near never
11	0.00264996386412913	0	0	never please
11	0.00264996386412913	0	0	please seemed
11	0.00264996386412913	0	0	seemed suddenly
11	0.00264996386412913	0	0	suddenly take
11	0.00264996386412913	0	0	take theres
11	0.00264996386412913	0	0	theres tone
10	0.0024090580582992	0	0	called eyes
10	0.0024090580582992	0	0	eyes half
10	0.0024090580582992	0	0	half hand
10	0.0024090580582992	0	0	hand ive
10	0.0024090580582992	0	0	ive poor
10	0.0024090580582992	0	0	poor ran
10	0.0024090580582992	0	0	ran rather
10	0.0024090580582992	0	0	rather size
10	0.0024090580582992	0	0	size something

Tableau 3. Most Common Bigrams (without TF-IDF) in Alice in Wonderland (mean n=1.163)

Word1	Word2	n
white	rabbit	15
thought	Alice	11
march	hare	8
golden	key	6
cried	Alice	5
kid	gloves	5
play	croquet	5
white	kid	5
Alice	hastily	4
Alice	heard	4
Alice	looked	4
feet	high	4
inches	high	4
low	voice	4
mary	ann	4
Alice	replied	3
Alice	thought	3
another	moment	3
caucus	race	3
cheshire	cat	3
come	back	3
croquet	ground	3
first	witness	3
glass	table	3
good	deal	3
great	hurry	3
hand	bit	3
next	witness	3
oh	dear	3
poor	Alice	3
rabbit	hole	3
right	size	3
tea	party	3
trembling	voice	3

Tableau 4. Trigrams in Alice in Wonderland (count>=10, mean count=2.681)

Count	tf	idf	tf_idf	Trigram
163	0.039267646350277	0	0	NA
31	0.00746807998072754	0	0	NA
28	0.00674536256323777	0	0	NA
26	0.00626355095157793	0	0	NA
24	0.00578173933991809	0	0	NA
23	0.00554083353408817	0	0	NA
22	0.00529992772825825	0	0	NA
21	0.00505902192242833	0	0	duchess found time
21	0.00505902192242833	0	0	found time voice
20	0.00481811611659841	0	0	first get know
20	0.00481811611659841	0	0	get know thought
19	0.00457721031076849	0	0	began great quite
19	0.00457721031076849	0	0	great quite see
18	0.00433630450493857	0	0	NA
17	0.00409539869910865	0	0	heard moment round
17	0.00409539869910865	0	0	moment round thing
17	0.00409539869910865	0	0	round thing think
16	0.00385449289327873	0	0	away got last
15	0.00361358708744881	0	0	back dont two
15	0.00361358708744881	0	0	dont two upon
14	0.00337268128161889	0	0	caterpillar come dear
14	0.00337268128161889	0	0	come dear head
14	0.00337268128161889	0	0	dear head high
14	0.00337268128161889	0	0	head high must
14	0.00337268128161889	0	0	high must next
14	0.00337268128161889	0	0	must next soon
14	0.00337268128161889	0	0	next soon table
13	0.00313177547578897	0	0	find garden ill
13	0.00313177547578897	0	0	garden ill much
13	0.00313177547578897	0	0	ill much nothing
13	0.00313177547578897	0	0	much nothing took
12	0.00289086966995905	0	0	cat feet going
12	0.00289086966995905	0	0	feet going house
12	0.00289086966995905	0	0	going house long
12	0.00289086966995905	0	0	house long say
11	0.00264996386412913	0	0	cried hatter looking

11	0.00264996386412913	0	0	hatter looking low
11	0.00264996386412913	0	0	looking low made
11	0.00264996386412913	0	0	low made near
11	0.00264996386412913	0	0	made near never
11	0.00264996386412913	0	0	near never please
11	0.00264996386412913	0	0	never please seemed
11	0.00264996386412913	0	0	please seemed suddenly
11	0.00264996386412913	0	0	seemed suddenly take
11	0.00264996386412913	0	0	suddenly take theres
11	0.00264996386412913	0	0	take theres tone
10	0.0024090580582992	0	0	called eyes half
10	0.0024090580582992	0	0	eyes half hand
10	0.0024090580582992	0	0	half hand ive
10	0.0024090580582992	0	0	hand ive poor
10	0.0024090580582992	0	0	ive poor ran
10	0.0024090580582992	0	0	poor ran rather
10	0.0024090580582992	0	0	ran rather size
10	0.0024090580582992	0	0	rather size something

Tableau 5. Most Common Trigrams (without TF-IDF) in Alice in Wonderland (mean n=1.037)

word1	word2	word3	n
white	kid	gloves	5
what	thought	Alice	2
Alice	hastily	replied	2
blew	three	blasts	2
good	deal	frightened	2
left	hand	bit	2
low	trembling	voice	2
mad	tea	party	2
nine	feet	high	2
rabbit	blew	three	2
white	rabbit	blew	2

Bibliography

American Scientist—Plutchik. (2001, July 16). Retrieved December 11, 2019, from https://web.archive.org/web/20010716082847/http://americanscientist.org/articles/01articles/Plutchik.html

Bulkeley, K. (2019). The subversive dreams of Alice in Wonderland. *International Journal of Dream Research*, 49–59. https://doi.org/10.11588/ijodr.2019.2.62445

Carroll, L. (1963). *Alice's Adventures in Wonderland & Through the Looking-Glass: with the illus. of John Tenniel*. New York, NY: Macmillian.

Clark, A. (n.d.). *A philosopher's view of robots*. 2.

Dehaene, S., Lau, H., & Kouider, S. (2017). What is consciousness, and could machines have it? *Science*, *358*(6362), 486–492. https://doi.org/10.1126/science.aan8871

Deng, Y., & Jiang, H. (2018). The Development Overview of Artificial Mind. *Proceedings of the 2nd International Conference on E-Education, E-Business and E-Technology - ICEBT 2018*, 111–116. https://doi.org/10.1145/3241748.3241755

Episodic Memory—An overview (pdf) | ScienceDirect Topics. (n.d.). Retrieved November 19, 2019, from https://www.sciencedirect.com/topics/neuroscience/episodic-memory/pdf

Hassabis, D., Kumaran, D., Summerfield, C., & Botvinick, M. (2017). Neuroscience-Inspired Artificial Intelligence. *Neuron*, *95*(2), 245–258. https://doi.org/10.1016/j.neuron.2017.06.011

Intelligence, n. (n.d.). In *OED Online*. Retrieved from http://www.oed.com/view/Entry/97396

Pathman, T., Coughlin, C., & Ghetti, S. (2018). Space and time in episodic memory: Effects of linearity and directionality on memory for spatial location and temporal order in children and adults. *PLOS ONE*, *13*(11), e0206999. https://doi.org/10.1371/journal.pone.0206999

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind, New Series*, *59*(236), 433–460.

```r
####PROJECT TITLE AND CONTACT####
#Diedre Brown | dbrow207@pratt.edu
#INFO 640 Data Analysis | Pratt Institute
#Final Project
#Text Analysis of Lewis Carroll's Alice in Wonderland
#15 December 2019

####LOAD PACKAGES AND LIBRARIES####
#install.packages("tidyverse")
#install.packages("ggplot2")
#install.packages("ggthemes")
#install.packages("gmodels")
#install.packages("broom")
#install.packages("GGally")

#install.packages("tidytext")
#install.packages("gutenbergr")#project gutenberg
#install.packages("tm")#to create and work with corpora
#install.packages("topicmodels")#for LDA topic models
#install.packages("Rpoppler")
#install.packages("data.table")
#install.packages("stringr")

#install.packages("qdap")
#install.packages("RSQLite") #SQLite Interface
#install.packages("SnowballC") #text stemming library
#install.packages("wordcloud") #for wordcloud visualizations
#install.packages("syuzhet") #for text sentiment analysis
#install.packages("quanteda") #for N-grams
#install.packages("textdata") #required for sentiment dictionaries

library(ggthemes)
library(ggplot2)
library(tidyverse)
library(data.table)
library(dplyr)
library(broom)
library(GGally)

library(tidytext)
library(tm)
library(stringr)
library(topicmodels)
library(gutenbergr)
library(qdap)

library(RSQLite)
library(SnowballC)
library(wordcloud)
library(syuzhet)
library(quanteda)
library(textdata)
```

```
####DOWNLOAD LEWIS CARROLL'S ALICE IN WONDERLAND (AIW) FROM PROJECT
GUTENBERG####
aiw_book <- gutenberg_download(gutenberg_id = 19033)
aiw_book #a tibble of 1299 x2 (this includes the pjt. gutenberg id's) which
can be turned into a tidy text dataset
#remove gutenberg_id variable
aiw_book_text <- aiw_book %>%
  select(-gutenberg_id)
aiw_book_text
summary(aiw_book_text)
    #text
    #Length:1299 rows
    #Class :character
    #Mode  :character
dim(aiw_book_text)

####CLEAN DOCUMENT AND CREATE CORPUS, DTM/TDM####
#create aiw corpus from book.
aiw_source <- VectorSource(aiw_book_text)
aiw_corpus <- VCorpus(aiw_source)#aiw_corpus
aiw_corpus
    #Metadata:  corpus specific: 0, document level (indexed): 0
    #Content:  documents: 1

#specify stopwords
#in addition to stopwords("en"), add illustration, york, sons, company,
1916, gabirel, sam'l, v, vi, vii, viii, alice, dinah, sister, storyland,
series, copyright, saml, alice's, alices
new_stops<-c("series_","_the","well", "way","now","illustration", "york",
"sons", "company", "1916", "gabriel", "sam'l", "v", "vi", "vii", "viii",
"alice", "dinah", "sister","storyland", "series", "copyright", "saml",
"alice's", "alices", "said","like", "little", "went", "came", "one","just",
stopwords("en"))
#also need a stopword list that doesn't include alice
wastops<-c("series_","_the","well", "way","now","illustration", "york",
"sons", "company", "1916", "gabriel", "sam'l", "v", "vi", "vii",
"viii","dinah", "sister","storyland", "series", "copyright",
"saml","said","like", "little", "went", "came", "one","just", "alices",
stopwords("en"))


#clean corpus
#create a function to clean the corpus
clean_corp <- function(corp){
  #lowercase {base r}
  corp<-tm_map(corp, content_transformer(tolower))
  #remove punctuation {tm}
  corp<-tm_map(corp, removePunctuation)
  #remove stopwords
  corp<-tm_map(corp, removeWords, words=new_stops)
  #strip whitespace {tm}
  corp<-tm_map(corp,stripWhitespace)
  return(corp)
}
```

```
#clean_corp function for the stopwords that do not include 'alice'
clean_wacorp <-function(corp){
  #lowercase {base r}
  corp<-tm_map(corp, content_transformer(tolower))
  #remove punctuation {tm}
  corp<-tm_map(corp, removePunctuation)
  #remove stopwords
  corp<-tm_map(corp, removeWords, words=wastops)
  #strip whitespace {tm}
  corp<-tm_map(corp,stripWhitespace)
  return(corp)
}

#clean the aiw_corpus with the clean_corp function and place in
aiw_cleancorpus
aiw_cleancorpus <- clean_corp(aiw_corpus)
summary(aiw_cleancorpus)
str(aiw_cleancorpus)

#clean the aiw_corpus with the clean_wacorp function and place in
aiw_cleanwacorpus
aiw_cleanwacorpus <- clean_wacorp(aiw_corpus)
summary(aiw_cleanwacorpus)
str(aiw_cleanwacorpus)

#create a term document matrix from aiw_cleancorpus
aiw_tdm <- TermDocumentMatrix(aiw_cleancorpus)
aiw_dtm <- DocumentTermMatrix(aiw_cleancorpus)
#term document matrix from aiw_cleanwacorpus
wa_aiw_tdm <- TermDocumentMatrix(aiw_cleanwacorpus)
wa_aiw_dtm <- DocumentTermMatrix(aiw_cleanwacorpus)

####TERM FREQUENCY VISUALIZATIONS####
#Previous review of the corpus showed that "alice" had the highest term
frequency (163 counts). With "alice" removed, let's look at term frequency.
#convert tdm to matrix
aiw_mat <- as.matrix(aiw_tdm)

#sum rows and sort by frequency
aiw_termfreq <- rowSums(aiw_mat)
aiw_termfreq <- sort(aiw_termfreq, decreasing = TRUE)
summary(aiw_termfreq)
    #Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    #1.000   1.000   1.000   3.075   3.000 144.000
glimpse(aiw_termfreq)

#plot frequency
#since the mean was 3.075, i want to see the top 50 terms that were used 3
or more times
barplot(aiw_termfreq[1:50],
        col = "dodgerblue",
        las = 3)
```

```r
#let's see the next 50 for comparison and what characters are included
barplot(aiw_termfreq[51:100],
        col = "dodgerblue",
        las = 3)

#view top 100 words as a wordcloud and wordnetwork
#sum rows and sort by frequency to create aiw data frame from aiw_termfreq
aiw_freqsum<-rowSums(aiw_mat)
aiw_wordFreq <- data.frame(term=names(aiw_freqsum), num=aiw_freqsum)
#make a word cloud of top 100
wordcloud(aiw_wordFreq$term,
          aiw_wordFreq$num,
          max.words = 100,
          colors = c("#416B8C","#6AAFE6","#8EC0E4"))
#wordcloud with "said" removed
glimpse(aiw_wordFreq)
aiw_wordFreq_sanssaid <- aiw_wordFreq %>%
  filter(term != "said")
wordcloud(aiw_wordFreq_sanssaid$term,
           aiw_wordFreq_sanssaid$num,
           max.words = 100,
           colors = c("#416B8C","#6AAFE6","#8EC0E4"))
#add said to stopword list? yes.

####TOPIC MODELING####
###Topics###
##non-alice lda function
wonderland_tm_terms_by_topic <-function(input_corpus, plot=TRUE,
number_of_topics=6, number_of_words=7,
                                   path="lda-121519/aiw_lda_norm_topics") {
  aiw_dtm <- DocumentTermMatrix(input_corpus)
  #unique indexes
  unique_indexes<-unique(aiw_dtm$i)
  aiw_dtm <-aiw_dtm[unique_indexes,]
  #lda
  aiw_lda <-LDA(aiw_dtm, k=number_of_topics, control = list(seed=1234))
  aiw_topics<-tidy(aiw_lda, matrix="beta")
  aiw_lda_words <-terms(aiw_lda, number_of_words)
  aiw_lda_topics <-as.matrix(aiw_lda_words)
  write.csv(aiw_lda_topics, file = paste(path, number_of_topics,".csv"))
  aiw_top_terms_2<-aiw_topics%>%
    group_by(topic)%>%
    top_n(number_of_words, beta)%>%
    ungroup()%>%
    arrange(topic, -beta)

  if(plot==TRUE){
    aiw_top_terms_2%>%
      mutate(term=reorder(term,beta))%>%
      ggplot(aes(term, beta, fill=factor(topic)))+
      geom_col(show.legend = FALSE)+
      facet_wrap(~topic, scales="free")+
      coord_flip()+
      labs(title = "Topic Model for Alice in Wonderland (Alice Omitted)")
  }
}

#alice_tm_function
alice_tm_terms_by_topic <-function(input_corpus, plot=TRUE,
number_of_topics=6, number_of_words=7,
                               path="lda-121519/with-alice/
alice_lda_norm_topics") {
  wa_aiw_dtm <- DocumentTermMatrix(input_corpus)
  #unique indexes
  unique_indexes<-unique(wa_aiw_dtm$i)
  wa_aiw_dtm <-wa_aiw_dtm[unique_indexes,]
  #lda
  wa_aiw_lda <-LDA(wa_aiw_dtm, k=number_of_topics, control =
list(seed=1234))
  wa_topics<-tidy(wa_aiw_lda, matrix="beta")
  wa_aiw_lda_words <-terms(wa_aiw_lda, number_of_words)
  wa_aiw_lda_topics <-as.matrix(wa_aiw_lda_words)
  write.csv(wa_aiw_lda_topics, file = paste(path, number_of_topics,".csv"))
  wa_aiw_top_terms_2<-wa_topics%>%
    group_by(topic)%>%
    top_n(number_of_words, beta)%>%
    ungroup()%>%
    arrange(topic, -beta)

  if(plot==TRUE){
    wa_aiw_top_terms_2%>%
      mutate(term=reorder(term,beta))%>%
      ggplot(aes(term, beta, fill=factor(topic)))+
        geom_col(show.legend = FALSE)+
        facet_wrap(~topic, scales="free")+
        coord_flip()+
        labs(title = "Topic Model for Alice in Wonderland")
  }
}

#functions run for 6 topics -- previous running showed 6 topics with 7
words to be a good balance of themes from the book both with and without
'alice'
alice_tm_terms_by_topic (aiw_cleanwacorpus, number_of_topics = 6,
number_of_words = 7)
wonderland_tm_terms_by_topic (aiw_cleancorpus, number_of_topics = 6,
number_of_words = 7)

#functions run for 2 topics--
alice_tm_terms_by_topic (aiw_cleanwacorpus, number_of_topics = 2,
number_of_words = 7)
wonderland_tm_terms_by_topic (aiw_cleancorpus, number_of_topics = 2,
number_of_words = 7)


#alice_tm but normalized for alice
#normalize for alice six topics
k<-6
```

```
wa_aiw_lda <-LDA(wa_aiw_dtm, k=k, control = list(seed=1234))
wa_aiw_lda
wa_aiw_lda_words <-terms(wa_aiw_lda, 7)
wa_aiw_lda_topics <-as.matrix(wa_aiw_lda_words)
head(wa_aiw_lda_topics)
write.csv(wa_aiw_lda_topics, file = paste("lda-121519/with-alice/
wa_lda_norm",k,".csv"))
#visualize
wa_aiw_lda_tidy<-tidy(wa_aiw_lda, matrix="beta")
wa_aiw_lda_tidy
wa_aiw_lda_tidy_norm <- wa_aiw_lda_tidy%>%
  mutate(betanorm=((beta - min(beta)) / (max(beta) - min(beta))))
wa_aiw_lda_tidy_norm
#order words from most prominent to least
wa_aiw_top_terms_norm<-wa_aiw_lda_tidy_norm%>%
  group_by(topic)%>%
  top_n(7,betanorm)%>%
  ungroup()%>%
  arrange(topic, -betanorm)
wa_aiw_top_terms_norm
#plot
wa_aiw_top_terms_norm%>%
  mutate(term=reorder(term,betanorm))%>%
  ggplot(aes(term, betanorm, fill=factor(topic)))+
  geom_col(show.legend = FALSE)+
  facet_wrap(~topic, scales="free")+
  coord_flip()+
  labs(title = "Topic Model for Alice in Wonderland (Normalized for
Alice)")

#normalize for alice 12 topics
k<-12
wa_aiw_lda12 <-LDA(wa_aiw_dtm, k=k, control = list(seed=1234))
wa_aiw_lda12
wa_aiw_lda12_words <-terms(wa_aiw_lda12, 7)
wa_aiw_lda12_topics <-as.matrix(wa_aiw_lda12_words)
head(wa_aiw_lda12_topics)
write.csv(wa_aiw_lda12_topics, file = paste("lda-121519/with-alice/
wa_lda_norm",k,".csv"))
#visualize
wa_aiw_lda12_tidy<-tidy(wa_aiw_lda12, matrix="beta")
wa_aiw_lda12_tidy
wa_aiw_lda12_tidy_norm <- wa_aiw_lda12_tidy%>%
  mutate(betanorm=((beta - min(beta)) / (max(beta) - min(beta))))
wa_aiw_lda12_tidy_norm
#order words from most prominent to least
wa_aiw_top_terms_norm12<-wa_aiw_lda12_tidy_norm%>%
  group_by(topic)%>%
  top_n(7,betanorm)%>%
  ungroup()%>%
  arrange(topic, -betanorm)
wa_aiw_top_terms_norm12
#plot
wa_aiw_top_terms_norm12%>%
  mutate(term=reorder(term,betanorm))%>%
  ggplot(aes(term, betanorm, fill=factor(topic)))+
  geom_col(show.legend = FALSE)+
    facet_wrap(~topic, scales="free")+
    coord_flip()+
    labs(title = "Twelve Topics for Alice in Wonderland (Normalized for 
Alice)")


####SENTIMENT ANALYSIS####
#After reviewing the four dictionaries (NRC, AFINN, Loughran, and Bing) 
available with R
#the most suited to the this material is NRC.
#NRC had a good mix of emotional terms that related to the story.

# Count the number of words associated with each sentiment in nrc
nrc<-get_sentiments("nrc")%>%
  count(sentiment)%>%
  arrange(desc(n))
#All terms of NRC are good descriptions of Alice's emotions throughout the 
story.

#APPENDING DICTIONARIES
#create new tidy dataframes for  texts (w/o alice term) from previous dfs
aiwsenttidy <-aiw_wordFreq_sanssaid%>%
  mutate(word=term, count=num)%>%
  select(-term,-num)%>%
  arrange(desc(count))
head(aiwsenttidy)
#append the nrc dictionary
aiwsenttidynrc<-aiwsenttidy%>%
  inner_join(get_sentiments("nrc"))

#--------------------------------------------------------------------#
  #CANNOT USE THIS AS CHARACTERS WERE REMOVED DURING THE JOIN TO THE 
SENTIMENT LEXICONS-----------
  #create new tidy dataframes for  texts (w/alice term) from previous dfs
  #convert wa_aiw_tdm to matrix
  #wa_aiw_mat <- as.matrix(wa_aiw_tdm)
  #sum rows and sort by frequency
  #wa_aiw_termfreq <- rowSums(wa_aiw_mat)
  #wa_aiw_termfreq <- sort(wa_aiw_termfreq, decreasing = TRUE)
  #wa_aiw_termfreq
  #sum rows and sort by frequency to create aiw data frame from 
aiw_termfreq
  #wa_aiw_freqsum<-rowSums(wa_aiw_mat)
  #wa_aiw_wordFreq <- data.frame(term=names(wa_aiw_freqsum), 
num=wa_aiw_freqsum)
  #wa_aiw_wordFreq
  #create new tidy dataframes for  texts (w/alice term) from previous dfs
  #wa_aiwsenttidy <-wa_aiw_wordFreq%>%
  #  mutate(word=term, count=num)%>%
  #  select(-term,-num)%>%
  #  arrange(desc(count))
  #head(wa_aiwsenttidy)

  #append the nrc dictionary
  #wa_aiwsenttidynrc<-wa_aiwsenttidy%>%
  #  inner_join(get_sentiments("nrc"))
  #many of the characters were removed during the join

  #wa_aiwsenttidyafinn<-wa_aiwsenttidy%>%
  #  inner_join(get_sentiments("afinn"))
  #many of the characters were removed during the join
  #----------------------------------------------------------------#
####SENTIMENT ANALYSIS CONTINUED####
#visualize positive and negative sentiment in the nrc
aiwnrp_n <- aiwsenttidynrc %>%
  filter(sentiment %in% c("positive", "negative"))%>%
  group_by(sentiment)
summary(aiwnrp_n) #mean count = 2.335

aiwnrp_n <- aiwsenttidynrc %>%
  filter(sentiment %in% c("positive", "negative"))%>%
  group_by(sentiment)%>%
  filter(count>=2.335)%>%
  ungroup()%>%
  mutate(word=reorder(word, count))
ggplot(aiwnrp_n, aes(word, count, fill=sentiment))+
    geom_col(show.legend = FALSE)+
    facet_wrap(~sentiment, scales = "free_y")+
    labs(y= "Contribution to Sentiment as measured by NRC (Average
Sentiment=2.335)", x=NULL, title = "Overall Sentiment of the Most
Frequented Terms in Alice in Wonderland (Not including the term Alice)" )+
    coord_flip()

#remove positive and negative sentiment and visualize each of the eight
emotions
#corresponding to plutchik's wheel of emotion
aiwemotion <- aiwsenttidynrc%>%
  filter(!grepl("positive|negative",sentiment))%>%
  group_by(sentiment)%>%
  ungroup()
summary(aiwemotion) #mean 2.39, length=467

aiwemoplot <- aiwemotion%>%
  group_by(sentiment)%>%
  filter(count>=2.39)%>%
  ungroup()%>%
  mutate(word=reorder(word, count))
ggplot(aiwemoplot, aes(word, count, fill=sentiment))+
  geom_col(show.legend = FALSE)+
  facet_wrap(~sentiment, scales = "free_y")+
  labs(y= "Contribution to Sentiment as measured by NRC (Average
Sentiment=2.39)", x=NULL, title = "Plutchik's Sentiments in Alice in
Wonderland (Not including the term Alice)" )+
  coord_flip()
```

```
####N-GRAMS####
#  wa_aiw_bigrams_sep <-wa_aiw_bigrams%>%
#    separate(bigram, c("word1", "word2"), sep = " ")
#  wa_aiw_bigrams_filt <- wa_aiw_bigrams_sep %>%#filter for stop words
#    filter(!word1 %in% wastops)%>%
#    filter(!word2 %in% wastops)
#  wa_aiw_bigrams_count<-wa_aiw_bigrams_filt%>%#count bigrams
#    count(word1, word2, sort = TRUE)
wa_aiw_dtm_tidy<-tidy(wa_aiw_dtm)
wa_aiw_tfidf<-bind_tf_idf(wa_aiw_dtm_tidy, term, document, count)%>%#find
words that are important but not too common
  select(-document) %>% #there's only 1 document so let's elim the document
column
  arrange(desc(tf)) #look at terms with high tf_idf
head(wa_aiw_tfidf, 20)
summary(wa_aiw_tfidf)#mean count 2.919
write.table(wa_aiw_tfidf,file = "INFO640-Brown-FinalProject-121519/alice-
tfterms.txt", sep = ",", quote = FALSE, row.names = F)
#bigrams part 1
wa_aiw_bigrams<- wa_aiw_tfidf%>%
  unnest_tokens(bigram, term, token = "ngrams", n=2)%>%
  arrange(desc(tf))
wa_aiw_bigrams #all characters turned into NA values
write.table(wa_aiw_bigrams, file = "INFO640-Brown-FinalProject-121519/
alice-bigrams.txt", sep = ",", quote = FALSE, row.names = F)
summary(wa_aiw_bigrams) #mean count = 2.795
wa_aiw_bigrams
#trigrams part 1
wa_aiw_trigrams<-wa_aiw_tfidf%>%
  unnest_tokens(trigram, term, token = "ngrams", n=3)%>%
  arrange(desc(tf))
wa_aiw_trigrams #all characters turned into NA values
write.table(wa_aiw_trigrams, file = "INFO640-Brown-FinalProject-121519/
alice-trigrams.txt", sep = ",", quote = FALSE, row.names = F)
summary(wa_aiw_trigrams) #mean count = 2.795

#bigrams part 2 (keep characters?)
wa_aiw_bigrams2<-aiw_book_text %>%
  unnest_tokens(bigram, text, token = "ngrams", n=2)
wa_aiw_bigrams2
wa_aiw_bigrams2_sep <-wa_aiw_bigrams2%>%
  separate(bigram, c("word1", "word2"), sep = " ")
wa_aiw_bigrams2_filt <- wa_aiw_bigrams2_sep %>%#filter for stop words
  filter(!word1 %in% wastops)%>%
  filter(!word2 %in% wastops)
wa_aiw_bigram2_count <-wa_aiw_bigrams2_filt%>%#count bigrams
  count(word1, word2, sort = TRUE)
wa_aiw_bigram2_count
write.table(wa_aiw_bigram2_count, file = "INFO640-Brown-
FinalProject-121519/alice-bigrams2.txt", sep = ",", quote = FALSE,
row.names = F)
summary(wa_aiw_bigram2_count) #mean count = 1.163
#trigrams part 2
```

```
wa_aiw_trigrams2<-aiw_book_text %>%
  unnest_tokens(trigram, text, token = "ngrams", n=3)
  wa_aiw_trigrams2
wa_aiw_trigrams2_sep <-wa_aiw_trigrams2%>%
  separate(trigram, c("word1", "word2", "word3"), sep = " ")
wa_aiw_trigrams2_filt <- wa_aiw_trigrams2_sep %>%#filter for stop words
  filter(!word1 %in% wastops)%>%
  filter(!word2 %in% wastops)%>%
  filter(!word3 %in% wastops)
wa_aiw_trigram2_count <-wa_aiw_trigrams2_filt%>%#count bigrams
  count(word1, word2, word3, sort = TRUE)
wa_aiw_trigram2_count
write.table(wa_aiw_trigram2_count, file = "INFO640-Brown-
FinalProject-121519/alice-trigrams2.txt", sep = ",", quote = FALSE,
row.names = F)
summary(wa_aiw_trigram2_count) #mean count = 1.037
```