Hemingway and Carroll Sentiment Analysis: Exploratory Analysis

1. **Description of the cleaning and analysis processes**

a. Purpose:

In Our Time by Ernest Hemingway and Alice's Adventures in the Wonderland by Lewis Carroll, the novels from homework one are further evaluated in homework 2. The purpose of homework 2 is to conduct exploratory analysis by extracting sentences that contain adjectives or adverb phrases to gain insights into how to approach the sentiment analysis of these novels. Both novels are labeled as Hemingway and Carroll for ease of discussion in this assignment.

b. Cleaning process:

i. Packages:

In Our Time and Alice's Adventures in the Wonderland were transformed into text files in homework 1. Natural Language Toolkit (NLTK) and Regular Expression (re) were imported for processing the text files in Jupyter. From NLTK imported sent_tokenize, nltk.RegexpParser, and FreqDist for the cleaning and analysis processes.

ii. Sentence Tokenization:

Two-step tokenization was used on each text. The first step is to tokenize the sentences using `nltk.sent_tokenize` based on punctuation like ".". The second step is using the nltk.word_tokenize the text within the tokenized sentence, `[nltk.word_tokenize(sent) for sent in textsplit]` . Then, Stanford Part-of-Speech (POS) tagger POS was applied to assign part of speech to the tokens of each sentence, `[nltk.pos_tag(tokens) for tokens in tokentext]`. The POS tagging was necessary to retrieve the adjective (JJ) and adverb (RB) tags for the next step. Both Hemingway and Carroll were labeled as "taggedtext" for the analysis processing.

iii. Chunking

The chunking technique is applied to parse the tokenized sentence to extract the adjective and adverb phrases. This technique uses regular expression (re) to segments and labels multi-token sequences to detect adjective and adverb phrases. The regular expressions define how adjective phrases (ADJPH), and adverb phrases (ADVPH) are identified in the chunk as the following:

```
grammar_adjph = "ADJPH: {<RB.?>+<JJ.?>}"
```

The ADJPH expression, `"ADJPH: {<RB.?>+<JJ.?>}"` reads to find a phrase of (<>) adverb (RB.?) and adjective (JJ.?) with "." as a wildcard for RBR, RBS, JJR, or JJS. Additionally, the "?" can be considered RB or JJ alone for both adverb and adjective.

```
grammar_advph = "ADVPH: {<RB>+<RB>}"
```

The ADVPH expression `"ADVPH: {<RB>+<RB>}"` reads to find a phrase of two consecutive adverbs (RB).

iv. NLTK Parser

Then, nltk.RegexParser is imported to process each sentence for adjective or adverb phrase. ADJPH or ADVPH expression is inputted for chunk_parser_adj or chuck_parser_adv, respectively. Then, python code processed the taggedtext through the regex parser tree to extracted for adjph_tags (Figure 1) or advph_tags (Figure 2) for the analysis process.

```
chunk_parser_adj = nltk.RegexpParser(grammar_adjph)

adjph_tags = []
for sent in taggedtext:
    if len(sent) > 0:
        tree = chunk_parser_adj.parse(sent)
        for subtree in tree.subtrees():
            if subtree.label() == 'ADJPH':
                adjph_tags.append(subtree)
```

Figure 1: Adjective Phrase Parser

```
chunk_parser_adv = nltk.RegexpParser(grammar_advph)

advph_tags = []
for sent in taggedtext:
    if len(sent) > 0:
        tree = chunk_parser_adv.parse(sent)
        for subtree in tree.subtrees():
            if subtree.label() == 'ADVPH':
                advph_tags.append(subtree)
```

Figure 2: Adverb Phrase Parser

c. Analysis Process

i. Adjective and Adverb Phrases

The adjph_tags and advph_tags were input and coded for extracting the lists of adjective_phrases (figure 3) and adverb_phrases (figure 4). Then, statistical analysis can be performed to determine the length (len(advph_tag) or len(advph_tag)), and the frequency of the top 50 adjectives (figure 5 & 7) and adverb (figure 6 & 8) phrases of Carroll and Hemingway.

```
adjective_phrases = []
for sent in adjph_tags:
    temp = ''
    for w, t in sent:
        temp += w+ ' '
    adjective_phrases.append(temp)
```

Figure 3: Adjective Phrase

```
adverb_phrases = []
for sent in advph_tags:
    temp = ''
    for w, t in sent:
        temp += w+ ' '
    adverb_phrases.append(temp)
```

Figure 4: Adverb Phrase

```
Top adjective phrases by frequency:
so much  8
very curious  6
very glad  5
very much  4
very little  4
very likely  3
so many  3
too much  3
very tired  2
quite natural  2
very deep  2
n't much  2
very few  2
very good  2
as much  2
so grave  2
always ready  2
very uncomfortable  2
almost wish  2
very difficult  2
quite silent  2
certainly too much  2
not much  2
so large  2
n't very civil  2
very interesting  2
once more  2
very sleepy  1
so VERY remarkable  1
so VERY much  1
too dark  1
VERY good  1
rather glad  1
no longer  1
too large  1
too small  1
not much larger  1
really impossible  1
almost certain  1
very nice  1
now only ten  1
too slippery  1
very fond  1
very small  1
quite surprised  1
quite dull  1
now more  1
so desperate  1
very hot  1
ever so many  1
```

Figure 5: Carroll's Top 50 Adjective Phrases

```
Top adverb phrases by frequency:
as well  15
very soon  6
very politely  5
down here  4
very much  4
just as well  4
so VERY  3
very well  3
so far  3
Just then  3
back again  3
well enough  3
down again  3
as soon  3
very carefully  3
As soon  3
so often  3
not quite  3
very nearly  3
n't quite  3
as long  3
just now  3
very slowly  2
very earnestly  2
not even  2
too far  2
as hard  2
rather not  2
very gravely  2
very humbly  2
very angrily  2
so easily  2
certainly too  2
rather timidly  2
'All right  2
n't very  2
'Exactly so  2
asleep again  2
never even  2
n't even  2
never before  1
never once  1
suddenly down  1
not much  1
not here before  1
too long  1
VERY deeply  1
now only  1
not possibly  1
quite plainly  1
```

Figure 6: Carroll's Top 50 Adverb Phrases

```
Top adjective phrases by frequency:
not worth  3
too much  3
not important  2
'Do many  2
very serious  2
so much  2
absolutely perfect  2
pretty good  2
very fine  2
n't engaged/  2
too many  2
very hot  2
too big  2
n't worth  2
quite dark  2
so big  2
then smaller  2
very hungry  2
too hot  2
so soused  1
very sick  1
very big  1
very bad  1
very pale  1
away wet  1
terribly sorry  1
very exceptional  1
very many  1
quite sure  1
very lazy  1
very uncomfortable  1
awfully surprised  1
n't striking  1
about right  1
not quite dark  1
afrightfully hot  1
simply priceless  1
Too heavy  1
very jine  1
sometimes slept  1
'How much  1
n't practical  1
consciously practical  1
quite proud  1
thoroughly practical  1
awfully big  1
very wise  1
probably bad  1
n't drunk  1
really drunk  1
```

Figure 7: Hemingway's Top 50 Adjective Phrases

```
Top adverb phrases by frequency:
'All right  10
n't ever  6
n't really  3
down beside  3
as well  2
not quite  2
as far  2
here now  2
n't much  2
n't so  2
up again  2
As soon  2
back there  2
far down  2
very hard  1
farther ahead  1
very badly  1
Just then  1
very carefully  1
rather not  1
away so  1
pretty quietly  1
'Hardly ever  1
yellow almost  1
once again  1
far behind  1
too late  1
'As long  1
so far away  1
n't drunk  1
n't there  1
n't even  1
'So long  1
sore as  1
'Were n't  1
still quite  1
Outside now  1
no longer so  1
not even very  1
very quietly  1
back again  1
ahead brilliantly  1
carefully away  1
probably not  1
absolutely unexpectedly  1
back much too  1
back so  1
so long back  1
away only  1
not very  1
```

Figure 8: Hemingway's Top 50 Adverb Phrases

ii. Adjective and Adverb Tokens

Carroll and Hemingway's tagged texts were input and coded for extracting the lists of adjective_tokens (figure 9) and adverb tokens (figure 10). Then, statistical analysis can be performed to determine the length (len(advph_tokens) or len(advph_tokens)), and the frequency of the top 50 adjectives (figure 11 & 13) and adverb (figure 12 & 14) tokens of Carroll and Hemingway.

```
adjective_tokens = []
for sentence in taggedtext:
    for word, pos in sentence:
        if pos in ['JJ', 'JJR', 'JJS']: # adjective, comparative, superlative
            if len(word)>1:
                adjective_tokens.append(word)
freq_adjective = nltk.FreqDist(adjective_tokens)

for word, freq in freq_adjective.most_common(50):
    print(word,freq)
```

Figure 9: Adjective Tokens

```
adverb_tokens = []
for sentence in taggedtext:
    for word, pos in sentence:
        if pos in ['RB', 'RBR', 'RBS']: # adverb, comparative, superlative
            if len(word)>1:
                adverb_tokens.append(word)
freq_adverb = nltk.FreqDist(adverb_tokens)

for word, freq in freq_adverb.most_common(50):
    print(word,freq)
```

Figure 10: Adverb Tokens

```
Top adjective tokens by frequency:
little 124
other 40
great 39
much 34
large 33
last 32
more 31
first 31
such 26
poor 25
thought 24
good 24
long 23
same 23
curious 19
sure 19
next 18
old 17
right 16
low 14
high 14
whole 13
mad 13
many 12
glad 11
own 10
small 10
few 9
best 9
different 9
least 9
afraid 8
white 8
ready 8
dear 8
beautiful 8
golden 7
larger 7
enough 7
deep 6
nice 6
dry 6
bright 6
melancholy 6
offended 6
full 6
sharp 6
hot 5
likely 5
nervous 5
```

Figure 11: Carroll's Top 50 Adjective Tokens

```
Top adverb tokens by frequency:
n't 203
not 128
very 126
so 91
again 83
then 72
quite 48
now 47
as 45
just 44
never 41
only 41
here 39
down 36
once 31
well 31
back 31
too 25
rather 25
soon 24
up 24
away 23
yet 21
ever 20
even 17
much 17
more 16
indeed 15
perhaps 14
anxiously 14
hastily 14
first 13
However 13
certainly 13
far 13
suddenly 12
there 12
still 12
about 12
always 12
else 11
hardly 11
enough 11
really 10
nearly 10
So 9
Then 9
angrily 9
together 9
timidly 9
```

Figure 12: Carroll's Top 50 Adjverb Tokens

```
Top adjective tokens by frequency:
old 90
big 75
good 59
other 44
little 42
current 37
long 28
hot 27
black 22
more 21
right 21
heavy 20
first 18
deep 17
white 16
high 16
open 15
much 14
great 14
young 12
Indian 12
hard 12
many 12
full 11
same 11
left 10
solid 10
happy 10
dark 10
dead 10
next 10
sick 9
bad 9
German 9
better 9
clear 9
funny 9
last 9
smaller 9
quiet 8
easy 8
fat 8
yellow 8
sweet 8
crazy 8
fine 8
smooth 8
fast 8
net 8
whole 7
```

Figure 13: Hemingway's Top 50 Adjective Tokens

```
Top adverb tokens by frequency:
n't 209
not 138
back 90
then 83
down 60
up 54
just 50
too 43
very 41
Then 41
away 40
so 38
again 36
always 34
never 34
right 33
there 33
now 32
ever 31
only 25
really 23
here 17
once 17
ahead 16
far 16
along 15
around 15
Now 15
still 14
together 14
quite 11
more 11
'All 10
out 10
forward 10
hard 9
carefully 9
first 9
as 9
well 9
about 9
over 9
maybe 9
slowly 8
sometimes 7
almost 7
even 7
long 7
much 7
later 6
```

Figure 14: Hemingway's Top 50 Adverb Tokens

iii. Statistic

Basic statistic codes are applied to provide the total number of a corpus, sentence tokens, tokens, average length of sentence, and phrase (table 1). Hemingway's statistic shows a higher corpus but a lower number of adjective and adverb phrases or tokens to Carroll's statistic. This observation could indicate Hemingway might have a higher usage of nouns or verbs, which need to be further investigated and confirmed.

Table 1: Basic Statistic for Hemingway and Carroll Corpus.

	Code	Hemingway	Carroll
# of Total Corpus	total_corpus = sum(len(sent) for sent in textsplit)	163807	141812
# of Sentence	len(tokentext)	2750	1625
Average Length of Sentence	total_corpus/ len(tokentext)	59.2	87.3
# of Adjective Phrase	len(adjph_tags)	160	222
# of Adverb Phrase	len(advph_tags)	140	235
# of Adjective Tokens	len(adjective_tokens)	1745	1488
# of Adverb Tokens	len(adverb_tokens)	1895	2107
# of Adjective Whole Sentence	len(adjph_whole_sentences)	343	464
# of Adverb Whole Sentence	len(advph_whole_sentences)	291	484
Average length of an adjective phrase sentence	total_adjph_sentences / len(adjph_whole_sentences)	9.15	9.20

2. Results of the analysis and interpretation

Hemingway's writing style is known to be concise, factual, and unadorned style. His sentences are usually short, which has 2750 sentences in the corpus. Carroll's writing style is a nonsensical style with a whimsical way of using words and long sentences, which has 1625 sentences in the corpus and is lesser than Hemingway. Hemingway's total corpus and sentence tokens are higher than Carroll's corpus by 13% and Carroll's sentence by 40%.

Compared to both sentence examples (Figure 15), Hemingway has 13 tokens, whereas Carroll has 60. With a comparable number of tokens in both corpora, Carroll's example has showcased the tendency to

write longer sentences than Hemingway. Also, this is further supported by the average length of sentence by statistic performed in Carroll is ~32% more than Hemingway.

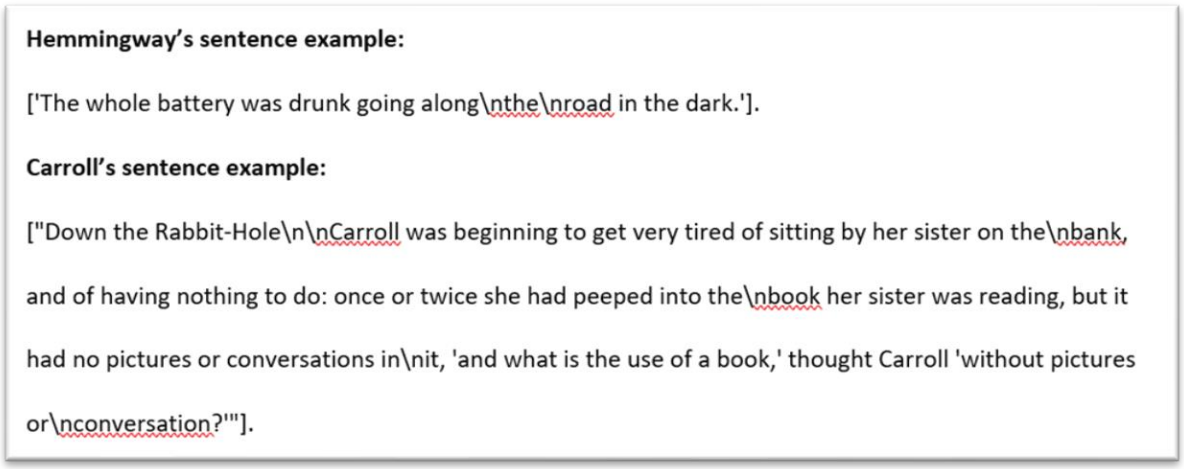

Figure 15: Sentence Example of Hemingway and Carroll

Adjective and adverb phrases are extracted from the Hemingway and Carroll corpora. Figure 16 shows Carroll has 27% more in adjective phrases and 40% more in adverb phrases than Hemingway. Adjective and adverb tokens are also extracted from the Hemingway and Carroll corpora. Figure 17 shows Carroll has 15% less in adjective tokens and 11% more in adverb tokens than Hemingway. Also, this indicated that Hemingway wrote fewer adjective phrases than Carroll, but he preferred a single word adjective in his sentence instead of a phrase or preferred verbs or nouns.

The combination of phrases and tokens for adjective and adverb in whole sentences (figure 18) consistently shows that Carroll has 35% higher in adjectives and 66% higher in adverbs than Hemingway. Also, this is supported by the average length of an adjective sentence in Carroll is 5% more than Hemingway, which is surprisingly not significantly more.

The conclusion from these results confirmed that Hemingway tended to use simple words and fewer adjectives in his writing style and adverbs. Also, these results have shown that Carroll crafted elaborate long sentences with lots of adjective and adverb phrases. Statistical analysis on both corpora has been

demonstrated that Carroll has a higher usage of adjectives and adverbs than Hemingway. Lastly, Carroll used more adverbs than adjectives, which was an unexpected finding from this assignment.

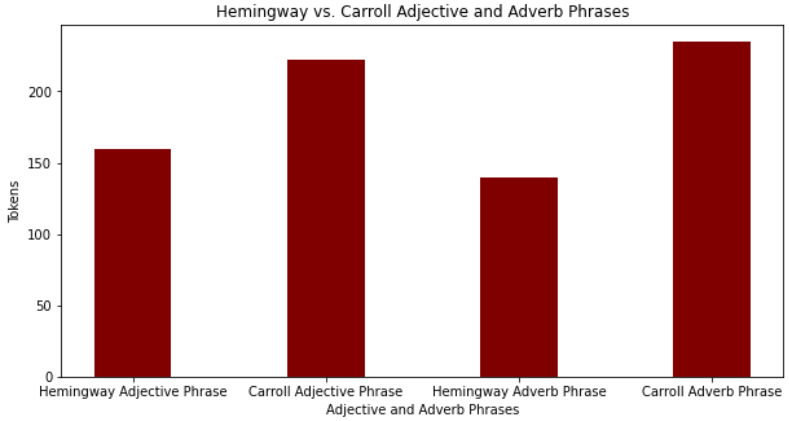

Figure 15: Hemingway vs. Carroll Adjective and Adverb Phrases

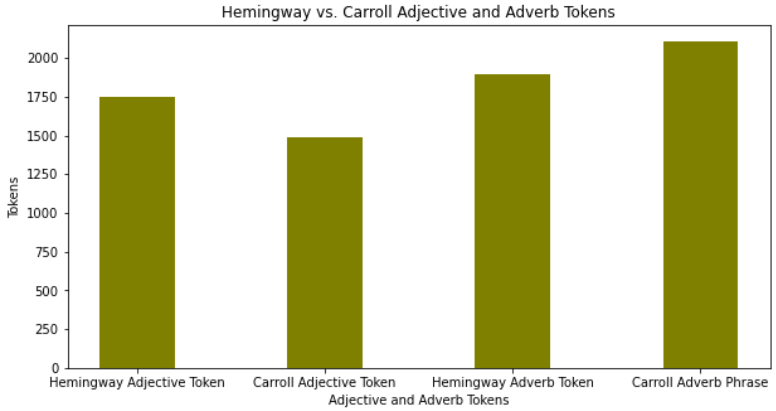

Figure 16: Hemingway vs. Carroll Adjective and Adverb Tokens

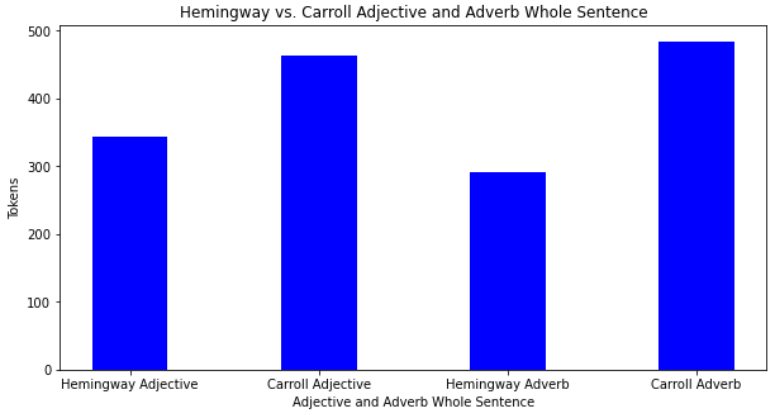

Figure 17: Hemingway vs. Carroll Adjective and Adverb Whole Sentence

3. Thoughts on sentiment analysis for Hemingway and Carroll Corpora

Hemingway and Carroll's tokens should be further cleaned by removing certain words like "n't" from the adverb_phrase and adverb_token lists that are not suitable for sentimental analysis. The generated list of adjective phrases, tokens, and combined phrases plus tokens and adverbs can be applied for sentiment analysis. The sentiment analysis can score if Hemingway or Carroll corpus is positive, negative, or neutral. Hemingway tends to be a serious and factual writer, which would be interesting to learn from the sentiment analysis if his corpus has more a negative or neutral score. Unlike Hemingway, Carroll tends to be whimsical and satirical, which could be more positive than negative. Overall, the sentimental analysis will provide further insights into the author's writing style, emotions, and content of both corpora.