EDS 231: Assignment 2

Juliet Cohen, Charles Hendrickson,Scout Leonard, Peter Menzies

05/30/2022

Download Data

Three Mile Island Accident data

```
threemileisland_files <- list.files(pattern = ".docx",
                                    path = here("data/threemileisland"),
                                    full.names = TRUE,
                                    recursive = TRUE,
                                    ignore.case = TRUE)

threemileisland_data <- lnt_read(threemileisland_files) #Object of class 'LNT output'
```

```
## Creating LNToutput from 1 file...
##  ...files loaded [0.07 secs]
##  ...articles split [0.079 secs]
##  ...lengths extracted [0.081 secs]
##  ...headlines extracted [0.082 secs]
##  ...newspapers extracted [0.083 secs]
##  ...dates extracted [0.092 secs]
##  ...authors extracted [0.093 secs]
##  ...sections extracted [0.094 secs]
##  ...editions extracted [0.094 secs]
##  ...dates converted [0.10 secs]
##  ...metadata extracted [0.11 secs]
##  ...article texts extracted [0.11 secs]
##  ...superfluous whitespace removed [0.11 secs]
## Elapsed time: 0.12 secs
```

```
threemileisland_meta_df <- threemileisland_data@meta %>%
  rename(Art_ID = ID)

threemileisland_articles_df <- threemileisland_data@articles

threemileisland_paragraphs_df <- threemileisland_data@paragraphs
```

```
#joined df with paraagraphs and dates
threemileisland_df <- full_join(threemileisland_paragraphs_df,
                                threemileisland_meta_df)
```

```
## Joining, by = "Art_ID"
```

Chernobyl disaster data

```
chernobyl_files <- list.files(pattern = ".docx",
                              path = here("data/chernobyl"),
                              full.names = TRUE,
                              recursive = TRUE,
                              ignore.case = TRUE)

chernobyl_data <- lnt_read(chernobyl_files) #Object of class 'LNT output'
```

```
## Creating LNToutput from 1 file...
##  ...files loaded [0.093 secs]
##  ...articles split [0.11 secs]
##  ...lengths extracted [0.11 secs]
##  ...headlines extracted [0.11 secs]
##  ...newspapers extracted [0.11 secs]
##  ...dates extracted [0.13 secs]
##  ...authors extracted [0.13 secs]
##  ...sections extracted [0.13 secs]
##  ...editions extracted [0.13 secs]
## Warning in lnt_asDate(date.v, ...): More than one language was detected. The
## most likely one was chosen (English 97%)
##  ...dates converted [0.13 secs]
##  ...metadata extracted [0.14 secs]
##  ...article texts extracted [0.14 secs]
##  ...superfluous whitespace removed [0.15 secs]
## Elapsed time: 0.15 secs
```

```
chernobyl_meta_df <- chernobyl_data@meta %>%
  rename(Art_ID = ID)

chernobyl_articles_df <- chernobyl_data@articles
chernobyl_paragraphs_df <- chernobyl_data@paragraphs

#joined chernobyl df with paraagraphs and dates
chernobyl_df <- full_join(chernobyl_paragraphs_df,
                                chernobyl_meta_df)
```

```
## Joining, by = "Art_ID"
```

fukushima disaster data

```
# NOTE: FOR DISCUSSION, DESCRIBE HOW WE TOOK MEAN SENITMENT OF PARAGRAPHS RATHER THAN ARTICLE (PERHAPS

fukushima_files <- list.files(pattern = ".docx",
                          path = here("data/fukushima"),
                          full.names = TRUE,
                          recursive = TRUE,
                          ignore.case = TRUE)

fukushima_data <- lnt_read(fukushima_files) #Object of class 'LNT output'
```

```
## Creating LNToutput from 1 file...
##  ...files loaded [0.12 secs]
##  ...articles split [0.13 secs]
##  ...lengths extracted [0.13 secs]
##  ...headlines extracted [0.13 secs]
##  ...newspapers extracted [0.13 secs]
##  ...dates extracted [0.15 secs]
##  ...authors extracted [0.15 secs]
##  ...sections extracted [0.15 secs]
##  ...editions extracted [0.15 secs]
## Warning in lnt_asDate(date.v, ...): More than one language was detected. The
## most likely one was chosen (English 99%)
##  ...dates converted [0.16 secs]
##  ...metadata extracted [0.16 secs]
##  ...article texts extracted [0.16 secs]
##  ...superfluous whitespace removed [0.17 secs]
## Elapsed time: 0.17 secs
```

```
fukushima_meta_df <- fukushima_data@meta %>%
  rename(Art_ID = ID)

fukushima_articles_df <- fukushima_data@articles

fukushima_paragraphs_df <- fukushima_data@paragraphs

#joined df with paragraphs and dates
fukushima_df <- full_join(fukushima_paragraphs_df,
                                fukushima_meta_df)
```

```
## Joining, by = "Art_ID"
```

Sentiment time series

Read in sentiment analysis Lexicon:

```
bing_sent <- get_sentiments('bing')
```

Three Mile Island disaster

Process data First, we create a dataframe that contains every sentence from publications before and after the disaster and their sentiments:

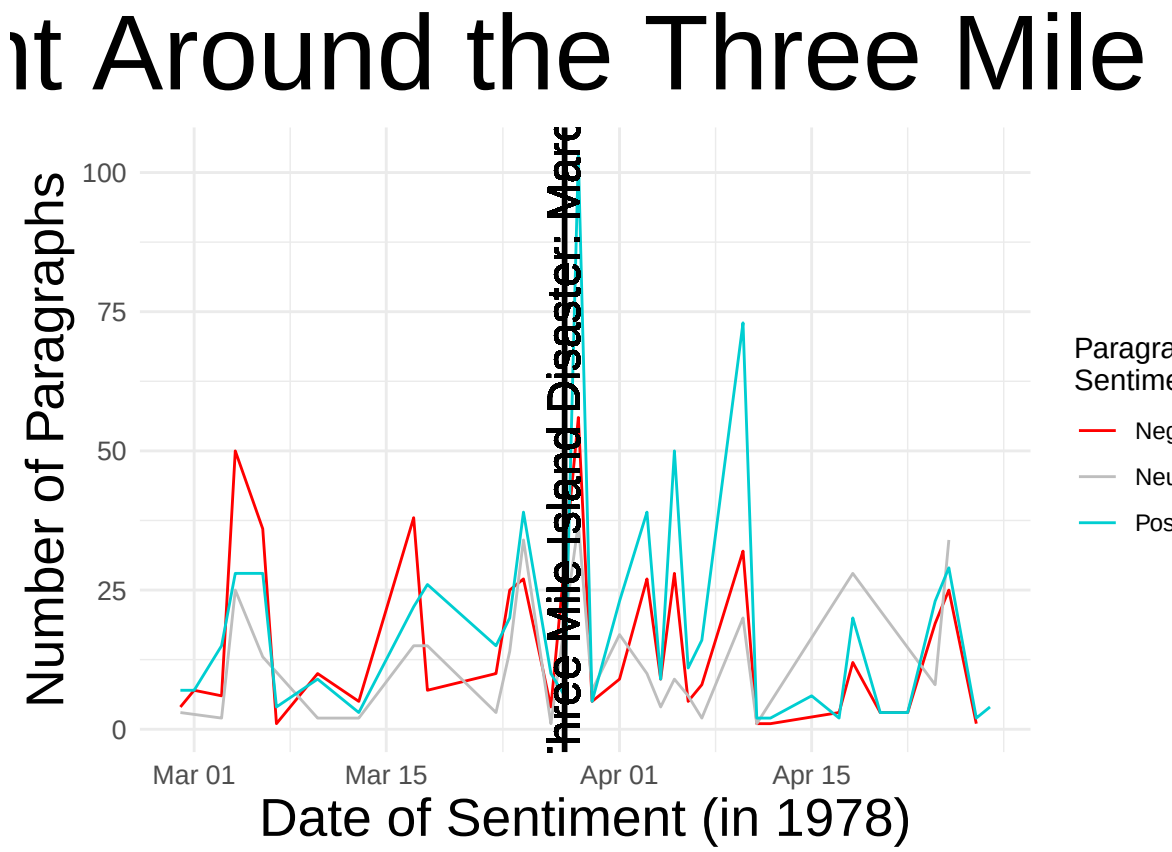
```
#get sentences from three mile island paragraphs
threemileisland_text <- get_sentences(threemileisland_df$Paragraph)

#get sentiment from each sentence
threemileisland_sent <- sentiment(threemileisland_text)

#join sentiment dataframe with article info (this way, our dataframe contains dates)
threemileisland_sent_df <- inner_join(threemileisland_df,
                                      threemileisland_sent,
                                      by = c("Par_ID" = "element_id")) %>%
  select(c("Par_ID", "sentiment", "Paragraph", "Date", "Newspaper"))
```

Next, we find the mean sentiment of each paragraph in all our publication data for this event:

```
#find mean sentiment per paragraph
threemileisland_sent_paragraphs <- threemileisland_sent_df %>%
  group_by(Par_ID) %>%
  summarise(paragraph_sent  = mean(sentiment))

# catgorize for ggplot
threemileisland_sent <- threemileisland_sent_paragraphs %>%
  mutate(sent_category = case_when(
    paragraph_sent < 0 ~ "Negative",
    paragraph_sent == 0 ~ "Neutral",
    paragraph_sent > 0 ~ "Positive"
  )) %>%
  inner_join(threemileisland_df, by = "Par_ID")

#generate counts of sentiment sentences by date to plot
threemileisland_sent_plot <- threemileisland_sent %>%
  count(sent_category, Date)
```

```
#plot it UP

threemileisland_date <- as.Date("1978-03-28")
threemileisland_date_text <- as.Date("1978-03-26")

ggplot(data = threemileisland_sent_plot, aes(x = Date, y = n)) +
        geom_line(aes(color = sent_category)) +
  theme_minimal() +
  labs(y = "Number of Paragraphs",
       x = "Date of Sentiment (in 1978)",
       title = "News Sentiment Around the Three Mile Island Disaster",
       color = "Paragraph\nSentiment") +
  scale_color_manual(values = c("red", "grey", "darkturquoise")) +
  theme(plot.title = element_text(size = 40, hjust = 0.5),
```

```
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10)) +
  geom_vline(xintercept = threemileisland_date,
             col = "black",
             size = 1) +
 geom_text(aes(x = threemileisland_date_text,
               label = "Three Mile Island Disaster: March 28, 1978",
               y = 70),
           angle = 90,
           vjust = 1.3,
           size = 6,
           color = "black")
```



Visualize sentiment

```
ggsave(filename = here("data",
                       "threemileisland",
                       "threemileisland_sentiment_plot.jpeg"),
       plot = last_plot(),
       width = 15,
       height = 10)
```

Chernobyl disaster

```
chernobyl_text <- get_sentences(chernobyl_df$Paragraph)
chernobyl_sent <- sentiment(chernobyl_text) # neutral = 0.0000, scale -1 to 1, 0.06 = slightly positive

# this tool considers more than just individual words, it understands the diff btw "it is too late" and
```

```
chernobyl_sent_df <- inner_join(chernobyl_df, chernobyl_sent,
                                by = c("Par_ID" = "element_id")) %>%
  # select only relevant columns
  select(Par_ID, Paragraph, Newspaper, Date, sentiment)
```

Process data

```
# approximate the sentiment (polarity of text) by grouping by paragraph
chernobyl_sent_summary <- chernobyl_sent_df %>%
  group_by(Par_ID) %>%
  summarize(sentiment_average = mean(sentiment))

chernobyl_sent_summary_date <- inner_join(chernobyl_sent_summary,
                                          chernobyl_sent_df, by = "Par_ID") %>%
  select(Par_ID, sentiment_average, Date)

# add pos, neg, neutral words according to the value of sentiment
chernobyl_pos_neg_neutral <- chernobyl_sent_summary_date %>%
  mutate(sent_category = case_when(
      sentiment_average < 0 ~ "Negative",
      sentiment_average > 0 ~ "Positive",
      T ~ "Neutral")) %>%
  count(sent_category, Date)

# convert the date column to class date
chernobyl_pos_neg_neutral$Date <- anydate(chernobyl_pos_neg_neutral$Date)
#class(chernobyl_pos_neg_neutral$Date)

# make the date of the event an object so we can create a vline on the plot on that day
chernobyl_event <- as.Date("1986-04-26")

chernobyl_plot <- ggplot(chernobyl_pos_neg_neutral, aes(x = Date, y = n)) +
  geom_line(aes(color = sent_category)) +
  theme_minimal() +
  theme(plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10)) +
  # change colors in legend
  scale_color_manual(values = c("red", "grey", "darkturquoise")) +
  geom_vline(xintercept = chernobyl_event,
             size = 1,
             color = "black") +
  geom_text(aes(x = chernobyl_event,
                label = "Chernobyl Disaster: April 26th, 1986",
                y = 150),
            angle = 90,
            vjust = 1.3,
            size = 6,
            color = "black") +
  labs(x = "Date of Sentiment (in 1986)",
       y = "Number of Paragraphs",
       title = "News Sentiment Around Chernobyl Disaster",
```

```
       color = "Paragraph\nSentiment")

# ggsave(filename = here("data",
#                        "chernobyl",
#                        "chernobyl_sentiment_plot.jpeg"),
#        plot = chernobyl_plot,
#        width = 15,
#        height = 10)
```

Visualize sentiment time series

Fukushima disaster

```
fukushima_text <- get_sentences(fukushima_df$Paragraph)
fukushima_sent <- sentiment(fukushima_text) # neutral = 0.0000, scale -1 to 1, 0.06 = slightly positive

# this tool considers more than just individual words, it understands the diff btw "it is too late" and
fukushima_sent_df <- inner_join(fukushima_df,
                                fukushima_sent,
                                by = c("Par_ID" = "element_id")) %>%
  # select only relevant columns
  select(Par_ID, Paragraph, Newspaper, Date, sentiment)
```

Process data

```
# approximate the sentiment (polarity of text) by grouping by paragraph
fukushima_sent_summary <- fukushima_sent_df %>%
  group_by(Par_ID) %>%
  summarize(sentiment_average = mean(sentiment))

fukushima_sent_summary_date <- inner_join(fukushima_sent_summary,
                                          fukushima_sent_df,
                                          by = "Par_ID") %>%
  select(Par_ID, sentiment_average, Date)

# check to see if any NA values are present in the df
apply(is.na(fukushima_sent_summary_date), 2, sum) # the argument "2" tells the function to sum over col
```

Visualize sentiment time series

```
##            Par_ID sentiment_average              Date
##                 0                 0                 7
```

```
# there are 7 missing dates...but idk why because i did an inner join. TBD later if it turns out to be

# add pos, neg, neutral words according to the value of sentiment
fukushima_pos_neg_neutral <- fukushima_sent_summary_date %>%
  mutate(sent_category = case_when(
      sentiment_average < 0 ~ "Negative",
      sentiment_average > 0 ~ "Positive",
      T ~ "Neutral")) %>%
  count(sent_category, Date) # count function used here is the same as group_by(sent_category, Date) %>
```

```
class(fukushima_pos_neg_neutral$Date) # CHARACTER GOSH DARNIT
```

```
## [1] "Date"
```

```
# convert the date column to class date
fukushima_pos_neg_neutral$Date <- anydate(fukushima_pos_neg_neutral$Date)

# make the date of the event an object so we can create a vline on the plot on that day
fukushima_event <- as.Date("2011-03-11")

fukushima_plot <- ggplot(fukushima_pos_neg_neutral, aes(x = Date, y = n)) +
  geom_line(aes(color = sent_category)) +
  theme_minimal() +
  theme(plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10)) +
  # change colors in legend
  scale_color_manual(values = c("red", "grey", "darkturquoise")) +
  geom_vline(xintercept = fukushima_event,
             size = 1,
             color = "black") +
  geom_text(aes(x = fukushima_event,
                label = "Fukushima Disaster: March 11th, 2011",
                y = 150),
             angle = 90,
             vjust = 1.3,
             size = 6,
             color = "black") +
  labs(x = "Date of Sentiment (in 2011)",
       y = "Number of Paragraphs",
       title = "News Sentiment Around the Fukushima Disaster",
       color = "Paragraph\nSentiment")

# ggsave(filename = here("data",
#                        "fukushima",
#                        "fukushima_sentiment_plot.jpeg"),
#        plot = fukushima_plot,
#        width = 15,
#        height = 10)
```

Emotion time series

```
#get nrc emotions lexicon
nrc_sent <- get_sentiments('nrc') %>%
  filter(sentiment != "negative" & sentiment != "positive") %>%
  rename(emotion = sentiment)
```

Three Mile Island disaster

```
# Splitting into individual words
threemile_words <- threemileisland_df %>%
  clean_names() %>%
  select(!c(source_file, newspaper, graphic, headline, length:edition)) %>%
```

```
  filter(!is.na(date)) %>%
  unnest_tokens(output = word, input = paragraph, token = 'words')

# Joining with nrc emotion words
threemile_emotion_counts <- threemile_words %>%
  inner_join(nrc_sent, by = "word") %>%
  count(emotion, date, sort = TRUE)

# Calculating percent of total emotion words used that day
threemile_emotion_pct <- threemile_emotion_counts %>%
  group_by(date) %>%
  mutate(percentiment = (n / sum(n)) * 100)
```

Tokenize article words and assign emotion categories

```
threemile_emotion_plot <- ggplot(threemile_emotion_pct, aes(x = date, y = percentiment)) +
  geom_line(aes(color = emotion), size = 0.75) +
  theme_minimal() +
  geom_vline(xintercept = as.numeric(threemileisland_df$Date[threemileisland_date]),
             col = "black") +
  labs(x = "Date",
       y = "Percent of total emotion words used",
       title = "Frequency of emotional sentiment words used in media\n surrounding Three Mile Island dis
       color = "Emotion") +
  theme(plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10)) +
  geom_vline(xintercept = threemileisland_date,
             col = "black",
             size = 1) +
 geom_text(aes(x = threemileisland_date_text,
               label = "Three Mile Island Disaster: March 28, 1978",
               y = 50),
           angle = 90,
           vjust = 1.3,
           size = 5,
           color = "black")

# ggsave(filename = here("data",
#                        "threemileisland",
#                        "threemileisland_emotion_plot.jpeg"),
#        plot = threemile_emotion_plot,
#        width = 15,
#        height = 10)
#
# ggplotly(threemile_emotion_plot)
```

Visualize emotion frequency over time

Chernobyl disaster

```r
# Splitting into individual words
chernobyl_words <- chernobyl_df %>%
  clean_names() %>%
  select(!c(source_file, newspaper, graphic, headline, length:edition)) %>%
  filter(!is.na(date)) %>%
  unnest_tokens(output = word, input = paragraph, token = 'words')

# Joining with nrc emotion words
chernobyl_emotion_counts <- chernobyl_words %>%
  inner_join(nrc_sent, by = "word") %>%
  count(emotion, date, sort = TRUE)

# Calculating percent of total emotion words used that day
chernobyl_emotion_pct <- chernobyl_emotion_counts %>%
  group_by(date) %>%
  mutate(percentiment = (n / sum(n)) * 100)
```

Tokenize article words and assign emotion categories

```r

chernobyl_emotion_plot <- ggplot(chernobyl_emotion_pct,
                                 aes(x = date, y = percentiment)) +
  geom_line(aes(color = emotion), size = 0.75) +
  theme_minimal() +
  labs(x = "Date",
       y = "Percent of total emotion words used",
       title = "Frequency of emotional sentiment words used in media\n surrounding Chernobyl disaster",
        color = "Emotion") +
  geom_vline(xintercept = chernobyl_event,
             size = 1,
             color = "black") +
  geom_text(aes(x = chernobyl_event,
                label = "Chernobyl Disaster: April 26th, 1986",
                y = 150),
             angle = 50,
             vjust = 1.3,
             size = 6,
             color = "black")

# ggsave(filename = here("data",
#                        "chernobyl",
#                        "chernobyl_emotion_plot.jpeg"),
#        plot = chernobyl_emotion_plot,
#        width = 15,
#        height = 10)
#
# ggplotly(chernobyl_emotion_plot)
```

Visualize emotion frequency over time

fukushima disaster

```r
# Splitting into individual words
fukushima_words <- fukushima_df %>%
  clean_names() %>%
  select(!c(source_file, newspaper, graphic, headline, length:edition)) %>%
  filter(!is.na(date)) %>%
  unnest_tokens(output = word, input = paragraph, token = 'words')

# Joining with nrc emotion words
fukushima_emotion_counts <- fukushima_words %>%
  inner_join(nrc_sent, by = "word") %>%
  count(emotion, date, sort = TRUE)

# Calculating percent of total emotion words used that day
fukushima_emotion_pct <- fukushima_emotion_counts %>%
  group_by(date) %>%
  mutate(percentiment = (n / sum(n)) * 100)
```

Tokenize article words and assign emotion categories

```r

fukushima_emotion_plot <- ggplot(fukushima_emotion_pct, aes(x = date, y = percentiment)) +
  geom_line(aes(color = emotion), size = 0.75) +
  geom_vline(xintercept = fukushima_event,
             col = "black") +
  theme_minimal() +
  labs(x = "Date",
       y = "Percent of total emotion words used",
       title = "Frequency of emotional sentiment words used in media\n surrounding Fukushima disaster",
       color = "Emotion") +
  geom_vline(xintercept = fukushima_event,
             size = 1,
             color = "black") +
  geom_text(aes(x = fukushima_event,
                label = "Fukushima Disaster: March 11th, 2011",
                y = 40),
             angle = 90,
             vjust = 1.3,
             size = 5,
             color = "black") +
  theme(plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10))

ggsave(filename = here("data",
                       "fukushima",
                       "fukushima_emotion_plot.jpeg"),
       plot = fukushima_emotion_plot,
       width = 15,
       height = 10)

#ggplotly(fukushima_emotion_plot)
```

Visualize emotion frequency over time

Word frequency

Three Mile Island

```
threemile_words <- threemile_words %>%
  anti_join(stop_words)
```

Clean tokens

```
## Joining, by = "word"
```

```
#remove numbers
threemile_clean_tokens <- str_remove_all(threemile_words$word, "[:digit:]")

#removes appostrophes
threemile_clean_tokens <- gsub("'s", '', threemile_clean_tokens)

threemile_words$clean <- threemile_clean_tokens

#remove the empty strings
threemile_tib <-subset(threemile_words, clean != "")

#reassign
threemile_tokenized <- threemile_tib
```

```
threemile_wordfreq_plot <- threemile_tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 50) %>%
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(x = n, y = clean)) +
  geom_col(aes(fill = clean)) +
  labs(y = NULL) +
  theme_minimal() +
  labs(x = "Word Frequency",
       title = "Word Frequency from Media Surrounding \nthe Three Mile Island Disaster") +
  scale_fill_manual(values = rep(met.brewer("Hokusai1", n = 7), times = 4)) +
  theme(legend.position = "none",
        plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10))

ggsave(filename = here("data",
                       "threemileisland",
                       "threemileisland_workfreq_plot.jpeg"),
       plot = threemile_wordfreq_plot,
       width = 15,
       height = 10)
```

Plot word frequency

Chernobyl disaster

```
chernobyl_words <- chernobyl_words %>%
  anti_join(stop_words)
```

Clean tokens

```
## Joining, by = "word"
```

```
#remove numbers
chernobyl_clean_tokens <- str_remove_all(chernobyl_words$word, "[:digit:]")

#removes appostrophes
chernobyl_clean_tokens <- gsub("'s", '', chernobyl_clean_tokens)

chernobyl_words$clean <- chernobyl_clean_tokens

#remove the empty strings
chernobyl_tib <-subset(chernobyl_words, clean != "")

#reassign
chernobyl_tokenized <- chernobyl_tib
```

```
chernobyl_wordfreq_plot <- chernobyl_tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 70) %>%
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(x = n, y = clean)) +
  geom_col(aes(fill = clean)) +
  labs(y = NULL) +
  theme_minimal() +
  labs(x = "Word Frequency",
       title = "Word Frequency from Media Surrounding \nthe Chernobyl Disaster") +
  scale_fill_manual(values = rep(met.brewer("Hokusai1", n = 7), times = 3)) +
  theme(legend.position = "none",
        plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10))

ggsave(filename = here("data",
                       "chernobyl",
                       "chernobyl_workfreq_plot.jpeg"),
       plot = chernobyl_wordfreq_plot,
       width = 15,
       height = 10)
```

Plot word frequency

Fukushima disaster

```
fukushima_words <- fukushima_words %>%
  anti_join(stop_words)
```

Clean tokens

```
## Joining, by = "word"
```

```
#remove numbers
fukushima_clean_tokens <- str_remove_all(fukushima_words$word, "[:digit:]")

#removes appostrophes
fukushima_clean_tokens <- gsub("'s", '', fukushima_clean_tokens)

fukushima_words$clean <- fukushima_clean_tokens

#remove the empty strings
fukushima_tib <-subset(fukushima_words, clean != "")

#reassign
fukushima_tokenized <- fukushima_tib
```

```
fukushima_wordfreq_plot <- fukushima_tokenized %>%
  count(clean, sort = TRUE) %>%
  filter(n > 70) %>%
  mutate(clean = reorder(clean, n)) %>%
  ggplot(aes(x = n, y = clean)) +
  geom_col(aes(fill = clean)) +
  labs(y = NULL) +
  theme_minimal() +
  labs(x = "Word Frequency",
       title = "Word Frequency from Media Surrounding \nthe Fukushima Disaster") +
  scale_fill_manual(values = rep(met.brewer("Hokusai1", n = 7), times = 4)) +
  theme(legend.position = "none",
        plot.title = element_text(size = 40, hjust = 0.5),
        axis.title = element_text(size = 20),
        axis.text = element_text(size = 10))

ggsave(filename = here("data",
                       "fukushima",
                       "fukushima_workfreq_plot.jpeg"),
       plot = fukushima_wordfreq_plot,
       width = 15,
       height = 10)
```

Plot word frequency