Map od the code

Claudia Morales Valiente

2024-10-02

Map of the loops

1. Loop 1: Iterating over each row to create Document-Term Matrices (DTMs)

```r
for (i in 1:total_rows) {
  row_df <- text[i, ]

  # Check if the tokens column is available and tokenize
  if (!is.list(row_df$tokens)) {
    tokens_data <- tokens(row_df$tokens)
  } else {
    tokens_data <- row_df$tokens
  }

  # Create the DTM (document-feature matrix)
  dtm <- dfm(tokens_data)  # Convert the tokenized text to a DTM

  # Create a unique name for each DTM
  doc_name <- paste0("P", i, "_dtm")
  dtm_list[[doc_name]] <- dtm   # Store the DTM for each individual row

  # Update the progress bar
  setTxtProgressBar(pb, i)
}
```

Purpose:

- This loop processes each row of the `text` data frame, which represents one document (or participant).

Breakdown:

1. **row_df <- text**[**i,**]:
   - For each iteration `i`, a single row (document) is selected from the `text` data frame.
2. **Tokenization**:
   - If the `tokens` column isn't already a list (i.e., if it's a string), it tokenizes the text using the `tokens()` function from the `quanteda` package.

3. **Create DTM**:
    - The `dfm()` function is used to create a Document-Term Matrix (DTM) from the tokenized text. Each DTM contains word counts for each term in that document.
4. **Store the DTM**:
    - A unique name is generated for each DTM (`P1_dtm`, `P2_dtm`, etc.), and the DTM is stored in the list `dtm_list`.
5. **Progress Bar Update**:
    - The `setTxtProgressBar()` function updates the progress bar, showing how many rows/documents have been processed.

2. Loop 2: Calculating word frequencies for Lancaster norms

```r
for (dtm_name in names(dtm_list)) {

  # Extract the current DTM (document)
  dtm <- dtm_list[[dtm_name]]

  # Convert DTM to data frame for easier manipulation
  dtm_df <- convert(dtm, to = "data.frame")

  # Create an empty data frame to store frequencies and first match for lanc words for the current docu
  word_frequencies <- data.frame(word = lanc_words)

  # Add a column to track if it's the first match
  word_frequencies$first_match <- 0

  # Loop through each word and calculate frequency and first match
  for (k in 1:nrow(word_frequencies)) {
    word <- word_frequencies$word[k]

    # Check if the word is present in the DTM
    if (word %in% colnames(dtm_df)) {
      word_freq <- sum(dtm_df[, word], na.rm = TRUE)

      # If the word appears and it's the first occurrence, set first_match to 1
      if (word_freq > 0) {
        word_frequencies$frequency[k] <- word_freq
        if (word_frequencies$first_match[k] == 0) {
          word_frequencies$first_match[k] <- 1
        }
      } else {
        word_frequencies$frequency[k] <- 0
      }
    } else {
      word_frequencies$frequency[k] <- 0
    }
  }

  # Store the resulting data frame for the current DTM in the list
  dtm_frequencies[[dtm_name]] <- word_frequencies
  # Update the progress bar
  setTxtProgressBar(pb, match(dtm_name, names(dtm_list)))
}
```

Purpose:

- This loop calculates the frequency of each word from the Lancaster norms in each document (DTM).

Breakdown:

1. **Iterating over each DTM**:
    - The loop iterates over each DTM stored in `dtm_list` by its name.
    - For each DTM, the corresponding document is extracted and stored in the variable `dtm`.
2. **Convert DTM to Data Frame**:
    - The DTM is converted to a data frame (`dtm_df`) for easier manipulation. The resulting data frame has columns representing words and rows representing documents.
3. **Prepare for Frequency Calculation**:
    - A new data frame `word_frequencies` is created to store the frequencies of Lancaster words and whether they were first matches in the document.
4. **Inner Loop**:
    - **for (k in 1:nrow(word_frequencies))**: This loop goes through each word in the `lanc_words` list, checking if it appears in the current document.
    - **Check if the word is present**: If the word exists in the document, its frequency is calculated from the DTM.
    - **First Match**: If it's the first occurrence of the word, the `first_match` column is set to `1`.
5. **Store the results**:
    - The `word_frequencies` data frame, which contains the frequency and first match data for all Lancaster words in the document, is stored in the list `dtm_frequencies`.
6. **Progress Bar Update**:
    - The progress bar is updated after each document's word frequencies are calculated.

3. Loop 3: Calculating Totals for Each Document

```r
for (dtm_name in names(dtm_frequencies)) {

  # Access the individual data frame
  df <- dtm_frequencies[[dtm_name]]

  # Calculate the column-wise total (sum) for numeric columns
  column_totals <- colSums(df[, sapply(df, is.numeric)], na.rm = TRUE)

  # Add a new column for the document identifier
  column_totals <- c(document = dtm_name, column_totals)

  # Store the result in the list with the name of the document (dtm)
  column_totals_list[[dtm_name]] <- column_totals
}
```

Purpose:

- This loop calculates the total sensory scores (e.g., olfactory, gustatory, visual, etc.) for each document.

Breakdown:

1. **Iterating over each document's word frequencies**:
   - The loop goes through each data frame in `dtm_frequencies`. Each data frame corresponds to the word frequencies for a particular document.
2. **Calculate Totals**:
   - **colSums()** is used to calculate the column-wise total (i.e., sum) for each sensory variable in the document's data frame. This sums the frequencies for the sensory-related columns (e.g., olfactory, gustatory).
3. **Store Results**:
   - The totals for each document, along with the document's identifier, are stored in the list `column_totals_list` with the document name as the key.

4. Final Combination

```r
# Combine all the column totals into a matrix
column_totals_matrix <- do.call(rbind, column_totals_list)
```

Purpose:

- This combines the results from all documents into a single matrix using `do.call()` with `rbind()`. Each row of the matrix corresponds to a document, and the columns represent the totals for each sensory variable. The matrix is then printed and saved as a CSV file.