Replication Project

Veronica Pletsch

2026-02-14

Contents

Introduction

The paper I chose to replicate is on the "Application of LDA and BERTopic Algorithms for Topic Analysis of Palestine–Israel Conflict News" by researchers from Indoneasia. I chose this paper because it most closely aligns with the type of methodology I want to implement in my thesis research. The study aims to find the most prominent topic structures within Indonesian news articles covering the Palestine–Israel conflict using Latent Dirichlet Allocation (LDA) and BERTopic algorithms as the empirical methodological approach. Data was gathered via web-scraping three Indonesian news sources for article titles and their descriptions. This includes Detik (510 articles), Kompas (506 articles), and CNN Indonesia (455 articles). Articles collected were based on the timeframe criteria of January 1, 2025 to March 31, 2025. The study found five thematic word clusters, all of which have low perplexity and high coherence scores. Theoretically, the study argues

for the adoption of computational frameworks such as LDA and BERTopic to analyze the latent narratives that exist within media on complex and quickly evolving conflicts.

Analysis Replication

For this study, I was not able to use the original studies dataset as it was not public facing. In consequence, I opted to recreate a US centric version of the dataset. I maintained the original study's parameters, focusing on news articles published between January and March 2025 on the topic of the Palestine-Israel conflict. However, unlike the original study, I was only able to gather the news article's title, not its description. I believe this played a pivotal role in my coherence and perplexity scores during analysis. Additionally, due to how time intensive the web-scrapping was, I was only able to gather a total of 309 articles. This was significantly less than the original study's 1,471 gathered articles. Once again, this discrepancy proved fatal to my iteration as my coherence and perplexity score were completely out of range, thus making my results insignificant.

Critique

The original study was successful in producing meaningful results. In conclusion, we now know that the main narratives covered by Indonesian news media on the Palestine-Israel Conflict include military strikes in Gaza, Indonesian diplomacy, international responses, humanitarian issues, and global political dynamics. From this analysis, it can be argued that Indonesian discourse understands that the Palestine-Israel conflict is more than a war, it is a humanitarian crisis that requires international support. It can also be assessed due to the lack of 'Israel' topics, that Indonesia is more aligned with supporting Palestinians and Gazans than Israel. Although this only provides a small window into the Indonesian discourse, I feel strongly in the potential insight this method can provide once scaled for cross-state comparative analysis. I believe the paper did very well with detailing the study for replicability, including outline the parameters. Moreover, it did well in its thorough cleaning preprocessing procedure, including cleansing, case folding, tokenization, normalization, filtering, and stemming. However, one conceptual issue with the paper is that focuses on news articles title's and descriptions, most likely due to the processing power and time that would be needed to carry out a larger scale analysis of full-text news articles. In consequence, the narrative may be misleading as new articles often carefully select titles with more "click-bait" friendly jargon. This leads to my view that the data does not operationalize the constructs of interest well. However, a counter to my own view of this conceptual issue, is that it leads to a better understanding of how the general public views the conflict in that the general public often relies on the title of a new article of their understanding instead of a full-text reading. Finally, I do believe that the method selected captures the underlying theory effectively as the theory does not argue for more than that the topic narratives uncovered are thematic rather than substantive.

Code

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.6
## v forcats   1.0.1     v stringr   1.6.0
## v ggplot2   4.0.1     v tibble    3.3.0
## v lubridate 1.9.5     v tidyr     1.3.2
## v purrr     1.2.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
```

```
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(tokenizers)
library(tidytext)
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(SnowballC)
library(ggplot2)
library(gt)
library(cld3)
library(topicmodels)
library(textmineR)
```

```
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
##
##
## Attaching package: 'textmineR'
##
## The following object is masked from 'package:Matrix':
##
##     update
##
## The following object is masked from 'package:topicmodels':
##
##     posterior
##
## The following object is masked from 'package:stats':
##
##     update
```

```
library(reticulate)
library(topicdoc)
library(wordcloud2)
```

```r
#load the cnn data
cnn_scraped_data <- read.csv("~/CompIR Replication/scraped_cnn_titles_2025.csv")
#load foxnews data
foxnews_scraped_data <- read.csv("~/CompIR Replication/scraped_foxnews_titles_2025.csv")
#load the msnbc data
msnbc_scraped_data <- read.csv("~/CompIR Replication/scraped_msnbc_titles_2025.csv")
```

```r
#combine all news datasets
all_scraped_data <- bind_rows(cnn_scraped_data, foxnews_scraped_data, msnbc_scraped_data)
```

Code Block 1

To ensure the frequency of all equivalent words within the dataset are properly counted during the LDA analysis, all words are case folded. This standardizes the casing of the words to lowercase so that the algorithm does not mistake "Gaza", "gaza", and "GAZA" as three distinct terms instead of one.

```r
#Case folding
all_scraped_data_2 <- all_scraped_data %>%
  mutate(Title = tolower(Title))
```

Code Block 2

Within this block of code, I used the distinct() function to search and delete duplicate titles. The output dataset (all_scraped_data_3) maintained the same number of observations as its antecedent, indicating that no duplicates were present. Additionally, within this 'cleaning' block of code, an identification number was assigned to each row. This is to ensure that later, after unnesting our words from their context, we are still able to identify which words were once within the same title.

```r
#Cleaning, add ID to each title for analysis later
all_scraped_data_3 <- all_scraped_data_2 %>%
    distinct(Title, .keep_all = TRUE)

all_scraped_data_4 <- all_scraped_data_3 %>%
  mutate(id = row_number())
```

Code Block 3

Now that the data is a bit cleaner and has an ID, it is ready for tokenization. That is, turning our larger units of data, the titles, into smaller units (words) that will be digestible for the LDA algorithm. I also loaded in the stop_words dataset and anti-joined it to my current working dataset. Anti-joining allows for the removal of any stop words (words that don't hold significance such as "the", "and" or "but") in my dataset. Additionally, I went ahead and fixed two spelling errors I noted while checking over my dataset.

```r
#Tokenization
data(stop_words)

all_scraped_data_5 <- all_scraped_data_4 %>%
    unnest_tokens(word, Title)

all_scraped_data_6 <- all_scraped_data_5 %>%
```

```
    anti_join(stop_words, by = "word")

#recode hamas
all_scraped_data_7 <- all_scraped_data_6 %>%
  mutate(word = recode(word,
                       "hama" = "hamas",
                       "jan" = "january"))
```

Code Block 4

Moving on, there were a few insignificant words I noticed still within my dataset. They technically are not stop words so they weren't removed the from anti-join. But since they won't provide much to my analysis later on I went ahead and removed them manually.

```
#Filtering, remove the unncessary words
removed_words <- c("cnn", "january", "2025")

all_scraped_data_8 <- all_scraped_data_7 %>%
  filter(!.data$word %in% removed_words)
```

Code Block 5

Some of my equivalent words still won't be completely identical to the algorith because of punctuation. For example, "gaza" and "gaza's" will be considered as distinct words even though they're not. So, I'm using a mutate function to remove all apostrophes and subsequent 's'.

```
#Normalization
all_scraped_data_9 <- all_scraped_data_8 %>%
  mutate(word = str_replace_all(word, "'s$", "")) %>%
  mutate(word = str_replace_all(word, "'", ""))
```

Code Block 6

I actually first attemped to stem (reduce my words to their root) by using the SnowballC package and the wordStem() function, but it didn't work properly. I'm new to word stemming so I don't think I did it correctly, because for some reason it didn't stem my words, it only removed the last letter of various words that were already at their root such as "fire -> fir". So, instead I attempted to do it manually in a way that is more familiar to me. I used the mutate function to replace non-stemmed words to their stem. This was tedious, especially for 1809 observations, so I only stemmed the most important words I gathered from doing a quick scan. I think it would be beneficial to revisit this code chunk and try again, either by debugging my original code using the wordStem() function or trying alternatives from the 'tm' or 'textstem' packages.

```
#Stemming
all_scraped_data_10 <- all_scraped_data_9 %>%
    mutate(word = str_replace_all(word, "israeli", "israel")) %>%
    mutate(word = str_replace_all(word, "palestinian", "palestine")) %>%
    mutate(word = str_replace_all(word, "gazab", "gaza")) %>%
    mutate(word = str_replace_all(word, "jordanian", "jordan")) %>%
    mutate(word = str_replace_all(word, "egyptian", "egypt")) %>%
    mutate(word = str_replace_all(word, "gazans", "gaza"))
```

Code Block 7

Now that my data is relatively clean and tidy, I needed to make a document term matrix (DTM) that will allow me to run my LDA analysis. The DTM is meant to help the algorithm process my data so that the underlying patterns can emerge. In the middle section of this code chunk I am calling the dtm and setting the number of word clusters (topics) I want (k=30). I initially went with 20 topics, the same as the original study, but settled for 30. My k value definitely influences my perplexity and coherence score so I tried various values from 6 to 30. I settled on 30 because it was my best option for score amongst the other options. In all truth, despite my k value influencing my perplexity and coherence scores, the real culprit was the size of my dataset (too small) and my lack of article descriptions. Next, I set my method to 'Gibbs' to help with deciding which words belong to which topic based on relevance. Using control= I set my seed to 42 for sake of replicability, I set the algorithms iterations to run 1000 times and finally set the progress log to provide mid level detail using verbose=1. Since 1000 iterations is a lot, I wanted to know the progress in case RStudio crashes. I'm honestly not very sure if 1000 iterations is too much or not enough for the size of my dataset, in retrospect it would've been a good idea to try a few different iters before settling. Lastly, I saved my model as an rds file so I don't have to keep running the model, a precaution to avoid the potential of the app crashing.

```
#create DTM then run LDA
dtm <- all_scraped_data_10 %>%
  count(id, word) %>%
  cast_dtm(id, word, n)

#trying to replicate the same number of the original study's key terms so k=30
lda_model <- LDA(dtm, k = 30,
                 method = "Gibbs",
                 control = list(
                     seed = 42,
                     iter = 1000,
                     verbose = 1))
```

```
## K = 30; V = 854; M = 309
## Sampling 1000 iterations!
## Iteration 1 ...
## Iteration 2 ...
## Iteration 3 ...
## Iteration 4 ...
## Iteration 5 ...
## Iteration 6 ...
## Iteration 7 ...
## Iteration 8 ...
## Iteration 9 ...
## Iteration 10 ...
## Iteration 11 ...
## Iteration 12 ...
## Iteration 13 ...
## Iteration 14 ...
## Iteration 15 ...
## Iteration 16 ...
## Iteration 17 ...
## Iteration 18 ...
## Iteration 19 ...
## Iteration 20 ...
## Iteration 21 ...
```

```
## Iteration 22 ...
## Iteration 23 ...
## Iteration 24 ...
## Iteration 25 ...
## Iteration 26 ...
## Iteration 27 ...
## Iteration 28 ...
## Iteration 29 ...
## Iteration 30 ...
## Iteration 31 ...
## Iteration 32 ...
## Iteration 33 ...
## Iteration 34 ...
## Iteration 35 ...
## Iteration 36 ...
## Iteration 37 ...
## Iteration 38 ...
## Iteration 39 ...
## Iteration 40 ...
## Iteration 41 ...
## Iteration 42 ...
## Iteration 43 ...
## Iteration 44 ...
## Iteration 45 ...
## Iteration 46 ...
## Iteration 47 ...
## Iteration 48 ...
## Iteration 49 ...
## Iteration 50 ...
## Iteration 51 ...
## Iteration 52 ...
## Iteration 53 ...
## Iteration 54 ...
## Iteration 55 ...
## Iteration 56 ...
## Iteration 57 ...
## Iteration 58 ...
## Iteration 59 ...
## Iteration 60 ...
## Iteration 61 ...
## Iteration 62 ...
## Iteration 63 ...
## Iteration 64 ...
## Iteration 65 ...
## Iteration 66 ...
## Iteration 67 ...
## Iteration 68 ...
## Iteration 69 ...
## Iteration 70 ...
## Iteration 71 ...
## Iteration 72 ...
## Iteration 73 ...
## Iteration 74 ...
## Iteration 75 ...
```

```
## Iteration 76 ...
## Iteration 77 ...
## Iteration 78 ...
## Iteration 79 ...
## Iteration 80 ...
## Iteration 81 ...
## Iteration 82 ...
## Iteration 83 ...
## Iteration 84 ...
## Iteration 85 ...
## Iteration 86 ...
## Iteration 87 ...
## Iteration 88 ...
## Iteration 89 ...
## Iteration 90 ...
## Iteration 91 ...
## Iteration 92 ...
## Iteration 93 ...
## Iteration 94 ...
## Iteration 95 ...
## Iteration 96 ...
## Iteration 97 ...
## Iteration 98 ...
## Iteration 99 ...
## Iteration 100 ...
## Iteration 101 ...
## Iteration 102 ...
## Iteration 103 ...
## Iteration 104 ...
## Iteration 105 ...
## Iteration 106 ...
## Iteration 107 ...
## Iteration 108 ...
## Iteration 109 ...
## Iteration 110 ...
## Iteration 111 ...
## Iteration 112 ...
## Iteration 113 ...
## Iteration 114 ...
## Iteration 115 ...
## Iteration 116 ...
## Iteration 117 ...
## Iteration 118 ...
## Iteration 119 ...
## Iteration 120 ...
## Iteration 121 ...
## Iteration 122 ...
## Iteration 123 ...
## Iteration 124 ...
## Iteration 125 ...
## Iteration 126 ...
## Iteration 127 ...
## Iteration 128 ...
## Iteration 129 ...
```

```
## Iteration 130 ...
## Iteration 131 ...
## Iteration 132 ...
## Iteration 133 ...
## Iteration 134 ...
## Iteration 135 ...
## Iteration 136 ...
## Iteration 137 ...
## Iteration 138 ...
## Iteration 139 ...
## Iteration 140 ...
## Iteration 141 ...
## Iteration 142 ...
## Iteration 143 ...
## Iteration 144 ...
## Iteration 145 ...
## Iteration 146 ...
## Iteration 147 ...
## Iteration 148 ...
## Iteration 149 ...
## Iteration 150 ...
## Iteration 151 ...
## Iteration 152 ...
## Iteration 153 ...
## Iteration 154 ...
## Iteration 155 ...
## Iteration 156 ...
## Iteration 157 ...
## Iteration 158 ...
## Iteration 159 ...
## Iteration 160 ...
## Iteration 161 ...
## Iteration 162 ...
## Iteration 163 ...
## Iteration 164 ...
## Iteration 165 ...
## Iteration 166 ...
## Iteration 167 ...
## Iteration 168 ...
## Iteration 169 ...
## Iteration 170 ...
## Iteration 171 ...
## Iteration 172 ...
## Iteration 173 ...
## Iteration 174 ...
## Iteration 175 ...
## Iteration 176 ...
## Iteration 177 ...
## Iteration 178 ...
## Iteration 179 ...
## Iteration 180 ...
## Iteration 181 ...
## Iteration 182 ...
## Iteration 183 ...
```

```
## Iteration 184 ...
## Iteration 185 ...
## Iteration 186 ...
## Iteration 187 ...
## Iteration 188 ...
## Iteration 189 ...
## Iteration 190 ...
## Iteration 191 ...
## Iteration 192 ...
## Iteration 193 ...
## Iteration 194 ...
## Iteration 195 ...
## Iteration 196 ...
## Iteration 197 ...
## Iteration 198 ...
## Iteration 199 ...
## Iteration 200 ...
## Iteration 201 ...
## Iteration 202 ...
## Iteration 203 ...
## Iteration 204 ...
## Iteration 205 ...
## Iteration 206 ...
## Iteration 207 ...
## Iteration 208 ...
## Iteration 209 ...
## Iteration 210 ...
## Iteration 211 ...
## Iteration 212 ...
## Iteration 213 ...
## Iteration 214 ...
## Iteration 215 ...
## Iteration 216 ...
## Iteration 217 ...
## Iteration 218 ...
## Iteration 219 ...
## Iteration 220 ...
## Iteration 221 ...
## Iteration 222 ...
## Iteration 223 ...
## Iteration 224 ...
## Iteration 225 ...
## Iteration 226 ...
## Iteration 227 ...
## Iteration 228 ...
## Iteration 229 ...
## Iteration 230 ...
## Iteration 231 ...
## Iteration 232 ...
## Iteration 233 ...
## Iteration 234 ...
## Iteration 235 ...
## Iteration 236 ...
## Iteration 237 ...
```

```
## Iteration 238 ...
## Iteration 239 ...
## Iteration 240 ...
## Iteration 241 ...
## Iteration 242 ...
## Iteration 243 ...
## Iteration 244 ...
## Iteration 245 ...
## Iteration 246 ...
## Iteration 247 ...
## Iteration 248 ...
## Iteration 249 ...
## Iteration 250 ...
## Iteration 251 ...
## Iteration 252 ...
## Iteration 253 ...
## Iteration 254 ...
## Iteration 255 ...
## Iteration 256 ...
## Iteration 257 ...
## Iteration 258 ...
## Iteration 259 ...
## Iteration 260 ...
## Iteration 261 ...
## Iteration 262 ...
## Iteration 263 ...
## Iteration 264 ...
## Iteration 265 ...
## Iteration 266 ...
## Iteration 267 ...
## Iteration 268 ...
## Iteration 269 ...
## Iteration 270 ...
## Iteration 271 ...
## Iteration 272 ...
## Iteration 273 ...
## Iteration 274 ...
## Iteration 275 ...
## Iteration 276 ...
## Iteration 277 ...
## Iteration 278 ...
## Iteration 279 ...
## Iteration 280 ...
## Iteration 281 ...
## Iteration 282 ...
## Iteration 283 ...
## Iteration 284 ...
## Iteration 285 ...
## Iteration 286 ...
## Iteration 287 ...
## Iteration 288 ...
## Iteration 289 ...
## Iteration 290 ...
## Iteration 291 ...
```

```
## Iteration 292 ...
## Iteration 293 ...
## Iteration 294 ...
## Iteration 295 ...
## Iteration 296 ...
## Iteration 297 ...
## Iteration 298 ...
## Iteration 299 ...
## Iteration 300 ...
## Iteration 301 ...
## Iteration 302 ...
## Iteration 303 ...
## Iteration 304 ...
## Iteration 305 ...
## Iteration 306 ...
## Iteration 307 ...
## Iteration 308 ...
## Iteration 309 ...
## Iteration 310 ...
## Iteration 311 ...
## Iteration 312 ...
## Iteration 313 ...
## Iteration 314 ...
## Iteration 315 ...
## Iteration 316 ...
## Iteration 317 ...
## Iteration 318 ...
## Iteration 319 ...
## Iteration 320 ...
## Iteration 321 ...
## Iteration 322 ...
## Iteration 323 ...
## Iteration 324 ...
## Iteration 325 ...
## Iteration 326 ...
## Iteration 327 ...
## Iteration 328 ...
## Iteration 329 ...
## Iteration 330 ...
## Iteration 331 ...
## Iteration 332 ...
## Iteration 333 ...
## Iteration 334 ...
## Iteration 335 ...
## Iteration 336 ...
## Iteration 337 ...
## Iteration 338 ...
## Iteration 339 ...
## Iteration 340 ...
## Iteration 341 ...
## Iteration 342 ...
## Iteration 343 ...
## Iteration 344 ...
## Iteration 345 ...
```

```
## Iteration 346 ...
## Iteration 347 ...
## Iteration 348 ...
## Iteration 349 ...
## Iteration 350 ...
## Iteration 351 ...
## Iteration 352 ...
## Iteration 353 ...
## Iteration 354 ...
## Iteration 355 ...
## Iteration 356 ...
## Iteration 357 ...
## Iteration 358 ...
## Iteration 359 ...
## Iteration 360 ...
## Iteration 361 ...
## Iteration 362 ...
## Iteration 363 ...
## Iteration 364 ...
## Iteration 365 ...
## Iteration 366 ...
## Iteration 367 ...
## Iteration 368 ...
## Iteration 369 ...
## Iteration 370 ...
## Iteration 371 ...
## Iteration 372 ...
## Iteration 373 ...
## Iteration 374 ...
## Iteration 375 ...
## Iteration 376 ...
## Iteration 377 ...
## Iteration 378 ...
## Iteration 379 ...
## Iteration 380 ...
## Iteration 381 ...
## Iteration 382 ...
## Iteration 383 ...
## Iteration 384 ...
## Iteration 385 ...
## Iteration 386 ...
## Iteration 387 ...
## Iteration 388 ...
## Iteration 389 ...
## Iteration 390 ...
## Iteration 391 ...
## Iteration 392 ...
## Iteration 393 ...
## Iteration 394 ...
## Iteration 395 ...
## Iteration 396 ...
## Iteration 397 ...
## Iteration 398 ...
## Iteration 399 ...
```

```
## Iteration 400 ...
## Iteration 401 ...
## Iteration 402 ...
## Iteration 403 ...
## Iteration 404 ...
## Iteration 405 ...
## Iteration 406 ...
## Iteration 407 ...
## Iteration 408 ...
## Iteration 409 ...
## Iteration 410 ...
## Iteration 411 ...
## Iteration 412 ...
## Iteration 413 ...
## Iteration 414 ...
## Iteration 415 ...
## Iteration 416 ...
## Iteration 417 ...
## Iteration 418 ...
## Iteration 419 ...
## Iteration 420 ...
## Iteration 421 ...
## Iteration 422 ...
## Iteration 423 ...
## Iteration 424 ...
## Iteration 425 ...
## Iteration 426 ...
## Iteration 427 ...
## Iteration 428 ...
## Iteration 429 ...
## Iteration 430 ...
## Iteration 431 ...
## Iteration 432 ...
## Iteration 433 ...
## Iteration 434 ...
## Iteration 435 ...
## Iteration 436 ...
## Iteration 437 ...
## Iteration 438 ...
## Iteration 439 ...
## Iteration 440 ...
## Iteration 441 ...
## Iteration 442 ...
## Iteration 443 ...
## Iteration 444 ...
## Iteration 445 ...
## Iteration 446 ...
## Iteration 447 ...
## Iteration 448 ...
## Iteration 449 ...
## Iteration 450 ...
## Iteration 451 ...
## Iteration 452 ...
## Iteration 453 ...
```

```
## Iteration 454 ...
## Iteration 455 ...
## Iteration 456 ...
## Iteration 457 ...
## Iteration 458 ...
## Iteration 459 ...
## Iteration 460 ...
## Iteration 461 ...
## Iteration 462 ...
## Iteration 463 ...
## Iteration 464 ...
## Iteration 465 ...
## Iteration 466 ...
## Iteration 467 ...
## Iteration 468 ...
## Iteration 469 ...
## Iteration 470 ...
## Iteration 471 ...
## Iteration 472 ...
## Iteration 473 ...
## Iteration 474 ...
## Iteration 475 ...
## Iteration 476 ...
## Iteration 477 ...
## Iteration 478 ...
## Iteration 479 ...
## Iteration 480 ...
## Iteration 481 ...
## Iteration 482 ...
## Iteration 483 ...
## Iteration 484 ...
## Iteration 485 ...
## Iteration 486 ...
## Iteration 487 ...
## Iteration 488 ...
## Iteration 489 ...
## Iteration 490 ...
## Iteration 491 ...
## Iteration 492 ...
## Iteration 493 ...
## Iteration 494 ...
## Iteration 495 ...
## Iteration 496 ...
## Iteration 497 ...
## Iteration 498 ...
## Iteration 499 ...
## Iteration 500 ...
## Iteration 501 ...
## Iteration 502 ...
## Iteration 503 ...
## Iteration 504 ...
## Iteration 505 ...
## Iteration 506 ...
## Iteration 507 ...
```

```
## Iteration 508 ...
## Iteration 509 ...
## Iteration 510 ...
## Iteration 511 ...
## Iteration 512 ...
## Iteration 513 ...
## Iteration 514 ...
## Iteration 515 ...
## Iteration 516 ...
## Iteration 517 ...
## Iteration 518 ...
## Iteration 519 ...
## Iteration 520 ...
## Iteration 521 ...
## Iteration 522 ...
## Iteration 523 ...
## Iteration 524 ...
## Iteration 525 ...
## Iteration 526 ...
## Iteration 527 ...
## Iteration 528 ...
## Iteration 529 ...
## Iteration 530 ...
## Iteration 531 ...
## Iteration 532 ...
## Iteration 533 ...
## Iteration 534 ...
## Iteration 535 ...
## Iteration 536 ...
## Iteration 537 ...
## Iteration 538 ...
## Iteration 539 ...
## Iteration 540 ...
## Iteration 541 ...
## Iteration 542 ...
## Iteration 543 ...
## Iteration 544 ...
## Iteration 545 ...
## Iteration 546 ...
## Iteration 547 ...
## Iteration 548 ...
## Iteration 549 ...
## Iteration 550 ...
## Iteration 551 ...
## Iteration 552 ...
## Iteration 553 ...
## Iteration 554 ...
## Iteration 555 ...
## Iteration 556 ...
## Iteration 557 ...
## Iteration 558 ...
## Iteration 559 ...
## Iteration 560 ...
## Iteration 561 ...
```

```
## Iteration 562 ...
## Iteration 563 ...
## Iteration 564 ...
## Iteration 565 ...
## Iteration 566 ...
## Iteration 567 ...
## Iteration 568 ...
## Iteration 569 ...
## Iteration 570 ...
## Iteration 571 ...
## Iteration 572 ...
## Iteration 573 ...
## Iteration 574 ...
## Iteration 575 ...
## Iteration 576 ...
## Iteration 577 ...
## Iteration 578 ...
## Iteration 579 ...
## Iteration 580 ...
## Iteration 581 ...
## Iteration 582 ...
## Iteration 583 ...
## Iteration 584 ...
## Iteration 585 ...
## Iteration 586 ...
## Iteration 587 ...
## Iteration 588 ...
## Iteration 589 ...
## Iteration 590 ...
## Iteration 591 ...
## Iteration 592 ...
## Iteration 593 ...
## Iteration 594 ...
## Iteration 595 ...
## Iteration 596 ...
## Iteration 597 ...
## Iteration 598 ...
## Iteration 599 ...
## Iteration 600 ...
## Iteration 601 ...
## Iteration 602 ...
## Iteration 603 ...
## Iteration 604 ...
## Iteration 605 ...
## Iteration 606 ...
## Iteration 607 ...
## Iteration 608 ...
## Iteration 609 ...
## Iteration 610 ...
## Iteration 611 ...
## Iteration 612 ...
## Iteration 613 ...
## Iteration 614 ...
## Iteration 615 ...
```

```
## Iteration 616 ...
## Iteration 617 ...
## Iteration 618 ...
## Iteration 619 ...
## Iteration 620 ...
## Iteration 621 ...
## Iteration 622 ...
## Iteration 623 ...
## Iteration 624 ...
## Iteration 625 ...
## Iteration 626 ...
## Iteration 627 ...
## Iteration 628 ...
## Iteration 629 ...
## Iteration 630 ...
## Iteration 631 ...
## Iteration 632 ...
## Iteration 633 ...
## Iteration 634 ...
## Iteration 635 ...
## Iteration 636 ...
## Iteration 637 ...
## Iteration 638 ...
## Iteration 639 ...
## Iteration 640 ...
## Iteration 641 ...
## Iteration 642 ...
## Iteration 643 ...
## Iteration 644 ...
## Iteration 645 ...
## Iteration 646 ...
## Iteration 647 ...
## Iteration 648 ...
## Iteration 649 ...
## Iteration 650 ...
## Iteration 651 ...
## Iteration 652 ...
## Iteration 653 ...
## Iteration 654 ...
## Iteration 655 ...
## Iteration 656 ...
## Iteration 657 ...
## Iteration 658 ...
## Iteration 659 ...
## Iteration 660 ...
## Iteration 661 ...
## Iteration 662 ...
## Iteration 663 ...
## Iteration 664 ...
## Iteration 665 ...
## Iteration 666 ...
## Iteration 667 ...
## Iteration 668 ...
## Iteration 669 ...
```

```
## Iteration 670 ...
## Iteration 671 ...
## Iteration 672 ...
## Iteration 673 ...
## Iteration 674 ...
## Iteration 675 ...
## Iteration 676 ...
## Iteration 677 ...
## Iteration 678 ...
## Iteration 679 ...
## Iteration 680 ...
## Iteration 681 ...
## Iteration 682 ...
## Iteration 683 ...
## Iteration 684 ...
## Iteration 685 ...
## Iteration 686 ...
## Iteration 687 ...
## Iteration 688 ...
## Iteration 689 ...
## Iteration 690 ...
## Iteration 691 ...
## Iteration 692 ...
## Iteration 693 ...
## Iteration 694 ...
## Iteration 695 ...
## Iteration 696 ...
## Iteration 697 ...
## Iteration 698 ...
## Iteration 699 ...
## Iteration 700 ...
## Iteration 701 ...
## Iteration 702 ...
## Iteration 703 ...
## Iteration 704 ...
## Iteration 705 ...
## Iteration 706 ...
## Iteration 707 ...
## Iteration 708 ...
## Iteration 709 ...
## Iteration 710 ...
## Iteration 711 ...
## Iteration 712 ...
## Iteration 713 ...
## Iteration 714 ...
## Iteration 715 ...
## Iteration 716 ...
## Iteration 717 ...
## Iteration 718 ...
## Iteration 719 ...
## Iteration 720 ...
## Iteration 721 ...
## Iteration 722 ...
## Iteration 723 ...
```

```
## Iteration 724 ...
## Iteration 725 ...
## Iteration 726 ...
## Iteration 727 ...
## Iteration 728 ...
## Iteration 729 ...
## Iteration 730 ...
## Iteration 731 ...
## Iteration 732 ...
## Iteration 733 ...
## Iteration 734 ...
## Iteration 735 ...
## Iteration 736 ...
## Iteration 737 ...
## Iteration 738 ...
## Iteration 739 ...
## Iteration 740 ...
## Iteration 741 ...
## Iteration 742 ...
## Iteration 743 ...
## Iteration 744 ...
## Iteration 745 ...
## Iteration 746 ...
## Iteration 747 ...
## Iteration 748 ...
## Iteration 749 ...
## Iteration 750 ...
## Iteration 751 ...
## Iteration 752 ...
## Iteration 753 ...
## Iteration 754 ...
## Iteration 755 ...
## Iteration 756 ...
## Iteration 757 ...
## Iteration 758 ...
## Iteration 759 ...
## Iteration 760 ...
## Iteration 761 ...
## Iteration 762 ...
## Iteration 763 ...
## Iteration 764 ...
## Iteration 765 ...
## Iteration 766 ...
## Iteration 767 ...
## Iteration 768 ...
## Iteration 769 ...
## Iteration 770 ...
## Iteration 771 ...
## Iteration 772 ...
## Iteration 773 ...
## Iteration 774 ...
## Iteration 775 ...
## Iteration 776 ...
## Iteration 777 ...
```

```
## Iteration 778 ...
## Iteration 779 ...
## Iteration 780 ...
## Iteration 781 ...
## Iteration 782 ...
## Iteration 783 ...
## Iteration 784 ...
## Iteration 785 ...
## Iteration 786 ...
## Iteration 787 ...
## Iteration 788 ...
## Iteration 789 ...
## Iteration 790 ...
## Iteration 791 ...
## Iteration 792 ...
## Iteration 793 ...
## Iteration 794 ...
## Iteration 795 ...
## Iteration 796 ...
## Iteration 797 ...
## Iteration 798 ...
## Iteration 799 ...
## Iteration 800 ...
## Iteration 801 ...
## Iteration 802 ...
## Iteration 803 ...
## Iteration 804 ...
## Iteration 805 ...
## Iteration 806 ...
## Iteration 807 ...
## Iteration 808 ...
## Iteration 809 ...
## Iteration 810 ...
## Iteration 811 ...
## Iteration 812 ...
## Iteration 813 ...
## Iteration 814 ...
## Iteration 815 ...
## Iteration 816 ...
## Iteration 817 ...
## Iteration 818 ...
## Iteration 819 ...
## Iteration 820 ...
## Iteration 821 ...
## Iteration 822 ...
## Iteration 823 ...
## Iteration 824 ...
## Iteration 825 ...
## Iteration 826 ...
## Iteration 827 ...
## Iteration 828 ...
## Iteration 829 ...
## Iteration 830 ...
## Iteration 831 ...
```

```
## Iteration 832 ...
## Iteration 833 ...
## Iteration 834 ...
## Iteration 835 ...
## Iteration 836 ...
## Iteration 837 ...
## Iteration 838 ...
## Iteration 839 ...
## Iteration 840 ...
## Iteration 841 ...
## Iteration 842 ...
## Iteration 843 ...
## Iteration 844 ...
## Iteration 845 ...
## Iteration 846 ...
## Iteration 847 ...
## Iteration 848 ...
## Iteration 849 ...
## Iteration 850 ...
## Iteration 851 ...
## Iteration 852 ...
## Iteration 853 ...
## Iteration 854 ...
## Iteration 855 ...
## Iteration 856 ...
## Iteration 857 ...
## Iteration 858 ...
## Iteration 859 ...
## Iteration 860 ...
## Iteration 861 ...
## Iteration 862 ...
## Iteration 863 ...
## Iteration 864 ...
## Iteration 865 ...
## Iteration 866 ...
## Iteration 867 ...
## Iteration 868 ...
## Iteration 869 ...
## Iteration 870 ...
## Iteration 871 ...
## Iteration 872 ...
## Iteration 873 ...
## Iteration 874 ...
## Iteration 875 ...
## Iteration 876 ...
## Iteration 877 ...
## Iteration 878 ...
## Iteration 879 ...
## Iteration 880 ...
## Iteration 881 ...
## Iteration 882 ...
## Iteration 883 ...
## Iteration 884 ...
## Iteration 885 ...
```

```
## Iteration 886 ...
## Iteration 887 ...
## Iteration 888 ...
## Iteration 889 ...
## Iteration 890 ...
## Iteration 891 ...
## Iteration 892 ...
## Iteration 893 ...
## Iteration 894 ...
## Iteration 895 ...
## Iteration 896 ...
## Iteration 897 ...
## Iteration 898 ...
## Iteration 899 ...
## Iteration 900 ...
## Iteration 901 ...
## Iteration 902 ...
## Iteration 903 ...
## Iteration 904 ...
## Iteration 905 ...
## Iteration 906 ...
## Iteration 907 ...
## Iteration 908 ...
## Iteration 909 ...
## Iteration 910 ...
## Iteration 911 ...
## Iteration 912 ...
## Iteration 913 ...
## Iteration 914 ...
## Iteration 915 ...
## Iteration 916 ...
## Iteration 917 ...
## Iteration 918 ...
## Iteration 919 ...
## Iteration 920 ...
## Iteration 921 ...
## Iteration 922 ...
## Iteration 923 ...
## Iteration 924 ...
## Iteration 925 ...
## Iteration 926 ...
## Iteration 927 ...
## Iteration 928 ...
## Iteration 929 ...
## Iteration 930 ...
## Iteration 931 ...
## Iteration 932 ...
## Iteration 933 ...
## Iteration 934 ...
## Iteration 935 ...
## Iteration 936 ...
## Iteration 937 ...
## Iteration 938 ...
## Iteration 939 ...
```

```
## Iteration 940 ...
## Iteration 941 ...
## Iteration 942 ...
## Iteration 943 ...
## Iteration 944 ...
## Iteration 945 ...
## Iteration 946 ...
## Iteration 947 ...
## Iteration 948 ...
## Iteration 949 ...
## Iteration 950 ...
## Iteration 951 ...
## Iteration 952 ...
## Iteration 953 ...
## Iteration 954 ...
## Iteration 955 ...
## Iteration 956 ...
## Iteration 957 ...
## Iteration 958 ...
## Iteration 959 ...
## Iteration 960 ...
## Iteration 961 ...
## Iteration 962 ...
## Iteration 963 ...
## Iteration 964 ...
## Iteration 965 ...
## Iteration 966 ...
## Iteration 967 ...
## Iteration 968 ...
## Iteration 969 ...
## Iteration 970 ...
## Iteration 971 ...
## Iteration 972 ...
## Iteration 973 ...
## Iteration 974 ...
## Iteration 975 ...
## Iteration 976 ...
## Iteration 977 ...
## Iteration 978 ...
## Iteration 979 ...
## Iteration 980 ...
## Iteration 981 ...
## Iteration 982 ...
## Iteration 983 ...
## Iteration 984 ...
## Iteration 985 ...
## Iteration 986 ...
## Iteration 987 ...
## Iteration 988 ...
## Iteration 989 ...
## Iteration 990 ...
## Iteration 991 ...
## Iteration 992 ...
## Iteration 993 ...
```

```
## Iteration 994 ...
## Iteration 995 ...
## Iteration 996 ...
## Iteration 997 ...
## Iteration 998 ...
## Iteration 999 ...
## Iteration 1000 ...
## Gibbs sampling completed!
```

```
saveRDS(lda_model, "lda_model.rds")
```

Code Block 8

For the sake of replicating the original study, I used the matrix= argument to calculate my beta and gamma scores. I don't think the gamma was actually needed for my specific analysis, but I did it anyways to have a more complete picture of my data. Especially since it's only one line of code. The beta is more important, it not only tells me the probability of each word appearing in each topic, but its needed to create my word cluster graph later on.

```
#find beta
beta_topics <- tidy(lda_model, matrix = "beta")

#fina gamma
gamma_topics <- tidy(lda_model, matrix = "gamma")
```

Code Block 9

This code block is a little more dense, but basically I am processing my beta score so that I can replicate Table 5 'Keywords of Topic 8 LDA' in the paper. The first chunk is helping me build the keywords section of the table, by grouping by topic and ensuring each topic has 20 of the top words with their individual beta score. The third code chunk 'term_totals' helps build the 'total_score' of my table by calling for all individual keyword beta scores to be added into a total. Finally, the last code chunk actually builds my table by bringing together all the code above.

```
#graph and table for the LDA model with top terms

top_terms <- beta_topics %>%
  group_by(topic) %>%
  slice_max(beta, n = 20) %>%
  ungroup() %>%
  arrange(topic, -beta)

print(top_terms)
```

```
## # A tibble: 1,194 x 3
##    topic term         beta
##    <int> <chr>       <dbl>
##  1     1 ceasefire 0.0976
##  2     1 northern  0.0215
##  3     1 palestine 0.0145
##  4     1 report    0.0145
```

```
##  5     1 hezbollah 0.0145
##  6     1 releases  0.0145
##  7     1 soldiers  0.00762
##  8     1 detained  0.00762
##  9     1 reject    0.00762
## 10     1 uae       0.00762
## # i 1,184 more rows
```

```r
term_totals <- top_terms %>%
  group_by(topic) %>%
  summarize(total_score = sum(beta))

final_table <- top_terms %>%
    mutate(term_with_score = paste0(term, " (", round(beta, 2), ")")) %>%
    group_by(topic) %>%
    summarize(Keywords = paste(term_with_score, collapse = ", ")) %>%
    left_join(term_totals, top_terms, by = "topic")
```

Code Block 10

This next code chunk is intended to reproduce 'Table 4 Coherence Score & Perplexity LDA' from the original study. At the top, I am setting my k values (like from a few code chunks prior) to start at topic 2 and end at topic 31. I started with the more obvious 1-30, but for some reason I kept getting an error that I had to start at 2 and not one. I still had 30 topics, so I compromised by shifting my originally intended 1-30 to 2:31. The next code chunk established my perplexity values, using sapply to go through each of my k values, calling my dtm, once again settling my methods and control. Next, I created a table for my perplexity scores, calculated the coherence scores and made another table to store said scores.

```r
#calculate perplexity score

ks <- 2:31

perplexity_values <- sapply(ks, function(k_val) {
  temp_model <- LDA(dtm, k = k_val,
                    method = "Gibbs",
                    control = list(alpha = 0.01, seed = 42))
  perp <- perplexity(temp_model, newdata = dtm)
  return(perp)
})

#turn into dataframe
perplexity_table <- data.frame(
  Model = "LDA",
  Topics_K = ks,
  Perplexity = perplexity_values
)

#calculate coherence score

lda_model_20 <- LDA(dtm, k = 30, control = list(seed = 1234))

coherence_scores <- topic_coherence(lda_model_20, dtm)
```

```r
#turn into dataframe

coherence_table <- data.frame(
  Model = "LDA",
  Topics_K = 1:30,
  Coherence = coherence_scores
)
```

Code Block 11

Here, I am simply combining the coherence and perplexity tables I made above into one for the purpose of replicating table 4 in the original study. Having both scores in one place allows me to compare all topics at once and see which topics are most valuable.

```r
#combine coherence and perplexity into one table
perp_coh_table <- coherence_table %>%
  left_join(perplexity_table, by = "Topics_K")

perp_coh_table <- perp_coh_table %>%
  select(Topics_K, Perplexity, Coherence) %>%
  rename(Topic = Topics_K)

print(perp_coh_table)
```

```
##    Topic Perplexity  Coherence
## 1      1         NA  -52.48431
## 2      2  232.09359  -71.68509
## 3      3  181.53671  -67.12749
## 4      4  152.36125  -97.77873
## 5      5  137.65711  -80.55551
## 6      6  123.06155  -44.38329
## 7      7  117.03862  -96.34579
## 8      8  106.49841 -100.38254
## 9      9  105.42530  -65.57667
## 10    10   98.91475  -78.10171
## 11    11   96.79236  -80.94532
## 12    12   94.42553 -104.74296
## 13    13   90.85541  -94.11562
## 14    14   89.56817  -74.66960
## 15    15   89.07621  -84.30794
## 16    16   87.72471  -74.88750
## 17    17   89.34691  -92.43622
## 18    18   86.40028  -57.57237
## 19    19   87.15909  -85.53588
## 20    20   87.37266  -80.77336
## 21    21   85.44829  -45.89147
## 22    22   86.97709 -102.92457
## 23    23   84.71764  -65.13075
## 24    24   87.95541 -100.89340
## 25    25   86.20843  -90.21004
## 26    26   88.11853 -103.10407
## 27    27   87.52034  -75.74971
```

```
## 28    28   89.36323  -62.53943
## 29    29   88.87845  -87.81485
## 30    30   89.48956  -98.11735
```

Code Block 12

For my final code block I made a simple word cloud graph similar to the study's based on the beta scores.

```
#Create a word cloud graph

cloud_data <- beta_topics %>%
  filter(topic == 8) %>%
  mutate(freq = beta * 1000) %>%
  select(term, freq)

wordcloud2(cloud_data)
```