Topic 7: Word Embeddings

Peter Menzies

This week's Rmd file here: https://github.com/MaRo406/EDS_231-text-sentiment/blob/main/topic_7.Rmd

Today we are using climbing incident data from this repo: https://github.com/ecaroom/climbing-accidents. Some analysis (in Excel) on the data was written up into a Rock and Ice magazine article.

But I've constructed our data set (link below) by pulling a few key variables including the full text of each incident report.

```
incidents_df <- read_csv("https://raw.githubusercontent.com/MaRo406/EDS_231-text-sentiment/825b159b6da4
```

```
## Rows: 2770 Columns: 4

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (3): ID, Accident Title, Text
## dbl (1): Publication Year

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

First, let's calculate the unigram probabilities, how often we see each word in this corpus.

```
unigram_probs <- incidents_df %>%
    unnest_tokens(word, Text) %>%
    anti_join(stop_words, by = 'word') %>%
    count(word, sort = TRUE) %>%
    mutate(p = n / sum(n))
unigram_probs
```

```
## # A tibble: 25,205 x 3
##    word          n       p
##    <chr>     <int>   <dbl>
##  1 rope       5129 0.00922
##  2 feet       5101 0.00917
##  3 climbing   4755 0.00855
##  4 route      4357 0.00783
##  5 climbers   3611 0.00649
##  6 climb      3209 0.00577
##  7 fall       3168 0.00569
##  8 climber    2964 0.00533
##  9 rescue     2928 0.00526
## 10 source     2867 0.00515
## # ... with 25,195 more rows
```

Next, we need to know how often we find each word near each other word – the skipgram probabilities. This is where we use the sliding window.

```
skipgrams <- incidents_df %>%
    unnest_tokens(ngram, Text, token = "ngrams", n = 5) %>%
    mutate(ngramID = row_number()) %>%
    tidyr::unite(skipgramID, ID, ngramID) %>%
    unnest_tokens(word, ngram) %>%
    anti_join(stop_words, by = 'word')

skipgrams
```

```
## # A tibble: 2,737,146 x 4
##    skipgramID `Accident Title`                              `Publication Yea~ word
##    <chr>      <chr>                                                     <dbl> <chr>
##  1 1_1        Failure of Rappel Setup (Protection Pull~                  1990 color~
##  2 1_1        Failure of Rappel Setup (Protection Pull~                  1990 rocky
##  3 1_1        Failure of Rappel Setup (Protection Pull~                  1990 mount~
##  4 1_1        Failure of Rappel Setup (Protection Pull~                  1990 natio~
##  5 1_1        Failure of Rappel Setup (Protection Pull~                  1990 park
##  6 1_2        Failure of Rappel Setup (Protection Pull~                  1990 rocky
##  7 1_2        Failure of Rappel Setup (Protection Pull~                  1990 mount~
##  8 1_2        Failure of Rappel Setup (Protection Pull~                  1990 natio~
##  9 1_2        Failure of Rappel Setup (Protection Pull~                  1990 park
## 10 1_3        Failure of Rappel Setup (Protection Pull~                  1990 mount~
## # ... with 2,737,136 more rows
```

```
#calculate probabilities
skipgram_probs <- skipgrams %>%
    pairwise_count(word, skipgramID, diag = TRUE, sort = TRUE) %>%
    mutate(p = n / sum(n))
```

```
## Warning: `distinct_()` was deprecated in dplyr 0.7.0.
## Please use `distinct()` instead.
## See vignette('programming') for more help
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

Having all the skipgram windows lets us calculate how often words together occur within a window, relative to their total occurrences in the data. We do this using the point-wise mutual information (PMI). It's the logarithm of the probability of finding two words together, normalized for the probability of finding each of the words alone. PMI tells us which words occur together more often than expected based on how often they occurred on their own.

```
#normalize probabilities
normalized_prob <- skipgram_probs %>%
    filter(n > 20) %>%
    rename(word1 = item1, word2 = item2) %>%
    left_join(unigram_probs %>%
                  select(word1 = word, p1 = p),
              by = "word1") %>%
    left_join(unigram_probs %>%
```

```
                  select(word2 = word, p2 = p),
              by = "word2") %>%
    mutate(p_together = p / p1 / p2)

#Which words are most associated with "rope"?
normalized_prob %>%
    filter(word1 == "rope") %>%
    arrange(-p_together)
```

```
## # A tibble: 295 x 7
##    word1 word2        n          p      p1        p2 p_together
##    <chr> <chr>    <dbl>      <dbl>   <dbl>     <dbl>      <dbl>
##  1 rope  rope     25494 0.00340    0.00922 0.00922         40.0
##  2 rope  lengths    101 0.0000135  0.00922 0.0000575       25.4
##  3 rope  skinny      24 0.00000320 0.00922 0.0000144       24.2
##  4 rope  drag       211 0.0000281  0.00922 0.000138        22.1
##  5 rope  taut        98 0.0000131  0.00922 0.0000701       20.2
##  6 rope  coiled      60 0.00000800 0.00922 0.0000431       20.1
##  7 rope  thicker     21 0.00000280 0.00922 0.0000162       18.8
##  8 rope  trailing    68 0.00000907 0.00922 0.0000539       18.3
##  9 rope  fed         48 0.00000640 0.00922 0.0000413       16.8
## 10 rope  70m         31 0.00000414 0.00922 0.0000270       16.6
## # ... with 285 more rows
```

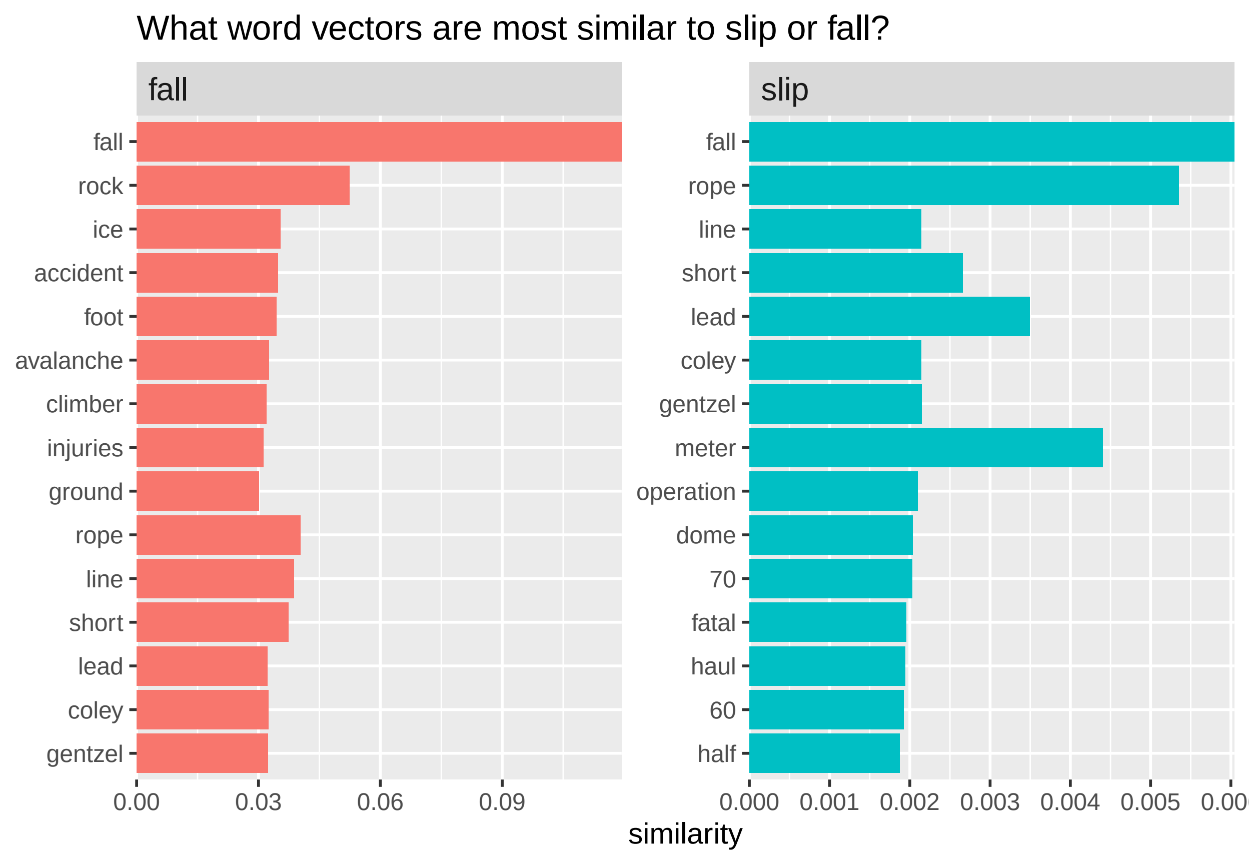
Now we convert to a matrix so we can use matrix factorization and reduce the dimensionality of the data.

```
pmi_matrix <- normalized_prob %>%
    mutate(pmi = log10(p_together)) %>%
    cast_sparse(word1, word2, pmi)

#remove missing data
pmi_matrix@x[is.na(pmi_matrix@x)] <- 0
#run SVD using irlba() which is good for sparse matrices
pmi_svd <- irlba(pmi_matrix, 100, maxit = 500) #Reducing to 100 dimensions
#next we output the word vectors:
word_vectors <- pmi_svd$u
rownames(word_vectors) <- rownames(pmi_matrix)
```

```
search_synonyms <- function(word_vectors, selected_vector) {
dat <- word_vectors %*% selected_vector

similarities <- dat %>%
        tibble(token = rownames(dat), similarity = dat[,1])

similarities %>%
       arrange(-similarity) %>%
        select(c(2,3))
}
```

```
fall <- search_synonyms(word_vectors,word_vectors["fall",])
slip <- search_synonyms(word_vectors,word_vectors["slip",])
```

```r
slip %>%
    mutate(selected = "slip") %>%
    bind_rows(fall %>%
                  mutate(selected = "fall")) %>%
    group_by(selected) %>%
    top_n(15, similarity) %>%
    ungroup %>%
    mutate(token = reorder(token, similarity)) %>%
    ggplot(aes(token, similarity, fill = selected)) +
    geom_col(show.legend = FALSE) +
    facet_wrap(~selected, scales = "free") +
    coord_flip() +
    theme(strip.text=element_text(hjust=0, size=12)) +
    scale_y_continuous(expand = c(0,0)) +
    labs(x = NULL, title = "What word vectors are most similar to slip or fall?")
```




```r
snow_danger <- word_vectors["snow",] + word_vectors["danger",]
search_synonyms(word_vectors, snow_danger)
```

```
## # A tibble: 9,104 x 2
##    token      similarity
##    <chr>           <dbl>
##  1 snow           0.396
##  2 avalanche      0.131
##  3 conditions     0.0918
```

```
##   4 soft           0.0806
##   5 wet            0.0783
##   6 ice            0.0769
##   7 icy            0.0735
##   8 slope          0.0703
##   9 fresh          0.0604
##  10 blindness      0.0596
## # ... with 9,094 more rows
```

```
no_snow_danger <- word_vectors["danger",] - word_vectors["snow",]
search_synonyms(word_vectors, no_snow_danger)
```

```
## # A tibble: 9,104 x 2
##    token     similarity
##    <chr>          <dbl>
##  1 avalanche     0.0882
##  2 danger        0.0547
##  3 rockfall      0.0540
##  4 gulch         0.0534
##  5 class         0.0507
##  6 hazard        0.0403
##  7 hazards       0.0394
##  8 occurred      0.0376
##  9 potential     0.0373
## 10 mph           0.0361
## # ... with 9,094 more rows
```

Assignment

Download a set of pretrained vectors, GloVe, and explore them.

Grab data here:

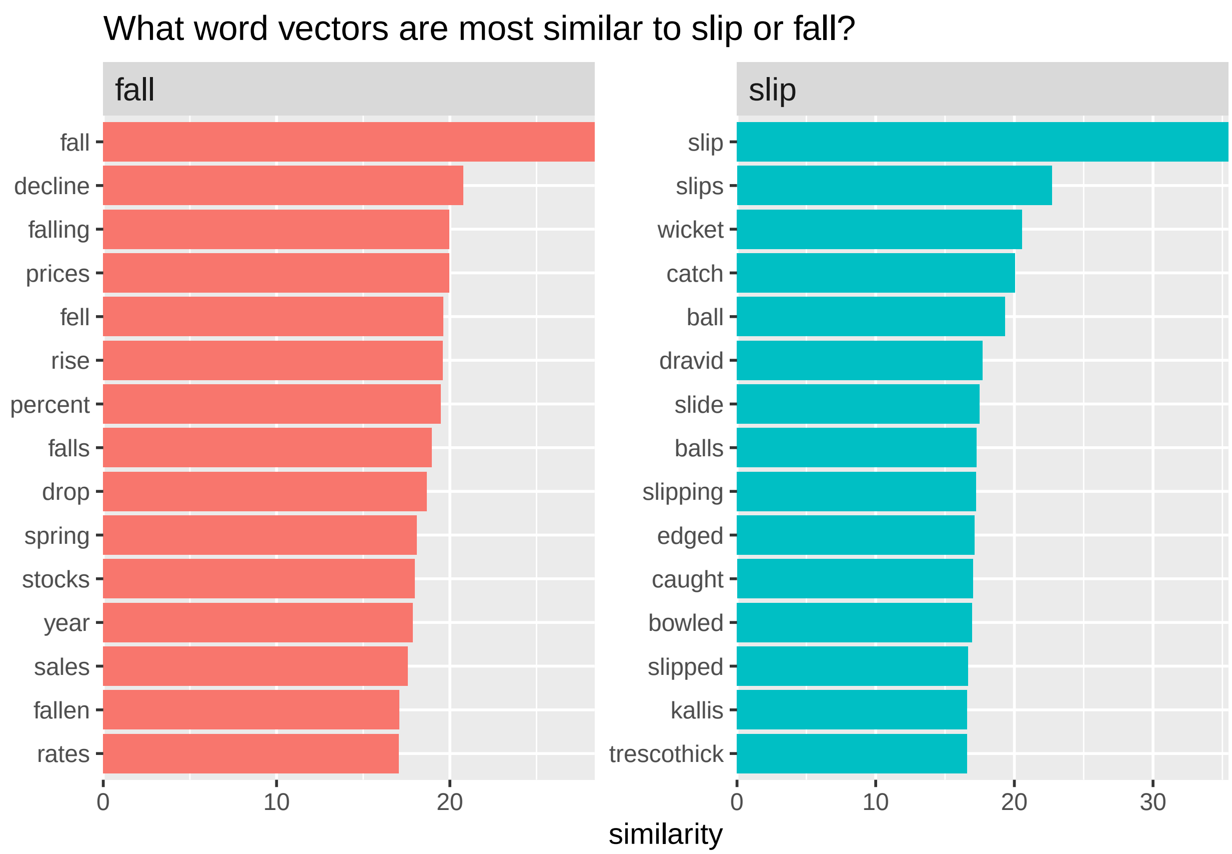
```
df <- data.table::fread(here("data/glove.6B/glove.6B.300d.txt"),
                        header = FALSE) %>%
  column_to_rownames("V1")

matrix <- as.matrix(df)
```

1. Recreate the analyses in the last three chunks (find-synonyms, plot-synonyms, word-math) with the GloVe embeddings. How are they different from the embeddings created from the climbing accident data? Why do you think they are different?

```
fall <- search_synonyms(matrix, matrix["fall",])
slip <- search_synonyms(matrix, matrix["slip",])
```

```
slip %>%
    mutate(selected = "slip") %>%
    bind_rows(fall %>%
              mutate(selected = "fall")) %>%
    group_by(selected) %>%
    top_n(15, similarity) %>%
```

```
    ungroup %>%
    mutate(token = reorder(token, similarity)) %>%
    ggplot(aes(token, similarity, fill = selected)) +
    geom_col(show.legend = FALSE) +
    facet_wrap(~selected, scales = "free") +
    coord_flip() +
    theme(strip.text=element_text(hjust=0, size=12)) +
    scale_y_continuous(expand = c(0,0)) +
    labs(x = NULL, title = "What word vectors are most similar to slip or fall?")
```




```
snow_danger <- matrix["snow",] + matrix["danger",]
search_synonyms(matrix, snow_danger)
```

```
## # A tibble: 400,000 x 2
##    token      similarity
##    <chr>           <dbl>
##  1 snow             57.6
##  2 rain             40.6
##  3 danger           40.5
##  4 snowfall         34.8
##  5 weather          34.4
##  6 winds            34.0
##  7 rains            34.0
##  8 fog              33.6
##  9 landslides       33.3
```

```
## 10 threat           33.0
## # ... with 399,990 more rows
```

```r
no_snow_danger <- matrix["danger",] - matrix["snow",]
search_synonyms(matrix, no_snow_danger)
```

```
## # A tibble: 400,000 x 2
##    token         similarity
##    <chr>              <dbl>
##  1 danger              23.3
##  2 risks               20.2
##  3 imminent            18.7
##  4 dangers             17.9
##  5 risk                17.8
##  6 32-team             17.6
##  7 mesdaq              17.5
##  8 inflationary        17.4
##  9 risking             17.2
## 10 2001-2011           17.0
## # ... with 399,990 more rows
```

These embedings are quite different because the climbing incident reports are focused on a particular topic whereas the GloVe embeddings are general. I wasn't familiar with the cricket term "slip" (or anything to do with the sport for that matter), and was surprised to see so many cricket-related words in the "slip" plot.

2. Run the classic word math equation, "king" - "man" = ?

```r
king <- matrix["king",] - matrix["man",]
search_synonyms(matrix, king)
```

```
## # A tibble: 400,000 x 2
##    token        similarity
##    <chr>             <dbl>
##  1 king               35.3
##  2 kalākaua           26.8
##  3 adulyadej          26.3
##  4 bhumibol           25.9
##  5 ehrenkrantz        25.5
##  6 gyanendra          25.2
##  7 birendra           25.2
##  8 sigismund          25.1
##  9 letsie             24.7
## 10 mswati             24.0
## # ... with 399,990 more rows
```

3. Think of three new word math equations. They can involve any words you'd like, whatever catches your interest.

```r
train <- matrix["train",] - matrix["transport",]
search_synonyms(matrix, train)
```

```
## # A tibble: 400,000 x 2
##    token      similarity
##    <chr>           <dbl>
##  1 train            26.7
##  2 derails          20.9
##  3 godhra           20.5
##  4 derailment       19.8
##  5 atocha           17.9
##  6 talgo            17.9
##  7 dorasan          17.5
##  8 trains           17.5
##  9 shinkansen       16.8
## 10 7:05             16.5
## # ... with 399,990 more rows
```

```
galactic_collision <- matrix["galactic",] + matrix["collision",]
search_synonyms(matrix, galactic_collision)
```

```
## # A tibble: 400,000 x 2
##    token         similarity
##    <chr>              <dbl>
##  1 galactic            56.6
##  2 collision           46.7
##  3 collisions          38.6
##  4 cosmic              35.3
##  5 colliding           35.3
##  6 galaxies            35.2
##  7 planets             34.7
##  8 gravitational       34.0
##  9 collided            33.8
## 10 orbiting            33.0
## # ... with 399,990 more rows
```

```
fusion <- matrix["nuclear",] + matrix["fusion",]
search_synonyms(matrix, fusion)
```

```
## # A tibble: 400,000 x 2
##    token        similarity
##    <chr>             <dbl>
##  1 nuclear            95.7
##  2 atomic             68.5
##  3 reactor            66.8
##  4 reactors           65.3
##  5 fusion             63.3
##  6 plutonium          61.0
##  7 yongbyon           59.3
##  8 uranium            58.2
##  9 enrichment         54.0
## 10 reprocessing       52.8
## # ... with 399,990 more rows
```