GloVe Demo

Demo

```
library(text2vec)
text8_file = "./text8"
if (!file.exists(text8_file)) {
  download.file("http://mattmahoney.net/dc/text8.zip", "./text8.zip")
  unzip ("./text8.zip", files = "text8", exdir = "./")
}
# Read in wiki data
wiki = readLines(text8_file, n = 1, warn = FALSE)
```

Read and tokenize 100MB of wiki data

```
# Create iterator over tokens
tokens <- space_tokenizer(wiki)
# Create vocabulary. Terms will be unigrams (simple words).
it = itoken(tokens, progressbar = FALSE)
vocab <- create_vocabulary(it)
vocab <- prune_vocabulary(vocab, term_count_min = 5L)
# Use our filtered vocabulary
vectorizer <- vocab_vectorizer(vocab)
# use window of 5 for context words
# TCM is "Term Co-occurrence Matrix"
tcm <- create_tcm(it, vectorizer, skip_grams_window = 15L)
```

Look at some examples within the tcm

```
exampleWords = c("pig","cow","fish","animal","apple","pear","tomato","fruit")
m <- as.matrix(tcm[exampleWords,exampleWords])
round((m + t(m)) - diag(diag(m)),digits=3)
```

```
##          pig    cow   fish animal  apple  pear tomato  fruit
## pig    1.300  0.593  0.083  2.317  0.167 0.000  0.000  0.000
## cow    0.593 10.260  0.125  0.492  0.083 0.000  0.000  0.000
## fish   0.083  0.125 59.376  5.288  0.341 0.000  0.424  2.703
## animal 2.317  0.492  5.288 42.975  0.167 0.000  0.000  1.613
## apple  0.167  0.083  0.341  0.167 78.835 5.450  0.071  2.853
## pear   0.000  0.000  0.000  0.000  5.450 0.610  0.000  0.583
## tomato 0.000  0.000  0.424  0.000  0.071 0.000  0.202  0.625
## fruit  0.000  0.000  2.703  1.613  2.853 0.583  0.625 16.526
```

Learn GLovE model

```
glove = GlobalVectors$new(rank = 50, x_max = 10)
system.time(word_vectors <- glove$fit_transform(tcm, n_iter = 30))
```

```
## INFO  [20:45:48.569] epoch 1, loss 0.1341
## INFO  [20:45:54.858] epoch 2, loss 0.0977
## INFO  [20:46:01.301] epoch 3, loss 0.0884
## INFO  [20:46:07.642] epoch 4, loss 0.0834
## INFO  [20:46:14.100] epoch 5, loss 0.0802
## INFO  [20:46:20.576] epoch 6, loss 0.0780
## INFO  [20:46:27.065] epoch 7, loss 0.0763
## INFO  [20:46:33.527] epoch 8, loss 0.0750
## INFO  [20:46:40.565] epoch 9, loss 0.0739
## INFO  [20:46:47.368] epoch 10, loss 0.0731
## INFO  [20:46:54.564] epoch 11, loss 0.0723
## INFO  [20:47:01.256] epoch 12, loss 0.0717
## INFO  [20:47:07.905] epoch 13, loss 0.0712
## INFO  [20:47:14.349] epoch 14, loss 0.0707
## INFO  [20:47:20.943] epoch 15, loss 0.0703
## INFO  [20:47:27.401] epoch 16, loss 0.0699
## INFO  [20:47:33.932] epoch 17, loss 0.0696
## INFO  [20:47:40.585] epoch 18, loss 0.0693
## INFO  [20:47:47.195] epoch 19, loss 0.0690
## INFO  [20:47:53.639] epoch 20, loss 0.0688
## INFO  [20:48:00.108] epoch 21, loss 0.0685
## INFO  [20:48:06.727] epoch 22, loss 0.0683
## INFO  [20:48:13.207] epoch 23, loss 0.0681
## INFO  [20:48:19.785] epoch 24, loss 0.0679
## INFO  [20:48:26.487] epoch 25, loss 0.0678
## INFO  [20:48:33.000] epoch 26, loss 0.0676
## INFO  [20:48:39.534] epoch 27, loss 0.0675
## INFO  [20:48:46.092] epoch 28, loss 0.0673
## INFO  [20:48:52.589] epoch 29, loss 0.0672
## INFO  [20:48:59.190] epoch 30, loss 0.0671
```

```
##     user   system  elapsed
## 1323.243   34.686  197.102
```

Try some vector operations, look at closest results

```r
# Function to find and print most similar vectors
print_sims <- function(wordvec) {
  cos_sim = sim2(x = word_vectors, y = wordvec, method = "cosine", norm = "l2")
head(sort(cos_sim[,1], decreasing = TRUE), 5)
}

# List of word vectors to test
paris <- word_vectors["paris", , drop = FALSE]
rome <- word_vectors["rome", , drop = FALSE]
madrid <- word_vectors["madrid", , drop = FALSE]
france <- word_vectors["france", , drop = FALSE]
germany <- word_vectors["germany", , drop = FALSE]
spain <- word_vectors["spain", , drop = FALSE]
italy <- word_vectors["italy", , drop = FALSE]
canada <- word_vectors["canada", , drop = FALSE]
europe <- word_vectors["europe", , drop = FALSE]
asia <- word_vectors["asia", , drop = FALSE]
africa <- word_vectors["africa", , drop = FALSE]
```

```r
print_sims(paris)
```

```
##     paris    france    london    venice    munich 
## 1.0000000 0.6967841 0.6875268 0.6780207 0.6721228
```

```r
print_sims(france)
```

```
##      france       spain     britain       italy netherlands 
##   1.0000000   0.8467072   0.7965850   0.7894758   0.7696812
```

```r
print_sims(germany)
```

```
##   germany     italy   hungary    russia   austria 
## 1.0000000 0.8516388 0.8011105 0.7917224 0.7909706
```

```r
print_sims(canada)
```

```
##    canada australia    united   america    quebec 
## 1.0000000 0.8593498 0.7871118 0.7588684 0.7571047
```

```r
print_sims(europe)
```

```
##    europe      asia   western   america   britain 
## 1.0000000 0.8832478 0.8540591 0.8309380 0.8254089
```

```r
print_sims(asia)
```

```
##      asia    europe southeast    africa   western 
## 1.0000000 0.8832478 0.8190913 0.8115017 0.7913134
```

```r
print_sims(africa)
```

```
##    africa     south      asia     north    europe 
## 1.0000000 0.8512636 0.8115017 0.7999412 0.7822796
```

Analogy test 1

```r
test <- paris - france + germany
print_sims(test)
```

```
##    berlin     paris   germany    munich   leipzig 
## 0.8445823 0.7514069 0.7376813 0.7222886 0.7032850
```

Analogy test 2

```r
test <- paris - france + canada
print_sims(test)
```

```
##    sydney      york    canada    london       usa
## 0.7123708 0.6774785 0.6748612 0.6748078 0.6733471
```

Analogy test 3

```r
test <- paris - europe + asia
print_sims(test)
```

```
##     paris       des     kabul        et        du
## 0.8774443 0.6317515 0.6208449 0.6153840 0.6067733
```