```
[1] "C"
```

The README dataset

There are 9706 readme's in our github dataset after getting rid of the empty ones.

For the topicmodels package its a 2-step process, Corpus(VectorSource(readme_df)) and DocumentTermMatrix

```
# corpus <- Corpus(VectorSource(readme_df$readme))


# dtm2 <- DocumentTermMatrix(corpus, control = list(stemming = TRUE,
# stopwords = TRUE, removeNumbers = TRUE, removePunctuation = TRUE))

# I had problems using tm's DocumentTermMatrix, but RTextTools worked ok
dtm = create_matrix(readme_df$readme, removeNumbers = TRUE, maxWordLength = 20,
    removeStopwords = TRUE, removePunctuation = TRUE, weighting = weightTfIdf)
```

This is a really messy term set. Here is a sample:

> apireadme, construire, sammy, paccessing, diasmartins, vdrs, hereemacsbranches, tunnelling, channavajhala, onvaluetap, pplaybr, robby, pathtorjs, challenging, servicesendreport, merz, vimtutor, ndp, emulates, bsee, containerrenderhello, value, decorators-decorators, backendwritetimeout, buildmarker, resnextresult, bkrservertemplates, httpwwwclictvtv, admob, dyldbootstrap-start, maskwithimageuiimage, addonspvrdemoaddon, connecter, httpsgithubcomnex, guidesummary, forceupdate, configjstemplate, glatzel, indexasynchtml, thw, classifiying, hpage, bthere, codemakeffaccessible, shiftcmdr, ipiht, bugge, withthe, contextsensitive, phillip, radtastical, fades, opertions, startedat, rhhmmss, connectionqueryuse, skyrimui, readmepdf, activedirectory, worldstep, lineargradientffff, gguardfile, akkumuletorenak, apimodule, infiniterecursion, pietersen, workersworkers, resplendent, enabledebugging, testupcase, ewidget, idehttpscio, httpraygrassocom, sparkly, documenttitle, addingblahtoreadme, firefoxchromium, hideshow, dimp, dbsavedesigncars, yadrs, mulligan, recurrence, nforce, datepicker, chapman, mongoidminitest, htmlstrong, megadhatjuk, sufficies, balncer, twolevel, mongoidyml, carnivorous, sortinggrouping, appenddatadata, aligning, antiferromagnetic, squeakcolors, treetree

Lovely to work with. Does cutting out more sparse (infrequent) terms help?

This does help a lot. A sample of the remaining terms looks more sensible:

> three, good, streams, man, minimum, batch, playing, details, subscribe, series, standard, reproduce, quickly, finish, allmodconfig, work, dealings, reach, json, goal, stage, proper, exit, model, rebuild, legal, authorization, restriction, spec, forget.

lda20 = LDA(dtm3, 20)

lda30 = LDA(dtm3, 30)

lda10 = LDA(dtm3, 10) dtm3 = dtm3[row_sums(dtm3) > 0,]

fit on a subset to check consistency of topics

sreadme_df = readme_df[1:4000,] corpus <- Corpus(VectorSource(sreadme_df$readme)) Sys.setlocale("LC_COLLATE", "C") sdtm <- DocumentTermMatrix(corpus, control = list(stemming = TRUE, stopwords = TRUE, wordLengths = c(1, 20), removeNumbers = TRUE, removePunctuation = TRUE))

sdtm3 = removeSparseTerms(sdtm, 0.99) sdtm3 = sdtm3[row_sums(sdtm3) > 0,] slda20 = LDA(sdtm3, 20)

dtms = dtm2 #calculate mean term frequency-inverse document frequency (tf-idf) term_tfidf <-+ tapply(dtm3$v/row_sums(dtm3)[dtm3$i], dtm3$j, mean) * log2(nDocs(dtm3)/col_sums(dtm3 > 0)) summary(term_tfidf)

use it to remove overly common terms

dtm4 <- dtm3[, term_tfidf >= 0.027] dtm4 <- dtm4[row_sums(dtm4) > 0,] summary(col_sums(dtm4))

lda1 = LDA(dtms, 20)

dtm2 <- dtm2[row_sums(dtm2) > 0,] ldafull = LDA(dtm2, 20)

implement the 10-fold cross-validation in LDA paper

setwd("C:\Dropbox\rmills\Postdoc\LDA")

do validation on a smaller scale (2k readmes, 10/20/30/40/50 topics, 4 folds, min wordlength 3)

setwd("C:\Dropbox\rmills\Postdoc\LDA2") sreadme_df = readme_df[1:2000,] corpus <- Corpus(VectorSource(sreadme_df$readme)) #does this help? Sys.setlocale("LC_COLLATE", "C")

sdtm <- DocumentTermMatrix(corpus, control = list(stemming = TRUE, stopwords = TRUE, wordLengths = c(3, 20), removeNumbers = TRUE, removePunctuation = TRUE))

sdtm <- removeSparseTerms(sdtm, 0.99) term_tfidf <-+ tapply(sdtm$v/row_sums(sdtm)[sdtm$i], sdtm$j, mean) * log2(nDocs(sdtm)/col_sums(sdtm > 0)) summary(term_tfidf) sdtm <- sdtm[, term_tfidf >= 0.027]

sdtm = sdtm[row_sums(sdtm) > 0,]

set.seed(0908) topics <- c(10, 20, 30, 40, 50) SEED <- 20080809

D <- nrow(sdtm) folding <-sample(rep(seq_len(4), ceiling(D))[seq_len(D)]) for (k in topics) { for (chain in seq_len(4)) { FILE <- paste("VEM", k, "", chain, ".rda", sep = "") training <- LDA(sdtm[folding != chain,], k = k, control = list(seed = SEED)) testing <- LDA(sdtm[folding == chain,], model = training, control = list(estimate.beta = FALSE, seed = SEED)) save(training, testing, file = file.path("results", FILE)) FILE <- paste("VEM_fixed", k, "", chain, ".rda", sep = "") training <- LDA(sdtm[folding != chain,], k = k, control = list(seed = SEED, estimate.alpha = FALSE)) testing <- LDA(sdtm[folding == chain,], model = training, control = list(estimate.beta = FALSE, seed = SEED)) save(training, testing, file = file.path("results", FILE)) FILE <- paste("Gibbs", k, "", chain, ".rda", sep = "") training <- LDA(sdtm[folding != chain,], k = k, control = list(seed = SEED, burnin = 400, thin = 40, iter = 400, best = FALSE), method = "Gibbs") best_training <- training@fitted[[which.max(logLik(training))]] testing <- LDA(sdtm[folding == chain,], model = best_training, control = list(estimate.beta = FALSE, seed = SEED, burnin = 400, thin = 40, iter = 400, best = FALSE)) save(training, testing, file = file.path("results", FILE)) } }

topics <- c(10, 20, 30, 40, 50) library("topicmodels") #readme_df("AssociatedPress", package = "topicmodels") D <- nrow(sdtm) folding <- sample(rep(seq_len(4), ceiling(D))[seq_len(D)]) AP_test <- AP_alpha <- list() for (method in c("VEM", "VEM_fixed", "Gibbs")) { AP_alpha[[method]] <- AP_test[[method]] <- matrix(NA, nrow = length(topics), ncol = 4, dimnames = list(topics, seq_len(4))) for (fold in seq_len(4)) { for (i in seq_along(topics)) { T <- topics[i] FILE <- paste(method, "", T, "", fold, ".rda", sep = "") load(file.path("results", FILE)) AP_alpha[[method]][paste(T),fold] <- if (is(training, "Gibbs_list")) training@fitted[[1]]@alpha else training@alpha AP_test[[method]][paste(T),fold]

<- perplexity(testing, sdtm[folding == fold,], use_theta = FALSE) } } } save(AP_alpha, AP_test, file = “AP.rda”)

p.perp = ggplot(AP_test) + geom_line(aes(x = Var1, y = Freq)) + scale_fill_brewer(type=“seq”, palette=“RdYlBu”) + labs(x = “Level - OP comments”, y = “No. of Comments”)

this is the full-on 10-fold validation... takes too long

set.seed(0908) topics <- 10 * c(1:5, 10, 20) SEED <- 20080809

D <- nrow(dtm4) folding <-sample(rep(seq_len(10), ceiling(D))[seq_len(D)]) for (k in topics) { for (chain in seq_len(10)) { FILE <- paste(“VEM*”, k, ””*, chain, “.rda”, sep = “”) training <- LDA(dtm4[folding != chain,], k = k, control = list(seed = SEED)) testing <- LDA(dtm4[folding == chain,], model = training, control = list(estimate.beta = FALSE, seed = SEED)) save(training, testing, file = file.path(“results”, FILE)) FILE <- paste(“VEM_fixed*”, k, ””*, chain, “.rda”, sep = “”) training <- LDA(dtm4[folding != chain,], k = k, control = list(seed = SEED, estimate.alpha = FALSE)) testing <- LDA(dtm4[folding == chain,], model = training, control = list(estimate.beta = FALSE, seed = SEED)) save(training, testing, file = file.path(“results”, FILE)) FILE <- paste(“Gibbs*”, k, ””*, chain, “.rda”, sep = “”) training <- LDA(dtm4[folding != chain,], k = k, control = list(seed = SEED, burnin = 1000, thin = 100, iter = 1000, best = FALSE), method = “Gibbs”) best_training <- training@fitted[[which.max(logLik(training))]] testing <- LDA(dtm4[folding == chain,], model = best_training, control = list(estimate.beta = FALSE, seed = SEED, burnin = 1000, thin = 100, iter = 1000, best = FALSE)) save(training, testing, file = file.path(“results”, FILE)) } }

exploring the cross-validation

topics <- c(10, 20) library(“topicmodels”) #readme_df(“AssociatedPress”, package = “topicmodels”) D <- nrow(dtm4) folding <- sample(rep(seq_len(10), ceiling(D))[seq_len(D)]) AP_test <- AP_alpha <- list() for (method in c(“VEM”, “VEM_fixed”, “Gibbs”)) { AP_alpha[[method]] <- AP_test[[method]] <- matrix(NA, nrow = length(topics), ncol = 10, dimnames = list(topics, seq_len(10))) for (fold in seq_len(10)) { for (i in seq_along(topics)) { T <- topics[i] FILE <- paste(method, “*”, T, ””*, fold, “.rda”, sep = “”) load(file.path(“results”, FILE)) AP_alpha[[method]][paste(T),fold] <- if (is(training, “Gibbs_list”)) training@fitted[[1]]@alpha else training@alpha AP_test[[method]][paste(T),fold] <- perplexity(testing, dtm4[folding == fold,], use_theta = FALSE) } } } save(AP_alpha, AP_test, file = “AP.rda”)

lda <- LDA(matrix, 10)

how many rows(documents) and columns(words) does the matrix have?

dim(dtm)

for inspecting the matrix

inspect(dtm[1:5, 2]) inspect(dtm2[1:5,1:50])

can save the document-topic assignments, but can I also re-label the documents as their full_name as opposed to the full text?

Topics = topics(lda, 2)

shows the 5 most common terms for each topic

Terms = terms(lda, 5)

shows terms for a topic which meet threshold probability

terms(lda, threshold = 0.05)

create_matrix is the Rtexttools version

try this to keep the size down and remove overly common terms

dtm <- create_matrix(readme_df$readme, language="english", removeNumbers=TRUE, stemWords=FALSE, weighting=weightTf, maxWordLength = 20, minWordLength = 1, minDocFreq = 5, maxDocFreq = 6000)

working on a subset

sreadme_df = readme_df[1:1000,] dtm <- create_matrix(sreadme_df$readme, language="english", removeNumbers=TRUE, stemWords=FALSE, weighting=weightTf, maxWordLength = 20, minWordLength = 1, minDocFreq = 5) rowTotals <- apply(dtm , 1, sum) #Find the sum of words in each Document dtm.new <- dtm[rowTotals> 0] #remove all docs without words

lda = LDA(dtm.new, 20)

sreadme_df = readme_df[1:500,]