

Practical 2 - Electronic Medical Records

Alexandra Del Favero-Campbell

2024-05-27

```
knitr::opts_chunk$set(echo = TRUE)
library(readr)

## Warning: package 'readr' was built under R version 4.3.3

library(dplyr)

## Warning: package 'dplyr' was built under R version 4.3.3

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
## 
##     filter, lag

## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union

library(tidyr)

## Warning: package 'tidyr' was built under R version 4.3.3

library(ggplot2)

## Warning: package 'ggplot2' was built under R version 4.3.3

library(scales)

##
## Attaching package: 'scales'

## The following object is masked from 'package:readr':
## 
##     col_factor
```

```
library(tidytext)

## Warning: package 'tidytext' was built under R version 4.3.3

library(textstem)

## Warning: package 'textstem' was built under R version 4.3.3

## Loading required package: koRpus.lang.en

## Warning: package 'koRpus.lang.en' was built under R version 4.3.3

## Loading required package: koRpus

## Warning: package 'koRpus' was built under R version 4.3.3

## Loading required package: syllly

## Warning: package 'syllly' was built under R version 4.3.3

## For information on available language packages for 'koRpus', run
##
##   available.koRpus.lang()
##
## and see ?install.koRpus.lang()

##
## Attaching package: 'koRpus'

## The following object is masked from 'package:readr':
##
##     tokenize

library(clinspacy)

## Warning: package 'clinspacy' was built under R version 4.3.3

## Welcome to clinspacy.

## By default, this package will install and use miniconda and create a "clinspacy" conda environment.

## If you want to override this behavior, use clinspacy_init(miniconda = FALSE) and specify an alternat

library(topicmodels)

## Warning: package 'topicmodels' was built under R version 4.3.3
```

```
library('reshape2')

## Warning: package 'reshape2' was built under R version 4.3.3

##
## Attaching package: 'reshape2'

## The following object is masked from 'package:tidyverse':
## 
##     smiths

library(stringr)
```

This practical is based on exploratory data analysis, named entity recognition, and topic modelling of unstructured medical note free-text data derived from electronic medical records (EMR). Real EMR data is very difficult to access without a specific need/request so this data set is derived from medical transcription data instead. I'll also caveat that the options of natural language processing (NLP) in R are far inferior to those available in Python.

First, install the packages in the setup block (`install.packages(c("readr", "dplyr", "tidyR", "ggplot2", "tidytext", "textstem", "clinspacy", "topicmodels", "reshape2"))`).

Note: To try and make it clearer which library certain functions are coming from clearer, I'll try to do explicit imports throughout this notebook.

Data Parsing

After that we can grab the dataset directly from the `clinspacy` library.

```
raw.data <- clinspacy::dataset_mtsamples()  
dplyr::glimpse(raw.data)
```

```
## Rows: 4,999
## Columns: 6
## $ note_id      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1~
## $ description   <chr> "A 23-year-old white female presents with complaint ~
## $ medical_specialty <chr> "Allergy / Immunology", "Bariatrics", "Bariatrics", ~
## $ sample_name    <chr> "Allergic Rhinitis", "Laparoscopic Gastric Bypass Co~
## $ transcription  <chr> "SUBJECTIVE:, This 23-year-old white female present~
## $ keywords       <chr> "allergy / immunology, allergic rhinitis, allergies,~
```

There is no explanation or data dictionary with this dataset, which is a surprisingly common and frustrating turn of events!

1 Using the output of dplyr's `glimpse` command (or rstudio's data viewer by clicking on `raw.data` in the Environment pane) provide a description of what you think each variable in this dataset contains.

Based on a view of the data, the following are the descriptions for each variable:

note_id: a unique numerical identifier to denote each clinical note written by a doctor about a patient.

description: a brief description of the patient's presentation at the appointment or the reason for the visit from the doctor.

medical_specialty: the medical specialty of the doctor that the patient is seeing or is there to see.

sample_name: the medical procedure or service the doctor was going to provide or that the patient was there for.

transcription: doctor's basic clinical notes about the patient, their reason for being at the given appointment, brief patient history, patient's allergies, preoperative and postoperative diagnoses, procedures, etc.

keywords: some of the keywords used to help describe the themes written within the "transcription" column for each "note_id."

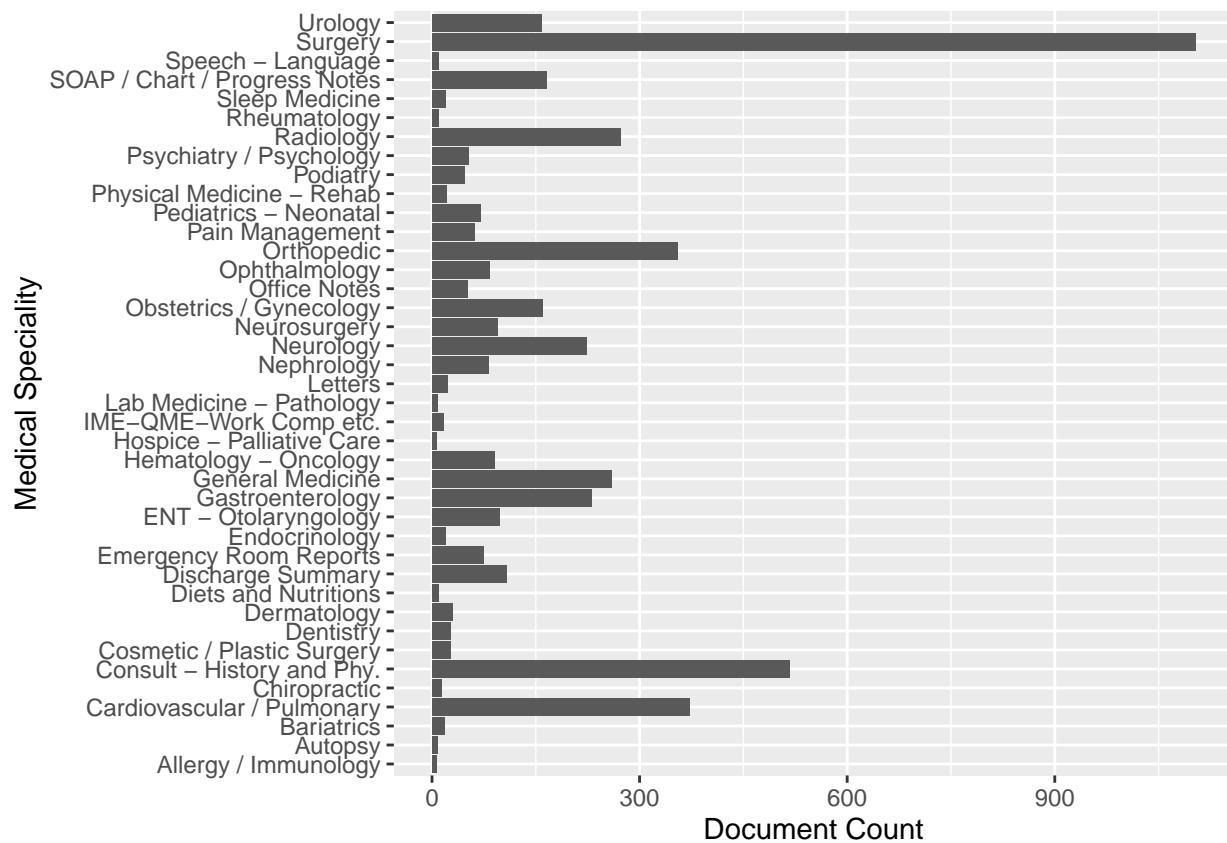
Let's see how many different medical specialties are featured in these notes:

```
raw.data %>% dplyr::select(medical_specialty) %>% dplyr::n_distinct()
```

```
## [1] 40
```

So, how many transcripts are there from each specialty:

```
ggplot2::ggplot(raw.data, ggplot2::aes(y=medical_specialty)) + ggplot2::geom_bar() + labs(x="Document C")
```



Let's make our life easier and filter down to 3 specialties: a diagnostic/lab, a medical, and a surgical specialty

```
filtered.data <- raw.data %>% dplyr::filter(medical_specialty %in% c("Orthopedic", "Radiology", "Surgery"))
```

Text Processing

Let's now apply our standard pre-processing to the transcripts from these specialties.
We are going to use the `tidytext` package to tokenise the transcript free-text.

Let's remove stop words first. e.g., "the", "of", "to", and so forth. These are known as stop words and we can remove them relative easily using a list from `tidytext::stop_words` and `dplyr::anti_join()`

```
analysis.data <- filtered.data %>%
  unnest_tokens(word, transcription) %>%
  mutate(word = str_replace_all(word, "[^[:alnum:]]", "")) %>%
  filter(!str_detect(word, "[0-9]")) %>%
  anti_join(stop_words) %>%
  group_by(note_id) %>%
  summarise(transcription = paste(word, collapse = " ")) %>%
  left_join(select(filtered.data, -transcription), by = "note_id")
```

```
## Joining with 'by = join_by(word)'
```

Now let's tokenize the `transcription` to words (unigram). By default this tokenises to words but other options include characters, n-grams, sentences, lines, paragraphs, or separation around a regular expression.

```
tokenized.data.unigram <- analysis.data %>% tidytext::unnest_tokens(word, transcription, to_lower=TRUE)
```

You can also do bi-grams

```
tokenized.data <- analysis.data %>% tidytext::unnest_tokens(ngram, transcription, token = "ngrams", n=2)
```

How many stop words are there in `tidytext::stop_words` from each lexicon?

```
tidytext::stop_words %>% dplyr::group_by(lexicon) %>% dplyr::distinct(word) %>% dplyr::summarise(n=dplyr::n())

## # A tibble: 3 x 2
##   lexicon      n
##   <chr>     <int>
## 1 SMART      570
## 2 onix       398
## 3 snowball   174
```

2 How many unique unigrams are there in the transcripts from each specialty:

```
tokenized.data.unigram %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(word) %>% dplyr::summarise(n=dplyr::n())

## # A tibble: 3 x 2
##   medical_specialty      n
##   <chr>                <int>
## 1 Orthopedic            7681
## 2 Radiology             5933
## 3 Surgery               11977
```

Let's plot some distribution of unigram tokens (words)

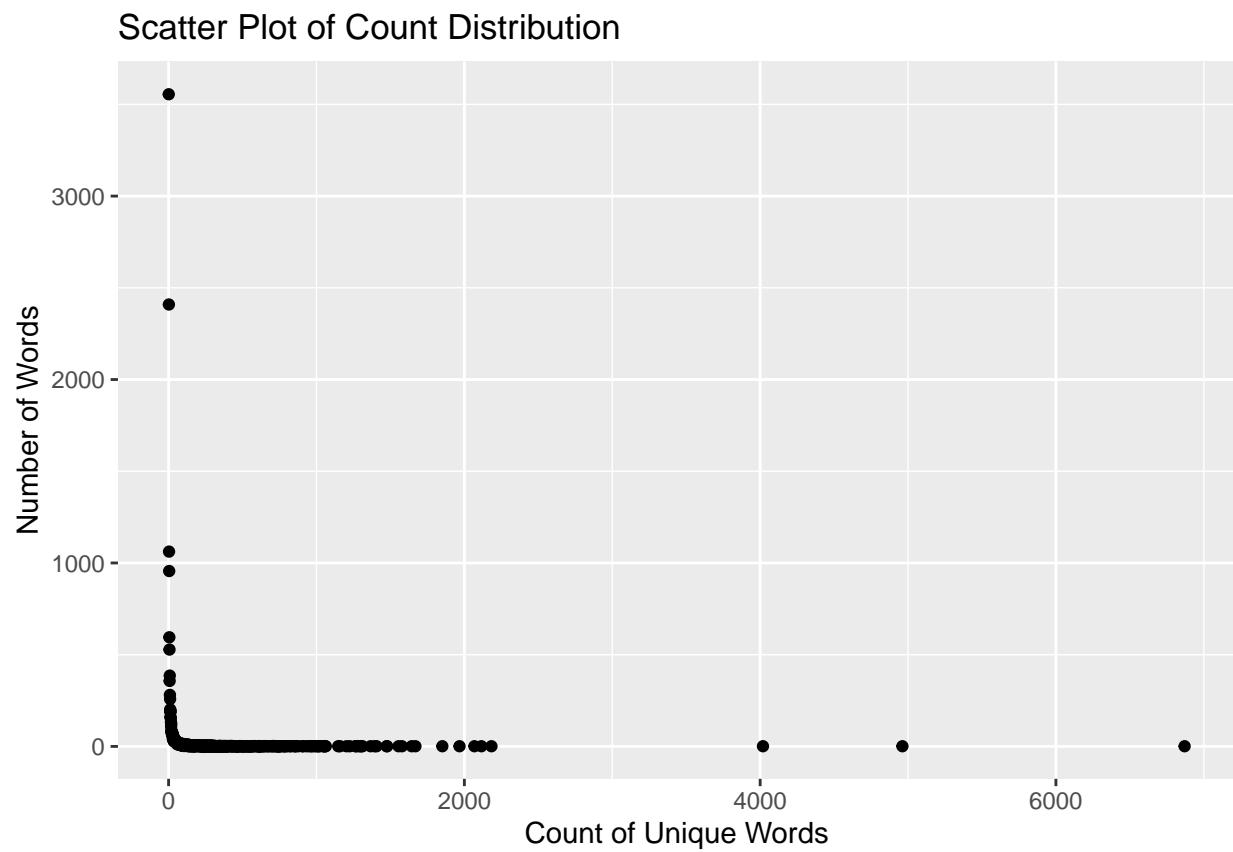
```

word_counts <- tokenized.data.unigram %>%
  group_by(word) %>%
  summarise(count = n()) %>%
  ungroup() %>%
  arrange(desc(count))

count_distribution <- word_counts %>%
  group_by(count) %>%
  summarise(num_words = n()) %>%
  ungroup()

ggplot2::ggplot(count_distribution, aes(x = count, y = num_words)) +
  geom_point() +
  labs(title = "Scatter Plot of Count Distribution",
       x = "Count of Unique Words",
       y = "Number of Words")

```



Let's plot some distribution of bigram tokens (words)

```

word_counts <- tokenized.data %>%
  group_by(ngram) %>%
  summarise(count = n()) %>%
  ungroup() %>%
  arrange(desc(count))

count_distribution <- word_counts %>%

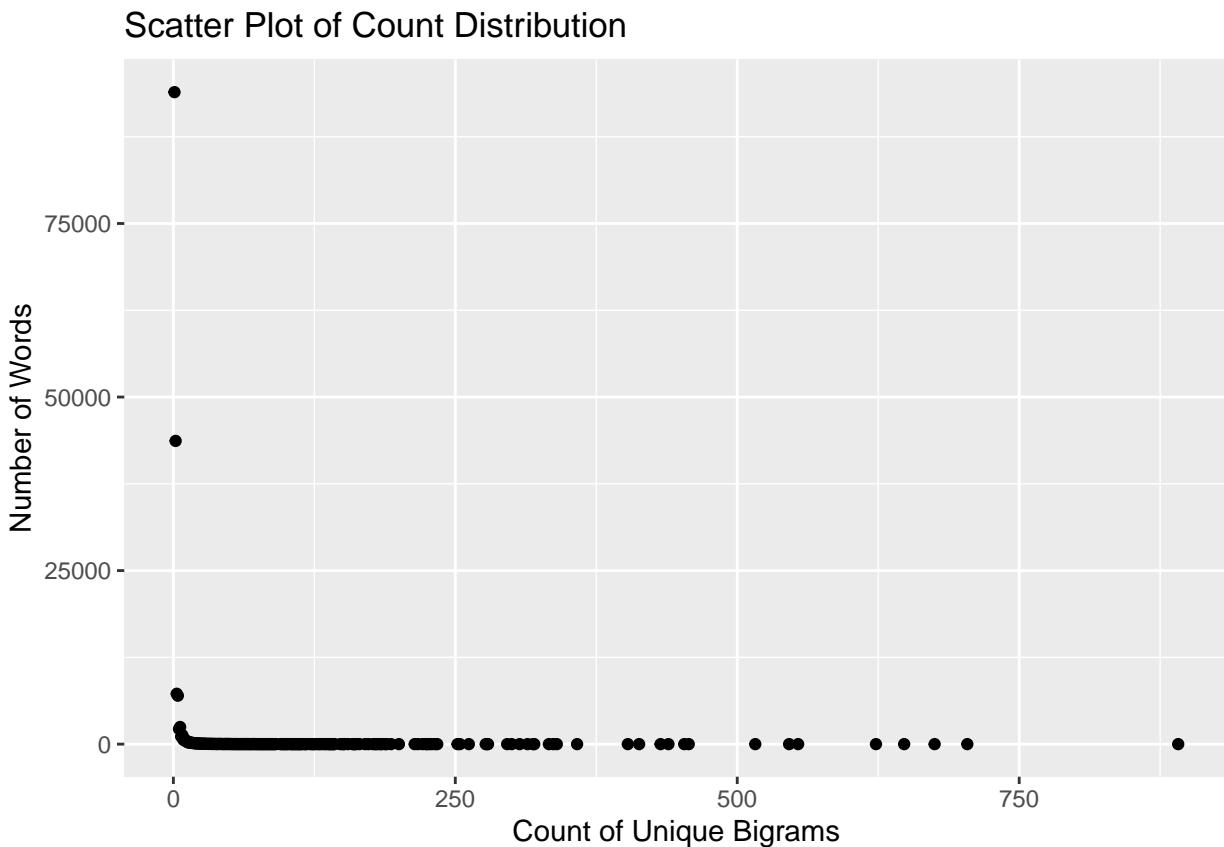
```

```

group_by(count) %>%
summarise(num_words = n()) %>%
ungroup()

ggplot2::ggplot(count_distribution, aes(x = count, y = num_words)) +
  geom_point() +
  labs(title = "Scatter Plot of Count Distribution",
       x = "Count of Unique Bigrams",
       y = "Number of Words")

```



3 How many unique bi-grams are there in each category without stop words and numbers?

```

# Filter the data for the selected specialties
filtered.data <- raw.data %>%
  dplyr::filter(medical_specialty %in% c("Orthopedic", "Radiology", "Surgery"))

# Process the data to remove stop words and numbers
cleaned.data <- filtered.data %>%
  unnest_tokens(word, transcription) %>%
  mutate(word = str_replace_all(word, "[^[:alnum:]]", "")) %>%
  filter(!str_detect(word, "[0-9]")) %>%
  anti_join(tidytext::stop_words) %>%
  group_by(note_id) %>%
  summarise(transcription = paste(word, collapse = " ")) %>%
  left_join(select(filtered.data, -transcription), by = "note_id")

```

```

## Joining with 'by = join_by(word)'

# Tokenize the cleaned transcription to bi-grams
tokenized.data <- cleaned.data %>%
  tidytext::unnest_tokens(ngram, transcription, token = "ngrams", n = 2, to_lower = TRUE) %>%
  mutate(ngram = str_replace_all(ngram, "[^[:alnum:]\\s]", "")) %>%
  separate(ngram, into = c("word1", "word2"), sep = " ") %>%
  filter(!word1 %in% tidytext::stop_words$word,
         !word2 %in% tidytext::stop_words$word,
         !str_detect(word1, "[0-9]"),
         !str_detect(word2, "[0-9]")) %>%
  unite("ngram", word1, word2, sep = " ")

# Count unique bi-grams in each category
unique_bigrams_per_category <- tokenized.data %>%
  group_by(medical_specialty) %>%
  dplyr::distinct(ngram) %>%
  dplyr::summarise(n = dplyr::n())

# Print the results
print(unique_bigrams_per_category)

## # A tibble: 3 x 2
##   medical_specialty     n
##   <chr>             <int>
## 1 Orthopedic          55730
## 2 Radiology           28293
## 3 Surgery             130403

```

Sometimes we are interested in tokenising/segmenting things other than words like whole sentences or paragraphs.

4 How many unique sentences are there in each category? Hint: use ?tidytext::unnest_tokens to see the documentation for this function.

```

tokenized.data.sentences <- analysis.data %>% tidytext::unnest_tokens(sentences, transcription, token="n")
tokenized.data.sentences %>% dplyr::group_by(medical_specialty) %>% dplyr::distinct(sentences) %>% dplyr::summarise(n = dplyr::n())

## # A tibble: 3 x 2
##   medical_specialty     n
##   <chr>             <int>
## 1 Orthopedic          354
## 2 Radiology           273
## 3 Surgery             1087

```

Now that we've tokenized to words and removed stop words, we can find the most commonly word used within each category:

```

tokenized.data %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::count(ngram, sort = TRUE) %>%
  dplyr::top_n(5)

```

```

## Selecting by n

## # A tibble: 16 x 3
## # Groups:   medical_specialty [3]
##   medical_specialty ngram          n
##   <chr>           <chr>        <int>
## 1 Surgery         prepped draped    696
## 2 Surgery         preoperative diagnosis 555
## 3 Surgery         procedure patient  551
## 4 Surgery         postoperative diagnosis 518
## 5 Surgery         tolerated procedure 515
## 6 Orthopedic      prepped draped    183
## 7 Orthopedic      preoperative diagnosis 141
## 8 Orthopedic      lower extremity   139
## 9 Orthopedic      range motion    139
## 10 Orthopedic     postoperative diagnosis 124
## 11 Radiology      carotid artery   59
## 12 Radiology      heart rate      52
## 13 Radiology      reason exam     51
## 14 Radiology      left ventricular 50
## 15 Radiology      coronary artery  43
## 16 Radiology      exam unremarkable 43

```

We should lemmatize the tokenized words to prevent over counting of similar words before further analyses. Annoyingly, `tidytext` doesn't have a built-in lemmatizer.

5 Do you think a general purpose lemmatizer will work well for medical data? Why might it not?
 It is likely to not work that well for medical data purposes because medical jargon is very different from language of every day general purpose use. Thus, general purpose lemmatizers are just not trained to handle medical contextual vocabulary and jargon as accurately. Furthermore, anatomical terms and even drug names are not really common in common every day language, so a general purpose lemmatizer likely would have a hard time processing such things correctly. Therefore, a general purpose lemmatizer may not be the best way to go when it comes to using this in a medical setting.

Unfortunately, a specialised lemmatizer like in `clinspacy` is going to be very painful to install so we will just use a simple lemmatizer for now:

```
lemmatized.data <- tokenized.data %>% dplyr::mutate(lemma=textstem::lemmatize_words(ngram))
```

We can now calculate the frequency of lemmas within each specialty and note.

```

lemma.freq <- lemmatized.data %>%
  dplyr::count(medical_specialty, lemma) %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::mutate(proportion = n / sum(n)) %>%
  tidyr::pivot_wider(names_from = medical_specialty, values_from = proportion) %>%
  tidyr::pivot_longer(`Surgery`:`Radiology`,
                     names_to = "medical_specialty", values_to = "proportion")

```

And plot the relative proportions

```
ggplot2::ggplot(lemma.freq, ggplot2::aes(x=proportion,
                                         y="Orthopedic",
```

```

color=abs(`Orthopedic` - proportion))) +
ggplot2::geom_abline(color="gray40", lty=2) +
ggplot2::geom_jitter(alpha=0.1, size=2.5, width=0.3, height=0.3) +
ggplot2::geom_text(ggplot2::aes(label=lemma), check_overlap=TRUE, vjust=1.5) +
ggplot2::scale_x_log10(labels=scales::percent_format()) +
ggplot2::scale_y_log10(labels=scales::percent_format()) +
ggplot2::scale_color_gradient(limits=c(0, 0.001), low="darkslategray4", high="gray75") +
ggplot2::facet_wrap(~medical_specialty, ncol = 2) +
ggplot2::theme(legend.position="none") +
ggplot2:: labs(y="Orthopedic", x = NULL)

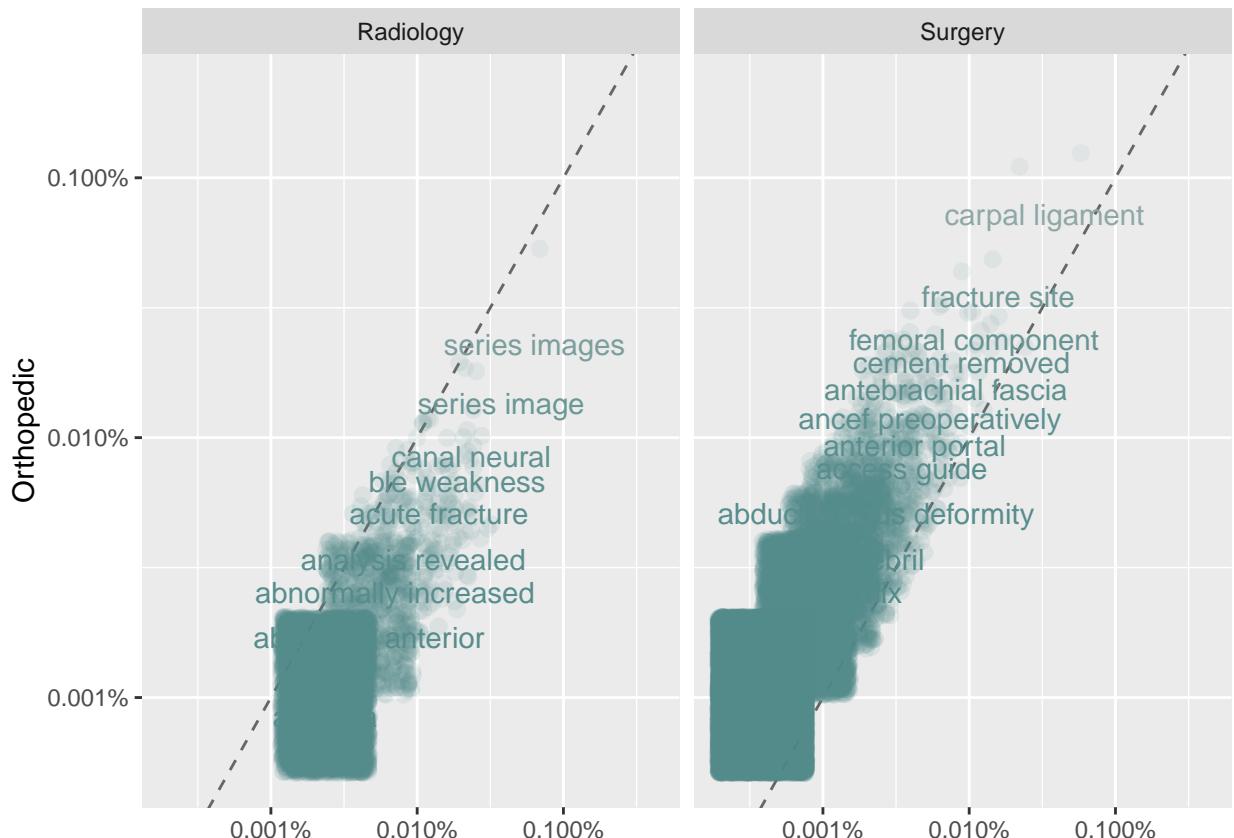
```

```

## Warning: Removed 314398 rows containing missing values or values outside the scale range
## ('geom_point()').

## Warning: Removed 314398 rows containing missing values or values outside the scale range
## ('geom_text()').

```



6 What does this plot tell you about the relative similarity of lemma frequencies between Orthopedic and Radiology and between Orthopedic and Surgery? Based on what these specialties involve, is this what you would expect?

Based on this plot, we see that for the plot for Orthopedic vs. Surgery, there are more data points scattered away from the diagonal line, which means that certain lemmas may be more frequently used in a Surgical field as compared to an Orthopedic field. There are more significant disparities between the lemma frequencies between those two categories and potentially greater differences in the use of certain terminologies between Orthopedics and Surgery.

We also see that there are higher relative similarities of lemma frequencies between Orthopedics and Radiology than between Orthopedics and Surgery. The datapoints are more concentrated along the diagonal line as compared to the Orthopedics vs. Surgery plot, potentially indicating a higher degree of overlap in the common language used in these two specialties. Based on what these specialties involve (i.e., Orthopedics involves patients' bones and joints, Radiology involves imaging of the body, and Surgery involves cutting into patients' bodies), these results do make sense and are what you would expect. Orthopedic doctors rely heavily on imaging from Radiologists to do their job and require multiple images from their patients over the course of every single patient's care. Surgeons also involve some imaging before being able to cut into patients, but definitely less constant imaging from Radiologists than Orthopedic doctors require. Although definitely not exactly the same, a lot of what the Radiologists find in their imaging and the jargon they use would be information that Orthopedic doctors would need to do their jobs and are likely to be more similar to the jargon and diagnostic terms that Orthopedic doctors would use.

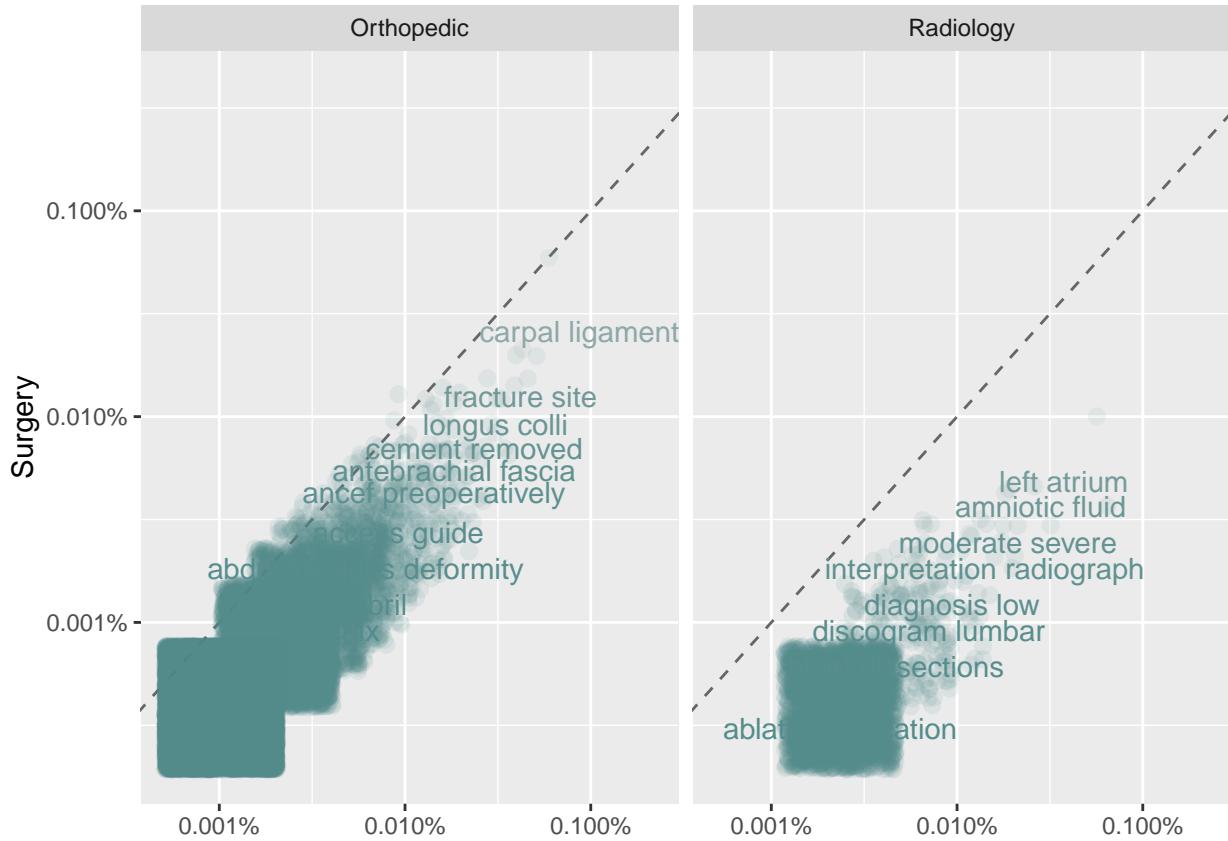
7 Modify the above plotting code to do a direct comparison of Surgery and Radiology (i.e., have Surgery or Radiology on the Y-axis and the other 2 specialties as the X facets)

```
lemma.freq.1 <- lemmatized.data %>%
  dplyr::count(medical_specialty, lemma) %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::mutate(proportion = n / sum(n)) %>%
  tidyr::pivot_wider(names_from = medical_specialty, values_from = proportion) %>%
  tidyr::pivot_longer(`Orthopedic`:`Radiology`,
    names_to = "medical_specialty", values_to = "proportion")

ggplot2::ggplot(lemma.freq.1, ggplot2::aes(x=proportion,
                                             y="Surgery",
                                             color=abs(`Surgery` - proportion))) +
  ggplot2::geom_abline(color="gray40", lty=2) +
  ggplot2::geom_jitter(alpha=0.1, size=2.5, width=0.3, height=0.3) +
  ggplot2::geom_text(ggplot2::aes(label=lemma), check_overlap=TRUE, vjust=1.5) +
  ggplot2::scale_x_log10(labels=scales::percent_format()) +
  ggplot2::scale_y_log10(labels=scales::percent_format()) +
  ggplot2::scale_color_gradient(limits=c(0, 0.001), low="darkslategray4", high="gray75") +
  ggplot2::facet_wrap(~medical_specialty, ncol=2) +
  ggplot2::theme(legend.position="none") +
  ggplot2:: labs(y="Surgery", x = NULL)

## Warning: Removed 317812 rows containing missing values or values outside the scale range
## ('geom_point()').

## Warning: Removed 317812 rows containing missing values or values outside the scale range
## ('geom_text()').
```



TF-IDF Normalisation

Maybe looking at lemmas across all notes in a specialty is misleading, what if we look at lemma frequencies across a specialty.

```
lemma.counts <- lemmatized.data %>% dplyr::count(medical_specialty, lemma)
total.counts <- lemma.counts %>%
  dplyr::group_by(medical_specialty) %>%
  dplyr::summarise(total=sum(n))

all.counts <- dplyr::left_join(lemma.counts, total.counts)

## Joining with `by = join_by(medical_specialty)`
```

Now we can calculate the term frequency / invariant document frequency (tf-idf):

```
all.counts.tfidf <- tidytext::bind_tf_idf(all.counts, lemma, medical_specialty, n)
```

We can then look at the top 10 lemma by tf-idf within each specialty:

```
all.counts.tfidf %>% dplyr::group_by(medical_specialty) %>% dplyr::slice_max(order_by=tf_idf, n=10)
```

```
## # A tibble: 30 x 7
```

```

## # Groups: medical_specialty [3]
##   medical_specialty lemma          n total      tf    idf   tf_idf
##   <chr>           <chr>      <int> <int>     <dbl> <dbl>    <dbl>
## 1 Orthopedic      range motion    139 97774 0.00142  0.405 0.000576
## 2 Orthopedic      carpal ligament  85 97774 0.000869 0.405 0.000352
## 3 Orthopedic      transverse carpal 81 97774 0.000828 0.405 0.000336
## 4 Orthopedic      extremity prepped 79 97774 0.000808 0.405 0.000328
## 5 Orthopedic      proximal phalanx  75 97774 0.000767 0.405 0.000311
## 6 Orthopedic      department anesthesia 63 97774 0.000644 0.405 0.000261
## 7 Orthopedic      dissection carried 63 97774 0.000644 0.405 0.000261
## 8 Orthopedic      steri strips     59 97774 0.000603 0.405 0.000245
## 9 Orthopedic      closed vicryl    58 97774 0.000593 0.405 0.000241
## 10 Orthopedic     dressing applied   58 97774 0.000593 0.405 0.000241
## # i 20 more rows

```

8 Are there any lemmas that stand out in these lists? Why or why not?

We notice that there are way more frequently used lemmas in the Surgical category (e.g., “closed vicryl”, “anesthesia endotracheal”, “steri strips”, “dissection carried”) than there are in either of the other 2 categories. In the other categories, we see potentially more general terms that could be used outside of medical context being more frequent. For example, in the Orthopedic category, “range motion” was found to be the most frequently used. This makes sense in why it is used frequently in this profession in particular (that is basically a big part of an Orthopedic doctor’s job and concern for their patients), but it is also a term that could be used in many other general contexts. In the Radiology category, there are way less frequent lemmas recognized, especially as compared to the other two medical specialty categories, and some of the lemmas that are identified are potentially not very useful or very specific-sounding to a radiology specialty (e.g., “pupils decreasing” or “evoked response”). In other words, these terms do not instantly scream “Radiology” to me.

However, each specialty category does show lemmas that seem very medically relevant for that particular specialty. For example, in Radiology, lemmas such as “myocardial perfusion” and “perfusion imaging” are very relevant to this specialty because they have to do with assessments and imaging that is required to be done for heart and heart function evaluations. In addition, the lemmas used in the Orthopedics and Surgical categories seemed very specific to orthopedic terminology (e.g., “carpal ligament”) and surgical terms or surgical body positions (e.g., “lithotomy position”), respectively. We also notice some lemmas that were found to be common between the Surgical and Orthopedic category lists, such as “closed vicryl” and “steri strips.” This means that these terms are commonly used across at least these two disciplines and for an assortment of different procedures and in various medical contexts.

We can look at transcriptions in full using these lemmas to check how they are used with `stringr::str_detect`

```

analysis.data %>% dplyr::select(medical_specialty, transcription) %>% dplyr::filter(stringr::str_detect

## # A tibble: 1 x 2
##   medical_specialty transcription
##   <chr>           <chr>
## 1 Surgery          preoperative diagnoses hallux rigidus left foot elevated me-

```

9 Extract an example of one of the other “top lemmas” by modifying the above code

```

analysis.data %>% dplyr::select(medical_specialty, transcription) %>% dplyr::filter(stringr::str_detect

## # A tibble: 1 x 2
##   medical_specialty transcription
##   <chr>           <chr>
## 1 Radiology        cc headache hx rhm moving furniture days prior presentation-

```

Topic Modelling

In NLP, we often have collections of documents (in our case EMR transcriptions) that we'd like to divide into groups so that we can understand them separately. Topic modeling is a method for unsupervised classification of such documents, similar to clustering on numeric data.

Latent Dirichlet allocation (LDA) is a particularly popular method for fitting a topic model. It treats each document as a mixture of topics, and each topic as a mixture of words. This allows documents to “overlap” each other in terms of content, rather than being separated into discrete groups, in a way that mirrors typical use of natural language.

- Every document is a mixture of topics. We imagine that each document may contain words from several topics in particular proportions. For example, in a two-topic model we could say “Document 1 is 90% topic A and 10% topic B, while Document 2 is 30% topic A and 70% topic B.”
- Every topic is a mixture of words. For example, we could imagine a two-topic model of American news, with one topic for “politics” and one for “entertainment.” The most common words in the politics topic might be “President”, “Congress”, and “government”, while the entertainment topic may be made up of words such as “movies”, “television”, and “actor”. Importantly, words can be shared between topics; a word like “budget” might appear in both equally.

LDA is a mathematical method for estimating both of these at the same time: finding the mixture of words that is associated with each topic, while also determining the mixture of topics that describes each document. There are a number of existing implementations of this algorithm, and we'll explore one of them in depth.

First lets calculate a term frequency matrix for each transcription:

```
lemma.counts <- lemmatized.data %>% dplyr::count(note_id, lemma)
total.counts <- lemma.counts %>%
  dplyr::group_by(note_id) %>%
  dplyr::summarise(total=sum(n))

all.counts <- dplyr::left_join(lemma.counts, total.counts)

## Joining with `by = join_by(note_id)`

emr.dcm <- all.counts %>% tidytext::cast_dtm(note_id, lemma, n)
```

Then we can use LDA function to fit a 5 topic ($k=5$) LDA-model

```
emr.lda <- topicmodels::LDA(emr.dcm, k=5, control=list(seed=42))
emr.topics <- tidytext::tidy(emr.lda, matrix='beta')
```

Then we can extract the top terms per assigned topic:

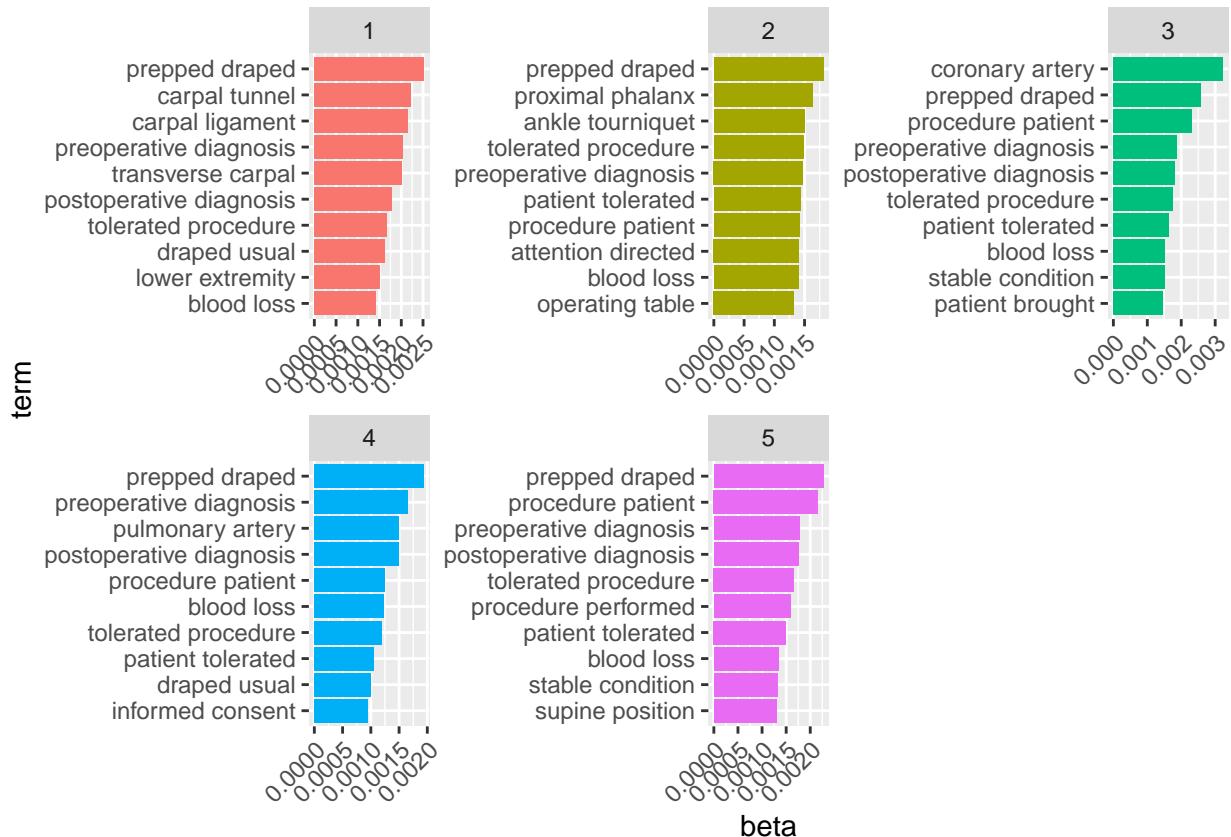
```
top.terms <- emr.topics %>% dplyr::group_by(topic) %>%
  dplyr::slice_max(beta, n=10) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(topic, -beta)

top.terms %>%
  dplyr::mutate(term=tidytext::reorder_within(term, beta, topic)) %>%
```

```

ggplot2::ggplot(ggplot2::aes(beta, term, fill=factor(topic))) +
  ggplot2::geom_col(show.legend=FALSE) +
  ggplot2::facet_wrap(~ topic, scales='free') +
  ggplot2::theme(axis.text.x = element_text(angle = 45,vjust = 1,hjust = 1)) +
  tidytext::scale_y_reordered()

```



Now we can ask how well do these assigned topics match up to the medical specialties from which each of these transcripts was derived.

```

specialty_gamma <- tidytext::tidy(emr.lda, matrix='gamma')

# we need to join in the specialty from the note_id
note_id_specialty_mapping <- lemmatized.data %>%
  dplyr::mutate(document=as.character(note_id)) %>%
  dplyr::select(document, medical_specialty) %>%
  dplyr::distinct()

specialty_gamma <- dplyr::left_join(specialty_gamma, note_id_specialty_mapping)

```

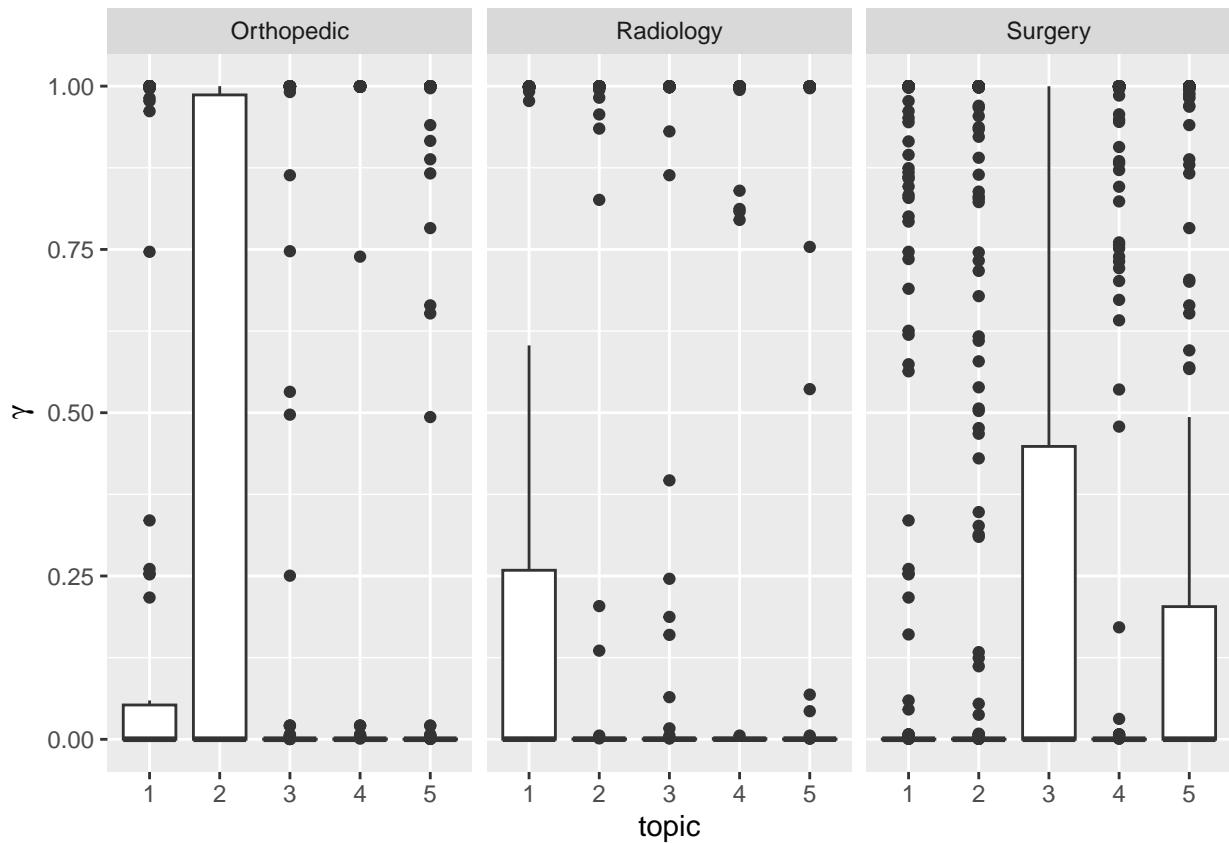
```
## Joining with 'by = join_by(document)'
```

```

specialty_gamma %>%
  dplyr::mutate(medical_specialty = reorder(medical_specialty, gamma * topic)) %>%
  ggplot2::ggplot(ggplot2::aes(factor(topic), gamma)) +
  ggplot2::geom_boxplot() +

```

```
ggplot2::facet_wrap(~ medical_specialty) +
  ggplot2::labs(x = "topic", y = expression(gamma))
```



Interestingly, Surgery, Orthopedic, and Radiology assign mostly to a single topics. We'd possibly expect this from radiology due to referring to imaging for many different diagnoses/reasons. However, this may all just reflect we are using too few topics in our LDA to capture the range of possible assignments.

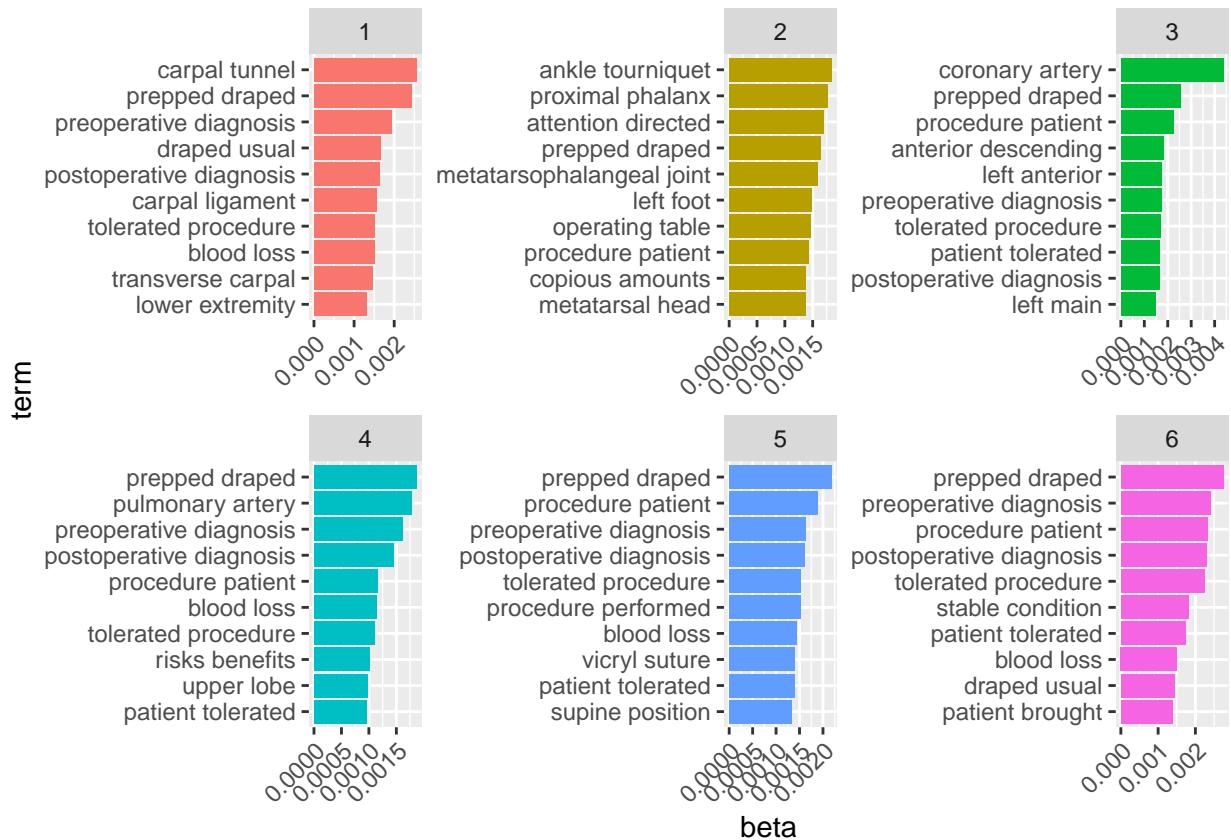
10 Repeat this with a 6 topic LDA, do the top terms from the 5 topic LDA still turn up? How do the specialties get split into sub-topics?

```
#LDA function to fit a 6 topic (`k=5`) LDA-model
emr.lda <- topicmodels::LDA(emr.dcm, k=6, control=list(seed=42))
emr.topics <- tidytext::tidy(emr.lda, matrix='beta')
```

```
#Extract the top terms per assigned topic
top.terms <- emr.topics %>% dplyr::group_by(topic) %>%
  dplyr::slice_max(beta, n=10) %>%
  dplyr::ungroup() %>%
  dplyr::arrange(topic, -beta)

top.terms %>%
  dplyr::mutate(term=tidytext::reorder_within(term, beta, topic)) %>%
  ggplot2::ggplot(ggplot2::aes(beta, term, fill=factor(topic))) +
  ggplot2::geom_col(show.legend=FALSE) +
  ggplot2::facet_wrap(~ topic, scales='free')
```

```
ggplot2::theme(axis.text.x = element_text(angle = 45,vjust = 1,hjust = 1)) +
tidytext::scale_y_reordered()
```



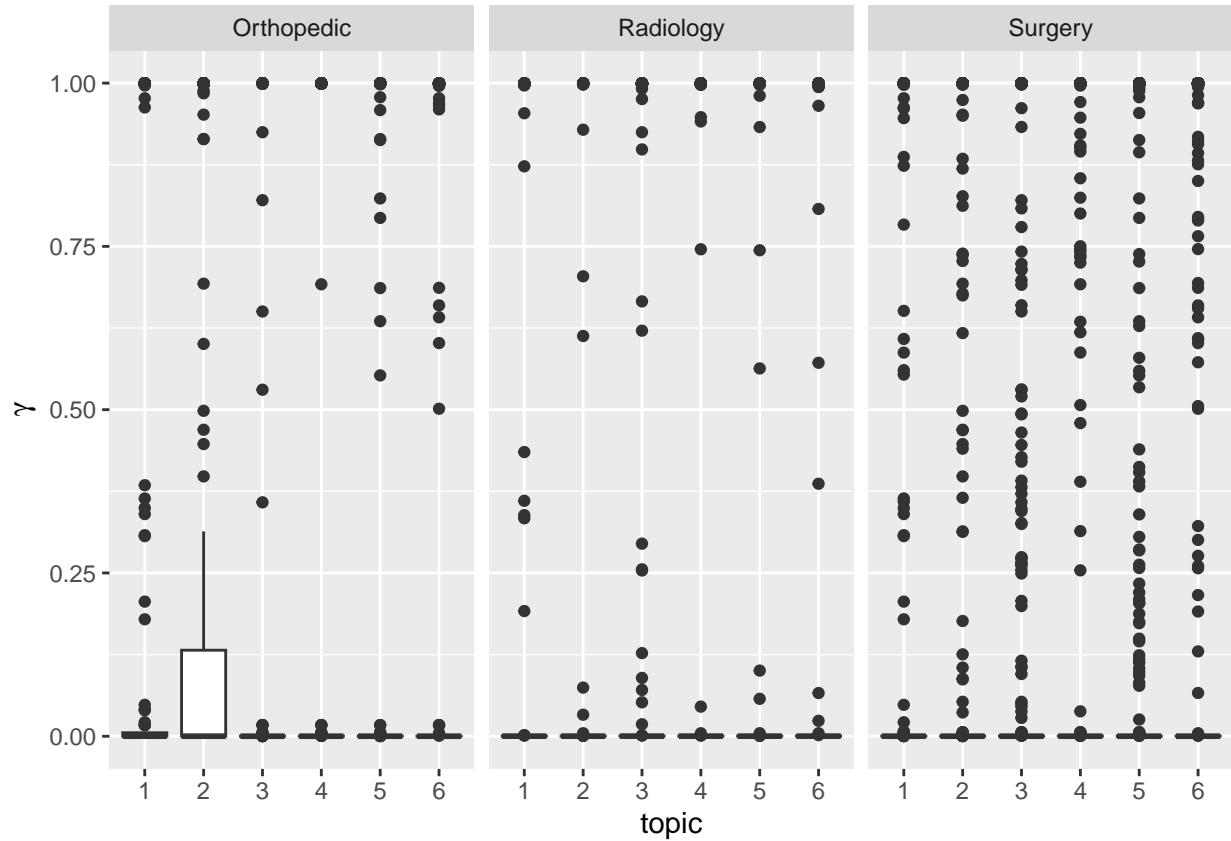
```
specialty_gamma <- tidytext::tidy(emr.lda, matrix='gamma')
```

```
# we need to join in the specialty from the note_id
note_id_specialty_mapping <- lemmatized.data %>%
  dplyr::mutate(document=as.character(note_id)) %>%
  dplyr::select(document, medical_specialty) %>%
  dplyr::distinct()
```

```
specialty_gamma <- dplyr::left_join(specialty_gamma, note_id_specialty_mapping)
```

```
## Joining with 'by = join_by(document)'
```

```
specialty_gamma %>%
  dplyr::mutate(medical_specialty = reorder(medical_specialty, gamma * topic)) %>%
  ggplot2::ggplot(ggplot2::aes(factor(topic), gamma)) +
  ggplot2::geom_boxplot() +
  ggplot2::facet_wrap(~ medical_specialty) +
  ggplot2::labs(x = "topic", y = expression(gamma))
```



Based on comparing the results of the 5-topic LDA and the 6-topic LDA, we see that top terms from the 5-topic LDA model still generally appear in the 6-topic LDA model. In fact, the same 10 terms for the 1st sub-topic are the same in both LDAs. However, we also see that the top terms from the 5-topic LDA are different from those that turn up for the 6-topic LDA for certain sub-topics. For example, in the 2nd topic part of both plots, we see that the terms, “metatarsophalangeal joint”, “left foot”, and “metatarsal head” appear in the 6-topic LDA but do not appear in the 5-topic LDA for this sub-topic. All of these terms sound very much like common Orthopedic terms, so it would make sense that in our plot of 6-topic LDA based on medical specialty categories that this sub-topic would somewhat match up with the Orthopedic specialty.

However, in my opinion, the specialties do not really get split into sub-topics as effectively as they could. This is likely because there could be a lot of overlap in the topics. For example, in all of the topic LDAs, the term, “prepped draped” shows up in every sub-topic quite frequently. This potentially makes how the specialties are split into sub-topics a bit more difficult, especially if you have too much overlap between terms.

Nevertheless, we do see that the specialties are further split into more detailed sub-topics when changing from a 5-topic LDA to a 6-topic LDA, as the 6-topic LDA seems to allow or help to differentiate some of the terms that were grouped together in the 5-topic LDA. Using more specific terms in the 6-topic LDA, thus, helping provide more detail and distinctions from the transcriptions by splitting broader topics into more precise sub-topics. For example, sub-topic 2 in the 6-topic LDA is more focused on things involving the foot (e.g., “metatarsal head”). The terms related to lower extremities and orthopedic procedures, such as “proximal phalanx” and “ankle tourniquet” in the 5-topic LDA are now more finely tuned and focused on detailed anatomical terminology (e.g., “metatarsophalangeal joint” and “metatarsal head”). Other examples include sub-topic 5 in the 6-topic LDA, which introduces topics like, “vicryl suture,” which is more indicative of a specific type of technique or procedure that involves suturing, and sub-topic 3 in the 6-topic LDA, which involves terms, such as “coronary artery” and “anterior descending” which seem to be more cardiac-related terminology.

Credits

Examples draw heavily on material (and directly quotes/copies text) from Julia Slige's `tidytext` textbook.