TidyText: Reviewing in a set of Clarisse Lispector Novels

Rodrigo Esteves de Lima-Lopes
State University of Campinas
rll307@unicamp.br

Contents

1 Introduction

1.1 Purpose of this notebook

In this post I am going to discuss some strategies of comparison between texts. It was produced in order to assist colleagues who work in the area of Corpus Linguistics and Systemic Functional Linguistics, as a way to use R in their research. It is part of my CNPq-funded project and seeks to make corpus tools and network analysis accessible. If you have any doubts or wish to make any research contact please send me an email.

This document is based mostly in the package Tidytext and in the following book:

Silge, Julia and David Robinson. 2017. *Text Mining with R: A Tidy Approach*. First edition. Beijing/Boston: O'Reilly.

1.2 Clarisse Lispector

In this notebook I am going to compare two novels by Clarisse Lispector: A Hora da Estrela and A Paixão segundo GH in their original Portuguese version. Some more information about Lispector might be found here.

2 Loading R packages

In this article we are going to use the following packages:

```
library(tidytext) #OK
library(dplyr) #OK
library(tidyverse)
library(tidyr)
library(stringr)
# library(ggraph) Only one function in the code
# # library(igraph) Only one function in the code
# library(textreadr) only one function in the code
# library(abjutils) only one command inside functions
# library(textclean) only one command inside functions
# library(readr) only one function in the code
# library(scales) Only one function in the code
# library(ggplot2) Only one function in the code
```

Each of these packages has a specific function:

1. `tidytext`: Text manipulation
2. `dplyr`: Data manipulation, such as tables and data frames
3. `stringr`: Manipulation of strings
4. `tidyverse`: Text manipulation, it loads some functions from other packages
5. `tidyr`: Data manipulation, such as tables and data frames

Since we are going to use just some functions, these packages are not going to be loaded, we are going to call their functions at the right moment:

1. `readr` and `textreadr`: Read documents and data into R
2. `scales`: Data representation
3. `abjutils`: Data manipulation for Brazilian attorneys of law
4. `textclean`: Clean texts
5. `ggraph and igraph`: Preparing and plotting network graphs
6. `ggplot2`: Doing some common plots.

3 Loading data

3.1 Data and stop-words

Since my focus here is on the semantically relevant Lexis, I will load a list of stop words, in order to filter prepositions, articles and conjunctions. There are a number of stop-words list for Portuguese, I would advise you to prepare your own based on your research objectives. The code bellow also loads the two novels in R.

```
My.Stopwords <- data.frame(
  readr::read_csv("stop_port2.csv", col_names = FALSE),stringsAsFactors = FALSE
  )
```

```
## Rows: 559 Columns: 1

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): X1

##
## 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.
```

```
colnames(My.Stopwords) <- "word"
```

```
head(My.Stopwords,10)
```

```
##       word
## 1       tá
## 2        a
## 3   acerca
## 4    adeus
## 5    agora
## 6    ainda
## 7     alem
## 8  algumas
## 9     algo
## 10 algumas
```

Now we are going to load the novels:

```
estrela <- textreadr::read_document("estrela.txt")
```

```
paixao <- textreadr::read_document("paixao.txt")
```

3.2 Cleaning data

Our first step is going to clean our textual data. For this study we will:

1. Remove diacritics

- This is not always necessary, it will depend of our objectives

1. Replace possible `HTML` codes
2. Change words to lower case
3. Delete numerals
4. Delete extra spaces and extra lines

To clean the data we going to import a previously function I have developed earlier. This function is in another `R` script, called `05_TidyText_functions.R`. We going to load it using the following command:

```
source("05_TidyText_functions.R")
```

4 Analysis

4.1 Create and compare a wordlist

Our fist step will be processing the two novels in a set of words with their frequencies.

```
estrela <- data.frame(text = estrela, stringsAsFactors = F)
```

```
paixao.tidy <- analise.paixao  |>
  unnest_tokens(word, text) |>
  anti_join(My.Stopwords, by = "word")
```

```
paixao <- data.frame(text = paixao, stringsAsFactors = F)
```

```
paixao.tidy <- paixao  |>
  unnest_tokens(word, text) |>
  anti_join(My.Stopwords, by = "word")
```

The code above creates a data frame from each novel. It is important to make the files processable by R. Our next step is to apply our customised function (`Clean_String`) to the files. Note that we keep the data frame format.

```
analise.paixao <- Clean_String(paixao$text) |>
  data.frame()
colnames(analise.paixao) <- 'text'

analise.estrela <- Clean_String(estrela$text) |>
  data.frame()
colnames(analise.estrela) <- 'text'
```

Now our next step is prepare the wordlist. The following lines process the data frame in order to:

- Make it in a word per line forma
- Filter stop-words out
- Count and present the wordlist

```
paixao.tidy <- analise.paixao  |>
  unnest_tokens(word, text) |>
  anti_join(My.Stopwords, by = "word")

estrela.tidy <- analise.estrela |>
  unnest_tokens(word, text) |>
  anti_join(My.Stopwords, by = "word")
head(paixao.tidy)
```

```
##         word
## 1 procurando
## 2 procurando
## 3   tentando
## 4   entender
## 5   tentando
## 6     alguem
```

```
head(estrela.tidy)
```

```
##        word
## 1   verdade
## 2   clarice
## 3 lispector
## 4    dedico
## 5        ai
## 6    antigo
```

As we can see, there is a column which displays a sequential number, representing the line of the list, and another which displays the words itself. However it is also possible to observe that *procurando* is present twice in the list. It happens because we are not counting occurrences, but only displaying the novel's words.

```
paixao.l<- paixao.tidy |>
  count(word, sort = TRUE)
```

```
esterla.l <- estrela.tidy |>
  count(word, sort = TRUE)
```

The code above transforms these lists in something similar to a word list. The first line creates the file itself, which I chose to name with the suffix "*.l". The operator " |>" is part of the tidyverse, its function is to help us to "pass" one line of code through to the other and perform more than a command at once. The following code counts the words and creates a new column with the results.

```
head(paixao.l, 10)
```

```
##      word   n
## 1     mim 227
## 2      so 195
## 3    vida 190
## 4      ja 185
## 5  barata 180
## 6     tao 131
## 7    amor 111
## 8    medo  87
## 9   mundo  87
## 10    ate  86
```

```
head(esterla.l, 10)
```

```
##        word  n
## 1        so 96
```

```
## 2   macabea 91
## 3      vida 83
## 4      voce 79
## 5       ate 67
## 6        ja 60
## 7      moca 52
## 8  historia 51
## 9       tao 49
## 10      mim 48
```

Our next step is to build a table comparing the frequency of such words. It is necessary in order to generate the visualisations and make the analysis possible. In the first line we create a new object and add two columns in it, one for each book, and make sure that the list of words identify to the book where it is from. I the following line we clear the text, keeping only alphabetical words and deleting numbers and special symbols. We group the words by book (*livro* in Portuguese) and calculate the relative proportion of the words within each novel. Since absolute values do not mean much, we delete them.

```
frequencia.clarisse <- bind_rows(mutate(paixao.tidy, livro = "P"),
                                 mutate(estrela.tidy, livro = "H"))  |>
  mutate(word = str_extract(word, "[a-z']+"))  |>
  count(livro, word)  |>
  group_by(livro)  |>
  mutate(proportion = (n / sum(n))*100)  |>
  select(-n)  |>
  spread(livro, proportion)
head(frequencia.clarisse)
```

```
## # A tibble: 6 x 3
##   word            H        P
##   <chr>       <dbl>    <dbl>
## 1 aaaar      0.0111 NA
## 2 abafada    0.0111 NA
## 3 abafado   NA       0.00591
## 4 abafar    NA       0.00591
## 5 abafavaa   0.0111 NA
## 6 abaixando NA       0.0118
```

It was possible to observe that the word **aaaar** is present in the top of the list. If we want to clean it out, we have to add it to our stop wordlist:

```
My.Stopwords <- data.frame(word = c(My.Stopwords$word,"aaaar"))
```

Then we rerun the four pieces of code above.

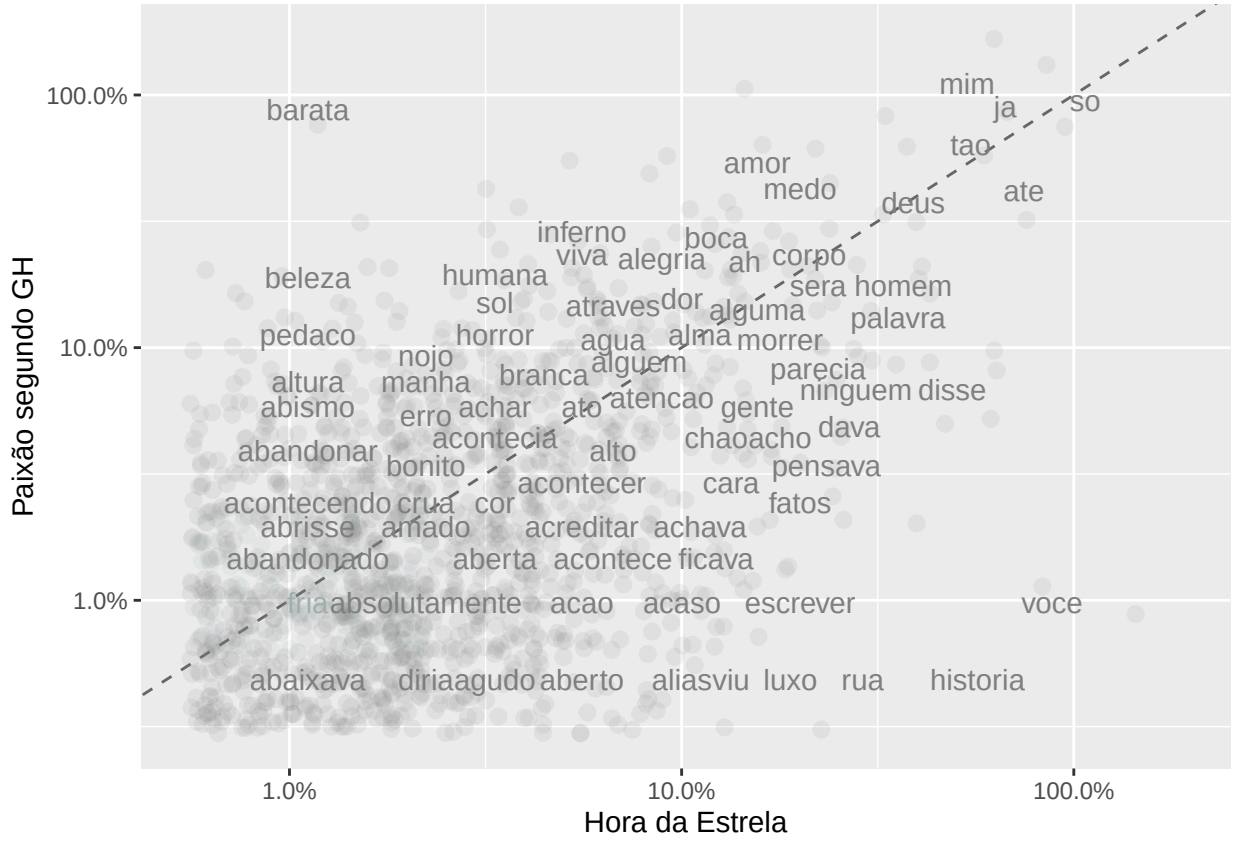
4.1.1 Visualising our wordlist

Now it is time to compare the Lexis in the two books. We will do so by using a ggplot command. Mostly we are putting both novels side by side, making them overlap. The position is calculated my a log scale that helps to represent the overlapping. The more central a word is in the graphic, the less "specific" of a book it is.

```
ggplot(frequencia.clarisse, aes(x = H, y = P,
                                color = abs(H - P))) +
  geom_abline(color = "gray40", lty = 2) +
  geom_jitter(alpha = 0.1, size = 2.5, width = 0.3, height = 0.3) +
  geom_text(aes(label = word), check_overlap = TRUE, vjust = 1.5) +
  scale_x_log10(labels = scales::percent_format()) +
  scale_y_log10(labels = scales::percent_format()) +
  scale_color_gradient(limits = c(0, 0.001),
                       low = "darkslategray4", high = "gray75") +
  theme(legend.position = "none") +
  labs(y = "Paixão segundo GH", x = "Hora da Estrela")
```

```
## Warning: Removed 6253 rows containing missing values (geom_point).
```

```
## Warning: Removed 6253 rows containing missing values (geom_text).
```



4.2 Analysing bigrams

In order to do a complete bigrams analysis. We will do some changes to the files. First we add a column to each and then make those a single data frame. Note that the book column will become a `factor`

```
# Adding a column with book titles
analise.estrela$book <- "Hora da Estrela"
```

```
analise.paixao$book <- "Paixao segundo GH"
# Joining
clarisse.livros <- rbind(analise.estrela,analise.paixao)
clarisse.livros$book <- as.factor(clarisse.livros$book)
head(clarisse.livros)
```

```
##                                                 text           book
## 1                          na verdade clarice lispector Hora da Estrela
## 2   pois que dedico esta coisa ai ao antigo schumann e Hora da Estrela
## 3          sua doce clara que sao hoje ossos ai de nos Hora da Estrela
## 4      dedicome a cor rubra muito escarlate como o meu Hora da Estrela
## 5 sangue de homem em plena idade e portanto dedicome a Hora da Estrela
## 6       meu sangue dedicome sobretudo aos gnomos anoes Hora da Estrela
```

Ou next step is to create the bigrams itself.

```
# Creating bigrams
clarisse.bigrams <- clarisse.livros |>
  unnest_tokens(bigram, text, token = "ngrams", n = 2)
bigrams.count <- clarisse.bigrams |>
  count(bigram, sort = TRUE)
head(clarisse.bigrams)
```

```
##              book            bigram
## 1 Hora da Estrela        na verdade
## 2 Hora da Estrela   verdade clarice
## 3 Hora da Estrela clarice lispector
## 4 Hora da Estrela          pois que
## 5 Hora da Estrela        que dedico
## 6 Hora da Estrela       dedico esta
```

```
head(bigrams.count)
```

```
##    bigram   n
## 1   o que 394
## 2  que eu 269
## 3    <NA> 254
## 4   e que 248
## 5     e o 173
## 6 que nao 165
```

In the first line, the command `unnest_tokens` creates a column which the bigrams found, taking from the column `text`, the `n=2` tells how long my ngram is supposed to be. In the second line, it creates a data frame telling how many of each are. The command `count` does it for me.

Ou next step is to separate the words, these will be important for our network analysis. Note that the command is separating two columns using a single space as a criterion.

```
clarisse.separated <- clarisse.bigrams |>
  separate(bigram, c("word1", "word2"), sep = " ")
head(clarisse.separated)
```

```
##              book   word1     word2
## 1 Hora da Estrela      na   verdade
## 2 Hora da Estrela verdade   clarice
## 3 Hora da Estrela clarice lispector
## 4 Hora da Estrela    pois       que
## 5 Hora da Estrela     que    dedico
## 6 Hora da Estrela  dedico      esta
```

Since we will also to analyse each novel separately, we will make the same procedure for each

```
hora.count <- clarisse.bigrams |>
  subset(book == "Hora da Estrela") |>
  count(bigram, sort = TRUE) |>
  separate(bigram, c("word1", "word2"), sep = " ")
hora.count <- hora.count[-3,]

paixao.count <- clarisse.bigrams |>
  subset(book == "Paixao segundo GH") |>
  count(bigram, sort = TRUE) |>
  separate(bigram, c("word1", "word2"), sep = " ")
paixao.count <- paixao.count[-3,]

head(hora.count)
```

```
##   word1 word2   n
## 1     o   que 109
## 2     e   que  88
## 4   que   nao  48
## 5   que    se  44
## 6   que     e  42
## 7   que     a  38
```

```
head(paixao.count)
```

```
##   word1 word2   n
## 1     o   que 285
## 2   que    eu 240
## 4     e   que 160
## 5     e     o 143
## 6     e     a 139
## 7    eu   nao 123
```

Now we are going to clean the stopwords and see the difference

```
clarisse.filtered <- clarisse.separated |>
  filter(!word1 %in% My.Stopwords$word) |>
  filter(!word2 %in% My.Stopwords$word)
head(clarisse.filtered)
```

```
##              book   word1     word2
## 1 Hora da Estrela verdade   clarice
## 2 Hora da Estrela clarice lispector
```

```
## 3 Hora da Estrela  antigo  schumann
## 4 Hora da Estrela    doce     clara
## 5 Hora da Estrela   ossos        ai
## 6 Hora da Estrela     cor     rubra
```

Now we count the filtered bigrams and have a look at it

```
bigrams.count <- clarisse.filtered |>
  count(word1, word2, sort = TRUE)
head(bigrams.count)
```

```
##     word1   word2   n
## 1    <NA>    <NA> 254
## 2   algum    modo   8
## 3  madama carlota   8
## 4  coisas     sao   7
## 5 materia    viva   7
## 6 propria    vida   6
```

4.2.1 Going forward

Now we will analyse the relevance of each bigrams in the novel

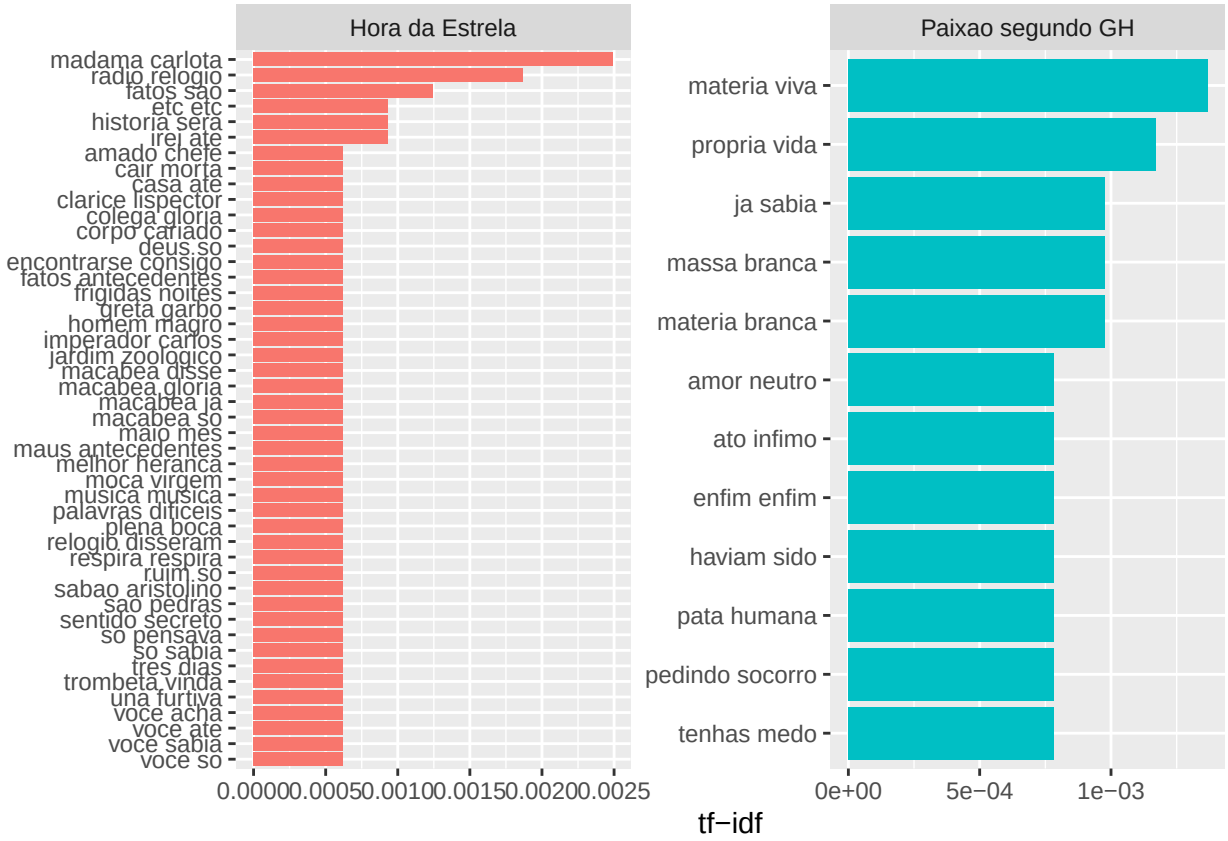
```
clarisse.tf_idf <- clarisse.united %>%
  count(book, bigram) %>%
  bind_tf_idf(bigram, book, n) %>%
  arrange(desc(tf_idf))
head(clarisse.tf_idf)
```

```
##                book         bigram n          tf       idf       tf_idf
## 1   Hora da Estrela madama carlota 8 0.003592277 0.6931472 0.0024899764
## 2   Hora da Estrela  radio relogio 6 0.002694207 0.6931472 0.0018674823
## 3 Paixao segundo GH   materia viva 7 0.001972387 0.6931472 0.0013671542
## 4   Hora da Estrela      fatos sao 4 0.001796138 0.6931472 0.0012449882
## 5 Paixao segundo GH   propria vida 6 0.001690617 0.6931472 0.0011718465
## 6 Paixao segundo GH       ja sabia 5 0.001408848 0.6931472 0.0009765387
```

Now we are going to plot these frequencies

```
## Ploting the relative frequency
clarisse.tf_idf  |>
  arrange(desc(tf_idf)) |>
  mutate(bigram = factor(bigram, levels = rev(unique(bigram)))) |>
  group_by(book) |>
  top_n(10) |>
  ungroup() |>
  ggplot2::ggplot(aes(bigram, tf_idf, fill = book)) +
  geom_col(show.legend = FALSE) +
  labs(x = NULL, y = "tf-idf") +
  facet_wrap(~book, ncol = 2, scales = "free") +
  coord_flip()
```

```
## Selecting by tf_idf
```

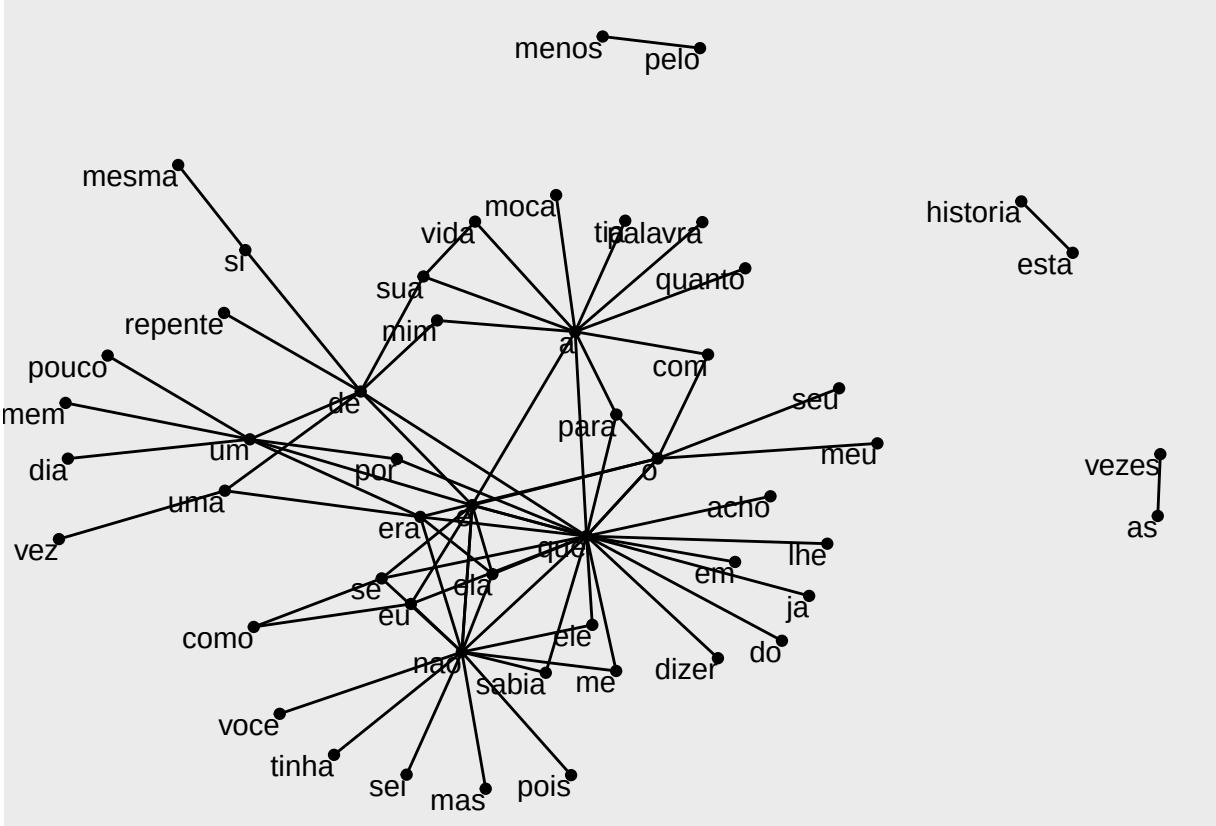


Our next step is to prepare and plot a couple of network graphics. Fist each novel individually:

```
set.seed(1973)
#Selecting the words and creating the interaction file
# Hora da estrela
hora.graph <- hora.count %>%
  filter(n > 10) %>%
  igraph::graph_from_data_frame()
hora.graph
```

```
## IGRAPH fe531bf DN-- 52 78 --
## + attr: name (v/c), n (e/n)
## + edges from fe531bf (vertex names):
##  [1] o   ->que   e   ->que   que ->nao   que ->se    que ->e     que ->a
##  [7] que ->ela   de  ->uma   nao ->e     e   ->o     que ->eu    que ->o
## [13] e   ->um    eu  ->nao   nao ->sabia com ->a     nao ->se    e   ->nao
## [19] ela ->nao   nao ->sei   a   ->moca  como->se    de  ->mim   e   ->a
## [25] mas ->nao   por ->que   que ->me    a   ->vida  de  ->que   ela ->era
## [31] si  ->mesma um  ->dia   um  ->pouco a   ->sua   que ->era   que ->ele
## [37] sua ->vida  as  ->vezes em  ->que   com ->o     ela ->e     era ->um
## + ... omitted several edges
```

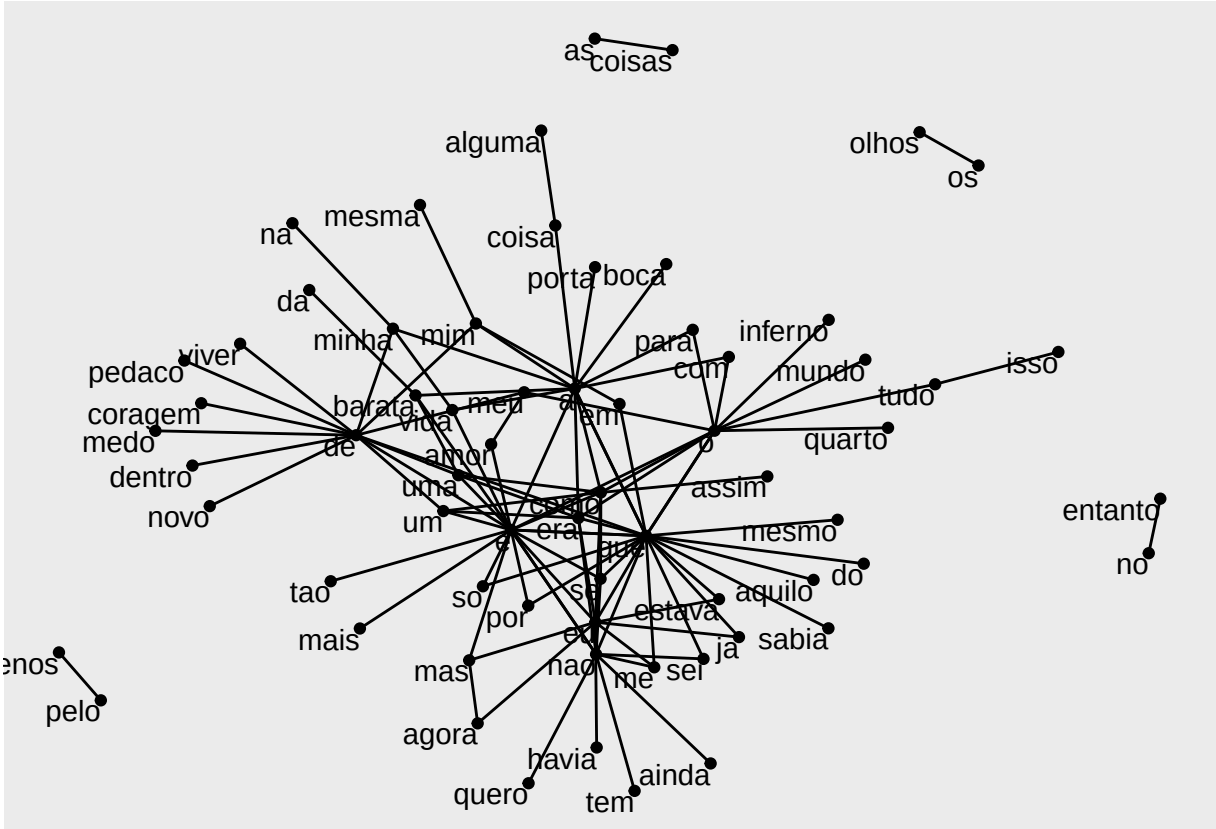
```
ggraph::ggraph(hora.graph, layout = "fr") +
  ggraph::geom_edge_link() +
  ggraph::geom_node_point() +
  ggraph::geom_node_text(aes(label = name),
                         vjust = 1, hjust = 1)
```




```
set.seed(1973)
paixao.graph <- paixao.count %>%
  filter(n > 20) %>%
  igraph::graph_from_data_frame()
print(paixao.graph)
```

```
## IGRAPH 3f3c195 DN-- 66 109 --
## + attr: name (v/c), n (e/n)
## + edges from 3f3c195 (vertex names):
##  [1] o    ->que    que  ->eu     e    ->que    e    ->o      e    ->a
##  [6] eu   ->nao    que  ->nao    a    ->minha  que  ->e      de   ->um
## [11] de   ->uma    que  ->me     que  ->se     de   ->mim    que  ->a
## [16] do   ->que    eu   ->estava eu   ->me     a    ->barata como ->se
## [21] de   ->minha  se   ->eu     nao  ->e      o    ->meu    em   ->mim
## [26] e    ->eu     que  ->o      e    ->nao    e    ->um     nao  ->me
## [31] de   ->que    a    ->vida   era  ->o      com  ->o      da   ->barata
## [36] e    ->de     com  ->a      que  ->era    eu   ->ja     assim->como
## + ... omitted several edges
```

```
ggraph::ggraph(paixao.graph, layout = "fr") +
  ggraph::geom_edge_link() +
  ggraph::geom_node_point() +
  ggraph::geom_node_text(aes(label = name), vjust = 1, hjust = 1)
```



Now doing both books together

```
set.seed(1973)
# Getting the filtered bigrams
clarisse.g <- clarisse.filtered[,2:3] |>
  count(word1, word2, sort = TRUE)
clarisse.g <- clarisse.g[-1,]
head(clarisse.g)
```

```
##     word1   word2 n
## 2   algum    modo 8
## 3  madama carlota 8
## 4  coisas     sao 7
## 5 materia    viva 7
## 6 propria    vida 6
## 7   radio relogio 6
```

```
general.graph <- clarisse.g |>
  filter(n > 4) |>
```

```
  igraph::graph_from_data_frame()
print(general.graph)
```

```
## IGRAPH f7102b4 DN-- 16 9 --
## + attr: name (v/c), n (e/n)
## + edges from f7102b4 (vertex names):
## [1] algum  ->modo    madama ->carlota coisas ->sao     materia->viva
## [5] propria->vida    radio  ->relogio ja     ->sabia   massa  ->branca
## [9] materia->branca
```

```
ggraph::ggraph(general.graph, layout = "fr") +
  ggraph::geom_edge_link() +
  ggraph::geom_node_point() +
  ggraph::geom_node_text(aes(label = name), vjust = 1, hjust = 1)
```