Práctica PLN

Grupo 6

Javier Pérez Vargas, javier.perez.vargas@alumnos.upm.es
Gonzalo Lope Carrasco, g.lope@alumnos.upm.es
Mario Ruiz Vazquett, mario.ruiz@alumnos.upm.es
Carlos Ramos Nieto, carlos.ramos.nieto@alumnos.upm.es

22-12-2022

Introducción

La práctica que realizamos en nuestro grupo consistía en crear un buscador de enfermedades. Dado una noticia, documento o corpus, el buscador debía detectar cada enfermedad que apareciera en base a sus afijos, y clasificarlas por su tipo: Infección, Dolor, Alteración o Cáncer.

Proceso de ejecución

Para comenzar, se carga el paquete rjson y se verifica si ya está instalado. Si no está instalado, se instala. Esto se hace porque rjson es un paquete que se utiliza para leer y escribir archivos JSON en R. Lo utilizaremos para extraer el texto de ejemplo con el que trabajaremos.

```
if (! require('rjson')){
  !install.packages('rjson');
}
```

```
## Loading required package: rjson
```

```
library('rjson');
```

Luego, se lee el archivo JSON y se crea un dataframe llamado 'df' con dos columnas: 'ids' y 'texto'. El archivo JSON es un archivo de texto que contiene información estructurada en un formato específico, y rjson se utiliza para leer este archivo y convertirlo en un dataframe de R.

```
f = fromJSON(file = 'reduced_article_list3.json');
ids = unlist(lapply(f$articles, function (x) x$id ));
texto = unlist(lapply(f$articles, function (x) x$abstractText ));
df = data.frame(ids,texto)
```

A continuación, se carga el modelo de lenguaje 'spanish-ancora-ud-2.5-191206.udpipe' del paquete udpipe y se guarda en una variable llamada 'udmodel_es'.

```
library(udpipe)
# udpipe_download_model(language = "spanish-ancora") #"spanish-ancora" or "spanish-gsd"
# Descarga "spanish-ancora-ud-2.5-191206.udpipe"
udmodel_es<-udpipe_load_model(file = 'spanish-ancora-ud-2.5-191206.udpipe');
```

Luego, se utiliza el modelo de lenguaje cargado para analizar el texto de cada artículo en 'df$texto' en busca de palabras que terminen en "itis", "oma", "algia", o empiecen con "hipo" o "hiper". Buscaremos los sintagmas nominales enteros viendo el tipo de palabra que sigue a las encontradas. Para hacer esto, se recorre el dataframe 'df' y se utiliza la función 'udpipe_annotate' del paquete udpipe para analizar el texto de cada artículo. Esta función toma el texto como entrada y devuelve una serie de información sobre cada palabra del texto, como su categoría gramatical (por ejemplo, sustantivo, verbo, etc.) y su relación con las demás palabras del texto. Luego, se buscan las palabras que cumplen con las condiciones mencionadas y se añaden a la lista 'enfermedades'.

```
# Modo texto a texto
library(stringr)
enfermedades = c();
for (texto in df$texto){
```

```
  texto_analizado = as.data.frame(udpipe_annotate(udmodel_es,texto));
  posibles_enfermedades = str_detect(texto_analizado$token,
                                     regex('.*?itis$|.*?oma$|.*?algia$|^hipo.*|^hiper.*'))
  for (j in 1:nrow(texto_analizado)){
    if (texto_analizado$upos[[j]] == 'NOUN' & posibles_enfermedades[[j]]){
      if (texto_analizado$dep_rel[[j+1]] == 'amod'){
        enfermedades = c(enfermedades,paste(texto_analizado$token[[j]],
                                            texto_analizado$token[[j+1]],sep = " ") )
      }else{
        enfermedades = c(enfermedades,texto_analizado$token[[j]])
      }
    }
  }
}
enfermedades = unique(enfermedades)
enfermedades = sort(enfermedades)
enfermedades
```

```
##  [1] "adenocarcinoma"                  "amigdalitis aguda"
##  [3] "amigdalitis pultácea"            "apendicitis aguda"
##  [5] "artritis periférica"             "cistoadenoma mucinoso"
##  [7] "colitis inespecífica"            "colitis pseudomembranosa"
##  [9] "cromosoma"                       "dermatofibrosarcoma protuberante"

...

## [61] "pancreatitis lúpica"             "papiloma humano"
## [63] "periodontitis"                   "piomiositis aguda"
## [65] "pulpitis irreversible"           "schwannoma"
## [67] "schwannoma intraparotídeo"       "sinusitis crónicas"
## [69] "toma"
```

Posteriormente, separamos las palabras compuestas y nos quedamos con el núcleo del sintagma nominal (por ejemplo, de carcinoma ductal nos quedamos con carcinoma) para que el diccionario pueda buscarlos y los guardamos en la variable enfermedades2.

```
enfermedades2 = c()
for (i in 1:length(enfermedades)){
  enfermedades2[i] = str_extract(enfermedades[i], "^\\w+")
  if (str_detect(enfermedades2[i], "(?<!i)s$") == TRUE){
  enfermedades2[i] = gsub("s$", "", enfermedades2[i])
}
  }
enfermedades2 = unique(tolower(enfermedades2))
enfermedades2
```

```
##  [1] "adenocarcinoma"       "amigdalitis"          "apendicitis"
##  [4] "artritis"             "cistoadenoma"         "colitis"
##  [7] "cromosoma"            "dermatofibrosarcoma"  "enterocolitis"
## [10] "entesitis"            "estomatitis"          "gastroenteritis"
## [13] "gingivitis"           "granuloma"            "hepatitis"
## [16] "hiperactividad"       "hiperamonemia"        "hiperextensibilidad"
## [19] "hipermetilación"      "hipermovilidad"       "hiperplasia"
```

```
## [22] "hiperpolimenorrea"    "hipertensión"         "hipertirotropinemia"
## [25] "hipertrigliceridemia" "hipertrofia"          "hipoacusia"
## [28] "hipoclorito"          "hipocondrio"          "hipokalemia"
## [31] "hipoplasia"           "hipoproteinemia"      "hipotiroidismo"
## [34] "hipotonía"            "hipoxia"              "idioma"
## [37] "leiomiosarcoma"       "linfoma"              "neurofibroma"
## [40] "otitis"               "pancreatitis"         "papiloma"
## [43] "periodontitis"        "piomiositis"          "pulpitis"
## [46] "schwannoma"           "sinusitis"            "toma"
```

Una vez creada esta variable, buscamos todas las enfermedades en el diccionario médico con el siguiente formato:

`https://www.cun.es/diccionario-medico/terminos/{CONCEPTO}`

Las que encontramos en dicho diccionario las añadimos a la variable enfermedades3. Las que no se encuentren, se asume que no son enfermedades y las desechamos. Nos quedaremos únicamente con aquellas que aparezcan en el diccionario médico.

```
library(stringi)
urls = c()
valid_urls = c()
enfermedades3 = c()
for (i in 1:length(enfermedades2)){
  url = "https://www.cun.es/diccionario-medico/terminos/"
  urlOK = paste(url, enfermedades2[i])
  urlOK = stri_replace_all_regex(urlOK, c(" ", "á", "é", "í", "ó", "ú"),
                                 c("", "a", "e", "i", "o", "u"), vectorize_all = FALSE)
  urls[i] = urlOK
  tryCatch(
  {
    lines = readLines(con=urlOK, warn = FALSE)
    enfermedades3 = c(enfermedades3, enfermedades2[i])
    valid_urls = c(valid_urls, urlOK)
  },
  error=function(cond){
    message(paste("URL no existe:", urlOK))
    return(NA)
  }
)
}
enfermedades3 = unique(enfermedades3)
enfermedades3
```

```
##  [1] "adenocarcinoma"       "amigdalitis"          "apendicitis"
##  [4] "artritis"             "colitis"              "cromosoma"
##  [7] "enterocolitis"        "gingivitis"           "granuloma"
## [10] "hepatitis"            "hiperactividad"       "hiperamonemia"
## [13] "hipermovilidad"       "hiperplasia"          "hipertrigliceridemia"
## [16] "hipertrofia"          "hipoacusia"           "hipocondrio"
## [19] "hipoplasia"           "hipoproteinemia"      "hipotiroidismo"
## [22] "hipotonía"            "hipoxia"              "linfoma"
```

```
## [25] "neurofibroma"         "otitis"               "pancreatitis"
## [28] "papiloma"             "periodontitis"        "pulpitis"
## [31] "schwannoma"           "sinusitis"
```

Tras esto, creamos la forma invariable de las enfermedades eliminando los prefijos y sufijos que hemos buscado anteriormente. Para eliminarlos utilizamos la función *stri_replace_all_regex*.

```
forma_invariable <- stri_replace_all_regex(enfermedades3,
                                           "hipo|hiper|itis|algia|algia|oma","")
```

Terminado el proceso de lematización, nos planteamos generar un documento en el que aparecería la definición de cada enfermedad encontrada. En este punto contamos con gran parte de la información necesaria.

Previamente, al hacer el cambio enfermedades2 => enfermedades3, tuvimos que hacer consultas al diccionario ya citado. Estas consultas eran precisamente para ver si la página en la que encontraríamos la definición existe o no, por lo que nos conviene tener un vector en el que almacene las url válidas (valid_urls).

Indagamos en algunos ejemplos el código fuente de las urls, y nos damos cuenta de que siempre localizamos la definición en la línea 891 de la página, por lo que hacemos un bucle sobre valid_urls para añadir iterativamente a un nuevo vector todas las definiciones. Nótese que antes de añadirlas a este nuevo vector, limpiamos todos los elementos html.

```
limpiaBloquesConTrim <- function(vec_cadenas){
a <- gsub("<[^<>]*>", "", vec_cadenas)
trimws(a)}
reemplazaElemsHTML <- function(vec_cadenas){
traduc <- list(c("&aacute;", "á"),
               c("&Aacute;", "Á"),
               c("&eacute;", "e"),
               c("&Eacute;", "É"),
               c("&iacute;", "í"),
               c("&Iacute;", "Í"),
               c("&oacute;", "ó"),
               c("&Oacute;", "Ó"),
               c("&uacute;", "ú"),
               c("&Uacute;", "Ú"),
               c("&ntilde;", "ñ"),
               c("&Ntilde;", "Ñ"),
               c("&iquest;", "¿"),
               c("&iexcl;", "¡"),
               c("&laquo;", "«"),
               c("&raquo;", "»"))
stri_replace_all_regex(vec_cadenas,
                       pattern=unlist(lapply(traduc, function(x){x[1]})),
                       replacement=unlist(lapply(traduc, function(x){x[2]})),
                       vectorize=FALSE)
}
url_base = "https://www.cun.es/diccionario-medico/terminos/"
definiciones = c()
for (i in 1:length(valid_urls)){
  lines = readLines(valid_urls[i],
                    encoding = "UTF-8")
  definicion = lines[891]
  definicion <- limpiaBloquesConTrim(lines[891])
```

```
  definicionOK <- reemplazaElemsHTML(definicion)
  definiciones = c(definiciones, definicionOK)
}
names(definiciones) = enfermedades3
```

Llegados a este punto contamos con 3 vectores:
- enfermedades3, que contiene todas las enfermedades.
- valid_urls, con las url de consulta del diccionario.
- definiciones, que contiene las definiciones en cadenas de texto.

Claramente, tienen la misma longitud y, observamos que, dada una posición i dentro del rango de los vectores, enfermedades3[i] y definiciones[i] harán alusión al mismo concepto, por lo que le damos el nombre de la enfermedad a cada definición (names(definiciones) = enfermedades3).

Con estos 3 vectores podemos crear nuestro archivo de texto que contenga las enfermedades encontradas junto a sus definiciones. Para ello, creamos un archivo .txt al que llamaremos "Definiciones.txt", sobre el que añadiremos cada una de las palabras utilizando la función *writeLines*.

Para detallar más cada enfermedad, añadiremos la siguiente información:

- En primer lugar, se indicará que tipo de enfermedad es (Infección, Cáncer, Dolor o Alteración). Para realizar esta tarea hemos utilizado un vector con los afijos de cada tipo, cuyos nombres (names) serán el tipo de enfermedad. Así, utilizando regex hemos podido asignar a cada enfermedad su tipo.
- También queríamos añadir las diferentes variantes de cada enfermedad (por ejemplo, debajo de la definición de carcinoma, añadiremos sus variantes encontradas: carcinoma ductal, carcinoma escamoso, etc... Y lo hemos conseguido, de nuevo, con la ayuda de expresiones regulares que buscan el núcleo de los sintagmas nominales.
- Por último, la forma invariable de cada enfermedad.

```
con1 <- file(
  description = "Definiciones.txt",
  open = "wt",
  encoding = "UTF-8")
dict_enf = c("algia", "hipo", "hiper", "itis", "oma")
names(dict_enf) = c("Dolor", "Alteración", "Alteración", "Infección", "Cáncer")
for (i in 1:length(definiciones)){
  definicionFinal = paste(names(definiciones[i]), ": ", definiciones[i], sep="")
  writeLines(definicionFinal, con1)
  for (j in 1:5){
    tipo = unlist(str_extract_all(names(definiciones[i]), dict_enf[j]))
    if (!(is.null(names(which(dict_enf==tipo))))){
      tipo = names(which(dict_enf==tipo))
      writeLines(paste("Tipo:", tipo), con1)
      break
    }
  }
  lema = paste("Forma invariable: ", forma_invariable[i], sep="")
  writeLines(lema, con1)
  variaciones = unlist(str_extract_all(enfermedades, paste(names(definiciones[i]), ".*")))
  if (length(variaciones) > 1){
    variaciones = unique(variaciones)
    variaciones = paste("Variaciones:", paste(as.character(variaciones), collapse = ", "))
    writeLines(variaciones, con1)
```

```
  }
  writeLines("\n", con1)
}
close(con1)
```

Ejemplo resultado

```
lines = readLines("Definiciones.txt")
lines[lines != ""]
```

```
## [1] "adenocarcinoma: f. Lesión tumoral maligna de naturaleza epitelial, con formación de
estructuras glandulares reconocibles u originadas a partir de un epitelio glandular. Al igual que
el adenoma (benigno), existen varios tipos según la estructura dominante."
## [2] "Tipo: Cáncer"
## [3] "Forma invariable: adenocarcin"
## [4] "amigdalitis: f. Inflamación aguda o crónica de las amígdalas palatinas o linguales."
## [5] "Tipo: Infección"

...

## [99] "Tipo: Cáncer"
## [100] "Forma invariable: schwann"
## [101] "sinusitis: f. Inflamación de los senos óseos paranasales. En adultos el seno más
frecuentemente afectado es el maxilar y en el niño las celdas etmoidales. Cuando hay varios senos
afectados se habla de polisinusitis, y si están afectados todos, pansinusitis. Puede ser uni o
bilateral. &gt; Saber más sobre la sinusitis"
## [102] "Tipo: Infección"
## [103] "Forma invariable: sinus"
```

Problemas o dificultades

La primera dificultad que nos surgió fue el detectar si las palabras encontradas con los afijos indicados eran enfermedades, o simplemente otras palabras que contenían esos afijos. Nuestra primera solución fue la búsqueda en el diccionario, y aunque los resultados fueron bastante buenos, seguía habiendo palabras como *síntoma* que aparecían en el diccionario y contenía uno de los afijos. A pesar de ello, la gran mayoría de enfermedades se podían detectar de esta forma.

Otra de las dificultades fue encontrar el sintagma nominal entero de cada enfermedad. Intentamos usar funciones que buscaran sintagmas nominales, pero ninguna nos dio un resultado convincente. Por tanto, aplicando la librería udpipe conseguimos detectar los modificadores (si es que los había) después de cada enfermedad. Así, enfermedades como *carcinoma ductal* pudieron ser detectadas de forma completa y no solo con la primera palabra.

Reparto del trabajo

- Gonzalo Lope: Se ha encargado de conseguir todos los sintagmas nominales que son enfermedades y ha realizado la memoria en conjunto con Carlos.

- Javier Pérez: Se ha hecho cargo de la comprobación de que las enfermedades aparecen en el diccionario y también se encargó de la creación del documento txt con los resultados.
- Mario Ruiz: Se ha encargado de realizar todas las búsquedas mediante regex de las posibles enfermedades y ha conseguido la forma invariable de todas las enfermedades.
- Carlos Ramos: Se ha hecho cargo de recoger las definiciones api de las enfermedades del diccionario y ha realizado la memoria en conjunto con Gonzalo.

Es destacable como todos nos hemos involucrado en solucionar errores que nos hemos ido encontrando en el proceso de la realización del trabajo y como todos no hemos tenido problemas en ayudarnos siempre que lo hemos necesitado.