PRÁCTICA 3: PREPROCESAMIENTO DE TEXTOS

RECUPERACIÓN DE INFORMACIÓN
LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN
PRIMAVERA 2022

Gustavo Iván Molina Rebolledo
Benemérita Universidad Autónoma de Puebla

Abstract. A rather small and simple preprocessing tool for CCOS 264 (Information retrieval) class. The full code, input and output is included in the project repository: https://github.com/ivanmoreau/prep_textos

01 de febrero de 2022

I. Introducción

En esta práctica el objetivo es obtener un documento limpio a partir de diversos tweets que cumplen con la estructura "User tweeted Tweet". Este programa se debe de apegar a los siguientes requerimientos[1]:

1. Eliminar signos de puntuación
2. Eliminar los usuarios colocando la etiqueta $user.
3. Eliminar los hashtag colocando la etiqueta $ht.
4. Eliminar los emoticones colocando el tipo de emoticón, por ejemplo si es el emoticón es :) colocar la etiqueta SMILE.
5. Pasar todo a minúscula.

EL resultado debe de constar de dos documentos: uno que incluya stopwords y otro que no las incluya.

II. Implementación

Para implementar esta herramienta utilizo el lenguaje de programación Haskell, así las librerías Emojis y Parsec; también se está haciendo uso de un diccionario de stopwords, por simplicidad.

El programa no es más que dos grandes parser combinators, que después pasa por una función que se encarga de limpiar el número excesivo de espacios que puedan haber quedado.

El código por sí mismo es relativamente pequeño y consta de 50LOC.

```
{- |
Copyright: (c) 2022 Ivan Molina Rebolledo
SPDX-License-Identifier: GPL-3.0-only
Maintainer: Ivan Molina Rebolledo <ivanmolinarebolledo@gmail.com>
```

[1] La lista fue copiada directamente de la especificación

```
See README for more info
-}

module PrepTextos
       ( parseWith,
         parseWithoutStopWords,
         parse
       ) where

import Data.Text (Text, pack, concat, replace, toLower, unpack)
import EmojiP (parseEmoji)
import Stopwords (stopwords)
import Text.Parsec ((<|>), many, many1, option, optionMaybe, try)
import Text.Parsec.Char (alphaNum, char, digit, letter, oneOf, string)
import Text.Parsec.Text (Parser, parseFromFile)
import Text.Parsec.Error (ParseError)

user :: Parser Text
user = char '@' >> many1 (alphaNum <|> char '_') >> return (pack "$user")

hashtag :: Parser Text
hashtag = char '#' >> many1 (alphaNum <|> char '_') >> return (pack " $ht ")

tweeted :: Parser Text
tweeted = string "tweeted:" >> return (pack "tweeted")

spaces :: Parser Text
spaces = many1 (char ' ') >> return (pack " ")
```

```
pmarks :: Parser Text
pmarks = oneOf "()¿?!¡;:,.'\'\"[]{}-_@#<>«»&…\\/- | " >> return (pack "")

-- http://t.co/NgmDgQFDf
link :: Parser Text
link = (try (string "http://") <|> string "https://")
       >> many1 (alphaNum <|> oneOf "_/:?#.-") >> return (pack " ")

word :: Parser Text
word = letter >>= \c -> many (letter <|> char (head "'")) >>= \x
       -> return (toLower (pack (c:x)))

wordWS :: Parser Text
wordWS = letter >>= \c -> many (letter <|> char (head "'")) >>= \x ->
       return (case (elem (unpack (toLower (pack (c:x)))) stopwords) of
              True -> pack ""
              False -> toLower (pack (c:x)))

newline :: Parser Text
newline = string "\n" >> return (pack "\n")

numbers :: Parser Text
numbers = digit >>= \x -> return (pack (x:[]))

per :: Parser Text
per = option "" (string "-") >>= \a ->
       many digit >>= \b ->
       optionMaybe (string ".") >>= \c ->
       case c of
```

```
            Nothing -> (string "%" >> return (pack (a ++ b)))
            Just d -> many digit >>= \e ->
                   (string "%" >> return (pack (a ++ b ++ d ++ e)))

parse :: Parser Text
parse =  ( many (try per <|> try link <|> spaces <|> try user
       <|> try tweeted <|> hashtag <|> try parseEmoji <|> word <|> pmarks
       <|> numbers <|> newline ) )
       >>= \x -> return (replaceSpaces (Data.Text.concat x))

parseWithoutStopWords :: Parser Text
parseWithoutStopWords =  ( many (try per <|> try link <|> spaces <|> try user
       <|> try tweeted <|> hashtag <|> try parseEmoji <|> wordWS <|> pmarks
       <|> numbers <|> newline ) )
       >>= \x -> return (replaceSpaces (Data.Text.concat x))

replaceSpaces :: Text -> Text
replaceSpaces l = rrep 10 l

rrep :: Int -> Text -> Text
rrep 1 l = l
rrep n l = rrep (n - 1) (replace (pack (replicate n ' ')) (pack " ") l)

parseWith :: Parser Text
               -> FilePath -> IO (Either Text.Parsec.Error.ParseError Text)
parseWith fun file = parseFromFile fun file
```

La única parte importante que es necesario resaltar es el orden de los parsers en cada una de las funciones ”grandes”. Este orden es importante porque nos permite distinguir ”tweeted” de una

palabra cualquiera, por ejemplo. Todo lo demás son combinaciones de parsers.

III. Resultados

Se prueban los siguientes datos[2]:

```
@Ramiro_Pedroza tweeted: I'm at Villa India (Aguascalientes, México)
    http://t.co/RsjqLzz6IG
@ReschMoris tweeted: Este fin estuvo  @gretelresch @AndreaReyesh
    @casandraresch @hreyess8
@JaAC9510 tweeted: @pauvarelam jajaja ya veeeeen
@PedroCast23 tweeted: Domingos de comer solo :/
@Edy_camarena tweeted: La mejor pareja de todo el mundo :') &lt;3.
    http://t.co/HgCVKziLc6
@NayeliTun tweeted: RT @oscarponce82: LOS INVITO A SEGUIR A MI CARNAL
    @losprimosjavier CANTANTE DE @PRIMOSMX ARRIBA DURANGO CABRONES!!
        http://t.co/AgAOJIo…
```

Resultado con stopwords[3]:

```
$user tweeted i'm at villa india aguascalientes méxico
$user tweeted este fin estuvo ok_hand $user $user $user $user
$user tweeted $user jajaja ya veeeeen
$user tweeted domingos de comer solo annoyed
$user tweeted la mejor pareja de todo el mundo tears_of_happiness lt3
$user tweeted rt $user los invito a seguir a mi carnal $user cantante
    de $user arriba durango cabrones gun gun gun gun gun
```

[2]Esta es solo una porción ilustrativa de ellos, además incluyen líneas adicionales para que se pueda visualizar correctamente en el documento. El contenido completo está en: https://github.com/ivanmoreau/prep_textos

[3]El apóstrofe de "i'm" está siendo aceptado por ser considerarlo un carácter especial no-propio del español que le da sentido a la palabra, como un acento en español.

Resultado sin stopwords:

```
$user tweeted i'm at villa india aguascali
entes méxico
$user tweeted ok_hand $user $user $user $user
$user tweeted $user jajaja veeeeen
$user tweeted domingos comer annoyed
$user tweeted pareja mundo tears_of_happiness lt3
$user tweeted rt $user invito seguir carnal $user cantante $user
    durango cabrones gun gun gun gun gun
```

IV. Referencias

emojis. (2022, February 01). Retrieved from https://hackage.haskell.org/package/emojis

parsec. (2022, February 01). Retrieved from https://hackage.haskell.org/package/parsec

Alir3z4. (2022, February 01). stop-words. Retrieved from https://github.com/Alir3z4/stop-words