PRÁCTICA 5: TF-IDF

RECUPERACIÓN DE INFORMACIÓN
LICENCIATURA EN CIENCIAS DE LA COMPUTACIÓN
PRIMAVERA 2022

Gustavo Iván Molina Rebolledo
Benemérita Universidad Autónoma de Puebla

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

2 de marzo de 2022

I. Introducción

En esta práctica el objetivo es obtener consultas a partir de la información conseguida en la práctica anterior. El programa debe

1. Programar el pesado de términos usando TF/IDF utilizando el corpus de la práctica de preprocesamiento.
2. Guardar el pesado obtenido.
3. Crear 5 consultas y mostrar el pesado.

II. Implementación

Para implementar esta herramienta utilizo el lenguaje de programación Haskell, así como, en especial, la librería Parsec.

Este programa consta de tres partes (o archivos) en los que se dividen los distintos funcionamientos. El sistema de búsqueda y de matriz se encuentran en TFIDF.hs.

Las demás partes son reutilizadas de las prácticas anteriores como librerías (sistema de Query, por ejemplo).

Main.hs sólo se dedica a manejar el IO, e interconectar todo el funcionamiento.

III. TF/IDF

Esta sección es Literate Haskell. El mismo código es parte del ejecutable y parte del documento.

Licencia:

```
{- |
Copyright: (c) 2022 Iván Molina Rebolledo

SPDX-License-Identifier: GPL-3.0-only
Maintainer: Iván Molina Rebolledo <ivanmolinarebolledo@gmail.com>

See README for more info
-}
```

Boilerplate:

```haskell
{-# LANGUAGE OverloadedStrings #-}

module TFIDF ( genMatrix, search ) where

import           Data.List (delete, nub, elemIndices)
import qualified Data.Map as Dm
import           Data.Map (Map, fromList, insert, (!))
import           Data.Serialize.Text ()
import           Data.Text (Text, pack, replace, splitOn)
import           Query

type MatrixW = Map Text (Map Int Double)
```

Lista de palabras a ignorar:

```haskell
rone :: [Text]
rone = ["$user tweeted ", " $ht", "$ht ", " $user", "$user "]
```

Es necesario limpiar todas las lineas del corpus (eliminar una lista de palabras)

```haskell
fixWords :: Text → [Text] → Text
fixWords xs [] = xs
fixWords xs (l : ls) = fixWords (replace l (pack "") xs) ls
```

con el fin de obtener una matriz limpia con las palabras.

```haskell
getWords :: Text → MatrixW
getWords l =
  fromList
    ( map
        (\x → (x, fromList [(0, 0)]))
        (Data.List.delete "" (nub (splitOn " " (replace "\n" " " l))))
    )
```

Se ha de inicializar la matriz con el número de documentos:

```haskell
emptymatrix :: MatrixW → Int → MatrixW
emptymatrix m c = Dm.map (\_ → fromList (map (\x → (x, 0)) [0 .. c])) m
```

En este código se establece la existencia de una palabra para el documento n; establece o actualiza la frecuencia.

```haskell
aa :: MatrixW → Text → Int → Double → MatrixW
aa m t i val = let r = m ! ( t) in insert t (insert i val r) m
```

Se define la variante del peso TF, idf y el tf-idf:

```haskell
tf :: Text → Text → Double
tf i t = let o = Prelude.length $ elemIndices i $ splitOn " " t in
  if o > 0 then
    1.0 + logBase 2.0 (fromIntegral o)
  else
    0

findcol :: Text → Text → Bool
findcol term text = (>0) $ Prelude.length $ filter (==term) $ splitOn " " text
findtex :: Text → [Text] → Int
findtex term textl = Prelude.length $ filter (==True) $ map (findcol term) textl

idf :: Text -- Termino.
  → [Text] -- [Lineas].
  → Double
idf t m = let n = findtex t m in
  logBase 2.0 $ (fromIntegral $ Prelude.length m) / (fromIntegral n)

tfidf :: Text → Text → [Text] → Double
tfidf term line lines_ = (tf term line) * (idf term lines_)
```

Se llena la matriz por linea:

```haskell
byLine :: [Text] → [Text] → MatrixW → Int → MatrixW
byLine [] _ m _ = m
byLine (l : ls) lines_ m c =
  let words_ = filter (/="") $ splitOn " " l
      w_ = map (\w → (w, tfidf w l lines_)) words_ in
  byLine ls lines_ (f m ( w_)) (c + 1)
  where f m_ [] = m_
        f m_ ((w, w_):xs) = f (aa m_ w c w_) xs
```

Se genera la matriz:

```haskell
genMatrix :: Text → MatrixW
genMatrix t =
  let ff = fixWords t rone
      gw = getWords ff
      ts = (splitOn "\n" ff)
      em = emptymatrix gw (length ts)
   in byLine ts ts em 0
```

Busca la palabra y regresa los indices de los documentos:

```haskell
readd :: Text → MatrixW → [(Int, Double)]
readd t m = case Dm.lookup t m of
  Just v → (Dm.toList (Dm.filter (>0) v))
  Nothing → []
```

Se evalua el árbol de Query:

```haskell
lmin :: [(Int, Double)]→ [(Int, Double)] → [(Int, Double)]
lmin _ [] = []
lmin l0 (l1:ls1) = lmin (filter (\(x, _) → x /= (fst l1)) l0) ls1

lunion :: [(Int, Double)]→ [(Int, Double)] → [(Int, Double)]
lunion l0 l1 = let n = filter (\(x,_) → x `notElem` (fst $ unzip l0))
↪  l1 in
  l0 ++ n

```

```haskell
lintersect :: [(Int, Double)]→ [(Int, Double)] → [(Int, Double)]
lintersect l0 l1 = let n = filter (\(x,_) → x `elem` (fst $ unzip l0))
 ↪  l1 in
  n

pw :: Query → Int → MatrixW → [(Int, Double)]
pw (QAnd l r) t m = (pw l t m) `lintersect` (pw r t m)
pw (QOr l r) t m = (pw l t m) `lunion` (pw r t m)
pw (QNot e) t m = (zip [0..t] (replicate t 0)) `lmin` (pw e t m)
pw (QP v) _ m = readd v m
```

Se realiza una consulta con una Query:

```haskell
search :: Query → MatrixW → [(Int, Double)]
search q m = pw q (Dm.size (snd (Dm.elemAt 0 m))) m
```

IV. Funcionamiento

Podemos usarlo de la siguiente forma:

```
ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- --help
The tfidf program

tfidf [COMMAND] ... [OPTIONS]

Common flags:
  -? --help         Display help message
  -V --version      Print version information

boleanmodel matrix [OPTIONS]

  -f --from=ITEM
  -t --to=ITEM

boleanmodel query [OPTIONS]
```

```
  -o --ogfile=ITEM
  -m --matrix=ITEM
  -q --query=ITEM
ivanmolinarebolledo@Ivans-macOS tf_idf %
```

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.

V. Grammar

```
Query → a
a → o ("and" o)*
o → term ("or" term)*
term → "not" Query | "(" Query ")" | Identifier
Identifier → alphaNum (alphaNum | "'")*
```

VI. Resultados

Se prueban las siguientes queries con resultados satisfactorios:

```
ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- query
↪  -o="Tweets.txt" -m="mm.ths" -q="cama or museo"
[13] @LuisGcortez tweeted: Parece que falta algo de atención y
↪  mantenimiento al museo de la ciudad por parte del @ICAdifusion
↪  http://t.co/FLbC5SbcoF (W: 8.45532722030456)
[201] @JaAC9510 tweeted: Hoy me niego a levantarme de mi cama (W:
↪  7.455327220304562)
[16] @yosoyloagui tweeted: Tu y yo juntos en un lugar perfectollamado mi
↪  cama *.* (W: 7.455327220304562)

ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- query
↪  -o="Tweets.txt" -m="mm.ths" -q="(invito and not carnal) or novio"
```

```
[186] @Andresrgtz tweeted: Mi amiga se besó a una mujer muy sexi el fin
↪  de semana no le digan a su novio (W: 6.870364719583405)
[49] @KarlyyRG tweeted: Nada mejor que tener un novio super divertido
↪  que en lugar de limitarte, te sigue el pedo. (W: 6.870364719583405)
[42] @KarlyyRG tweeted: Lo que mas me encanta de mi novio es que me hace
↪  morir de risa. (W: 6.870364719583405)

ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- query
↪  -o="Tweets.txt" -m="mm.ths" -q="vida and riesgo or dios"
[206] @alex_dsr01 tweeted: Ni pedo la vida es un riesgo -Fergus (W:
↪  8.45532722030456)
[345] @CaballoNegroII tweeted: RT @MyRyCaR: Gracias por tu presencia en
↪  mi vida … un regalo de Dios  teee amooo @CaballoNegroII
↪  http://t.co/grQRS43NK1 (W: 6.133399125417199)
[203] @abrilushiZ tweeted: RT @_equiswe: Si Dios me quita la vida antes
↪  que a ti … (W: 6.133399125417199)

ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- query
↪  -o="Tweets.txt" -m="mm.ths" -q="tortillas or hambre or feliz"
[69] @PiernasAlLomo tweeted: @La_Montserrat a las tortillas? (W:
↪  8.45532722030456)
[27] @IAN_BLACK26 tweeted: Mitad de quincena que comiencen los juegos
↪  del hambre. (W: 7.455327220304562)
[14] @mc_gabucha tweeted: Ahora entiendo por que se juntó la necesidad
↪  con el hambre … ¡Que gente tan acomplejada! (W: 7.455327220304562)
[325] @123anotaz tweeted: RT @CarlaMorrAgs: La mujer es una obra de arte
↪  que ilumina los ojos de quien la mira. Feliz Dia de la Mujer
↪  @CarlaMorrisonmx ♥ (W: 6.455327220304562)
[265] @fantasmaluigui tweeted: Siempre recuerda: No tomes decisiones
↪  cuando estés enojado, y no hagas promesas cuando estés feliz. (W:
↪  6.455327220304562)
[236] @nichelopezc tweeted: Hoy ni la pizza me hace feliz carajo!!' (W:
↪  6.455327220304562)
```

```
[109] @zubyelynl tweeted: Hola feliz domingo tengan tod@s hoy les
↪  presento esta capa o poncho que se puede usar tanto en temporada de
↪  frio ... http://t.co/qh1xeinf22 (W: 6.455327220304562)

ivanmolinarebolledo@Ivans-macOS tf_idf % stack run -- query
↪  -o="Tweets.txt" -m="mm.ths" -q="amiga and novio"
[186] @Andresrgtz tweeted: Mi amiga se besó a una mujer muy sexi el fin
↪  de semana no le digan a su novio (W: 6.870364719583405)
```

VII. Referencias

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