Similitud entre documents

Programació Declarativa, Aplicacions.

10 d'octubre de 2020

En aquesta pràctica es demana que implementeu algunes funcions bàsiques per al tractament de documents de text per tal de poder cercar similituds entre ells. La noció de similitud entre documents és complexa i hi ha nombroses aproximacions. En aquesta pràctica n'usarem una de força senzilla. Per començar intentarem buscar el coeficient de similitud entre alguns llibres.

El que es demana en aquesta primera part de la pràctica és que implementeu un programa `simil` tal que donats dos fitxers ens digui la similitud entre ells. Per aconseguir-ho farem servir una mesura anomenada similitud de cosinus sobre els vectors de *freqüencies normalitzades* dels fitxers, que descriurem més avall. Per a fer això volem que implementeu les funcions següents (tot i que no les fareu servit totes en aquesta part de la pràctica):

- **Freqüències de paraules:** Caldrà que feu una funció: `freq` que donat un `String` ens torni la llista d'ocurrències de les paraules de l'`String`, `List[(String,Int)]`. *Pista, primer normalitzeu: elimineu tots els caràcters que no fan paraules (es a dir, us quedeu només amb les lletres) i passeu totes les lletres a minúscules, de manera que us quedi un* `String` *format de paraules en minúscules separades per espais.*

  Per comprovar que aneu bé, caldrà que em pogueu mostrar el nombre de paraules de l'`String` donat, el nombre de paraules diferents i em mostreu també les freqüències de les 10 paraules més freqüents (tant l'absoluta com la relativa). Comproveu que pel llibre "Alice in Wonderland" us dona quelcom com ara:

```
Num de Paraules:           30419      Diferents: 3007
Paraules       ocurrencies   frequencia
---------------------------------------
the            1818    5,98
and            940    3,09
to             809    2,66
a              690    2,27
of             631    2,07
it             610    2,01
she            553    1,82
i              545    1,79
you            481    1,58
said           462    1,52
```

- **Sense stop-words:** com podem veure les paraules més freqüents corresponen a *stop-words*, és a dir a paraules descartables a priori pel que fa a la semàntica del document (articles, pronoms, etc.).

  Cal que creeu una funció `nonstopfreq` que faci com `freq`, és a dir, rebi un `String`, però que a més rebi una llista d'stop-words a descartar i llavors ens torni la llista de freqüències de `String` que ens han passat sense les stop-words. Obtingueu aquesta llista del fitxer `english-stop.txt`. Comproveu que pel llibre "Alice in Wonderland" us dona quelcom com ara:

```
Num de Paraules:           30419     Diferents: 3007
Paraules       ocurrencies   frequencia
----------------------------------------
alice          403    1,32
gutenberg      93    0,31
project        87    0,29
queen          75    0,25
thought        74    0,24
time           71    0,23
king           63    0,21
don            61    0,20
turtle         59    0,19
began          58    0,19
```

- **Distribució de paraules:** comproveu que efectivament passa que hi ha poques paraules que apareixen molt i moltes que apareixen poc. Feu una funció `paraulafreqfreq` que donat un `String` ens mostri les 10 freqüències més freqüents i les 5 menys freqüents. Comproveu que pel llibre "Alice in Wonderland" us dona quelcom com:

```
Les 10 frequencies mes frequents:
1330 paraules apareixen 1 vegades
468 paraules apareixen 2 vegades
264 paraules apareixen 3 vegades
176 paraules apareixen 4 vegades
101 paraules apareixen 5 vegades
74 paraules apareixen 8 vegades
72 paraules apareixen 6 vegades
66 paraules apareixen 7 vegades
39 paraules apareixen 9 vegades
35 paraules apareixen 10 vegades
Les 5 frequencies menys frequents:
1 paraules apareixen 631 vegades
1 paraules apareixen 690 vegades
```

```
1 paraules apareixen 809 vegades
1 paraules apareixen 940 vegades
1 paraules apareixen 1818 vegades
```

- **ngrames:** un *ngrama* és una seqüència de n paraules. Aquest concepte també és molt útil en l'anàlisi de textos (per exemple per al reconeixement d'idiomes). Adapteu/utilitzeu les funcions implementades per a poder trobar les freqüències, donat una n i un `String`, dels ngrames de l'`String` rebut. Comproveu que pel llibre "Alice in Wonderland" us dona quelcom com:

```
project gutenberg tm      57
the mock turtle           53
i don t                   31
the march hare            30
the project gutenberg     29
said the king             29
the white rabbit          21
said the hatter           21
said to herself           19
said the mock             19
```

- **Vector space model**: per a tenir una mesura "semàntica" del document i poder-la fer servir per calcular similituds, sovint es recorre a la representació en forma de vector del "pes" de les paraules. Dit d'una altra manera, per a cada document es crea una llista de parelles (paraula-i-èssima, pes), de manera que aquests pesos representen un vector. Llavors, per comparar dos documents podrem recorrer a la noció de similitud de cosinus. Caldrà doncs que feu una funció `cosinesim` que donats dos `String`s ens doni el seu coeficient de similitud de cosinus, un valor entre 0(menys similar) i 1(més similar).

  - Per a pes farem servir la freqüència normalitzada, és a dir,

    $$\frac{f_{p,d}}{MAXf_d}$$

    on $f_{p,d}$ serà la freqüència (absoluta) de la paraula en el document i $MAXf_d$ la freqüència (absoluta) de la paraula que més apareix al document.
  - la similitud de cosinus donats dos vectors a, b de dimensió m es defineix com a:

    $$sim(a,b) = \frac{a \cdot b}{\sqrt{\sum_{i=1}^{m} a[i]^2} \cdot \sqrt{\sum_{i=1}^{m} b[i]^2}}$$

  Compte, les freqüències les farem sobre el text sense stop-words, a més els vectors per al càlcul de similituds ha d'estar "alineat", és a dir, tots dos vectors han de tenir les freqüències de les mateixes paraules a la mateixa posició. Per exemple $a[i] = 0, b[i] = 0.004$ vol dir que la paraula i-èssima no apareix al text del vector a mentre que si que apareix al text del vector b amb una freqüència normalitzada de 0.004.

- Comproveu quant s'assemblen els llibres penjats a la web.
- Comproveu també si enlloc de paraules soles busqueu la similitud considerant digrames i trigrames.
- Doneu una taula amb els resultats de les comparacions.

Ajuts

Fer servir recursivitat amb documents tant grans pot donar problemes de pila si no feu recursivitat final. Us convido enèrgicament a fer servir funcions d'ordre superior, conversores, etc.. De la meva implementació us dic que us poden ser útils: `foldLeft`, `contains`, `updated`, `toList`, `map`, `isLetter`, `toLower`, `split`, `filter`, `sortWith`, `take`, `drop`, `toFloat`, `length`, `filterNot`, `sliding`, `groupBy`, `zip`, `...`

Lliurament

- Les pràctiques es poden fer en equips de dos.
- Cal que documenteu el codi.
- Lliurareu un document amb el codi de cada apartat i exemples d'execució així com les taules de resultats de les comparacions finals.
- Properament es publicarà la segona part de la pràctica i les instruccions de lliurament.