Bugiardino

Un tool per la ricerca di informazioni
all'interno dei fogli illustrativi

https://github.com/alema-r/Bugiardino

Idea

- Raccogliere i dati relativi ai fogli illustrativi dei farmaci
- Analizzare ed estrarre le informazioni cruciali
- Ricercare, data una malattia o sintomo, i farmaci adeguati per quella patologia

Raccolta dati

- La raccolta dati è stata effettuata attraverso la banca dati dei farmaci dell'Agenzia Italiana del Farmaco (AIFA).
- Nella banca dati è possibile effettuare tre diversi tipi di ricerca:
  - per farmaco
  - per principio attivo
  - per casa farmaceutica
- Quella più rilevante allo scopo del progetto è la ricerca per farmaco.

Raccolta dati

- Attraverso diversi tentativi è emerso che per ricavare tutti i farmaci è sufficiente immettere nel campo di ricerca due asterischi `**`.
- Come risposta alla richiesta si ottiene un json contenente tutti i farmaci che soddisfano i criteri di ricerca (quindi in questo caso tutti i farmaci presenti nella banca dati).

Parsing del JSON

- La richiesta viene effettuata attraverso il modulo `requests` di Python
- Dal json si ricavano le seguenti informazioni per ogni farmaco:
  - codice del farmaco
  - nome del farmaco
  - casa farmaceutica
  - principio attivo
  - URL del foglio illustrativo

Parsing del JSON

- Il json ottenuto viene poi filtrato nei seguenti modi:
  - rimozione dei duplicati
  - rimozione dei farmaci che non presentano il principio attivo o l'url del foglio illustrativo
- Utilizzando la classe `Farmaco`, viene creato un oggetto per ogni farmaco presente nel json

Estrazione dei fogli illustrativi

- Ogni oggetto di tipo `Farmaco`, ha un attributo che specifica il link del PDF del foglio illustrativo
- Per ricavare quindi, le informazioni sul foglio illustrativo è necessario "leggere" il PDF
- Per prima cosa ricaviamo il PDF, facendo una richiesta GET al link del foglio illustrativo
- In questo caso non è stato possibile utilizzare `requests`, visto che, anche modificando lo User Agent e utilizzando i Cookie non era possibile accedere alla risorsa

Estrazione dei fogli illustrativi

- Per risolvere il problema si è utilizzato curl attraverso il modulo `subprocess` di Python:

```
proc = subprocess.Popen(["curl", "-s", url], stdout=subprocess.PIPE)

(out, _) = proc.communicate()

return out
```

- Una volta ricavati fogli illustrativi si passa al filtraggio degli stessi

Filtraggio dei fogli illustrativi

I fogli illustrativi sono stati filtrati nei seguenti modi:

- rimozione dei punti dopo il numero 1. (se il foglio illustrativo non presenta questo punto viene scartato)
- rimozione di tab e newline (\t , \n)
- sostituzione degli apostrofi con degli spazi
- sostituzione di "s.p.a." con "s.p.a" per evitare il punto finale
- rimozione di frasi non rilevanti

Da Python a Prolog

Una volta estratti i fogli illustrativi popoliamo il file `farmaci.pl` con i fatti rilevanti i farmaci. Questi avranno la seguente struttura:

```
farmaco('Nome Farmaco', 'Casa Farmaceutica', 'Principio attivo', ['lista', 'di', 'parole']).
```

Dove l'ultimo termine rappresenta il foglio illustrativo sotto forma di lista di parole

Estrazione delle frasi più importanti

Per estrarre le frasi più importanti dai fogli illustrativi, cioè quelle che mi dicono a cosa serve quel farmaco, è stato utilizzato il seguente metodo:

1. Ogni lista che rappresenta il foglio illustrativo è stata divisa in una lista di frasi. Per riconoscere le frasi si fa affidamento al punto.
2. Di tutte le frasi, vengono salvate in memoria (tramite `assertz`) solo le frasi che contengono una parola chiave. (La lista di queste parole è salvata nel file `parole_chiave.pl`)
3. Questo processo viene ripetuto per tutti i farmaci

Ricerca dei farmaci

- La ricerca dei farmaci avviene in maniera diversa in base al numero di parole inserite dall'utente
- In particolare si ha distinzione tra singola parola e più parole (ad es: febbre vs mal di testa)

Ricerca con singola parola

Nella ricerca con singola parola, si effettuano le seguenti operazioni:

- si prende una frase rilevante di un farmaco
- tramite `memberchk` si controlla se la parola appartiene alla frase
- si ripete il processo per tutte le frasi rilevanti

Ricerca con più parole

Questo tipo di ricerca risulta leggermente più articolata rispetto alla precedente:

- come prima, si prende una frase rilevante
- tramite `nth0` stavolta, si controlla se la prima parola ricercata appartiene alla frase
- se la risposta è sì, allora parte la ricorsione tenendo conto dell'indice trovato con `nth0` (la seconda parola si dovrà trovare subito dopo la prima, la terza subito dopo la seconda, ecc)

Interfaccia grafica

Per interagire con il tool è stata progettata una semplice interfaccia grafica realizzata con Python attraverso il modulo `tkinter`.

L'interfaccia grafica permette di:

- ricercare un sintomo o malattia
- ottenere una lista di farmaci relativi alla ricerca
- cliccando su un farmaco della lista, permette di vedere i dettagli del farmaco, quali: nome del farmaco, casa farmaceutica, principio attivo e frase (o frasi) estratta dal foglio illustrativo

Utilizzare Prolog direttamente da Python

Per rendere tutto ciò possibile è stato utilizzato `PySwip`.
Dalla pagina Github:

> PySwip is a Python - SWI-Prolog bridge enabling to query SWI-Prolog in your Python programs. It features an (incomplete) SWI-Prolog foreign language interface, a utility class that makes it easy querying with Prolog and also a Pythonic interface.

Possibili miglioramenti

Di seguito, un elenco (non esaustivo) dei possibili miglioramenti del progetto o di possibili estensioni:

- Estendere le funzioni del tool agli altri punti del foglio illustrativo:
  - avvertenze per l'uso
  - posologia (adulti, bambini, persone in stati particolari come ad esempio donne incinta)
  - effetti collaterali

Possibili miglioramenti

- Estendere le funzioni del tool anche ad altri schemi di fogli illustrativi presenti nella banca dati AIFA
- Estendere la ricerca di un farmaco anche ai sinonimi della parola cercata (es: cercando "febbre", si ottengono risultati anche per "influenza" o "stati febbrili")
- Migliorare le funzionalità di ricerca aggiungendo un'analisi semantica
- Utilizzare il tool su altre fonti relative ai fogli illustrativi (es: Farmadati)

Chiunque può prendere e migliorare questo progetto. Il codice è open-source e mi farebbe piacere se qualcuno riprendesse dove io ho lasciato.
Link alla pagina github: https://github.com/alema-r/Bugiardino