MÁSTER EN INTELIGENCIA ARTIFICIAL

INFORMÁTICA BIOMÉDICA

Buscador de LOINC basado en SVM optimizado usando clicks de usuarios*

Autores
AÍDA MUÑOZ MONJAS
CÉSAR PANTOJA ROSALES
GABRIEL RIVERA CÁRDENAS

December 12, 2022

1 Introducción

Tradicionalmente, algoritmos como el TF-IDF o el BM25 usados para realizar motores de búsqueda clasifican los resultados obtenidos para cada consulta en ”relevantes” e ”irrelevantes”. Esta clasificación no caracteriza de manera completamente correcta las opiniones de los usuarios, ya que hay resultados más relevantes que otros, de manera que se puede establecer un orden prácticamente total de la relevancia óptima de los resultados.

Obtener este ranking sin tener un feedback explícito no es trivial, y conseguir estos comentarios por parte de los usuarios es difícil. El conocimiento sobre a qué entradas de la búsqueda acceden los usuarios nos puede proporcionar información equivalente, de manera mucho menos costosa. El principal inconveniente de utilizar el conocido como ”click-through data”, datos sobre los clicks de los usuarios, es la cantidad de ruido presente en los datos y la dependencia que existe entre los clicks de los usuarios y el orden de los documentos recibidos.

Sin duda este tipo de datos son útiles y poco costosos de conseguir, pero su calidad no se puede comparar con aquella de los juicios de relevancia generados por expertos del dominio.

En este trabajo, se nos pide implementar los algoritmos descritos en el artículo [1] sobre un set de tres búsquedas sobre la terminología LOINC.

LOINC (Logical Observation Identifiers, Names and Codes)[2] es una terminología de términos de laboratorio, donde cada concepto viene definido por el componente medido (component), el sistema sobre el que se observa (system), la propiedad observada (property) y su nombre (long common name), este último agrupando las otras tres características del término.

2 Desarrollo del buscador

Utilizando el lenguaje python, se ha adaptado el dataset proporcionado a las necesidades del proyecto, y se ha preparado una implementación de un buscador basado en el algoritmo BM25, optimizado mediante los clicks de los usuarios.

2.1 Procesado de datos

En el dataset se proporcionaron tres búsquedas sobre LOINC, de las cuales cada consulta tenía una lista de posibles respuestas. Para cada una de las consultas se obtuvo un dataset como el de la Figura 1.

id	loinc_num	long_common_name	component	system	property
0	1988-5	C reactive protein [Mass/volume] in Serum or Plasma	C reactive protein	Ser/Plas	MCnc
1	1959-6	Bicarbonate [Moles/volume] in Blood	Bicarbonate	Bld	SCnc
2	10331-7	Rh [Type] in Blood	Rh	Bld	Type
3	18998-5	Trimethoprim+Sulfamethoxazole [Susceptibility]	Trimethoprim+Sulfamethoxazole	Isolate	Susc
4	1975-2	Bilirubin.total [Mass/volume] in Serum or Plasma	Bilirubin	Ser/Plas	MCnc

Figure 1: Ejemplo del dataset recibido para la consulta ”GLUCOSE IN BLOOD”

Los autores de este trabajo seleccionaron, actuando como usuarios, a cuáles de los códigos harían click según la descripción textual de la búsqueda. De esta manera, se pudo generar un dataset con las tripletas habituales para este tipo de datos. Esta información se guardó en un segundo dataset, representando la consulta, el orden de respuestas presentadas y aquellas respuestas a las que se les hizo click, indicando el número de veces.

Como se indica en el articulo [1], la información relativa al clickthrough data se codifica como tripletas, donde cada búsqueda (query) se relaciona con los resultados obtenidos en un orden, y el número de clicks realizados sobre cada uno de ellos.

El procesado de los datos de este problema se ve afectado por dos importantes decisiones de diseño: la elección del algoritmo de búsqueda y los campos utilizados para realizar la búsqueda.

Como algoritmo de búsqueda se utilizó el algoritmo BM25Okapi, por ser uno de los algoritmos base más robustos habitualmente utilizado en este campo. Para la implementación de la búsqueda se utilizó la librería `rank_bm25` y su implementación del algoritmo.

En segundo lugar, se decidió utilizar los campos *long common name*, *component* y *system* como la base de este motor de búsqueda, ya que proporcionan información suficiente para poder realizar la búsqueda, y así eliminar la necesidad del campo *component*, cuya información no se podía utilizar con las consultas propuestas.

2.2 Implementación

La implementación realizada se puede observar en la carpeta adjunta `code`, que contiene los ficheros de código necesarios, así como los ficheros de entrada del algoritmo en la carpeta `code/loinc_dataset` y los resultados de la ejecución en la carpeta `code/result`.

Para la implementación, la información se separó en dos dataset. El dataset *train*, que contiene las primeras 50 filas, se utilizó para entrenar el algoritmo, mientras que el conjunto *test* se utilizó para mejorar la búsqueda y así poder evaluar el algoritmo.

La implementación realizada sigue el siguiente esquema

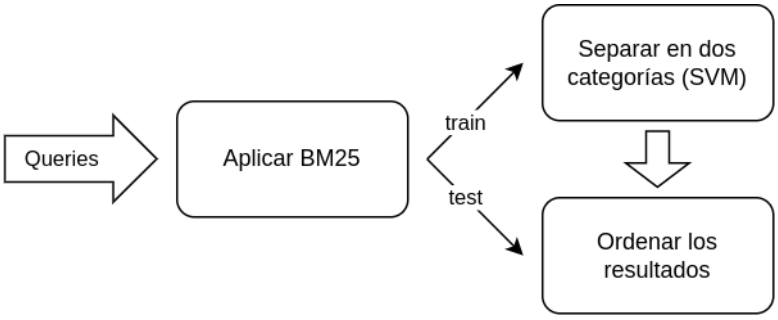

Figure 2: Esquema de la implementación

Veamos como se aplica esta implementación a la query *"glucose in blood"*, glucosa en sangre, sobre el dataset proporcionado.

En primer lugar, se aplica el algoritmo BM25 sobre la query a procesar, obteniendo una estructura de datos que representa la importancia relativa de cada uno de los atributos del código de LOINC respecto de la query.

```
[ glucose in blood ]BM25 rank:
    loinc_num  long_common_name  component  system  sum_clicks
0      1988-5          4.746541    0.00000     0.0           0
66    23658-8          4.746541    0.00000     0.0           0
23    74774-1          3.245196    2.91821     0.0           3
12    15076-3          3.073961    2.91821     0.0           3
19    14747-0          2.851333    2.91821     0.0           2
..        ...               ...        ...     ...         ...
48    18955-5          0.000000    0.00000     0.0           0
51    19000-9          0.000000    0.00000     0.0           0
60    20629-2          0.000000    0.00000     0.0           0
63    18878-9          0.000000    0.00000     0.0           0
64    18928-2          0.000000    0.00000     0.0           0
```

Figure 3: Resultado de aplicar el algoritmo BM25.

Cabe destacar que con el algoritmo BM25, la importancia de los contenidos del system se marca como 0 en todos los casos. Esto se debe a que este campo contiene abreviaturas, representando la palabra *blood* como *Bld*, por lo que una búsqueda por palabra exacta no obtiene ningún resultado.

A continuación, utilizando la transformada pairwise y el modelo SVC linear, convertimos nuestro problema de búsqueda en un problema de clasificación bidimensional, teniendo en cuenta los clicks de los usuarios y el resultado de la búsqueda anterior.

Por último, ordenamos los resultados del dataset `test`, y generamos unas gráficas que nos permitan visualizar los resultados.

2.3 Testeo y resultados

3 Conclusiones

References

[1] T. Joachims, "Optimizing search engines using clickthrough data," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '02. New York, NY, USA: Association for Computing Machinery, 2002, p. 133–142. [Online]. Available: https://doi.org/10.1145/775047.775067

[2] LOINC. Regenstrief Institute - Home. https://loinc.org/. (Accessed on 06/12/2022).