Ricard Meyerhofer Parra

Jose Martín Nogueira

2016-2017 Q1

CAIM - Third Deliverable: User Relevance Feedback

a) Esquema seguido durante la realización de la práctica

Durante toda esta práctica hemos seguido el guión que se nos ha proporcionado.

Empezamos creando la función, userRelevanceFeedback en la que usaremos el algoritmo de Rocchio.

private static Query userRelevanceFeedback(Query query, IndexSearcher searcher, int nrounds)

Para implementar el algoritmo de Rocchio hemos utilizado dos funciones:

- private static TermWeight[] compressVectors(TermWeight[] a, TermWeight[] b)

Esta función comprime los dos vectores en uno, sumando los pesos de los repetidos.

- private static TermWeight[] multK(TermWeight[] a, double k)

Donde multK multiplica cada elemento del vector a por k.

El algoritmo de Rocchio se basa principalmente en en la siguiente fórmula:

Q' = a*Q + b*(d1+...+dk)/k.

Donde Q es un vector de la primera query, [d1...dk] sin vectores del top k documentos y donde a y b son constantes que definen la credibilidad.

Para conseguir las variables de la fórmula, hemos implementado las siguientes funciones:

- private static void normalize(TermWeight[] t)

Para que el resultado sea correcto, los vectores tienen que estar normalizados.

- private static TermWeight[] toTfIdf(IndexReader reader, int docId)

Con esta función obtenemos los [d1...dk] de la fórmula.

- **private static String purge(TermWeight[] q)**

Esta función nos permite seleccionar los N componentes más pesados del vector resultante.

b) Dificultades que hemos experimentado durante la realización de esta

Las dificultades que hemos encontrado durante la implementación se deben en parte a al algoritmo de Rocchio. Cuando normalizar los vectores, qué weight debían tener los elementos y cómo tendrían sumarse con los otros términos o si se deberían normalizar antes de hacerlo.

c) Experimentos y resultados observados

En consecuencia de tener muchas variables relacionadas entre sí, es difícil obtener resultados concluyentes.

Pseudorelevance feedback genera una query con nuevos términos relacionados con la query original hacinedo que seleccione documentos que antes no seleccionaba. Esto mejor el recall sacrificando algo de precisión.

El número de nrounds establecidas modifican en cierto grado la query original, cuantas más nrounds más modificada quedará. Para obtener resultados óptimos, establecemos que nrounds = 2 y así solo utilizamos Rocchio una vez.

La variable k marca los k documentos que serán relevantes. Como es totalmente arbitraria y selecciona los k primeros, hay documentos que no se tratan y podrían contener algo interesante. También afecta al recall, a que será menor y tampoco aseguramos un aumento de precisión. La k óptima seria k = ndocs.

Para generar un recall sin perder mucha precisión, N debe ser relativamente pequeña ya que hará que los términos de la query sean mucho más relevantes.

Los valores de a y b son los responsables de la medida del vector respecto al original. Si a = 0, la nueva query no tendrá presentes ningún término de la query original, si b = 0, no habrá nuevos términos.