

# Retorno de información

El **Retorno de informacion** es el proceso de retornar documentos a partir de una coleccion de documentos en respuesta a una consulta.

- Los documentos suelen estar en lenguaje natural no estructurado
  - No tiene un modelo formal bien definido
  - Se basa en la comprension del lenguaje natural
  - Se almacena en una variedad amplia de formatos estandares.

Los sistemas de retorno de informacion **SRI** permiten *expresiones de consulta* formadas usando palabras clave y conectivos proposicionales.

- Algunos lenguajes de consulta pueden ser:
  - Consultas usando frases: una frase es secuencia de palabras
  - Consultas de palabras clave: se escribe texto de palabras clave y se asume and entre esas palabras
  - Consultas booleanas: las expresiones involucran terminos y conectivos booleanos
  - Consultas basadas en regex: busqueda basada en patrones
  - Consultas de proximidad: se expresa que tan cerca deben estar ciertos terminos entre si.
  - Consulta lenguaje natural: requiere entender la estructura y su significado, puede ser pregunta o narrativa.

Los **resultados de una busqueda** pueden ser una lista de identificadores de documentes y tambien algunas piezas de texto. Ademas los doc se ordenan segun algun puntaje de relevancia.

- Esta se mide en factores como
  - Frecuencia de términos de una consulta en un doc
  - Frecuencia inversa de documentos, si ocurre en menos documentos se le da mas importancia a la palabra
  - Enlaces de documentos, si hay mas enlaces a un doc, entonces es mas importante
  - Las palabras que ocurren en el título, lista de autores, títulos se les da más importancia.
  - Las palabras cuya primera ocurrencia es tarde en el documento se les da poca importancia.
  - Proximidad: si las palabras de una consulta aparecen cerca entre sí en el documento, el documento tiene más importancia que si las palabras ocurren bien lejos unas de otra.

No es eficiente acceder a todo un doc, por lo que para cada uno se construye una estructura que resume su contenido. En un **enfoque estadístico** los doc son analizados y descompuestos en piezas (palabras, frases, n-gramas). Luego dada una consulta se comparan sus términos con las piezas del doc para determinar la correspondencia y relevancia. Existen diferentes modelos estadísticos para establecer los resultados.

## Modelo Booleano

- Los documentos se representan como un conj de términos
- Se usan consultas booleanas
- El retorno es una correspondencia exacta (sirve o no sirve)

- No existe relevancia

## Modelo de espacio vectorial

- Cada documento se representa con un vector de valores de dimension n. Donde cada termino es una dimension.
- En cada dimension hay un numero que puede representar
  - valor booleano representando presencia en el doc
  - frecuencia del termino en el doc
  - peso (usando TF-IDF)
- Una consulta se representa como un vector de terminos, luego el vector de una consulta es comparado(funcion de similitud) con los vectores de los documentos para establecer una medida de relevancia (similitud/relevancia)
  - Los terminos tienen peso en los vectores. Este puede ser la frecuencia de cada termino en el doc/consulta o usando **TF-IDF**.
- Las respuestas se ordenan por relevancia

### TF-IDF (frecuencia de termino-frecuencia inversa de doc)

Un termino que captura la esencia de un documento aparece frecuentemente. Pero para poder discriminar documentos a partir de un termino, debe ocurrir en pocos docs de la colección.

- Sea término i y documento Dj

- $TF\text{-}IDF_{i,j} = T_{f_{i,j}} * IDF_i$

$$TF_{ij} = f_{ij} / \sum_{i=1 \text{ to } |V|} f_{ij}$$

$$IDF_i = \log(N / n_i)$$

- $T_{f_{i,j}}$  es la frecuencia del término i en el documento Dj normalizada.
- $f_{ij}$  es la cantidad de ocurrencias del término i en el documento Dj.
- N es el número de documentos en la colección.
- $n_i$  es el número de documentos donde el término i ocurre.

- Se puede calcular la relevancia de un documento  $D_j$  con respecto a una consulta  $Q$  de la siguiente manera:

$$\text{rel}(D_j, Q) = \sum_{i \in Q} TF_{ij} \times IDF_i$$

- Se usa función de similitud de vectores.
- La función del coseno del ángulo entre los vectores de la consulta y el documento se usa frecuentemente para estimar la similitud.

$$\text{cosine}(d_j, q) = \frac{\langle d_j, q \rangle}{\|d_j\| \times \|q\|} = \frac{\sum_{i=1}^{|V|} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{|V|} w_{ij}^2} \times \sqrt{\sum_{i=1}^{|V|} w_{iq}^2}}$$

- Donde  $d_j$  es el vector del documento  $j$ ,  $q$  es el vector de la consulta.
- $w_{i,j}$  es el peso del término  $i$  en el documento  $j$ ,  $w_{i,q}$  es el peso del término  $i$  en el vector de la consulta  $q$ .
- $|V|$  es el número de dimensiones en el vector.

## Como crear la representación de un doc?

Es necesario procesarlo para encontrar sus terminos relevantes. Pero no todos son relevantes.

- **stop word** : palabras que se espera que ocurran en el 80% o mas del doc
  - ej: el, la, lo, de, un, etc.
  - No contribuyen a la relevancia -> se ignoran (en la representacion de doc y en consultas)
- Para un termino que aparece de multiples formas (un verbo y sus conjugaciones) existe el **algoritmo de Martin Porter** para utilizar una sola forma del termino en la representacion
  - P.ej: usar 'comput' para computer, computing, computable, and computation.
- Para los **sinonimos de un termino** , se puede utilizar un termino por concepto.
  - No es tan simple pues una palabra puede significar diferentes cosas segun su contexto.
  - **WordNet**: agrupa palabras en conjuntos de sinónimos llamados **synsets**.
    - Los synsets se dividen en **categorías**: sustantivo, verbo, adjetivo y adverbio.
    - Dentro de categoría los synsets se organizan usando **relaciones** clase/subclase (o ES).
- A veces interesa recolectar **entidades** en lugar de términos.
  - P.ej: hechos, eventos, lugares, relaciones, etc.
  - Se extrae contenido estructurado a partir de texto.
  - Uso de **enfoques basados en reglas** con expresiones regulares o gramática, uso de sinónimos.
  - Se usan técnicas de análisis sintáctico o correspondencia de patrones.

Cuando tenemos una coleccion de documentos, queremos poder buscar las ocurrencias de un termino en cada documento. La solucion mas "sencilla" es por cada documento tener un listado de los terminos que contiene (cada uno con alguna propiedad como su frecuencia, peso, etc.)

<b>Documento</b>	<b>terminos</b>
identificador	lista de terminos

Esto resulta en que un termino puede aparecer multiples veces en la tabla, ya que puede estar contenido en n documentos.

## Indice invertido

En un indice invertido almacena para **cada terminio** los identificadores de los documentos donde aparece el termino y alguna informacion adicional (posicion donde aparece para relevancia basada en proximidad, frecuencia, peso, etc.)

<b>Termino</b>	<b>apariciones</b>
nombre	[(idDoc, info extra)]

La lista de un termino puede requerir varios bloques de disco. Para la eficiencia de acceso, se puede tener esa lista en un conjunto consecutivo de bloques. E incluso tener un arbol B+ de indice para mapear cada termino a su lista invertida asociada.

<p><b>Document 1</b></p> <p>This example shows an example of an inverted index.</p>	
<p><b>Document 2</b></p> <p>Inverted index is a data structure for associating terms to documents.</p>	
<p><b>Document 3</b></p> <p>Stock market index is used for capturing the sentiments of the financial market.</p>	

ID	Term	Document: position
1.	example	1:2, 1:5
2.	inverted	1:8, 2:1
3.	index	1:9, 2:2, 3:3
4.	market	3:2, 3:13

## Construcción de índices invertidos:

1. Extraer el vocabulario (términos) de los documentos de la colección
2. Armar estadísticas para cada documento dependiendo del modelo usado.
3. Invertir el stream de documentos con sus términos en un stream de términos y sus documentos.
  - Aquí se puede agregar información adicional como frecuencias de términos, posiciones de términos y pesos de términos.

## Busqueda de doc a partir de una consulta

1. Búsqueda de vocabulario: cada termino de la consulta se busca en el vocabulario, los terminos se pueden ordenar lexicograficamente
2. Retorno de la informacion de documentos: se retorna la info del doc para cada termino
3. Manipulacion/tratamiento de la informacion retornada: la informacion de cada doc es procesada para incorporar las distintas formas de logica de consulta.

### Tratamiento de consultas booleanas:

- **Suposiciones:** La consulta involucra  $n$  términos.  $S_i$  es conjunto de identificadores de documentos donde aparece término  $i$  ( $i$  en  $\{1, \dots, n\}$ ).
- **Operación and:** Los documentos deseados son:  $S_1 \cap S_2 \cap \dots \cap S_n$ .
- **Operación or:** Los documentos deseados son:  $S_1 \cup S_2 \cup \dots \cup S_n$ .
- **Operación not:** Sea  $t_i$  término  $i$ . Explicamos  $\text{not } t_i$ . Sea  $S$  el conjunto de todos los identificadores de documentos. Podemos eliminar los documentos que contienen término  $i$  haciendo:  $S - S_i$ .





**Otra forma de procesar operación and:** se retornan los documentos conteniendo al menos uno de los términos de la consulta, pero ordenados por su medida de relevancia.

## Medición de la eficacia (relevancia) de los resultados

Los SRI soportan solo retorno aproximado.

Se pueden dar las siguientes situaciones:

- **Falsos negativos:** algunos documentos relevantes pueden no ser retornados.
- **Falsos positivos:** algunos documentos irrelevantes pueden ser retornados.
- Los documentos relevantes retornados se llaman **verdaderos positivos** y los resultados irrelevantes no retornados se llaman **verdaderos negativos**.

		Relevant?	
		Yes	No
Retrieved?	Yes	 Hits TP	 False Alarms FP
	No	Misses FN 	Correct Rejections TN 

Existen métricas relevantes para medir el desempeño

- Precision: proporcion de los documentos relevantes retornados en el total de documentos retornados por la consulta.
- Cobertura: proporcion de los documentos relevantes retornados en el total de documentos relevantes en la BD (retornados y no retornados)

+cobertura -> -precision

Poder definir que documentos son realmente relevantes implica comprender el lenguaje natural y comprender el proposito de una consulta. Se termina creando consultas y etiquetando

manualmente documentos como relevantes e irrelevantes.

- **Problema:** Alta precisión es lograda casi siempre a expensas de cobertura y recíprocamente.
- La medición **F-score** es usada como una medida que combina precisión y cobertura para comparar distintos conjuntos de resultados (es el promedio armónico de los dos números).

- $$F = \frac{2pr}{p+r}$$
 equivalentemente: 
$$F = \frac{2}{\frac{1}{p} + \frac{1}{r}}$$
  - F tiende a ser cercano al más pequeño de p y r. Para F alto tanto r como p deben ser altos.
- 

## Lucene

Es una maquina de busqueda e indexado popular en la industria y academia

- Los documentos no estructurados pasan por un proceso de indexado previo a estar disponibles para consultas.
- Un documento de lucene se forma de **campos** que tienen un **tipo** (binario,numerico,texto)
  - texto puede ser no tokenizado o un stream de simbolos
- Posee una **API de consultas**
  - Las consultas retornan una lista ordenada de docs por rango, usando variante de [TF-IDF](#) para dar valor a los docs.
  - La API de consultas es configurable, se puede crear consultas por expr booleanas, regex o proximidad.
- Usa [modelo de espacio vectorial](#)