

TEMA 4. MODELO DE ESPACIO VECTORIAL

Contenidos

1. Conceptos previos.

- 1.1 Recuperación ordenada.
- 1.2 Puntuación de documentos.
- 1.3 Frecuencia del término y su pesado.
- 1.4 Frecuencia de documento.

2. Modelo de Espacio Vectorial

- 2.1 Documentos como vectores
- 2.2 Consultas como vectores
- 2.3 Medir la similitud entre dos documentos en el espacio vectorial.
- 2.4 Esquemas de pesado.

Hasta ahora, todas las consultas que hemos resuelto son booleanas, los documentos pertenecen al resultado o no.

Este es un modelo de recuperación de información muy útil para usuarios expertos con una comprensión muy precisa de sus necesidades y de la colección.

Este planteamiento no suele ser útil para la mayoría de los usuarios de un sistema de recuperación de información. En general no son capaces de formular una consulta lo suficientemente precisa haciendo uso de operadores booleanos y a menudo los resultados contienen 'demasiados' o 'demasiado pocos' documentos. Esto es particularmente cierto cuando la colección de documentos es la web.

En este tema se estudiará el modelo de espacio vectorial como alternativa al modelo booleano.

Tanto las consultas como los documentos se representarán en un mismo espacio vectorial, ello nos permitirá medir similitudes entre documentos y entre consulta y documentos. Para ello se definirán previamente ciertas estrategias de pesado.

Bibliografía

A Introduction to Information Retrieval:

Christopher D. Manning, Prabhakar Raghavan, Hinrich Schütze.

Cambridge University Press, 2009.

Capítulo 6

Speech and Language Processing: International Version, 2/E.

Daniel Jurafsky, James H. Martin.

Pearson International Edition, 2009. ISBN-10: 0135041961.

Capítulo 23



1. CONCEPTOS PREVIOS


- 1.1 Recuperación ordenada.
- 1.2 Puntuación de documentos.
- 1.3 Frecuencia del término y su pesado.
- 1.4 Frecuencia de documento.

.

1.1 Recuperación ordenada


Dada una consulta q y una colección C :

Q : expresiones y operadores correspondientes a un lenguaje de consultas específico,

Recuperación booleana 

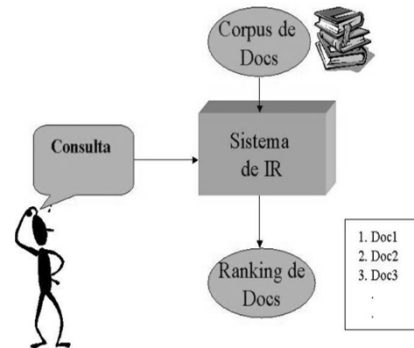
El sistema devuelve: un conjunto de documentos que satisfacen la expresión lógica correspondiente a la consulta (Las consultas Booleanas a menudo obtienen muy pocos o demasiados resultados)

Q : texto plano (free text), forma asociada a la recuperación ordenada.

Recuperación ordenada 

El sistema devuelve: un conjunto de documentos de la colección ordenado según relevancia.

- Objetivo: devolver en orden los documentos que con mayor probabilidad sean los más útiles para el usuario.
- Cuando el sistema genera un conjunto de resultados ordenados, el tamaño del conjunto resultante no es un problema:
 - Mostramos del resultado sólo el tope K (≈ 10).
 - No abrumamos al usuario
 - Premisa: el algoritmo de ordenación funciona.



1.2 Puntuación (score) de documentos

¿Cómo ordenar documentos respecto a una consulta?

- Asignar una puntuación – dentro del rango $[0, 1]$ – para cada documento es la base de la Recuperación ordenada de documentos.
- Esta puntuación mide cómo de bien “se emparejan” el documento y la consulta.

Queremos que el sistema devuelva los documentos que con mayor verosimilitud vayan a ser útiles al usuario.

Emparejamiento Consulta-documento

Asignar una puntuación a un par (consulta, documento):
se basará en el peso de los términos de la consulta q en el documento d .

Para un término t de la consulta q :

- Si t no se encuentra en $d \rightarrow$ la puntuación será 0.
- ¿Cuanto más frecuente sea el t de la consulta q en el documento d más alta será la puntuación del d ?
- Estudiaremos alternativas para esto

Métricas para puntuar dos conjuntos A y B

- **Solapamiento:**

$$\text{overlap}(A,B) = |A \cap B|$$

no está normalizado (prima los documentos mas grandes)

- **Coeficiente de Jaccard:**

$$\text{jaccard}(A,B) = |A \cap B| / |A \cup B|$$

$$\text{jaccard}(A,A) = 1$$

$$\text{jaccard}(A,B) = 0 \quad \text{si } A \cap B = 0$$

- A y B no tienen porqué ser del mismo tamaño.
- siempre se asigna un valor entre 0 y 1.

Overlap tiene el problema de que no está normalizado: documentos más grandes tendrán mayor opción de ser elegidos al tener mayor número de palabras.

Donde $|A \cap B|$ representan el número de términos que coinciden.

Ejercicio#1: Calcula la puntuación correspondiente al par consulta-documento calculado según overlap y jaccard,

Consulta: *días de lluvia en primavera*

Documento 1: *resbaló en un día de lluvia*

Documento 2: *la lluvia ácida es muy perjudicial para los árboles, sobre todo en la primavera que es cuando florecen.*

Ejercicio#1_Solución:

Calcula la puntuación correspondiente al par consulta-documento calculado según overlap y jaccard,

Consulta: **días de lluvia en primavera**

Documento 1: **resbaló en un día de lluvia**

Documento 2: **la lluvia ácida es muy perjudicial para los árboles, sobre todo en la primavera que es cuando florecen.**

$$\text{overlap}(q, d1) = |Q \cap D1| = 2$$

$$\text{overlap}(q, d2) = |Q \cap D2| = 2$$

$$\text{jaccard}(q, d1) = |Q \cap D1| / |Q \cup D1| = 2/4$$

$$\text{jaccard}(q, d2) = |Q \cap D2| / |Q \cup D2| = 2/7$$

Nótese que $|Q \cap D|$ representa el número de términos que coinciden.

Problemas con puntuación usando Jaccard:

- No considera la frecuencia de aparición de cada término en los documentos (*term frequency*)
- Los términos “raros” son una información muy relevante a la hora de elegir un documento. Jaccard no tiene en cuenta esa información, para esta medida todos los términos tienen la misma consideración.

Necesitamos una estrategia más elaborada para normalizar las longitudes.

Recordemos...

Matriz Incidencia Binaria término-documento

Cada documento (término) se representa por un vector binario $\in \{0,1\}^V$

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Ejemplo.

Consulta: Brutus AND Caesar AND NOT Calpurnia,
110100 AND 110111 AND 101111 (Compl. De Calpurnia)= 100100

Respuesta: “*Antony and Cleopatra* “ y “*Hamlet*”

Todos los términos se consideran igual de importantes para evaluar su relevancia sobre una consulta.

Matriz contador término-documento

Considera el número de ocurrencias de un término en un documento:

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

- Cada documento se representa por un vector contador en \mathbb{N}^v : una columna de la matriz
- La frecuencia de t en d es considerado para evaluar su relevancia sobre una consulta (Mod. Boleano: si $f > 0$ peso=1 sino peso= 0).

Una primera forma de incluir información sobre la importancia de las palabras en el documento es considerar su frecuencia de aparición.

- La representación mediante un vector no considera el orden de las palabras en un documento.
- *Julia es mas valiente que Pedro y Pedro es mas valiente que Julia* representan el mismo vector.
- Esto es conocido como modelo de *Bolsa de Palabras* (*Bag of words*).

En los sistemas conocidos, tal como haremos en este capítulo, no se tiene en consideración el orden en que aparecen las palabras.

1.3 Frecuencia del término($f_{t,d}$) y su pesado($tf_{t,d}$)

- La **frecuencia** del término t en el documento d ($f_{t,d}$): se define como el número de veces que t ocurre en d .

¿Cómo podemos usar f cuando calculamos la puntuación del par consulta-documento?

Un documento con 10 ocurrencias del término es más relevante que un documento con 1 ocurrencia,...PERO NO 10 VECES MÁS.

- La relevancia no se incrementa proporcionalmente con su frecuencia, en caso contrario se primaría los documentos de mayor talla.

La frecuencia de aparición de un término en un documento es un valor que representa la importancia del término a la hora de caracterizar el documento. Existen múltiples variantes en sus definiciones (aunque todas ellas se basan en la misma idea), según el interés en destacar unos u otros aspectos.

El **peso** de la frecuencia del término t en d se representa por $tf_{t,d}$

- El pesado **log** (muy extendido):

$$tf_{t,d} = \begin{cases} 1 + \log_{10} f_{t,d}, & \text{si } f_{t,d} > 0 \\ 0, & \text{otro caso} \end{cases}$$

$$f_{t,d} \rightarrow tf_{t,d}: 0 \rightarrow 0, 1 \rightarrow 1, 2 \rightarrow 1.3, 10 \rightarrow 2, 1000 \rightarrow 4, \dots$$

Es decir, el f es el valor absoluto de la frecuencia de un término en un documento y para que sea una medida comparable con su frecuencia de aparición en otros documentos, que pueden ser de distinto tamaño hay que normalizar/suavizar la frecuencia. A este valor normalizado le llamaremos tf . Existen diversas formas de obtener el tf a partir de f . En las explicaciones posteriores consideraremos que se calcula su logaritmo, de este modo se suaviza la diferencia entre los valores obtenidos.

Puntuación para el par consulta-documento.

- Es la suma de los pesos de los términos t que *aparecen* en ambos q y d :

$$\text{puntuación} = \sum_{t \in q \cap d} (\text{tf}_{t,d})$$

- Con un pesado tomando **log**

$$\text{puntuación} = \sum_{t \in q \cap d} (1 + \log f_{t,d})$$

- La puntuación es 0 si ninguno de los términos de la consulta se encuentra en el documento.

Si sólo consideramos como medida los tf de los términos, la distancia (o puntuación) entre un query y un documento se calcula sumando los tf de los términos (comunes al query y al documento) en el documento. No tiene sentido considerar los tf de los términos del query, porque si el objetivo es obtener los documentos cercanos a un query, la importancia de los términos en el query es la misma para todos los documentos con los que lo comparemos.

1.4. Frecuencia de documento

- Los términos raros son más informativos que los términos frecuentes.
- Considera un término en la consulta que sea raro en la colección (p.e., *diplodocus*) → Un documento conteniendo este término raro es muy probable que sea relevante para la consulta.

Deseable asignar un peso alto para términos raros.

- Los términos frecuentes son menos informativos que los raros (véase las stop words).
- Considera un término en la consulta que sea frecuente en la colección de la industria automovilística (p.e., *coche*)
 - Un documento conteniendo este término es más probable que sea relevante para la consulta que otro que no lo tenga, pero esto no puede ser el único indicador de relevancia.
 - Para términos frecuentes deseamos un peso alto pero más bajo que para los términos raros.

Usaremos la frecuencia de documento (*df*) para capturarlo.

Peso idf

- df_t es la frecuencia de documento de t : el número de documentos de la colección que contienen el término t
 - $df_t \leq N$
 - *siendo N el número total de documentos de la colección.*
- Definimos idf_t (frecuencia de documento inversa de t),

$$idf_t = \log_{10} (N/df_t)$$

idf_t es 0 para los términos que aparecen en todos los documentos

Al igual que con la frecuencia, aquí tenemos un valor absoluto (df) que es el número de documentos de la colección que contienen el término, y vamos a normalizarlo/suavizarlo para luego combinar adecuadamente su valor con el tf . Como lo que se quiere representar es que los términos que aparecen en pocos documentos sean muy relevantes, se calcula la “frecuencia inversa” (N/df), que proporciona valores muy grandes cuando el denominador es pequeño (pocas apariciones), y valores pequeños cuando son términos muy habituales en todos los documentos. Este cálculo se suaviza mediante la función \log .

Ejercicio#2. Calcular idf de los términos (nºdocs $N=1$ millón)

term	df_t	idf_t
calpurnia	1	6
animal	100	4
sunday	1,000	3
fly	10,000	2
under	100,000	1
the	1,000,000	0

$$idf_t = \log (N/df_t)$$

Hay un valor idf para cada término t en una colección.

$\log 1=0$; $\log 10=1$; $\log 100=2$; $\log 1.000=3$; $\log 10.000=4$; $\log 100.000 = 5$; $\log 1.000.000=6....$

Efecto de idf sobre la ordenación

- idf afecta a la ordenación de documentos para consultas con al menos dos términos.

- Ejemplo.

Q: persona caprichosa,

las ocurrencias de caprichosa contarán mucho más en la ordenación final que las ocurrencias de persona.

Pesado $\text{tfidf}_{t,d}$

El peso **tfidf** (nombrado alternativamente tf.idf o tf-idf) de un término t es el producto de su peso tf y su peso idf .

$$w_{t,d} = \text{tf}_{t,d} \times \text{idf}_t = (1 + \log(f_{t,d})) \times \log(N / df_t)$$

tfidf $_{t,d}$ asigna a un término t un peso en el documento d , tal que es:

1. Alto cuando t sucede muchas veces dentro de un pequeño número de documentos.
2. Bajo cuando t sucede muy pocas veces, o cuando sucede en muchos documentos.
3. El más bajo cuando t sucede en todos los documentos.

2. MODELO DE ESPACIO VECTORIAL

- 2.1 Documentos como vectores
- 2.2 Consultas como vectores
- 2.3 Medir la similitud entre dos documentos en el espacio vectorial.
- 2.4 Esquemas de pesado.

Modelo de Espacio Vectorial

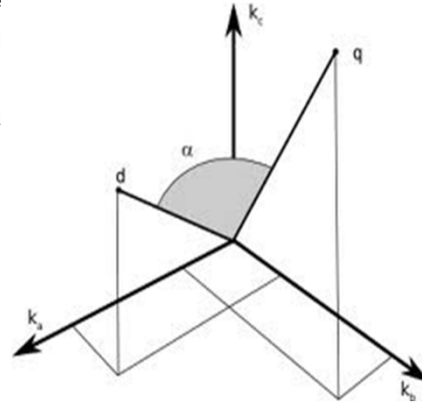
Las consultas y los documentos se representan como vectores en un espacio V-dimensional común.

$V = n^{\circ}$ de términos diferentes de la colección

Un eje por cada término

Modelo estándar en RI para:

- puntuación de documentos sobre una consulta,
- clasificación de documentos,
- agrupación de documentos.



Cada documento, o query, es un vector de V dimensiones.

Ejemplo:

Receta de pollo frito (d_j): aparecen los términos *chicken*, *fried*, *oil* y *pepper* con frecuencias 8, 2, 7 y 4 respectivamente.

Receta de pollo escalfado (d_k): aparece el término *chicken* 6 veces

Consulta (q): *fried*, *chicken*

- Representamos las recetas y la consulta como vectores:

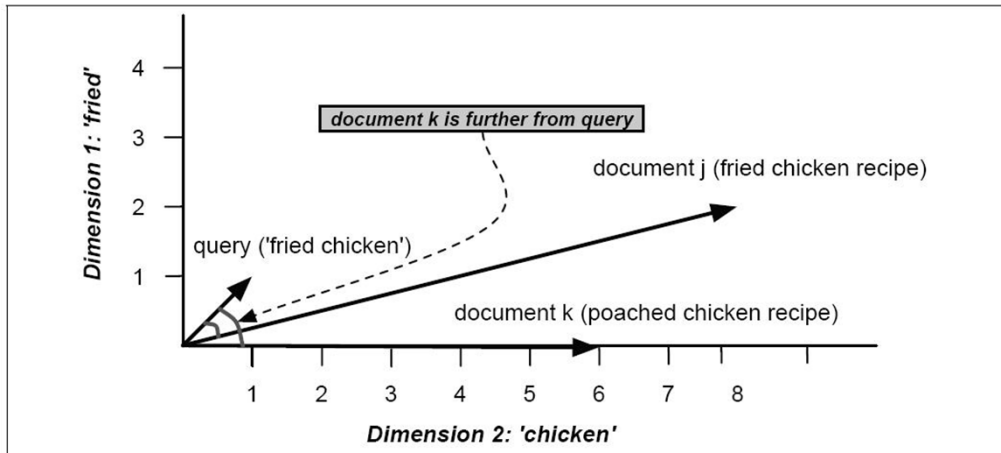
$$\vec{q} = (1, 1, 0, 0) \quad \vec{d}_j = (8, 2, 7, 4) \quad \vec{d}_k = (6, 0, 0, 0)$$

- La colección se representa como una matriz de términos:

$$A = \begin{pmatrix} 8 & 6 \\ 2 & 0 \\ 7 & 0 \\ 4 & 0 \end{pmatrix}$$

Para facilitar la explicación, y no usar números reales, se considera en este ejemplo que la representación de un documento está dada por un vector de 4 elementos (uno por cada término del vocabulario), donde el peso de cada término está dado por su frecuencia absoluta de aparición en el documento.

Documentos y consulta como puntos en el mismo espacio vectorial



Como cada documento o query es un vector, la similitud vendrá dada por alguna medida que nos diga lo “ceranos” que son entre sí los vectores.

2.1 Documentos como vectores

- Así tenemos un espacio vectorial de $|V|$ -dimensiones.
- Los términos son los ejes del espacio.
- Los documentos son puntos o vectores en este espacio.
- Muy alta-dimensión: decenas de millones de dimensiones cuando se aplica a un motor de búsqueda web.
- Son vectores muy dispersos, la mayoría de las entradas son cero.

Matriz binaria → contador → pesos

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	5.25	3.18	0	0	0	0.35
Brutus	1.21	6.1	0	1	0	0
Caesar	8.59	2.54	0	1.51	0.25	0
Calpurnia	0	1.54	0	0	0	0
Cleopatra	2.85	0	0	0	0	0
mercy	1.51	0	1.9	0.12	5.25	0.88
worser	1.37	0	0.11	4.15	0.25	1.95

Cada documento se representa por un vector con los valores de pesos $tf \cdot idf \in \mathbb{R}^{|V|}$

Es decir, para representar un documento (o query) usamos toda la información que hemos comentado anteriormente (term frequency tf y inverse document frequency idf) combinándolas mediante una función, que en nuestro caso será simplemente la multiplicación. Por tanto un documento es un vector de k dimensiones donde en cada posición se almacena el $tf \cdot idf$ del correspondiente término.

2.2 Consultas como vectores

- Paso1: representar las consultas como vectores en el espacio.
- Paso 2: ordenar los documentos de acuerdo a su proximidad a la consulta en este espacio.
 - proximidad = similitud de vectores
 - proximidad \approx inversa de distancia
- Objetivo: ordenar los documentos más relevantes para la consulta de mayor a menor similitud

2.3 Medir la similitud entre dos documentos en el espacio vectorial.

- Uso del ángulo en vez de la distancia: El ángulo entre los dos documentos es 0 cuando se alcanza la similitud máxima.
- La distancia euclídea entre dos documentos “semánticamente” similares puede ser muy alta si son de longitudes muy diferentes. En cambio su ángulo se acercará a 0.
- **Objetivo:** Ordenar los documentos de acuerdo al ángulo con la consulta.

Una primera opción podría ser utilizar la distancia Euclídea entre los puntos que representan los documentos.

Por ejemplo, dado un query Q y un documento D, $d(D,Q)=\text{SQRT}(\sum (d_i - q_i)^2)$. Sin embargo esta medida no es la más adecuada, porque depende mucho de la longitud del documento (ya que los más largos tendrán más alta el tf de sus términos).

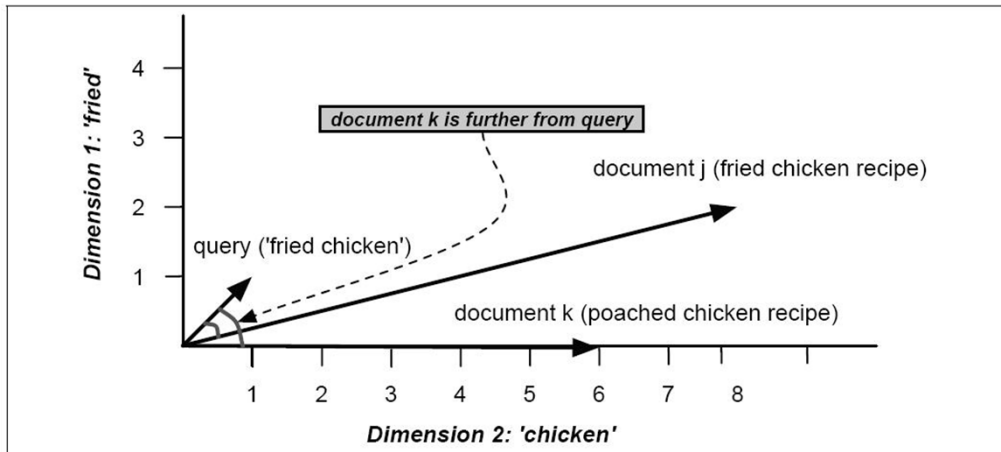
Por ejemplo (considerando por simplicidad sólo la frecuencia en valor absoluto como el peso de cada término), un documento representado por el vector $d_1 = (4,0,2,3)$ y otro documento $d_2 = (8,0,4,6)$ tendrán una distancia euclídea de $\text{SQRT}(16+4+9)$. Sin embargo son casi equivalentes, ya que el d_1 contiene 4 veces el primer término, 2 veces el tercero y 3 veces el cuarto, mientras que d_2 contiene los mismos términos sólo que el doble de veces.

Por ello es mejor considerar como medida de similitud el ángulo que forman los vectores (en este caso 0).

Desde ángulos a cosenos

- Las dos nociones siguientes son equivalentes:
 - Ordenar documentos en orden creciente del ángulo entre la consulta y el documento
 - Ordenar documentos en orden decreciente del coseno entre la consulta y el documento ($\cos(0)=1$)
- Coseno es una función monótona decreciente para el intervalo $[0^\circ, 180^\circ]$

¿Qué documento es más similar a la consulta? Razonar la respuesta respecto al ángulo y el coseno.



Consideremos los documentos Receta de pollo escalfado (6,0,0,0) y Receta de pollo frito (8,2,7,4) representado en un espacio de dos dimensiones (fried y chicken)

La consulta fried chicken (1,1,0,0) también se representa en este plano

Con esta representación, el ángulo entre consulta y documento da una idea sobre la similitud entre ellos, p.e. más cerca el documento j (fried chicken). Realmente se utiliza la métrica del coseno en lugar del ángulo real. Cuando dos documentos son idénticos el coseno del ángulo que forman es 1 mientras que si son ortogonales el coseno del ángulo que forman es 0 (no tienen términos comunes). Esta medida también se puede ver como el producto escalar normalizado de los dos vectores.

Longitud-Normalización

Experimento: Si a un documento d le añadimos su propio contenido obteniendo el documento d' :

- “Semánticamente” d y d' tienen el mismo contenido.
- La distancia entre los dos documentos puede ser grande por la distinta longitud de ambos documentos.

Necesitamos normalizar los vectores correspondientes a ambos documentos en función de su longitud.

Longitud-Normalización

- Un vector puede normalizarse dividiendo cada uno de sus componentes por su longitud – para esto usamos la norma L_2 de dicho vector:

$$\|\vec{x}\|_2 = \sqrt{\sum_i x_i^2}$$

- De esta forma, los documentos d y d' (d añadido a sí mismo) serían vectores idénticos después de la Longitud-normalización.
 - documentos largos y cortos ahora tienen pesos comparables.

Coseno (consulta, documento)

producto escalar

Vectores normalizados

$$\cos(\vec{q}, \vec{d}) = \frac{\vec{q} \cdot \vec{d}}{|\vec{q}| |\vec{d}|} = \frac{\vec{q}}{|\vec{q}|} \cdot \frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|k|} q_i d_i}{\sqrt{\sum_{i=1}^{|k|} q_i^2} \sqrt{\sum_{i=1}^{|k|} d_i^2}}$$

q_i representa el peso del término i en la consulta q (w_{tq})

d_i representa el peso del término i en el documento d (w_{td})

Esta fórmula es la que calcula el coseno del ángulo entre dos vectores, y se obtiene a partir del producto escalar de vectores.

Coseno para vectores Longitud-normalizados

Para vectores Longitud-normalizados, la similitud del coseno es simplemente el producto punto (o producto escalar):

$$\cos(\vec{q}, \vec{d}) = \vec{q} \bullet \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

Similitud coseno entre 3 documentos

Cómo de similares
son las novelas

SaS: *Sense and
Sensibility*

PaP: *Pride and
Prejudice*

WH: *Wuthering
Heights?*

term	SaS	PaP	WH
affection	115	58	20
jealous	10	7	11
gossip	2	0	6
wuthering	0	0	38

Frecuencia de términos (conteo)

Para simplificar no hemos usado el idf

Similitud coseno entre 3 documentos cont.

$$\cos(\vec{q}, \vec{d}) = \vec{q} \cdot \vec{d} = \sum_{i=1}^{|\mathcal{V}|} q_i d_i$$

Pesado log frecuencia

term	SaS	PaP	WH
affection	3.06	2.76	2.30
jealous	2.00	1.85	2.04
gossip	1.30	0	1.78
wuthering	0	0	2.58

Long-normalización de vectores

term	SaS	PaP	WH
affection	0.789	0.832	0.524
jealous	0.515	0.555	0.465
gossip	0.335	0	0.405
wuthering	0	0	0.588

$\cos(\text{SaS}, \text{PaP}) \approx 0.789 \times 0.832 + 0.515 \times 0.555 + 0.335 \times 0.0 + 0.0 \times 0.0 \approx 0.94$
 $\cos(\text{SaS}, \text{WH}) \approx 0.79$
 $\cos(\text{PaP}, \text{WH}) \approx 0.69$

A la hora de calcular la similitud se pueden considerar o descartar algunos valores, y usar distintas formas de calcular los pesos.

En el ejemplo, en la parte de la izquierda están los tf de los términos, calculándolos como $1 + \log(f)$

Por ejemplo $1 + \log(115) = 3.06$ $1 + \log(10) = 2$ $1 + \log(2) = 1.30$

En la parte izquierda están esos valores normalizados (es decir, divididos por su norma). Por ejemplo,

$3.06 / \text{SQRT}(3.06^2 + 2^2 + 1.30^2 + 0^2) = 0.789$

Cálculo puntuación coseno

ALGORITMO PUNTUACIÓN_COSENO (q)

1. $Scores[N] \leftarrow 0$ /*N es el número de documentos
2. Inicializa $Length[N]$
3. **para cada** término t de q
4. **hacer** calcular $w_{t,q}$ **and** recorrer postings list para t
5. **para cada** par($d, tf_{t,d}$) in postings list
6. **hacer** $Scores[d] += w_{t,d} \times w_{t,q}$
7. **para cada** d
8. **hacer** $Scores[d] = Scores[d] / Length[d]$
9. **devuelve** los K componentes mejor puntuados de $Scores[]$

La diapositiva muestra el algoritmo básico para calcular las puntuaciones en un modelo de espacio vectorial.

El vector *Length* contiene las longitudes (factores de normalización) de cada uno de los N documentos, mientras que el vector *Scores* contiene las puntuaciones de cada uno de los documentos para la consulta q . El vector *Length* se usará en un paso final para la normalización.

Para cualquier documento d , la similitud coseno $v(q) \cdot v(d)$ es la suma pesada, sobre todos los términos de la consulta q , de los pesos de esos términos en d .

Navegamos a través de las postings en el índice invertido para los términos de q , acumulando la puntuación total de cada documento, en gran medida como en el procesamiento de una consulta booleana, excepto que asignamos una puntuación positiva a cada documento que aparece en alguno (OR) de los postings que se atraviesa. Mantenemos un valor *idf* para cada término del diccionario y un valor *tf* para cada entrada de la posting list (estos valores se habrán calculado en la construcción del índice invertido y se guardarán en el mismo, nótese que son valores para los documentos, y son independientes de las consultas). Este esquema calcula una puntuación para cada documento en la posting de cualquiera de los términos de la consulta; el número total de dichos documentos puede ser considerablemente menor que N.

Teniendo en cuenta estas puntuaciones, el último paso antes de presentar los resultados a un usuario es seleccionar los K documentos con mayor puntuación. Si bien se podría reordenar el conjunto completo de puntuaciones, un mejor enfoque consiste en utilizar un montículo (heap) para recuperar sólo la parte superior de K documentos en orden.

Cálculo puntuación coseno

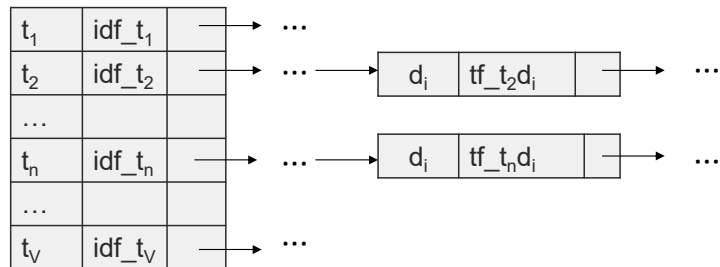
Scores

d_1	d_2	...	d_i	...	d_N
0	0		0		0

q

t_1	0
t_2	2
...	
t_n	1
...	
t_V	0

Índice invertido



El vector *Scores* se inicializa a 0

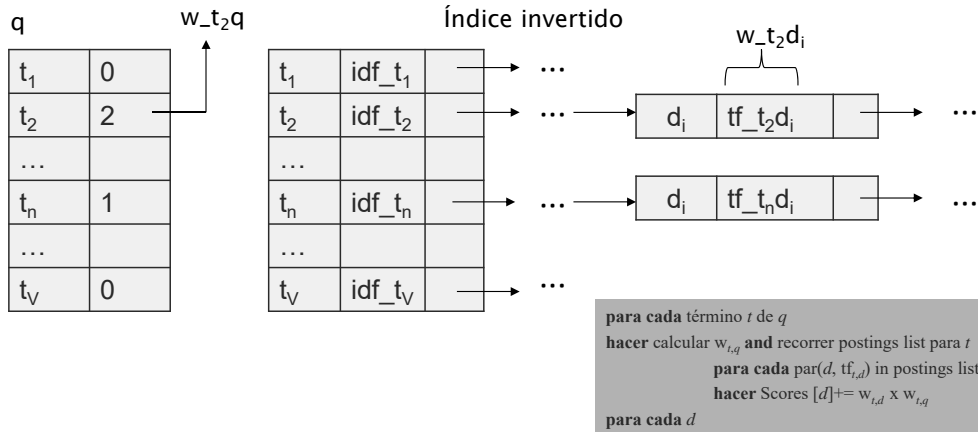
El vector que representa la consulta *q* contiene las frecuencias de ocurrencia de cada término del vocabulario en la consulta

El índice invertido contiene el diccionario, con los términos, el idf de cada uno y un puntero a su postings list. Cada registro de la postings list contiene, entre otros posibles campos, el peso de la frecuencia del término en el documento.

Cálculo puntuación coseno

Scores

d_1	d_2	...	d_i	...	d_N
			$w_{t_2q} \times w_{t_2d_i}$		

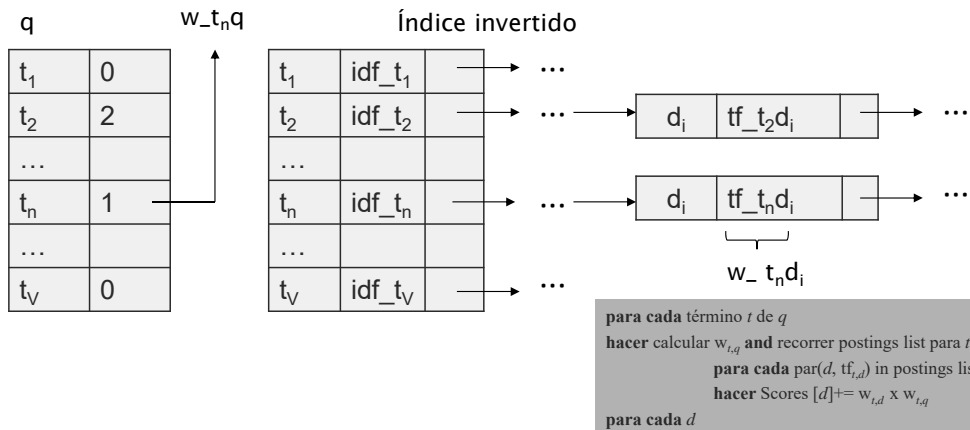


En el proceso del término t_2 , se calcula su peso en la consulta q , y para cada documento de su postings list, por ejemplo d_i , se calcula el peso del término en el documento y se añade el correspondiente $tf.idf$ a lo acumulado en la posición i del vector *Scores*.

Cálculo puntuación coseno

Scores

d_1	d_2	...	d_i	...	d_N
			$(w_{t_2q} \times w_{t_2d_i}) + \dots + (w_{t_nq} \times w_{t_nd_i})$		



Cuando se procesa el término t_n , se calcula su peso en la consulta y se recorre su postings list. Como contiene también el documento d_i , se calcula el peso del término en el documento y se añade el $tf.idf$ a la posición i en el vector *Scores*.

2.4 Esquemas de pesado.

Esquemas de pesado se representan por *ddd.qqq*:

- *ddd* pesado del vector de documentos.
- *qqq* pesado del vector de consulta

(1ª letra: frecuencia término; 2ª letra: Frecuencia documento; 3ª Letra: Normalización)

Term frequency		Document frequency		Normalization	
n (natural)	$tf_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1 + \log(tf_{t,d})$	t (idf)	$\log \frac{N}{df_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2 + w_2^2 + \dots + w_k^2}}$
a (augmented)	$0.5 + \frac{0.5 \times tf_{t,d}}{\max_t(tf_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-df_t}{df_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases} 1 & \text{if } tf_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$			b (byte size)	$1/CharLength^\alpha$, $\alpha < 1$
L (log ave)	$\frac{1 + \log(tf_{t,d})}{1 + \log(\text{ave}_{t \in d}(tf_{t,d}))}$				

Inc.Itc:

- el vector de documentos tiene una frecuencia de términos log-pesado, no idf y coseno normalizado
- el vector de consulta usa frecuencia de términos log-pesado, idf y coseno normalizado.

En esta tabla representamos diversas alternativas para calcular el tf, el idf, y el mecanismo de normalización. Se codificará con tres letras el mecanismo de pesado del documento y con otras tres letras el mecanismo de pesado del query. Lo habitual, tal como se ha explicado en las transparencias anteriores será usar l (logaritmo de la frecuencia) t (el idf tal como se ha explicado) y c (normalización). Sin embargo habrá casos en que no es necesario alguno de estos cálculos, Por ejemplo si queremos calcular la similitud entre un query y dos documentos, solo hace falta que calculemos el idf en el query, porque el idf es el mismo en todas partes (está ligado al término).

Ejercicio#3. Calcular la puntuación del documento respecto a la consulta suponiendo una colección de 1.000.000 de documentos con un esquema de pesado **Inc.Itc**

Inc.Itc:

- el vector de documentos tiene una frecuencia de términos log-pesado, no idf y coseno normalizado
- el vector de consulta usa frecuencia de términos log-pesado, idf y coseno normalizado.

Term	Consulta						Documento				Producto
	$f_{t,q}$	$tf_{t,q}$	df_t	idf_t	$w_{t,q}=tf_{t,q}idf_t$	L-Normaliz	$f_{t,d}$	$tf_{t,d}$	$w_{t,d}=tf_{t,d}$	L-Normaliz	
auto	0		5000				1				
mejor	1		50000				0				
coche	1		10000				1				
seguro	1		1000				2				

Fijaos que se pide una medida con Inc.Itc, es decir calcular el peso de los elementos del documento considerando sólo su tf normalizada (sin multiplicar por el idf), y los pesos de los elementos del vector que representa el query con el producto $tf.idf$ normalizado.

Ejercicio#3_Solución. Calcular la puntuación del documento respecto a la consulta suponiendo una colección de 1.000.000 de documentos con un esquema de pesado **Inc.ltc**

Term	Consulta						Documento				Producto	
	$f_{t,q}$	$tf_{t,q}$	df_t	idf_t	$w_{t,d}=tf \times idf$	L-Normaliz	$f_{t,d}$	$tf_{t,d}$	$w_{t,d}=tf \times idf$	L-Normaliz		
auto	0	0	5000	2,3	0	0,00	1	1	1	0,52	0,00	
mejor	1	1	50000	1,3	1,3	0,34	0	0	0	0,00	0,00	
coche	1	1	10000	2	2	0,52	1	1	1	0,52	0,27	
seguro	1	1	1000	3	3	0,78	2	1,3	1,3	0,68	0,53	

$$\text{Puntuación} = \text{Cos}(q,d) = 0 + 0 + 0.27 + 0.53 = 0.8$$

Ejercicio#4. Similitud entre 4 documentos esquema pesado **Inc** (log-noidf-coseno)

Similitud entre 4 documentos esquema pesado **Inc**

Pesado Log

L-normalizacion

term	SaS	PaP	WH	WH+WH	SaS	PaP	WH	WH+WH	SaS	PaP	WH	WH+WH
affection	115	58	20	40	3,06	2,76	2,30	2,60	0,789	0,832	0,524	0,522
jealous	10	7	11	22	2,00	1,85	2,04	2,34	0,515	0,555	0,465	0,470
gossip	2	0	6	12	1,30	0,00	1,78	2,08	0,335	0,000	0,405	0,417
wuthering	0	0	38	76	0,00	0,00	2,58	2,88	0,000	0,000	0,588	0,578

$$\cos(\text{SaS}, \text{PaP}) \approx 0,94$$

$$\cos(\text{SaS}, \text{Wh}) \approx 0,79$$

$$\cos(\text{SaS}, \text{Wh+Wh}) \approx 0,79$$

$$\cos(\text{PaP}, \text{Wh}) \approx 0,69$$

$$\cos(\text{PaP}, \text{Wh+Wh}) = 0,69$$

$$\cos(\text{Wh}, \text{Wh+Wh}) \approx 1,00$$

¿Qué podemos observar a la
vista de los resultados?

Resumen

- Representar la consulta como un vector tfidf pesado
- Representar cada documento como un vector tfidf pesado
- Calcular la puntuación similitud del coseno para el vector de la consulta y para el vector de cada documento
- Ordenar documentos respecto a la consulta por puntuación
- Devolver los K (p.e. $K=10$) primeros documentos al usuario.