Information retrieval

Flexible querying methods

Clovis Galiez

Laboratoire Jean Kuntzmann, Statistiques pour les sciences du Vivant et de l'Homme

October 23, 2024

Today's outline

- Short summary of last lecture
- tf-idf
- Querying in the vector-space model
- (Latent semantics)

What to remember from last time?

Remember...

What are the main points you remember from last lecture?

What to remember from last time?

Remember...

What are the main points you remember from last lecture?

- Web IR is split in distinct steps:
  - Gathering and indexing data from the web (**crawling**)
  - Retrieving documents relevant to a query
  - Ranking the valid answers according to relevance
- The involved data is **big**

  Need efficient representation and algorithms

Drawbacks of the boolean querying systems

What drawback for boolean querying?

Drawbacks of the boolean querying systems

What drawback for boolean querying?

The boolean queries are not flexible

Query: `result elections United States`
Doc title: "White House election: live results!"

Drawbacks of the boolean querying systems

What drawback for boolean querying?

The boolean queries are not flexible

Query: `result elections United States`
Doc title: "White House election: live results!"

With a good stemming and tokenization, we will match `result` and `election`... we miss the match between `United States` and `White House` :-/

Drawbacks of the boolean querying systems

What drawback for boolean querying?

The boolean queries are not flexible

Query: `result elections United States`
Doc title: "White House election: live results!"

With a good stemming and tokenization, we will match `result` and `election`... we miss the match between `United States` and `White House` :-/

The boolean querying does not rank

When querying using a boolean querying system, the output is binary.
$\rightarrow$Unable to distinguish the relevant matches from non-relevant ones.

The vector space model and the latent semantics

Representing documents as vectors in $\mathbb{R}^T$

From binary presence/absence...

	tok 1	tok 2	tok 3	tok 4	tok 5	...
	election	president	crazy	united	United States	...
doc 1	1	1	0	0	1	...
doc 2	0	1	1	0	1	...
doc 3	1	1	1	0	1	...
...	...	...	...	...	...	...

Representing documents as vectors in $\mathbb{R}^T$

...to real vector space.

	tok 1	tok 2	tok 3	tok 4	tok 5	...
	election	president	crazy	united	United States	...
doc 1	0.01	0.02	0	0	0.006	...
doc 2	0	0.013	0.001	0	0.001	...
doc 3	0.0031	0.008	0.0043	0	0.0021	...
...	...	...	...	...	...	...

What numbers can be useful here ?

Not every term is informative

How do you quantify information according to Shannon theory?

Not every term is informative

How do you quantify information according to Shannon theory?

Example: which book are you talking about?

Piece of information	Probability	Information content
"the" is frequent	~ 1	Low
"Zarathustra" is frequent	~ 0	High

Not every term is informative

How do you quantify information according to Shannon theory?

Example: which book are you talking about?

Piece of information	Probability	Information content
"the" is frequent	~ 1	Low
"Zarathustra" is frequent	~ 0	High

Exercise

I throw a die. What is the more informative:

- the outcome is even
- the outcome is ≥ 5

Information

Requirements for information measure:

- information of an event depends on its probability: $I(e) = f(P(e))$

Information

Requirements for information measure:

- information of an event depends on its probability: $I(e) = f(P(e))$
- it should be contravariant with the probability:

$$P(e_1) < P(e_2) \Rightarrow I(e_1) > I(e_2)$$

Information

Requirements for information measure:

- information of an event depends on its probability: $I(e) = f(P(e))$
- it should be contravariant with the probability:

$$P(e_1) < P(e_2) \Rightarrow I(e_1) > I(e_2)$$

- when e_1 and e_2 are independent, we would like that:

$$I(e_1 \& e_2) = I(e_1) + I(e_2)$$

Information

Requirements for information measure:

- information of an event depends on its probability: $I(e) = f(P(e))$
- it should be contravariant with the probability:

$$P(e_1) < P(e_2) \Rightarrow I(e_1) > I(e_2)$$

- when e_1 and e_2 are independent, we would like that:

$$I(e_1 \& e_2) = I(e_1) + I(e_2)$$

If we moreover ask for f to be continuous, there is only one possible class of functions: $-log_b$

The information of an event e is defined as $I(e) = -log_2(P(e))$

Information in the context of documents

Definition

We can now compute the information of a token as:

$$I(t) = -\log\left(\frac{\#\text{doc including token } t}{\#\text{docs}}\right)$$

Vector representation of a document

A document can be represented by a vector of the fraction information associated to each of its token:

$$D_t = \frac{\text{\# t in D}}{\text{\# tokens in D}} \times I(t)$$

Vector representation of a document

A document can be represented by a vector of the fraction information associated to each of its token:

$$D_t = \frac{\text{\# t in D}}{\text{\# tokens in D}} \times I(t)$$

What does $||\vec{D}||_1$ represent?

Vector representation of a document

A document can be represented by a vector of the fraction information associated to each of its token:

$$D_t = \frac{\#\text{ t in D}}{\#\text{ tokens in D}} \times I(t)$$

What does $||\vec{D}||_1$ represent?

$||\vec{D}||_1$ carries the total information carried by a document:

- low if the document contains only common tokens
- average if the document contains few exceptional tokens
- high if the document contains only exceptional items

The tf-idf matrix

Definition

The matrix M which rows – corresponding to each document – are:

$$D_t = \frac{\#\text{ t in D}}{\#\text{ tokens in D}} \times I(t)$$

is called the **tf-idf** (term frequency-inverse document frequency) representation.

The tf-idf matrix

Definition

The matrix M which rows – corresponding to each document – are:

$$D_t = \frac{\text{\# t in D}}{\text{\# tokens in D}} \times I(t)$$

is called the **tf-idf** (term frequency-inverse document frequency) representation.

Question

What is the unit of elements of the tf-idf matrix?

Querying a set of vector

Represent the query the same way:

$$Q_t = \frac{\text{\# t in Q}}{\text{\# tokens in Q}} \times I(t)$$

How to retrieve documents related to the query?

Querying a set of vector

Represent the query the same way:

$$Q_t = \frac{\text{\# t in Q}}{\text{\# tokens in Q}} \times I(t)$$

How to retrieve documents related to the query? Naïve approach: dot product.

Indeed, it makes sense: For each document, compute:

$$\vec{D} \cdot \vec{Q} = \sum_t D_t.Q_t$$

The higher the dot product, the more informative tokens $\vec{Q}$ and $\vec{D}$ share... and the more relevant should be the D with respect to the query Q.

Querying a set of vector

Represent the query the same way:

$$Q_t = \frac{\text{\# t in Q}}{\text{\# tokens in Q}} \times I(t)$$

How to retrieve documents related to the query? Naïve approach: dot product.

Indeed, it makes sense: For each document, compute:

$$\vec{D} \cdot \vec{Q} = \sum_t D_t . Q_t$$

The higher the dot product, the more informative tokens $\vec{Q}$ and $\vec{D}$ share... and the more relevant should be the D with respect to the query Q.

Exercise

Code this scalar product in an efficient way!

Querying a set of vector

Represent the query the same way:

$$Q_t = \frac{\text{\# t in Q}}{\text{\# tokens in Q}} \times I(t)$$

How to retrieve documents related to the query? Naïve approach: dot product.

Indeed, it makes sense: For each document, compute:

$$\vec{D} \cdot \vec{Q} = \sum_t D_t.Q_t$$

The higher the dot product, the more informative tokens $\vec{Q}$ and $\vec{D}$ share... and the more relevant should be the D with respect to the query Q.

Exercise

Code this scalar product in an efficient way!

For querying purposes, one can select documents such that $\vec{D} \cdot \vec{Q} > \tau$, but it can directly be used for ranking documents.

Correcting for cheaters

Problem

Imagine a way of cheating with this approach.

Correcting for cheaters

Problem

Imagine a way of cheating with this approach.

Content farms

$$\begin{array}{rcl}\vec{D}\cdot\vec{Q} & = & \sum_t D_t.Q_t \\ & = & \sum_t \frac{\#\text{ t in D}}{\#\text{ tokens in D}} \times I(t).\frac{\#\text{ t in Q}}{\#\text{ tokens in Q}} \times I(t) \\ & \propto & \frac{1}{\#\text{ tokens in D}} \sum_t \#\text{t in D} \times \#\text{t in Q} \times I(t)^2\end{array}$$

Correcting for cheaters

Problem

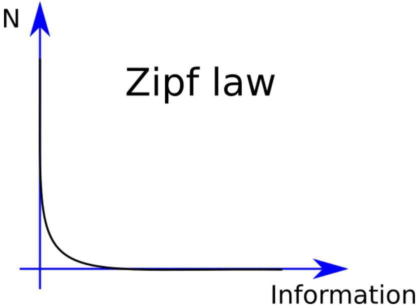
Imagine a way of cheating with this approach.

Content farms

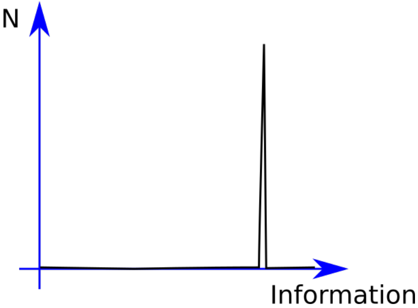
$$\begin{aligned}
\vec{D} \cdot \vec{Q} &= \sum_t D_t . Q_t \\
&= \sum_t \frac{\#\text{ t in D}}{\#\text{ tokens in D}} \times I(t) . \frac{\#\text{ t in Q}}{\#\text{ tokens in Q}} \times I(t) \\
&\propto \frac{1}{\#\text{ tokens in D}} \sum_t \#\text{t in D} \times \#\text{t in Q} \times I(t)^2
\end{aligned}$$

Documents containing many informative words will be selected and ranked first.

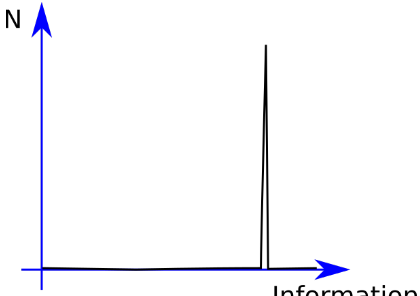
Content farms: pull informative words together



Content farms: pull informative words together



Content farms: pull informative words together



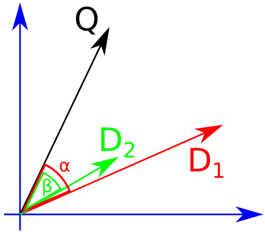
...Can "answer" many more queries

The cosine similarity

How could you correct for content farms cheats?

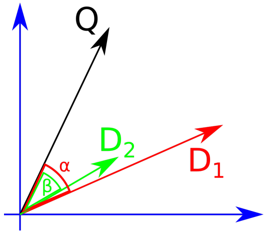
The cosine similarity

How could you correct for content farms cheats?



The cosine similarity

How could you correct for content farms cheats?



Correct by normalizing the similarity:

Consine similarity

$$\text{cosim}(\vec{D}, \vec{Q}) = \frac{\vec{D} \cdot \vec{Q}}{||\vec{D}||_2 . ||\vec{Q}||_2}$$

A flexible querying system?

With the vector space model, information of the tokens are now automatically taken into account.
Does it solve the synonymous problem?

Example

Query: `result elections United States`
Doc title: "White House election: live results!"

A flexible querying system?

With the vector space model, information of the tokens are now automatically taken into account.
Does it solve the synonymous problem?

Example

Query: `result elections United States`
Doc title: "White House election: live results!"

As already pointed out, we could use a semantic approach (ontologies), but need a fixed and manually curated work.

A flexible querying system?

With the vector space model, information of the tokens are now automatically taken into account.
Does it solve the synonymous problem?

Example

Query: `result elections United States`
Doc title: "White House election: live results!"

As already pointed out, we could use a semantic approach (ontologies), but need a fixed and manually curated work.
Can we work directly from the data?

Embeddings

From TF-IDF to Embeddings

TF-IDF allows to have a vector representation of documents in the "space" of tokens.

From TF-IDF to Embeddings

TF-IDF allows to have a vector representation of documents in the "space" of tokens.

Embeddings

Embeddings aim at reducing space of tokens to less dimension in an useful way: a token will live in a small dimensional space ($D_E = 300$) such that semantically similar token lie close to each other in space.

Embeddings: the many derivatives

Many models have been developed for representing various type of data. Here is a small list of freely available models:

Model	Data represented
`word2vec`	Tokens
`GloVe`	Tokens
`fastText`	Tokens
`doc2vec`	Documents
`dna2vec`	Genomic sequences

... to be continued next lecture