Topic 3: The Vector Space Model

COMP3009J: Information Retrieval

Dr. David Lillis (david.lillis@ucd.ie)

UCD School of Computer Science
Beijing Dublin International College

Introduction

- So far, we have seen Information Retrieval (IR) as a process where a user submits a **query** consisting of some **terms** connected with **boolean operators**.
- This is known as the "Boolean Model" of IR, and has some serious disadvantages:
  - The model predicts that each document is either **relevant** or **non-relevant** to the **query**.
  - There is no notion of a "partial" match to the query conditions and so this can lead to too few documents being retrieved.
  - Every document that is considered to be relevant is treated the same, so **no ranking** occurs.

Introduction

- One of the most common IR models in use is the **Vector Space Model**.
- This models both **documents** *and* **queries** in an **N-dimensional space**.
- Basic vector algebra is used in order to calculate the most similar documents to a query.
- The basic vector space model has been adapted for many other purposes.
- The most common are:
  - Machine Learning (e.g. *K*-NN algorithm)
  - Clustering
- See page 77 of Modern Information Retrieval (2nd Ed.).

Basis

- A collection contains a set of documents each of which is composed of a “bag of words” (also known as terms).
  - Each unique **term in the collection** is represented by a dimension in the vector space.
  - A document is then represented by giving a non-zero value to the dimension representing a term it contains.
    - This number is known as a **term weight**.
  - Other dimensions get a value of zero.
    - i.e. the weight of a term is 0 in documents that do not contain it.
- A **similarity score** is calculated based on the proximity of two vectors in the N-dimensional space.

Partial Matching

- Unlike the Boolean Model, the Vector Space Model facilitates **partial matching** by using **non-binary** term weights.
- In models like this, each document must have a **similarity score** calculated for it, which measures how similar it is to the given query.
- This is usually shown as $sim(q, d)$ (i.e. the similarity between a query q *and a document* d).
- These models return a **ranked list** of documents, where the documents with the highest similarity scores are at the top of the list.
- In this way, it is hoped that the most relevant documents are at the top of the result set, so that the user can find relevant information easier.

Vector Space Model

Document and Query Representation

Document Representation

- Consider the following documents:
  1. Information Retrieval is an exciting subject
  2. Mathematics is important in Information Retrieval
- Ignoring stopwords the collection contains **6 distinct terms**: information; retrieval; exciting; subject; mathematics; important.
- The vector space will have **six dimensions** (one for each term in the document collection).
- This means that **EVERY DOCUMENT** is mathematically represented by six dimensions, regardless of how many terms it contains.

Document Representation

- For the introduction to this topic, we first assume a simple weighting scheme:
  - if term i is contained in document j, it will have a weight of 1 (i.e. $w_{i,j} = 1$).
  - if term i is not contained in document j, $w_{i,j} = 0$.
- In practice, the weight will be a real number where:
  - $0 \leq w_{i,j} \leq 1$
- We will see how this is calculated later.

Term	Doc 1	Doc 2
information	1	1
retrieval	1	1
exciting	1	0
subject	1	0
mathematics	0	1
important	0	1

Query Representation

- The query is represented in an identical way.
- Consider the query: *important information*.
- This is represented in the vector model as:
  - (1, 0, 0, 0, 0, 1)
- It is important that the **order of the terms** is the same as for the documents (i.e. the first dimension relates to the term "information", the second relates to "retrieval", etc.).
  - If the order changes, then we are no longer comparing like with like.

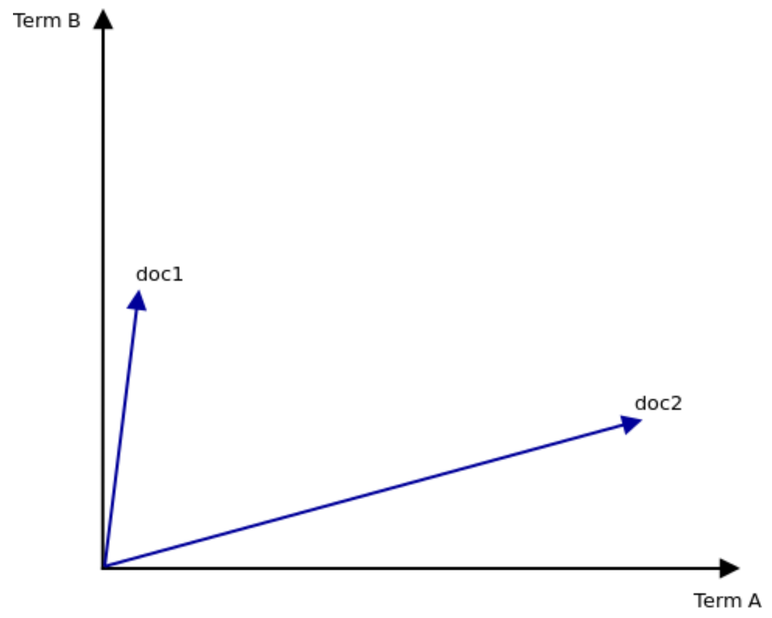
Vector Space Model

Cosine Similarity

Retrieval

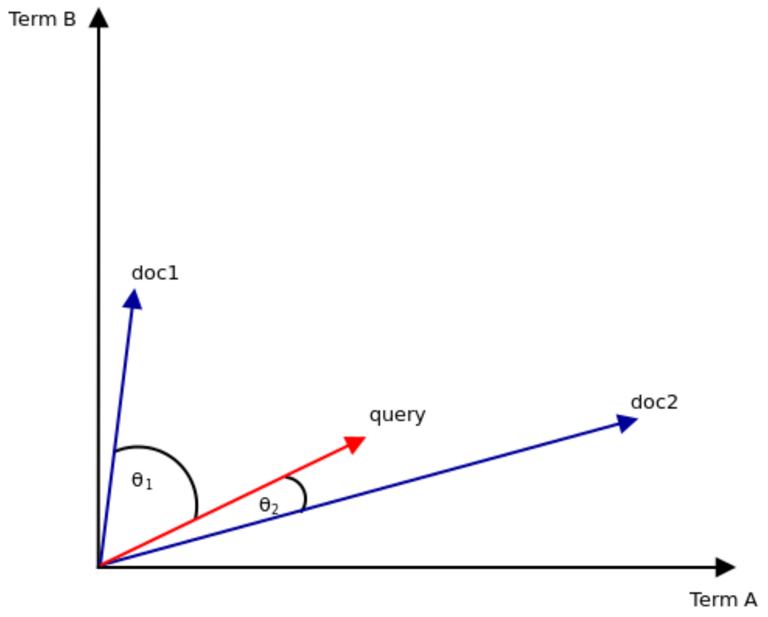
- The theory behind retrieval says that the closer the vectors are in the N-dimensional space, the more similar the items they represent are.
- In retrieval, the query vector is compared to **all of the document vectors** in the index.
- Vector algebra contains an operator known as the **dot product**, defined for two vectors $\vec{a}$ and $\vec{b}$ as:
  - $\vec{a}.\vec{b} = cos.\theta \times |\vec{a}|\times|\vec{b}|$
  - Here, $|\vec{a}|$ and $|\vec{b}|$ are the lengths of vectors $\vec{a}$ and $\vec{b}$ respectively.
  - θ is the angle between the two vectors.

Similarity Example



- Here, we see a VERY simple vector model with only two terms.
- In the document “doc1”, Term B is far more important than Term A.
- The opposite is the case with document “doc2”.
- (This is for illustration only: a realistic example will have many more dimensions).

Similarity Example



- We now introduce a query, **also represented by a vector**.
- The key to the Vector Space Model is the angle between two vectors (θ_1 and θ_2).
- Here, θ_2 is clearly the smaller angle, suggesting that the query is more similar to “doc2”.

Retrieval

- Consider the dot product formula again:
  - $\vec{a}.\vec{b} = cos.\theta \times |\vec{a}| \times |\vec{b}|$
- This can be rewritten as:
  - $cos.\theta = \frac{\vec{a}.\vec{b}}{|\vec{a}| \times |\vec{b}|}$
- This means that if we can calculate the dot product of the two vectors, and the lengths of the two vectors, we can easily find the cosine of the angle between them.
- Why would we want to do this?

Cosine Similarity: Why?

- Principle: the smaller the angle between two vectors, the more similar they are.
- We do not use negative weights, therefore: $0° \leq \theta \leq 90°$
  - The angle is $90°$ if the vectors have nothing in common.
  - The angle is $0°$ if the vectors are identical (or near-identical)
- $cos. 0° = 1$ and $cos. 90° = 0$
  - Therefore, the cosine function gives a value between 0 (for very different documents) and 1 (for identical documents).
  - This is ideal for a measure of **similarity**.

Cosine Similarity Formulae

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \cdot \vec{q}}{|\overrightarrow{d_j}| \times |\vec{q}|}$$

- $\overrightarrow{d_j}$ is a vector representing a document d_j
- $\vec{q}$ is a vector representing the query q
- $\overrightarrow{d_j} \cdot \vec{q}$ is the dot product of $\overrightarrow{d_j}$ and $\vec{q}$
- $|\overrightarrow{d_j}|$ is the length of $\overrightarrow{d_j}$
- $|\vec{q}|$ is the length of $\vec{q}$

Cosine Similarity Formulae

$$sim(d_j, q) = \frac{\overrightarrow{d_j} \cdot \vec{q}}{|\overrightarrow{d_j}| \times |\vec{q}|}$$

$$= \frac{\sum_{i=1}^{T} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{T} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{T} w_{i,q}^2}}$$

- T is the number of terms in the entire document collection
- $w_{i,j}$ is the weight of term i in document j
- $w_{i,q}$ is the weight of term i in the query q

Vector Definitions

- The top line of this formula is the the dot product in *N*-dimensions of $\vec{a}$ and $\vec{b}, \vec{a}.\vec{b}$, which is defined as:
  - $\vec{a}.\vec{b} = \sum_{i=1}^{T}(w_{i,a} \times w_{i,b})$
  - where T is the number of dimensions and $w_{i,a}$ is the weight of term i in $\vec{a}$ (i.e. the ith component of the vector $\vec{a}$).
- Or in pseudocode:

```
Begin with a total of 0
For each dimension in the vectors:
    Multiply the value in that dimension of vector $\vec{a}$ by the value in that dimension of vector $\vec{b}$.
    Add this to the total.
The final total is the dot product.
```

Dot Product: Example

- Example: dot product of (1,1,1,1,0,0) and (1,0,0,0,0,1)
- $(1\times1)+(1\times0)+(1\times0)+(1\times0)+(0\times0)+(0\times1) = 1$

Vector Definitions

- The bottom line of the similarity formula requires us to calculate the length of a vector.
- The length of a vector $\vec{a}$, given by $|\vec{a}|$ is defined as:
- $|\vec{a}| = \sqrt{\sum_{i=1}^{T} w_{i,a}^2}$
- Or in pseudocode:

```
Begin with a total of 0
For each dimension in vector $\vec{a}$
    Take the value in that dimension of vector $\vec{a}$ and square it.
    Add this to the total.
The length of vector $\vec{a}$ is the square root of the total
```

Vector Length: Example

- Example: length of (1, 1, 1, 1, 0, 0)
- $\sqrt{1^2 + 1^2 + 1^2 + 1^2 + 0^2 + 0^2}$
- $= \sqrt{4} = 2$

Cosine Similarity: Example

- Returning to the documents and the query we saw earlier:
  - D1: 1, 1, 1, 1, 0, 0
  - D2: 1, 1, 0, 0, 1, 1
  - Q: 1, 0, 0, 0, 0, 1
- The similarities between these documents and the query are calculated as:
  - $sim(d_1, q) = \frac{\overrightarrow{d_1} \cdot \vec{q}}{|\overrightarrow{d_1}| \times |\vec{q}|} = \frac{1}{2 \times \sqrt{2}} = 0.3535$
  - $sim(d_2, q) = \frac{\overrightarrow{d_2} \cdot \vec{q}}{|\overrightarrow{d_2}| \times |\vec{q}|} = \frac{2}{2 \times \sqrt{2}} = 0.7071$
- Hence document D2 is most similar to the query.

Cosine Similarity: Some Questions

- Some questions that will help us when planning to implement this:
  - What effect on the similarity score does a term have if it does not appear in the query?
  - What effect on the similarity score does a term have if it does not appear in the document?
  - What effect on the length of a document vector does a term have if that terms is not contained in the document?
- Earlier, we said:
  - … **EVERY DOCUMENT** is mathematically represented by six dimensions, regardless of how many terms it contains.
  - "**mathematically**" represented ≠ how we represent using a computer program.

Vector Space Model

Term Weighting

Term Weighting

- In order to illustrate the model, we have used a simple binary weighting scheme.
- This gives the representation of the vectors in terms of 1s and 0s only.
- This is not commonly used in practice.
- Frequently used metric is **TF-IDF**:
  - **TF:** Term Frequency
  - **IDF:** Inverse Document Frequency

Term Weights: Why?

- Consider a document collection of 100,000 documents.
- A word that appears in every document is not very useful as an index term, as it tells us very little about which documents a user might be interested in.
- A word that appears in 5 documents considerably narrows the space of documents that might be of interest to the user.
- This is the basic rationale behind stopwords, but it has wider consequences also.

Term Weights

- We can also take the **number of times** a term occurs in a document into account (i.e. the *frequency* of the term), for example:
  - The term "Trump" will appear in a biography of Donald Trump, for obvious reasons.
  - The term "Trump" will also appear in a biography of Hillary Clinton, as she once lost an election to Donald Trump. It will not be as common, however.
- The first document is more relevant to a search for "Trump" but without using weights, both would be treated equally.
- The greater number of occurrences of the term in the first document should result in a greater weight.

Term Weights

- When talking about term weights, we use the following notation:
- A weight $w_{i,j} > 0$ is associated with **every index term** k_i of **document** d_j.
- For an index term that does not appear in the document, $w_{i,j} = 0$
- This weight quantifies the importance of the index term for describing the document's semantic contents.
- Any time we examine a new IR model, we must firstly consider how its weighting scheme works.

Term Weights

- Index terms are usually assumed to be **independent**, though this is a simplification.
- Consider a document about computer networks. Here, the terms “computer” and “network” are clearly related, as the appearance of one attracts the appearance of the other.
- You may argue that their weights should be altered to take this dependency into account.

TF-IDF Weighting

- TF-IDF consists of two parts:
  - The **Term Frequency (TF)** assumes that a term that occurs many times in a document is more important in describing that document than one that occurs rarely. **Every term** has a separate term frequency for **every document** in the IR system.
  - The **Inverse Document Frequency (IDF)** argues that terms that only occur in very few documents are more important than those that are found in many documents. This is based on the same idea as stopwords. **Each term** has only **one** inverse document frequency within an entire document collection.

Term Frequency (TF)

- Term Frequency is calculated by using the formula below.
- $tf_{i,j}$ is the term frequency of term i in document j and is calculated as follows:
  - $tf_{i,j} = \frac{freq_{i,j}}{maxfreq_j}$
    - where $freq_{i,j}$ is the number of times term i occurs in document j and $maxfreq_j$ is the maximum number of times *any* term occurs in document j.
- The reason we divide by the maximum frequency is to ensure that terms in **long documents** do not get an unfair advantage.

Inverse Document Frequency (IDF)

- A term's Inverse Document Frequency is designed to give a higher weight to terms that are rare.
- **Note** that each term only has one IDF score within an entire document collection.
- It is calculated using the following formula:
  - $idf_i = log_2\left(\frac{N}{n_i}\right)$
- N is the total number of documents in the collection
- n_i is the number of documents in the collection that contain term i.

TF-IDF

- Finally, the TF-IDF weight for a term i in a document j is found by multiplying its TF for that document by its IDF score, i.e.
- $w_{i,j} = tf_{i,j} \times idf_i$
- Note again that each term has a **different TF-IDF score** for **each document** in the document collection.

Example

- D1: “new york times”
- D2: “new york post”
- D3: “los angeles times”
- Q: “new new times”

Vector Space Model

Conclusions and Summary

TF-IDF Advantages

- The advantages of the TF-IDF weighting scheme over others are:
  - Using term frequency gives more influence to a document that contains many occurrences of a particular term.
  - Using inverse document frequency lessens the impact of terms that appear very often in the collection. These terms are regarded as being less informative.

Vector Space Model: Advantages and Disadvantages

- There are three main advantages of the vector space model:
  - The term weighting schemes can improve retrieval performance.
  - Partial matching strategy allows for retrieval of documents that match part of the query.
  - The cosine similarity can be used to return a ranked list of documents .
- The main disadvantage of the model is the assumption that terms are independent, which can sometimes harm performance.

Vector Space Model: Implementation

- The main steps in implementing the vector space model are:
  1. Read in the documents.
  2. Divide each document into its constituent terms.
  3. Remove stopwords.
  4. Stem terms.
  5. Add documents to an index.
  6. Calculate TF-IDF.
     - Each term has one IDF score.
     - Each term has a separate TF score in each document.
  7. Receive a query.
  8. Retrieve query results.

Summary

- The **Vector Space Model** of Information Retrieval is based on concepts from vector algebra.
- Documents and queries are represented as vectors.
- Cosine similarity is used to decide which documents most closely match a query.
- Terms can be weighted in multiple ways.
- A common weighting scheme is TF-IDF (Term Frequency-Inverse Document Frequency).