Query type

We shall only consider *simple queries* of the form:

- Find documents containing *word1*, *word2*, . . . , *wordn*

More specific tasks are:

- Find documents containing all the words *word1*, *word2* . . . *wordn*;
- or find documents containing as many of the words *word1*, *word2* . . . *wordn* as possible.

Going beyond these forms, queries can also be much more complex: they can be combined using boolean operations, look for whole phrases, substrings of words, look for matches of regular expressions, etc.

A retrieval model

If we look for all documents containing all words of the query — or all documents that contain some of the words of the query — then this may well result in a large number of documents, of widely varying relevance.

In this situation, it can help if IR systems can rank documents according to likely relevance.

There are many such ranking methods.

We focus on one, which uses the *vector space model*.

This model is the basis of many IR applications; it originated in the work of Gerard Salter and others in the 1970's, and is still actively developed.

In this course, we shall only use it in one particularly simple way.

The vector space model

Core ideas:

- Treat documents as points in a high-dimensional vector space, based on words in the document collection.
- The query is treated in the same way.
- The documents are ranked according to document-query similarity.

N.B. You do not need a detailed understanding of vector spaces to follow the working of the model.

The vector associated to a document

Suppose $\text{Term}_1, \text{Term}_2, \ldots, \text{Term}_n$ are all the different words occurring in the entire collection of documents $\text{Doc}_1, \text{Doc}_2, \ldots, \text{Doc}_K$.

Each document, Doc_i, is assigned an n-valued vector:

$$(m_{i1}, m_{i2}, \ldots, m_{in})$$

where m_{ij} is the number of times word Term_j occurs in document Doc_i.

Similarly, the query is assigned an n-valued vector by considering it as a document itself.

Example

Consider the document

Sun, sun, sun, here it comes

and suppose the only words in the document collection are: *comes*, *here*, *it*, *sun*.

The vector for the document is $(1, 1, 1, 3)$

comes	here	it	sun
1	1	1	3

Similarly, the vector for the query *sun comes* is $(1, 0, 0, 1)$

Document matrix

The frequency information for words in the document collection is normally precompiled in a *document matrix*.

This has:

- Columns represent the words appearing the document collection
- Rows represent each document in the collection.
- each entry in the matrix represents the frequency of the word in the document.

Document matrix — example

	Term_1	Term_2	Term_3	...	Term_n
Doc_1	14	6	1	...	0
Doc_2	0	1	3	...	1
Doc_3	0	1	0	...	2
...	...	...	...	...	...
Doc_K	4	7	0	...	5

N.B. Each row gives the vector for the associated document.

Vector similarity

We want to rank documents according to relevance to the query.

We implement this by defining a measure of *similarity* between vectors.

The idea is that the most relevant documents are those whose vectors are most similar to the query vector.

Many different similarity measures are used. A simple one that is conceptually appealing and enjoys some good properties is the *cosine* of the angle between two vectors.

Cosines (from school trigonometry)

Recall that the *cosine* of an angle $\boldsymbol{\theta}$ is:

$$\frac{\text{adjacent}}{\text{hypotenuse}}$$

in a right-angled triangle with angle $\boldsymbol{\theta}$.

Crucial properties:

$$\cos(0) = 1 \qquad \cos(90^\circ) = 0 \qquad \cos(180^\circ) = -1$$

More generally, two n-dimensional vectors will have cosine: 1 if they are identical, 0 if they are orthogonal, and -1 if they point in opposite directions.

The value $\cos(x)$ *always* lies in the range from -1 to 1.

Vector cosines

Suppose $\vec{x}$ and $\vec{y}$ are n-value vectors:

$$\vec{x} = (x_1, \ldots, x_n) \qquad \vec{y} = (y_1, \ldots, y_n)$$

Their *cosine* (that is, the cosine of the angle between them) is calculated by:

$$\cos(\vec{x}, \vec{y}) = \frac{\vec{x} \cdot \vec{y}}{|\vec{x}||\vec{y}|} = \frac{\sum_{i=1}^{n} x_i y_i}{\sqrt{\sum_{i=1}^{n} x_i^2}\sqrt{\sum_{i=1}^{n} y_i^2}}$$

Here $\vec{x} \cdot \vec{y}$ is the *scalar product* of vectors $\vec{x}$ and $\vec{y}$, while $|\vec{x}|$ is the length (or *norm*) of the vector $\vec{x}$.

Vector cosines — example

Continuing the example from slide 11.18, suppose:

$$\vec{x} = (1, 1, 1, 3) \qquad \vec{y} = (1, 0, 0, 1)$$

Then:

$$\begin{aligned} \vec{x} \cdot \vec{y} &= 1 + 0 + 0 + 3 = 4 \\ |\vec{x}| &= \sqrt{1 + 1 + 1 + 9} = \sqrt{12} \\ |\vec{y}| &= \sqrt{1 + 0 + 0 + 1} = \sqrt{2} \end{aligned}$$

So

$$\cos(\vec{x}, \vec{y}) = \frac{4}{\sqrt{12} \times \sqrt{2}} = \frac{2}{\sqrt{6}} = 0.82$$

to two significant figures.

Ranking documents

Suppose $\vec{y}$ is the query vector, and $\vec{x_1}, \ldots, \vec{x_K}$ are the K document vectors.

We calculate the K values:

$$\cos(\vec{x_1}, \vec{y}), \quad \ldots \quad, \cos(\vec{x_K}, \vec{y})$$

Sorting these, the documents with the highest cosine values when compared to the query $\vec{y}$ are the best match, and those with the lowest cosine values are counted as least suitable.

N.B. On this slide $\vec{x_1}, \ldots, \vec{x_K}$ are K (potentially) different vectors, each with n values.

Discussion of cosine measure

The cosine similarity measure, as discussed here, is very crude.

- It only takes word frequency into account, not position or ordering
- It takes all words in the document collection into account (whether very common "stop" words which are useless for IR, or very uncommon words unrelated to the search)
- All words in the document collection are weighted equally
- It ignores document size (just the angles between vectors not their magnitude are considered)

Nevertheless, the cosine method can be refined in various ways to avoid these problems. (This is beyond the scope of this course.)

Other issues

- Precision and recall, as defined, only evaluate the set of documents returned, they do not take *ranking* into account. Other more complex evaluation measures can be introduced to deal with ranking (e.g., *precision at a cutoff*).
- We have not considered the efficient implementation of the search for documents matching a query. This is often addressed using a purpose-built index such as an *inverted index* which indexes all documents using the words in the document collection as keys.
- Often useful ranking methods make use of information extraneous to the document itself. E.g., Google's *pagerank* method evaluates documents according to their degree of *connectivity* with the rest of the web (e.g., number of links to page from other pages).

These are important issues, but are beyond the scope of this course.

Part III — Unstructured Data

Data Retrieval:

Statistical Analysis of Data:

Analysis of data

There are many reasons to *analyse* data.

Two common goals of analysis:

- Discover implicit structure in the data.
  E.g., find patterns in empirical data (such as experimental data).
- Confirm or refute a hypothesis about the data.
  E.g., confirm or refute an experimental hypothesis.

Statistics provides a powerful and ubiquitous toolkit for performing such analyses.

Data scales

The type of analysis performed (obviously) depends on:

- The reason for wishing to carry out the analysis.
- The type of data to hand.
  For example, the data may be *quantitative* (i.e., numerical), or it may be *qualitative* (i.e., descriptive).

One important aspect of the kind of data is the form of *data scale* it belongs to:

- *Categorical* (also called *nominal*) and *Ordinal* scales (for qualitative data).
- *Interval and ratio* scales (for quantitative data).

This affects the ways in which we can manipulate data.

Categorical scales

Data belongs to a *categorical scale* if each *datum* (i.e., data item) is classified as belonging to one of a fixed number categories.

Example: The British Government (presumably) classifies Visa applications according to the nationality of the applicant. This classification is a categorical scale: the categories are the different possible nationalities.

Example: Insurance companies classify some insurance applications (e.g., home, possessions, car) according to the postcode of the applicant (since different postcodes have different risk assessments).

Categorical scales are sometimes called *nominal scales*, especially in cases in which the value of a datum is a name.

Ordinal scales

Data belongs to an *ordinal scale* if it has an associated ordering but arithmetic transformations on the data are not meaningful.

Example: The *Beaufort wind force scale* classifies wind speeds on a scale from **0** (calm) to **12** (hurricane). This has an obvious associated ordering, but it does not make sense to perform arithmetic operations on this scale. E.g., it does not make much sense to say that scale **6** (strong breeze) is the average of calm and hurricane force.

Example: In many institutions, exam marks are recorded as grades (e.g., A,B,. . . , G) rather than as marks. Again the ordering is clear, but one does not perform arithmetic operations on the scale.

Interval scales

An *interval scale* is a numerical scale (usually with real number values) in which we are interested in *relative value* rather than *absolute value*.

Example: Points in time are given relative to an arbitrarily chosen zero point. We can make sense of comparisons such as: moment $\boldsymbol{x}$ is 2009 years later than moment $\boldsymbol{y}$. But it does not make sense to say: moment $\boldsymbol{x}$ is twice as large as moment $\boldsymbol{z}$.

Mathematically, interval scales support the operations of subtraction (returning a real number for this) and weighted average.

Interval scales do not support the operations of addition and multiplication.

Ratio scales

A *ratio scale* is a numerical scale (again usually with real number values) in which there is a notion of *absolute value*.

Example: Most physical quantities such as mass, energy and length are measured on ratio scales. So is temperature if measured in kelvins (i.e. relative to absolute zero).

Like interval scales, ratio scales support the operations of subtraction and weighted average. They also support the operations of addition and of multiplication by a real number.

Question for physics students: Is time a ratio scale if one uses the Big Bang as its zero point?