INFORMATION EXTRACTION (IE)

- It is the process of retrieving structured information from unstructured text.
- It is the type of information retrieval whose goal is to automatically extract structured information from unstructured and / or semi-structured machine-readable documents.
- In most of the cases, this activity concerns processing human language texts by means of natural language processing.
- Identify specific pieces of information (data) in an unstructured or semi-structured textual document.
- It is applied to different types of texts like:
  - Newspaper articles
  - Web pages
  - Scientific articles
  - Medical notes

APPLICATIONS OF IE

- Question answering
- Job postings (example: web pages: Flip dog)
- Job resumes (example: Burning Glass)
- Seminar announcements
- Company information from the web
- University information from the web

Example

- **Sample Job Posting**

  Subject: US-TN-SOFTWARE PROGRAMMER

  Date: 17th Nov 1996 17:37:29 GMT

  Organization: Reference.com Posting Service

  MessageID: 56nigp$mrs@bilbo.reference.com

  SOFTWARE PROGRAMMER

  Position is available for Software Programmer experienced in generating software for PC-based voice mail system. Should be experienced in C programming. Must be familiar with communicating with and controlling voice cards. Prefer 5 years or more experience with PC based voice mail, but will consider as little as 2 years. Need to find a senior level person who can come on board and pick up code with very little training. Present operating system is DOS. May go to OS/2 or UNIX in future.

  Please reply to

  Kim Anderson

AdNET

(901)458-2888 Fax

- **Extracted Job Template**

Computer_science_job	
Id	: 56nigp$mrs@bilbo.reference.com
Title	: SOFTWARE PROGRAMMER
Salary	: --
Company	: Reference.com Posting Service
State	: TN
City	: --
Country	: US
Language	: C
Platform	: PC /DOS/OS-2/UNIX
Area	: Voice mail
Required_years_experience	: 2
Desired_years_experience	: 5
Required_degree	: --
Desired_degree	: --
Post_date	: 17th Nov 1996
Due_date	: --

WEB EXTRACTION

- Many web pages are generated automatically form an underlying database.
- Therefore, the HTML structure of pages is fairly specific and regular (semi-structured).
- An IE system for such generated pages allows the web site to be viewed as structured database.
- An extractor for a semi-structured web site is called a wrapper.
- Wrapper is a program that extracts contents of a particular information source and translates it into a relational form.
- If extracting from more natural, unstructured human-written text, NLP may help.
  - POS (Part of Speech) Tagging
    - Mark each word as a noun, verb, preposition, etc.
  - Syntactic Parsing
    - Identify phrases (NP, VP)
  - Semantic Word Categories (from Word Net)
    - Example: KILL kill, murder, assassinate, strangle, suffocate.

INFORMATION INTEGRATION

- Answering certain questions using the web requires integrating information from multiple web sites.
- Information integration concerns methods for automating this integration.
- Example:
  Question What is the closest theater to my home where I can see both Monsters and Harry Potter?
  Process
  - From austin360.com, extract theatres and their address where Harry Potter and Monster are playing.
  - Intersect the two to find the theatres playing both.
  - Query mapquest.com for driving directions from your home address to the address of each theatre.
  - Extract distance and driving instruction for each.
  - Sort results by driving distance.
  - Present driving instruction for closest theatre.

XML & INFORMATION EXTRACTION

- XML enables documents designers to design rich tag sets where tags for section headings contain information about each section.
- Easy to extract facts from semi-formatted online documents.
- XML makes IE easy.
- IE provides a way of automatically transforming semi-structured or unstructured data into an XML compatible format.
- For example: SIFT (Specification Information From Text).

SEMANTIC WEB

- It is a web of linked data.
- To describe things in a way that computer can understand.
- For example:
  - The Beatles was a popular brand from Liverpool.
  - John Lennon was a member of the Beatles.
  - “Hey dude” was recorded by the Beatles.
- Sentences like the ones can be understood by people, but how can they be understood by computers.
- Statements are build-up with syntax rules.
- The syntax of a language defines the rules for building the language statement.
- But how can syntax become semantic.
- So, semantic web is describing things in a way that computers can understand.
- Semantic web is not about links between web pages.
- Semantic web describes the relationship between things like, A is a part of B, Y is a number of Z, etc or the properties of the things like size, weight, age, price, etc.
- RDF (Resource Description Framework) is a language for describing information and resources on the web.

- Putting information into RDF makes it possible for computer to search, discover, pick, analyze and process information from the web.
- For example: it creates a class “dog” which contains all of the dogs in the world.

  dog rdf : type rdf’s : class
- Then we can say that “puppy is a type of dog” as,
  puppy rdf : type dog
- *See application at w3school.*
  IBA – I Buy Application
  ISA – I Sell Application

SIFT

(Relevant information that can easily be translated into the correct format to test the system)

- For example: “The maximum value you can specify with the ABC argument is 65535”.

  “The maximum value of ABC is 65535”.

  (maximum_value ABC 65535)

PROBALISTIC INFORMATION RETRIEVAL

- Users start with information needs, which they translate into query representations.
- Similarly, there are documents which are converted into document representations.
- Based on these two representations, a system tries to determine how well documents satisfy information needs.
- In the Boolean or vector space model, matching is done with index terms.
- Given the query and document representations, a system has an uncertain guess of whether a document has content relevant to the information need.
- Probability theory provides a principle foundation of such reasoning under uncertainty.

THE 1/0 LOSS CASE

- The user issues a query and an ordered list of documents is returned.
- For a query 'q' and a document 'd' in the collection, let $R_{d,\,q}$ be an indicator random variable that says whether 'd' is relevant with respect to a given query 'q'.
- That is, it takes on a value of 1 when the document is relevant and 0 otherwise.
- Using a probabilistic model, the order in which to present documents to the user is to rank documents by their estimated probability of relevance with respect to information need, $P(R=1|d,q)$, which is the basis of Probability Ranking Principle (PRP).
- Ranking of the documents in the collection is in order of decreasing probability of relevance to the user who submitted the request.
- You lose a point for either returning a non-relevant document or failing to return a relevant document.
- Such a binary situation where you are evaluated on your accuracy is called 1/0 loss.
- PRP rank all the documents in the decreasing order of $P(R=1|d,q)$
- d is relevant iff,

$$P(R=1|d, q) > P(R=0|d, q).$$

BINARY INDEPENDENCE MODEL (BIM)

- Binary is equivalent to Boolean.
- Documents and queries are both represented as binary term incidence vectors.
- i.e. a document d is represented by the vector $x(x_1, x_2, \ldots\ldots\ldots\ldots, x_m)$, where $x_t = 1$ if term t is present in document d and $x_t = 0$ if t is not present in d.
- Similarly, query q is represented by query vector q.
- Independence means that terms are modeled as occurring in documents independently.
- i.e. the model recognizes no association between terms.

$$P(R=1|x,q) = \frac{P(x|R=1,q)\,P(R=1|q)}{P(x|q)}$$

$$P(R=0|x,q) = \frac{P(x|R=0,q)\,P(R=0|q)}{P(x|q)}$$

- But some assumptions like terms that are independent of BIM can be removed.
- Example: term pairs "Hong" and "Kong"are strongly dependent.
- Others are Stock, Exchange, New, York, etc.

INFORMATION RETRIEVAL

- Text is the primary way that human knowledge is stored after speech.
- Techniques for storing and searching for textual documents are nearly as old as written language itself.
- In past, information retrieval means going to town's library and asking the librarian for help.
- The librarian usually knew all the books in the possession and could give one a define answer.
- As the number of books grew, it became impossible. Then tools for information retrieval had to be devised.
- One of the most important tools is indexing.
- Index is a terms with pointer to places where information about them can be found.
- The terms can be subject matter, author names, etc.
- Oliver Wendell Holmes wrote in 1872, "It is the province of knowledge to speak and it is the privilege of wisdom to listen".
- In future, "It is the province of knowledge to write and it is the privilege of wisdom to query."
- The field of computer science that deals with the automated storage and retrieval of a document is called information retrieval.
- Requires:
    - Algorithm – For manipulating natural language.
    - Data Structures – To efficiently store and process data.

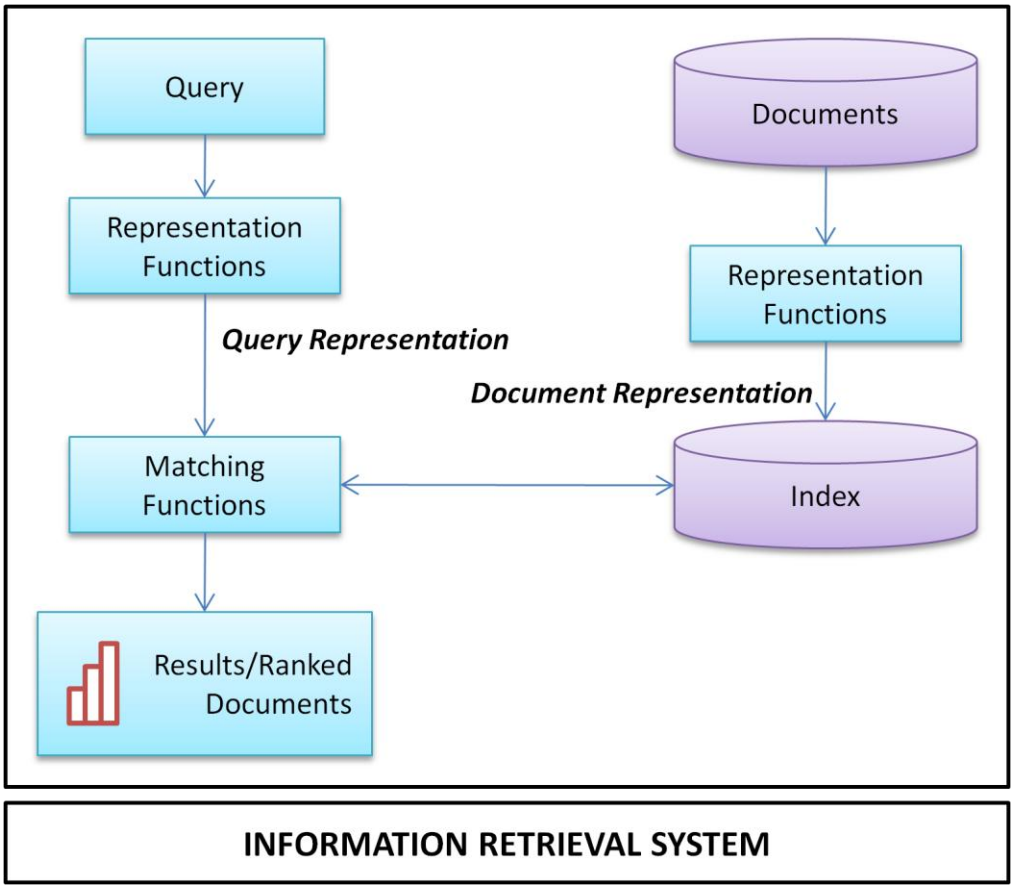
WHAT MAKES IR A HARD PROBLEM?

1. Under good circumstances
    - Text is unstructured.
    - Requires understanding of semantics. For example: restaurant → café, PRC → China, fast automobiles → fast cars.
    - Human language presents distinct problems like ambiguity. For example: bat (mammal or baseball), apple (company or fruit), bit (unit of data or act of eating), etc.

2. Under hard circumstances
   - Web pages change rapidly.
   - Many pages lie about their content.
   - New pages are not linked to.

3. Multimedia information
   - Hard to store (size), represent and compare.

IR SYSTEM



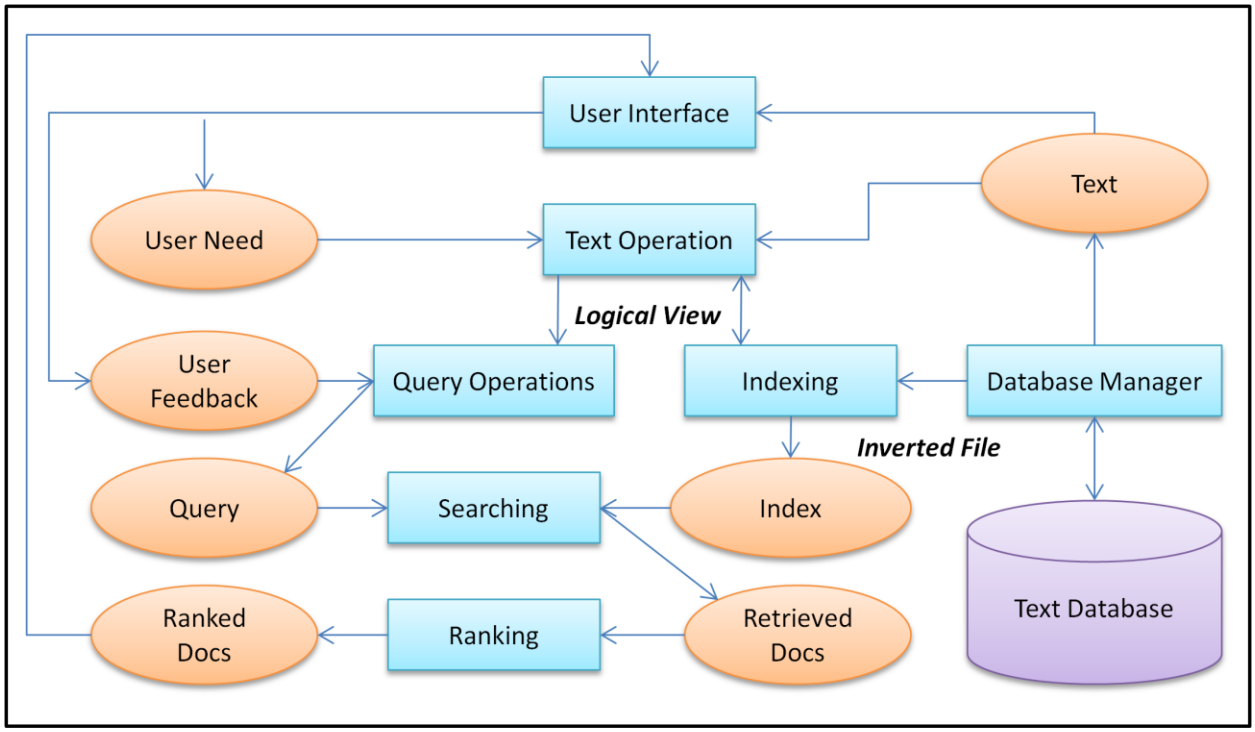
- Searching for pages on the World Wide Web is the most recent killer application.
- IR concerns firstly with retrieving relevant documents as a query.

- Relevance is a subjective judgment and may include:
  1. Being on the proper subject.
  2. Being timely (recent information).
  3. Being authoritative (from a trusted source).
  4. Satisfy the goals of the user.

TYPICAL IR

1. Given
   - A corpus of textual natural language documents.
   - A user query in the form of a textual string.
2. Find
   - A ranked set of documents that is relevant to the query.

IR SYSTEM ARCHITECTURE

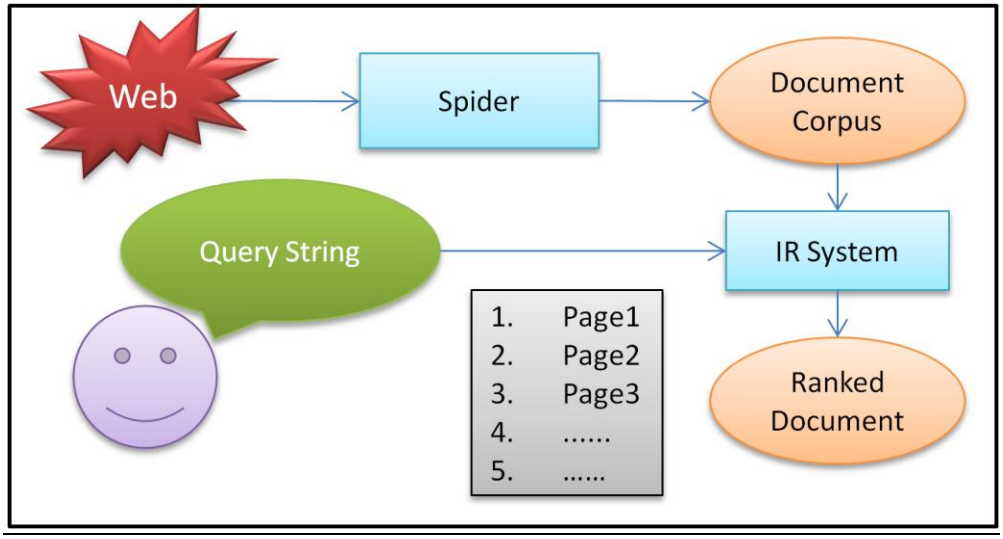


INFORMATION RETRIEVAL ARCHITECTURE

IR SYSTEM COMPONENTS

1. Text Operations
   - Forms index words (tokens) by stop-word removal and stemming.
2. Indexing
   - Constructs an inverted index of word to document pointers.
3. Searching
   - Retrieves documents that contain a given query token from the inverted index.
4. Ranking
   - Scores all retrieved documents according to relevance matrix.
5. User Interface
   - Manage interaction with the user.
   - Query input and document output.
   - Relevance feedback.
   - Visualization of results.
6. Query Operations
   - Transform the query to improve retrieval.
   - Query expansion using thesaurus.

WEB SEARCH AND IR (Application of IR to HTML documents on the World Wide Web)



WEB CHALLENGES OF IR

1. Distributed Data
   - Documents spread over millions of different web servers.
2. Volatile Data
   - Many documents change or disappear rapidly. For example: dead link.
3. Large Volume
   - Billions of separate documents.
4. Unstructured and Redundant Data
   - No Uniform Structure.
   - Up to 30% (near) are duplicate documents
5. Quality of Data
   - No editorial control.
   - False information.
   - Poor quality writing.
6. Heterogeneous Data
   - Multiple media types (image, video)
   - Languages.

AREAS OF AI FOR IR

1. Natural Language Processing
   - Focused on syntactic, semantic and pragmatic analysis of natural language text.
   - Retrieval based should be focused on semantic.
   - Methods for determining the sense of ambiguous word based on context.
   - Question answering.
2. Machine Learning
   - Focused on the development of computational system that improves their performance with experience.
   - Automated classification of examples based on learning concepts from labeled training.

- For example: supervised learning.
- Automated methods for clustering unlabeled examples into meaningful groups (unsupervised).
- Text categorization (For example: spam filtering).
- Text clustering (clustering of IR query results).
- Text mining.

3. Knowledge Representation
   - Expert system
4. Reasoning Under Uncertainty
   - Bayesian network
5. Cognitive Theory

STEPS IN IR PROCESS (RETRIEVAL PROCESS)

1. Indexing (Creating document representation)
   - Indexing is the manual or automated process of making statements about a document, lesson, and person and so on.
   - For example: author wise, subject wise, text wise, etc.
   - Index can be:
     i. Document oriented: – the indexer accesses the document relevance to subjects and other features of interests to user.
     ii. Request oriented: – the indexer accesses the document relevance to subjects and other features of interests to user.
   - Automated indexing begins with feature extraction such as extracting all words from a text, followed by refinements such as eliminating stop words (a, an, the, of), stemming (walking → walk), counting the most frequent words, mapping the concepts using thesaurus (tube → pipe).

2. Query Formulation (Creating query representation)
   - Retrieval means using the available evidence to predict the degree to which a document is relevant or useful for a given user need as described in a free form query description.

- A query can specify text words or phrase, the system should look for.
- The query description is transformed manually or automatically into a formed query representation, ready to match with document representation.

3. Matching the Query Representation With Entity Representation
   - The match uses the features specified in the query to predict document relevance.
   - Exact match (0 or 1).
   - Synonym expansion (pipe → tube).
   - Hierarchical expansion (pipe → capillary).
   - The system ranks the result.

4. Selection
   - User examines the results and selects the relevant items.

5. Relevance Feedback & Interactive Retrieval
   - The system can assist the user in improving the query by showing a list of features (option) found in many relevant items.

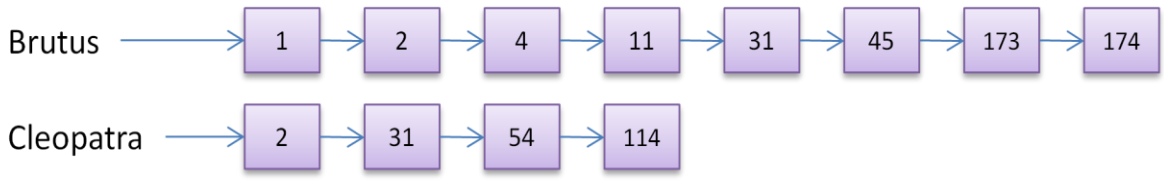
BOOLEAN RETRIEVAL

- Most simple retrieval and relies on the use of Boolean operators.
- The term in a query are linked together with AND, OR and NOT.
- Terms weights are set to 1 if the terms are occurred in the documents.

INTERSECTION ALGORITHM TO COMPUTE BOOLEAN QUERY

```
INTERSECT (p1, p2)
answer ← ( )
while p1! = NIL and p2! = NIL
        do if docID (p1) – docID (p2)
        then ADD (answer, docID (p1))
                p1 ← next (p1)
                p2 ← next (p2)
        else if docID (p1) < docID (p2)
        then p1 ← next (p1)
        else p2 ← next (p2)
return answer
```

EXAMPLE



Intersection (Brutus AND Cleopatra) → 2 → 31

EXAMPLE

d1 = English tutorial and fast track

d2 = Book on semantic analysis

d3 = Learning latent semantic indexing

d4 = Advance in structure and semantic indexing

d5 = Analysis of latent structures

Query → "advance AND structure AND NOT analysis"

Terms	d1	d2	d3	d4	d5
English	1	0	0	0	0
Tutorial	1	0	0	0	0
Fast	1	0	0	0	0
Track	1	0	0	0	0
Book	0	1	0	0	0
Semantic	0	1	1	1	0
Analysis	0	1	0	0	1
Learning	0	0	1	0	0
Structure	0	0	0	1	1
Indexing	0	0	1	1	0
Latent	0	0	1	0	1
Advance	0	0	0	1	0

Fig: Term Document Matrix

Solution:

Query Terms	d1	d2	d3	d4	d5
Advance	0	0	0	1	0
Structure	0	0	0	1	1
	0	0	0	1	0
NOT Analysis	1	0	1	1	0
	0	0	0	1	0

LIMITATION OF BOOLEAN RETRIEVAL

- Very rigid → AND means all & OR means any.
- Difficult to control the number of documents retrieved, i.e. all matched documents will be returned.
- Incapable to rank the output [i.e. all matched documents logically satisfy the query].
- Using many Boolean operators make the query complex to formulate.
- Good for specific user having good knowledge on Boolean operation.
- Not good for majority of the users.

RANK RETRIEVAL

- System decides which documents best satisfy the query.
- Vector space model.

VECTOR SPACE MODEL (VSM)

- A vector space model is a mathematical structure formed by a collection of vectors.
- A point in the space represents a vector.
- The set of all n-tuples (x1, x2,....., xn) of n real numbers is known as n-space where n being a positive integer.
- All the documents are represented by a point in a space of n dimension by n term co-ordinate.
- Queries are treated like documents.
- Documents are ranked by closeness to the query.
- Closeness is determined by a similarity score calculation.

MAJOR PROPERTIES OF VSM

- Ranking of documents according to similarity value.
- Documents can be retrieved even if they don't contain some query keyword.

COSINE SIMILARITY

- Scores the similarity between two document vectors.
- The similarity between the two vectors is defined by the angle between them.
- If the two vectors are exactly similar then the angle between the two vectors are zero and thus cosine equal to 1, representing the perfect match.
- If the two vectors are perfectly dissimilar, then the angle between the vectors is perfect 90° and the cosine equal to 0, represents the perfect dissimilar.

POINTS IN A PLANE

- Points in a two dimension XY plane is defined by a pair of co-ordinates.

DOT PRODUCT

- Dot product is an algebraic operation that takes two co-ordinates vector and returns a single number obtained by multiplying corresponding entries and adding up those products.
- A.B = $x_1x_2 + y_1y_2$
- If A and B are in 3D, A.B = $x_1x_2 + y_1y_2 + z_1z_2$
- In general, if A = (a_1, a_2, ..., a_n) and B = (b_1, b_2, ..., b_n), then, A.B = $\sum_{i=1}^{n} a_i \ . b_i$

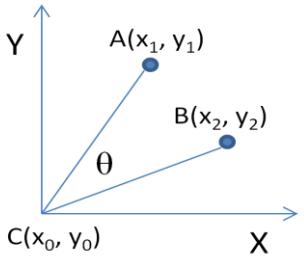
EUCLIDEAN DISTANCE

- Euclidean distance is the distance between two points, one being the origin point.
- i.e. $d_{AC} = \sqrt{(x_1 - x_0)^2 + (y_1 - y_0)^2} = \sqrt{x_1{}^2 + y_1{}^2}$

  $d_{BC} = \sqrt{(x_2 - x_0)^2 + (y_2 - y_0)^2} = \sqrt{x_2{}^2 + y_2{}^2}$

REPRESENTING DOCUMENT VECTOR

- A vector is a quality with direction and magnitude.
- The head and angle of the arrow indicates the direction of the vector.
- Magnitude is defined by Euclidean distance.



DOCUMENT LENGTH NORMALIZATION

- To normalize A.B, the dot product, it is divided by the Euclidean distances of A and B,

  i.e. $\frac{A.B}{|A||B|}$
- The ratio defines the cosine angle between the vectors, with values between 0 and 1.
- This ratio is used as a similarity measure between any two vectors representing documents, queries denoted by sim (A, B)

  i.e. sim (A, B) = cosine θ

  $= \frac{A.B}{|A||B|}$

  $= \frac{x_1 x_2 + y_1 y_2}{\sqrt{x_1{}^2+y_1{}^2}\sqrt{x_2{}^2+y_2{}^2}}$

QUERIES OF VECTORS

- By viewing a query as a "bag of words", it is able to treat as a very short document.

  Score (q, d) = $\frac{\vec{v}(q) \cdot \vec{v}(d)}{|\vec{v}(q)||\vec{v}(d)|}$

- A document may have a high score for a query even if it does not contain all query terms.

LINEAR ALGEBRA APPROACH TO TERM VECTOR

- Example:

  DOC 1 → Linear (3 times), algebra (1 times), approach (3 times)

  DOC 2 → Linear (1 times), algebra (2 times), approach (4 times)

  DOC 3 → Linear (2 times), algebra (3 times), approach (0 times)

  Query → Approach

Term	DOC 1	DOC 2	DOC 3	Query
Linear	3	1	2	0
Algebra	1	2	3	0
Approach	3	4	0	1
Co-ordinate	(3, 1, 3)	(1, 2, 4)	(2, 3, 0)	(0, 0, 1)
Magnitude (L_d)	$\sqrt{19}$	$\sqrt{21}$	$\sqrt{13}$	$\sqrt{1}$

A = term – document $\begin{bmatrix} 3 & 1 & 2 \\ 1 & 2 & 3 \\ 3 & 4 & 0 \end{bmatrix}$

q = query matrix = $\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$ and $q^T = [0 \quad 0 \quad 1]$

Normalize A, $\begin{bmatrix} \frac{3}{\sqrt{19}} & \frac{1}{\sqrt{21}} & \frac{2}{\sqrt{13}} \\ \frac{1}{\sqrt{19}} & \frac{2}{\sqrt{21}} & \frac{3}{\sqrt{13}} \\ \frac{3}{\sqrt{19}} & \frac{4}{\sqrt{21}} & \frac{0}{\sqrt{13}} \end{bmatrix}$

Now, $q^T A = [\frac{3}{\sqrt{19}} \quad \frac{4}{\sqrt{21}} \quad \frac{0}{\sqrt{13}}]$

$= (0.68, 0.87, 0)$

i.e. sim (q, DOC 1) = 0.68; sim (q, DOC 2) = 0.87; sim (q, DOC 3) = 0

WEIGHTING

- Weight of a term is a value given to the term.
- Value is the dependent factor of its occurrence in the document.
- Weight of a term is a basic element for the document ranking.
- Weighting mechanism:

(1) Term Frequency

- Term frequency is a measure of how often a term is found in a collection of documents.
- A reasonable scoring mechanism is computed a score for each query terms that matches with the document terms.
- Count the frequency of the terms that matches between the query terms and the document terms list.
- Denoted by $tf_{t,d}$.

(2) Inverse Document Frequency

- Term frequency suffers from a critical problem that all terms are considered equally important.
- In fact, certain terms have little or no selective power in determining relevance.
- For example: a collection of documents of the "Noodle" industry is likely to have the term "Noodle" in almost every document.
- Terms which appear very few in numbers may have higher probability of being relevant.
- So, we have to scale down the term weights of term with high collection frequency.
- Collection frequency is the total number of occurrence of a term in the collection.
- Document frequency is the number of documents in the collection that contain a term t.

Words	c.f	df
Book	10200	8532
Pen	10198	4502

$$idf_t = \log \frac{N}{df_t}$$

- For example:

Terms (t)	df_t	idf_t
Computer	1054	0.152
Monitor	508	0.470
Keyboard	475	0.500
Device	1247	0.080
Optical	1500	0

N = Total number of documents = 1500

- It is seen that the term having the highest df has the lowest idf and vice-versa.

TF – IDF WEIGHTING

- Terms are weighted according to a given weighting model which may include local weight, global weight or both.
- Local weights are functions of how many times each term appear in a document.
- Global weights are functions of how many times each term appears in the entire collection.
- The tf – idf weight for a term t in a document d is given by, tf – $idf_{t,d} = tf_{t,d}$ X idf_t , which is
  - Highest when t occurs within a small number of documents.
  - Lowers when the term t occurs fewer times in a document.
  - Lowest when the terms t occurs in virtually all documents.

ALGORITHM (VECTOR SPACE MODEL FOR DOCUMENT RANKING)

- A term document matrix 'A' is constructed.
- Weight for each element of the matrix is defined, $a_{ij} = L_{ij}$ X G_i X N_j

  where, L_{ij} = local weight of a term i in document j($tf_{i,j}$)

  G_i = global weight (idf_i)

  N_j = Normalization function = 1/l; l = Euclidean distance of document j
- Query matrix Q is defined.
- A X Q^T is computed.
- Obtained result shows the rank of the document.

CHOOSING A DOCUMENT UNIT

- Determine what the document unit for indexing is.
- For very long documents, the issue of indexing granularity arises.
- For example: for a collection of books, it would usually be a bad idea to index an entire book as a document.
- A search for "Chinese toys" might bring up a book that mentions "China" in the first chapter and "toys" in the last chapter but this does not make it relevant to the query.
- Instead we may well wish to index each chapter or paragraph as mini-document.
- Matches are then more likely to be relevant.

TOKENIZATION

- Given a character sequence and a defined document unit, tokenization is the task of chopping it up into pieces called tokens.
- So it is the process of breaking a stream of text up into words, phrase, symbols or other meaningful element called tokens.
- The list of tokens becomes input for further processing such as parsing, text mining, etc.
- During this phase all remaining text is parsed, lowercased and all punctuation removed.
- For example: Input: Friends, Romans, Countrymen, Lend me your ears

  Output: |Friends| |Romans| |Countrymen| |Lend| |me| |your| |ears|
- A token is an instance of characters in some particular document that are grouped together as a useful semantic unit for processing.
- A type is a class of all tokens containing the same character sequence.
- A term is a type that is included in the IR systems dictionary, i.e. a term means a normalized document.
- For example: if document to be indexed is 'to sleep per chance to dream'
  - There are 5 tokens.
  - There are 4 types (2 instances of "to").
  - There are 3 terms ("to" is defined as stop word)

- Issues of tokenization are language specific.
- It thus requires the language of the document to be known.
- Computer technology has introduced new types of character sequences that a tokenizer should probably tokenize as a single token.
- For example: email id → gblack@gmail.com
  URLs → http://stuf.big.com/new/special .htm
  IP address → 192.168.0.1
- In English, hyphenation is used for various purposes ranging from splitting up vowels in words (co-education), to joining noun as names (Hewlett-Packaged).
- The first one can be regarded as one token (co-education), but difficult in second one.

DIFFICULTIES OF TOKENIZATION

- Splitting on white spaces can also split what should be regarded as a single token.
- Splitting on spaces can cause bad retrieval result.
- Example: search for "York University" mainly returns documents containing "New York University".
- Regarding to hyphen and space, a query for "over-eager", should search for "over-eager" OR "over eager" OR "overeager".
- Each new language presents some new issues.
- The languages like Chinese, Japanese; there is no space as splitter.
- In such cases, we use word segmentation.
- Segmentation is the method of taking the longest vocabulary match with some heuristic for unknown words to use of machine learning such as HMM (Hidden Marker Model).

DROPPING COMMON TERMS (STOP WORDS)

- Sometimes, some extremely common words which would appear to be of little value in helping select documents matching a user need are excluded from the vocabulary.
- These words are called stop words.

- The general strategy for determining a stop list is to sort the terms by collection frequency and then to take the most frequent terms and are then discarded during indexing.
- Using a stop list significantly reduce the number of postings that a system has to store.
- For example: a, am, and, are, as, at, be, by, for, from, has, he, in, is, it, its, of, on, that, the, to, was, were, will, with, etc.
- A lot of time not indexing stop words does little harm.
- For example: the phrase query "President of the United States" which contains two stop words, is more precise that "President" AND "United States".
- Also, the meaning of "flights to London" is likely to be lost if the word "to" is stopped out.
- A search for Vannevar Bush's article "As we may think" will be difficult if the first three stopped words are dropped and the system simple search for documents containing word "think".
- Some song titles (to be or not to be, let it be, I don't want to be) are common with stop words.
- So IR system has focused precisely on how we can exploit the statistics of language so as to be able to cope with common words in better ways.

NORMALIZATION (EQUIVALENCE CLASSING OF TERMS)

- After document tokenization, we have to match query tokens to documents token lists, but it is somehow difficult.
- There are cases, where tokens are not quite the same, but we still want to match them.
- Example: U.S.A should match (USA) or even (US).
- Token normalization is about transforming tokens into a standard form.
- This allows matches to occur despite superficial differences.
- Usual way to normalize is to create equivalence classes.
- Example: anti-discriminating and anti-discriminatory are both in same class, so that searches for one term will retrieve documents that contain either.
- Alternative to equivalence classes are explicit rules.
- Example: window → window, windows

  windows → Windows, windows

- But some normalization may do more harm than good.
- Example: WHO → who

STEMMING AND LEMMATIZATION

- For grammatical reasons, documents are going to use different forms of a word such as organize, organizes and organizing.
- Additionally, there are families of derivationally related words with similar meanings such as democracy, democratic and democratization.
- The goal of both stemming and lemmatization is to reduce inflectional forms.
- Example: am, are, is → be
  car, cars, car's, cars' → car
- Example: the result of this mapping of text will be something like "the boy's cars are different colors" → "the boy car be differ color".
- Stemming is defined as crude heuristic process that chops off the ends of words.
- Language dependent.
- Works quite well for English language.
- Example: Automate automatic, automation all reduce to automat.
- Lemmatization usually refers to doing things properly with the use of vocabulary and morphological analysis of words.
- For example: with the token "saw", stemming might return "s" whereas lemmatization would attempt to return "see".

PORTER ALGORITHM

- Most common algorithm for stemming English.
- Result suggests that is at least as good as other stemming option.
- Removing suffixes by automatic means is an operation which is especially useful in the field of IR.
- Terms with a common stem will usually have similar meanings.
- For example: CONNECT, CONNECTED, CONNECTING, CONNECTION, CONNECTIONS.

- The performance of an IR system will be improved if term groups are conflated into a single term.
- This may be done by removal of the various suffixes "ED", "ING", "ION", "IONS" to leave the single term CONNECT.
- A consonant in a word is a letter other than A, E, I, O or U and other than Y preceded by a consonant.
- Example: in "TOY", the consonant are T and Y.
- In "SYZYGY", they are S, Z and G.
- If a letter is not a consonant, it is a vowel.
- A consonant is denoted by c, a vowel is denoted by v.
- A list ccc... of length greater than 0 will be denoted by c.
- A list vvv... of length greater than 0 will be denoted by v.
- Any word has one of the following forms: c...c, c...v, v...v, v...c
- These may all be represented by the single form: [c]vcvc...[v], where the square brackets denote arbitrary presence of their contents.
- This may again be written as [c] $(vc)^m$ [v], where, m is called the measure of word or word part represented in "vc" form.
- Examples: m = 0 TR, EE, TREE, Y, BY
  m = 1 TROUBLE, OATS, TREES, IVY
  m = 2 TROUBLES, PRIVATE, OATEN, ORRERY
- The rules for removing a suffix will be given in the form:
  <condition> S_1 → S_2
- Example:

Rules	Example
SSES → SS	caresses → caress
IES → I	ponies → poni
SS → SS	caress → caress
S →	cats → cat
(m > 1) EMENT	Replacement → replace (btu not cement → c) cement → cement

PHRASE QUERIES

- We want to answer a query such as "Stanford University" as a phrase.
- Thus, "the inventor Stanford Orshinsky never went to university" shouldn't be matched.
- About 10% of web queries are phrase queries.

BUILDING AN INVERTED INDEX

- Inverted index, also called postings file or inverted file, is an index data structure storing a mapping from content, such as words or numbers to its locations in a database file or in a document or a set of documents.
- The purpose of an inverted index is to allow fast full text searches.
- An index always maps back from terms to the parts of a document where they occur.
- A dictionary of terms is kept.
- Then for each term, a list is maintained in which documents the term occurs in.
- Each item in the list which records that a term appeared in a document is called a posting.
- The list is then posting list.
- The dictionary will be sorted alphabetically and each postings list is sorted by document ID.
- Example: DOC 1 = new home sales top forecasts

  DOC 2 = home sales rise in July

  DOC 3 = increase in home sales in July

  DOC 4 = July new home sales rise

  forecasts → |DOC 1|

  home → |DOC 1|→|DOC 2|→|DOC 3|→|DOC 4|→ posting list

  increase → |DOC 3|

  July → |DOC 2|→|DOC 3|→|DOC 4|→ increasing

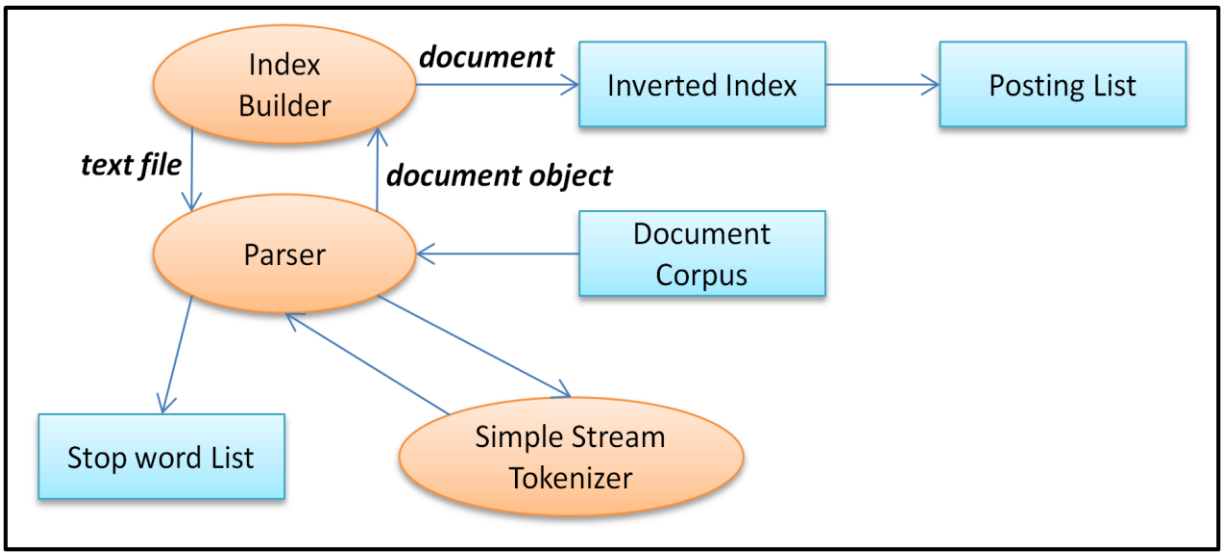
  new → |DOC 1|→|DOC 4|

  rise → |DOC 2|→|DOC 4|

  sales → |DOC 1|→|DOC 2|→|DOC 3|→|DOC 4|

  top → |DOC 1|

INDEXING ARCHITECTURE



BIWORD INDEX

- Index every consecutive pair of terms in the text as a phrase.
- Example: Friends, Romans, Countrymen would generate two bi-words "Friends Romans" and "Romans Countrymen".
- Each of these bi-word is now a vocabulary term.

POSITIONAL INDEXES

- Posting lists in a positional index in which each posting is a docID and a list of positions.
- Example:

Cat, 100

<1, 6 :<7, 18, 33, 72, 86, 231>;

2, 5 : <1, 17, 74, 222, 255>;

4, 2 : <8, 16>;

..

..

>

- The word "cat" has a document frequency 100 and occurs 6 times in document 1 at positions 7, 18, 33, 72, 86, 231 and so on.

SPARSE VECTORS

- Most documents and queries do not contain most word, so vectors are sparse.
  i.e. most entries are zero (0).
- Need efficient methods for storing and computing with sparse vectors.
- We can use sparse vectors as lists, sparse vectors as trees, sparse vectors as Hash Table.

EVALUATION IN INFORMATION RETRIEVAL

- There are many retrieval models/algorithms/systems.
- Which one is the best?
- Which is the best component for?
  - Ranking function (dot product, cosine, ..)
  - Term selection (stop word removal, stemming, ..)
  - Term weighting (TF, TF-IDF, ..)
- Effectiveness is related to the relevancy of retrieved items.
- Relevancy from a human stand point is,
  - Subjective → depends upon a specific user's judgment.
  - Situational → relates to user's current needs.
  - Cognitive → depends on human perception.
  - Dynamic → changes over time.
- Key utility measure is user happiness.
- Speed of response is a factor of user happiness.
- But blindingly fast with useless answers do not make a user happy.
- The standard approach to IR system evaluation revolves around the notion of relevant and non-relevant documents.
- With a user information need, a document in the collection is given a binary classification as either relevant or non-relevant.
- This decision is referred to as the gold standard or ground truth judgment of relevance.
- Relevance is assessed relative to an information need, not a query.
- For example: an information need might be "Information on whether drinking red wine is more effective at reducing your risk of heart attacks than white wine".
- This might be translated to a query such as "Wine AND red AND white AND heart AND attack AND effective".
- A document is relevant if it addresses the stated information need not because it just happens to contain all the words in the query.

- If a user type "Python" into a web search engine, they might want to know where they can purchase a pet python, or they might want information on the programming language "python".
- So, from a one word query, it is very difficult for a system to know what an information need is.

EVALUATION OF UNRANKED RETRIEVAL SETS

- There are two measures:
  - Precision (P)
    - Precision is the fraction of retrieved documents that are relevant.
    - Indicates what proportion of the retrieved documents is relevant.
    - Precision = $\frac{\#(relevant\ \ time\ \ retrieved)}{\#(retrieved\ \ items)}$ = P(relevant/retrieved)
  - Recall (R)
    - Recall is the fraction of relevant document that are retrieved.
    - Indicates what proportion of all the relevant documents have been retrieved from the collection.
    - Recall = $\frac{\#(relevant\ \ items\ \ retrieved)}{\#(relevant\ \ items}$ = P(retrieved/relevant)

QUESTION:

An IR system returns 8 relevant documents and 10 non-relevant documents. There are a total of 20 relevant documents in the collection. What is the precision of the system on this search and what was its recall?

- Precision and recall can also be expressed in the following terms:

	Relevant	Non-Relevant
Retrieved	True Positive (tp)	False Positive (fp)
Not-retrieved	False Negative (fn)	True Negative (tn)

- Precision = $\frac{tp}{tp+fp}$ and Recall = $\frac{tp}{tp+fn}$
- An alternative way to judge an information retrieval system is by its accuracy.
- Accuracy is the fraction of its classification that is correct.

- Accuracy = $\frac{tp+tn}{tp+fp+fn+tn}$
- A single measure that trades off precision versus recall is the F-measure which is the weighted harmonic mean of precision and recall.

  i.e. F-measure = $\frac{2PR}{P+R}$
- *Why do we use harmonic mean rather than the simple arithmetic mean?*
  - It is because we can get 100% recall by just returning all documents and therefore we can always get a 50% arithmetic mean by the same process.
  - This strongly suggests that the arithmetic mean in an unsuitable measure to USP.
  - The harmonic mean is always less than or equal to the arithmetic mean and geometric mean.

EVALUATION OF RANKED RETRIEVAL RESULTS

- Number of relevant = 6

n	doc #	Relevant
1	588	X → P = ? R = ? P = 1/1 R = 1/6
2	589	X → P = ? R = ? P = 2/2 R = 2/6
3	576	
4	590	X → P = ? R = ? P = 3/4 R = 3/6
5	986	
6	592	X → P = ? R = ?
7	984	
8	988	
9	578	
10	985	
11	103	
12	591	
13	772	X → P = ? R = ?
14	990	

- In a ranked retrieval context, the set of retrieved documents are given by the top k retrieved documents.
- If $(k+1)^{th}$ document retrieved is non-relevant then recall is the same as for the top k documents but precision is dropped.

- If it is relevant, then both precision and recall increases.

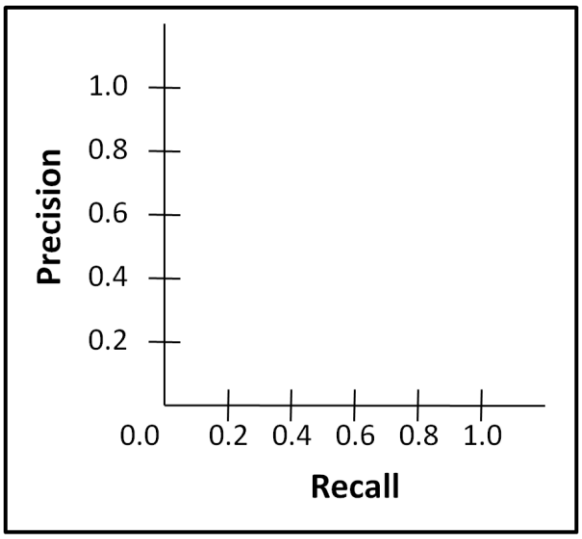


Fig: Precision/Recall Graph (Saw-Tooth Shape)

SYSTEM QUALITY

- There are many practical bench marks on which to rate an IR system beyond its retrieval quality, which includes:
  - How fast does it index? [i.e. how many documents per hour does it can index.]
  - How fast does it search?
  - How expressive is its query language? How fast is it on complex queries?
  - How large is its document collection?

USER UTILITY

- A way of quantifying user happiness is based on relevance, speed and user interface of a system.
- For a web search engine, happy search users are those who find what they want.
- One indirect measure of such users is that they tend to return the same engine.
- Advertisers are also users of modern web search engines and they are happy, if customer's clicked through to their sites and then make purchases.

QUERY LANGUAGE

- A query is the formulation of a user information need.
- Most query languages try to use the content (semantics) and the structure of the text (syntax), to find relevant documents.
- The retrieved unit is the basic element which can be retrieved as an answer to a query.
- The retrieval unit can be a file, a document, a webpage, etc.
  - Keyword Based Queries
    - Single Word Queries
    - Context Queries
    - Boolean Queries
    - Natural Languages
  - Pattern Matching
  - Structural Queries

1. KEYWORD BASED QUERY

- A query is composed of keywords and the documents containing such keyword are searched for.
  - Single Word Queries
    - The most elementary query that can be formulated in text retrieval is a word.
    - Documents are assumed to be long sequences of words.
    - Word is a sequence of letters surrounded by separators.
    - Some characters are not letters but do not split a word. For eg: hyphen (co-education).
    - The result of word queries is the set of documents containing at least one of the words of the query.
    - Further, the set of resulting documents are ranked according to a degree of similarity to the query, i.e. tf, tdf.
  - Context Queries
    - Many systems complement single word queries with the ability to search words in a given context, i.e. hear other words.

- Words which appear near each other may signal a higher likelihood of relevance than if they appear apart.
- Two types of queries:

1. *Phrasal Queries*
   - Phrase is a sequence of single word queries.
   - An occurrence of the phrase is a sequence of words.
   - Relevance documents are those that contain a specific phrase, i.e. ordered list of contiguous word. For example: "buy camera" matches "buy a camera", "buying the cameras", etc.
   - Must have an inverted index that also stores positions of each keyword in a document.
   - Retrieving a document and position for each individual word, intersect documents and then finally checks for ordered contiguity of keyword positions.

2. *Proximity Queries*
   - A more relaxed version of the phrase query.
   - In this case, a sequence of single words or phrase is given together with a maximum allowed distance between them.
   - List of words with specific maximal distance constraints between terms.
   - For example: "dogs" and "race" within 4 words match.

     "……dog will begin the race……"
   - May also perform stemming and/or/not count stop words.

- Boolean Queries
  - A Boolean query has a syntax composed of basic queries that retrieve documents and of Boolean operators which work on their operands and deliver set of documents.
  - Since this schema is general compositional, a query syntax tree is naturally defined, where the leaves corresponds to the basic queries and the internal nodes to the operators.

- For example:

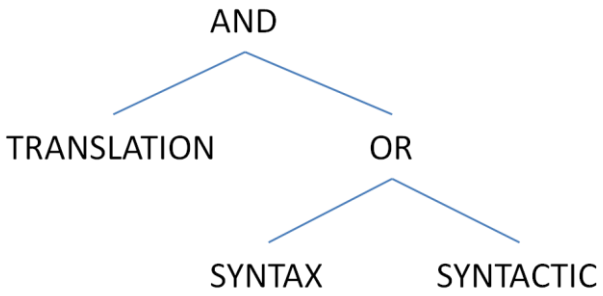


Fig.: it will retrieve all the documents which contain the word "translate" as well as either the word "syntax" or the word "syntactic".

- Operators most commonly used in Boolean queries are:
  1. *OR (e1 OR e2)*
  2. *AND (e1 AND e2)*
  3. *BUT (e1 BUT e2)* → *satisfy e1 but not e2*

- Natural Language
  - Full text queries as arbitrary strings.
  - All the documents matching a portion of the user query are retrieved.
  - Higher ranking is assigned to those documents matching more part of the query.
  - Typically, such process is used by vector space model.

2. PATTERN MATCHING

- A pattern is a set of syntactic features that must occur in a text segment.
- Those segments satisfying the pattern specifications are said to match the pattern.
- Examples:
  1. ***Prefixes:*** Pattern that matches start of the word. For example: "anti" means "antiquity", antibody", etc.

  2. ***Suffixes:*** Pattern that matches end of the word. For example: "ix" matches "fix", "matrix", etc.

3. ***Substrings:*** Pattern that matches arbitrary sub-sequence of characters. For example: "rapt" matches "enrapture", "velociraptor", etc.

4. ***Ranges:*** Pair of strings that matches any word alphabetically between them. For example: "tin" to "tix" matches "tip", "tire", "title", etc.

ALLOWING ERRORS

- What if query or document contains misspellings?
- Judge the similarity of words using edit distance.

EDIT (LEVENSTEIN) DISTANCE

- Minimum number of character deletions, additions or replacements needed to make two strings equivalent.
- For example: "misspell" to "misspell" is distance 1.
  "misspell" to "mistell" is distance 2.
  "misspell" to "misspelling" is distance 3.

REGULAR EXPRESSIONS

- Some text retrieval systems allow searching for regular expression.
- Examples:
  1. (u/e) nabl (e/ing) matches
     - unable, unabling, enable, enabling
  2. (un/en) *able matches
     - able, unable, untenable, enununenable

RELEVANCE FEEDBACK & QUERY EXPANSION

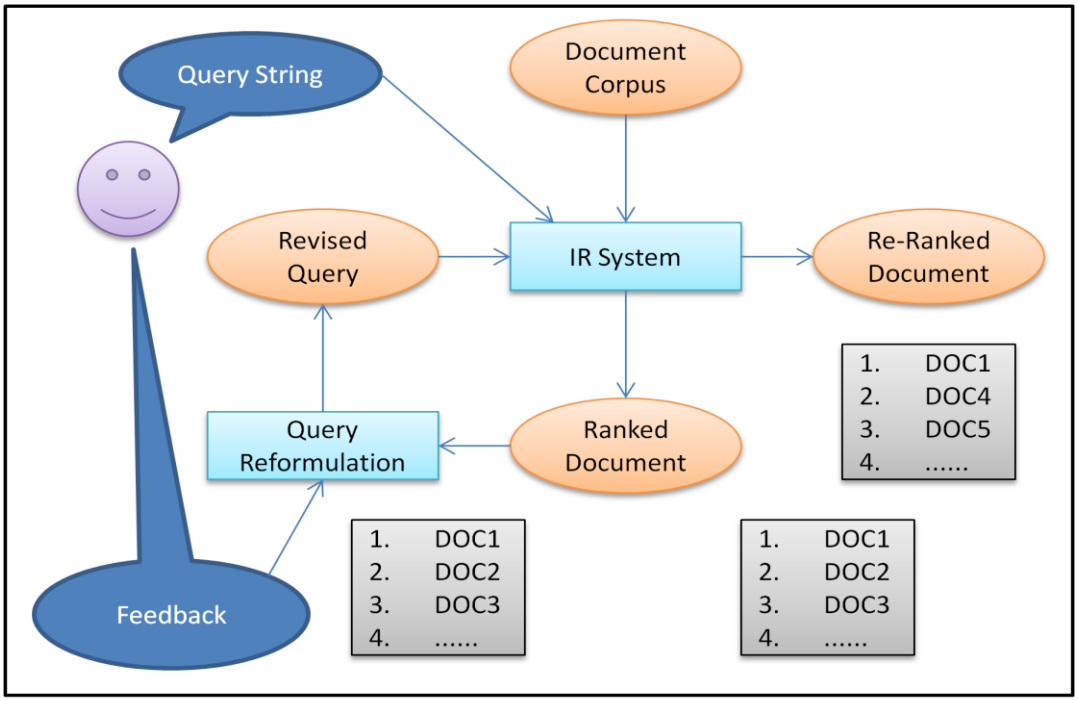
- In most collections, the same concept may be referred to using different words (synonyms).
- For example: a search for 'restaurant" to match "café".

- Such problem can be addressed by user manually.
- Also the system can help with query refinement.
- Such methods for tackling this problem by system are classified into two classes.

1. *Global Methods*
   - Query expansion/reformulation with a thesaurus or word net.
   - Techniques like spelling correction.
2. *Local Methods*
   - Relevance feedback
   - Pseudo-relevance feedback
   - Indirect relevance feedback

RELEVANCE FEEDBACK

- The idea of relevance feedback is to involve the user in the retrieval process so as to improve the final result set.
- In particular, the user gives feedback on the relevance of documents in an initial set of results.



RELEVANCE FEEDBACK ARCHITECTURE

BASIC PROCEDURE

- The user issues a query.
- The system returns an initial set of retrieval results.
- The user marks some returned documents as relevant or non-relevant.
- The system computes a better representation of the information need based on the user feedback.
- The system displays a revised set of retrieval results.
- Seeing some documents may lead users to refine their understanding of the information they are seeking.
- Image search provides a good example or relevance feedback.

WHEN DOES RELEVANCE FEEDBACK WORK?

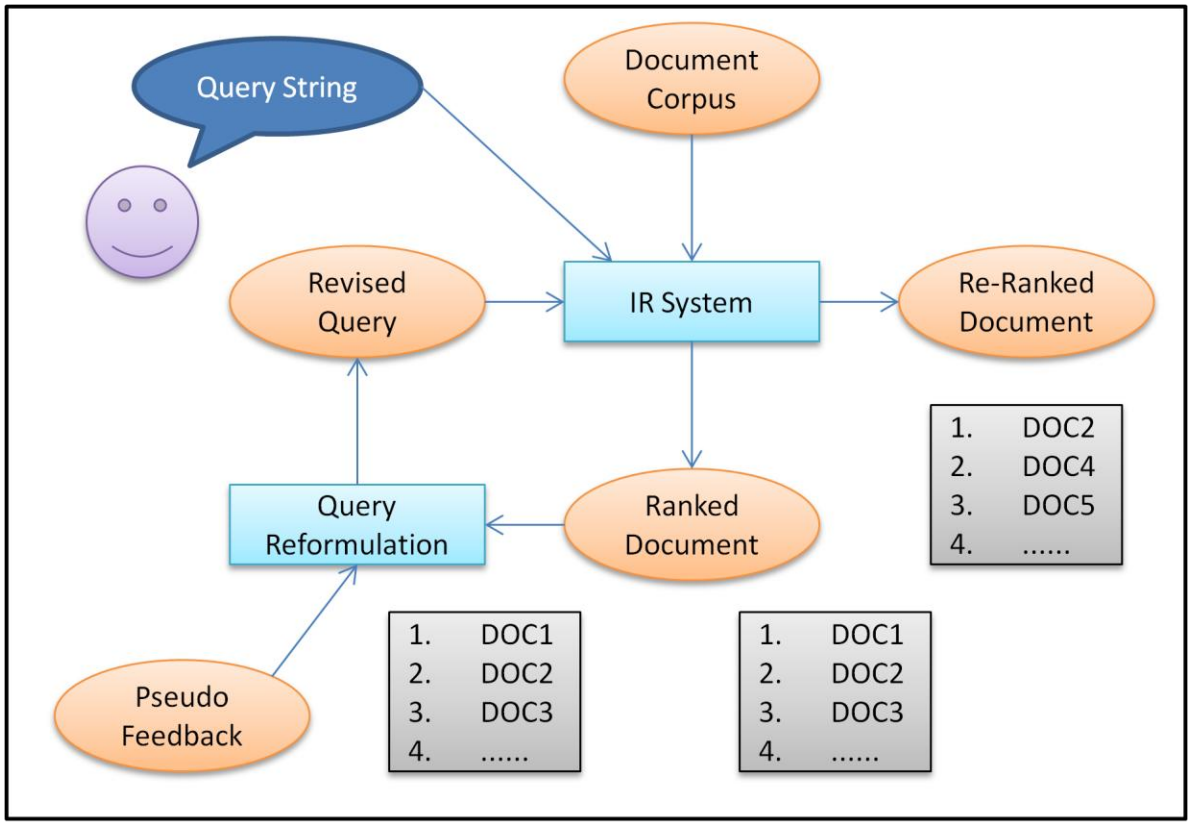
- The success of relevance feedback depends on certain assumptions.
- Firstly, the user has to have sufficient knowledge to be able to make an initial query which is at least somewhere close to the documents they desire.
- Secondly, the relevance feedback approach requires relevant documents to be similar to each other, i.e. they should cluster.

CASES WHERE RELEVANCE FEEDBACK ALONE IS NOT SUFFICIENT

1. Misspellings
   - If the user spells a term in different way to the way it is spelled in any document in the collection then relevance feedback is unlikely to be effective.
2. Cross language information retrieval (CLIR)
   - It is difficult to cluster the same relevance documents in different language rather than some language.
3. Mismatch of searcher's vocabulary versus collection vocabulary
   - For example: if the user searches for "laptop" but all the documents use the term "notebook computer" then the query fails.

PSEUDO RELEVANCE FEEDBACK

- Also called blind relevance feedback.
- Provides a method for automatic local analysis.
- Use relevance feedback methods without explicit user input.
- Just assume the top m retrieved documents are relevant and use them to reformulate the query.
- Allows for query expansion that includes terms that are correlated with the query terms.



PSEUDO FEEDBACK ARCHITECTURE

INDIRECT RELEVANCE FEEDBACK

- On the web, direct hit introduced the idea of ranking more highly documents that users choose at more often.
- Clicks on the links were assumed to indicate that the page was likely relevant to the query.
- Click stream mining.

THESAURUS

- A thesaurus provides information on synonyms and semantically related words and phrases.
- The IR system might also suggest search terms by means of a thesaurus.
- A user can also be allowed to browse lists of the terms that are in the inverted index and thus find good terms that appear in the collection.
- For example: Physician

  Syn (Synonyms): doc, doctor, MD, medical, medicines, medico

  Rel (Related): medic, general practitioner, surgeon

THESAURUS BASED QUERY EXPANSION

- For each term t in a query, expand the query with synonyms and related words of t from the thesaurus.
- May weight added terms less than original query terms.
- Generally increases recall.
- May significantly decrease precision, particularly with ambiguous terms.
- For example: "interest rate" → "interest rate fascinate evaluate".

WORD NET

- Word net is a lexical database for the English language.
- It groups English words into sets of synonyms called synsets providing various semantic relations between these synonym sets.
- Word net is more detailed database of semantic relationships between English words.
- Developed by famous cognitive psychologist George Miller and a team at Princeton University.
- About 144000 English words.

WORD NET SYNSET RELATIONSHIP

1. Antonym: front → back
2. Attribute: benelovelence → good (noun to adjective)

3. Pertainym: alphabetical → alphabet (adjective to noun)
4. Similar: unquestioning → absolute
5. Cause: kill → die
6. Entailment: breathe → inhale
7. Holonym: chapter → text (part of)
8. Meronym: computer → CPU (whole of)
9. Hyponym: tree → plant (specialization)
10. Hypernym: fruit → apple (generalization)

WORD NET QUERY EXPANSION

- Add synonyms in the same synset.
- Add hyponyms to add specialized term.
- Add hypernyms to generalize a query.
- *Note:* *Y is a holonym of X, if X is a part of Y*
  Y is a meronym of X, if Y is a part of X

SPELLING CORRECTION

- Correcting spelling errors in queries.
- For instance, we may wish to retrieve documents containing the term "carrot" when the user types the query "carot".
- Two steps to solve this problem:
  i. *Edit distance*
  ii. *K-gram overlap*

IMPLEMENTING SPELLING CORRECTION

- Of various alternative correct spellings for a misspelled query, choose the nearest one (i.e. the smallest edit distance).

- When two correctly spelled queries are tied, select the one that is more common. For example: "grunt" and "grant" both seem equally plausible as correction for "grnt". Correction is done then by examining which term (grunt or grant) is typed by the user in the query.

FORMS OF SPELLING CORRECTION

- Two forms:
  i. *Isolated term correction*
  ii. *Context sensitive correction*
- In isolated term correction, correct a single query term at a time, even when we have multiple term queries.
- But sometimes, such isolated term correction fails to detect.
- For example: "flew form Nepal" → contains the misspelling of the term "from" but not detected by isolated term correction. In such case we need context sensitive correction.

EDIT DISTANCE

- Given two character strings S_1 and S_2, the edit distance between them is the minimum number of edit operations required to transform S_1 into S_2.
- Most commonly edit operations include the following operations:
  i. Insert a character into a string.
  ii. Delete a character from a string.
  iii. Replace a character of string by another character.
- Edit distance is also called Levenstein distance.
- **Algorithm:**

```
EDIT DISTANCE (S1, S2)
int M[i, j] = 0
for i = 1 to |S1|
        do M[i, 0] = i
for j = 1 to |S2|
```

do M[0, j] = j

for i = 1 to $|S_1|$

do for j = 1 to $|S_2|$

do M[i, j] = min { M[i−1, j−1] + if (S_1[i] = S_2[j]) then 0 else 1, M[i−1, j] + 1, M[i, j−1] + 1}

return M[$|S_1|$, $|S_2|$]

- The [i, j] entry of the matrix (after execution of algorithm) will hold the edit distance between the strings consisting of the first i characters of S_1 and first j characters of S_2.

K-GRAM INDEXES FOR SPELLING CORRECTION

- A k-gram is a sequence of k characters.
- Example: "cas", "ast", "stl" are 3 grams occurring in term "castle".
- Use the k-gram index to retrieve vocabulary terms that have many k-grams in common with the query.
- Example:

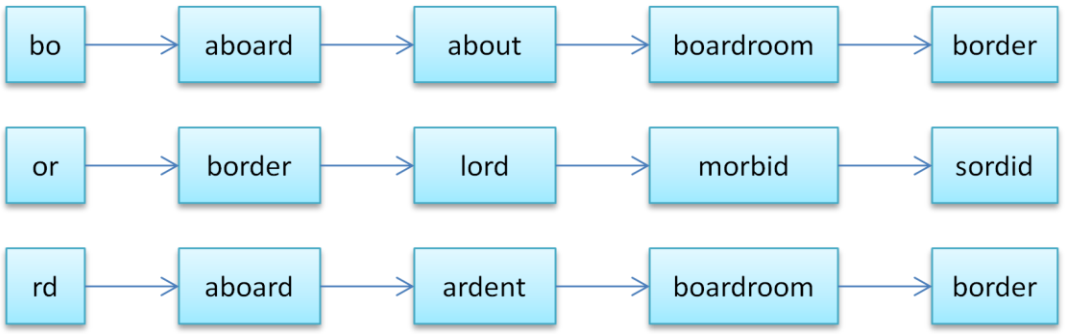


Fig.: Matching at least two of the three 2 gram in the query "bord"

- Suppose we want to retrieve vocabulary terms that contained at least two of these bigrams. We would enumerate aboard, boardroom and border.

STATISTICAL PROPERTIES OF TERMS IN IR

- The number of terms is the main factor in determining the size of the dictionary.
- Stemming reduces the number of distinct terms by 14%.
- How is the frequency of different words distributed?
- How fast does vocabulary size grow with the size of a corpus?
- Such factors affect the performance of IR.
- A few words are very common.
- Two most frequent words ("the", "of") can account for about 10% of words occurrences.

ZIPF'S LAW

- Zipf's law states that some corpus of natural language, the frequency of any word is inversely proportional to its rank.
- Named after the Harvard linguistic professor George Kingsely Zipf.
- Is used to understand how terms are distributed across documents.
- It states that if t_1 is the most common term in the collection, t_2 is the next most common and so on, and then the collection frequency cf_i of the i^{th} most common term is proportional to 1/i, (i.e. $cf_i \propto 1/i$).
- So the most frequent word, three times as often as the third frequent one.
- Example: in Brown corpus, "the" is the most frequently occurring word and by itself accounts for nearly 7% of all words occurrences (69971 out of 1 million).
- Second place word "of" accounts for slightly over 35% of word (36411).
- Third place is "and" (28852).
- The intuition is that frequency decreases very rapidly with rank.

DOCUMENT PROCESSING

- Can be divided into the following five operations:
  1. Lexical analysis (Morphological analysis)
  2. Elimination of stop words

3. Stemming
4. Selection of index term
5. Construction of term categorization structure such as Thesaurus

LEXICAL ANALYSIS OF THE TEXT

- Lexical analysis is the process of converting a stream of characters (the text of documents) into a stream of words (the candidate words to be adopted as index terms).
- Basically space is involved as word separator, but however the following cases also have to consider, (1) Digits, (2) Hyphens, (3) Punctuation marks and (4) Case of the letters.

1. Digits

- Numbers are usually not good index terms because without a surrounding context, they are inherently vague.
- For example: a user is interested in documents about the number of deaths due to car accidents between the year 1910 and 1989.
- Such a request could be specified as the set of index terms {deaths, car accidents, year, 1910, 1989}.
- However the presence of the numbers 1910 and 1989 in the query could lead to retrieval a variety of documents which refer to either o these two years.
- Thus in general, numbers are disregarded as index terms.
- But numbers like 510 B.C., sequence of 16 digits verifying a credit card number might be index term.

2. Hyphens

- Pose another difficult decision to the lexical analyzer.
- Breaking up hyphenated words might be useful due to inconsistency of usage.
- For example: "state-of-the-art" and "state of the art" are identical.
- But there are words which includes hyphen as an integral part.
- For example: co-education, B-49, etc.

3. Punctuation Marks
   - Normally, punctuation marks are removed entirely in the process of lexical analysis, while some punctuation marks are integral part of the world.
   - For example: Dr., B.C., etc.

4. Case of letters
   - The case of letters is usually not important for the identification of index terms.
   - As a result, the lexical analyzer normally converts all the text to either lower or upper case.
   - But, it may not work all the time. For example: the words "Bank" and "bank" have different meaning. UNIX commands are in uppercase.

INDEX TERM SELECTION

- If a full text representation of the text is adopted then all words in the text are used as index terms.
- The alternative is not all words are used as index terms.
- This implies that the set of terms used as indices must be selected.
- In the area of bibliographic sciences, such a selection of index terms is usually done by a specialist.
- A good approach is the identification of noun groups.
- A sentence in natural language text is usually composed of nouns, pronouns, articles, verbs, adjectives, adverbs and connectives.
- Most of the semantics is carried by the noun words.
- So it is like to use the noun as index terms.
- Also, the combination of noun ("Computer Science") can also be used as index.
- A noun group is a set of nouns whose syntactic distance in the text does not exceed a predefined threshold (for example: 3).

THESAURI

- A thesaurus is a collection of words with its synonyms and related words.

- It consists of:
  1. *A precompiled list of important words in a given domain knowledge.*
  2. *For each word in the list, a set of related words.*
- Thesaurus provides a standard vocabulary for indexing and searching.
- The terms are the indexing components of the thesaurus.
- Terms are basically noun.
- Thesaurus also contain phrase if a single word is unable to express semantic meaning. For example: "ballistic missiles".
- Basically, the terms are used in plural form, since the thesaurus represents class.
- Sometimes it is need to specify the precise meaning of a context in a particular thesaurus. For example: "seal" has different meaning in context of "documents" and "marine animals".

METADATA

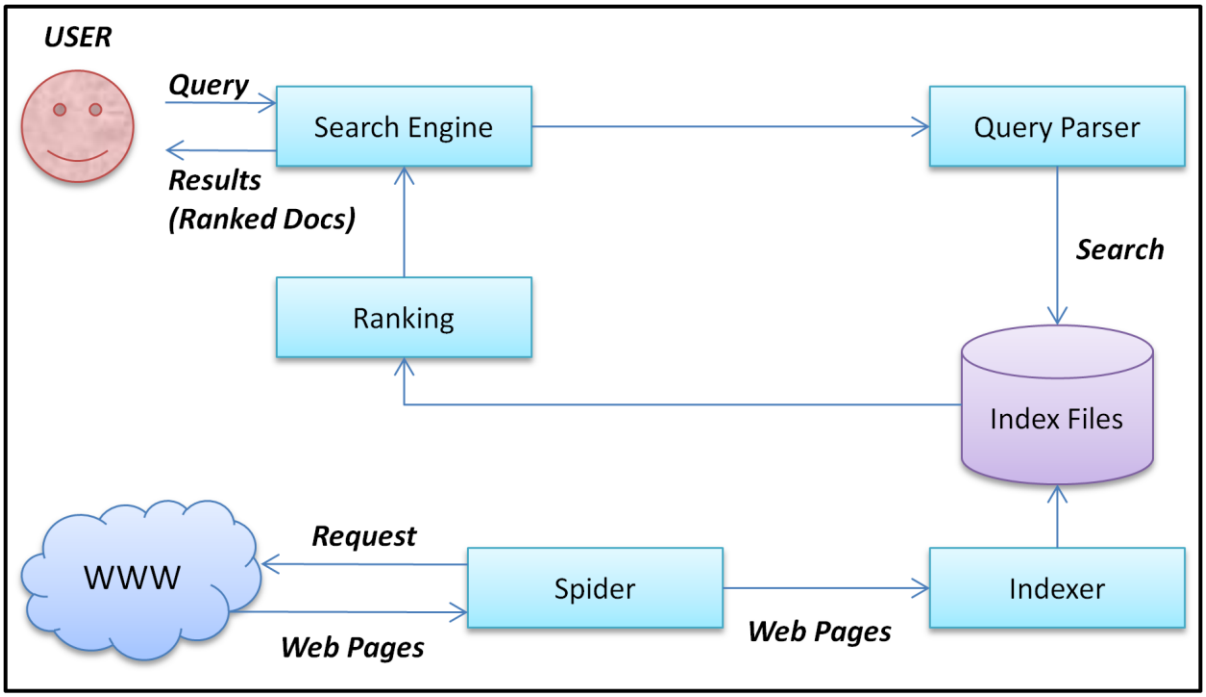
- Information about a document that may not be a part of the document itself, i.e. data about data.
  1. *Descriptive metadata*
  2. *Semantic metadata*
- Descriptive metadata is external to the meaning of the document.
- For example: author, title, source, date, ISBN, length, etc.
- Semantic metadata concerns the content (semantic meaning).
- For example: abstract, keywords, etc.

WEB METADATA

- Meta tag in HTML.
- For example: <meta name = "keywords" content = "pets, cats, dogs">.

SEARCH ENGINE

- A program that searches for documents for specified keyword and returns a list of the documents where the keywords are found.
- Typically, a search engine works by sending out a spider to fetch as many as documents.
- Another program indexer then reads these documents and creates index based on the word contained in each document such that only meaningful results are retrieved to query.
- A web search engine is designed to search the information on WWW.



ARCHITECTURE OF SEARCH ENGINE

HOW DOES SEARCH ENGINE WORK?

- A search engine operates in the following order:
  1. Web crawling
  2. Indexing
  3. Searching
- Web search engine works by storing information about many web pages which they retrieve.

- These pages are retrieved by a web crawler (sometimes also called spiders), i.e. automated web browser which follows every links on the site.
- Exclusions can be made by the use of robots.txt.
- The contexts of each page can be analyzed to determine how it should be indexed.
- When a user enters a query into a search engine, the engine examines and provides the listing of best matching web pages with ranking.

WEB CRAWLING

- Web crawling is the process by which we gather pages form the web, in order to index them and support a search engine.
- The feature of a crawler must provide; (1) *Robustness* [detect the spider trap] and (2) *Politeness* [follow the restrictions to spider (robots.txt)].

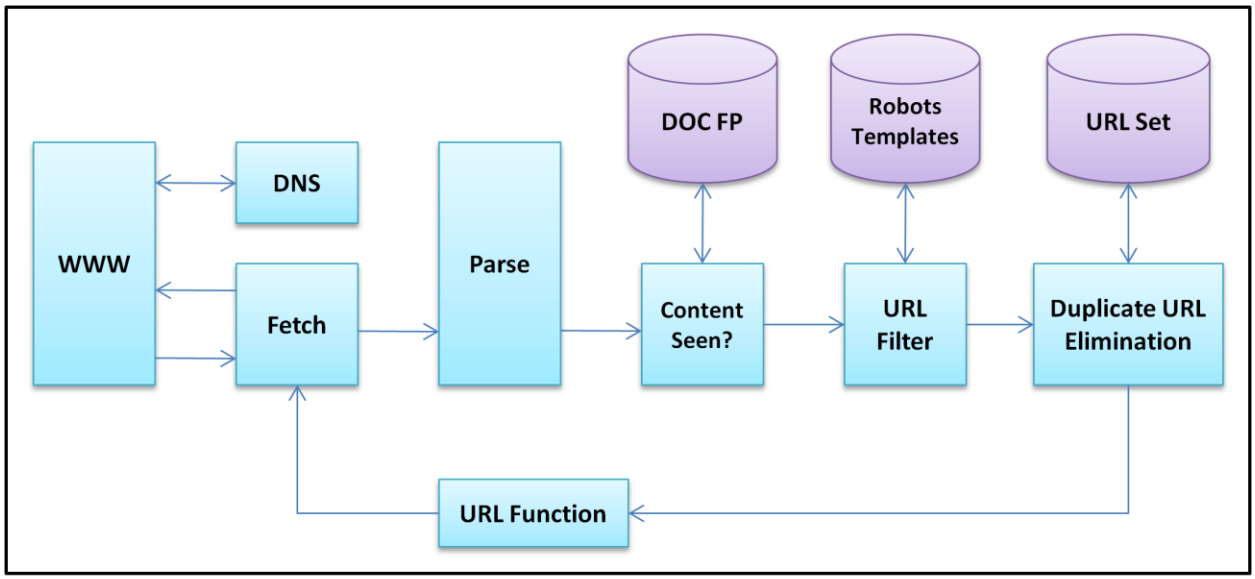
FEATURES A CRAWLER SHOULD PROVIDE

1. Distributed
2. Performance and Efficiency
3. Quality
4. Freshness
5. Extensible

CRAWLING OPERATION

- The crawler begins with one or more URLs that constitute a seed set.
- It picks a URL from this seed set, and then fetches the web page at that URL.
- The fetched page is then parsed to extract both the text and the links form the page.
- The extracted text is fed to a text indexer.
- The extracted links (URLs) are then added to a URL frontier which at all time consists of URLs whose corresponding pages have yet to be fetched by crawler.
- Initially URL frontier contains the seed set.

CRAWLER ARCHITECTURE



THE BASIC CRAWLER ARCHITECTURE

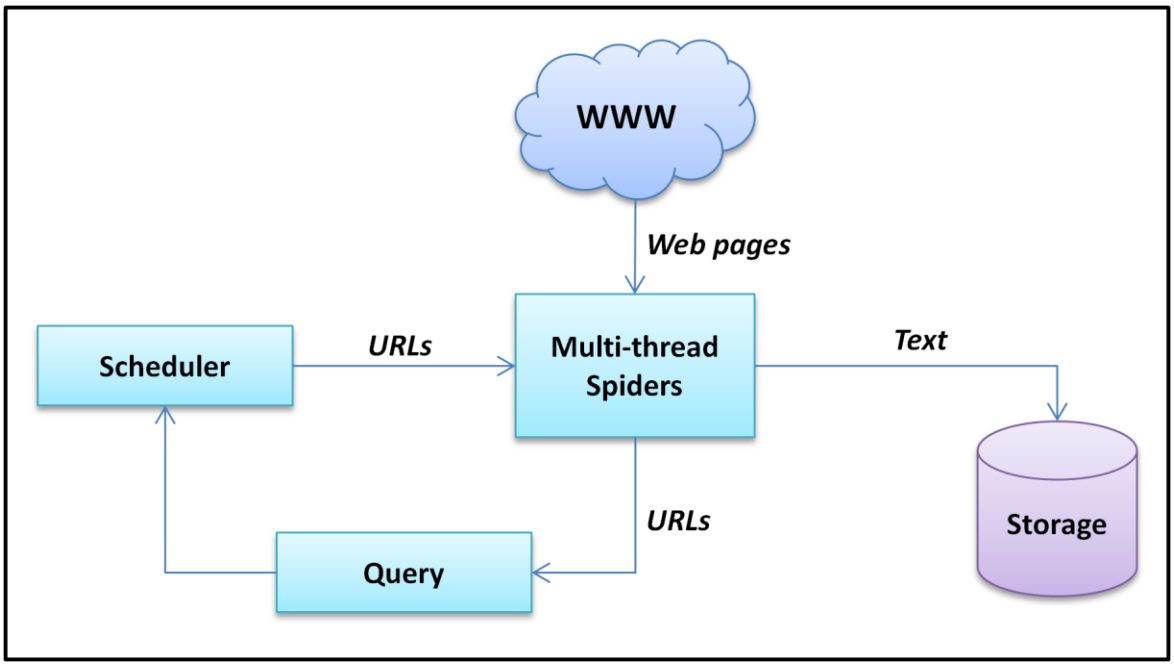
- URL Frontier: Contains URL's yet to be fetched.
- DNS Resolution Module: Determines the web server from which to fetch the page specified by the URL.
- Fetch Module: Retrieve web page at URL.
- Parsing Module: Extracts the text and set of links from a fetched web page.
- Duplicate Elimination Module: Determines whether an extracted link is already in the URL frontier.
- URL Filter: Determine whether the extracted link should be excluded from the URL frontier (for example: Robots Exclusion Protocol).
- Document Finger Print Module: Checks whether a web page with the same content has been already seen at another URL.

SEARCH ENGINE SPIDER

- A spider is a program that a search engine uses to seek out information on WWW as well as to index the information that it finds so that actual search results appear when a search query for a keyword is entered.

- The spider reads the text on the web page and records any hyperlinks found.
- The search engine spider then follows these URL's, spider those pages, collects all the data by saving copies of the web pages into the index or the search engine for use by visitors.
- Search engine spiders are always working, sometimes to index new web pages and sometimes to update ones that change frequently.
- Goal of search engine spiders is to supply up-to-date materials to search engine.
- There are four distinct styles of behavior of search engine spider. They are:
  1. Selection: decide which page needs to be downloaded.
  2. Re-visitation: to check for changes in pages that has already been indexed.
  3. Politeness: obey the restrictions to spider.
  4. Parallelization: spiders are working on parallelization to co-ordinate with other spiders.

STRUCTURE OF A SPIDER



SPIDERING ALGORITHM

- Initialize queue (Q) with initial set of known URL's.
- Until Q is empty or page or time limit exhausted,

  - Pop URL, L from front of Q.
  - If L is not to an HTML page (.gif, .jpeg, .ps, .pdf, .ppt, etc) then continue loop.
  - If L is already visited then continue loop.
  - Download page P for L.
  - If cannot download P (For example: 404 error, robot excluded) then continue loop.
  - Index P.
  - Parse P to obtain list of new links N.
  - Append N to the end of Q.
- Loop

MULTI-THREADED SPIDERING

- Bottleneck is network delay in downloading individual pages.
- Best to have multiple threads running in parallel, each requesting a page from different host.
- Distribute URL's to thread to guarantee equitable distribution of requests across different hosts to maximize throughput.
- Early Google spider had multiple co-ordinate crawlers with about 300 threads each. Together able to download 100 pages per second.

DIRECTED / FOCUSED SPIDERING

- Sort queue to explore more interesting pages first.
- Two styles of focus: (1) *Topic directed* and (2) *Link directed*.

1. Topic Directed Spidering
   - Assume desired description or sample pages of interest are given.
   - Sort queue of links by similarity like using cosine similarity of their source pages and/or anchor text to this topic description.
   - Explores pages related to a specific topic.
2. Link Directed Spidering
   - Monitor links and keep track of in-degree and out-degree of each page encountered.

- Sort queue of preferred popular pages with many incoming links (authorities).
- Sort queue to preferred summary pages with many outgoing links (hubs).

LINK ANALYSIS

- Use of hyperlinks for ranking web search results
- Link analysis is one of many factors considered by web search engines in computing a score for a web page on any given query
- Two methods for link analysis
  - Page rank
  - HITS (hyperlinks induced topic search)

PAGE RANK

- Developed by Larry Page at Stanford University.
- Link analysis algorithm
- A hyperlink to a page counts as a vote of support
- A page that is linked to by many pages receives a high rank and if there is no links to a web page there is no support for that page.
- Assigns to every node in the web graph a numerical score between 0 and 1 to each element of hyperlinked set of documents.
- The rank value indicates the importance of a particular page.
- A page rank of 0.5 means there is a 50% chance that a person clicking on a random link will be directed to the document with 0.5 page rank.
- Algorithm
  - Assume a small universe of four web pages A, B, C and D.
  - The initial approximation of Page Rank would be evenly divided between the four documents.
  - Hence each document would begin with an estimated Page Rank of 0.25.
  - If pages B, C and D each only link to A, they would each confer 0.25 page rank to A.

    i.e. PR(A) = PR(B) + PR(C) + PR(D) = 0.75

- Suppose that page B has link to page C as well as to page A, while pages d has links to all three pages.
- The value of link votes is divided among all the outbound links on the page.
- Thus B gives vote worth 0.125 to page A and a vote 0.125 to page C.
- Similarly, D's page rank is 0.083 (approximately)

  i.e. PR(A) = PR(B)/2 + PR(C)/1 + PR(D)/3
- In general, PR(A) = PR(B)/L(B) + PR(C)/L(C) + PR(D)/L(D)

  i.e. PR(m) = $\sum_{n \in B_m} \frac{PR(n)}{L(n)}$

  L(page) → normalized number of outbound links

  B_m → set of all pages link to page m

AUTHORITIES AND HUBS

- Jon Kleinberg developed an algorithm that made use of the link structure of the web in order to discover and rank pages relevant for particular topics.
- A page is called an authority for the query if it contains the valuable information on the subject.
- For example: for query "car" → www.bmw.com, www.mercedes-benz.com
- Authoritative pages are truly relevant to the given query.
- However there is a second category of pages relevant to the process of finding authoritative pages called hubs.
- Hubs contain useful links towards the authoritative pages, i.e. hubs point the search engine to the right direction.
- Jon Kleinberg's algorithm called HITS identifies good authorities and hubs for a topic by assigning two numbers on a page.
  - Authority weight
  - Hub weight
- The weights are defined recursively
- A higher authority weight occurs if the page is pointed to by pages with high hub weights.

- A higher hub weight occurs if the page points to many pages with high authority weights.
- For a web page (p), h(p) = $\sum_{p \to y} a(y)$, a(p) = $\sum_{y \to p} h(y)$; where m → n denotes the existence of hyperlink from m to n.

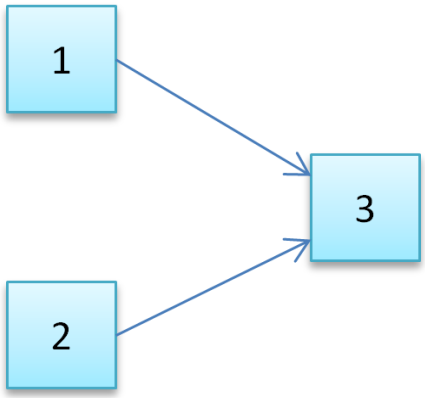
CALCULATION PROCESS

- Find adjacency matrix A, $A_{ij} = \begin{cases} 1, if\ there\ is\ a\ hyperlink\ from\ page\ i\ to\ j \\ 0, otherwise \end{cases}$
  - $a = A^T h$
  - $h = Aa$
  - In general, $a_i = A^T h_{i-1} = A^T A a_{i-1}$

    $h_i = A a_{i-1} = A A^T h_{i-1}$

- Example

$A = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}$ $A^T = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix}$



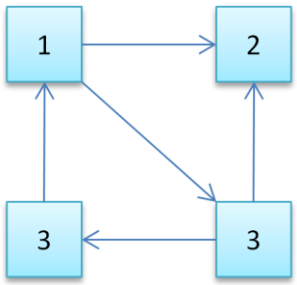
Assume that initially hub vector $h = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

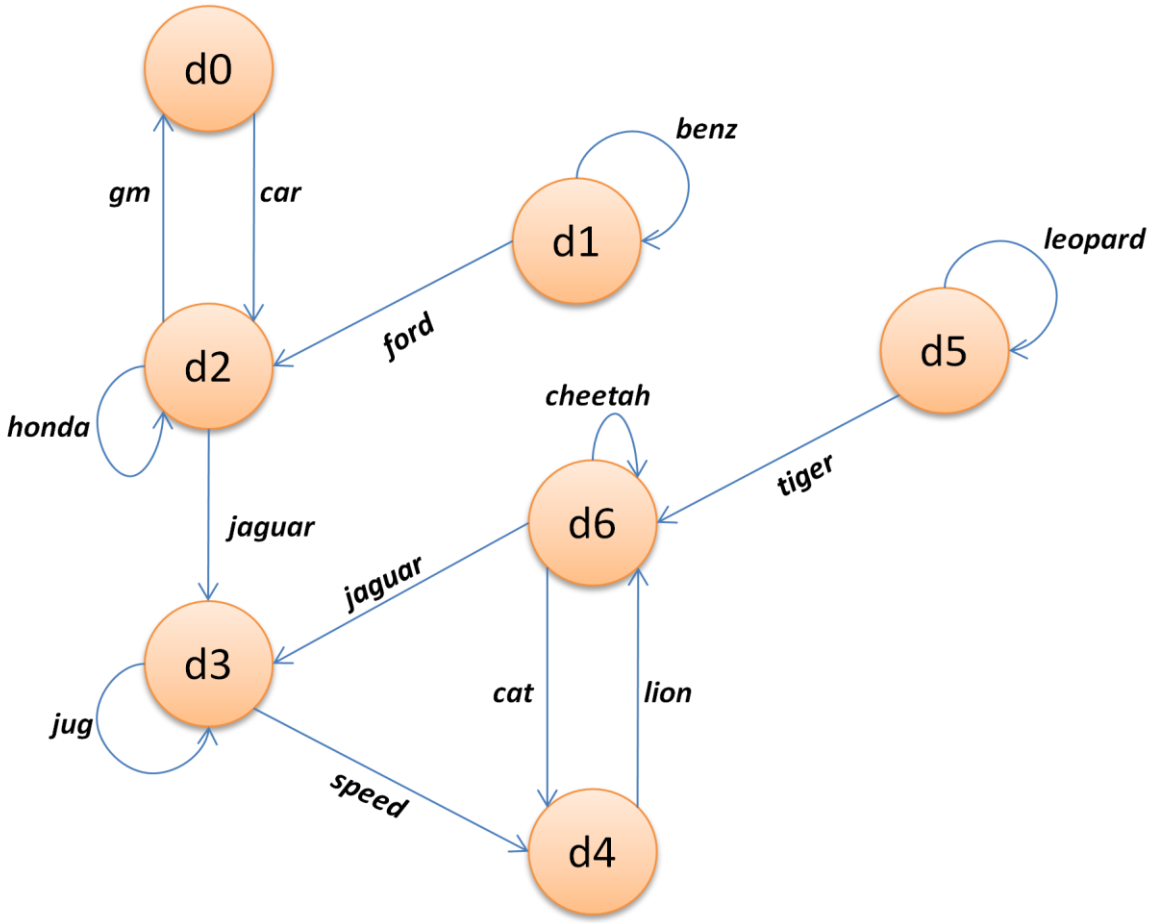
$a = A^T h = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix}$

Updated hub, $h = Aa = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix}$

i.e. node 3 is the most authority weight, since it is only one with incoming edges, and node 1 & 2 are equally important hubs.

Homework :





Assuming the query "jaguar" and double weighting of links whose anchors contain the query word

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 2 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 2 & 1 & 0 & 1 \end{bmatrix}$$

$$\vec{h} = \left(\frac{1}{\lambda h}\right) AA^T \vec{h} \; (\lambda h = AA^T)$$

$$\vec{a} = \left(\frac{1}{\lambda a}\right) AA^T \vec{a} \; (\lambda a = AA^T)$$

$\vec{h}$ = (0.03 0.04 0.33 0.18 0.04 0.35)

$\vec{a}$ = (0.10 0.01 0.12 0.47 0.16 0.01 0.13)

SHOPPING AGENTS

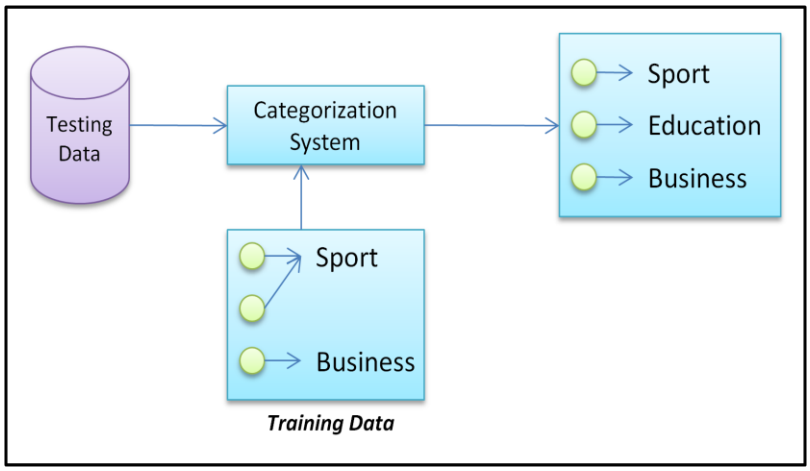
- Some people would be happy if they could find a product on the web at any price.
- Others are bargain shopping and want to find the best price available anywhere on the web.
- Software for comparison shopping are shopping agents, shopping bots, shop bots.
- Not only compare products but keep looking for them over time so that you can be notified as new items that suit your personal tasks becomes available.
- They may also be able to suggest other items that might substitute for or enhance the item you are looking for.
- Example:
  - shopping.yahoo.com → comparison shopping for broad range of products
  - shopping.com → shopping ideas with reviews
  - epinions.com → helps you decide what to buy and where to buy

INTERNET BOT (BOT, WEB ROBOTS)

- Internet bots are software applications that run automated tasks over the internet.
- Largest use of bots is in web spidering in which an automated script fetches and analyzes the information from WWW.
- Each server can have a file called robots.txt containing rules for the spidering of that server that bot is supplied to obey.
- Other examples are chat bot.

TEXT CATEGORIZATION

- Text categorization is a task of automatically sorting a set of documents into categories (classes) from predefined set.
- Classify new document.
- Supervised learning.



APPLICATIONS

1. News article classification
2. Automatic email filtering
3. Web page classification
4. Word sense disambiguity

CATEGORIZATION ALGORITHM

1. Manually –Rule based
2. Automatic (Learning Algorithm)
   a. Rochhi algorithm
   b. Baye's Theorem
   c. Decision Trees
   d. KNN
   e. SVM (Support Vector Machine)

Given,

- A description of an instance x Є X, where X is the instance language or instance space.
- A fixed set of categories C= $\{c_1, c_2, \ldots\ldots, c_n\}$

Determine: The category of x: c(x) Є C where, c(x) is a categorization function whose domain is X and whose range is C.

ROCCHIO ALGORITHM

- Using relevance feedback, i.e. relevance feedback methods can be adopted for text categorization.
- Use TF/IDF weights vectors to represent text document.
- For each category compute a prototype vector by summing the vectors of the training documents in the category.
- Assign test documents to the category with the closest prototype vector based on cosine similarity.

TRAINING ALGORITHM

- Assume the set of categories is $\{c_1, c_2, \ldots\ldots, c_n\}$
- For i = 1 to n
    - let Pi = <0, 0,...., 0> (initial prototype vectors)
- For each training example x, let d be the normalized TF/IDF term vector for document x.
- For all i, $P_i = P_i + d$

TESTING ALGORITHM

- Given test document x
- Let d be the TF/IDF weighted vector for x
- Let m = −2 (initial minimum cosine)
- For i = 1 to n
    - Let s = cossim(d, P_i) → compute similarity to each prototype
    - if (s>m)

      {

      m = s;

      r = c_i; → update with most closest class

      }
- loop
- return class r

Exercise

- Assume the following training set

  Food: "Turky stuffing"

  Food: "Buffalo wings"

  Beverage: "Cream soda"

  Beverage: "Orange soda"

  Apply the Rocchio algorithm to classify a new name "Turky Soda".

BAYESIAN METHODS

- Learning and classification methods based on probability theory.
- Baye's theorem plays a critical role in probabilistic learning and classification.
- Uses prior probability of each category given no information about an item.
- Categorization produces a posterior probability distribution over the possible categories given a description of an item.

BAYE'S THEOREM

- $P(A|B) = \frac{P(A)*P(B \mid A)}{P(B)}$
- Example:

D =

Size	Color	Shape	Category
Small	Red	Circle	Positive
Large	Red	Circle	Positive
Small	Red	Triangle	Negative
Large	blue	Circle	Negative

AFTER TRAINING

Probability	Positive	Negative
P(Y)	0.5	0.5
P(small|Y)	0.5	0.5
P(medium|Y)	0.0	0.0
P(large|Y)	0.5	0.5
P(red|Y)	1.0	0.5
P(blue|Y)	0.0	0.5
P(green|Y)	0.0	0.0
P(square|Y)	0.0	0.5
P(triangle|Y)	0.0	0.5
P(circle|Y)	1.0	0.5

Size<small, medium, large>

Color<red, blue, green>

Shape<circle, triangle, square>

Category<positive, negative>

- Testing Sample X: <medium, red, circle>

- P(pos | X) $= \frac{P(pos) * P(X|pos)}{P(X)}$

  = P(pos) * P(medium | pos) * P(red | pos) * P(circle | pos)

  = 0.5 * 0.001 * 1.0 * 1.0

  = 0.0005

- P(neg | X) $= \frac{P(neg) * P(X|neg)}{P(X)}$

  = P(neg) * P(medium | neg) * P(red | neg) * P(circle | neg)

  = 0.5 * 0.001 * 0.5 * 0.5

  = 0.000125

TRAINING ALGORITHM

- Let v be the vocabulary of all words in D
- For each category Ci Ɛ C
  - Let D_i be the subset of documents in category C_i
  - $P(C_i) = |D_i|/|D|$
  - Let T_i be the concatenation of all documents in d_i
  - Let n_i be the total number of word occurrences in T_i
  - For each word W_j Ɛ V
    - Let n_{ij} be the number of occurrences of W_j in T_i
    - Let $P(W_j|Ci) = (n_{ij} + 1)/(n_i+|V|)$

DECISION TREES

- Decision tree induction is the learning of decision trees from class labeled training tuples.
- A decision tree is a flowchart like tree structure where each internal node denotes a test on an attribute, each branch represents an outcome of the test and each node holds a class label.

- Example:

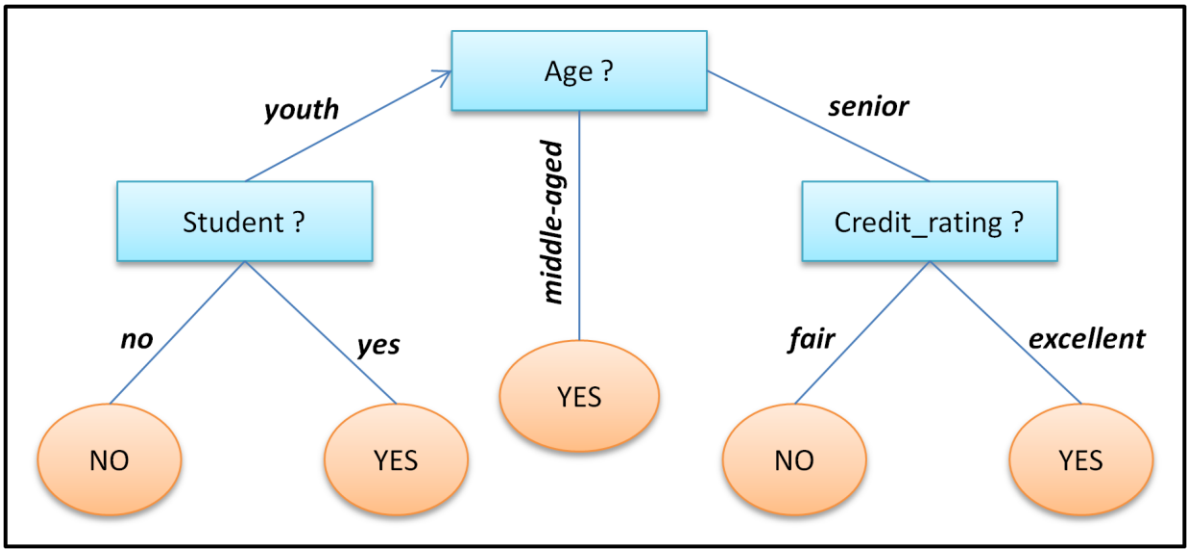


Fig. A decision tree for the concept buys_computer indicating whether a customer is likely to purchase a computer

- Example:

Consider a data with a number of examples for several days with a class "Play Tennis".

Day	Outlook	Temperature	Humidity	Wind	Play Tennis
D1	SUNNY	HOT	HIGH	WEAK	NO
D2	SUNNY	HOT	HIGH	STRONG	NO
D3	OVERCAST	HOT	HIGH	WEAK	YES
D4	RAIN	MILD	HIGH	WEAK	YES
D5	RAIN	COOL	NORMAL	WEAK	YES
D6	RAIN	COOL	NORMAL	STRONG	NO
D7	OVERCAST	COOL	NORMAL	STRONG	YES
D8	SUNNY	MILD	HIGH	WEAK	NO
D9	SUNNY	COOL	NORMAL	WEAK	YES
D10	RAIN	MILD	NORMAL	WEAK	YES

D11	SUNNY	MILD	NORMAL	STRONG	YES
D12	OVERCAST	MILD	HIGH	STRONG	YES
D13	OVERCAST	HOT	NORMAL	WEAK	YES
D14	RAIN	MILD	HIGH	STRONG	NO

Outlook <sunny, overcast, rain>

Temperature <hot, mild, cool>

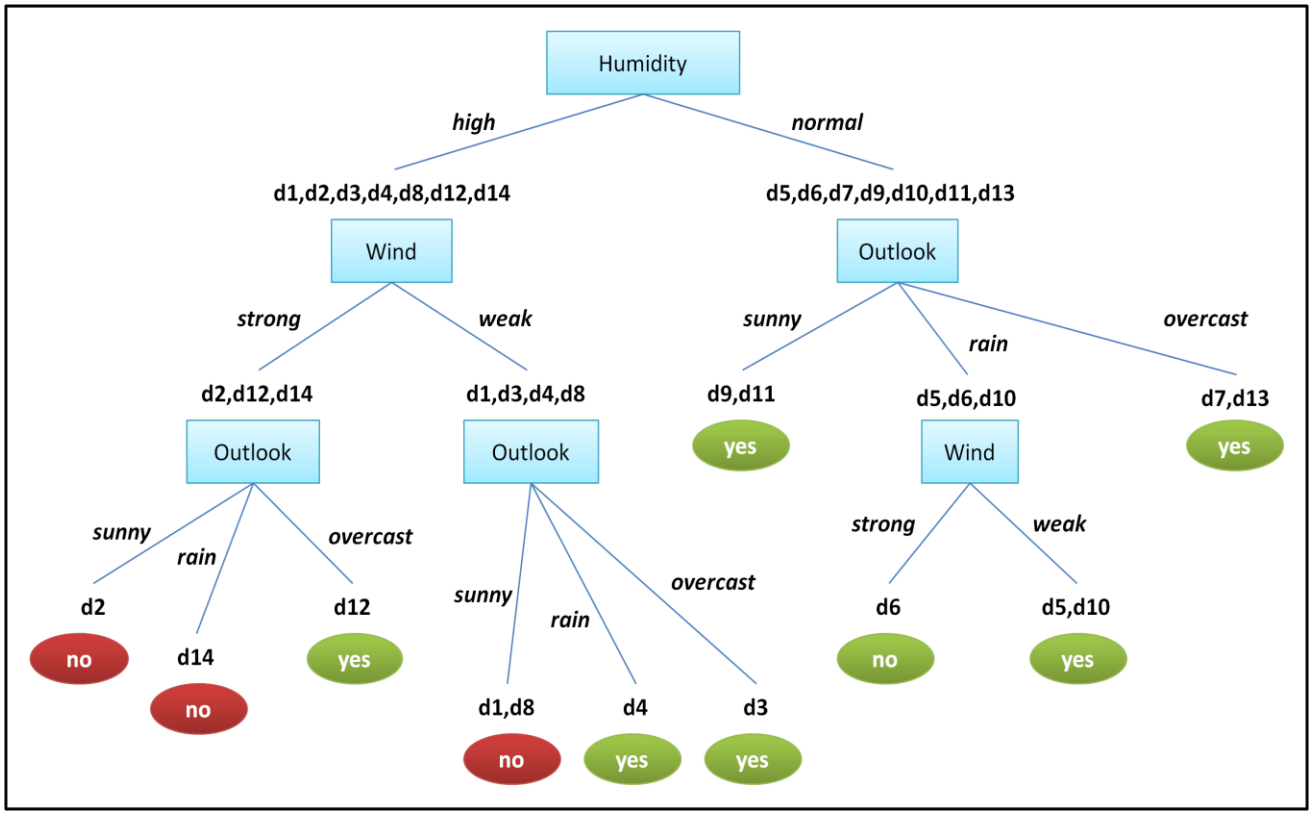
Humidity <high, normal>

Wind <weak, strong>

Play Tennis <no, yes>

Building Decision Tree

- We first need to decide which attribute to make a decision. Let's say we selected "humidity".



Testing : <sunny, hot, normal, weak> → YES

NEAREST NEIGHBOR LEARNING ALGORITHM

- Learning is just storing the representations of the training examples in D.
- Testing instance x.
  - Compute similarity between x and all examples in D.
  - Assign x the category of the most similar example in D.
- Find the K-most similar examples and return the majority category of these K-examples.
- Value of K is typically odd to avoid ties, 3 and 5 are most common.
- Nearest neighbor method depends on similarity (Euclidean distance in m-dimensional instance).
- For text, cosine similarity of TF-IDF weighted vectors is most effective.
- Training Algorithm
  - For each training example <x, c(x)> Ɛ D.

    Compute the corresponding TF-IDF vector dx for document x
- Testing Algorithm
  - For testing instance y
  - Compute TF-IDF vector d for document y
  - For each <x, C(x)>ƐD
    - Let Sx = cossim(d,dx)
  - Sort x, in D by decreasing value of Sx
  - Let N be the first K examples in D
  - Return the majority class of examples in N

Exercise

- Assume the following training set (2 classes)

  Food: “turkey stuffing”

  Food: “buffalo wings”

  Beverage: “cream soda”

  Beverage: “orange soda”

  Apply KNN with K=3 to classify new name “turkey soda”.

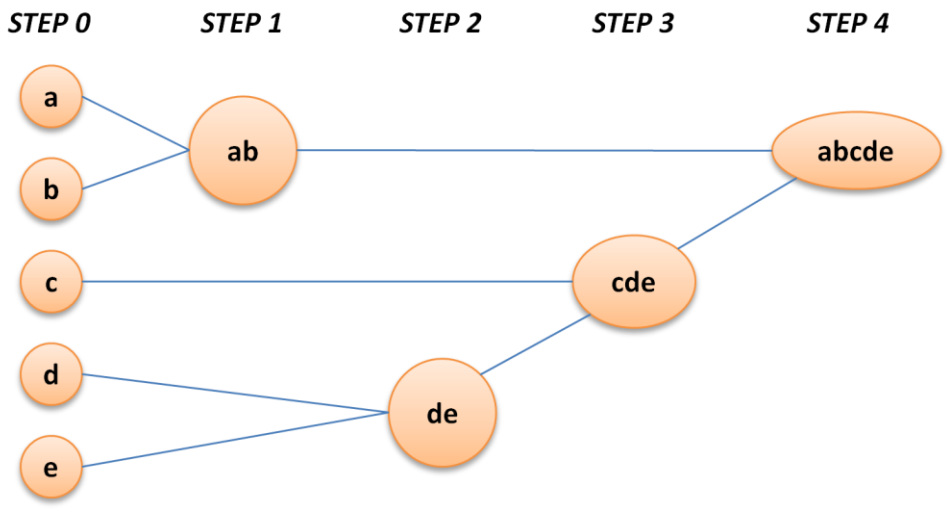
CLUSTERING ALGORITHM

- Clustering algorithm group a set of documents into subsets or clusters.
- To create clusters that is coherent internally but clearly different form each other, i.e. documents within a cluster should be as similar as possible and documents in one cluster should be as dissimilar as possible form documents in other clusters.
- Unsupervised learning i.e. no involvement of human expert who assigned documents into classes.
- Flat clustering creates a set of clusters without any explicit structure.
- Hierarchical clustering creates a hierarchy of clusters.
- Hard clustering assigns each document exactly in one cluster.
- Soft clustering distributes a document over all clusters.

EXPECTATION MAXIMIZATION -SELF STUDY

AGGLOMERATIVE CLUSTERING ALGORITHM

- The algorithm forms clusters in a bottom up manner as follows:
  - Initially put each article in its own cluster.
  - Among all current clusters, pick the two clusters with smallest distance
  - Replace these two clusters with a new cluster formed by merging the two original ones.
  - Repeat the above two steps until there is only one remaining cluster.
- Example:



K-MEANS ALGORITHM

- Each cluster is represented by the centre of the cluster
- Algorithm
  - Choose k number of clusters to be determined.
  - Choose k objects randomly as the initial cluster centers
  - Repeat
    - Assign each object to their closest cluster
    - Compute new clusters, calculate mean points
  - Until
    - No change in cluster entities OR
    - No objects change its clusters.
- Example: Consider the following instances in the table given (2D Form)

Instance	X	Y
1	1.0	1.5
2	1.0	4.5
3	2.0	1.5
4	2.0	3.5
5	3.0	2.5
6	3.0	4.0

1. If the objects are to be partitioned into 2 clusters then k = 2.
2. Next choose two points are random, object 1 and 3 are chosen,
   i.e. C_1 = (1.0, 1.5) and C_2 = (2.0, 1.5)
3. Euclidean distance between i's and j's:

$$D(i - j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$$

INFORMATION FILTERING SYSTEM

- Information filtering system is a system that removes redundant or unwanted information from an information stream using automated or computerized methods.
- A filtering system consists of several tools that help people find the most valuable information so in the limited time, you can dedicate to read/listen/view correctly directional and valuable documents.
- It reduces or eliminates the harmful information
- Application:
  (a) Spam filtering, (b) Censorship and (c) Entrance selection

RECOMMENDER SYSTEMS

- A specific type of information filtering system technique that attempts to recommend information items (movies, TV programs, music, books, research papers, etc) that are likely to be of interest to the user.
- Many online stores provide recommendations (For example: Amazon, CD now, etc).
- Recommenders have been shown to substantially increase sales at online stores.
- Typically a recommender system compares a user profile to some reference characteristics and seeks to predict the rating or preference that a user would give to an item they had yet not considered.
- User's profile can be built using explicit data collection and implicit data collection.
- **Explicit data collection includes:**
    1. Asking a user to rate an item.
    2. Asking a user to rank a collection of items from favorite to least favorite.
    3. Presenting two items to a user and asking him/her to choose the better one of them.
    4. Asking a user to create a list of items that he/she likes.
- **Implicit data collection includes:**
    1. Observing the items that a user views in an online store.
    2. Analyzing viewed items.
    3. Keeping a record of the items that a user purchases online.
    4. Obtaining a list of items that a user has listened or watched on their computer.
    5. Analyzing the user's social network and discovering similar likes and dislikes.
- Recommender systems are useful alternative to search algorithms, since they help users discover items they might not have found by themselves.
- There are two basic approaches to recommending.
- They are:
    - Collaborative Filtering
    - Content Based Recommending

PERSONALIZATION

- Recommenders are instances of personalization software.
- Personalization concerns adapting to the individual needs, interests and preferences of each user.
- It includes:
    1. Recommending
    2. Filtering
    3. Predicting

COLLABORATIVE FILTERING

- Maintain a database of many users' ratings of a variety of items.
- For a given user, find other similar users whose ratings strongly correlate with the current user.
- Recommend items rated highly by the similar users, but not rated by the current user.

- Collaborative filtering methods are based on collecting and analyzing a large amount of information on user's behavior.
- Types:
  - User based collaborative filtering
  - Items to item collaborative filtering

1. User based collaborative filtering

- Look for users who share the same rating patterns with the active user (the user who the prediction is for).
- Use the ratings from those likeminded users found in above step to calculate the prediction for the active user.

2. Items to item collaborative filtering

- People who buy X also buy Y.
- Association rule.

User 6

- Typically Pearson correlation coefficient is used between ratings for active user, 'a' and another user, 'u'.

- $\mathbf{C_{a,u}} = \dfrac{covar(r_a, r_u)}{\sigma_{ra}\sigma_{ru}}$

- r_a and r_u are the rating vectors for the m items rated by both a and u.
- $r_{i,j}$ is users i's rating for item j.

i.e. $$covar(r_a, r_u) = \frac{\sum_{i=1}^{m}(r_{a,i}-\overline{r_a})(r_{u,i}-\overline{r_u})}{m}$$

$$\overline{r_x} = \frac{\sum_{i=1}^{m} r_{x,i}}{m}$$

$$\sigma_{ra} = \sqrt{\frac{\sum_{i=1}^{m}(r_{x,i}-\overline{r}_x)^2}{m}}$$

PROBLEMS WITH COLLABORATIVE FILTERING

1. Cold start:
   There needs to be enough other users already in the system to find a match.
2. Sparsing:
   If there are many items to be recommended, even if there are many users, the user/rating matrix is spare and it is hard to find users that have rated the same items.
3. First rater:
   It cannot recommend on item that has not previously rated.
4. Popularity basis:
   It cannot recommend items to someone with unique taste.

SIGNIFICANCE WEIGHTING

$W_{a,u} = S_{a,u} C_{a,u}$

$S_{a,u}$ = 1, if m>50

m, if m<=50

RATING PREDICTION

$$P_{a,i} = r_a + \frac{\sum_{u=1}^{n} W_{a,u}(r_{u,i} - r_u)}{\sum_{u=1}^{n} W_{a,u}}$$

CONTENT BASED RECOMMENDING

- Recommendations are based on information on the content of items rather than on other user's opinions.
- Content based filtering methods are based on the information about the items that are going to be recommended.
- Try to recommend the items similar to those that a user liked in the past.
- In particular, various candidate items are compared with items previously rated by the user and the best matching items are recommended.
- Some previous applications are:
    1. News weeder (1995)
    2. Syskill and Webert (1996)
- Example of content based recommending: LIBRA (Learning Intelligent Book Recommending Agent).
- **Advantages**
    - No need for data on other users. (No cold start or sparsity problems.)
    - Able to recommend to users with unique tasks.
    - Able to recommend new and unpopular items. (No first rater problem.)
    - Can provide explanations of recommended items by listing content features that caused an item to be recommended.
- **Disadvantages**
    - Requires content that can be encoded as meaningful features.
    - User's taste must be represented as a learnable function of these content features.
    - Unable to exploit quality judgments of other users.