# CS-AD 220 – Spring 2016

# Natural Language Processing

**Session 27: 10-May-16**

Prof. Nizar Habash

Slides from Jurafsky and Martin with modifications

# NYUAD Course CS-AD 220 – Spring 2016

# Natural Language Processing

# Assignment #4

# Phrase-based Statistical Machine Translation

# Assigned Apr 19, 2016

# Due May 10, 2016 (11:59pm)

**Introduction**[1]

In this laboratory exercise, you will build a complete phrase-based statistical machine translation system from small amounts of training data, evaluate their performance, and identify ways that translation quality can be improved. Resulting systems will be evaluated on test data (released a few days before the deadline). You will build the MT system using Moses, an open-source phrase-based statistical machine translation decoder.

*Assignment #4 posted on NYU Classes*

*START EARLY!*

*DEADLINE IS May 10 (11:59pm)*

# This Week and Next Week

- May 12 will include review for final
- **Final is May 16, 1pm to 4pm in CR-002**

# Information Retrieval

Slides from Prof. Jerome White

# What is information retrieval

- Obtaining documents pertinent to a natural language query
  - What are documents?
  - What is pertinent?
- Google’s already done this, right?

# Nature of information is changing

- Big search engines are good generalists
  - *but our retrieval expectations are evolving*

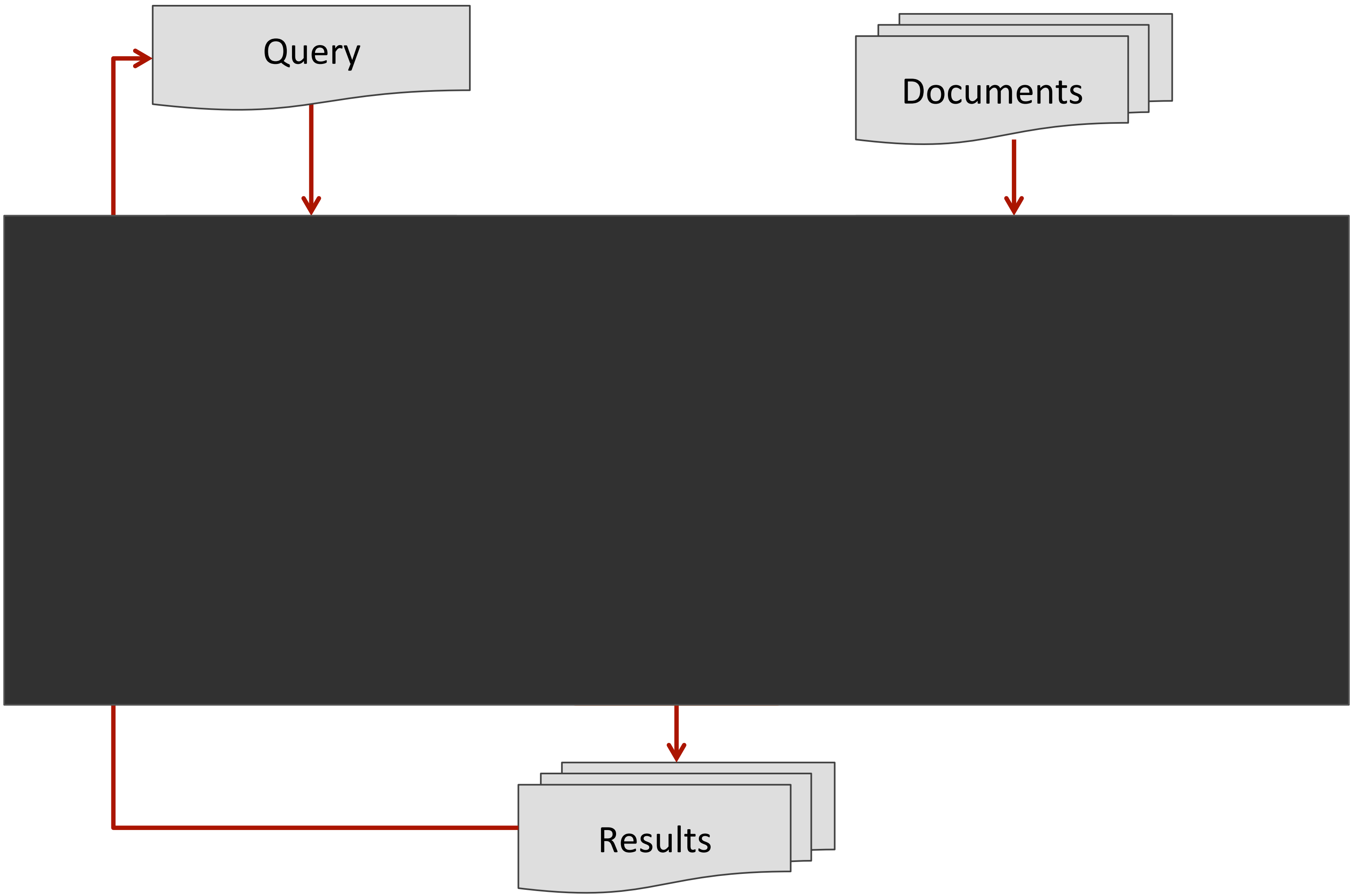
Partially as a result of retrieval!

| | Documents are changing | Queries are changing | Consumption is changing |
|---|---|---|---|
| Then | • Print media<br>• Plain text | • Single words<br>• Simple expressions<br>• Traditional keyboards | • Microfiche<br>• Monitor |
| Now | • Structured text<br>• Images<br>• Audio<br>• Video | • Thoughts<br>• Images<br>• Audio<br>• Swipe<br>• Digital languages | • Voice<br>• Mobile<br>• Wrist |

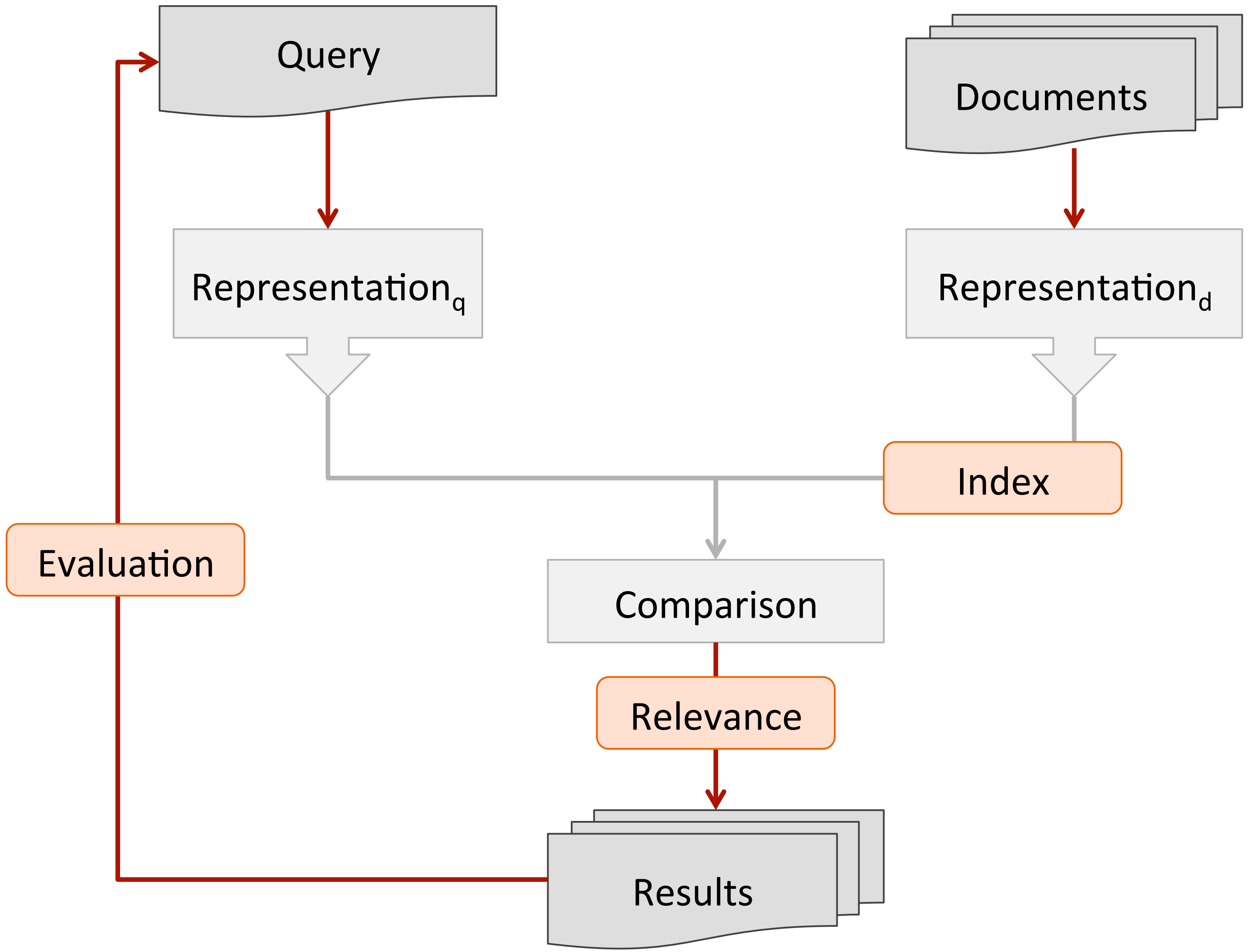
Search engines do this really well

Search engines are *trying* to do this really well

# The IR black box



# The IR black box



# Indexing

- In its most basic form, an index is a mapping between *terms* and *documents*

| Terms | Document ID | | | |
|---|---|---|---|---|
| brutus | 3 | 4 | 7 | 10 |
| caesar | 1 | 4 | 10 | 12 |
| capitol | 4 | 12 | | |
| told | 8 | 16 | 17 | 20 |
| you | 4 | 5 | 11 | |

# Common transformations

- To build this structure, we extract information from text
- To improve retrieve-ability, put the text through various transformations
  - Ensure this is done for the queries, as well!

| Concept | Idea | Example |
| --- | --- | --- |
| Tokenization | Cut character sequence into word tokens | "John's", a state-of-the art solution |
| Normalization | Map text and query term to same form | U.S.A. should match USA |
| Stemming | Different forms of a root to match | authorize, authorization |
| Stop words | Omit very common words | the, a, to, of |

# Indexing appearance

- Build a mapping between the terms, and the documents in which they were found

| Term | Document | | | | |
|---|---|---|---|---|---|
| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello |
| Antony | 1 | 1 | 0 | 0 | 0 |
| Brutus | 1 | 1 | 0 | 1 | 0 |
| Caesar | 1 | 1 | 0 | 1 | 1 |
| Calpurnia | 0 | 1 | 0 | 0 | 0 |
| Cleopatra | 1 | 0 | 0 | 0 | 0 |
| mercy | 1 | 0 | 1 | 1 | 1 |
| worser | 1 | 0 | 1 | 1 | 0 |

# Indexing appearance

- A "query" is a Boolean expression over vectors:

```
Caesar AND Brutus, but NOT Antony
11011 & 11010 & 00111
```

| Term | Document | | | | |
|---|---|---|---|---|---|
| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello |
| Antony | 1 | 1 | 0 | 0 | 0 |
| Brutus | 1 | 1 | 0 | 1 | 0 |
| Caesar | 1 | 1 | 0 | 1 | 1 |
| Calpurnia | 0 | 1 | 0 | 0 | 0 |
| Cleopatra | 1 | 0 | 0 | 0 | 0 |
| mercy | 1 | 0 | 1 | 1 | 1 |
| worser | 1 | 0 | 1 | 1 | 0 |

# But it’s still Boolean retrieval

- Boolean retrieval is good
  - Accurate, if you know the right strategies
  - Efficient for the computer
- Boolean retrieval has its drawbacks
  - Often results in too many documents, or none
  - Users must know Boolean logic
  - Words can have many meanings
  - Choosing the right word can be difficult

# Ideally

**What we'd like**

- A method of retrieval that's closer to how we think
  - Free(er) form queries
  - Results that are estimates about our preference

**What we need**

- Term-document mapping
- Measure of similarity
- Measure of importance

# Indexing counts

- Instead of appearance, entries now count the number of occurrences of a term in a document
- Each document is thus a count vector

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

# Is this enough?

- Can we differentiate documents based on counts?
- Is “Antony and Cleopatra” the best result for “Caesar”?

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

# Important concepts

- ✓ Does the term occur in the document?
- ✓ How many times does the term occur in the document?
- How important is the word to the document?

# The intuition

- *Terms* tell us about documents
  - If “rabbit” appears a lot, it may be about rabbits
- *Documents* tell us about terms
  - “White” is in every document – not discriminating
  - “Habash” is rare – it might be special
- Documents are most likely described well by *rare* terms that occur in them *frequently*
  - Higher “term frequency” is stronger evidence
  - Low “document frequency” makes it stronger still

# The idea

- High term frequency (TF) is evidence of meaning
- Low document frequency (DF) is evidence of term importance
  - Equivalently, high inverse document frequency (IDF)
- Term weight is the product of the two (TF-IDF)
- Add up the weights for each query/document vector

# Document frequency

- "white" appears everywhere – it's probably not that important

| Term | DF | IDF | |
|---|---|---|---|
| habash | 1 | 1,000,000 | |
| animal | 100 | 100,000 | |
| sunday | 1,000 | 10,000 | |
| fly | 10,000 | 1,000 | |
| under | 100,000 | 100 | |
| white | 1,000,000 | 1 | |

# TF-IDF weighting

$$w_{t,d} = tf_{t,d} \times \log \frac{N}{df_t}$$

"Weight" of term *t* within document *d*

How often term *t* appears in document *d*

Number of documents in which term *t* appears, normalized, inverted, and scaled

# Why the log?

$$w_{t,d} = tf_{t,d} \times \log \frac{N}{df_t}$$

| Term | DF | IDF | log(IDF) |
|---|---|---|---|
| habash | 1 | 1,000,000 | 13.82 |
| animal | 100 | 100,000 | 11.51 |
| sunday | 1,000 | 10,000 | 9.21 |
| fly | 10,000 | 1,000 | 6.91 |
| under | 100,000 | 100 | 4.61 |
| white | 1,000,000 | 1 | 0.00 |

"dampens" the effect

# TF-IDF as weights

- Each document is now a real-valued vector of TF-IDF weights

| Term | Document | | | | |
|---|---|---|---|---|---|
| | Antony and Cleopatra | Julius Caesar | The Tempest | Hamlet | Othello |
| Antony | 5.25 | 3.18 | 0.00 | 0.00 | 0.00 |
| Brutus | 1.21 | 6.10 | 0.00 | 1.00 | 0.00 |
| Caesar | 8.59 | 2.54 | 0.00 | 1.51 | 0.25 |
| Calpurnia | 0.00 | 1.54 | 0.00 | 0.00 | 0.00 |
| Cleopatra | 2.95 | 0.00 | 0.00 | 0.00 | 0.00 |
| mercy | 1.51 | 0.00 | 1.90 | 0.12 | 5.25 |
| worser | 1.37 | 0.00 | 0.11 | 4.15 | 0.25 |

# On to similarity

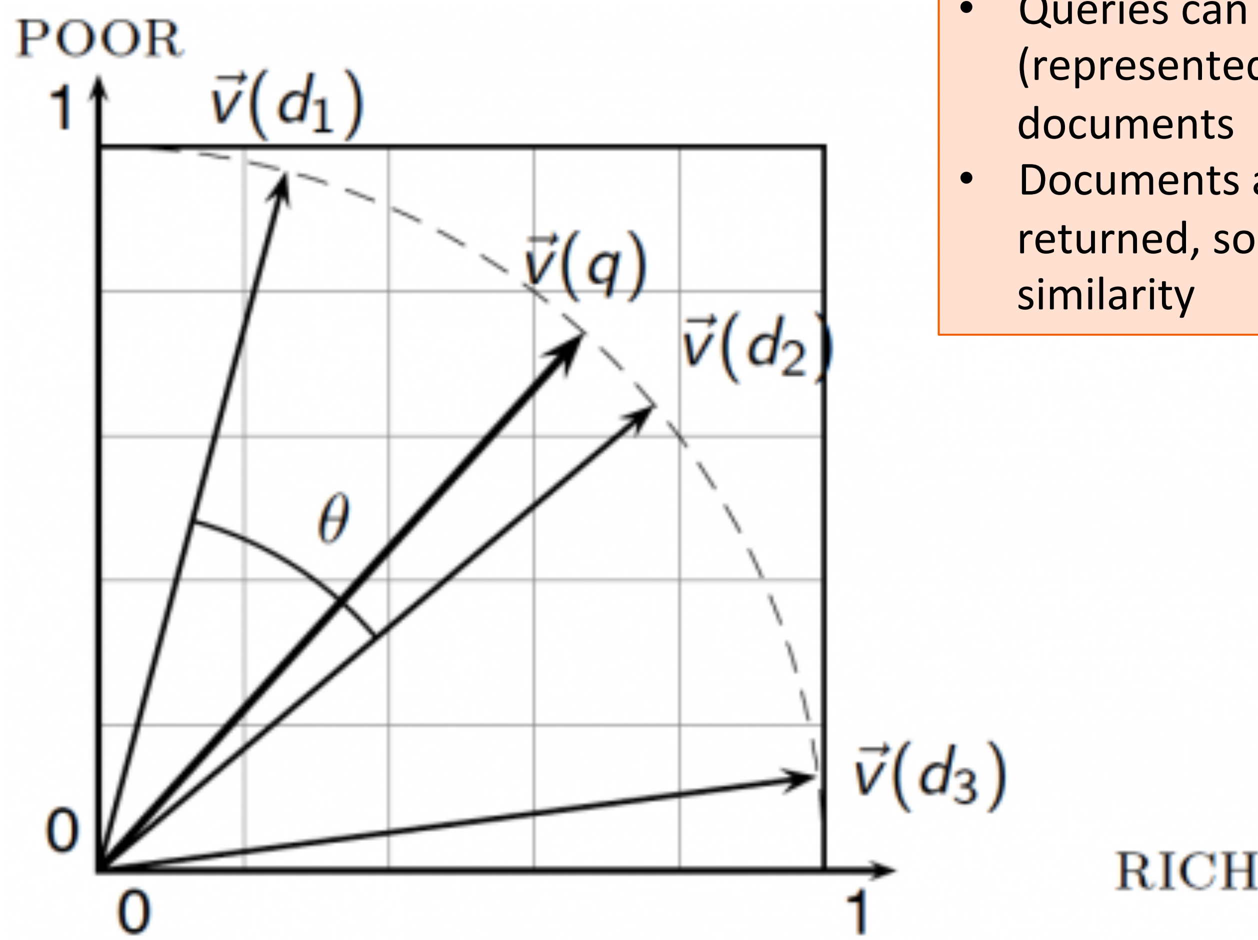
- Now that we have a measure of importance, we need a measure of similarity
- Intuition
  - Each document is a vector
  - Are there measures of vector similarity?

# Cosine similarity

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

- Cosine similarity is a measure of similarity between two vectors
- Ranges
  - Positive 1: A and B are exactly the same
  - Negative 1: A and B are exactly opposite
- In IR, the range is 0 to 1 since TF-IDF is always positive

# Treating queries like documents




- Queries can be treated (represented) like documents
- Documents are then returned, sorted by similarity

# And now… the results!

- We now have
  - A way of indexing documents
  - A structure for treating queries like documents
  - A scoring function for comparing the two
- How do we know if what's returned is any good?
- Can we alter the query to do better?

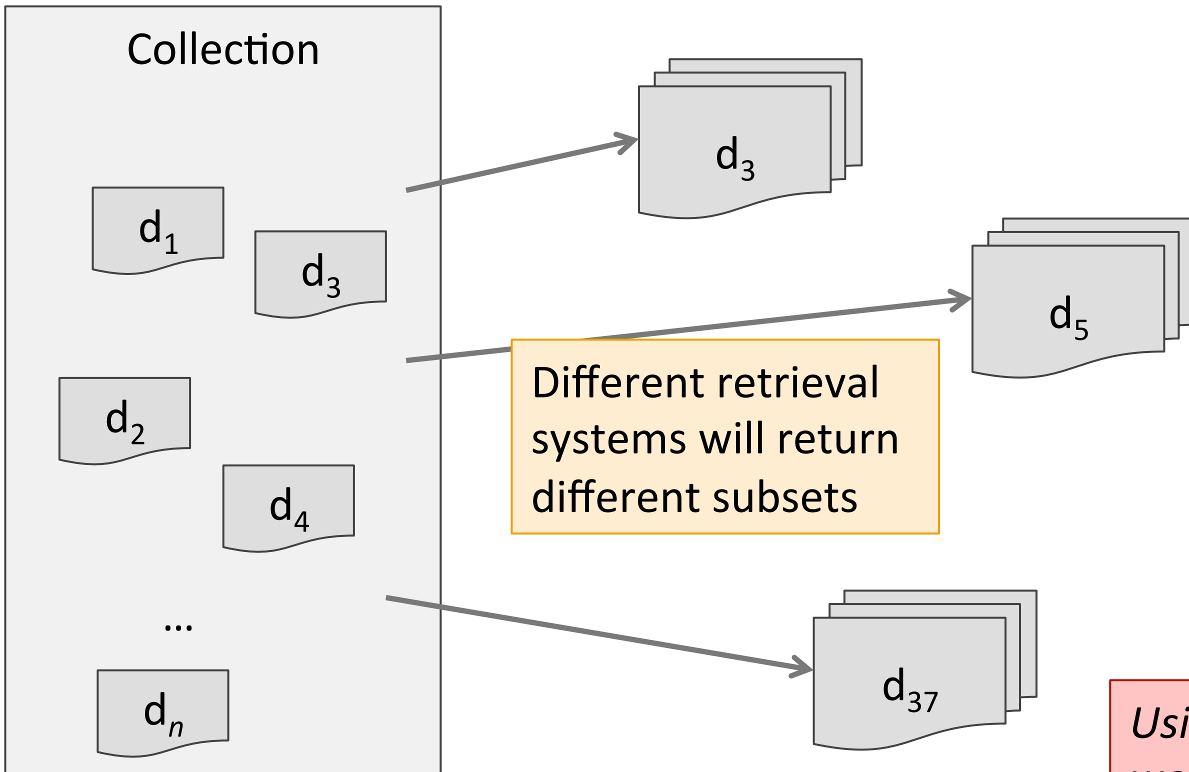
# Relevance

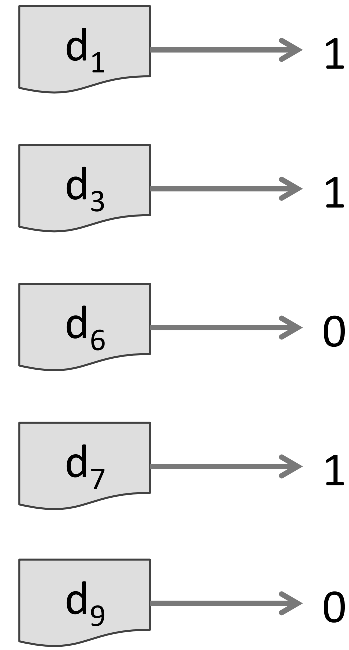
- TF-IDF is nice, but true relevance is subjective
- We can get closer by asking humans to judge
  - Mark documents within a collection as relevant or not relevant to a given *topic* (not a query!)
- New systems then have a way of estimating whether a query was effective at satisfying an information need

# Judging relevance

**Topic**: Information on whether drinking red wine is more effective at reducing your risk of heart attacks than drinking white wine

Human judges determine which documents are relevant to the topic




*Using these relevance judgments, we can now evaluate new queries that are inline with the topic*

# Precision and recall

**Precision**

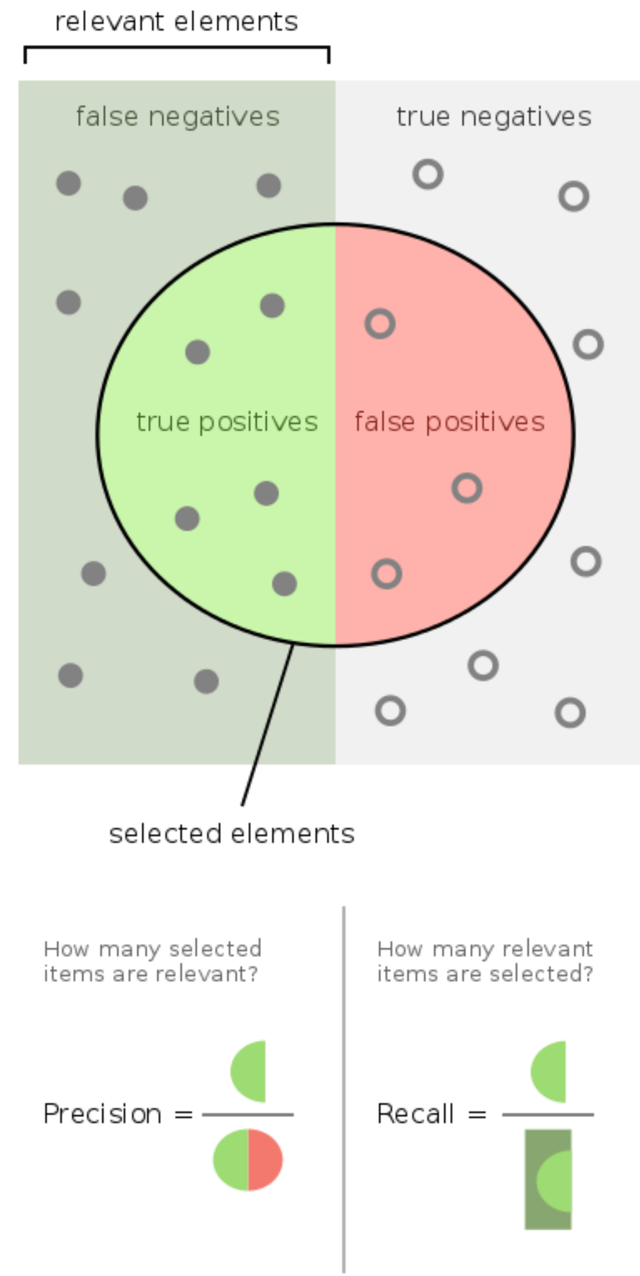
- The fraction of retrieved documents that are relevant
- How many selected documents are relevant?

**Recall**

- Fraction of relevant instances that are retrieved
- How many relevant documents are selected?

# Foundation

- Precision/recall are actively used to describe system performance
- Form the basis of many IR evaluation metrics
  - "This is a recall-based metric"
- Ranked retrieval isn't necessarily required



# Cutoffs

- It is common to quote precision/recall metrics at certain *cutoffs*
- Rather than the entire set of retrieved documents, just consider a top-subset
  – Precision at *n*: score assuming *n* was the total number of documents retrieved
- Cutoffs make the metric more meaningful
  – Most users don't consider all retrieved documents
  – At some point, recall will always be 1

# Nature of the beast

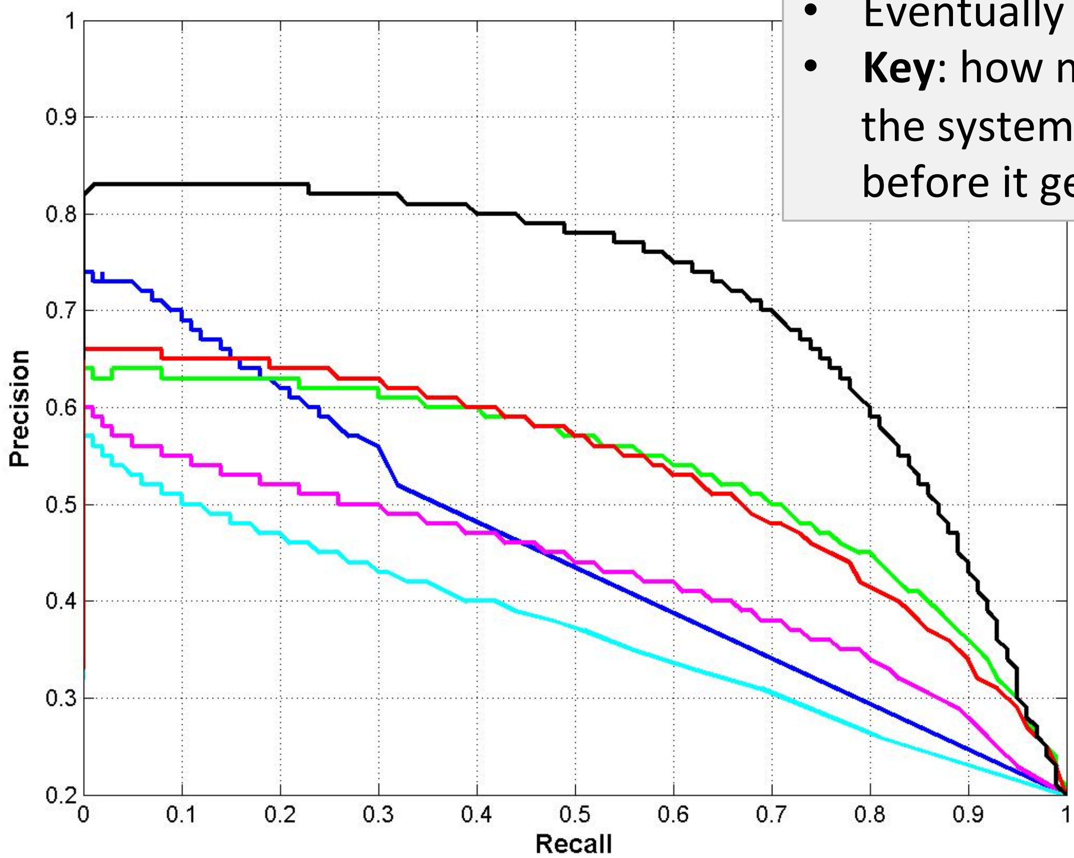
| Rank | Judgment | $Precision_{Rank}$ | $Recall_{Rank}$ |
|---|---|---|---|
| 1 | R | 1.0 | .11 |
| 2 | N | .50 | .11 |
| 3 | R | .66 | .22 |
| 4 | N | .50 | .22 |
| 5 | R | .60 | .33 |
| 6 | R | .66 | .44 |
| 7 | N | .57 | .44 |
| 8 | R | .63 | .55 |
| 9 | N | .55 | .55 |
| 10 | N | .50 | .55 |
| 11 | R | .55 | .66 |
| 12 | N | .50 | .66 |
| 13 | N | .46 | .66 |
| 14 | N | .43 | .66 |
| 15 | R | .47 | .77 |
| 16 | N | .44 | .77 |
| 17 | N | .44 | .77 |
| 18 | R | .44 | .88 |
| 19 | N | .42 | .88 |
| 20 | N | .40 | .88 |
| 21 | N | .38 | .88 |
| 22 | N | .36 | .88 |
| 23 | N | .35 | .88 |
| 24 | N | .33 | .88 |
| 25 | R | .36 | 1.0 |

**Figure 23.4** Rank-specific precision and recall values calculated as we proceed down through a set of ranked documents.

Precision moves away from 1 as rank increases

Recalls moves toward 1 as rank increases

# Precision recall curve




- Eventually recall will hit 1
- **Key**: how much area can the system maintain before it gets there?

# Course Evaluation Today!

- Please log into Albert, www.albert.nyu.edu
  - Go to the "My Class Schedule" tab.
  - You will see a list of all the courses in which you are enrolled.
  - Select the Evaluate link for this class.
- When you have completed the evaluation, be sure to click the Submit button.
- You are expected to complete the evaluation today in class.
  - However, if you are unable to finish in time, you may complete the process later by logging back into Albert. The final deadline to complete the final evaluations is Saturday, May 14.
- You have 10 minutes to complete the assessment.

# Next Time

- May 12$^{th}$ →
  - J+M Chap 23
  - Come with questions about final!