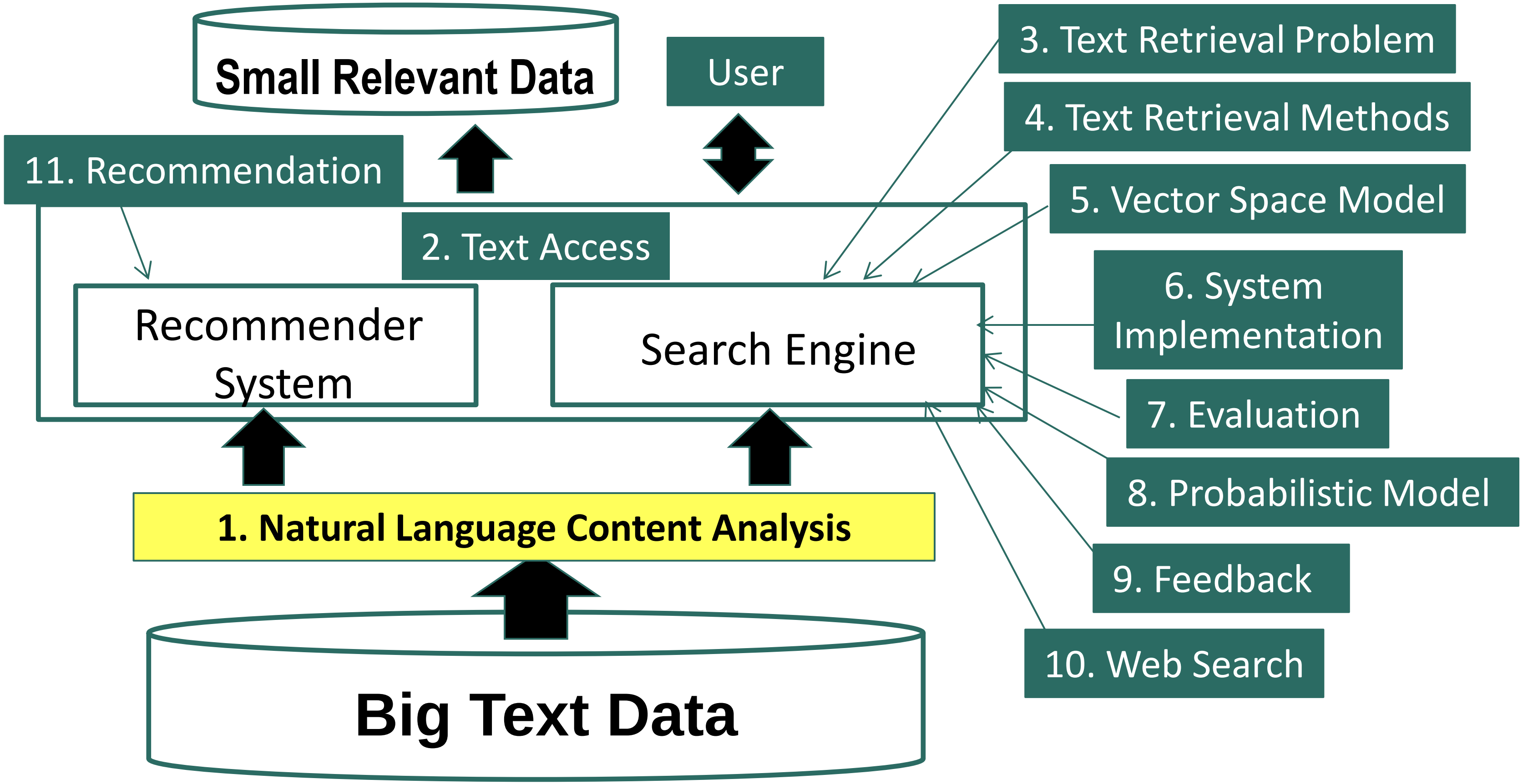
Text Retrieval and Search Engines

Natural Language Content Analysis

ChengXiang "Cheng" Zhai
Department of Computer Science
University of Illinois at Urbana-Champaign

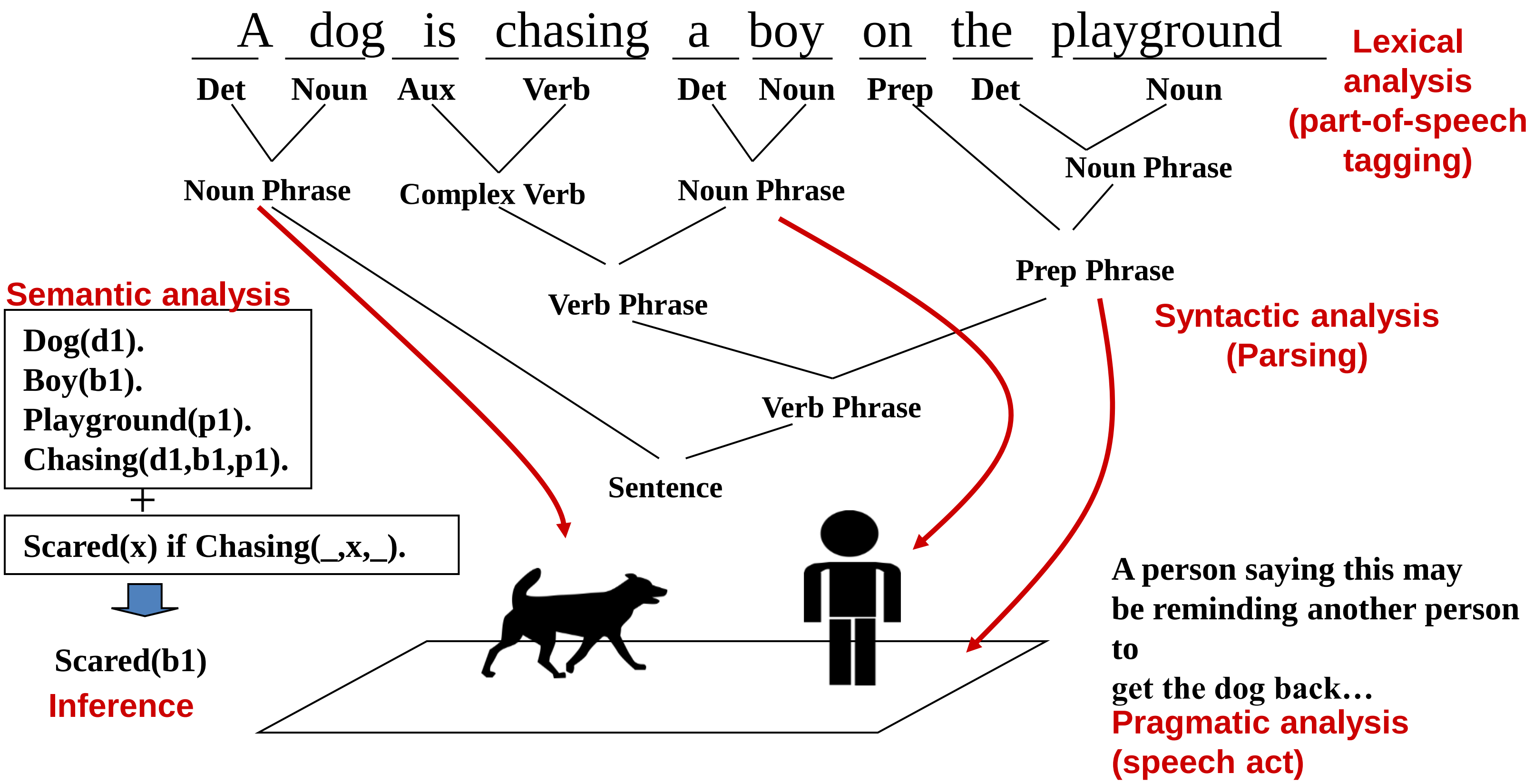
Course Schedule



1. Overview

- What is Natural Language Processing (NLP)?
- State of the Art in NLP
- NLP for Text Retrieval

An Example of NLP



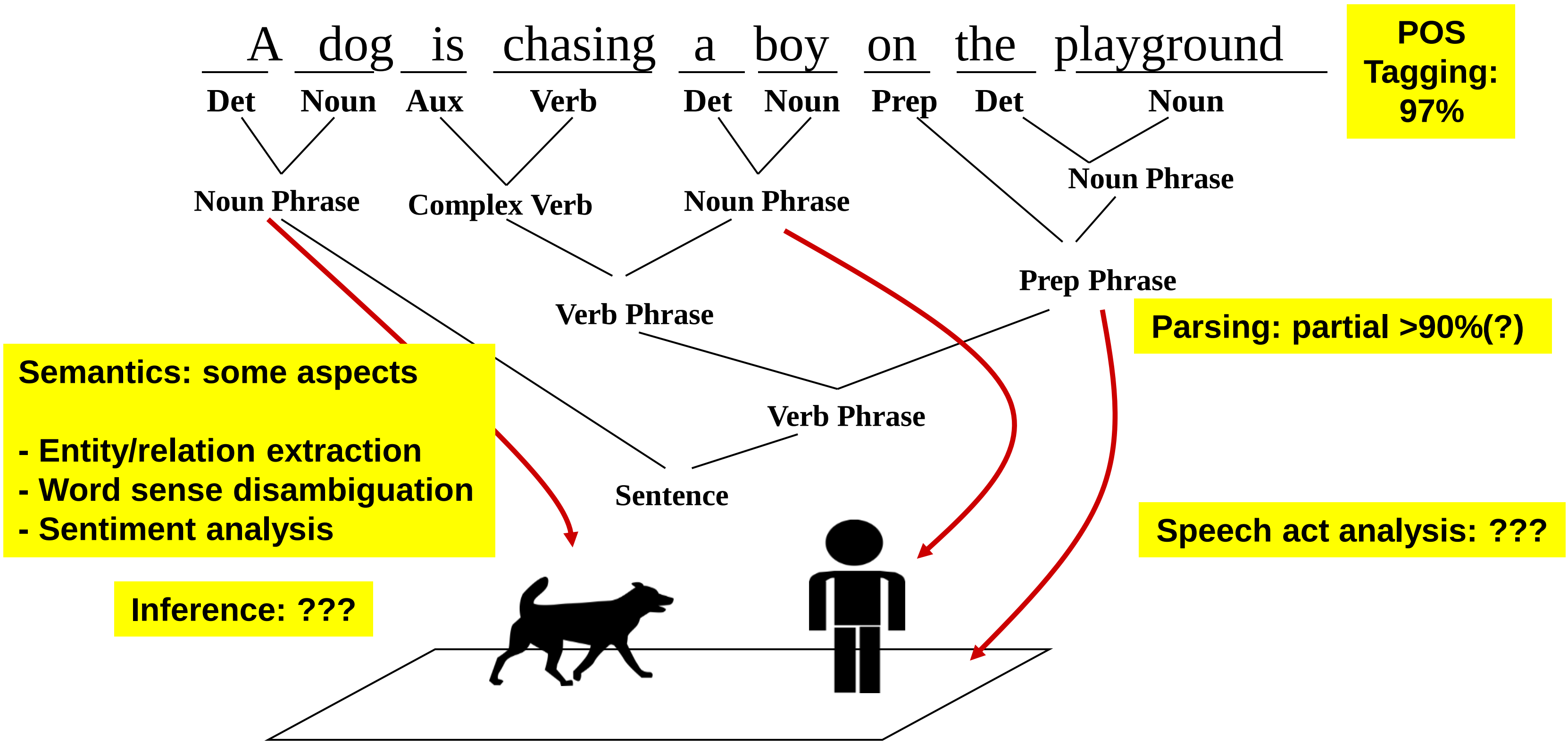
NLP Is Difficult!

- Natural language is designed to make human communication efficient. As a result,
  - we omit a lot of “common sense” knowledge, which we assume the hearer/reader possesses
  - we keep a lot of ambiguities, which we assume the hearer/reader knows how to resolve
- This makes EVERY step in NLP hard
  - Ambiguity is a “killer”!
  - Common sense reasoning is pre-required

Examples of Challenges

- Word-level ambiguity: E.g.,
  - "design" can be a noun or a verb (Ambiguous POS)
  - "root" has multiple meanings (Ambiguous sense)
- Syntactic ambiguity: E.g.,
  - "natural language processing" (Modification)
  - "A man saw a boy *<u>with a telescope</u>*." (PP Attachment)
- Anaphora resolution: "John persuaded Bill to buy a TV for *<u>himself</u>*." (himself = John or Bill?)
- Presupposition: "He has quit smoking." implies that he smoked before.

The State of the Art



What We Can't Do

- 100% POS tagging
  - "He turned off the highway." vs "He turned off the fan."
- General complete parsing
  - "A man saw a boy with a telescope."
- Precise deep semantic analysis
  - Will we ever be able to precisely define the meaning of "own" in "John owns a restaurant."?

Robust & general NLP tends to be "shallow" while "deep" understanding doesn't scale up

NLP for Text Retrieval

- Must be general robust & efficient → Shallow NLP
- **"Bag of words" representation tends to be sufficient** for most search tasks (but not all!)
- Some text retrieval techniques can naturally address NLP problems
- However, deeper NLP is needed for complex search tasks

Additional Reading

Chris Manning and Hinrich Schütze, Foundations of Statistical Natural Language Processing, MIT Press. Cambridge, MA: May 1999.

2. Access to Relevant Text Data

How can a text information system help users get access to the relevant text data?

- Push vs. Pull
- Querying vs. Browsing

Two Modes of Text Access: Pull vs. Push

- **Pull** Mode (**search engines**)
  – Users take initiative
  – Ad hoc information need
- **Push** Mode (**recommender systems**)
  – Systems take initiative
  – Stable information need or system has good knowledge about a user's need

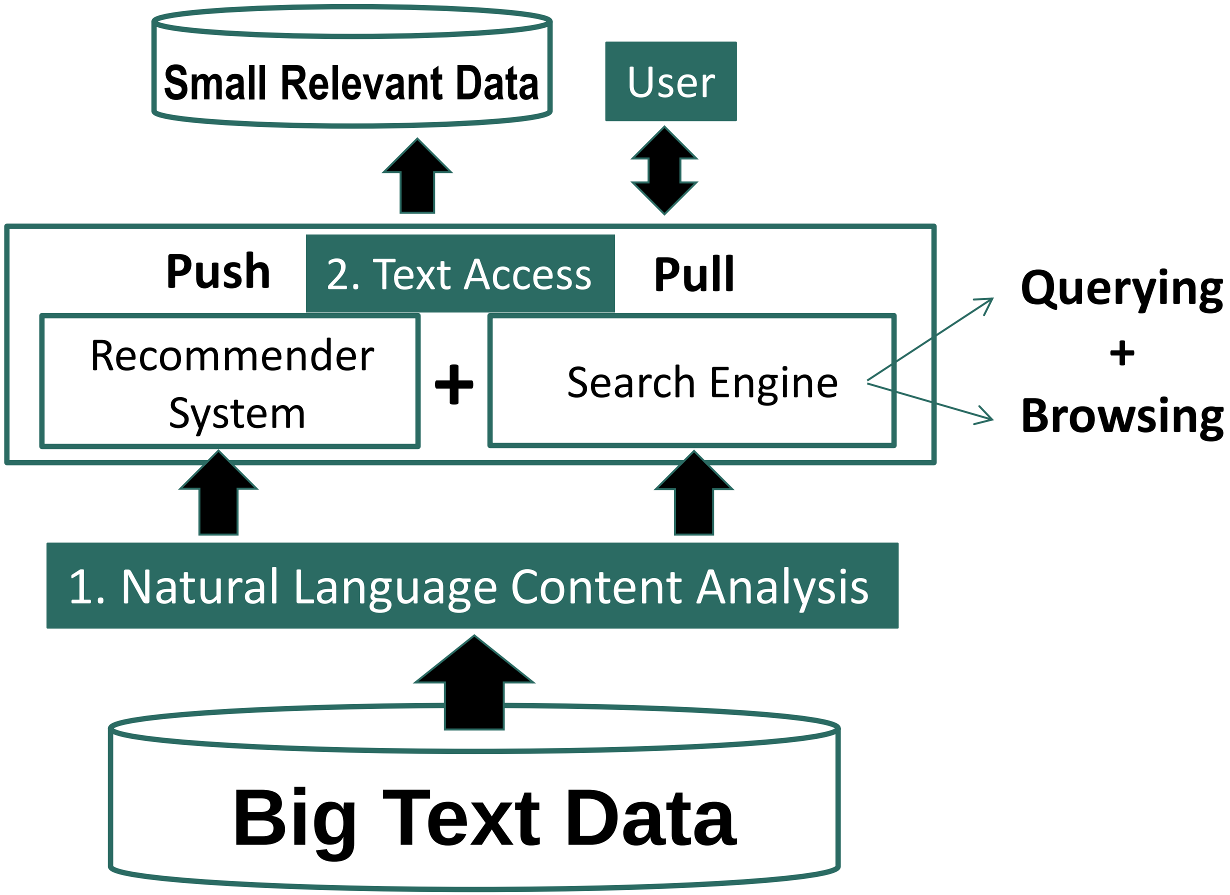
Pull Mode: Querying vs. Browsing

- Querying
  - User enters a (keyword) query
  - System returns relevant documents
  - Works well when the user knows what keywords to use
- Browsing
  - User navigates into relevant information by following a path enabled by the structures on the documents
  - Works well when the user wants to explore information, doesn't know what keywords to use, or can't conveniently enter a query

Information Seeking as Sightseeing

- Sightseeing: Know address of an attraction?
  - Yes: take a taxi and go directly to the site
  - No: walk around or take a taxi to a nearby place then walk
- Information seeking: Know exactly what you want to find?
  - Yes: use the right keywords as a query and find the information directly
  - No: browse the information space or start with a rough query and then browse

Summary



Small Relevant Data
User
Push
2. Text Access
Pull
Recommender System
+
Search Engine
Querying
+
Browsing
1. Natural Language Content Analysis
Big Text Data

Additional Reading

N. J. Belkin and W. B. Croft. 1992. Information filtering and information retrieval: two sides of the same coin?. *Commun. ACM* 35, 12 (Dec. 1992), 29-38.

3. Text Retrieval Overview

- What is Text Retrieval?
- Text Retrieval vs. Database Retrieval
- Document Selection vs. Document Ranking

What Is Text Retrieval (TR)?

- Collection of text documents exists
- User gives a query to express the information need
- Search engine system returns relevant documents to users
- Often called “information retrieval” (IR), but IR is actually much broader
- Known as “search technology” in industry

TR vs. Database Retrieval

- Information
  - Unstructured/free text vs. structured data
  - Ambiguous vs. well-defined semantics
- Query
  - Ambiguous vs. well-defined semantics
  - Incomplete vs. complete specification
- Answers
  - Relevant documents vs. matched records
- TR is an empirically defined problem
  - Can’t mathematically prove one method is better than another
  - Must rely on **empirical evaluation** involving users!

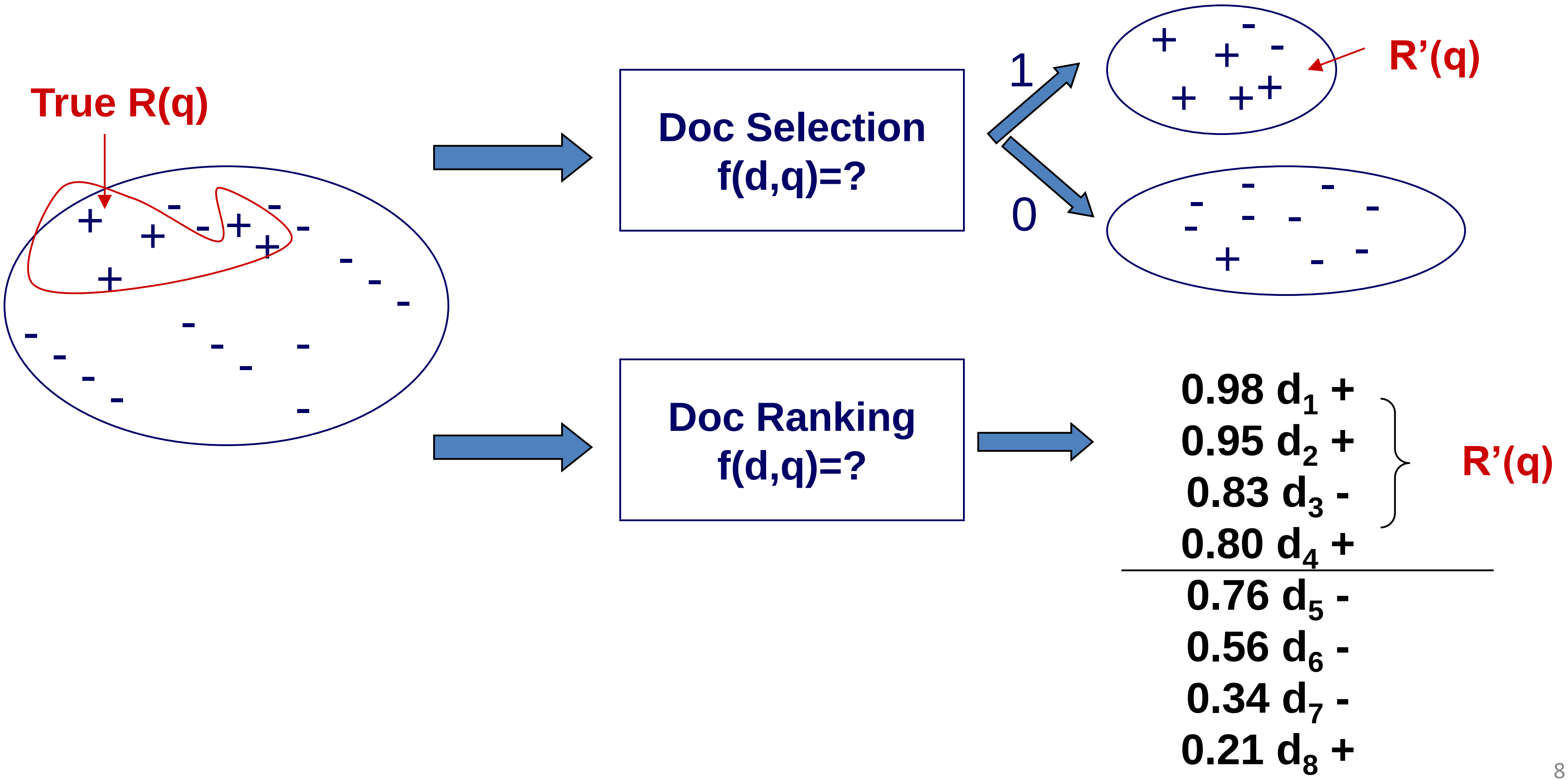
Formal Formulation of TR

- **Vocabulary**: $V=\{w_1, w_2, \ldots, w_N\}$ of language
- **Query**: $q = q_1,\ldots,q_{m,}$ where $q_i \in V$
- **Document**: $d_i = d_{i1},\ldots,d_{im_i,}$ where $d_{ij} \in V$
- **Collection**: $C= \{d_1, \ldots, d_M\}$
- **Set of relevant documents**: $R(q) \subseteq C$
  - Generally unknown and user-dependent
  - Query is a "hint" on which doc is in $R(q)$
- **Task** = compute $R'(q)$, an approximation of $R(q)$

How to Compute R'(q)

- Strategy 1: Document selection
  - $R'(q)=\{d\in C|f(d,q)=1\}$, where $f(d,q) \in\{0,1\}$ is an indicator function or binary classifier
  - System must decide if a doc is relevant or not (**absolute relevance**)
- Strategy 2: Document ranking
  - $R'(q) = \{d\in C|f(d,q)>\theta\}$, where $f(d,q) \in\Re$ is a relevance measure function; θ is a cutoff determined by the user
  - System only needs to decide if one doc is more likely relevant than another (**relative relevance**)

Document Selection vs. Ranking



Problems of Document Selection

- The classifier is unlikely accurate
  - “Over-constrained” query → no relevant documents to return
  - “Under-constrained” query → over delivery
  - Hard to find the right position between these two extremes
- Even if it is accurate, all relevant documents are not equally relevant (relevance is a matter of degree!)
  - Prioritization is needed
- Thus, ranking is generally preferred

Theoretical Justification for Ranking

- **Probability Ranking Principle** [Robertson 77]: Returning a ranked list of documents in descending order of probability that a document is relevant to the query is the optimal strategy under the following two assumptions:
  – The utility of a document (to a user) is **independent** of the utility of any other document
  – A user would browse the results **sequentially**
- Do these two assumptions hold?

Summary

- Text retrieval is an empirically defined problem
  - Which algorithm is better must be judged by users
- Document ranking is generally preferred to
  - Help users prioritize examination of search results
  - Bypass the difficulty in determining absolute relevance (users help decide the cutoff on the ranked list)
- Main challenge: design an effective ranking function **f(q,d) =?**

Additional Readings

- S.E. Robertson, The probability ranking principle in IR. *Journal of Documentation* **33**, 294-304, 1977
- C. J. van Rijsbergen, Information Retrieval, 2nd Edition, Butterworth-Heinemann, Newton, MA, USA, 1979
  - A must-read for anyone doing research in information retrieval. Chapter 6 has an in-depth discussion of PRP.

4. TR Methods.
How to Design a Ranking Function

- **Query:** $q = q_1,\ldots,q_{m,}$ where $q_i \in V$
- **Document:** $d = d_1,\ldots,d_{n,}$ where $d_i \in V$
- **Ranking function:** $f(q, d) \in \Re$
- A good ranking function should rank relevant documents on top of non-relevant ones
- Key challenge: how to measure the likelihood that document d is relevant to query q
- **Retrieval model** = formalization of relevance (give a computational definition of relevance)

Many Different Retrieval Models

- **Similarity-based models**: f(q,d) = similarity(q,d)
  – Vector space model
- **Probabilistic models**: f(d,q) = p(R=1|d,q), where R $\in\{0,1\}$
  – Classic probabilistic model
  – Language model
  – Divergence-from-randomness model
- **Probabilistic inference model**: f(q,d) = p(d➔q)
- **Axiomatic model**: f(q,d) must satisfy a set of constraints
- These different models tend to result in similar ranking functions involving similar variables

Common Ideas in State of the Art Retrieval Models

f(q="presidential campaign news", $\boxed{d}$)

"Bag of Words"

g("presidential", $\boxed{d}$)   g("campaign", $\boxed{d}$)   g("news", $\boxed{d}$)

How many times does "presidential" occur in d?

Term Frequency (TF): c("presidential", d)

How long is d? **Document length**: |d|

How often do we see "presidential" in the entire collection?

Document Frequency: df("presidential")

P("presidential"|collection)

Which Model Works the Best?

- When optimized, the following models tend to perform equally well [Fang et al. 11]:
  - **Pivoted length normalization**
  - **BM25**
  - **Query likelihood**
  - **PL2**
- BM25 is most popular

Summary

- Design of ranking function f(q,d) pre-requires a computational definition of relevance (retrieval model)
- Many models are equally effective with no single winner
- State of the art ranking functions tend to rely on
  - Bag of words representation
  - Term Frequency (TF) and Document Frequency (DF) of words
  - Document length

Additional Readings

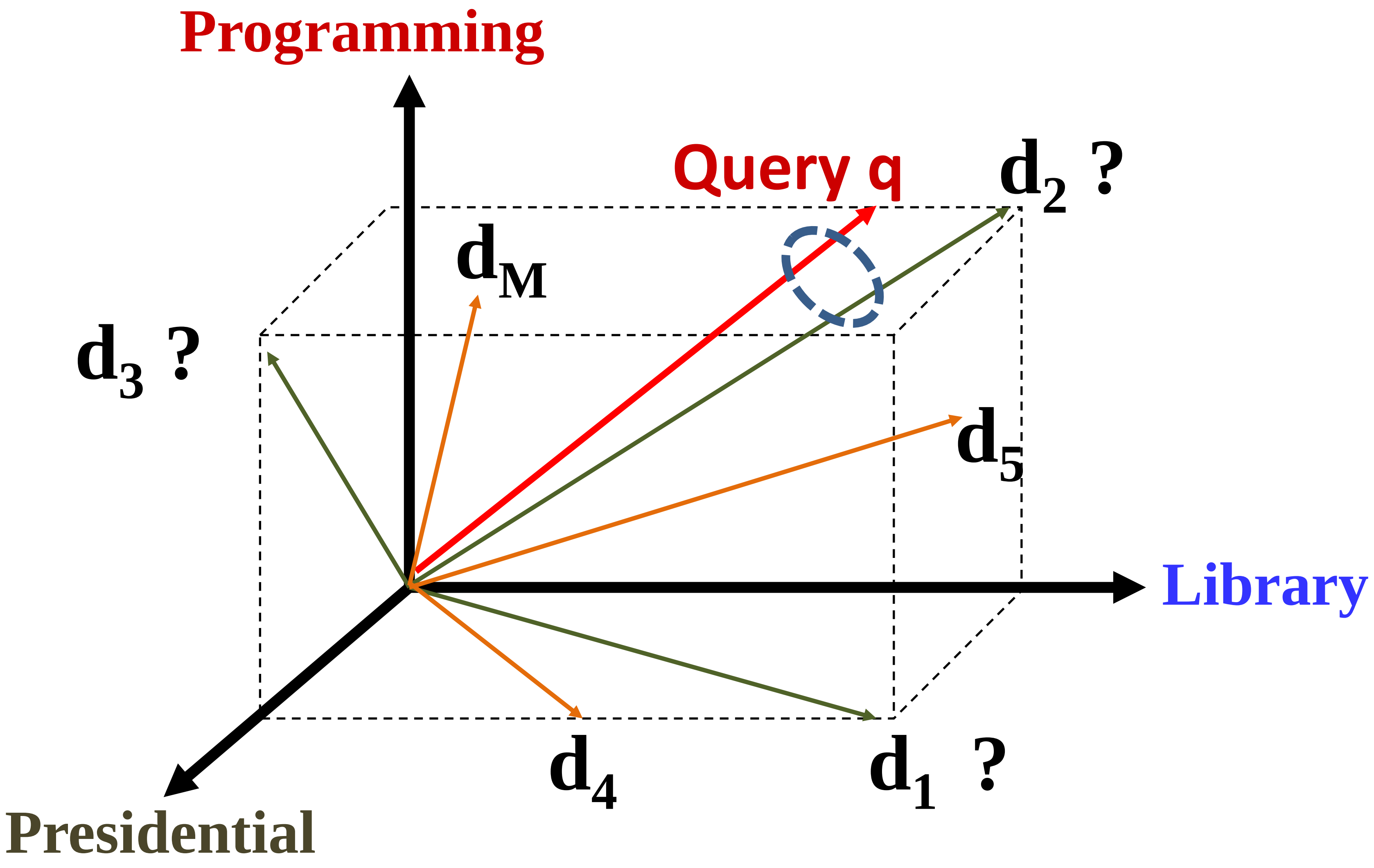
- Detailed discussion and comparison of state of the art models
  - Hui Fang, Tao Tao, and Chengxiang Zhai. 2011. Diagnostic Evaluation of Information Retrieval Models. *ACM Trans. Inf. Syst.* 29, 2, Article 7 (April 2011)

- Broad review of different retrieval models
  - ChengXiang Zhai, *Statistical Language Models for Information Retrieval* , Morgan & Claypool Publishers, 2008. (Chapter 2)

5. Many Different Retrieval Models

- Similarity-based models: f(q,d) = similarity(q,d)
  - Vector space model

Vector Space Model (VSM): Illustration



VSM Is a Framework

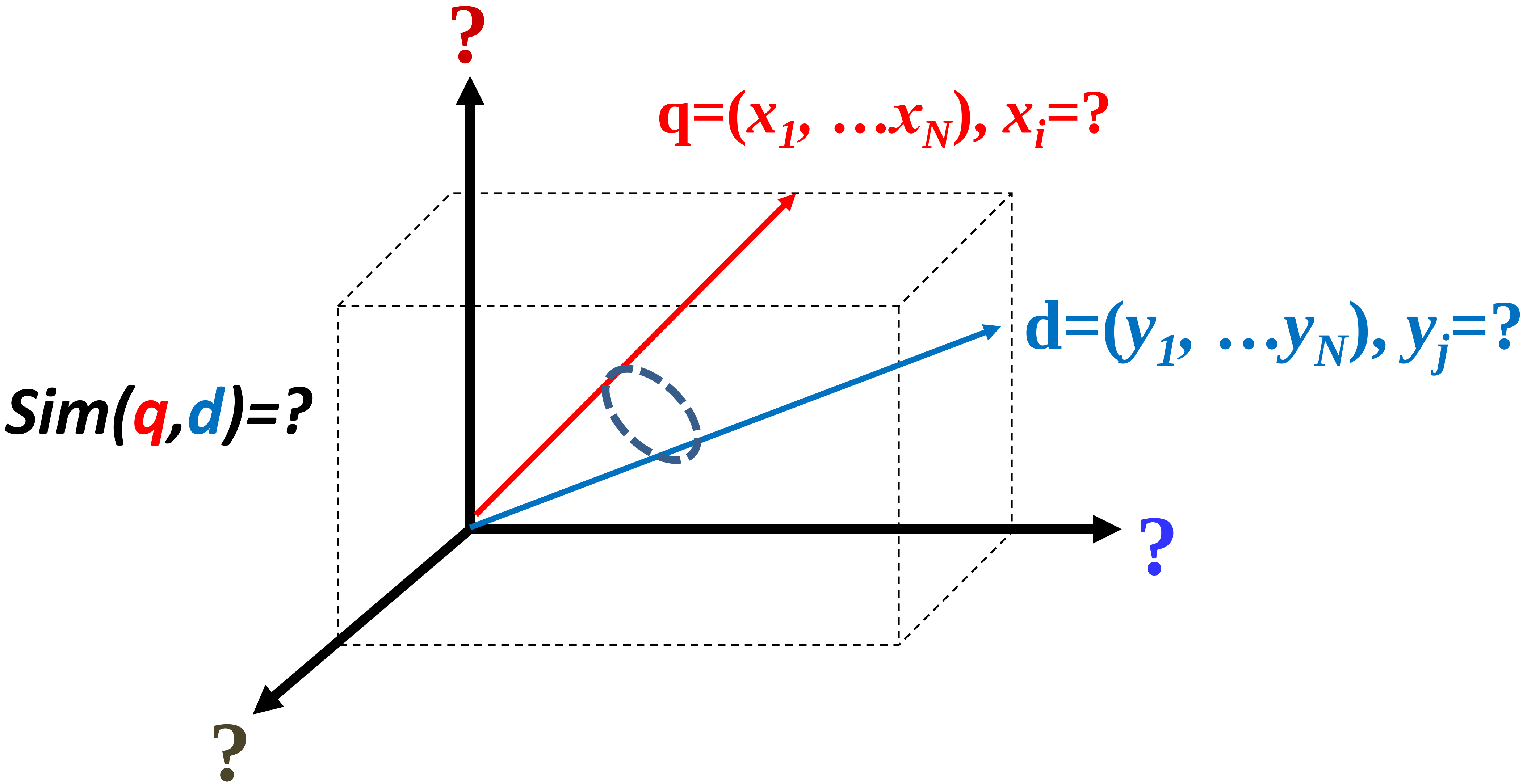
- Represent a doc/query by a term vector
  - **Term**: basic concept, e.g., word or phrase
  - Each term defines one dimension
  - N terms define an **N-dimensional space**
  - **Query** vector: $\mathbf{q}=(x_1, \ldots x_N)$, $x_i \in \Re$ is query term weight
  - **Doc** vector: $\mathbf{d}=(y_1, \ldots y_N)$, $y_j \in \Re$ is doc term weight

- relevance(q,d) $\propto$ similarity(**q**,**d**) =f(q,d)

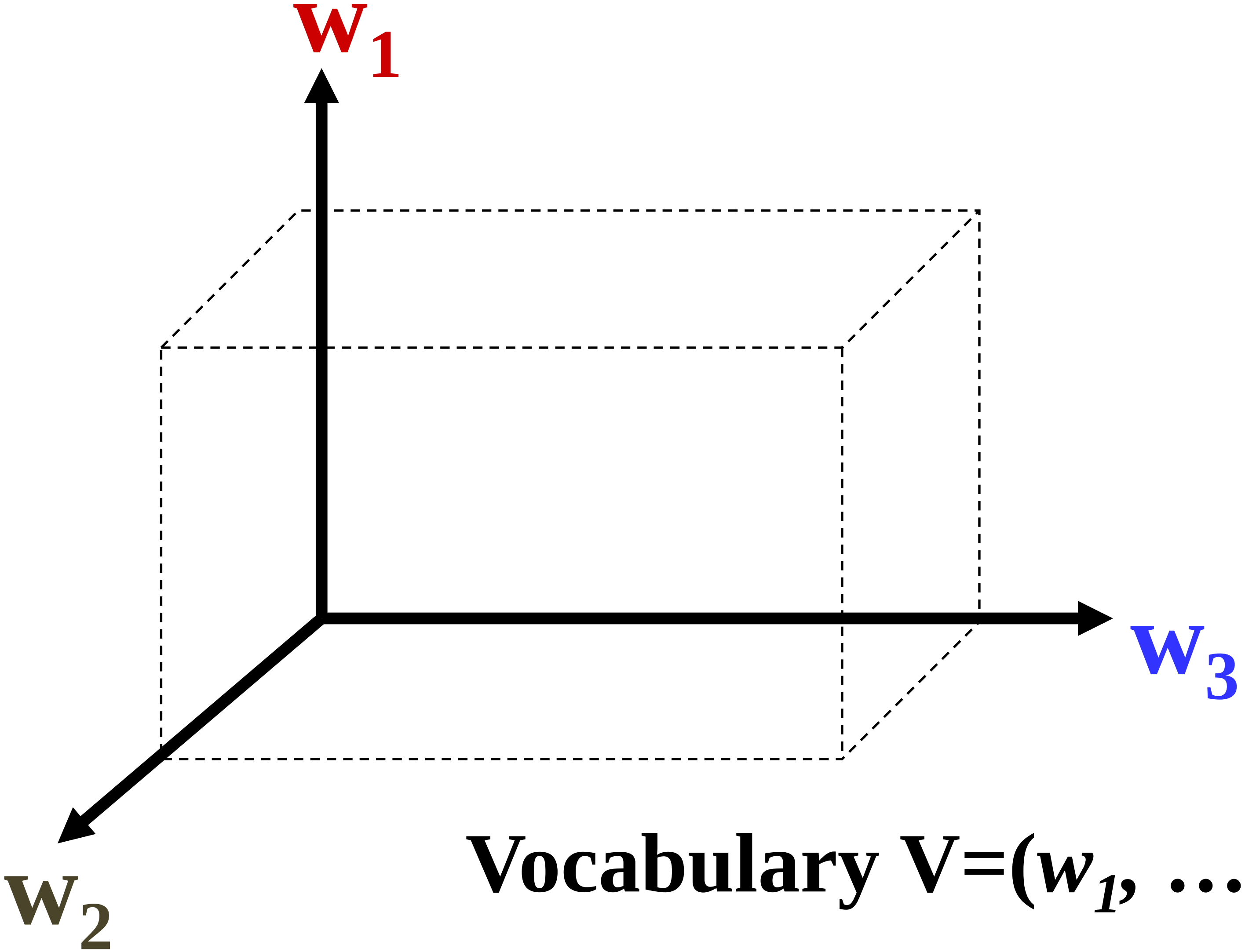
6. What VSM Doesn't Say

- How to define/select the "basic concept"
  - Concepts are assumed to be orthogonal
- How to place docs and query in the space (= how to assign term weights)
  - Term weight in query indicates importance of term
  - Term weight in doc indicates how well the term characterizes the doc
- How to define the similarity measure

What VSM Doesn't Say

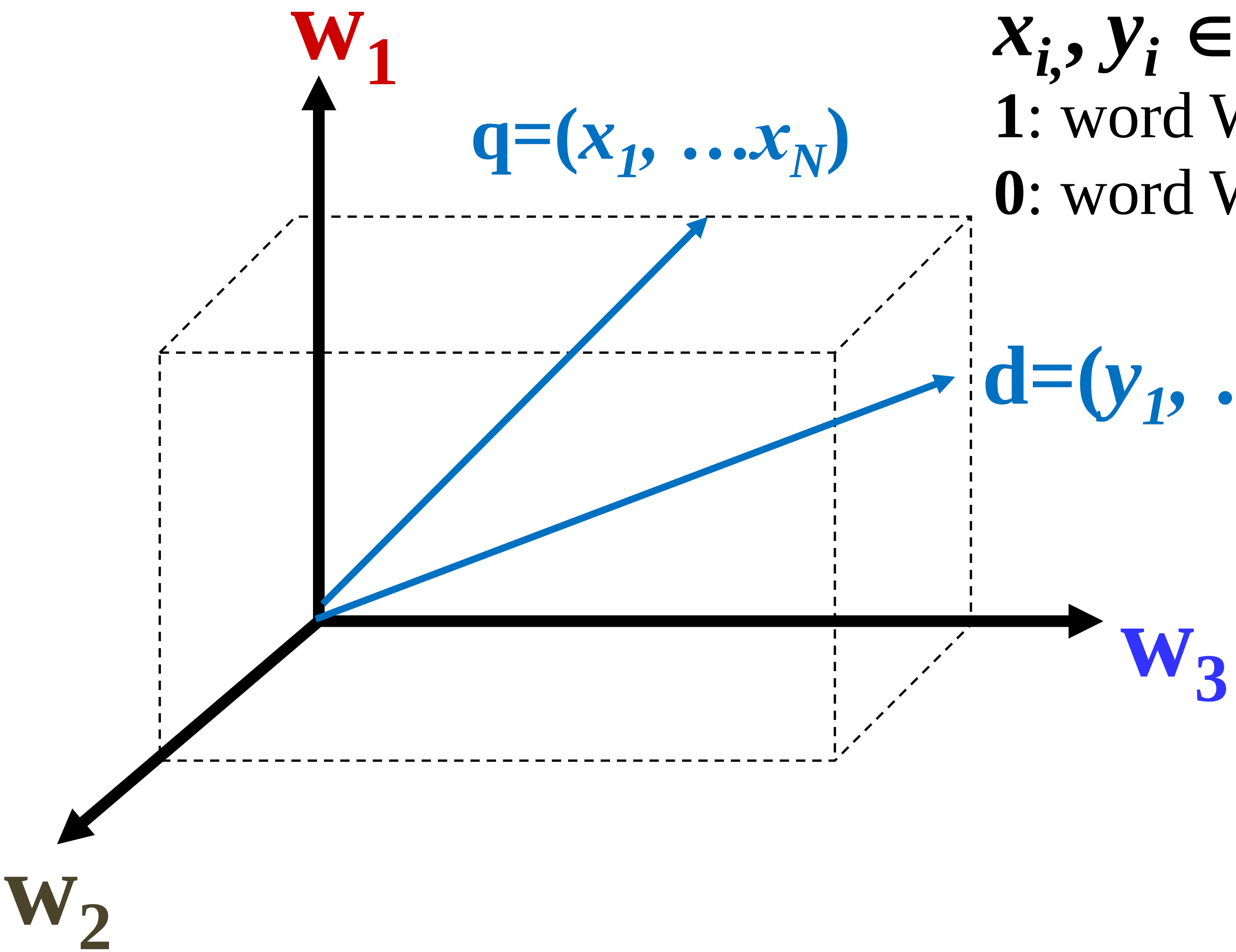


Dimension Instantiation: Bag of Words (BOW)



Vocabulary V=(w_1, …w_N)

Vector Placement: Bit Vector



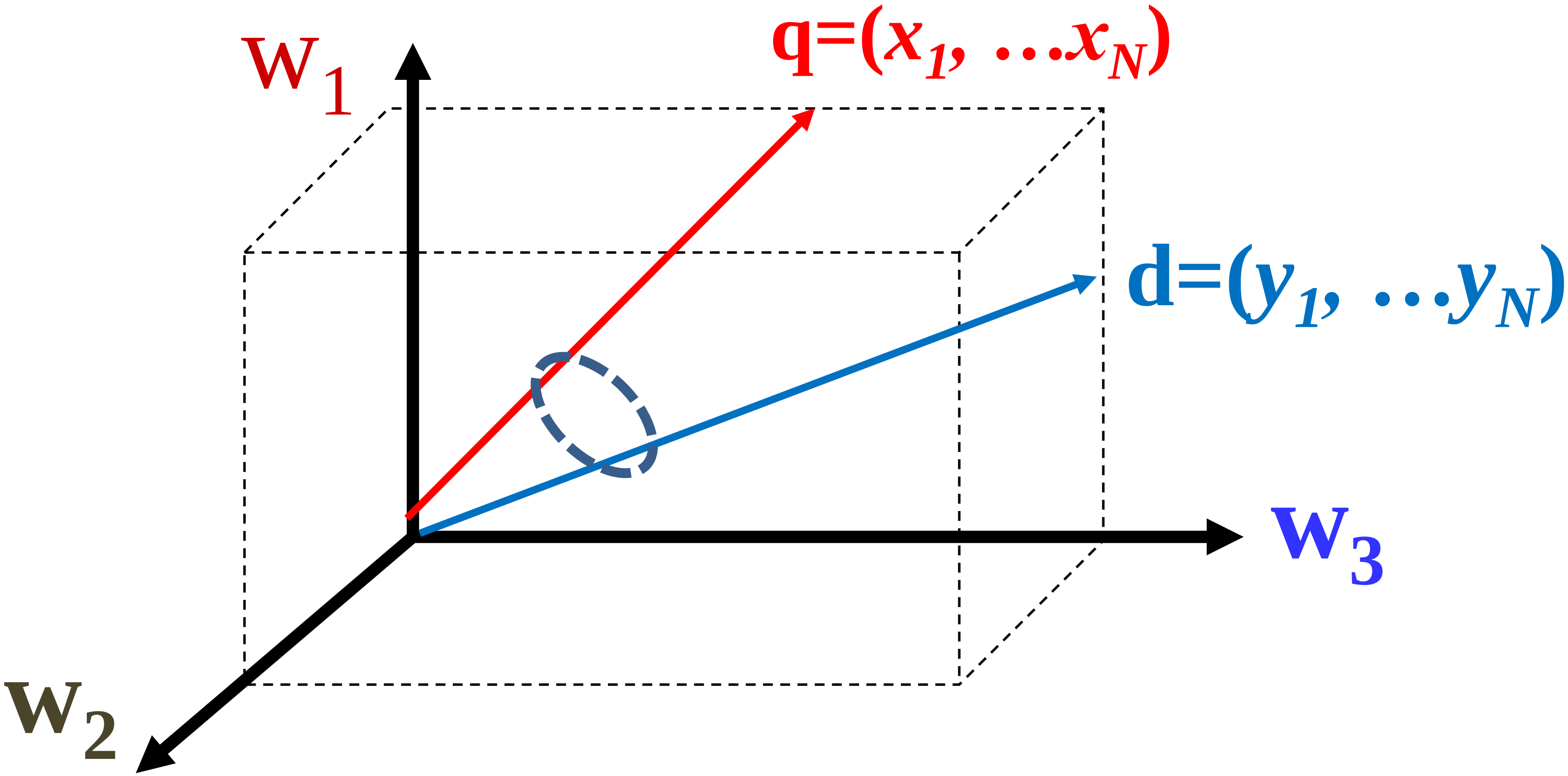
$x_i, y_i \in \{0,1\}$

1: word W_i is **present**

0: word W_i is **absent**

Similarity Instantiation: Dot Product

$$Sim(q,d)=q.d= x_1 y_1 + \ldots + x_N y_N = \sum_{i=1}^{N} x_i y_i$$



Simplest VSM= Bit-Vector + Dot-Product + BOW

$q=(x_1, \ldots x_N)$

$\mathbf{d=(y_1, \ldots y_N)}$

$x_i, y_i \in \{0,1\}$

1: word W_i is **present**

0: word W_i is **absent**

$$Sim(q,d)=q.d= x_1 y_1 + \ldots + x_N y_N = \sum_{i=1}^{N} x_i y_i$$

What does this ranking function intuitively capture?
Is this a good ranking function?

An Example: How Would You Rank These Documents?

Query = "**news about presidential campaign**"

Ideal Ranking?

d1 … **news about** …

d2 … **news about** organic food **campaign**…

d3 … **news** of **presidential campaign** …

d4 … **news** of **presidential campaign** …
… **presidential** candidate …

d5 … **news** of organic food **campaign**…
campaign…**campaign**…**campaign**…

d4 +
d3 +

d1 -
d2 -
d5 -

Ranking Using the Simplest VSM

Query = "news about presidential campaign"

d1 | … **news about** …

d3 | … **news** of **presidential campaign** …

V= {news, about, presidential, campaign, food …. }

q=	**(1,**	**1,**	**1,**	**1,**	**0,**	**…)**
d1=	**(1,**	**1,**	**0,**	**0,**	**0,**	**…)**

$$f(q,d1) = 1*1+1*1+1*0+1*0+0*0+\ldots = 2$$

d3=	**(1,**	**0,**	**1,**	**1,**	**0,**	**…)**

$$f(q,d3) = 1*1+1*0+1*1+1*1+0*0+\ldots = 3$$

Is the Simplest VSM Effective?

Query = "news about presidential campaign"

d1 … **news about** … f(q,d1)=2

d2 … **news about** organic food **campaign**… f(q,d2)=3

d3 … **news** of **presidential campaign** … f(q,d3)=3

d4 … **news** of **presidential campaign** …
… **presidential** candidate … f(q,d4)=3

d5 … **news** of organic food **campaign**…
campaign…**campaign**…**campaign**… f(q,d5)=2

Summary

- VSM instantiation: dimension, vector placement, similarity
- Simplest VSM
  - Dimension = word
  - Vector = 0-1 bit vector (word presence/absence)
  - Similarity = dot product
  - f(q,d) = number of **distinct** query words matched in d