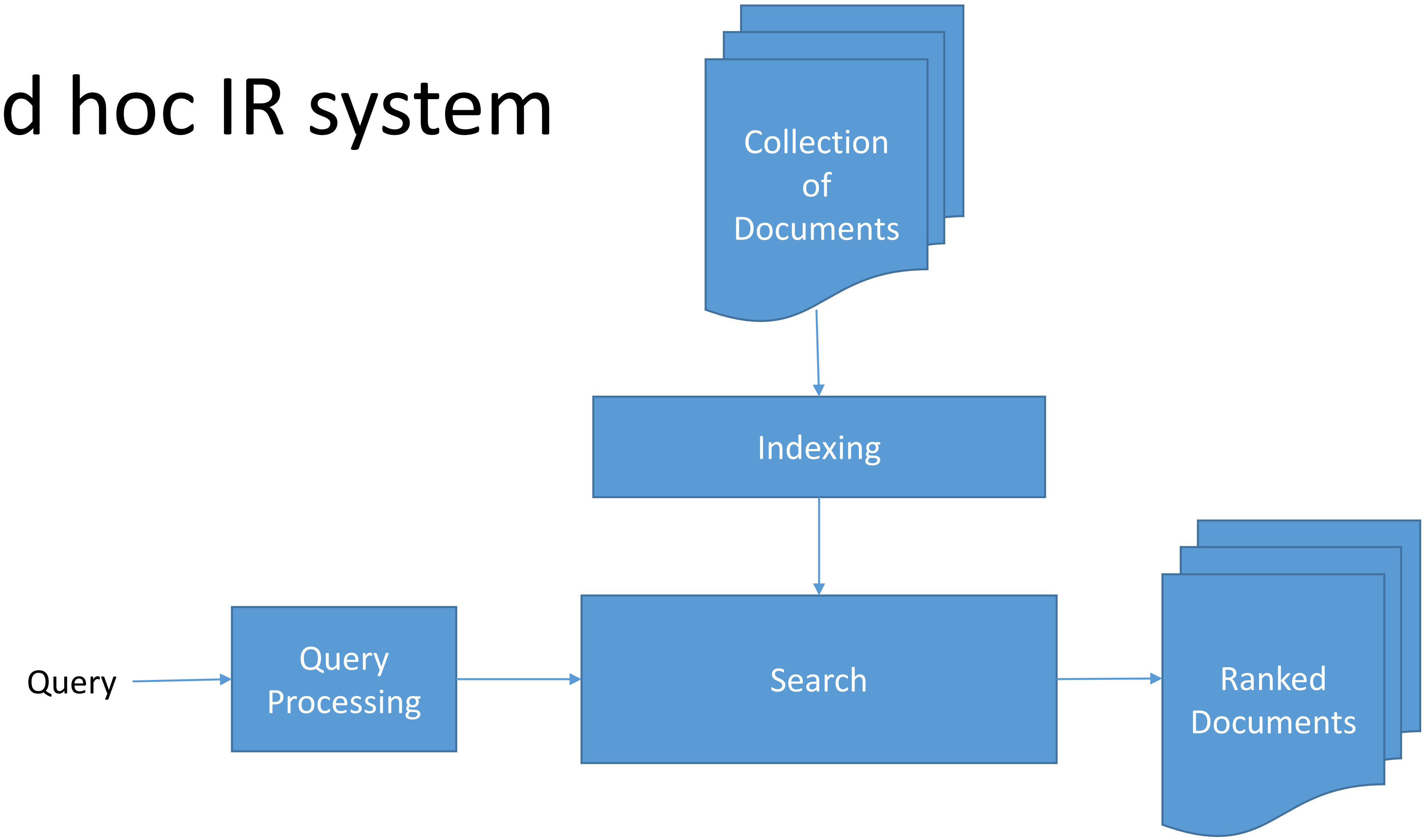
Information Retrieval (IR)

Chapter 23

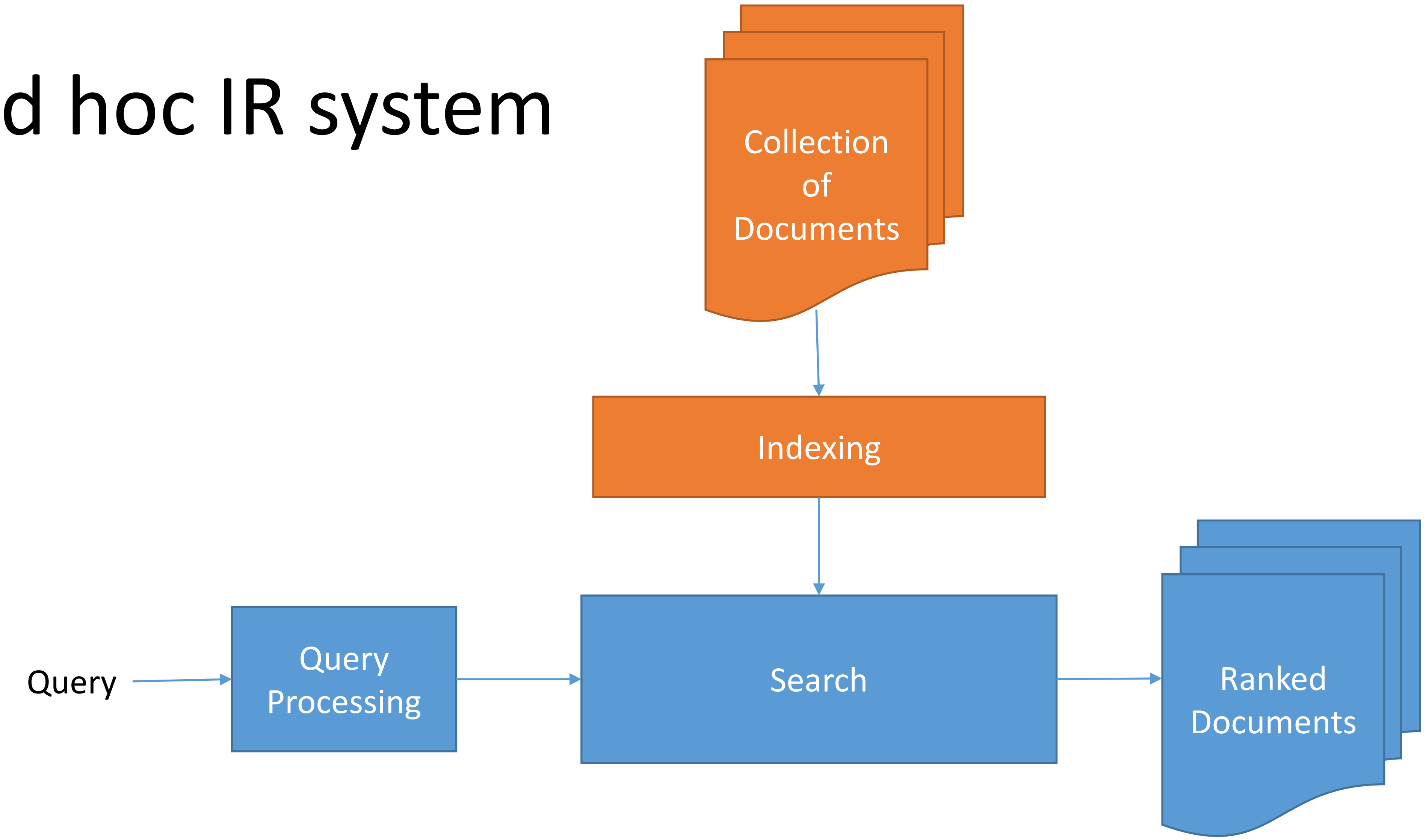
Information Retrieval

- Focus:
  - Storage of text documents
  - Retrieval of documents based on users’ query

Ad hoc IR system



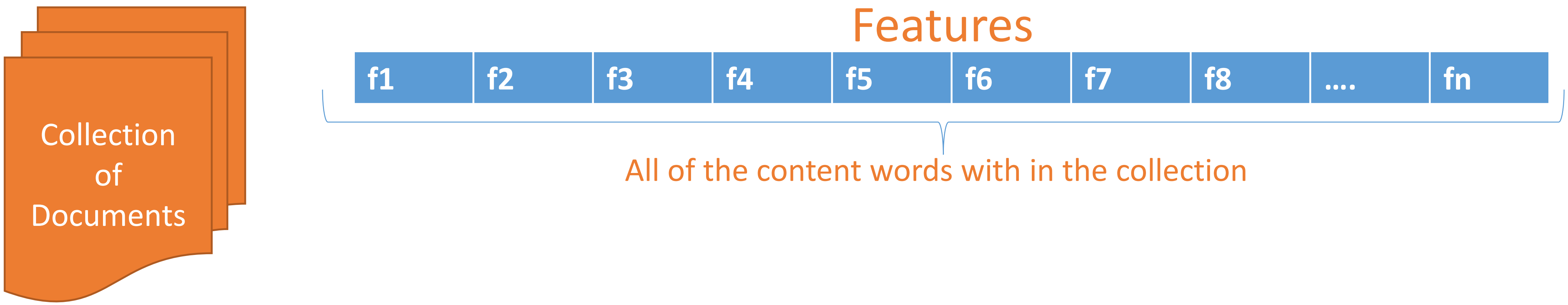
Ad hoc IR system



How do we go about representing a document?

Vector Space Model

- Documents are represented as a vector of features representing terms (words) that occur within the collection



Vector Space Model

- Documents and queries are represented as a vector of features representing terms (words) that occur with in the collection

Collection of Documents

Features

f1	f2	f3	f4	f5	f6	f7	f8	….	fn

All of the content words with in the collection

Document

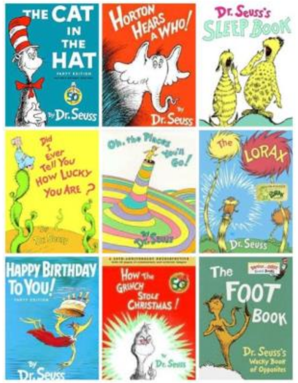
Feature Vector for Document

1	0	0	0	1	0	0	0	….	1

whether the feature exists with in the document

Vector Space Model

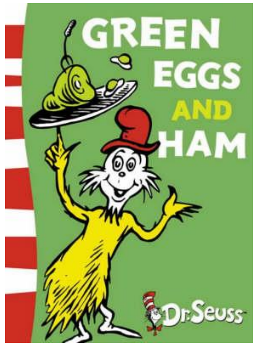
- Documents and queries are represented as a vector of features representing terms (words) that occur with in the collection



Features

cat	hat	green	eggs	ham	sam	grinch	stole	….	tree

All of the content words with in the collection

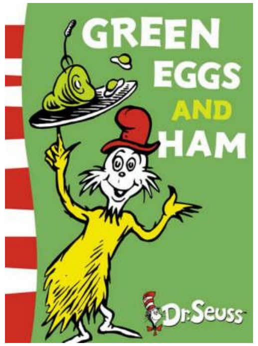


Feature Vector for Document

0	0	1	1	1	1	0	0	….	0

whether the feature exists with in the document

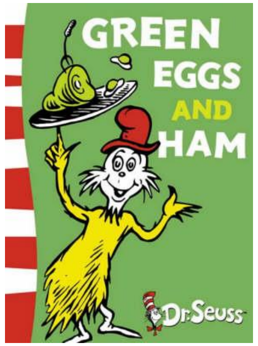
Mathematically



0	0	1	1	1	1	0	0	….	0

$$\vec{d}_j = (0,0,1,1,1,1,0,0, \dots, 0)$$

Term weighting



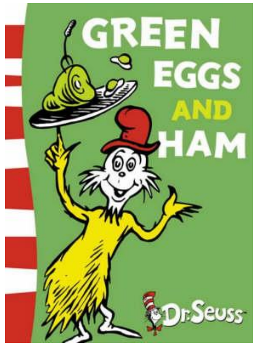
Binary Vector

0	0	1	1	1	1	0	0	….	0

But we know additional ways to represent a word (or feature) in the vector

what are some?

Term weighting



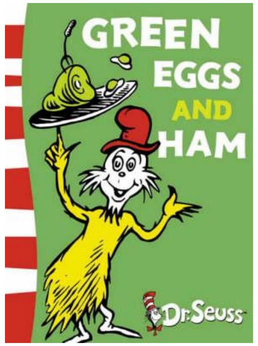
Binary Vector

0	0	1	1	1	1	0	0	….	0

Frequency Vector

0	0	5	10	6	50	0	0	….	0

Term weighting



Binary Vector

0	0	1	1	1	1	0	0	….	0

Frequency Vector

0	0	5	10	6	50	0	0	….	0

IDF Vector

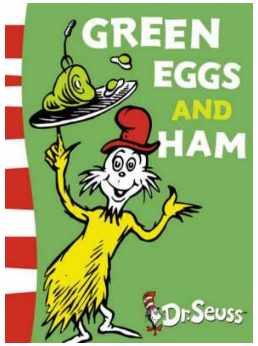
0	0	2.9	2.3	2.8	.69	0	0	….	0

$$\mathrm{idf}(t, D) = \log \frac{N}{|\{d \in D : t \in d\}|}$$

with

- N: total number of documents in the corpus
- $|\{d \in D : t \in d\}|$: number of documents where the term t appears (i.e., $\mathrm{tf}(t, d) \neq 0$). If the term is not in the corpus, this will lead to a division-by-zero. It is therefore common to adjust the denominator to $1 + |\{d \in D : t \in d\}|$.

Term weighting



Binary Vector

0	0	1	1	1	1	0	0	….	0

Frequency Vector

0	0	5	10	6	50	0	0	….	0

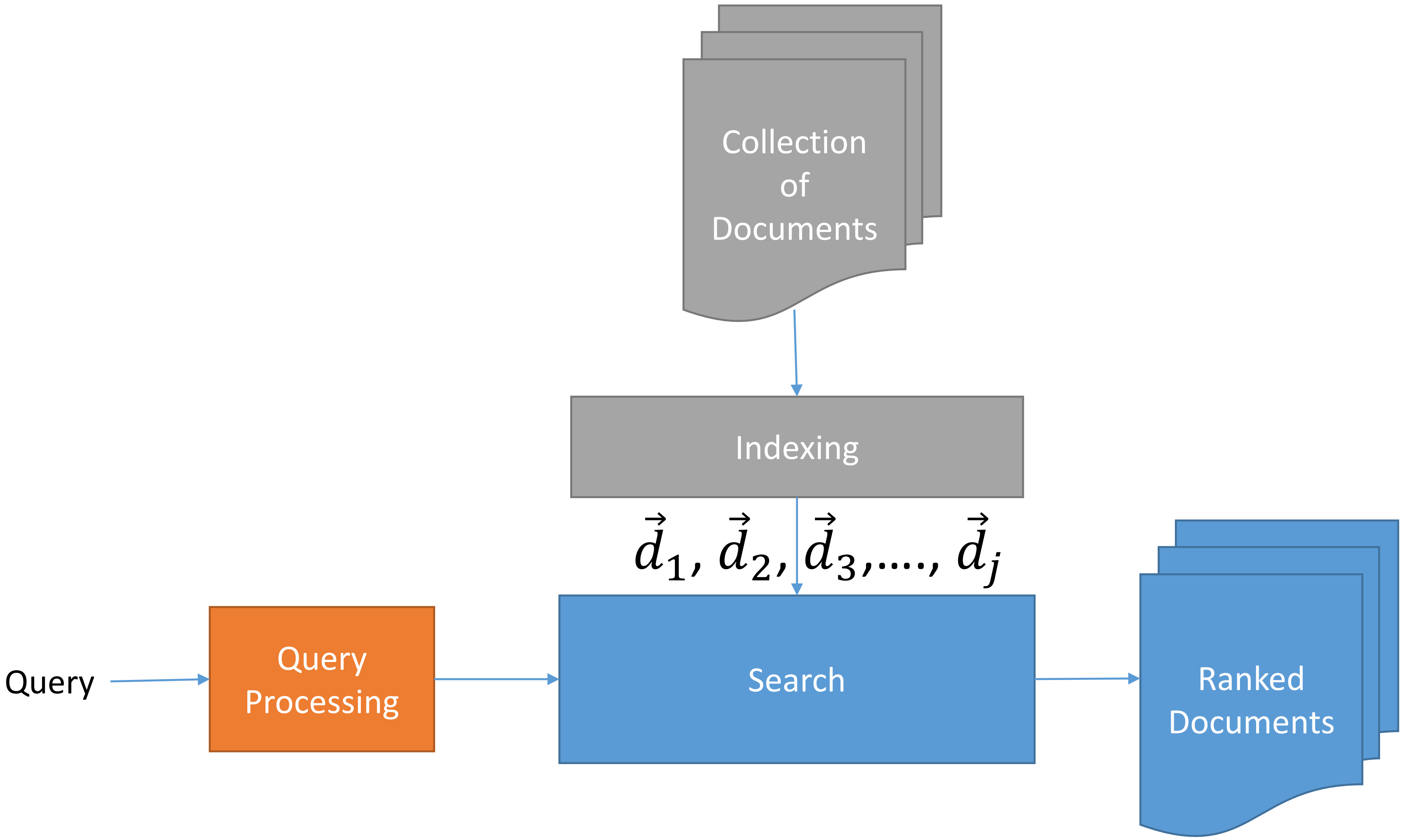
IDF Vector

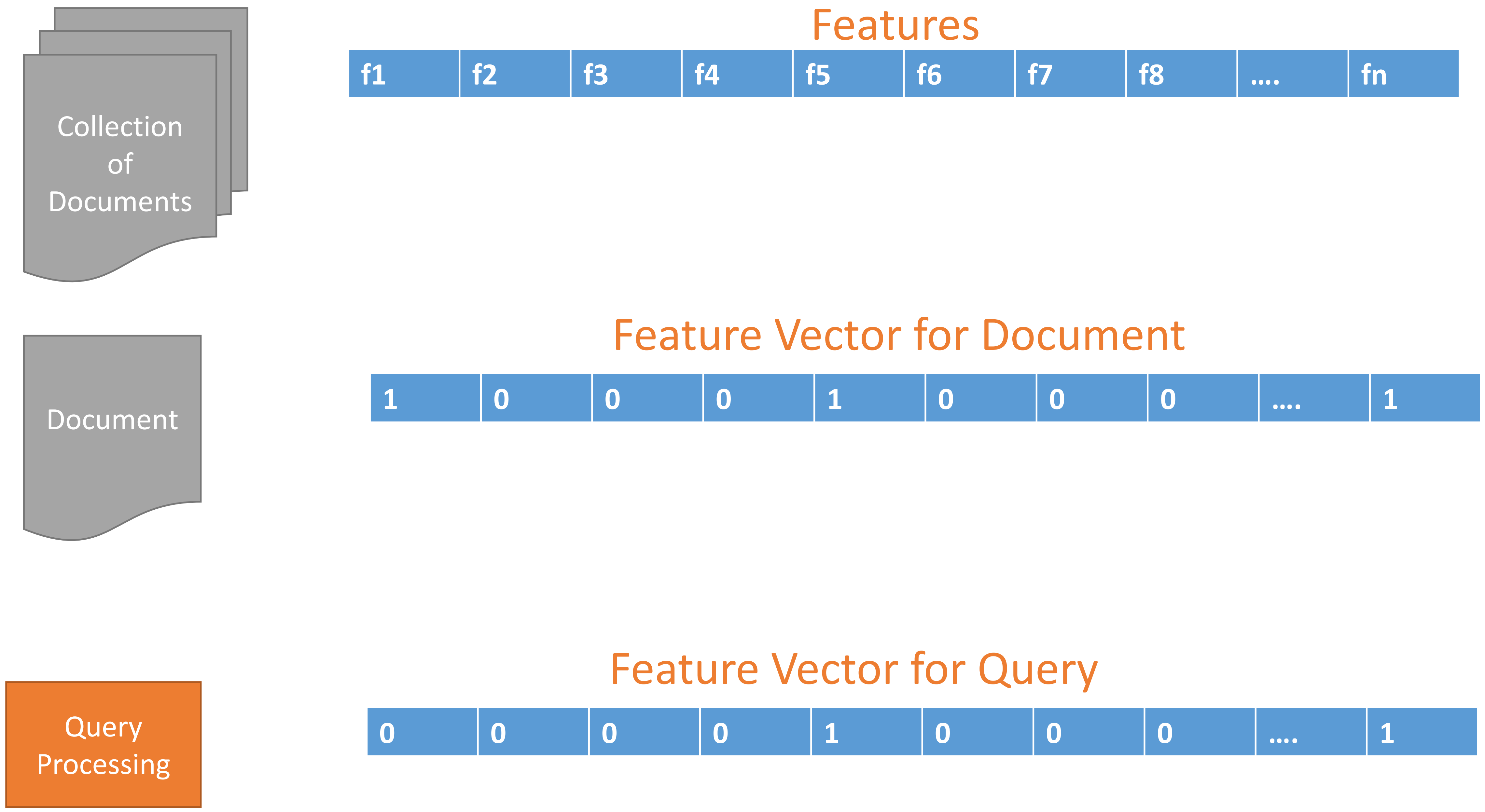
0	0	2.9	2.3	2.8	.69	0	0	….	0

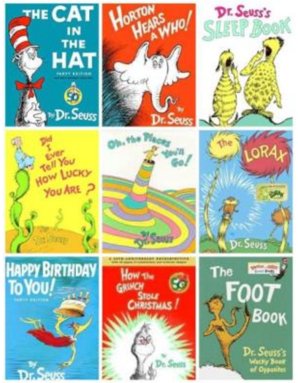
TF-IDF Vector

0	0	14.5	23	16.8	16.8	0	0	….	0

$$\mathrm{tfidf}(t, d, D) = \mathrm{tf}(t, d) \times \mathrm{idf}(t, D)$$

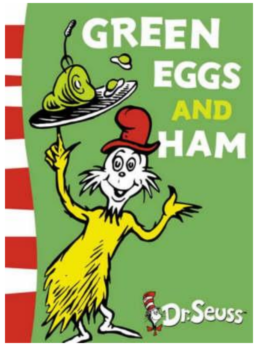

Collection of Documents
Indexing
$\vec{d}_1, \vec{d}_2, \vec{d}_3, \ldots, \vec{d}_j$
Query
Query Processing
Search
Ranked Documents


Collection of Documents
Features
f1 f2 f3 f4 f5 f6 f7 f8 .... fn
Document
Feature Vector for Document
1 0 0 0 1 0 0 0 1
Query Processing
Feature Vector for Query
0 0 0 0 1 0 0 0 1



Features

cat	hat	green	eggs	ham	sam	grinch	stole	….	tree



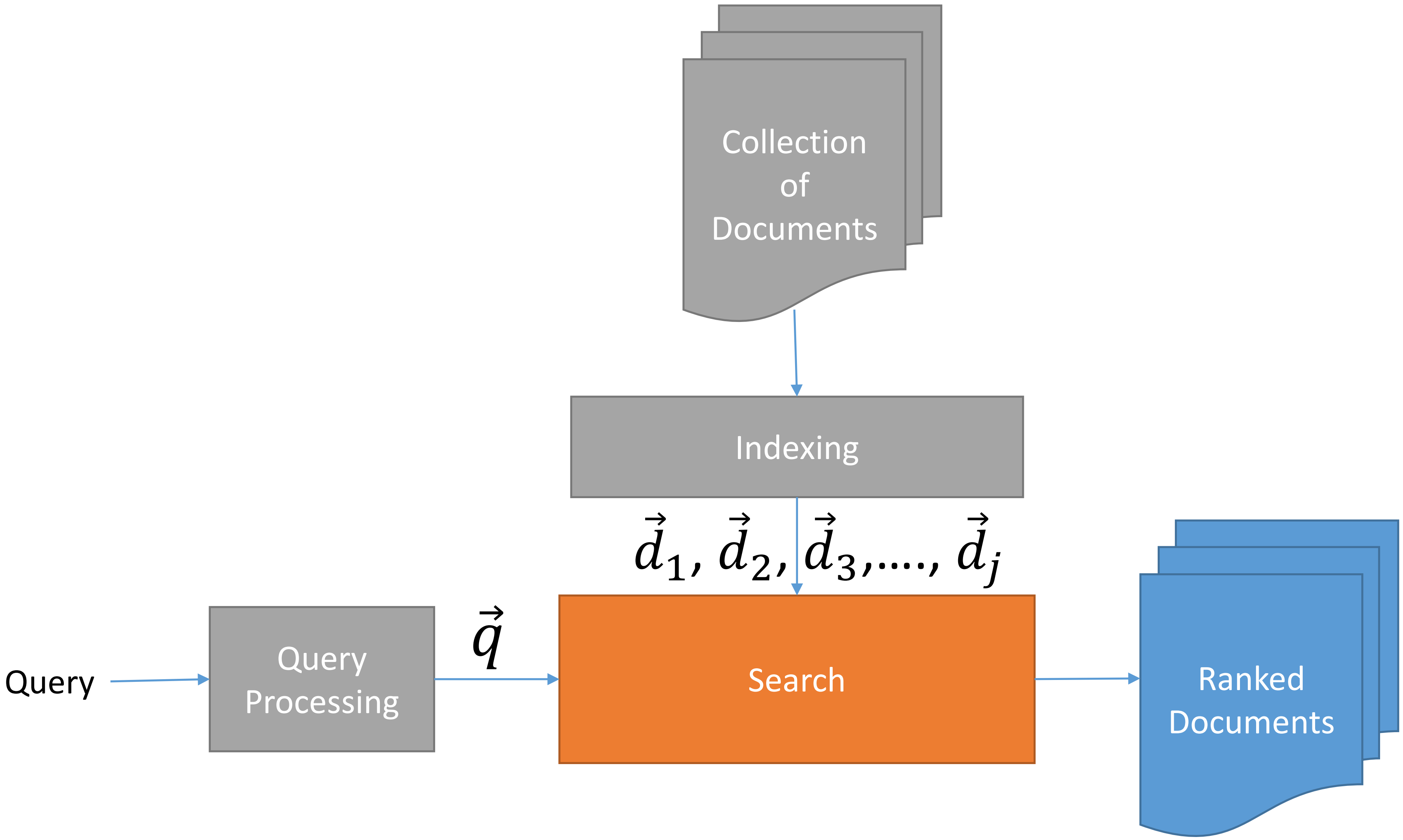
Feature Vector for Document

0	0	1	1	1	1	0	0	….	0

sam I am

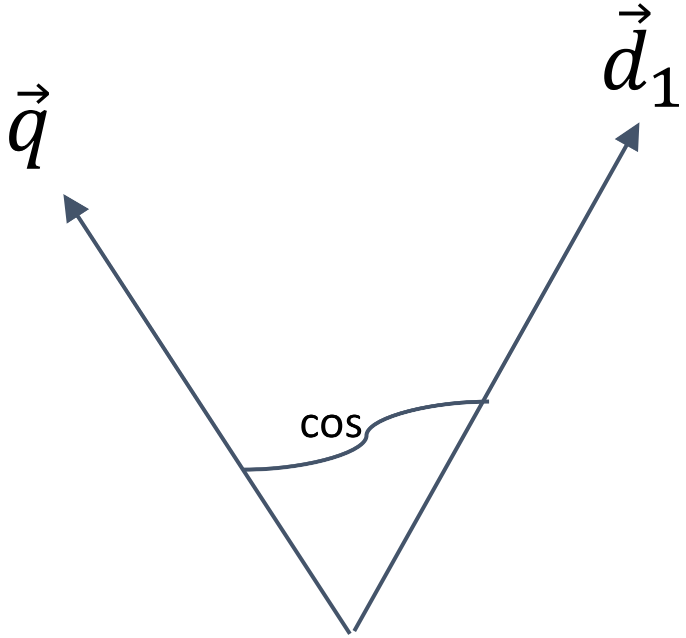
Feature Vector for Query

0	0	0	0	0	1	0	0	….	0

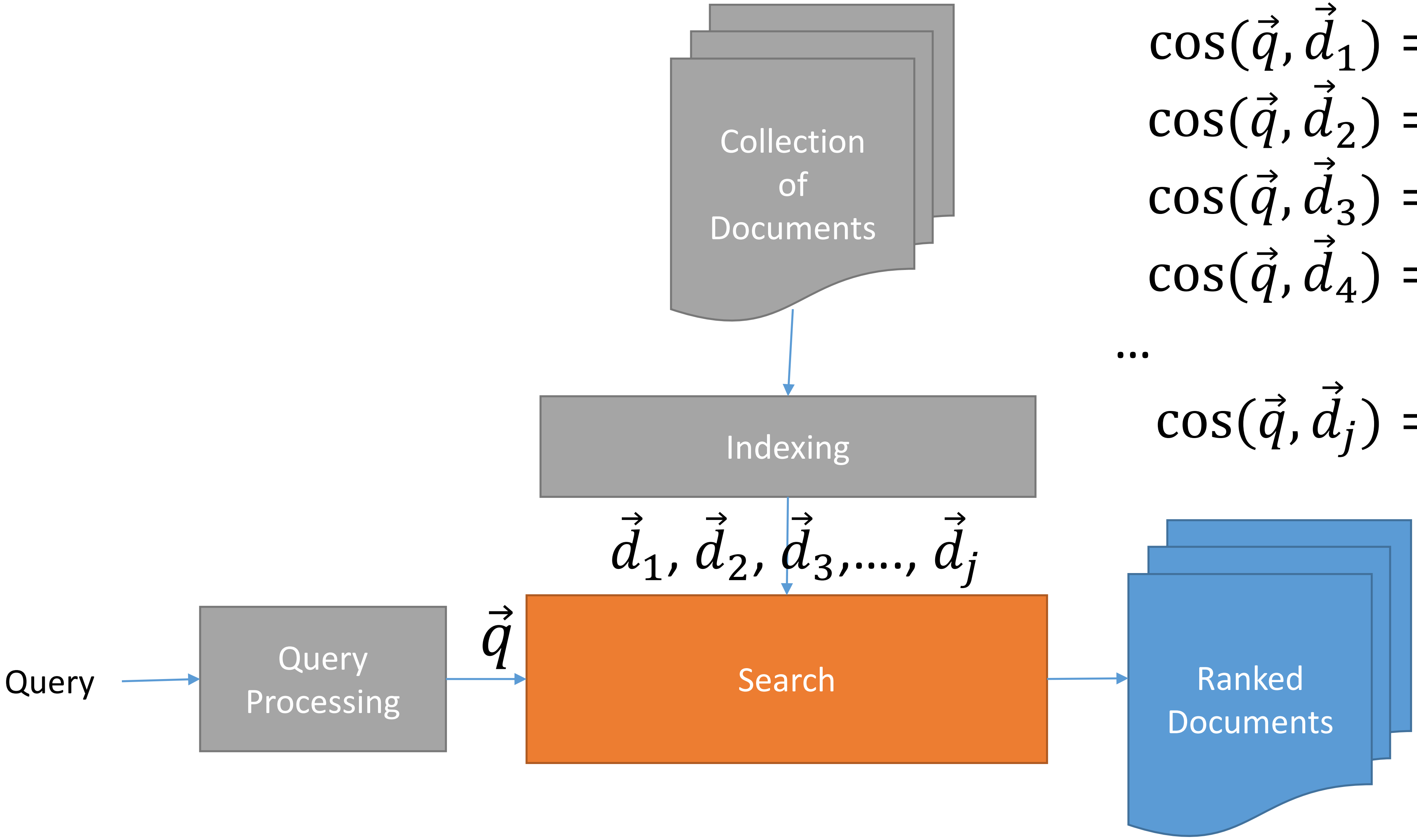


Given our query how do we return relevant documents

Cosine similarity



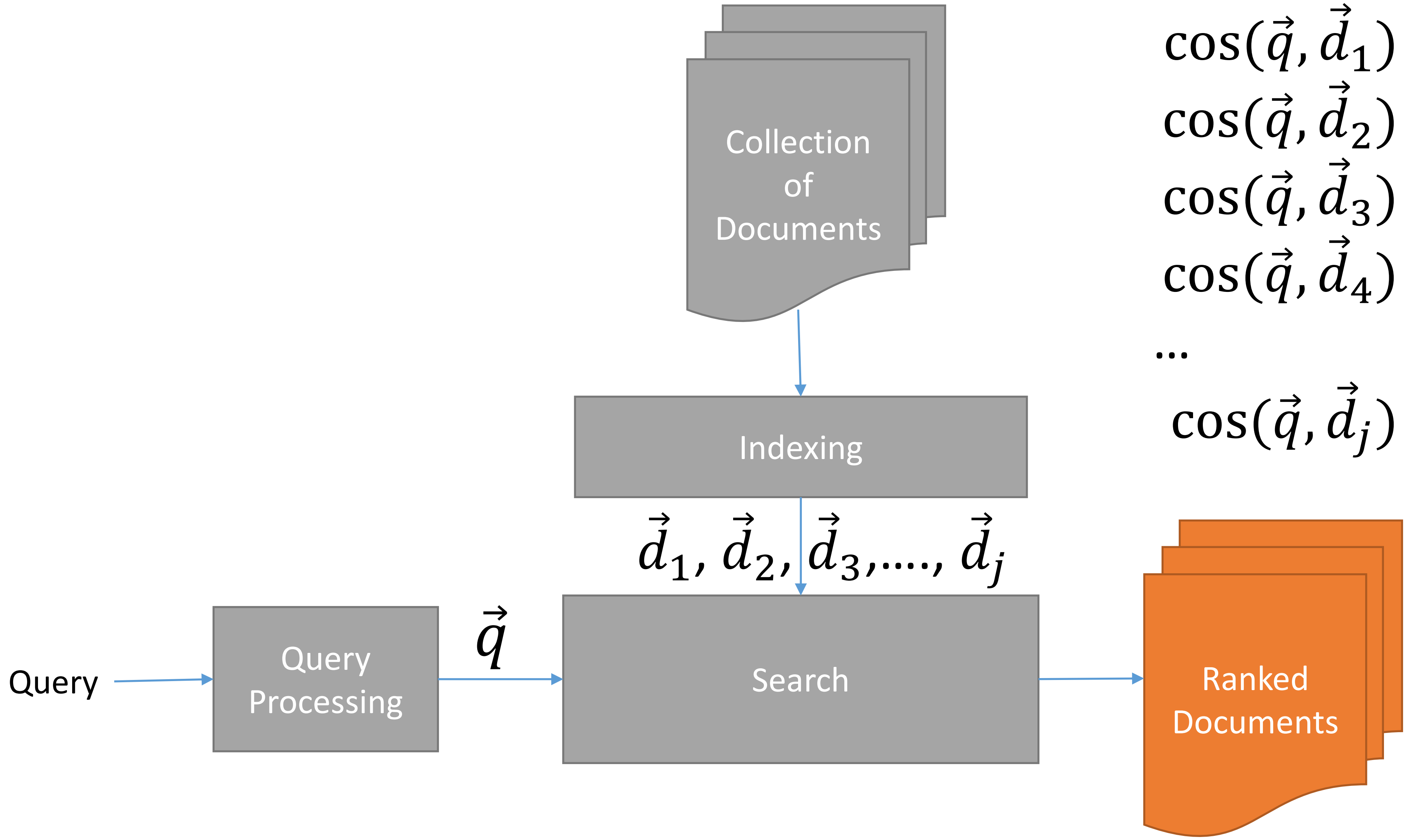
$$Cosine(\overrightarrow{x}\ \overrightarrow{y}) = \frac{\overrightarrow{x} \cdot \overrightarrow{y}}{|\overrightarrow{x}||\overrightarrow{y}|}$$



$$\cos(\vec{q}, \vec{d}_1) = rank_1$$
$$\cos(\vec{q}, \vec{d}_2) = rank_2$$
$$\cos(\vec{q}, \vec{d}_3) = rank_3$$
$$\cos(\vec{q}, \vec{d}_4) = rank_4$$
…
$$\cos(\vec{q}, \vec{d}_j) = rank_j$$

Calculate the Cosine between query vector and each of the document vectors

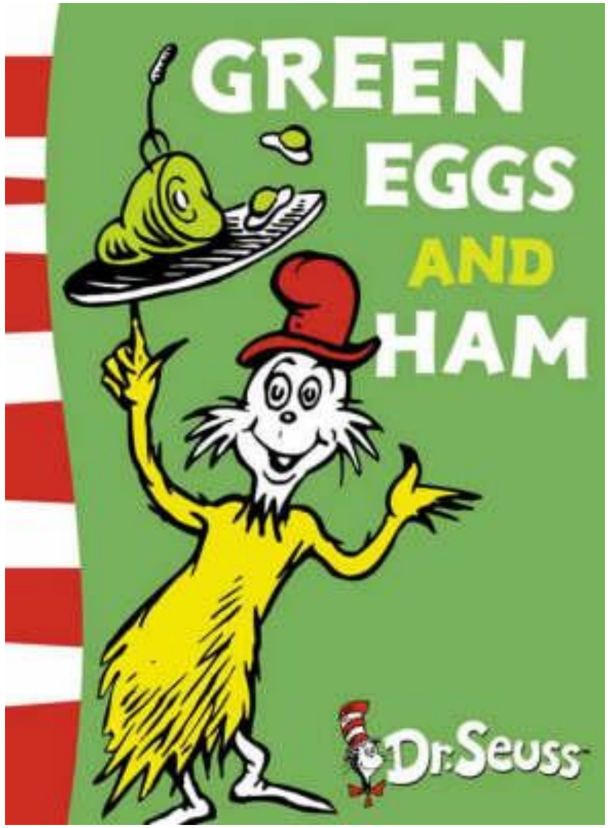
$$\cos(\vec{q},\vec{d}_1) = rank_1$$
$$\cos(\vec{q},\vec{d}_2) = rank_2$$
$$\cos(\vec{q},\vec{d}_3) = rank_3$$
$$\cos(\vec{q},\vec{d}_4) = rank_4$$
…
$$\cos(\vec{q},\vec{d}_j) = rank_j$$



$\vec{d}_5$
$\vec{d}_{19}$
$\vec{d}_{30}$
$\vec{d}_1$
…
$\vec{d}_i$

where all the documents returned are above some threshold cutoff

Features are words in the documents



I am sam. Sam I am. That Sam I am. That Sam I am. I do not like that Sam I am.

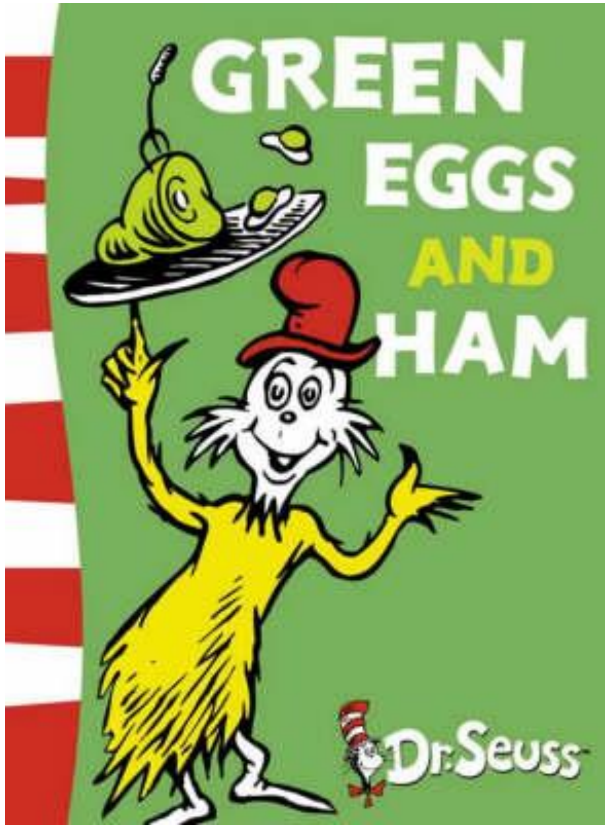
Do you like green eggs and ham?

I do not like them Sam I am. I do not l like green eggs and ham.

Do you like them in box?
Would you like them with a fox?

what tools have we used to *clean* the data?

Features are words in the documents



I am sam. Sam I am. That Sam I am.
That Sam I am. I do not like that Sam
I am.

Do you like green eggs and ham?

I do not like them Sam I am. I do not I
like green eggs and ham.

Do you like them in box?
Would you like them with a fox?

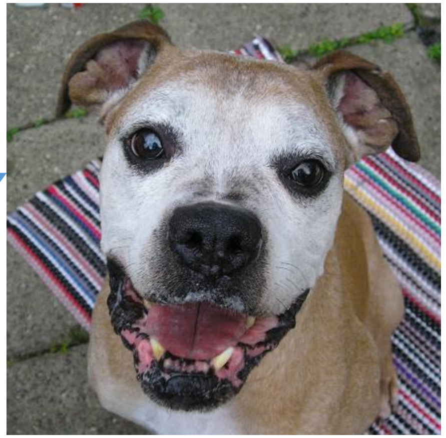
- stoplist
- punctuation
- lemmatization
- Stemming
  - eggs -> egg

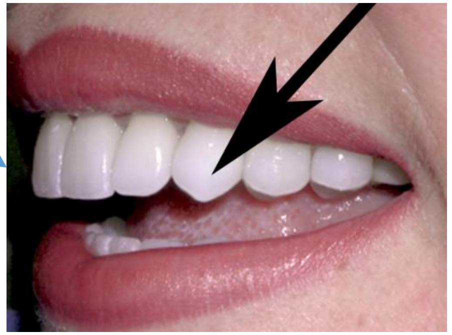
Disadvantages to term based VSM

north american canine breeders

Query Processing

Search



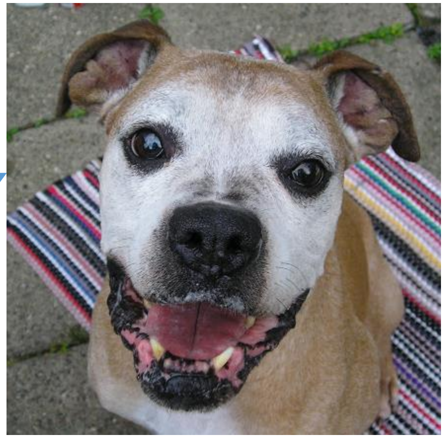


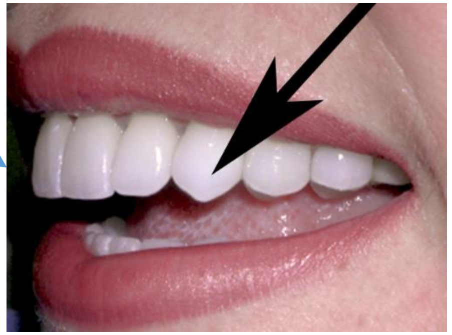
What NLP system could help us with this?

VSM = Vector Space Model

Disadvantages to term based VSM

north american canine breeders → Query Processing → Search



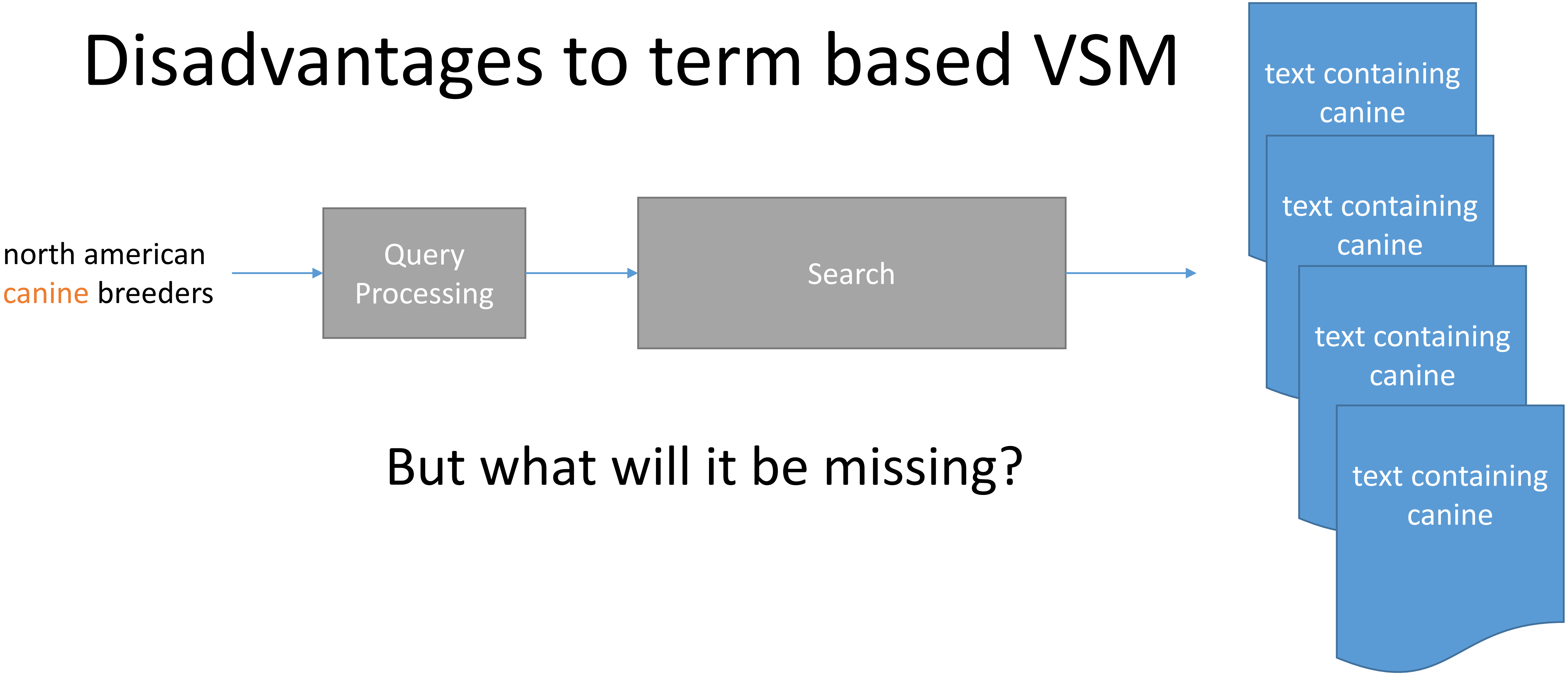


What NLP system could help us with this?

WSD

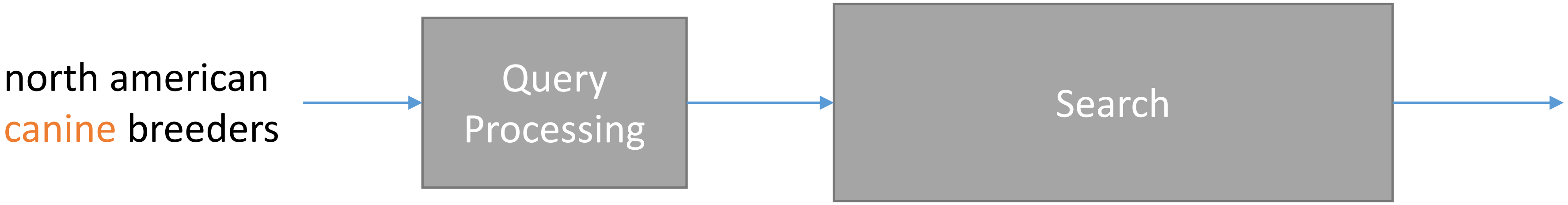
VSM = Vector Space Model

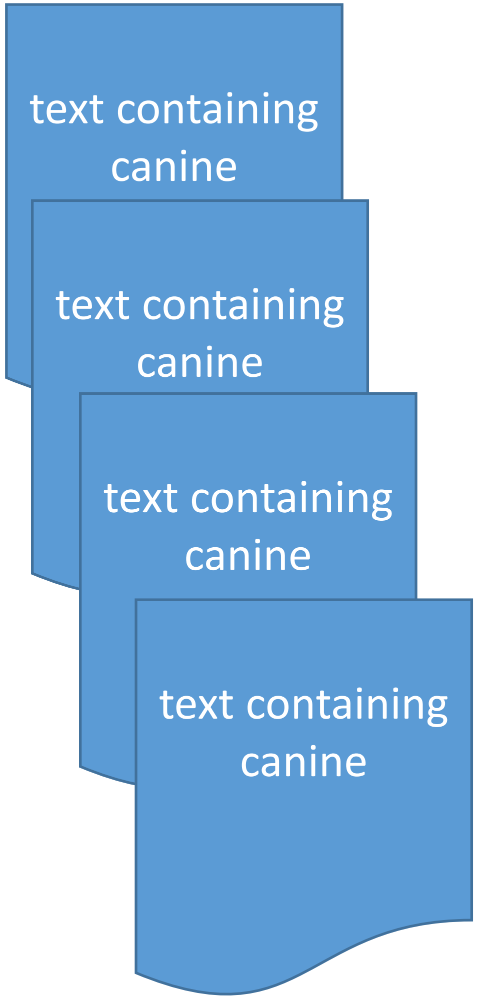
Disadvantages to term based VSM



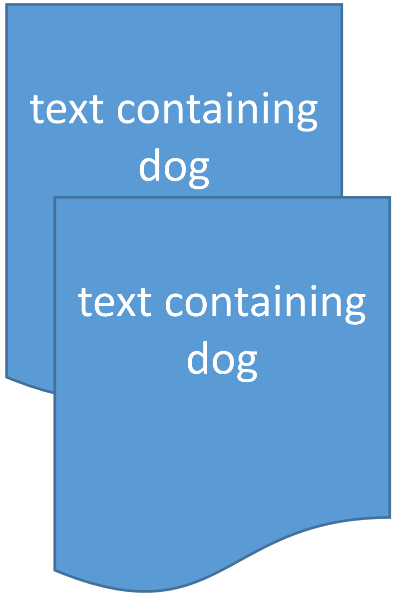
But what will it be missing?

Disadvantages to term based VSM





But what will it be missing?



Disadvantages to term based VSM

north american canine breeders → Query Processing → Search →

text containing canine

text containing canine

text containing canine

text containing canine

But what will it be missing?

text containing dog

text containing dog

What NLP system can help us with this?

semantic similarity and relatedness

Query Expansion

The user's original query is expanded by addition of terms that are synonymous with or related to the original terms

what doggy breeds are good with kids

Evaluation of Information Retrieval

- Precision: how many documents returned are relevant
- Recall: how many of the relevant documents are returned

$$Precision = \frac{|R|}{|T|} \qquad Recall = \frac{|R|}{|U|}$$

R = relevant documents returned by the system

T = total documents returned by the system

U = documents in the collection that are relevant

Problem with these two metrics for IR

- Does not incorporate any rank information.

System 1 ranking: $\underbrace{\vec{d}_1\ \vec{d}_2\ \vec{d}_3\ \vec{d}_4\ \vec{d}_5}_{\text{Not relevant}}\ \underbrace{\vec{d}_6\ \vec{d}_7\ \vec{d}_8\ \vec{d}_9\ \vec{d}_{10}}_{\text{Relevant}}$

System 2 ranking: $\underbrace{\vec{d}_1\ \vec{d}_2\ \vec{d}_3\ \vec{d}_4\ \vec{d}_5}_{\text{Relevant}}\ \underbrace{\vec{d}_6\ \vec{d}_7\ \vec{d}_8\ \vec{d}_9\ \vec{d}_{10}}_{\text{Not relevant}}$

Precision and Recall for both systems is the same but which is the better system?

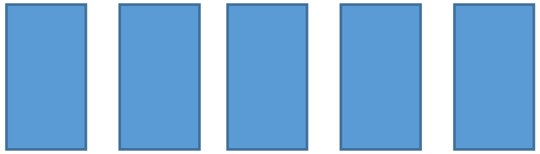
Mean Average Precision (MAP)

- In this approach, descend through the ranked list of terms and note the precision only at those points where a relevant item has been encountered

so we are weighting the precision on the ranking

$$MAP = \frac{1}{R_r} \sum_{d \in R_r} Precision_r(d)$$

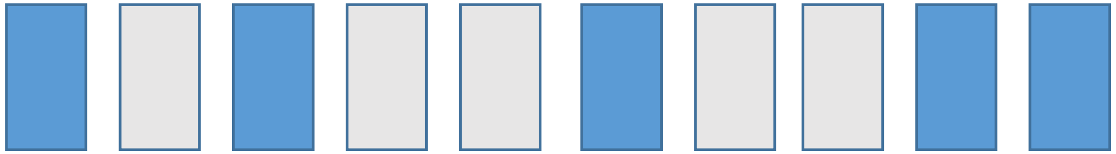
Mean Average Precision (MAP) Example



= relevant documents for query

$$MAP = \frac{1}{R_r} \sum_{d \in R_r} Precision_r(d)$$

Ranking #1



Precision 1.0 0.67 0.5 0.44 0.5

$$MAP = \frac{(1.0 + 0.67 + 0.5 + 0.44 + 0.5)}{5} = 0.62$$

More References

- Stanford University NLP Course/Textbook: Speech and Language Processing
  - Dan Jurafsky and James Martin
  - Vector Semantics
  - https://web.stanford.edu/~jurafsky/slp3/slides/vector1.pdf

Added by Dr. Liao

Next Up

- Coming up
  - Question Answering