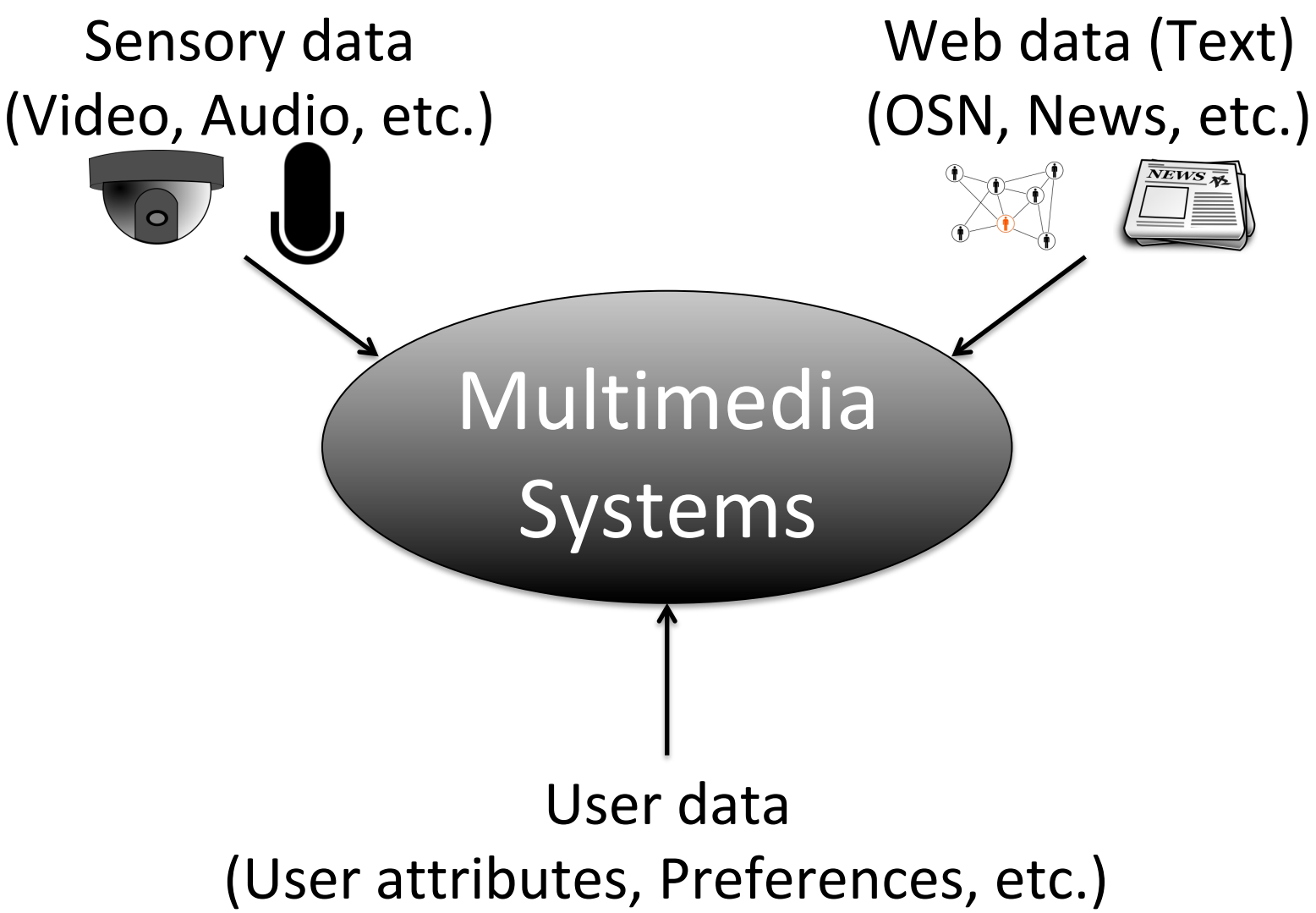
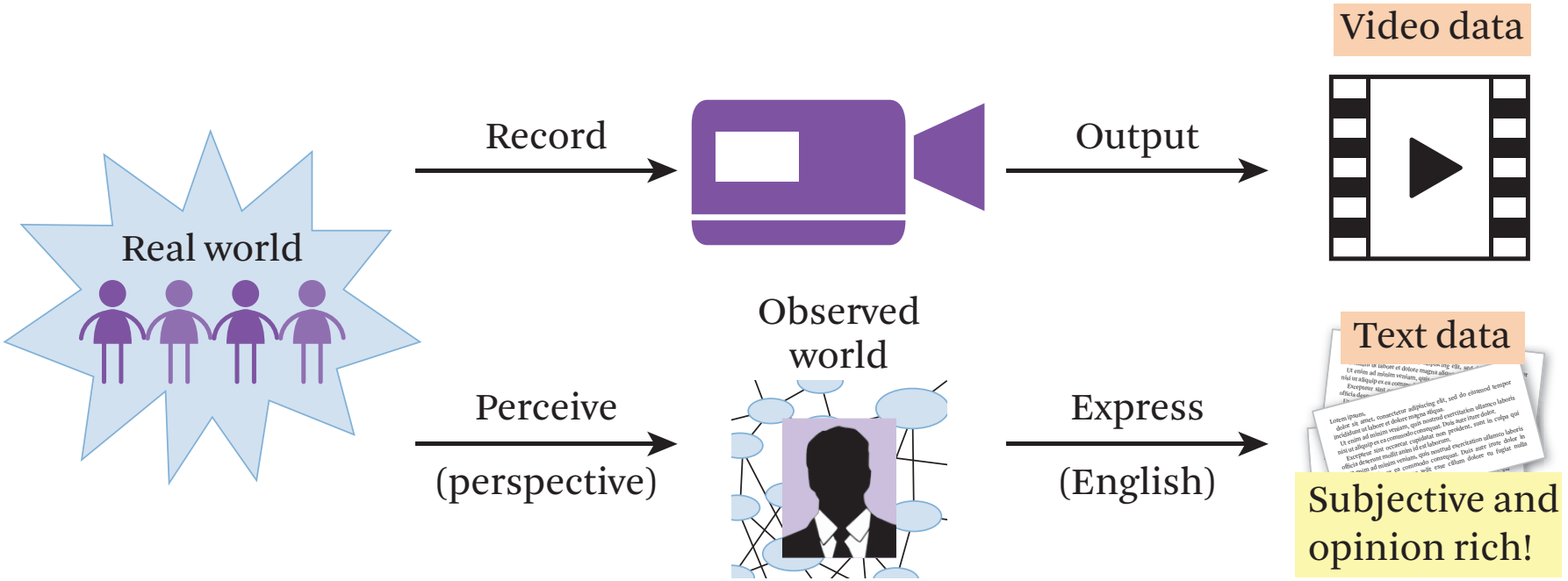
Week 7
Sentiment Analysis, Topic Detection

Reference and Slide Source: ChengXiang Zhai and Sean Massung. 2016. Text Data Management and Analysis: a Practical Introduction to Information Retrieval and Text Mining. Association for Computing Machinery and Morgan & Claypool, New York, NY, USA.


Sensory data
(Video, Audio, etc.)
Web data (Text)
(OSN, News, etc.)
NEWS
Multimedia
Systems
User data
(User attributes, Preferences, etc.)

Opinions

Text is generated by humans, therefore rich in subjective information!



Applications

- Decision support – whether to go for the movie or not?
- Understanding human preferences
- Advertising
- Recommendations
- Business intelligence – product feature evaluation

What is an opinion?

A *subjective* statement describing what a person *thinks* or *believes* about *something*!

"Chomu is the best town in India"

- Something = Chomu
- Belief = Best town!

Naam Shabana Reviews

- Awesome, I saw twice.....
- Unbelievably disappointing
- A good movie, if seen baby previously then it must be watched, Role of Tapsee Pannu is really appreciable.
- Instead of Naam Shabana they would have made Sar Dabaana ya jo dikhaye uska gala dabana

Opinion is generally analyzed in terms of sentiment, e.g., positive, negative & neutral!

Sentiment analysis has many other names

- Opinion extraction
- Opinion mining
- Sentiment mining
- Subjectivity analysis

Opinion Mining Tasks

- Detecting opinion holder
- Detecting opinion target
- Detecting opinion sentiment

Sentiment Analysis

- Simplest task:
  - Is the attitude of this text positive or negative?
- More complex:
  - Rate the attitude of this text from 1 to 5
- Advanced:
  - Detect the target, source, or complex attitude types

Sentiment Analysis

Assumes all other parameters are knows!

E.g. in teaching feedback, students writing about instructor.

Feature Selection

Choose all words or only adjectives?

- Generally all words turns out to work better
- Experiments

Word occurrence may matter more than word frequency

- The occurrence of the word *fantastic* tells us a lot
- The fact that it occurs 5 times may not tell us much more

I didn’t like this movie

vs

I really like this movie

How to handle negation?

Add NOT_ to every word between negation and following punctuation:

didn’t like this movie , but I

didn’t NOT_like NOT_this NOT_movie but I

n-grams

- Character n-grams: sequences of n adjacent characters treated as a unit
- Word n-grams: sequences of n adjacent words treated as a unit
- n-grams generally move by 1 unit

Character n-grams example

- Text: “student”
- 2-grams
  - st
  - tu
  - ud
  - de
  - en
  - nt

Character n-grams example

- Text: “In this paper”
- 4-grams
  - In_t
  - n_th
  - _thi
  - this
  - his_
  - is_p
  - s_pa
  - _pap
  - pape
  - aper

Word n-grams example

- Text: “The cow jumps over the moon”
- 2-grams
  - the cow
  - cow jumps
  - jumps over
  - over the
  - the moon

n=1 - unigram

n=2 - bigram

n=3 - trigram

What happens to n-grams when when one character is misspelled in word?

Assignment Vs Asignment

n-grams are robust to grammatical errors!

Word Vs Character n-grams, which one is more sparse?

Character n-grams are much less in number than word n-grams.

Other n-grams

- n-grams of PoS Tags
  - E.g. adjective followed by a noun
- n-gram of word classes
  - place, names, people
- n-gram of word topics
  - politics, technical

Analysis Methods

Classification/Regression

- Support Vector Machines
- Naïve Bayse Classifier
- Logistic Regression
- Ordinal logistic regression

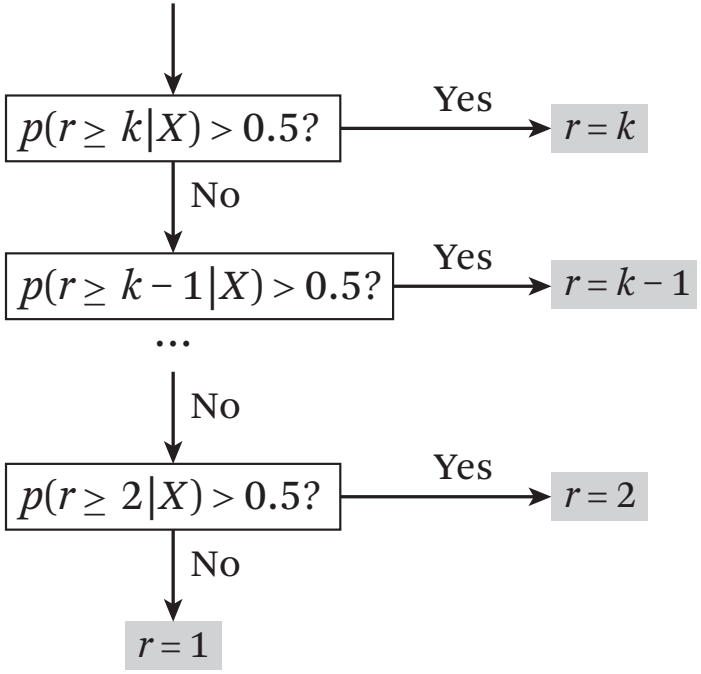
Logistic Regression for Binary Classification

$$\log \frac{p(y=1 \mid X)}{p(y=0 \mid X)} = \log \frac{p(y=1 \mid X)}{1 - p(y=1 \mid X)} = \alpha_j + \sum_{i=1}^{M} x_i \beta_i$$

Rating Detection

- Sentiment is generally measured on a scale from positive to negative with other labels in-between.
- Logistic regression is improvised to analyze multi-class classification, e.g. ratings from 1 to 5.

Multi-level Logistic Regression



Problems

- Too many parameters
- Classifiers are not not really independent

Ordinal Logistic Regression

- Assume **β** parameters to be the same for all classifiers
- Keel **α** different for each classifier
- Hence, M+K-1 parameters in total

$$\log\frac{p(Y_j=1|X)}{p(Y_j=0|X)}=\log\frac{p(r\geq j|X)}{1-p(r\geq j|X)}=\alpha_j+\sum_{i=1}^{M}x_i\beta_i \quad \beta_i\in\Re$$

More Sentiments

- Emotion – angry, sad, ashamed…
- Mood – cheerful, depressed, irritable...
- Interpersonal stances – friendly, cold...
- Attitudes – loving, hating...
- Personality traits – introvert, extrovert...

Topic detection: Given a text document, find main topic of the document cover?

1. I like to eat broccoli and bananas.
2. My sister adopted a kitten yesterday.
3. Look at this cute hamster munching on a piece of broccoli.

Topic detection has two main tasks

- Discover k topics covered across all documents
- Measure individual topic coverage by each document

Idea: choose individual terms as topics!

E.g. Sports, Travel, Food

Terms can be chosen based on TF or TF-IDF ranking!

- Problem: The top few terms can be similar or even synonym

- Solution: Go down the ranking and choose the terms that are different from the already chosen terms

Maximal Marginal Relevance (MMR)

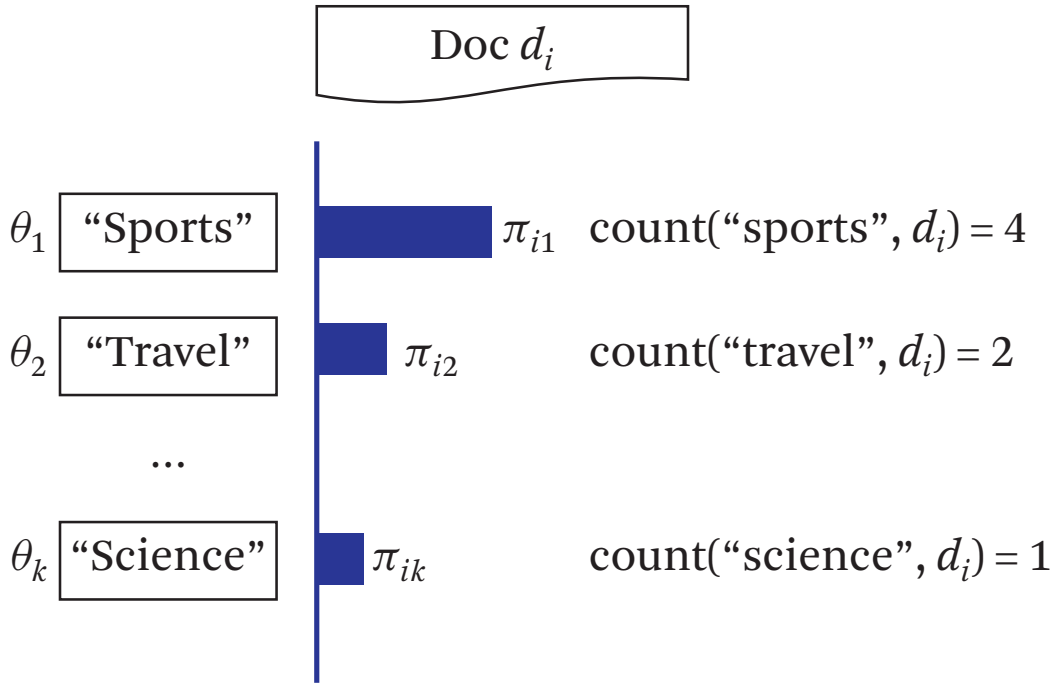
Maximal Marginal Relevance (MMR)

$$Score = \lambda \times \text{Re}levance - (1 - \lambda) \times \text{Re}dundancy$$

Example

- Relevance: Original ranking
- Redundancy: Similarity with already selected terms

Count the frequency of these terms in the document for coverage!

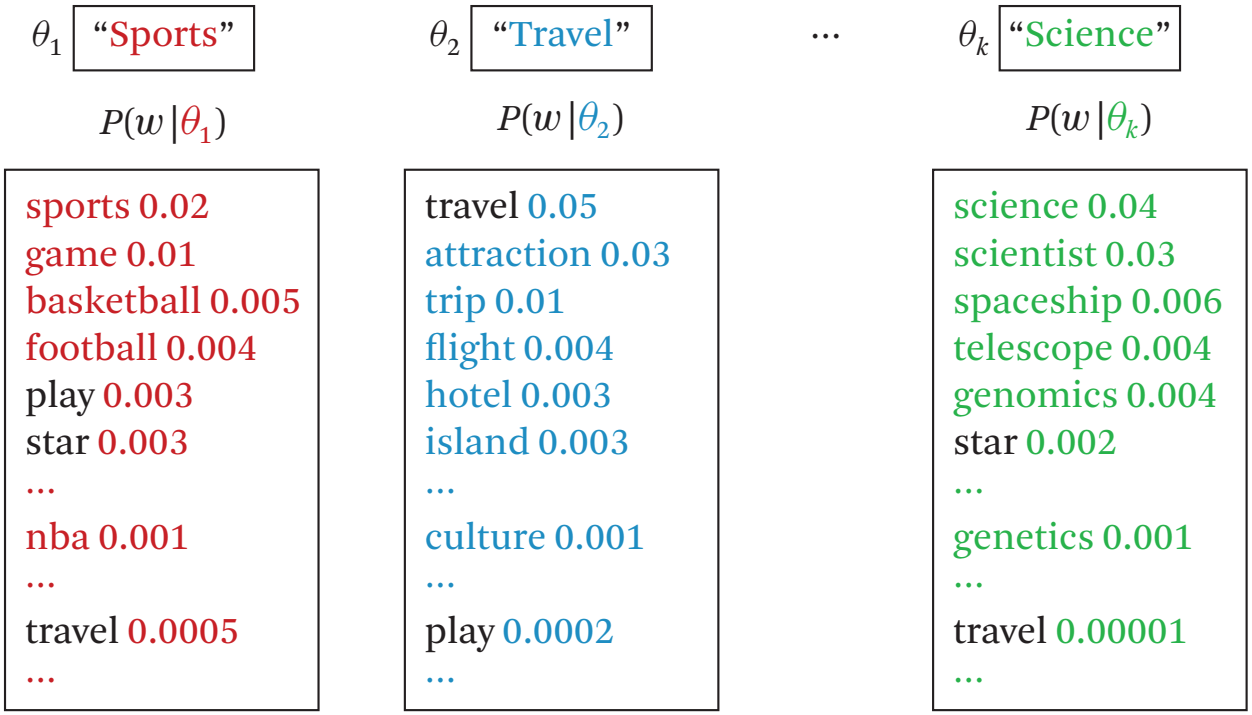


$$\pi_{ij} = \frac{\text{count}(\theta_j, d_i)}{\sum_{L=1}^{k} \text{count}(\theta_L, d_i)}$$

Limitations of Single Term as a topic

1. It is difficult which terms are “similar” while choosing a topic
2. Topics can be complicated, e.g. “Politics in Sports” or “Sports Injuries”
3. There can be multiple topics in a document, “Sports” and “Politics”
4. Ambiguous words

Solution: Model topics as word distribution!



$\sum_{w \in V} p(w|\theta_i) = 1$ Vocabulary set: $V = \{w_1, w_2, \ldots\}$

Refined Topic Modeling

- Input: A set of documents
- Output: output consists of two types of distributions
  - Word distribution foe each topic
  - Topic distribution for each document

Two Distributions

- Word distribution for topic i – global across all documents

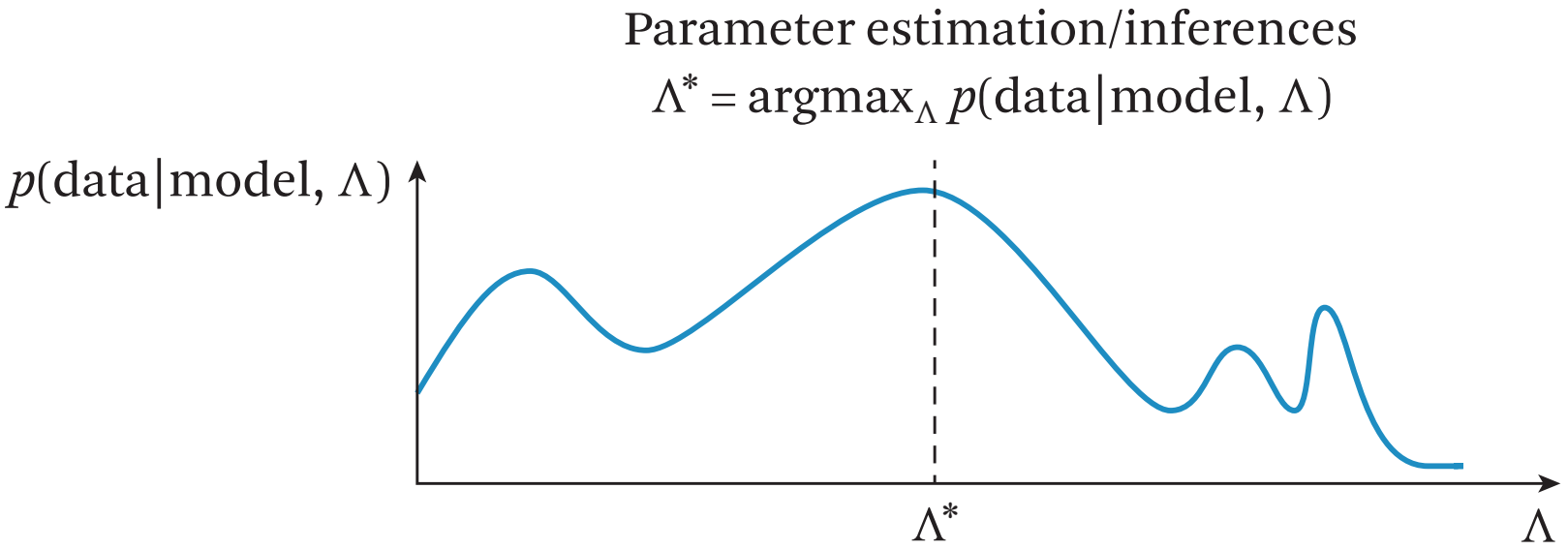
$$\sum_{w \in V} p(w \mid \theta_i) = 1.$$

- Topic distribution for document i – local to a document

$$\sum_{j=1}^{k} \pi_{ij} = 1, \forall i\,.$$

How to obtain the the probability distributions from the input data?

Maximum Likelihood Estimate of a Generative Model



Choose a model that has maximum probability of producing the training data.

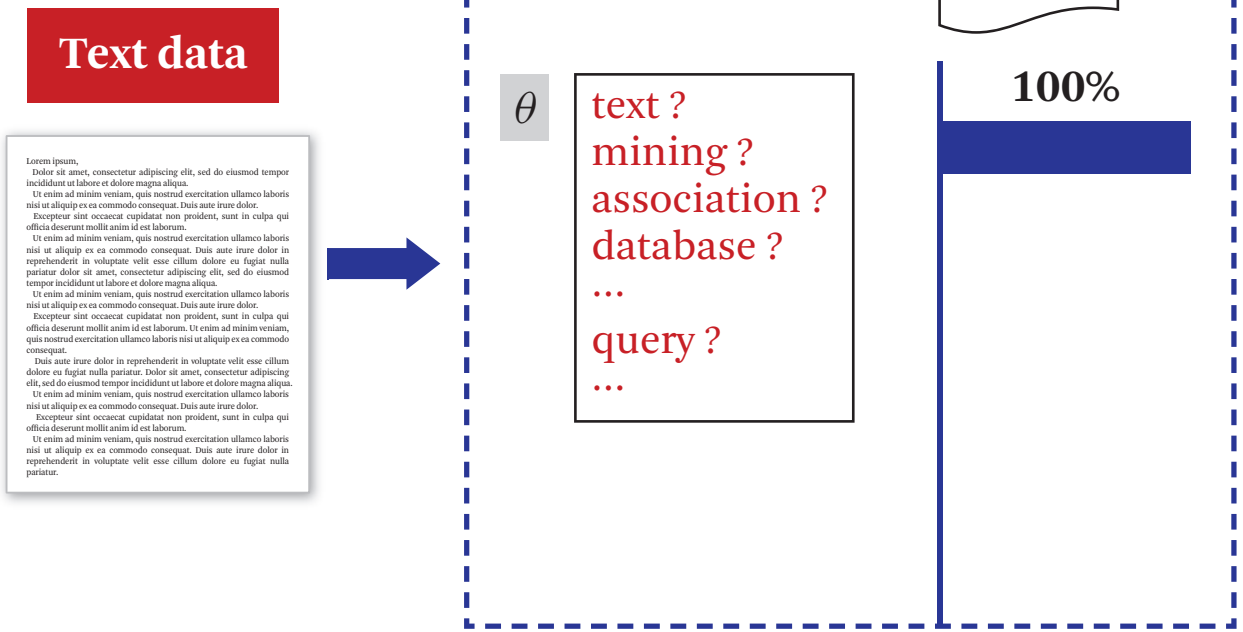
Advantages

- Multiple words allow us to describe fairly complicated topics.
- The term weights model subtle differences of semantics in related topics.
- Because we have probabilities for the same word in different topics, we can accommodate multiple senses of a word, addressing the issue of word ambiguity.

Mining One Topic

Input: $C = \{d\}, V$

Output: $\{\theta\}$

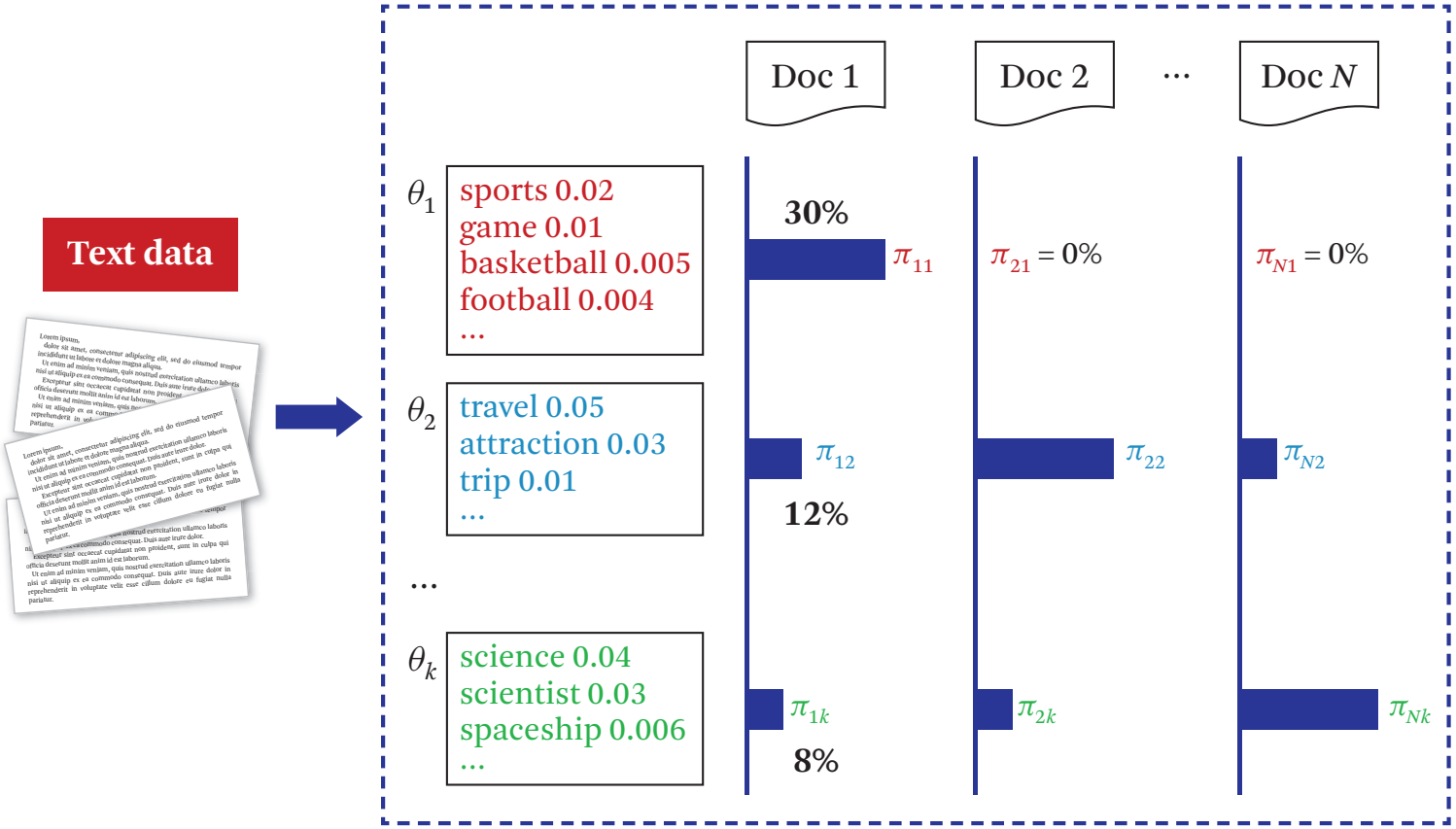


Mining One Topic

$$p(w_i \mid \theta) = \frac{c(w_i, d)}{|d|}$$

i =1 to M where M is the number of words in the dictionary!

Mining Multiple Topics



How to Compress Text?

The computer science and engineering department offers many computer related courses. Although there is science term in the name, no science courses are offered in the department.

Run Length Coding?

The computer science and engineering department offers many computer related courses. Although there is science term in the name, no science courses are offered in the department.

Not a good idea!

Differential Coding?

The computer science and engineering department offers many computer related courses. Although there is science term in the name, no science courses are offered in the department.

Not a good idea!

Huffman Coding?

The computer science and engineering department offers many computer related courses. Although there is science term in the name, no science courses are offered in the department.

Huffman Coding

- Removes the source information
- Frequent symbols assigned shorter codes
- Prefix coding

Can we refer to a dictionary entry of the word?

Dictionary-based Compression

- Do not encode individual symbols as variable-length bit strings
- Encode variable-length string of symbols as single token
- The tokens form an index into a phrase dictionary
- If number of tokens are smaller than number of phrases, we have compression

Lempel-Ziv-Welsh (LZW) Coding

- A dictionary-based coding algorithm
- Build the dictionary dynamically
- Initially the dictionary contains only character codes
- The remaining entries in the dictionary are then build dynamically

LZW Compression

```
set w = NIL
    loop
        read a character k
        if wk exists in the dictionary
            w = wk
        else
            output the code for w
            add wk to the dictionary
            w=k
    endloop
```

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

Example

Input String: ^WED^WE^WEE^WEB^WET

w	k	Output	Index	Symbol
NIL	^			
^	W	^	256	^W
W	E	W	257	WE
E	D	E	258	ED
D	^	D	259	D^
^	W			
^W	E	256	260	^WE
E	^	E	261	E^
^	W			
^W	E			
^WE	E	260	262	^WEE
E	^			
E^	W	261	263	E^W
W	E			
WE	B	257	264	WEB
B	^	B	265	B^
^	W			
^W	E			
^WE	T	260	266	^WET
T	EOF	T		

```
set w = NIL
 loop
        read a character k
        if wk exists in the dictionary
                w = wk
        else
                output the code for w
                add wk to the dictionary
                w = k
endloop
```

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

LZW Decompression

read fixed length token k (code or char)

output k
w=k
loop

    read a fixed length token k

    entry = dictionary entry for k

    output entry
    add w + first char of entry to the dictionary

    w = entry

endloop

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

Example

Input String (to decode): ^WED<256>E<260><261><257>B<260>T

w	k	Output	Index	Symbol
	^	^		
^	W	W	256	^W
W	E	E	257	WE
E	D	D	258	ED
D	<256>	^W	259	D^
^W	E	E	260	^WE
E	<260>	^WE	261	E^
^WE	<261>	E^	262	^WEE
E^	<257>	WE	263	E^W
WE	B	B	264	WEB
B	<260>	^WE	265	B^
^WE	T	T	266	^WET

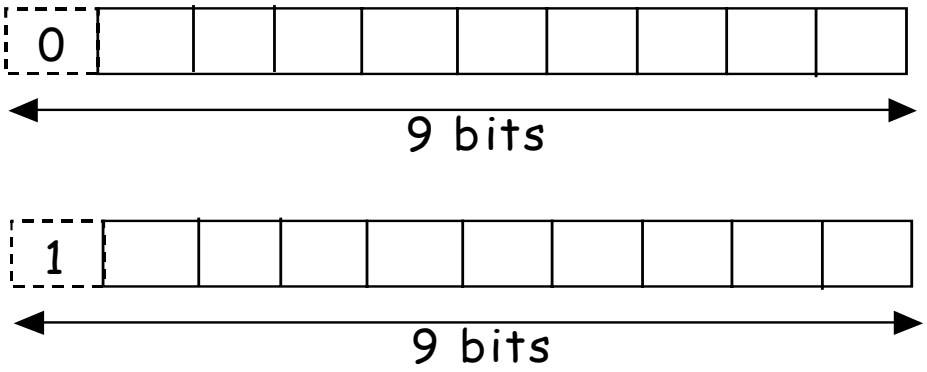
```
read a fixed length token k
      (code or char)
output k
w = k
loop
    read a fixed length token k
        (code or char)
    entry = dictionary entry for k
    output entry
    add w + first char of entry to
        the dictionary
    w = entry
endloop
```

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

Where is Compression?

- Input String: ^WED^WE^WEE^WEB^WET
  - 19 * 8 bits = 152 bits
- Encoded: ^WED<256>E<260><261><257>B<260>T
  - 12*9 bits = 108 bits (7 symbols and 5 codes, each of 9 bits)
- Why 9 bits?

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

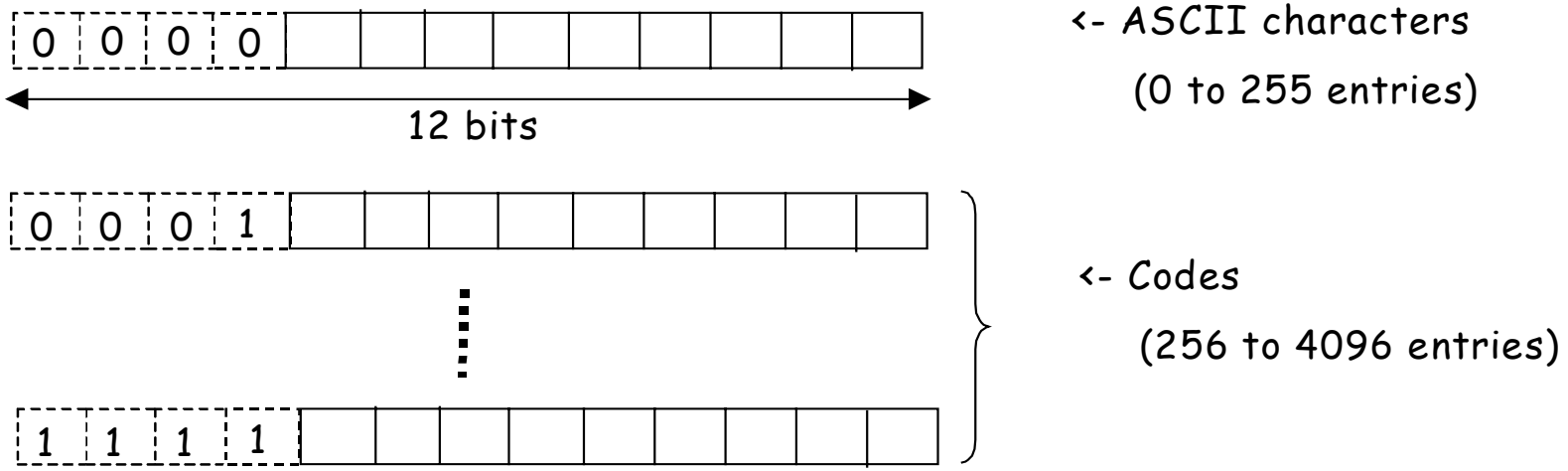


<- ASCII characters
(0 to 255)

<- Codes
(256 to 512)

Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

Original LZW Uses 12-bit Codes!



Ref: http://users.ece.utexas.edu/~ryerraballi/MSB/pdfs/M2L3.pdf

What if we run out of codes?

- Flush the dictionary periodically
- Or Grow length of codes dynamically