Topic Modelling

COMP90042

Natural Language Processing

Lecture 20

Semester 1 2022 Week 10
Jey Han Lau

Making Sense of Text

- English Wikipedia: 6M articles
- Twitter: 500M tweets per day
- New York Times: 15M articles
- arXiv: 1M articles
- What can we do if we want to learn something about these document collections?

Questions

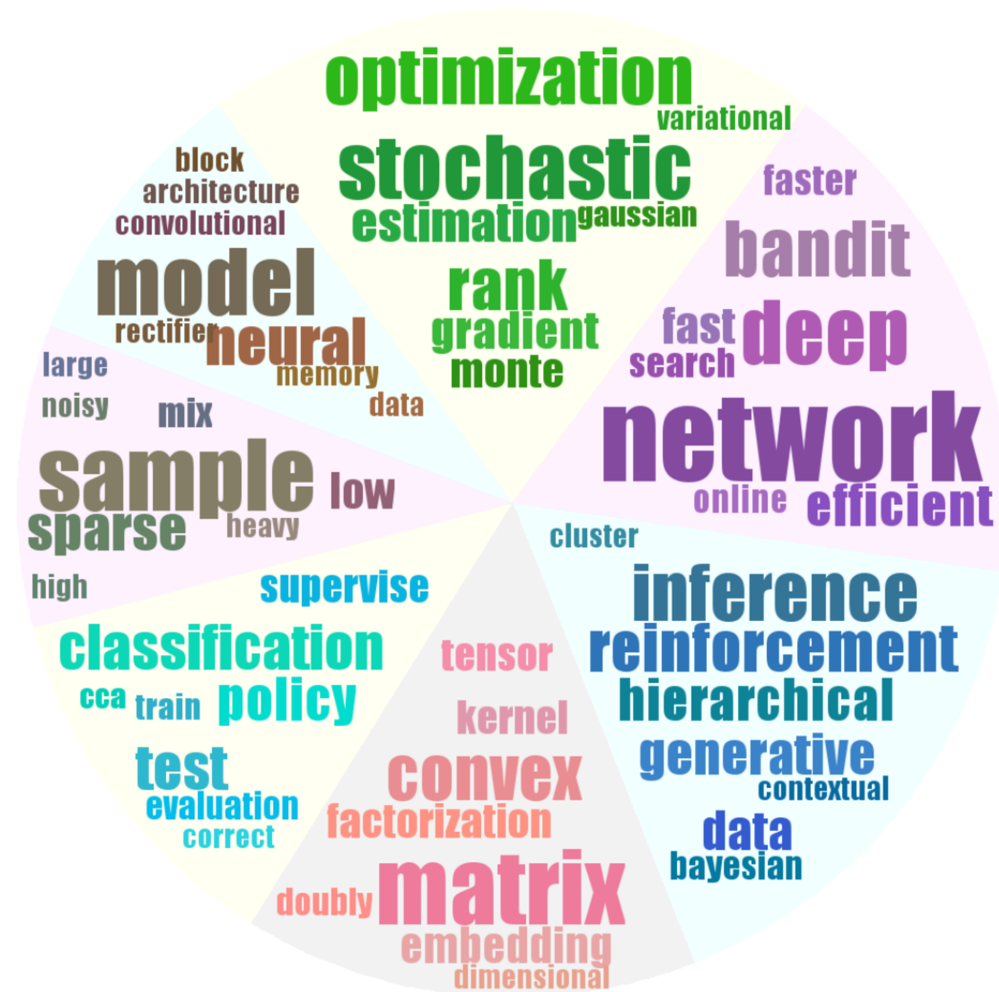
- What are the less popular topics on Wikipedia?
- What are the big trends on Twitter in the past month?
- How do the themes/topics evolve over time in New York Times from 1900s to 2000s?
- What are some influential research areas?

Topic Models To The Rescue

- Topic models learn common, overlapping themes in a document collection
- Unsupervised model
  - No labels; input is just the documents!
- What's the output of a topic model?
  - Topics: each topic associated with a list of words
  - Topic assignments: each document associated with a list of topics

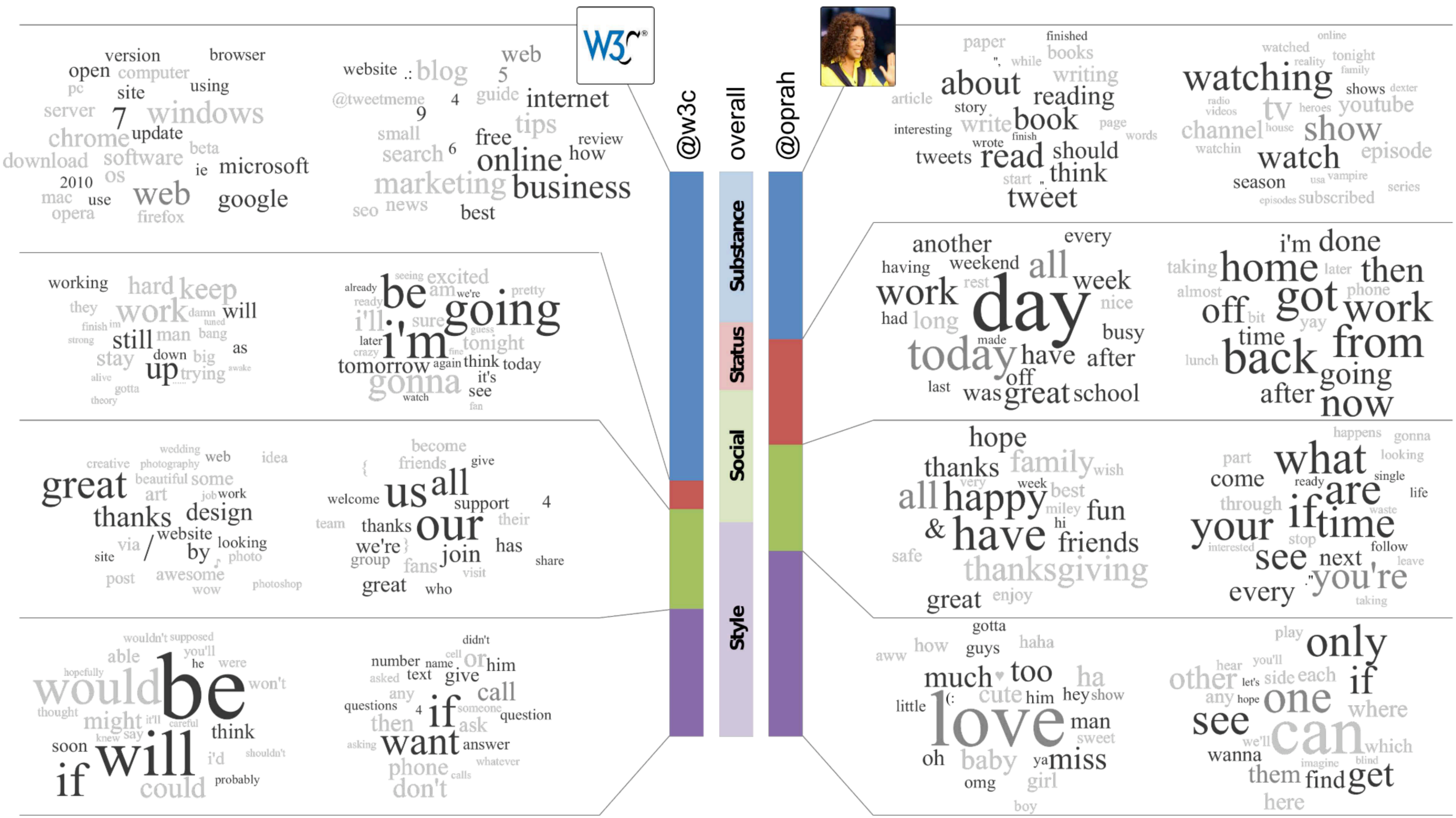
What Do Topics Look Like?

- A list of words
- Collectively describes a concept or subject
- Words of a topic typically appear in the same set of documents in the corpus



Wikipedia Topics

Twitter Topics



Ramage, Dumais, Liebling, 2010

New York Times Topics

music	book	art	game	show
band	life	museum	knicks	film
songs	novel	show	nets	television
rock	story	exhibition	points	movie
album	books	artist	team	series
jazz	man	artists	season	says
pop	stories	paintings	play	life
song	love	painting	games	man
singer	children	century	night	character
night	family	works	coach	know

theater	clinton	stock	restaurant	budget
play	bush	market	sauce	tax
production	campaign	percent	menu	governor
show	gore	fund	food	county
stage	political	investors	dishes	mayor
street	republican	funds	street	billion
broadway	dole	companies	dining	taxes
director	presidential	stocks	dinner	plan
musical	senator	investment	chicken	legislature
directed	house	trading	served	fiscal

http://journalofdigitalhumanities.org/2-1/topic-modeling-and-digital-humanities-by-david-m-blei/

Applications of topic models?

- Personalised advertising
- Search engine
- Discover senses of polysemous words
- Part-of-speech tagging

Outline

- A Brief History of Topic Models
- Latent Dirichlet Allocation
- Evaluation

A Brief History of Topic Models

Latent Semantic Analysis (L10)

$$A \quad \begin{array}{cc} & |D| \\ |V| & \begin{bmatrix} 0 & 1 & 0 & & 0 \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \end{array}$$

$$A = U\Sigma V^T$$

We use U to get word embeddings in L10

We didn't do anything with V

$$U \quad \begin{array}{cc} & m \\ |V| & \begin{bmatrix} 2.2 & 0.3 & & 8.7 \\ 5.5 & -2.8 & \cdots & 0.1 \\ -1.3 & 3.7 & & 3.5 \\ \vdots & & \ddots & \vdots \\ 2.9 & -2.1 & \cdots & -1.9 \end{bmatrix} \end{array}$$

$$\Sigma \quad \begin{array}{cc} & m \\ m & \begin{bmatrix} 9.1 & 0 & 0 & & 0 \\ 0 & 4.4 & 0 & \cdots & 0 \\ 0 & 0 & 2.3 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0.1 \end{bmatrix} \end{array}$$

$$V^T \quad \begin{array}{cc} & |D| \\ m & \begin{bmatrix} -0.2 & 4.0 & & -1.3 \\ -4.1 & 0.6 & \cdots & -0.2 \\ 2.6 & 6.1 & & 1.4 \\ \vdots & & \ddots & \vdots \\ -1.9 & -1.8 & \cdots & 0.3 \end{bmatrix} \end{array}$$

$A = U\Sigma V^T$ LSA: Truncate

$$U$$

$$
\begin{array}{cc}
 & \begin{array}{cc} m & k \end{array} \\
|V| & \begin{bmatrix} 2.2 & 0.3 & & -2.4 & & 8.7 \\ 5.5 & -2.8 & \cdots & 1.1 & \cdots & 0.1 \\ -1.3 & 3.7 & & 4.7 & & 3.5 \\ \vdots & & \ddots & \vdots & \ddots & \vdots \\ 2.9 & -2.1 & \cdots & -3.3 & \cdots & -1.9 \end{bmatrix}
\end{array}
$$

$$U_k$$

Word Vector

$$
\begin{array}{cc}
 & k \\
|V| & \begin{bmatrix} 2.2 & 0.3 & & -2.4 \\ 5.5 & -2.8 & \cdots & 1.1 \\ -1.3 & 3.7 & & 4.7 \\ \vdots & & \ddots & \vdots \\ 2.9 & -2.1 & \cdots & -3.3 \end{bmatrix}
\end{array}
$$

Topic

topic 1 = {dog: 2.2, cat: 5.5, shadow: -1.3, …}

topic 2 = {dog:0.3, cat:-2.8, shadow: 3.7, …}

$$V^T$$

$$
\begin{array}{cc}
 & |D| \\
\begin{array}{c} m \\ \\ k \end{array} & \begin{bmatrix} -0.2 & 4.0 & & -1.3 \\ -4.1 & 0.6 & \cdots & -0.2 \\ 2.6 & 6.1 & & 1.4 \\ \vdots & & \ddots & \vdots \\ -1.9 & -1.8 & \cdots & 0.3 \end{bmatrix}
\end{array}
$$

$${V^T}_k$$

$$
\begin{array}{cc}
 & |D| \\
k & \begin{bmatrix} -0.2 & 4.0 & & -1.3 \\ -4.1 & 0.6 & \cdots & -0.2 \\ 2.6 & 6.1 & & 1.4 \\ \vdots & & \ddots & \vdots \end{bmatrix}
\end{array}
$$

Topic Assignment

doc 1 = {t1: -0.2, t2: -4.1, t3: 2.6, …}

doc2 = {t1: 4.0, t2: 0.6, t3: 6.1, …}

Issues

- Positive and negative values in the U and V^T
- Difficult to interpret

$$U_k$$

$$\begin{array}{cc} & k \\ |V| & \begin{bmatrix} 2.2 & 0.3 & & -2.4 \\ 5.5 & -2.8 & \cdots & 1.1 \\ -1.3 & 3.7 & & 4.7 \\ & \vdots & \ddots & \vdots \\ 2.9 & -2.1 & \cdots & -3.3 \end{bmatrix} \end{array}$$

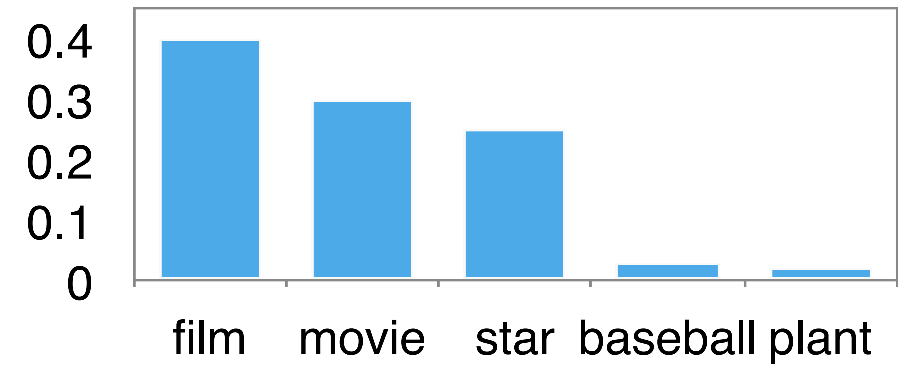
Topic

topic 1 = {dog: 2.2, cat: 5.5, shadow: -1.3, …}
topic 2 = {dog:0.3, cat:-2.8, shadow: 3.7, …}

Probabilistic LSA

- Based on a probabilistic model

Topic 1



Word distribution for a topic

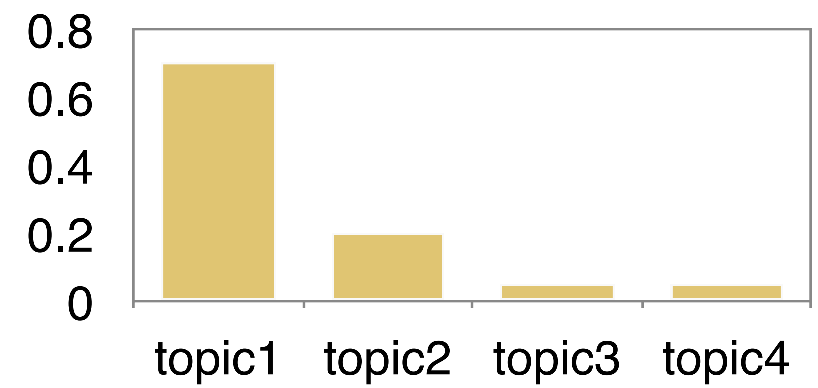
$$P(w,d) = P(w|d)P(d)$$
$$= P(d)\sum_{T} P(w|t)P(t|d)$$

Joint probability of a word and a document

Number of topics

Topic distribution for a document

Document 1



Issues

- No more negative values!
- PLSA can learn topics and topic assignment for documents in the train corpus
- But it is unable to infer topic distribution on **new documents**
- PLSA needs to be re-trained for new documents

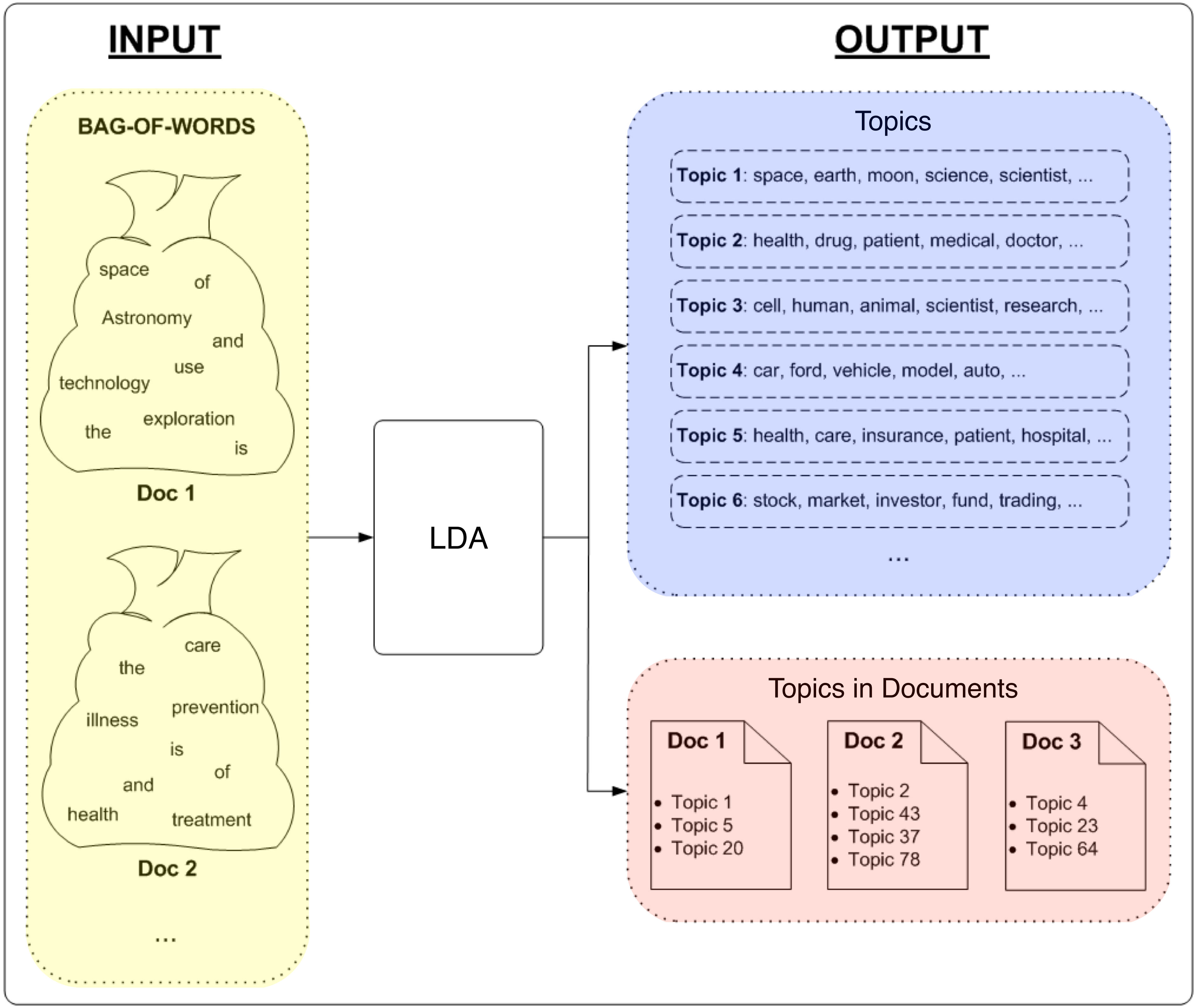
Latent Dirichlet Allocation

- Introduces a prior to the document-topic and topic-word distribution
- Fully generative: trained LDA model can infer topics on unseen documents!
- LDA is a Bayesian version of PLSA

Latent Dirichlet Allocation

LDA

- Core idea: assume each document contains **a mix of topics**
- But the topic structure is **hidden (latent)**
- LDA infers the topic structure given the observed words and documents
- LDA produces soft clusters of documents (based on topic overlap), rather than hard clusters
- Given a trained LDA model, it can infer topics on new documents (not part of train data)

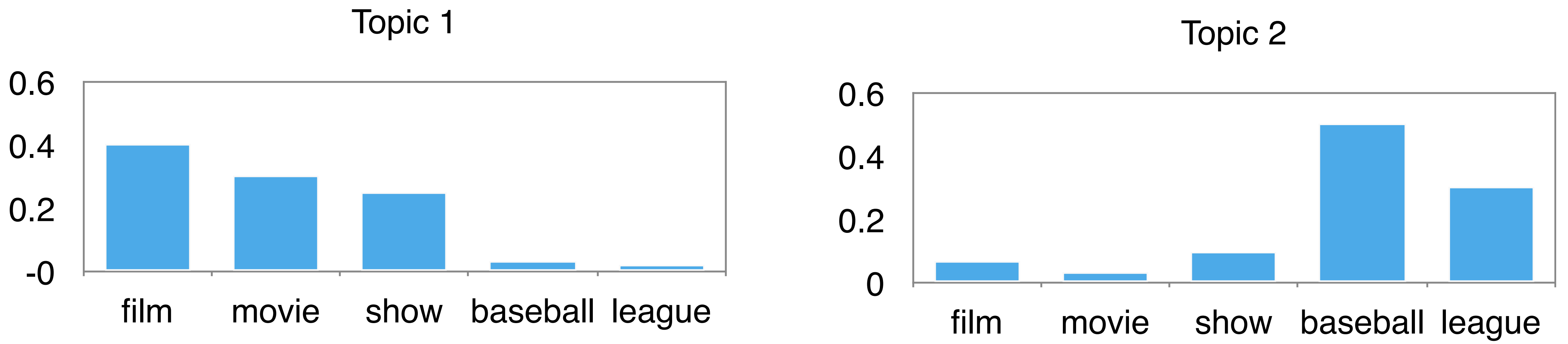

INPUT
OUTPUT
BAG-OF-WORDS
space
of
Astronomy
and
use
technology
exploration
the
is
Doc 1
care
the
prevention
illness
is
of
and
health
treatment
Doc 2
...
LDA
Topics
Topic 1: space, earth, moon, science, scientist, ...
Topic 2: health, drug, patient, medical, doctor, ...
Topic 3: cell, human, animal, scientist, research, ...
Topic 4: car, ford, vehicle, model, auto, ...
Topic 5: health, care, insurance, patient, hospital, ...
Topic 6: stock, market, investor, fund, trading, ...
...
Topics in Documents
Doc 1
Topic 1
Topic 5
Topic 20
Doc 2
Topic 2
Topic 43
Topic 37
Topic 78
Doc 3
Topic 4
Topic 23
Topic 64

Input

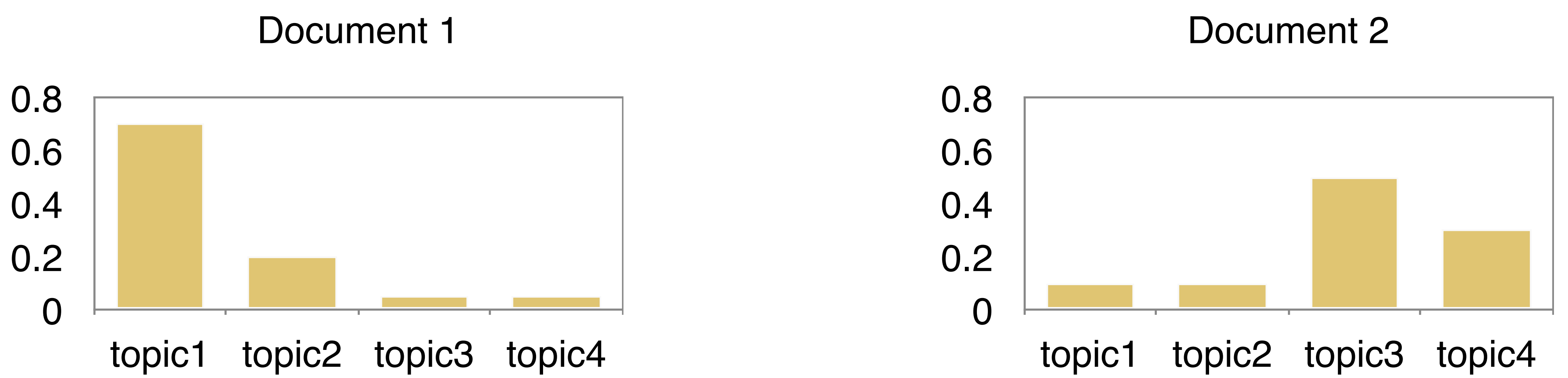
- A collection of documents
- Bag-of-words
- Good preprocessing practice:
  - Remove stopwords
  - Remove low and high frequency word types
  - Lemmatisation

Output

- **Topics:** distribution over words in each topic



- **Topic assignment:** distribution over topics in each document



Learning

- How do we learn the latent topics?
- Two main family of algorithms:
  - Variational methods
  - Sampling-based methods

Sampling Method (Gibbs)

1. Randomly assign topics to all tokens in documents

doc_1	mouse: t_1	cat: t_3	rat: t_2	chase: t_1	mouse: t_3
doc_2	scroll: t_1	mouse: t_3	scroll: t_3	scroll: t_2	click: t_2
doc_3	tonight: t_2	baseball: t_1	tv: t_2	exciting: t_1	

2. Collect **topic-word and document-topic co-occurrence statistics** based on the assignments

	mouse	cat	scroll	tv	…
t_1	0.01	0.01	0.01	0.01	
t_2	0.01	0.01	0.01	0.01	
t_3	0.01	0.01	0.01	0.01	

	t_1	t_2	t_3
d_1	0.1	0.1	0.1
d_2	0.1	0.1	0.1
…			

Initialise co-occurrence matrix with β (=0.01) and α (=0.1) priors

Sampling Method (Gibbs)

1. Randomly assign topics to all tokens in documents

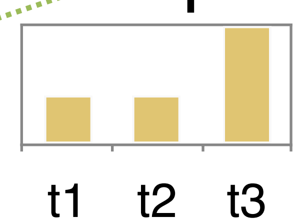
doc_1	mouse: **?**	cat: t_3	rat: t_2	chase: t_1	mouse: t_3
doc_2	scroll: t_1	mouse: t_3	scroll: t_3	scroll: t_2	click: t_2
doc_3	tonight: t_2	baseball: t_1	tv: t_2	exciting: t_1	

2. Collect **topic-word and document-topic co-occurrence statistics** based on the assignments

	mouse	cat	scroll	tv	…
t_1	1.01-1	0.01	1.01	0.01	
t_2	0.01	0.01	1.01	1.01	
t_3	2.01	1.01	1.01	0.01	

	t_1	t_2	t_3
d_1	2.1-1	1.1	2.1
d_2	1.1	2.1	2.1
…			

3. Go through every word token in corpus and sample a new topic:
   - $P(t_i \mid w, d) \propto P(t_i \mid w) P(t_i \mid d)$



Need to de-allocate the current topic assignment and update the co-occurrence matrices before sampling

4. Go to step 2 and repeat until convergence

$$P(t_1 \mid w, d) = P(t_1 \mid \text{mouse}) \times P(t_1 \mid d_1)$$

$$\frac{0.01}{0.01 + 0.01 + 2.01} \times \frac{1.1}{1.1 + 1.1 + 2.1}$$

Sampling Method (Gibbs)

1. Randomly assign topics to all tokens in documents

doc_1	mouse: ?	cat: t_3	rat: t_2	chase: t_1	mouse: t_3
doc_2	scroll: t_1	mouse: t_3	scroll: t_3	scroll: t_2	click: t_2
doc_3	tonight: t_2	baseball: t_1	tv: t_2	exciting: t_1	

2. Collect **topic-word and document-topic co-occurrence statistics** based on the assignments

	mouse	cat	scroll	tv	…
t_1	1.01-1	0.01	1.01	0.01	
t_2	0.01	0.01	1.01	1.01	
t_3	2.01	1.01	1.01	0.01	

	t_1	t_2	t_3
d_1	2.1-1	1.1	2.1
d_2	1.1	2.1	2.1
…			

3. Go through every word token in corpus and sample a new topic:
   - $P(t_i \mid w, d) \propto P(t_i \mid w) P(t_i \mid d)$

4. Go to step 2 and repeat until convergence

$$P(t_3 \mid w, d) = ?$$

$$\frac{2.01}{0.01 + 0.01 + 2.01} \times \frac{2.1}{1.1 + 1.1 + 2.1}$$

When Do We Stop?

- Train until convergence
- Convergence = model probability of training set becomes stable
- How to compute model probability?

$$\log P(w_1, w_2, \ldots, w_m) = \log \sum_{j=0}^{T} P(w_1 \mid t_j) P(t_j \mid d_{w_1}) + \ldots + \log \sum_{j=0}^{T} P(w_m \mid t_j) P(t_j \mid d_{w_m})$$

  - m = #word tokens

Based on the topic-word co-occurrence matrix

Based on the document-topic co-occurrence matrix

Infer Topics For New Documents

1. Randomly assign topics to all tokens in new/test documents

testdoc$_1$	tiger: t_2	cow: t_1	cat: t_3	tiger: t_3	
testdoc$_2$	football: t_2	live: t_2	men: t_2	fun: t_3	soccer: t_1
testdoc$_3$	news: t_1	cnn: t_3	tonight: t_1		

2. Update document-topic matrix based on the assignments; but use the trained topic-word matrix (kept fixed)

from trained topic model

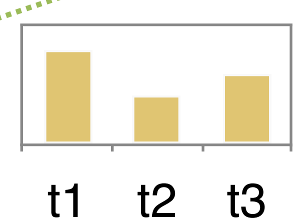
	mouse	cat	scroll	tv	…
t_1	5.01	4.01	1.01	0.01	
t_2	0.01	2.01	1.01	8.01	
t_3	0.01	0.01	4.01	0.01	

new matrix

	t_1	t_2	t_3
td_1	1.1	1.1	2.1
td_2	1.1	3.1	1.1
…			

3. Go through every word in the test documents and sample topics:

- $P(t_i \mid w, d) \propto P(t_i \mid w) P(t_i \mid d)$



4. Go to step 2 and repeat until convergence

Hyper-Parameters

- T: number of topic

Low T (<10): broad topics

High T (100+): fine-grained, specific topics

Hyper-Parameters

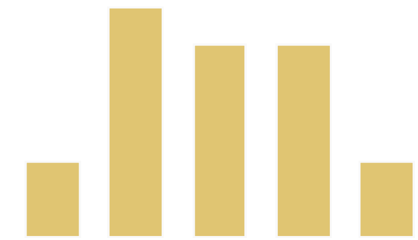
- β: prior on the topic-word distribution
- α: prior on the document-topic distribution
- Analogous to *k* in add-*k* smoothing in *N*-gram LM
- Pseudo counts to initialise co-occurrence matrix:
  - $p(w\,|\,t)$: β
  - $p(t\,|\,d)$: α

	mouse	cat	scroll	tv	...
t_1	0.01	0.01	0.01	0.01	
t_2	0.01	0.01	0.01	0.01	
t_3	0.01	0.01	0.01	0.01	

	t_1	t_2	t_3
d_1	0.1	0.1	0.1
d_2	0.1	0.1	0.1
...			

Hyper-Parameters

- High prior values → flatter distribution



  - a very very large value would lead to a uniform distribution

- Low prior values → peaky distribution
- β: generally small (< 0.01)
  - Large vocabulary, but we want each topic to focus on specific themes
- α: generally larger (> 0.1)
  - Multiple topics within a document

Evaluation

How To Evaluate Topic Models?

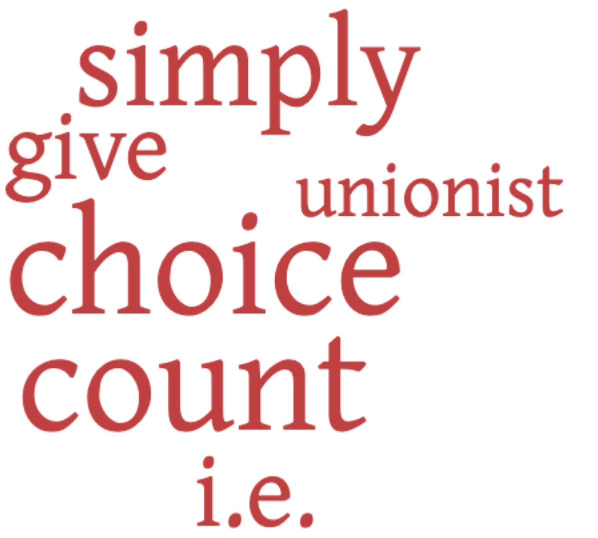
- Unsupervised learning → no labels
- Intrinsic evaluation:
  - model logprob / perplexity on test documents
  - $$\log L = \sum_{W} \sum_{T} \log P(w \mid t) P(t \mid d_w)$$
  - ppl $= \exp^{\frac{-\log L}{W}}$

Issues with Perplexity

- More topics = better (lower) perplexity
- Smaller vocabulary = better perplexity
  - Perplexity not comparable for different corpora, or different tokenisation/preprocessing methods
- Does not correlate with human perception of topic quality
- Extrinsic evaluation the way to go:
  - Evaluate topic models based on downstream task

Topic Coherence

- A better intrinsic evaluation method
- Measure how **coherent** the generated topics



- A good topic model is one that generates more coherent topics

Word Intrusion

- Idea: inject one random word to a topic

{farmers, farm, food, rice, agriculture}

↓

{farmers, farm, food, rice, cat, agriculture}

- Ask users to guess which is the **intruder word**
- Correct guess → topic is coherent
- Try guess the intruder word in:
  - {choice, count, village, i.e., simply, unionist}
- Manual effort; does not scale

PMI ≈ Coherence?

- High PMI for a pair of words → words are correlated
  - PMI(farm, rice) ↑
  - PMI(choice, village) ↓
- If all word pairs in a topic has high PMI → topic is coherent
- If most topics have high PMI → good topic model
- Where to get word co-occurrence statistics for PMI?
  - Can use same corpus for topic model
  - A better way is to use an external corpus (e.g. Wikipedia)

PMI

- Compute pairwise PMI of top-*N* words in a topic

$$\text{PMI}(t) = \sum_{j=2}^{N} \sum_{i=1}^{j-1} \log \frac{P(w_i, w_j)}{P(w_i)P(w_j)}$$

- Given topic: {farmers, farm, food, rice, agriculture}
- **Coherence** = sum PMI for all word pairs:
  - PMI(farmers, farm) + PMI(farmers, food) + … + PMI(rice, agriculture)

Variants

- Normalised PMI

$$\text{NPMI}(t) = \sum_{j=2}^{N} \sum_{i=1}^{j-1} \frac{\log \frac{P(w_i, w_j)}{P(w_i)P(w_j)}}{-\log P(w_i, w_j)}$$

- Conditional probability

$$\text{LCP}(t) = \sum_{j=2}^{N} \sum_{i=1}^{j-1} \log \frac{P(w_i, w_j)}{P(w_i)}$$

PMI Examples

Topic	PMI	NPMI
cell hormone insulin muscle receptor	0.61	0.59
electron laser magnetic voltage wavelength	0.54	0.52
magnetic neutrino particle quantum universe	0.55	0.55
album band music release song	0.37	0.56
college education school student university	0.38	0.57
city county district population town	0.34	0.52

A Final Word

- Topic model: an unsupervised model for learning latent concepts in a document collection
- LDA: a popular topic model
  - Learning
  - Hyper-parameters
- How to evaluate topic models?
  - Topic coherence