Topic models, vector semantics and applications

Vasileios Lampos

Computer Science, UCL

@lampos

In this lecture...

- **Topic models**
  — Latent Semantic Analysis (LSA)
  — Probabilistic Latent Semantic Analysis (pLSA)
  — Latent Dirichlet Allocation (LDA)

- **Vector semantics**
  — Early approaches (sparse)
  — Dense vector semantics (word embeddings) including word2vec

- **Applications**
  — Predicting judicial decisions
  — Improving the accuracy of disease models from Web searches
  — Inferring the occupational class of a Twitter user

Material

Book chapters

— Jurafsky and Martin. Speech and Language Processing (ed. 2017; draft). Chapters 15 and 16, web.stanford.edu/~jurafsky/slp3/

Papers

— pLSA (Hofmann), http://cis.csuohio.edu/~sschung/CIS660/PLSIHoffman.pdf

— LDA (Blei, Ng and Jordan), jmlr.org/papers/volume3/blei03a/blei03a.pdf

— word2vec (Mikolov et al.), papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf

Videos

— Blei on LDA, videolectures.net/mlss09uk_blei_tm/

— Boyd-Graber on topic models, youtube.com/watch?v=yK7nN3FcgUs

— Manning on word2vec, youtube.com/watch?v=ERibwqs9p38

Other

— Slides from WSDM 2014 tutorial on "Multilingual Probabilistic Topic Modelling", liir.cs.kuleuven.be/tutorial/WSDM2014Tutorial.pdf

Main software libraries

— MALLET (Java), http://mallet.cs.umass.edu/

— gensim (Python), github.com/RaRe-Technologies/gensim

Part I

What is a topic model?

- **Informally: ?**

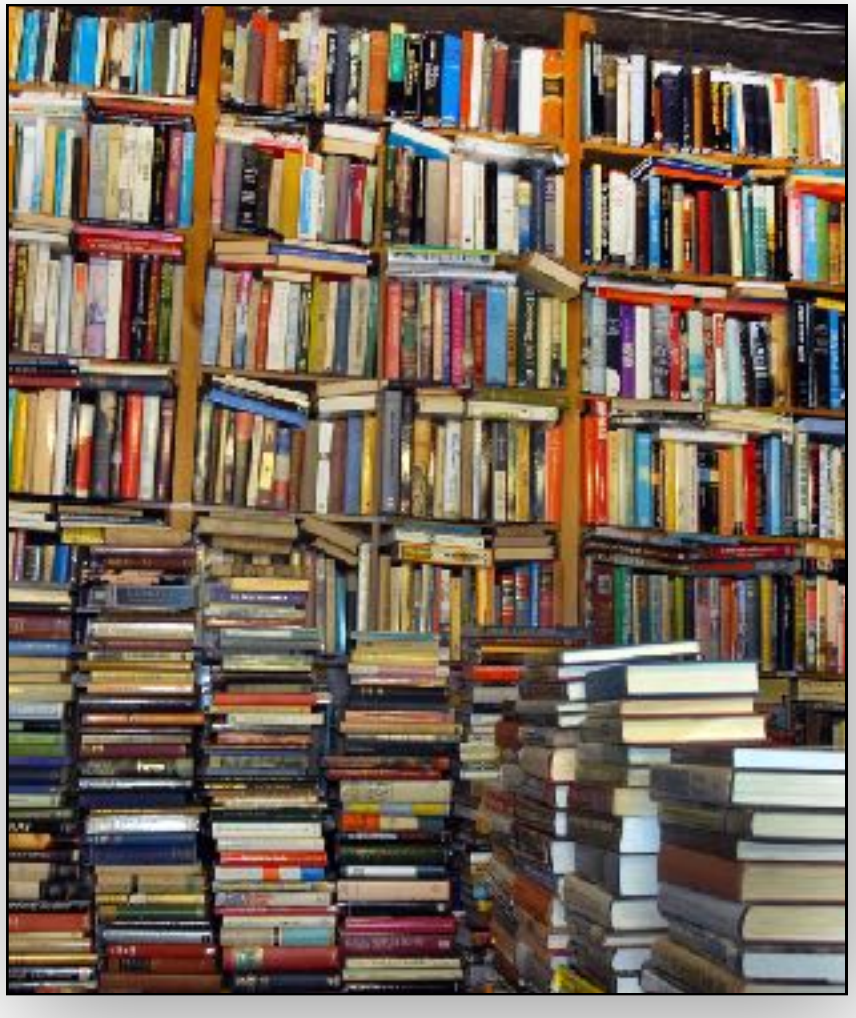
What is a topic model?

- **Informally:** group of words that are somehow related
- **Still informally:** method for automatically organising, understanding, searching, and summarising large (digitised) document collections
  — uncovers hidden (latent) topical patterns (topics!) in the collection
  — can annotate (and then organise or summarise) the documents based on these topics
- As we will see, it is just a **probabilistic structure** expressing a certain set of assumptions about how the documents in our collection were generated

What is a topic model?

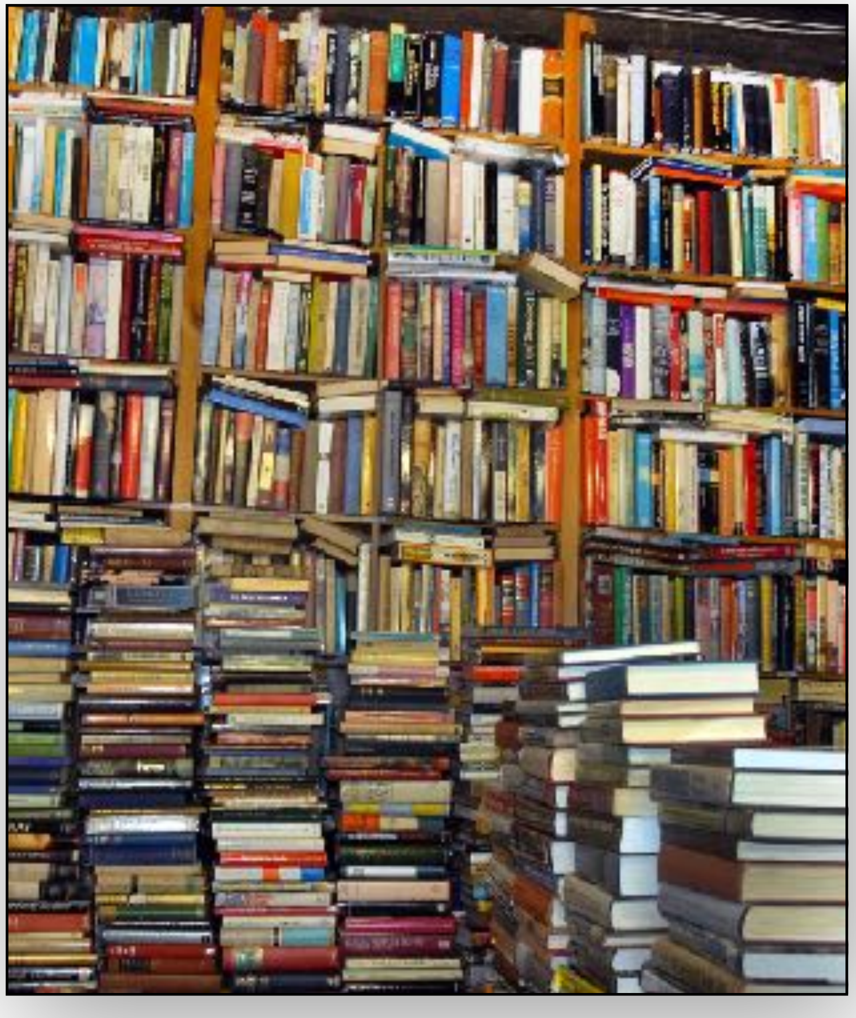
- **Informally:** group of words that are somehow related
- **Still informally:** method for automatically organising, understanding, searching, and summarising large (digitised) document collections
  — uncovers hidden (latent) topical patterns (topics!) in the collection
  — can annotate (and then organise or summarise) the documents based on these topics
- As we will see, it is just a **probabilistic structure** expressing a certain set of assumptions about how the documents in our collection were generated
- **Note:** we can derive topic models (word clusters) using clustering techniques with no explicit probabilistic structure

Why do we need topics?



- Too **many documents** and we can't read them all!
- Topic models can automatically categorise large document collections, so that we can browse through them much more efficiently
- Applicable on various corpus collections attracting multi-disciplinary interest (newspapers, books, social media, health reports, ...)
- Can **improve natural language processing tasks** (machine translation, word sense disambiguation, ...)
- Can **improve downstream tasks** in text mining

Why do we need topics?



- Too **many documents** and we can't read them all!
- Topic models can automatically categorise large document collections, so that we can browse through them much more efficiently
- Applicable on various corpus collections attracting multi-disciplinary interest (newspapers, books, social media, health reports, ...)
- Can **improve natural language processing tasks** (machine translation, word sense disambiguation, ...)
- Can **improve downstream tasks** in text mining
- Let's see a few examples

Topics in news articles

"Arts"	"Budgets"	"Children"	"Education"
NEW	MILLION	CHILDREN	SCHOOL
FILM	TAX	WOMEN	STUDENTS
SHOW	PROGRAM	PEOPLE	SCHOOLS
MUSIC	BUDGET	CHILD	EDUCATION
MOVIE	BILLION	YEARS	TEACHERS
PLAY	FEDERAL	FAMILIES	HIGH
MUSICAL	YEAR	WORK	PUBLIC
BEST	SPENDING	PARENTS	TEACHER
ACTOR	NEW	SAYS	BENNETT
FIRST	STATE	FAMILY	MANIGAT
YORK	PLAN	WELFARE	NAMPHY
OPERA	MONEY	MEN	STATE
THEATER	PROGRAMS	PERCENT	PRESIDENT
ACTRESS	GOVERNMENT	CARE	ELEMENTARY
LOVE	CONGRESS	LIFE	HAITI

The William Randolph Hearst Foundation will give $1.25 million to Lincoln Center, Metropolitan Opera Co., New York Philharmonic and Juilliard School. "Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services," Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. Lincoln Center's share will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The Juilliard School, where music and the performing arts are taught, will get $250,000. The Hearst Foundation, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual annual $100,000 donation, too.

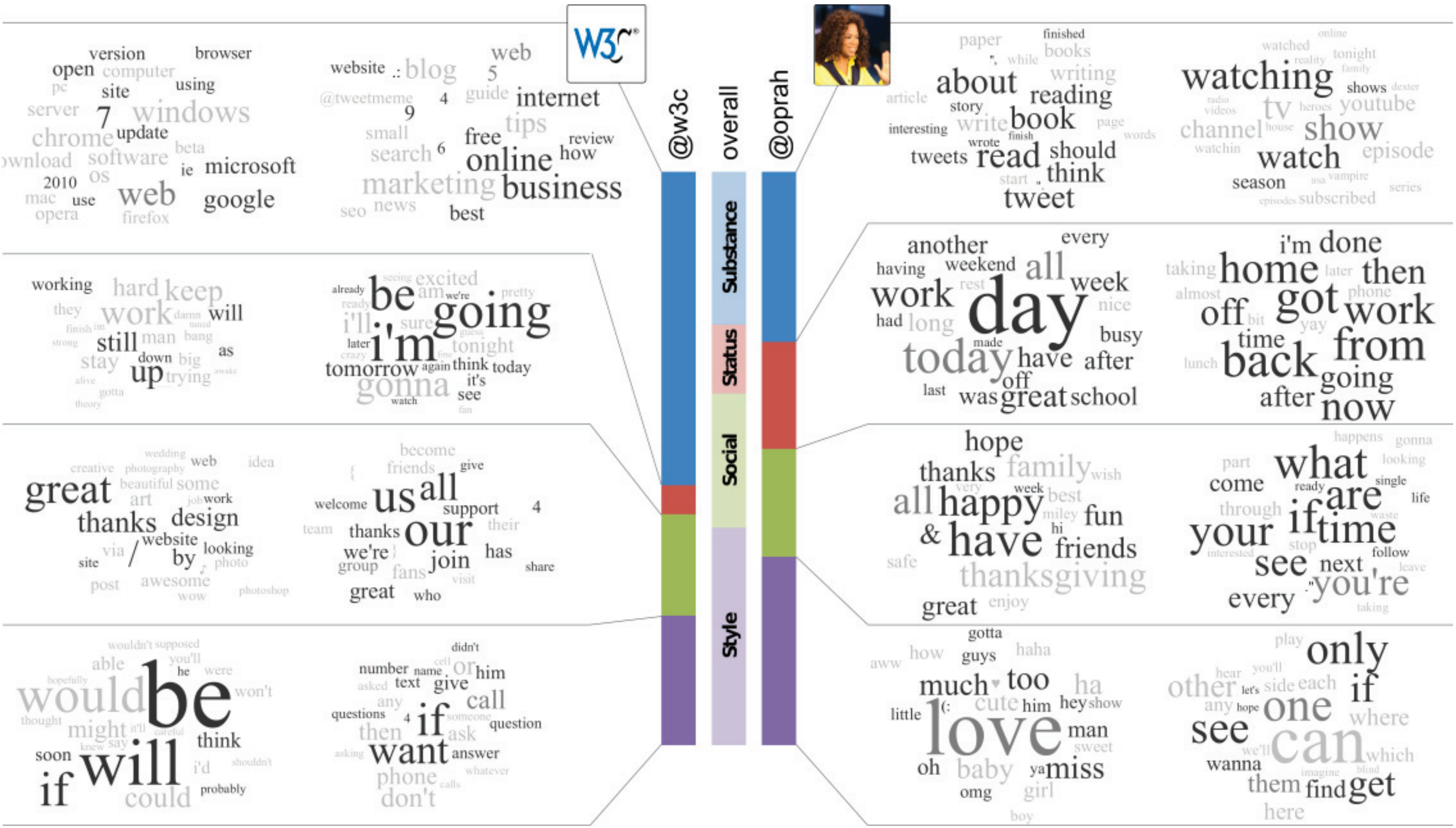
Blei, Ng & Jordan. JMLR, 2003. jmlr.org/papers/volume3/blei03a/blei03a.pdf

17K articles from the journal "Science"

"Genetics"	"Evolution"	"Disease"	"Computers"
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Blei. CACM, 2012. doi.org/10.1145/2133806.2133826

Characterising Twitter users

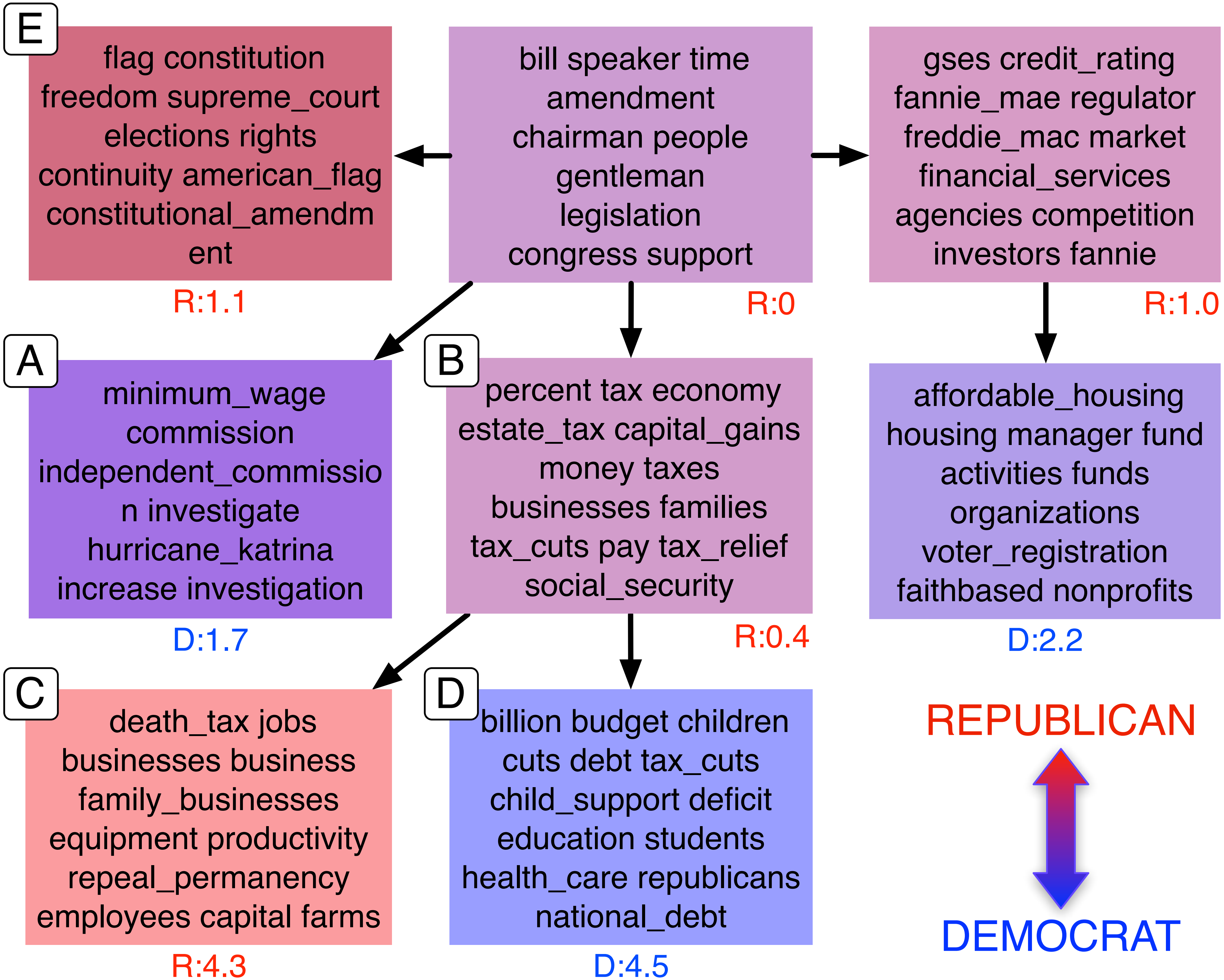


Ramage et al. ICWSM, 2010. aaai.org/ocs/index.php/ICWSM/ICWSM10/paper/view/1528

Age-group discriminants on Facebook

Schwartz et al. PLOS ONE, 2013. doi.org/10.1371/journal.pone.0073791

Congressional floor debates



Nguyen et al. NIPS, 2013. papers.nips.cc/paper/5163-lexical-and-hierarchical-topic-regression.pdf

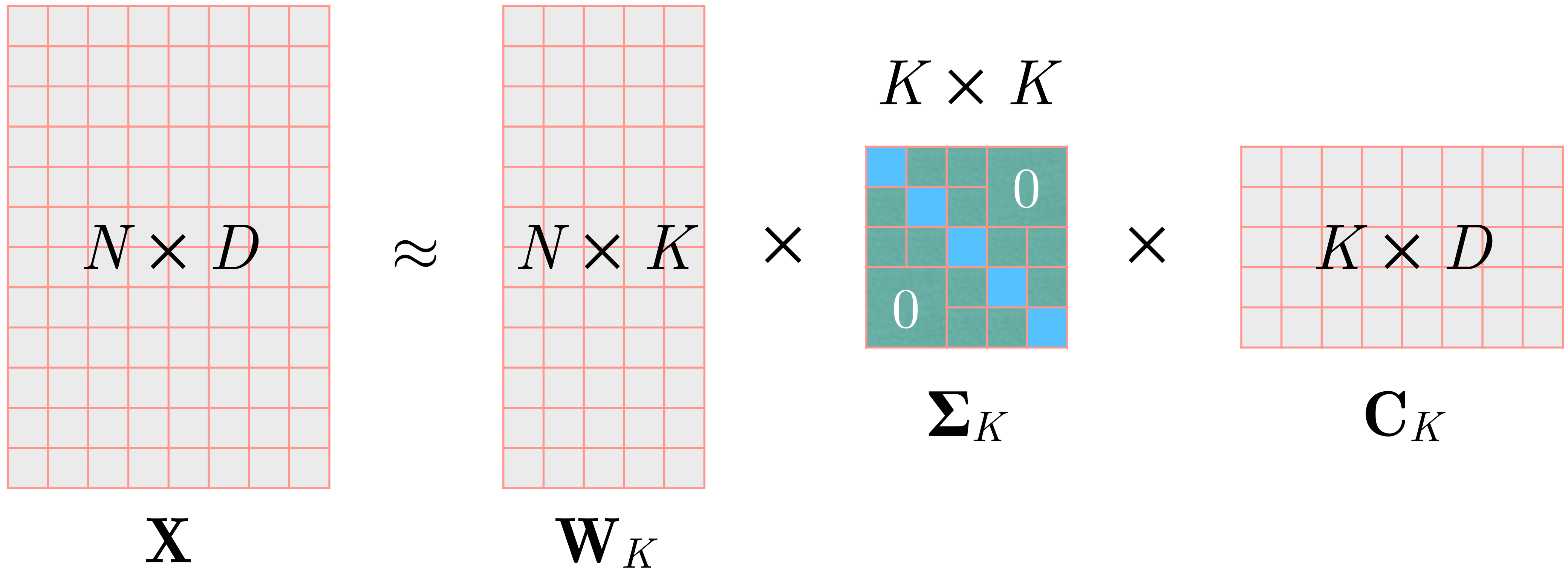
Predicting judicial decisions

Violation of Article 3 that prohibits inhuman treatment

Label	Words	*w*
	Top-5 Violation	
Positive State Obligations	injury, protection, ordered, damage, civil, caused, failed, claim, course, connection, region, effective, quashed, claimed, suffered, suspended, carry, compensation, pecuniary, ukraine	13.50
Detention conditions	prison, detainee, visit, well, regard, cpt, access, food, situation, problem, remained, living, support, visited, establishment, standard, admissibility merit, overcrowding, contact, good	11.70
Treatment by state officials	police, officer, treatment, police officer, July, ill, force, evidence, ill treatment, arrest, allegation, police station, subjected, arrested, brought, subsequently, allegedly, ten, treated, beaten	10.20
	Top-5 No Violation	
Prior Violation of Article 2	june, statement, three, dated, car, area, jurisdiction, gendarmerie, perpetrator, scene, June applicant, killing, prepared, bullet, wall, weapon, kidnapping, dated June, report dated, stopped	−12.40
Issues of Proof	witness, asked, told, incident, brother, heard, submission, arrived, identity, hand, killed, called, involved, started, entered, find, policeman, returned, father, explained	−15.20
Sentencing	sentence, year, life, circumstance, imprisonment, release, set, president, administration, sentenced, term, constitutional, federal, appealed, twenty, convicted, continued, regime, subject, responsible	−17.40

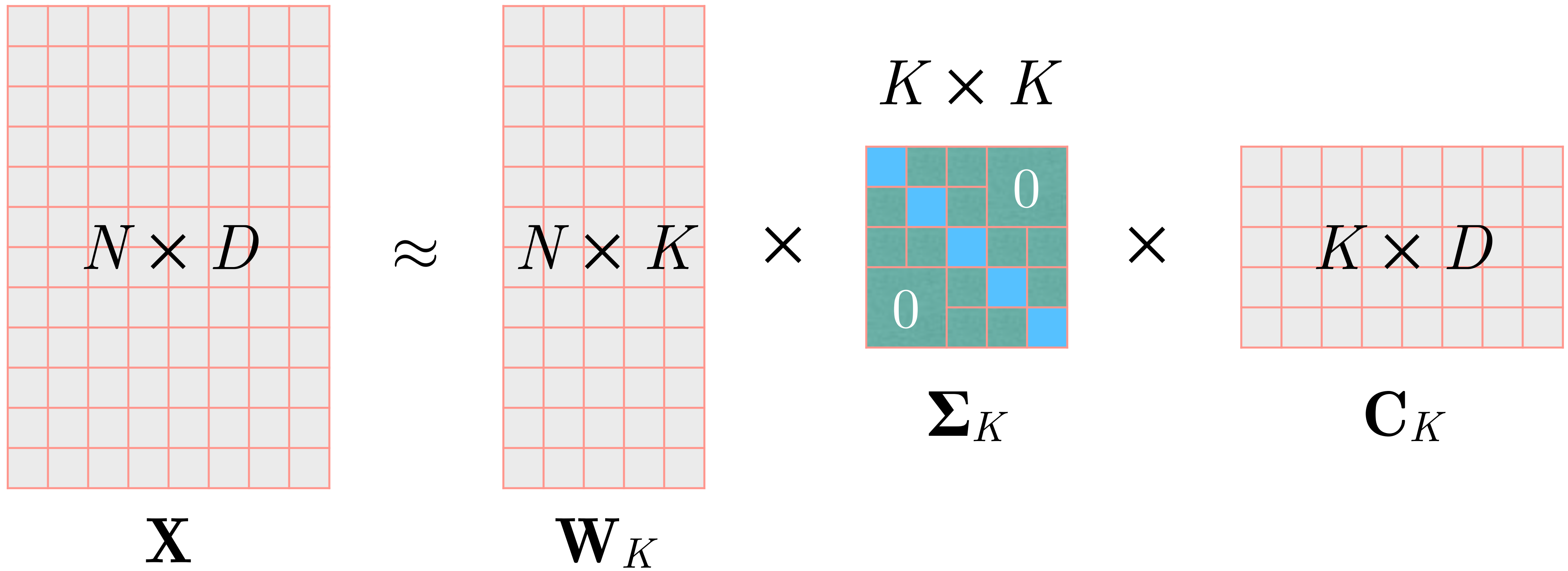
Aletras et al. PeerJ Computer Science, 2016. doi.org/10.7717/peerj-cs.93

Latent Semantic Analysis (or Indexing) — LSA



Singular Value Decomposition (SVD; truncated) on the term-document matrix $\mathbf{X}$ representing N terms (words or n-grams) in D documents

Latent Semantic Analysis (or Indexing) — LSA



Singular Value Decomposition (SVD; truncated) on the term-document matrix $\mathbf{X}$ representing N terms (words or n-grams) in D documents

$\mathbf{W}_K$: each topic's (K) distribution over N terms
$\mathbf{C}_K$: each document's (D) distribution over K topics
$\mathbf{\Sigma}_K$: topic importance

Probabilistic LSA — pLSA

For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

— For each word position i (1 to N) in the document d_j:

—— Choose a topic z_k with probability $p(z_k | d_j)$

—— Choose a word w_i with probability $p(w_i | z_k)$

Probabilistic LSA — pLSA

For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

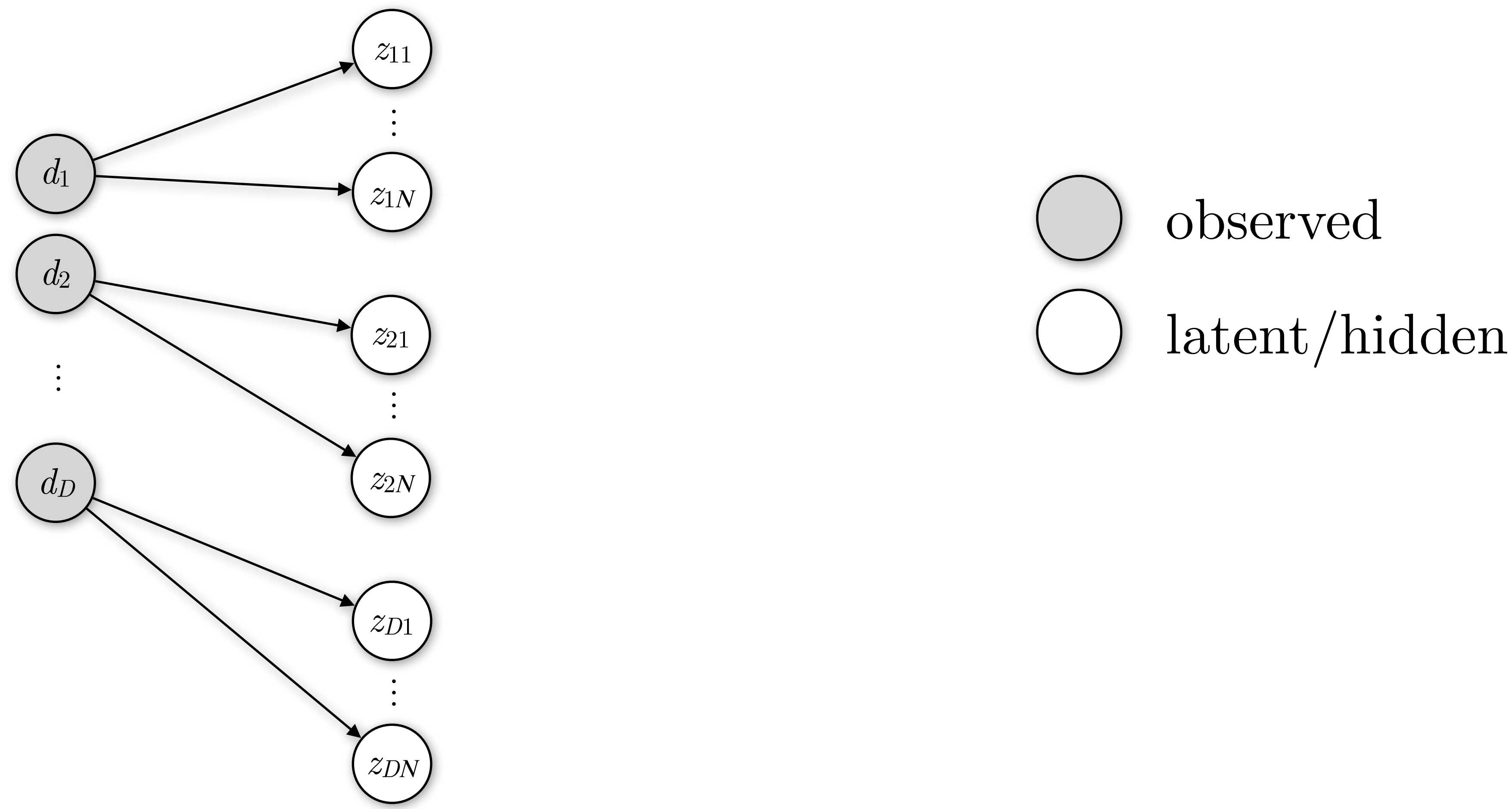
— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

— For each word position i (1 to N) in the document d_j:

—— Choose a topic z_k with probability $p(z_k | d_j)$

—— Choose a word w_i with probability $p(w_i | z_k)$

Probabilistic LSA — pLSA



For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

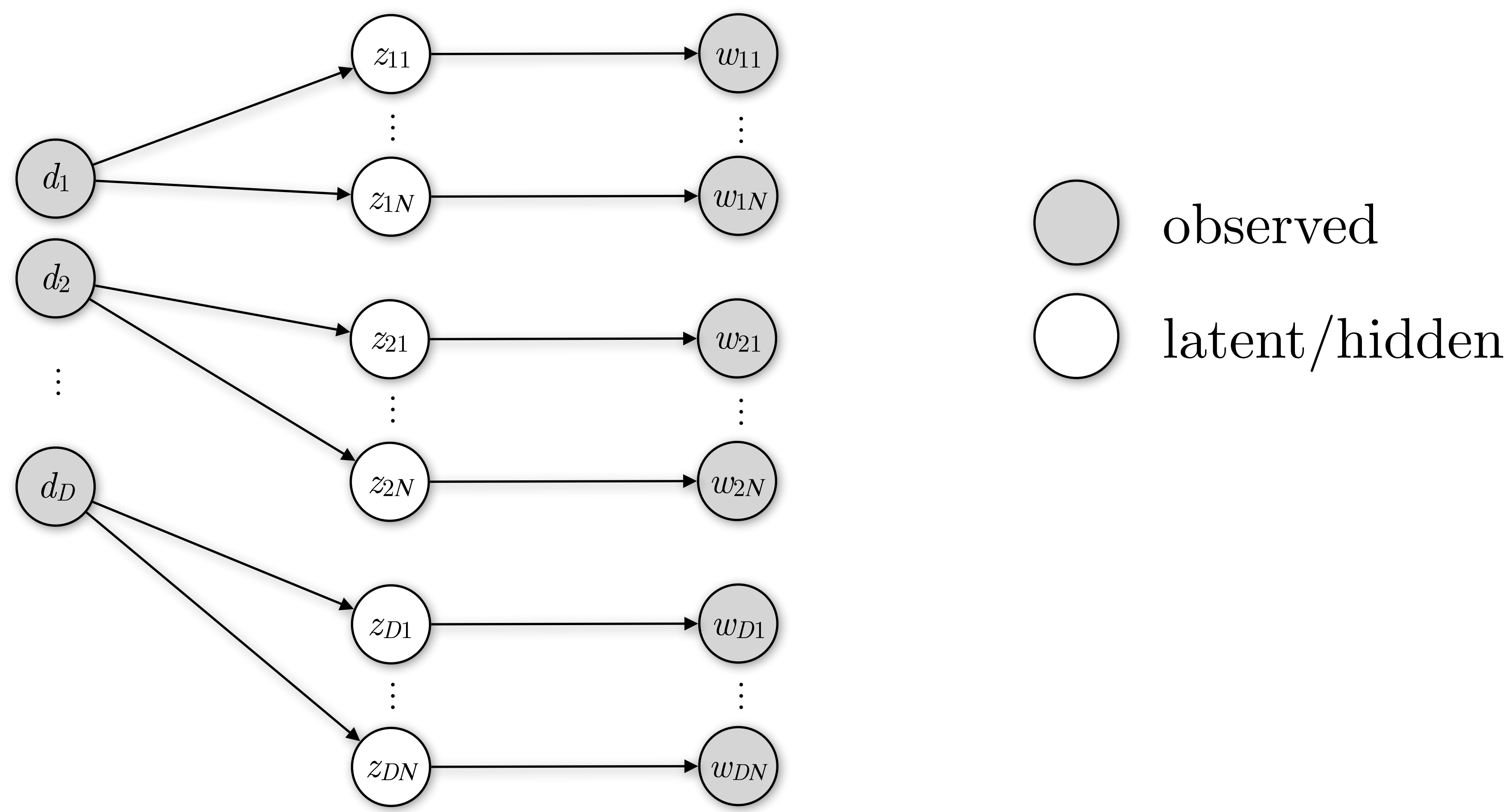
— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

— For each word position i (1 to N) in the document d_j:

—— Choose a topic z_k with probability $p(z_k | d_j)$

—— Choose a word w_i with probability $p(w_i | z_k)$

Probabilistic LSA — pLSA



For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

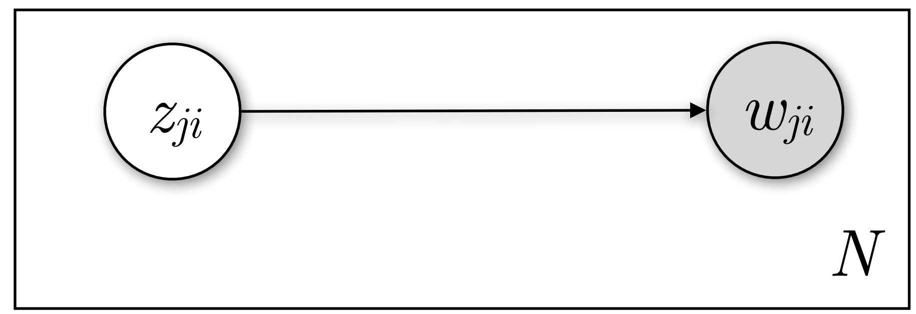
— For each word position i (1 to N) in the document d_j:

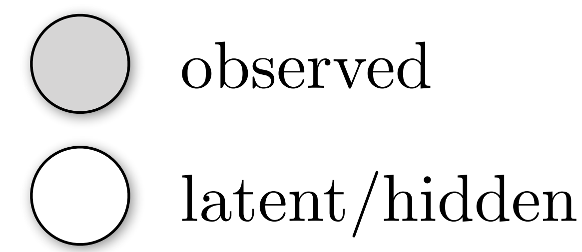
—— Choose a topic z_k with probability $p(z_k|d_j)$

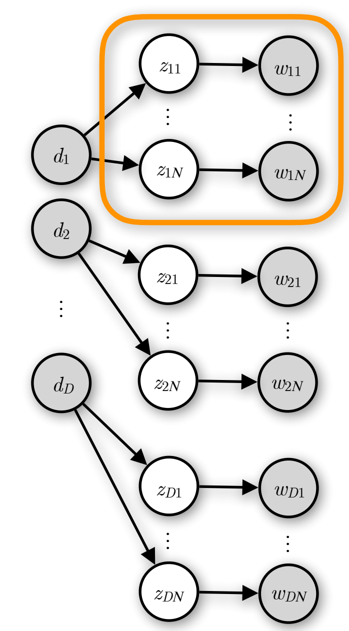
—— Choose a word w_i with probability $p(w_i|z_k)$

Probabilistic LSA — pLSA

Plate notation







For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

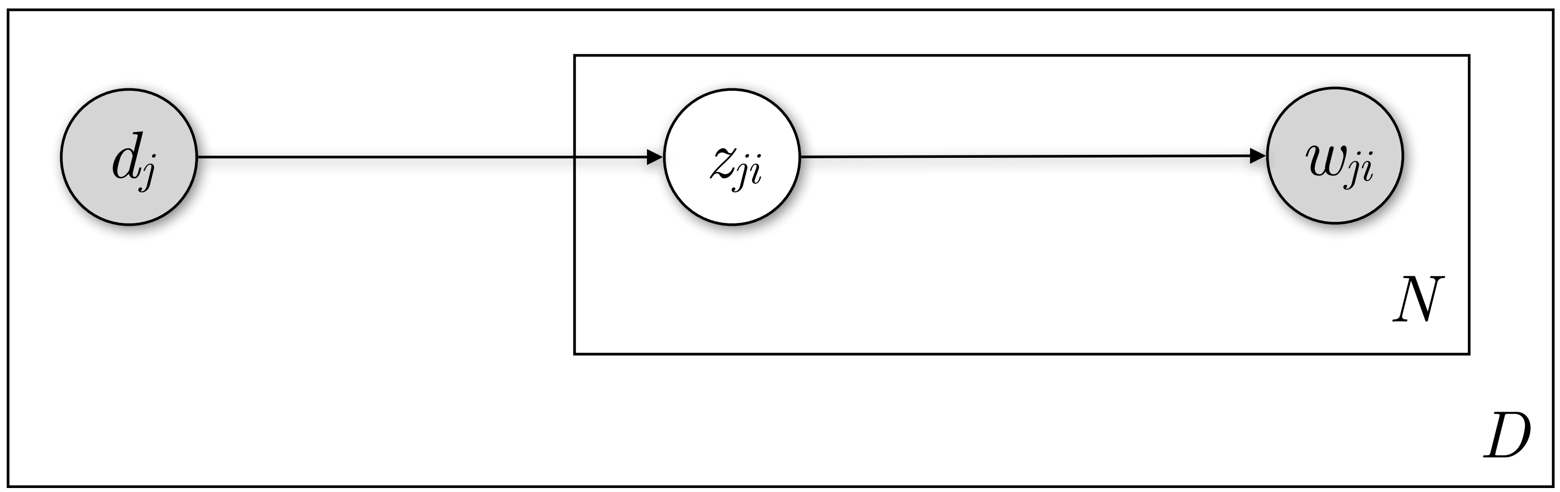
— For each word position i (1 to N) in the document d_j:

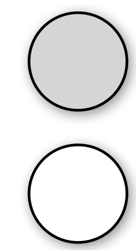
—— Choose a topic z_k with probability $p(z_k|d_j)$

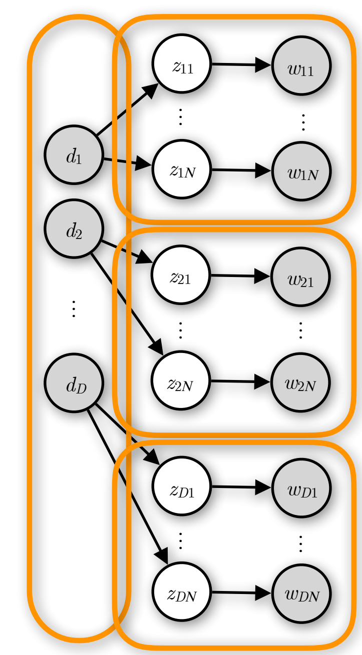
—— Choose a word w_i with probability $p(w_i|z_k)$

Probabilistic LSA — pLSA

Plate notation







For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

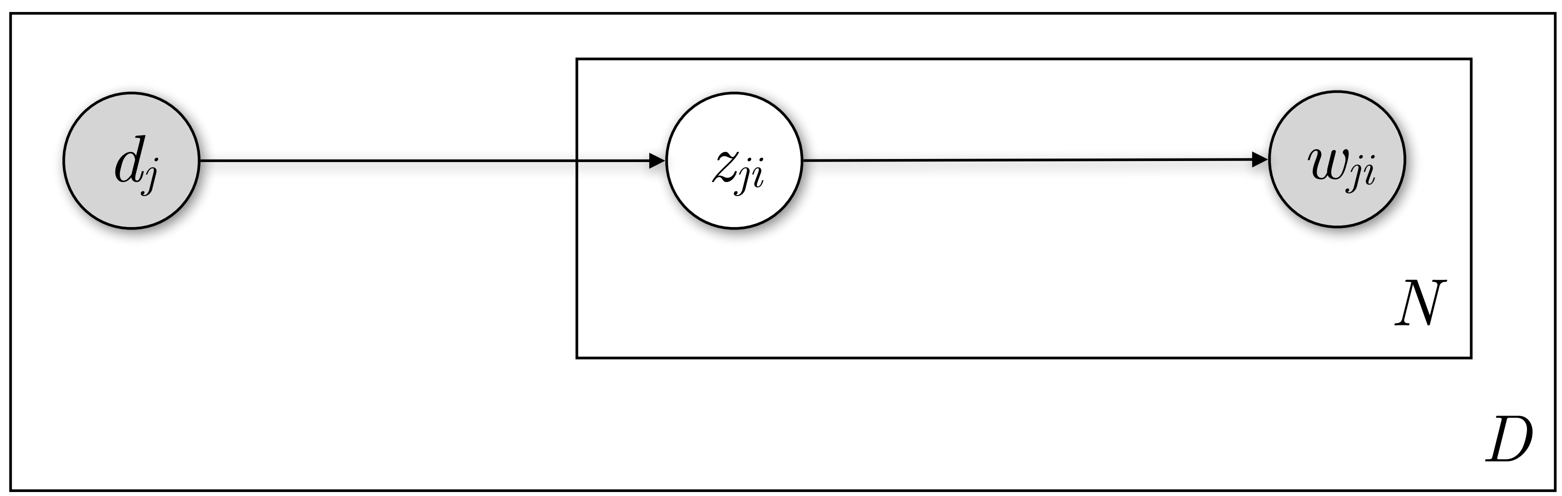
— For each word position i (1 to N) in the document d_j:

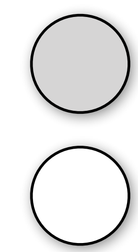
—— Choose a topic z_k with probability $p(z_k|d_j)$

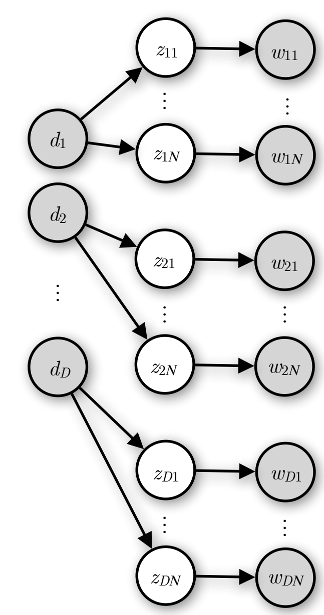
—— Choose a word w_i with probability $p(w_i|z_k)$

Probabilistic LSA — pLSA

Plate notation







$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_{ji}=k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji}=k)$$

For all j documents (1 to D):

— Select a document d_j with probability $p(d_j)$

— Choose a mixture of K topics $\boldsymbol{\theta}_j$ for document d_j

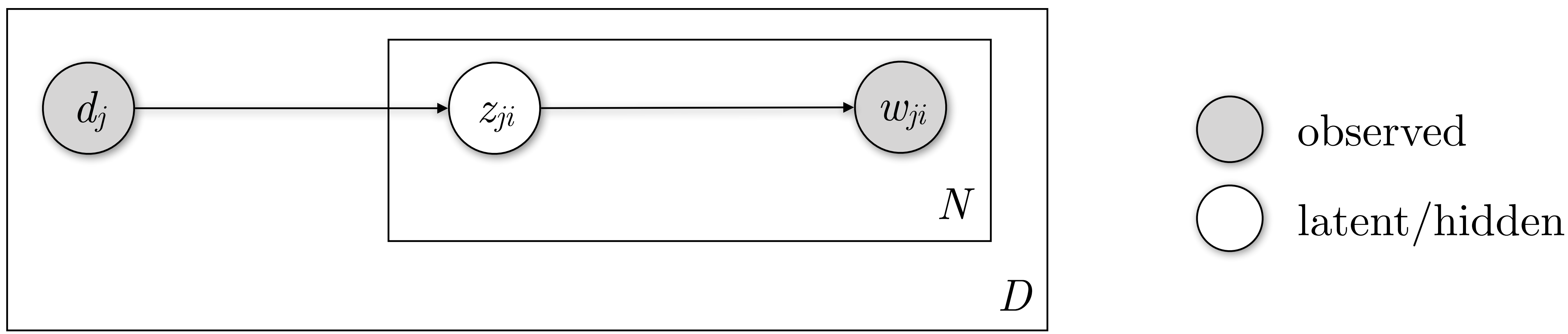
— For each word position i (1 to N) in the document d_j:

—— Choose a topic z_k with probability $p(z_k|d_j)$

—— Choose a word w_i with probability $p(w_i|z_k)$

Probabilistic LSA — pLSA

Plate notation

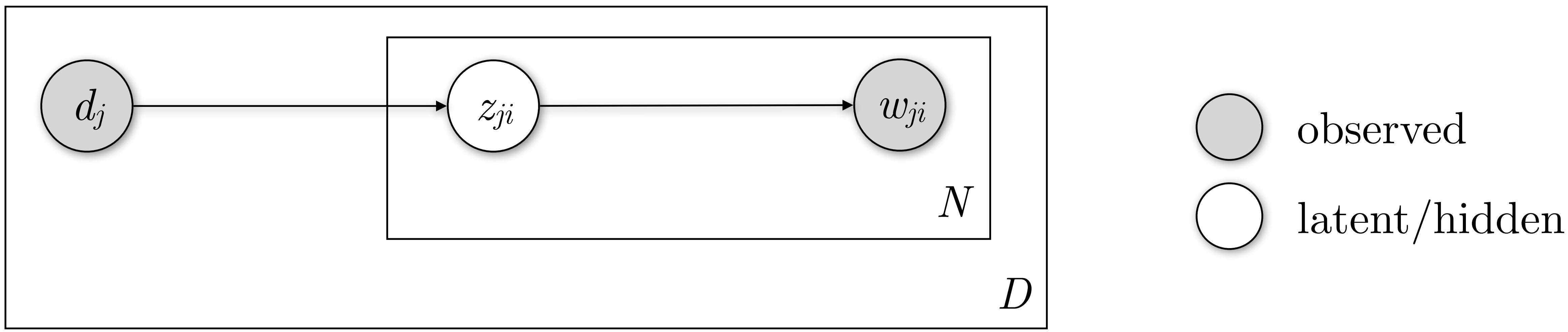


<u>Assumptions</u>: In a document (d_j), every word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

$$p(d_j, w_i) = p(d_j)\, p(w_i \,|\, d_j) = p(d_j) \sum_{k=1}^{K} p(z = k \,|\, d_j)\, p(w_i \,|\, z = k)$$ Joint prob. dist. for d_j and w_i

Probabilistic LSA — pLSA

Plate notation



<u>Assumptions</u>: In a document (d_j), every word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

$$p(d_j, w_i) = p(d_j)\, p(w_i \mid d_j) = p(d_j) \sum_{k=1}^{K} p(z = k \mid d_j)\, p(w_i \mid z = k)$$

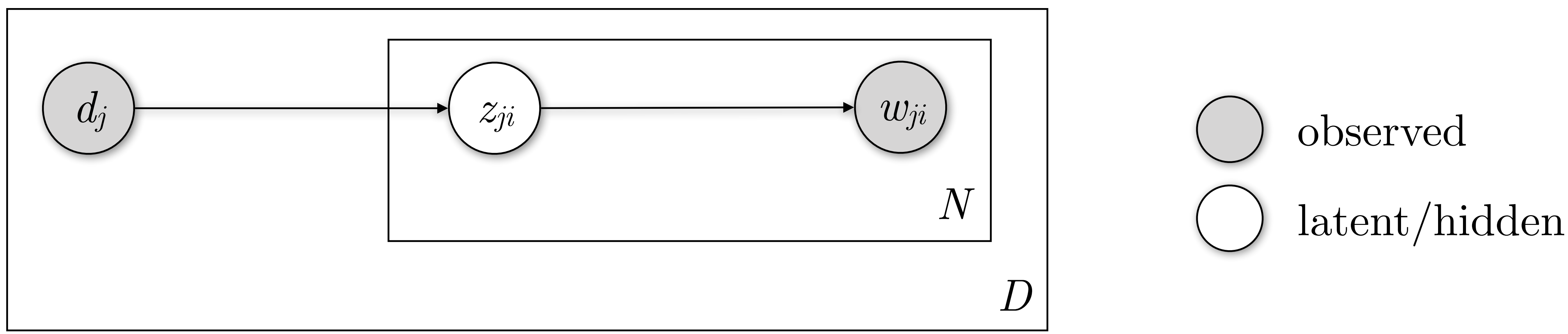
Joint prob. dist. for d_j and w_i

$$p(d_j, w) = p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_i = k \mid d_j)\, p(w_i \mid z_i = k)$$

Joint prob. dist. for d_j and w

Probabilistic LSA — pLSA

Plate notation



<u>Assumptions</u>: In a document (d_j), every word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

$$p(d_j, w_i) = p(d_j)\, p(w_i \,|\, d_j) = p(d_j) \sum_{k=1}^{K} p(z = k \,|\, d_j)\, p(w_i \,|\, z = k)$$

Joint prob. dist. for d_j and w_i

$$p(d_j, w) = p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_i = k \,|\, d_j)\, p(w_i \,|\, z_i = k)$$

Joint prob. dist. for d_j and w

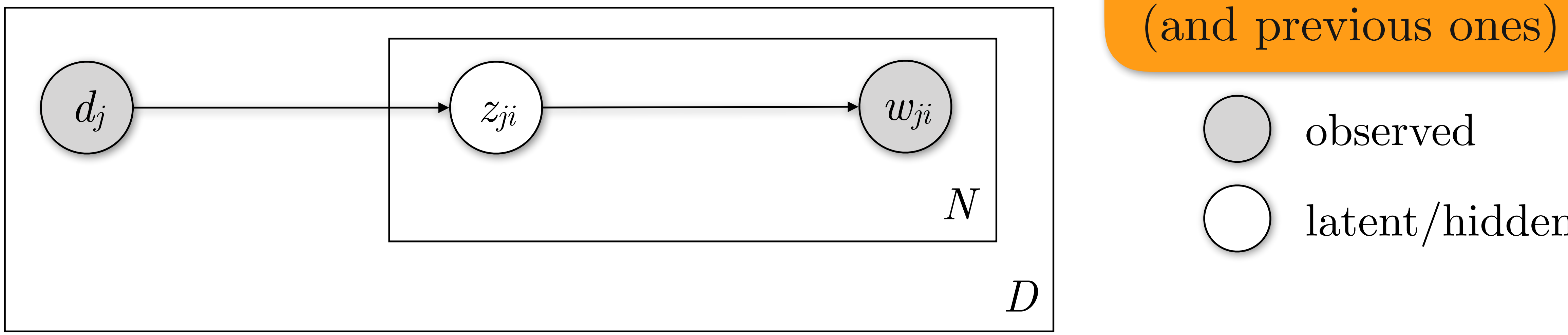
$$p(d, w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_{ji} = k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji} = k)$$

Joint probability distribution

Probabilistic LSA — pLSA

Find a *minor* **mistake** in this slide (and previous ones)

Plate notation



Assumptions: In a document (d_j), every word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

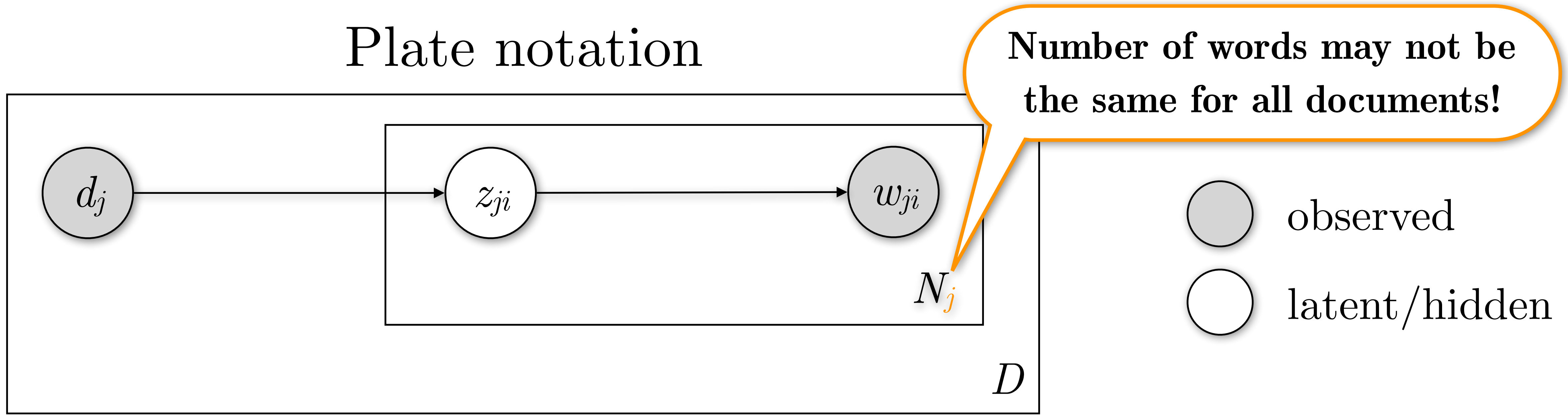
$$p(d_j, w_i) = p(d_j)\, p(w_i \mid d_j) = p(d_j) \sum_{k=1}^{K} p(z = k \mid d_j)\, p(w_i \mid z = k)$$ Joint prob. dist. for d_j and w_i

$$p(d_j, w) = p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_i = k \mid d_j)\, p(w_i \mid z_i = k)$$ Joint prob. dist. for d_j and w

$$p(d, w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N} \sum_{k=1}^{K} p(z_{ji} = k \mid d_j)\, p(w_{ji} \mid z_{ji} = k)$$ Joint probability distribution

Probabilistic LSA — pLSA

Plate notation



Assumptions: In a document (d_j), every word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

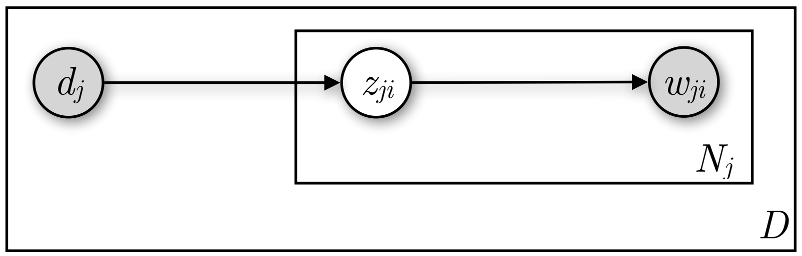
$$p(d_j, w_i) = p(d_j)\, p(w_i \mid d_j) = p(d_j) \sum_{k=1}^{K} p(z = k \mid d_j)\, p(w_i \mid z = k)$$ Joint prob. dist. for d_j and w_i

$$p(d_j, w) = p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_i = k \mid d_j)\, p(w_i \mid z_i = k)$$ Joint prob. dist. for d_j and w

$$p(d, w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji} = k \mid d_j)\, p(w_{ji} \mid z_{ji} = k)$$ Joint probability distribution

pLSA — Inference

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji} = k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji} = k)$$

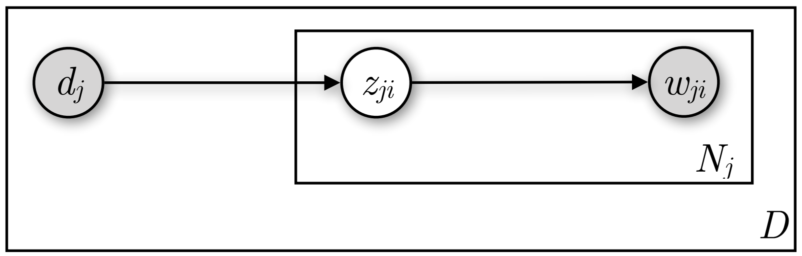


<u>Expectation Maximisation (EM)</u>:

- Compute expected values of the variables, given the current parametrisation of the model. In the very beginning, start with a *random* or *uniform* parametrisation (**E-step**)
- Then, pretending that the above values are correct, update the model parameters (**M-step**)
- Go back to the E-step; repeat until convergence

pLSA — Inference

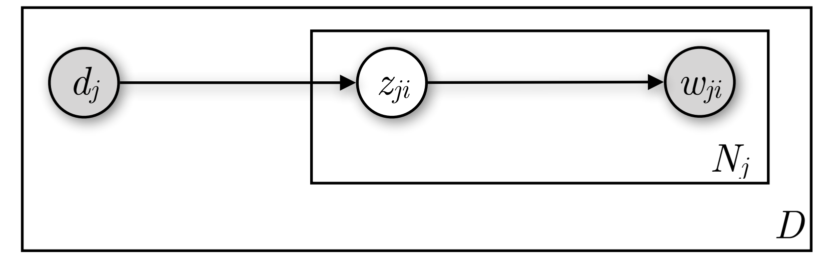
$$p(d, w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji} = k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji} = k)$$



- Initialise $p(z_k | d_j)$ and $p(w_i | z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

pLSA — Inference

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji} = k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji} = k)$$

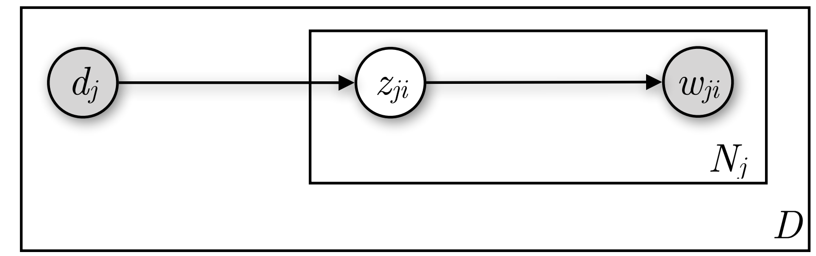


- Initialise $p(z_k | d_j)$ and $p(w_i | z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

$$p(z_k \,|\, d_j, w_i) = \frac{p(z_k \,|\, d_j)\, p(w_i \,|\, z_k)}{\sum_{k'=1}^{K} p(z_{k'} \,|\, d_j)\, p(w_i \,|\, z_{k'})}$$

pLSA — Inference

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji} = k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji} = k)$$



- Initialise $p(z_k|d_j)$ and $p(w_i|z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

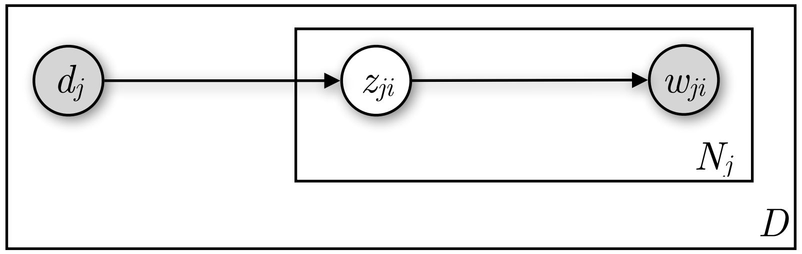
$$p(z_k \,|\, d_j, w_i) = \frac{p(z_k \,|\, d_j)\, p(w_i \,|\, z_k)}{\sum_{k'=1}^{K} p(z_{k'} \,|\, d_j)\, p(w_i \,|\, z_{k'})}$$

- **M-step:** Re-estimate $p(z_k|d_j)$, $p(w_i|z_k)$ given the revised $p(z_k|d_j,w_i)$

$$p(z_k \,|\, d_j) = \frac{\sum_{j=1}^{N_j} n(d_j, w_i)\, p(z_k \,|\, d_j, w_i)}{\sum_{i=1}^{N_j} \sum_{k'=1}^{K} n(d_j, w_i)\, p(z_{k'} \,|\, d_j, w_i)} \qquad p(w_i \,|\, z_k) = \frac{\sum_{j=1}^{D} n(d_j, w_i)\, p(z_k \,|\, d_j, w_i)}{\sum_{j=1}^{D} \sum_{i'=1}^{N_j} n(d_j, w_{i'})\, p(z_k \,|\, d_j, w_{i'})}$$

pLSA — Inference

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji}=k\,|\,d_j)\, p(w_{ji}\,|\,z_{ji}=k)$$



- Initialise $p(z_k|d_j)$ and $p(w_i|z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

$$p(z_k\,|\,d_j, w_i) = \frac{p(z_k\,|\,d_j)\, p(w_i\,|\,z_k)}{\sum_{k'=1}^{K} p(z_{k'}\,|\,d_j)\, p(w_i\,|\,z_{k'})}$$

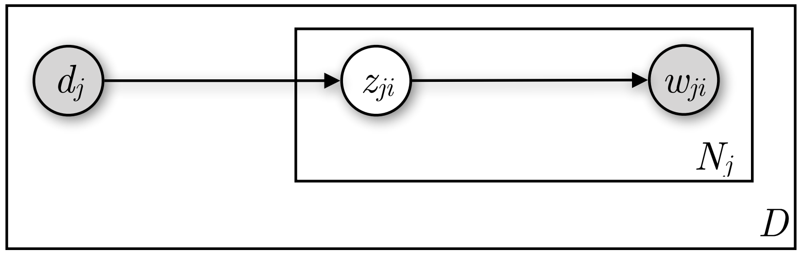
Fear not! This is just a weighted sum. $n(d_j,w_i)$ is the number of times word i appears in document j.

- **M-step:** Re-estimate $p(z_k|d_j)$, $p(w_i|z_k)$ given the revised $p(z_k|d_j,w_i)$

$$p(z_k\,|\,d_j) = \frac{\boxed{\sum_{j=1}^{N_j} n(d_j,w_i)\, p(z_k\,|\,d_j,w_i)}}{\sum_{i=1}^{N_j}\sum_{k'=1}^{K} n(d_j,w_i)\, p(z_{k'}\,|\,d_j,w_i)} \qquad p(w_i\,|\,z_k) = \frac{\boxed{\sum_{j=1}^{D} n(d_j,w_i)\, p(z_k\,|\,d_j,w_i)}}{\sum_{j=1}^{D}\sum_{i'=1}^{N_j} n(d_j,w_{i'})\, p(z_k\,|\,d_j,w_{i'})}$$

pLSA — Inference

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji}=k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji}=k)$$



- Initialise $p(z_k|d_j)$ and $p(w_i|z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

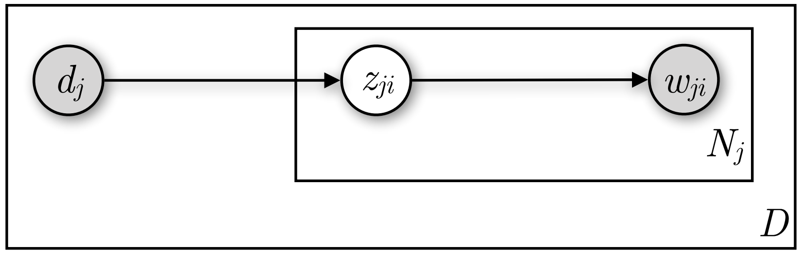
$$p(z_k \,|\, d_j, w_i) = \frac{p(z_k \,|\, d_j)\, p(w_i \,|\, z_k)}{\sum_{k'=1}^{K} p(z_{k'} \,|\, d_j)\, p(w_i \,|\, z_{k'})}$$

- **M-step:** Re-estimate $p(z_k|d_j)$, $p(w_i|z_k)$ given the revised $p(z_k|d_j,w_i)$

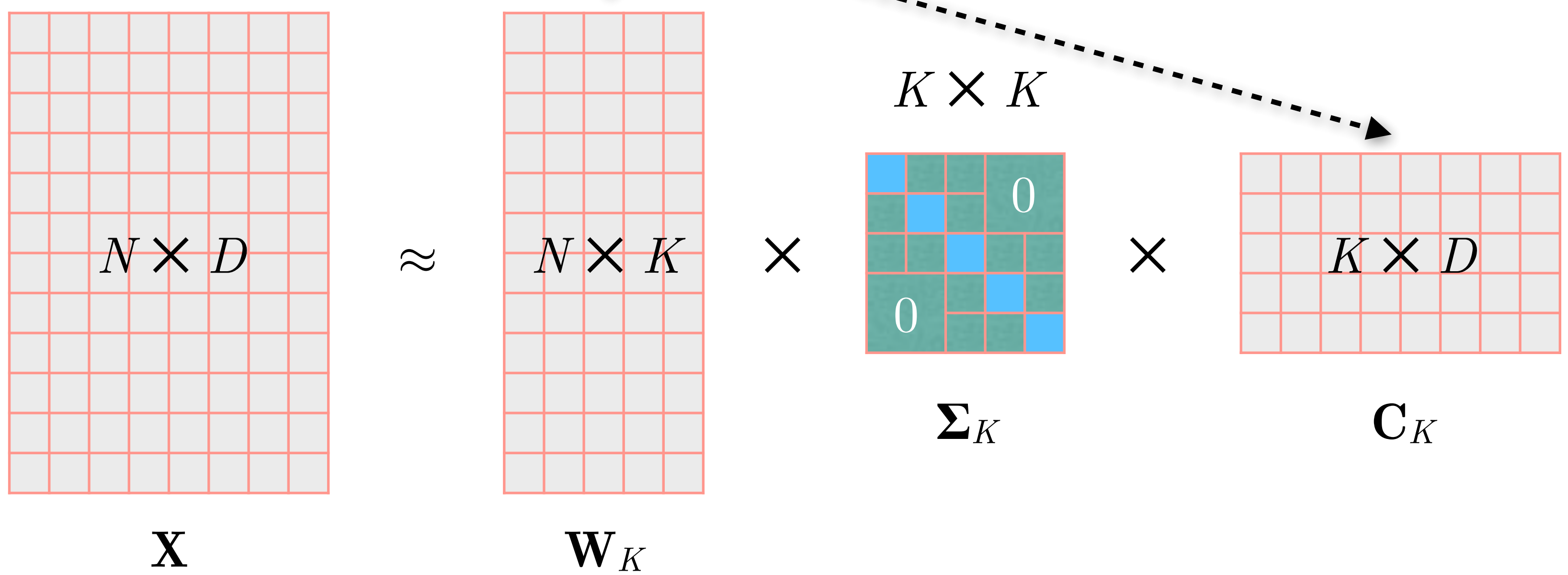
$$p(z_k \,|\, d_j) = \frac{\sum_{j=1}^{N_j} n(d_j, w_i)\, p(z_k \,|\, d_j, w_i)}{\sum_{i=1}^{N_j} \sum_{k'=1}^{K} n(d_j, w_i)\, p(z_{k'} \,|\, d_j, w_i)} \qquad p(w_i \,|\, z_k) = \frac{\sum_{j=1}^{D} n(d_j, w_i)\, p(z_k \,|\, d_j, w_i)}{\sum_{j=1}^{D} \sum_{i'=1}^{N_j} n(d_j, w_{i'})\, p(z_k \,|\, d_j, w_{i'})}$$

pLSA and LSA

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} \underline{p(z_{ji}=k\,|\,d_j)}\ \underline{p(w_{ji}\,|\,z_{ji}=k)}$$

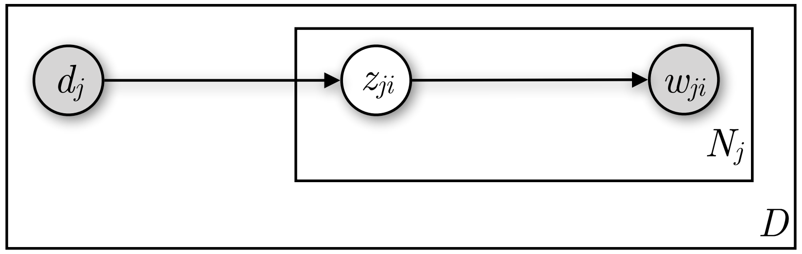


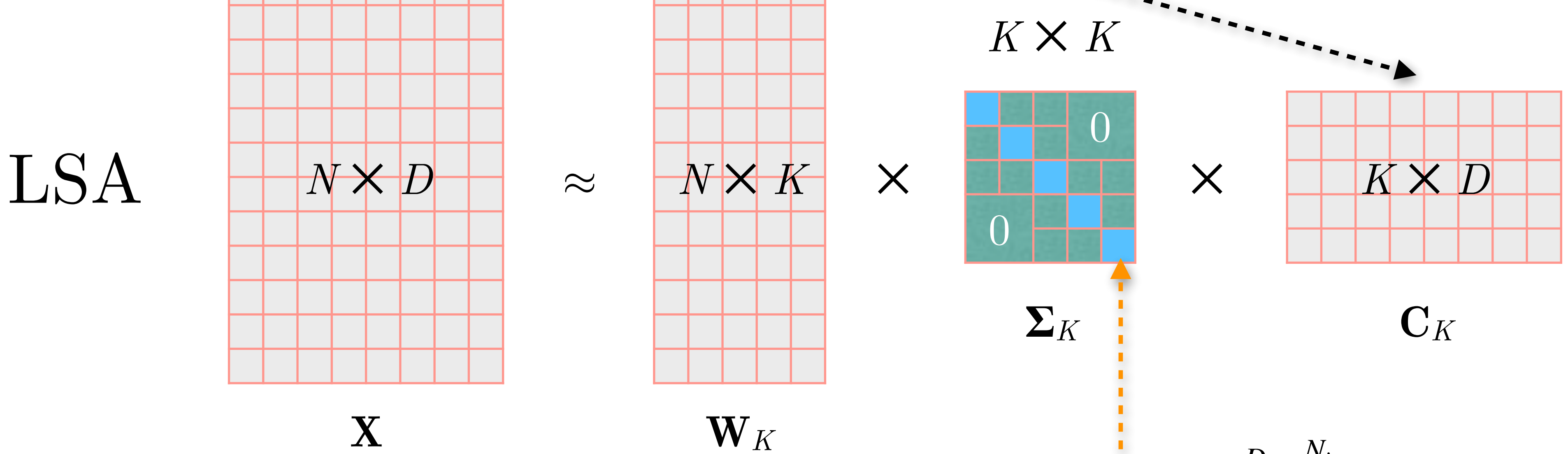
LSA



pLSA and LSA

$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} \underline{p(z_{ji}=k \,|\, d_j)}\; \underline{p(w_{ji} \,|\, z_{ji}=k)}$$

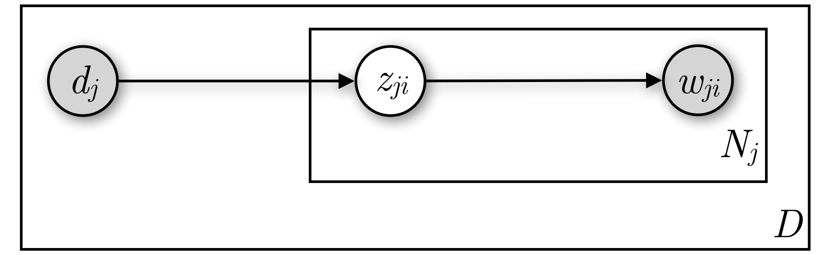


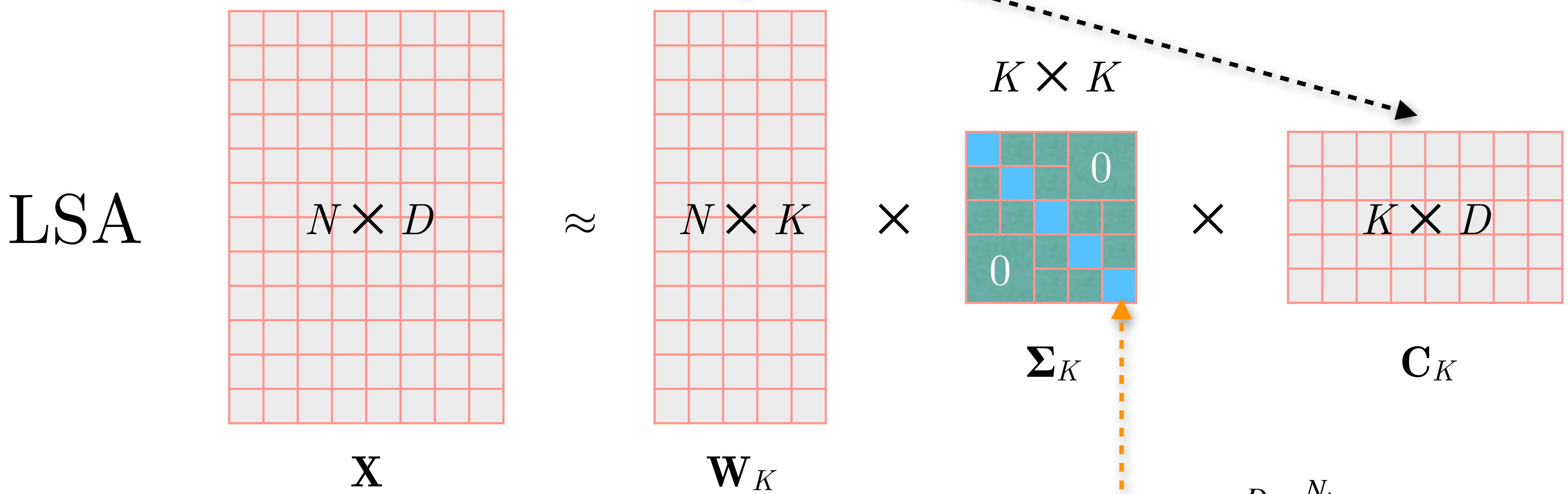


$$p(z_k) = \frac{\sum_{j=1}^{D} \sum_{i=1}^{N_j} n(d_j, w_i)\; p(z_k \,|\, d_j, w_i)}{\sum_{j=1}^{D} \sum_{i=1}^{N_j} n(d_j, w_i)}$$

pLSA and LSA

$$p(d, w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} \underline{p(z_{ji} = k \,|\, d_j)}\; \underline{p(w_{ji} \,|\, z_{ji} = k)}$$



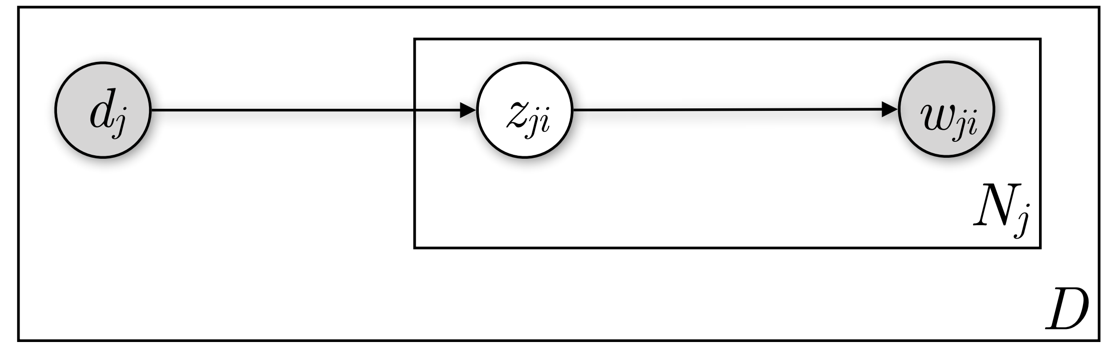


$$p(z_k) = \frac{\sum_{j=1}^{D} \sum_{i=1}^{N_j} n(d_j, w_i)\; p(z_k \,|\, d_j, w_i)}{\sum_{j=1}^{D} \sum_{i=1}^{N_j} n(d_j, w_i)}$$

<u>Main difference</u>: The two techniques have a different objective function — probabilistic vs. deterministic approach

pLSA — Disadvantages

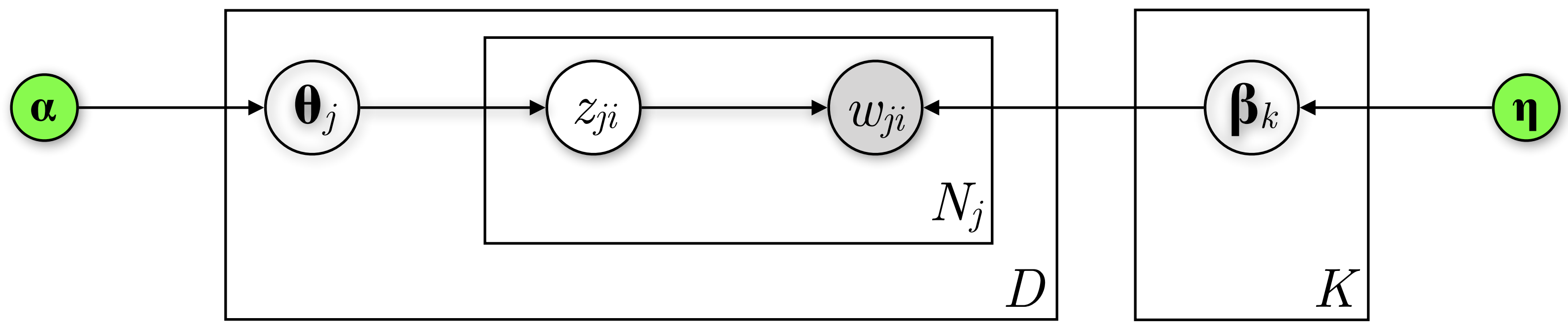
$$p(d,w) = \prod_{j=1}^{D} p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^{K} p(z_{ji}=k \,|\, d_j)\, p(w_{ji} \,|\, z_{ji}=k)$$



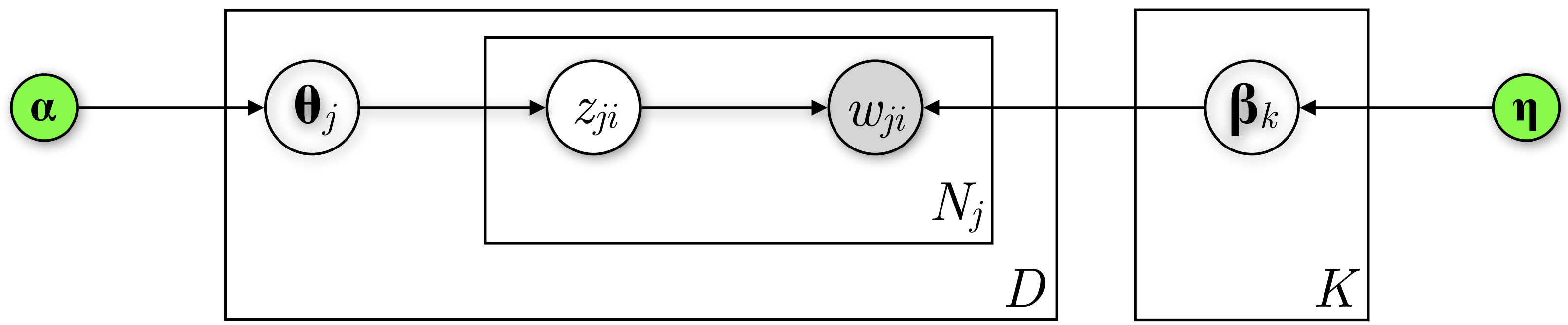
- The number of parameters that we need to infer during training grows linearly with the number of documents (D), which ultimately leads to overfitting.

- pLSA learns $p(z_k|d_j)$ only for the documents it sees during the training phase. To deal with a new document, it needs to repeat EM (retrain).

Latent Dirichlet Allocation (LDA)

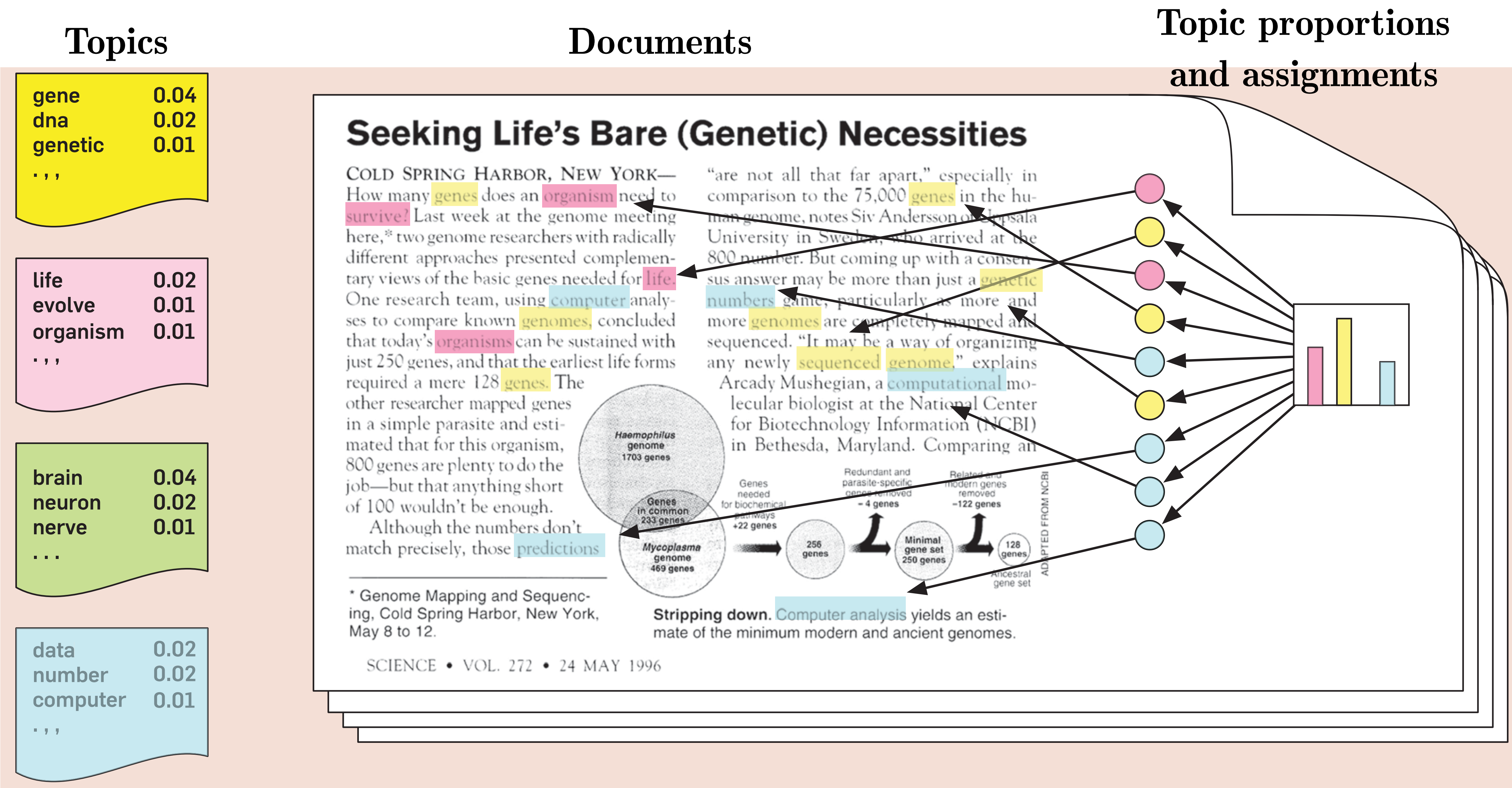


Latent Dirichlet Allocation (LDA)



- For each of the K topics draw a multinomial distribution $\boldsymbol{\beta}_k$ from a Dirichlet distribution with parameter $\boldsymbol{\eta}$
- For each of the D documents draw a multinomial distribution $\boldsymbol{\theta}_j$ from a Dirichlet distribution with parameter $\boldsymbol{\alpha}$
- For each word position i (1 to N_j) in a document j:
  - Select a latent topic z_{ji} from the multinomial distribution parametrised by $\boldsymbol{\theta}_j$
  - Choose the observed word w_{ji} from the multinomial distribution parametrised by $\boldsymbol{\beta}_{z_{ji}}$

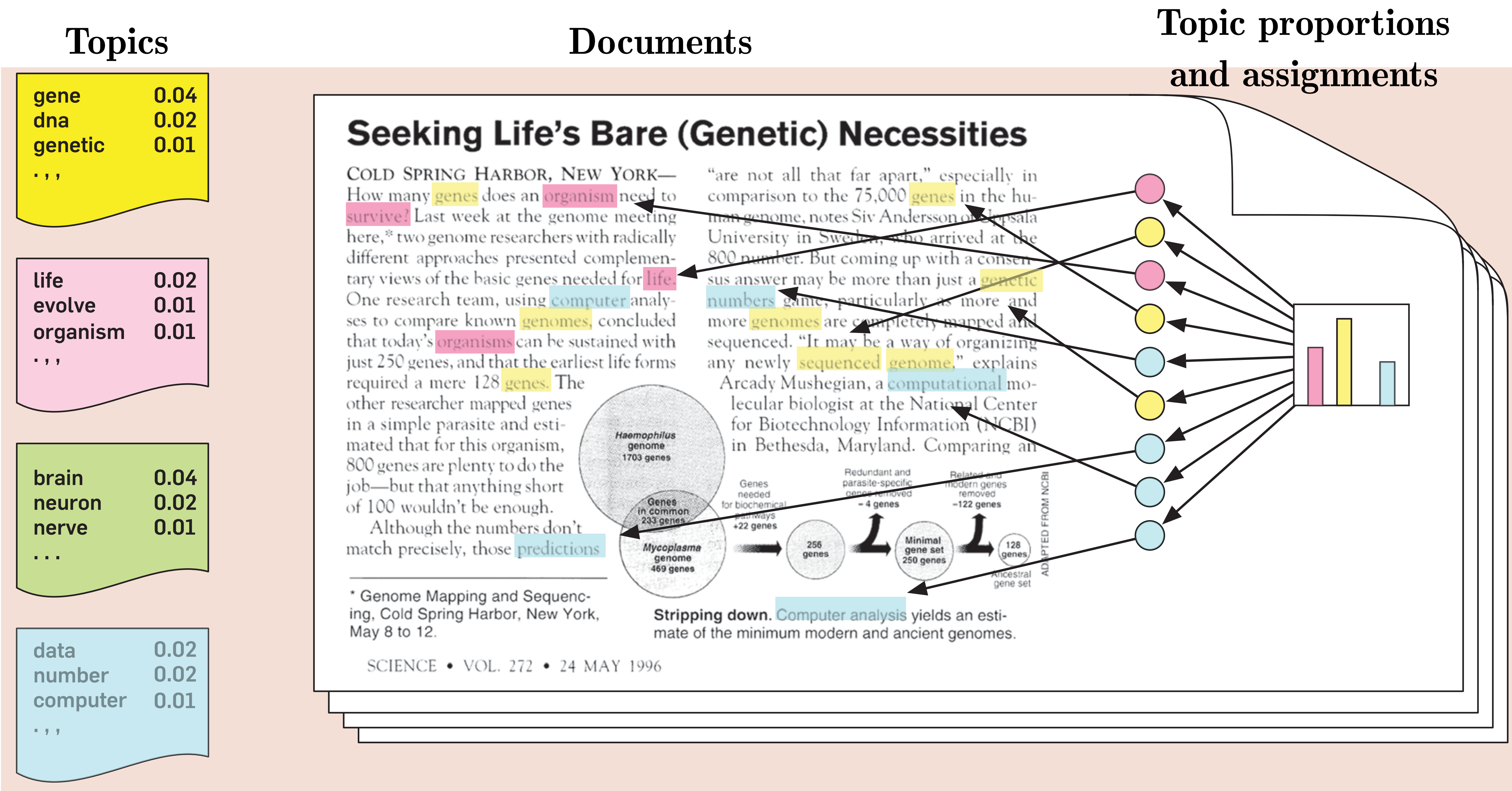
LDA — Generative story

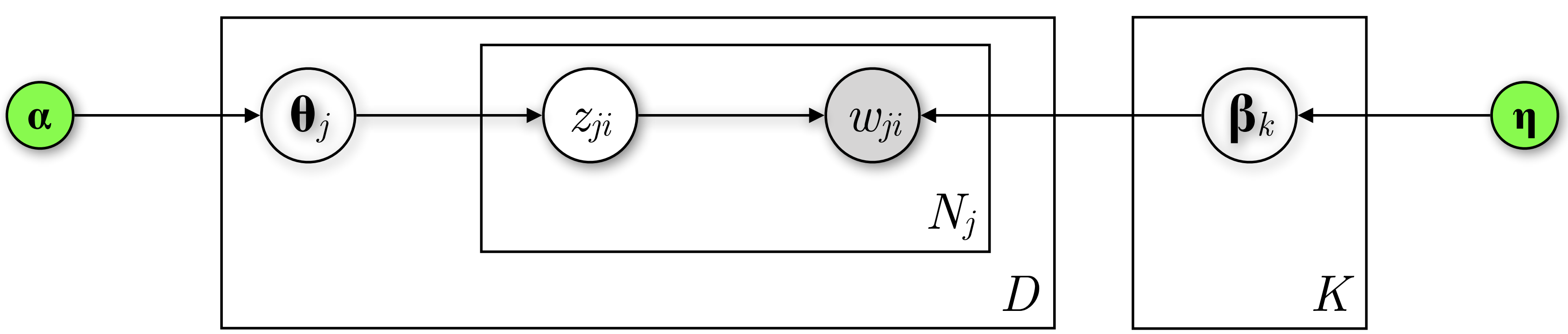


Assume a number of topics, defined as distributions over words (far left). A document is generated by first choosing a distribution over the topics (far right), then for each word position choosing a topic assignment (coloured coins), then choosing a word from the corresponding topic.

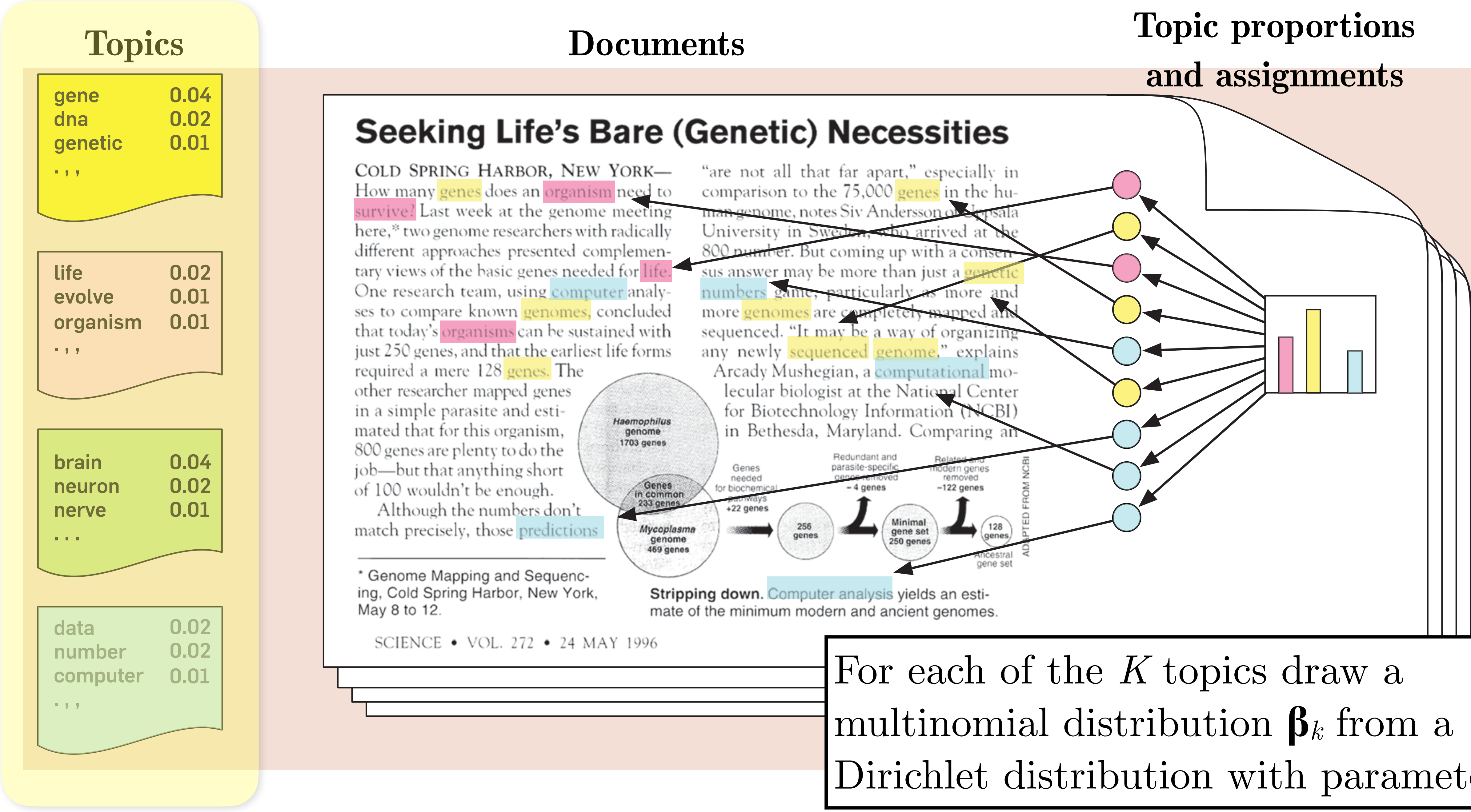
Blei. CACM, 2012. doi.org/10.1145/2133806.2133826

LDA — Generative story

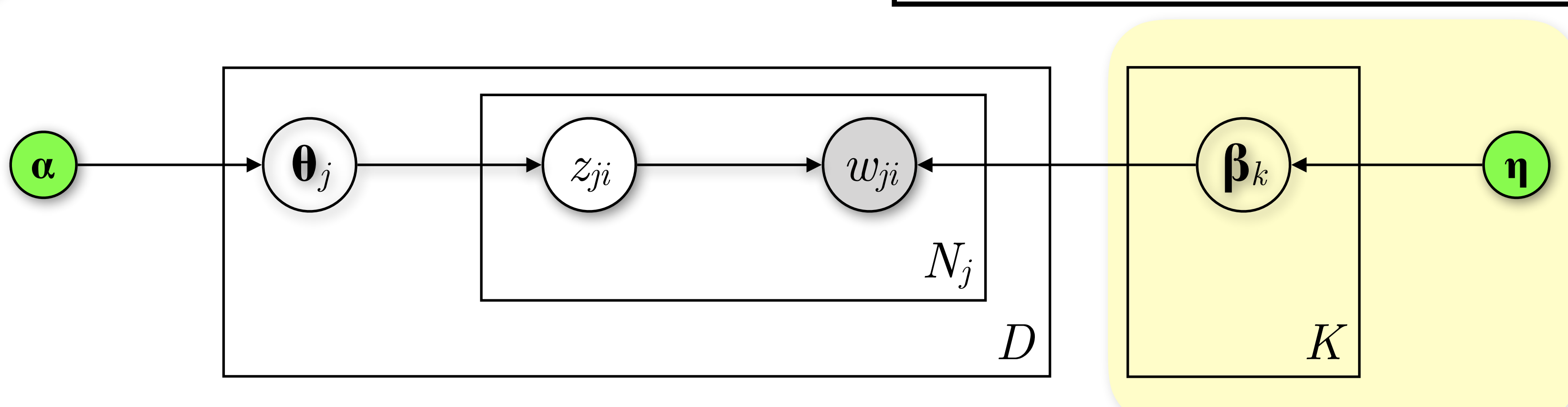




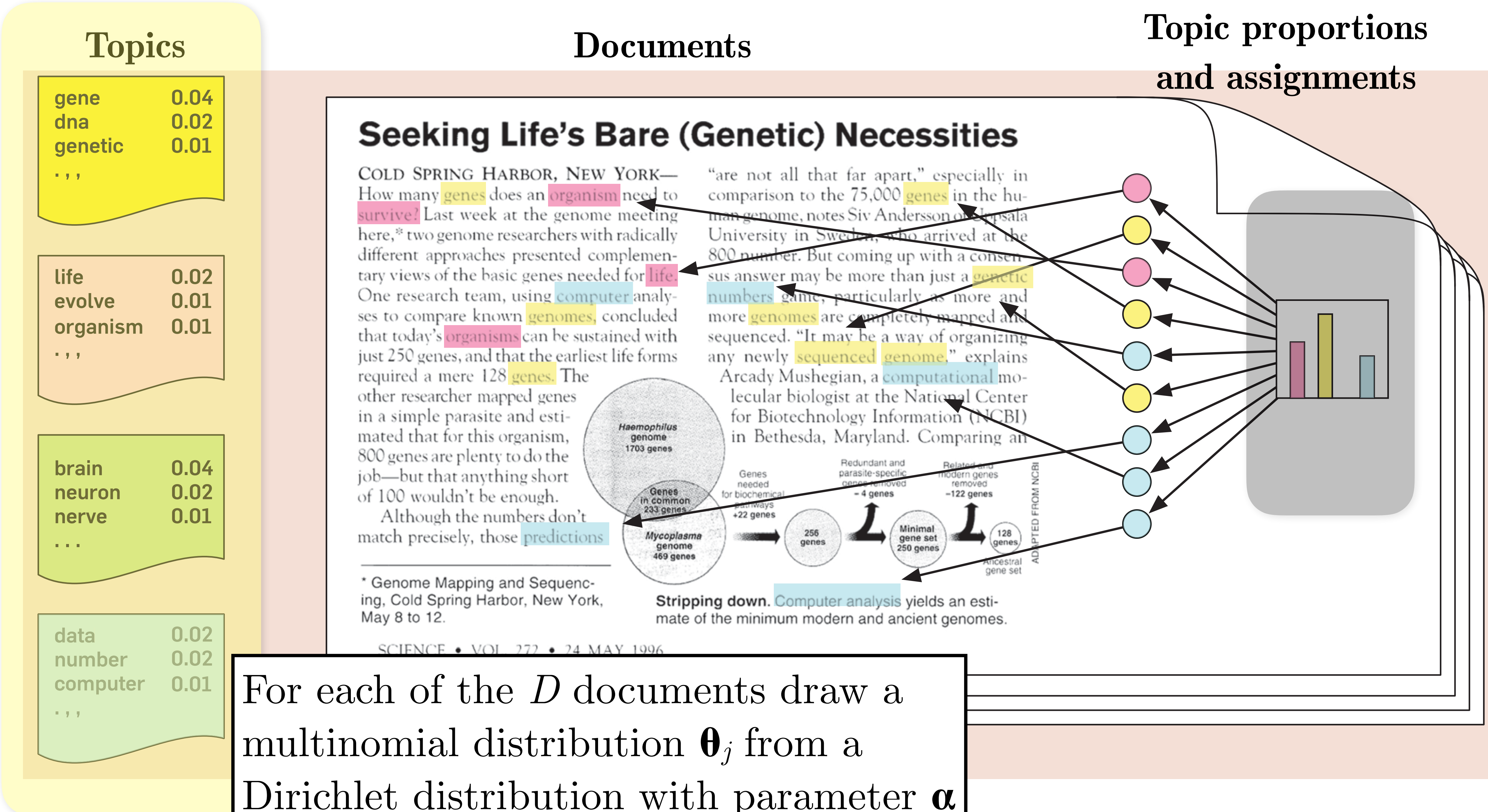
LDA — Generative story



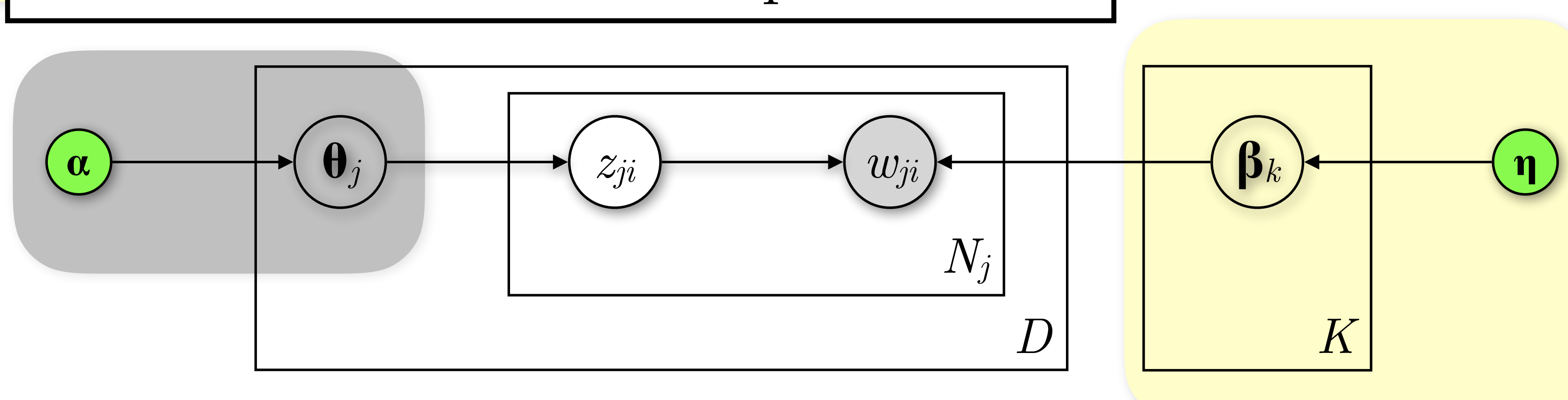
For each of the K topics draw a multinomial distribution $\boldsymbol{\beta}_k$ from a Dirichlet distribution with parameter $\boldsymbol{\eta}$



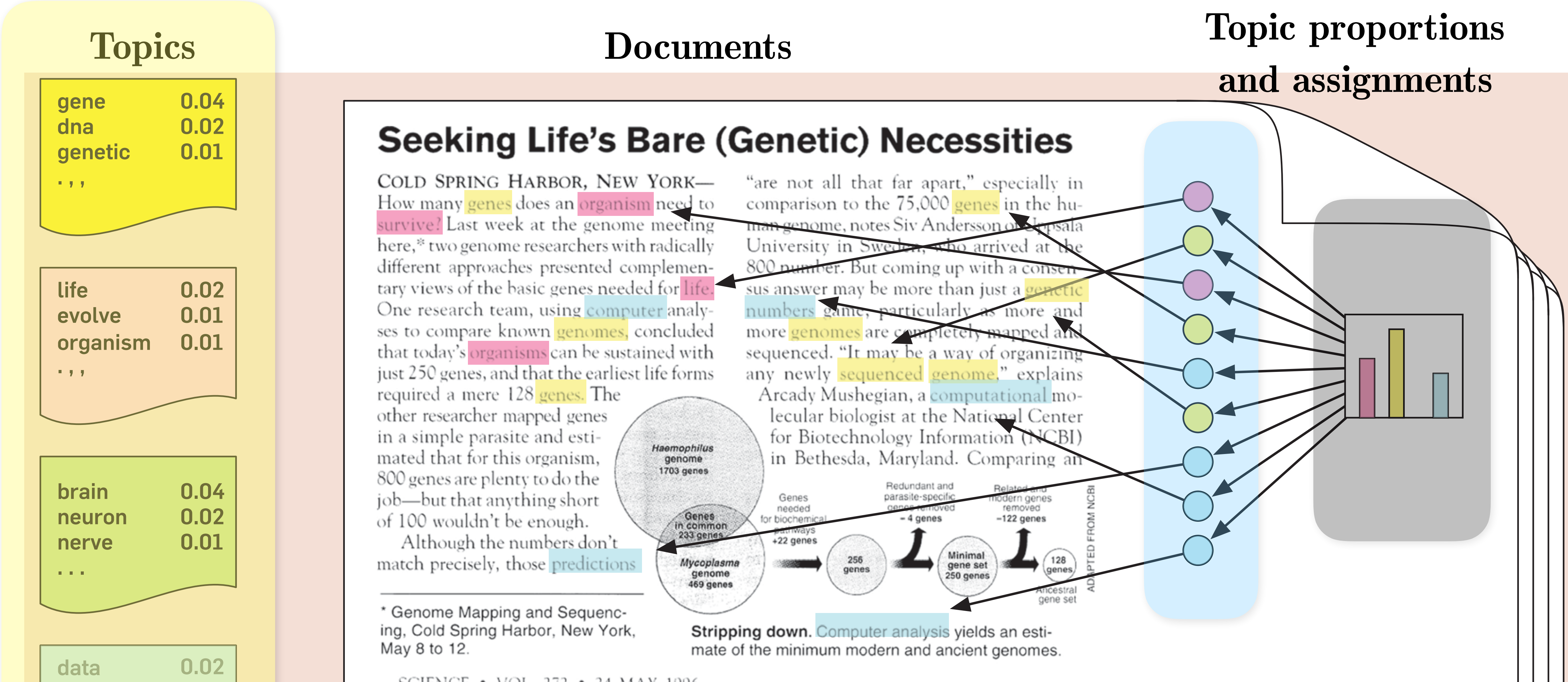
LDA — Generative story



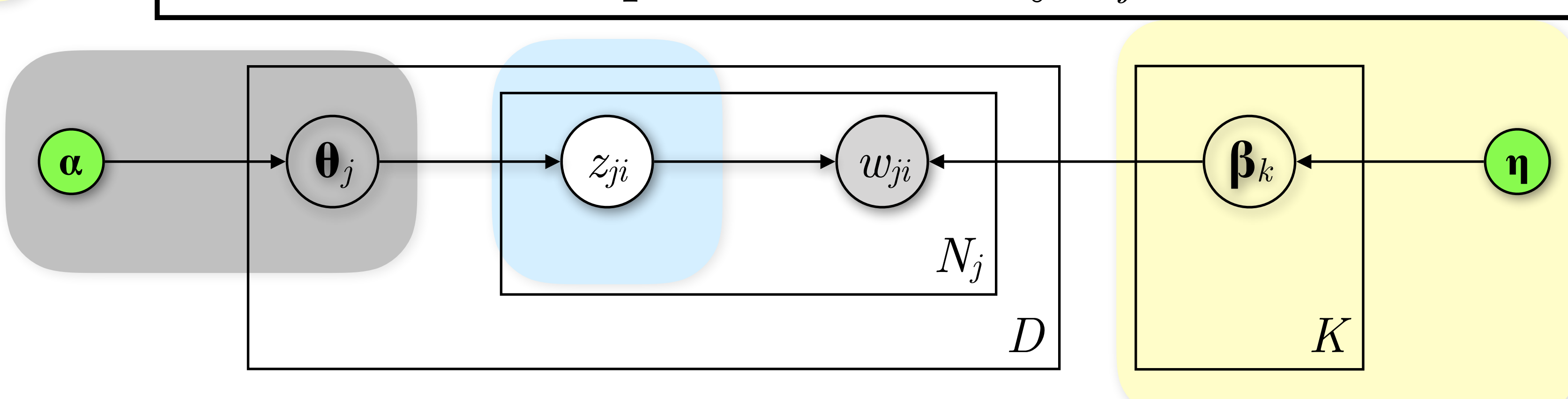
For each of the D documents draw a multinomial distribution $\boldsymbol{\theta}_j$ from a Dirichlet distribution with parameter $\boldsymbol{\alpha}$



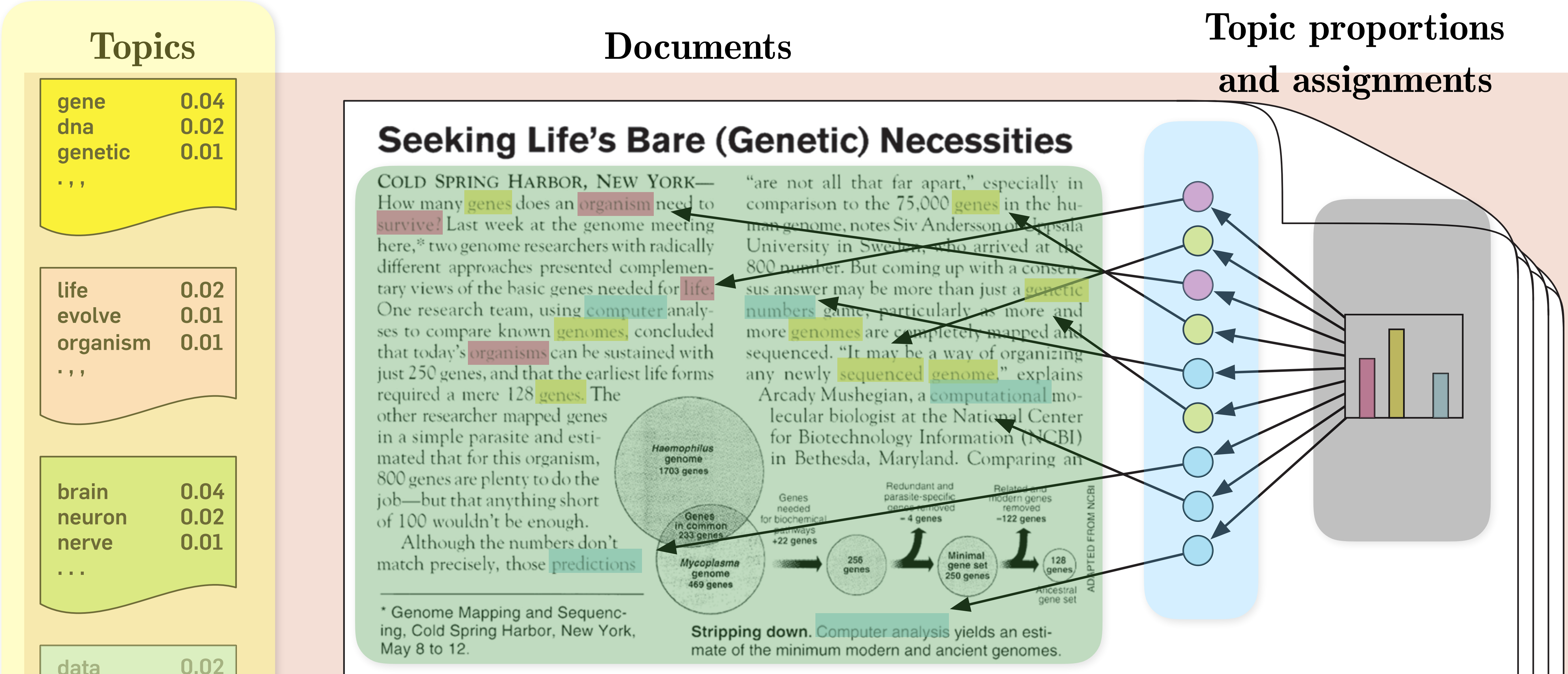
LDA — Generative story



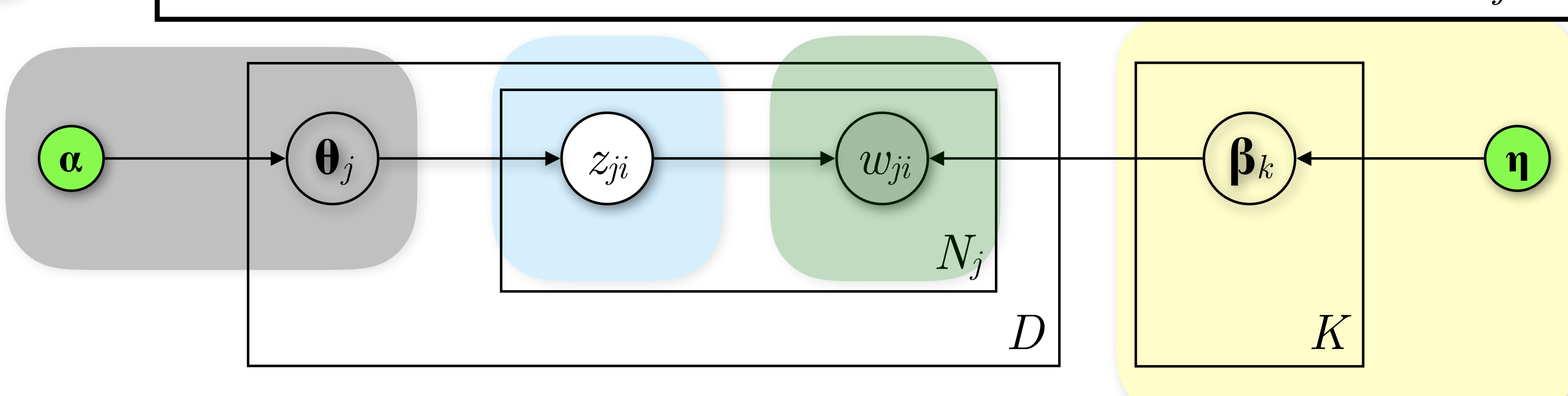
For each word position i (1 to N_j) in a document j:
— Select a latent topic z_{ji} from the multinomial distribution parametrised by $\boldsymbol{\theta}_j$



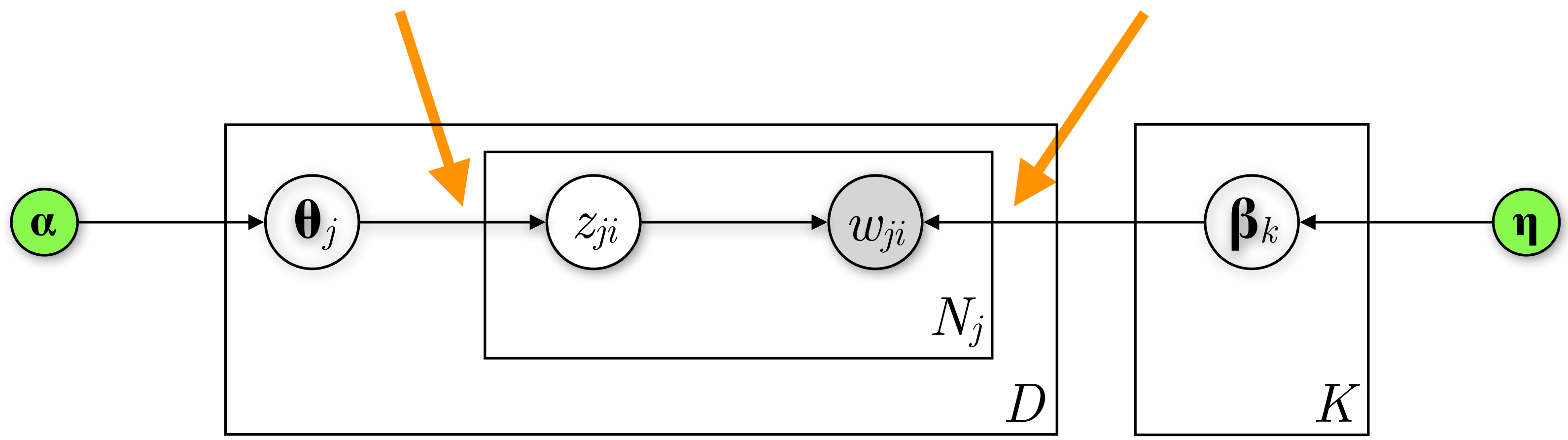
LDA — Generative story



For each word position i (1 to N_j) in a document j:
— Choose the observed word w_{ji} from the multinomial distribution parametrised by $\boldsymbol{\beta}_{z_{ji}}$



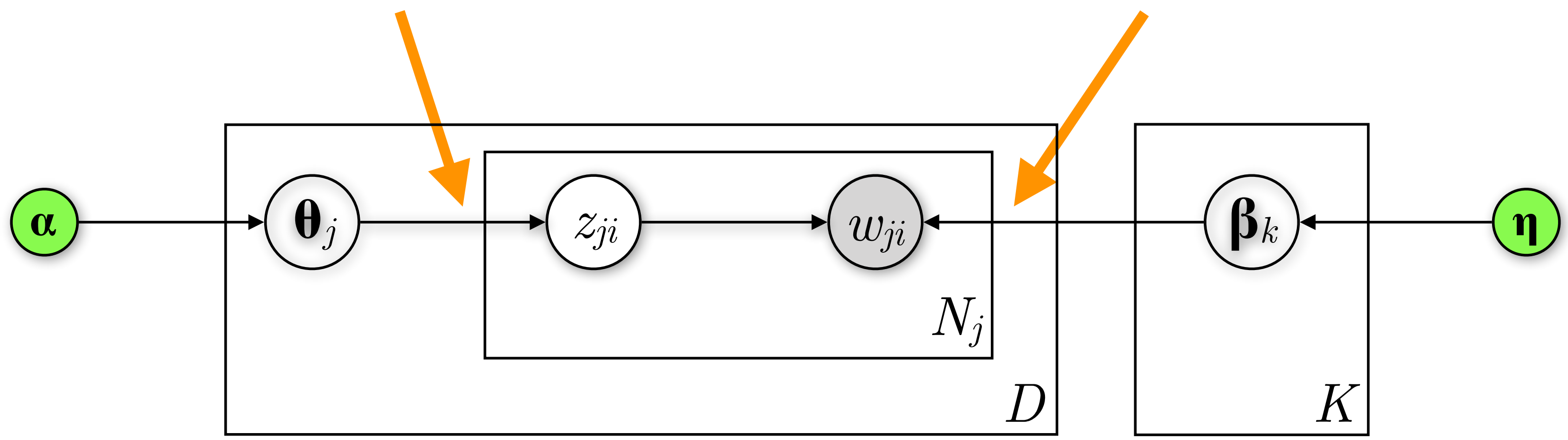
LDA — Multinomial distribution (Mult)



What is the probability of a set of outcomes for an event that has multiple outcomes?

— We roll a 6-sided dice 5 times. What is the probability of getting a "3" 1 time and a "6" 4 times?

LDA — Multinomial distribution (Mult)

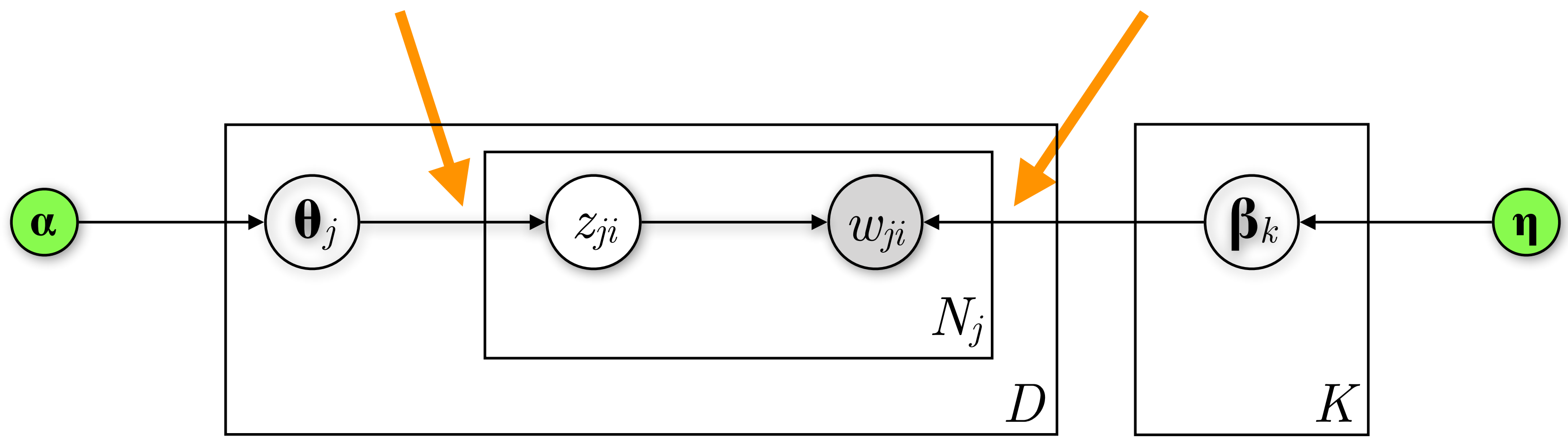


What is the probability of a set of outcomes for an event that has multiple outcomes?

— We roll a 6-sided dice 5 times. What is the probability of getting a "3" 1 time and a "6" 4 times?

$$\underbrace{\frac{5!}{1!4!}}_{\text{\#ways to get 1 "3" and 4 "6"s}} \cdot \underbrace{\left(\frac{1}{6}\right)}_{\text{prob. of 1 "3"}} \cdot \underbrace{\left(\frac{1}{6}\right)^4}_{\text{prob. of 4 "6"s}} \approx 0.00064$$

LDA — Multinomial distribution (Mult)



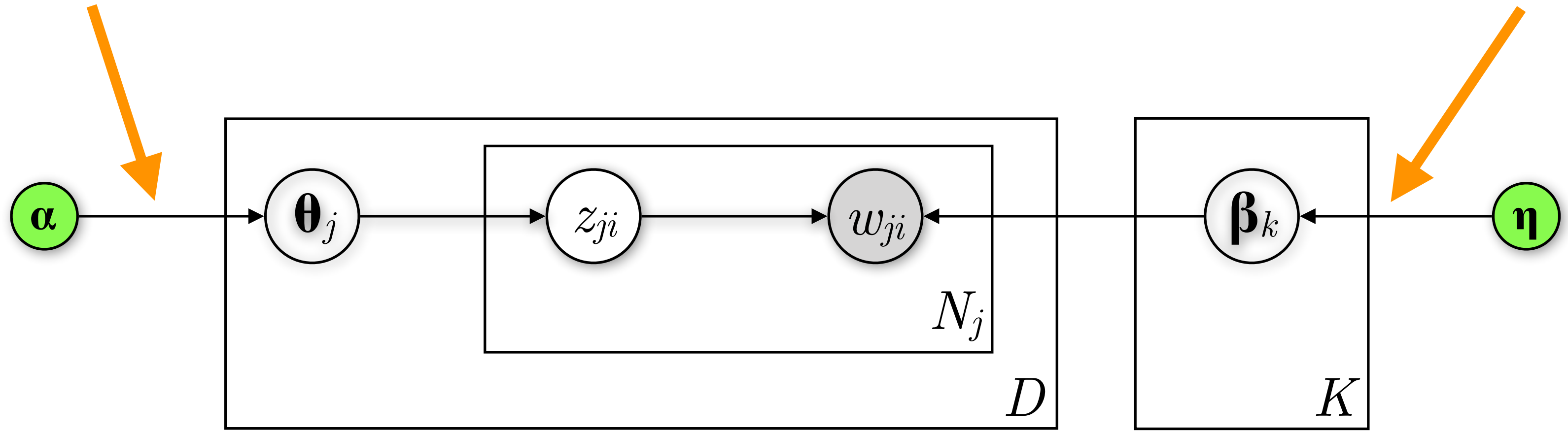
What is the probability of a set of outcomes for an event that has multiple outcomes?

— We roll a 6-sided dice 5 times. What is the probability of getting a "3" 1 time and a "6" 4 times?

$$\underbrace{\frac{5!}{1!4!}}_{\text{\#ways to get 1 "3" and 4 "6"s}} \cdot \underbrace{\left(\frac{1}{6}\right)}_{\text{prob. of 1 "3"}} \cdot \underbrace{\left(\frac{1}{6}\right)^4}_{\text{prob. of 4 "6"s}} \approx 0.00064$$

Formally: $p(n_1, \ldots, n_k) = \dfrac{n!}{n_1! \cdot \ldots \cdot n_k!} \cdot p_1^{n_1} \cdot \ldots \cdot p_k^{n_k}$ given $n, \{p_1, \ldots, p_k\}$

LDA — Dirichlet distribution (Dir)



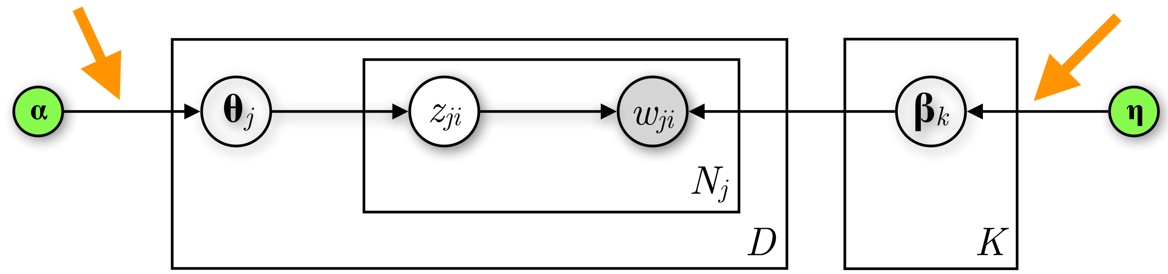
Exponential family distribution over the simplex (= positive vectors that sum up to 1), essentially a distribution over multinomial distributions

$$p(\boldsymbol{\theta}|\boldsymbol{\alpha}) = \frac{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}{\prod_{k=1}^{K} \Gamma\left(\alpha_k\right)} \cdot \prod_{k=1}^{K} \theta_k^{\alpha_k - 1} \quad \text{where} \quad \Gamma(n) = (n-1)!$$

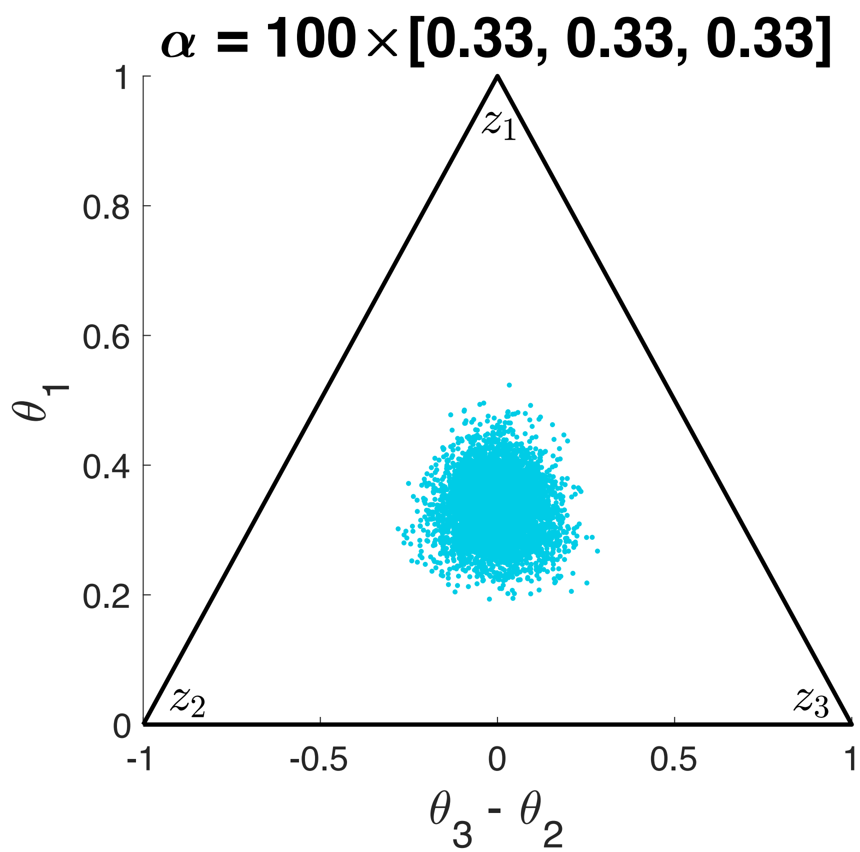
Parameter $\boldsymbol{\alpha}$ controls the mean shape and sparsity of $\boldsymbol{\theta}$ (and $\boldsymbol{\beta}$)

<u>Note</u>: $\boldsymbol{\alpha}$ is a vector of K parameters for $\boldsymbol{\theta}$ and $\boldsymbol{\eta}$ has V parameters for $\boldsymbol{\beta}$, where V is the size of the entire vocabulary (unique words across all D documents)

LDA — Dirichlet distribution (Dir)

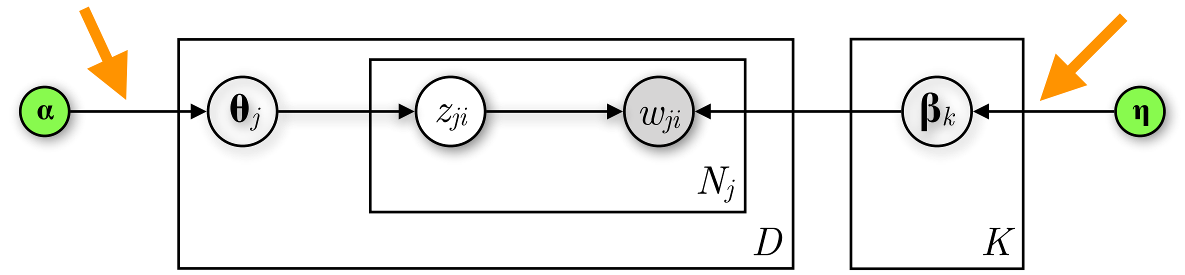


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

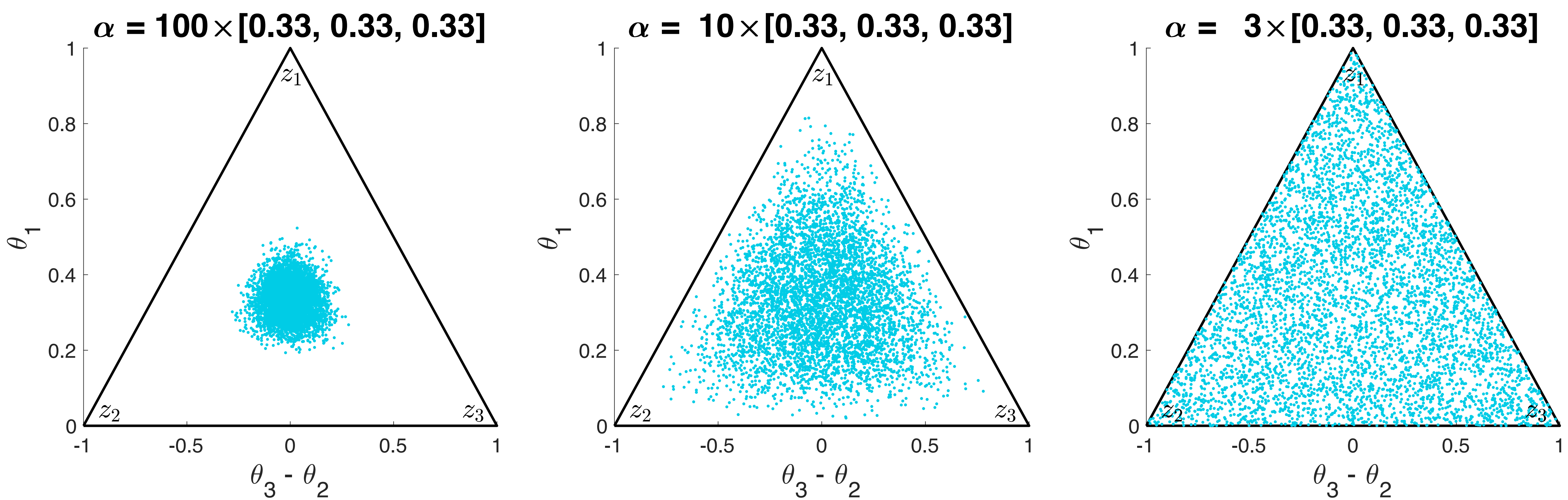


Large values of $\boldsymbol{\alpha}$ lead to more dense $\boldsymbol{\theta}$'s

LDA — Dirichlet distribution (Dir)

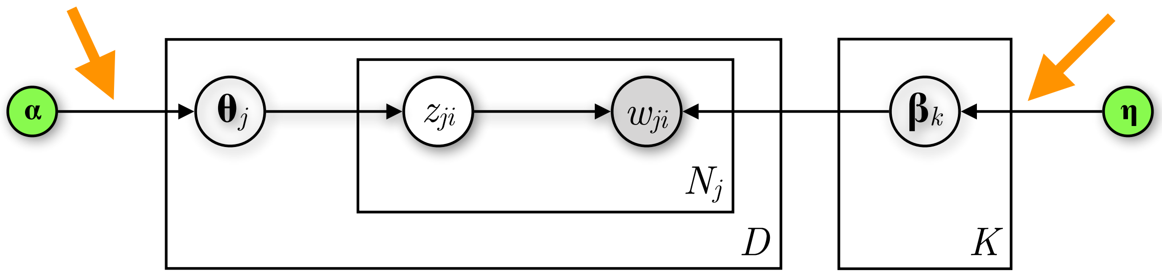


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

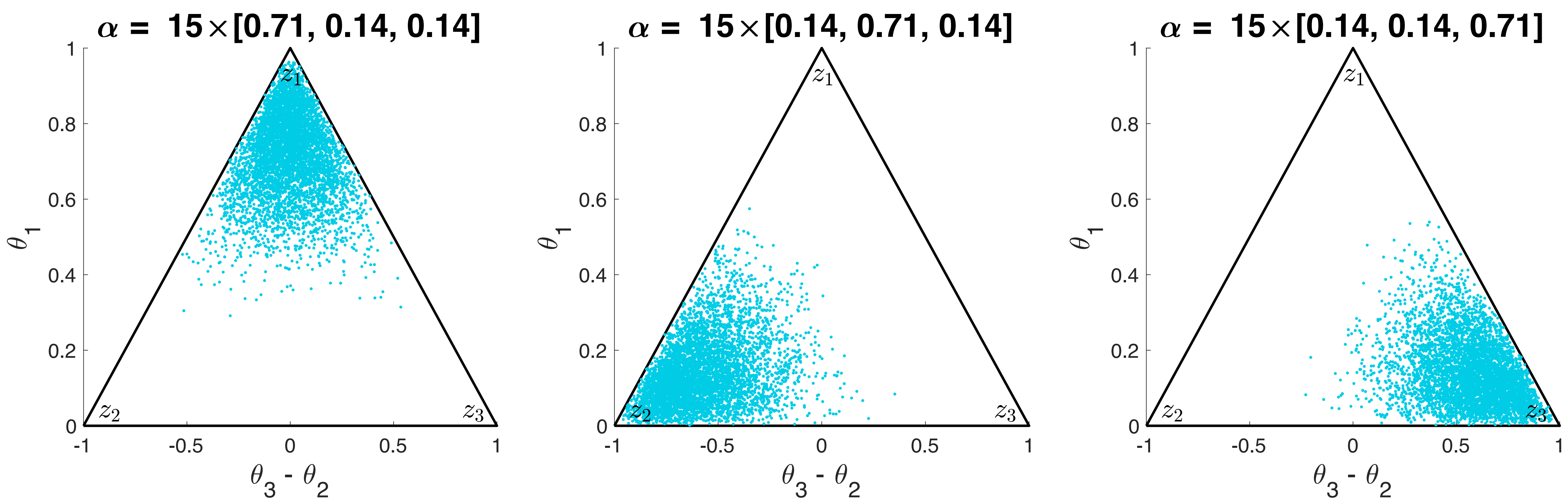


Large values of $\boldsymbol{\alpha}$ lead to more dense $\boldsymbol{\theta}$'s

LDA — Dirichlet distribution (Dir)

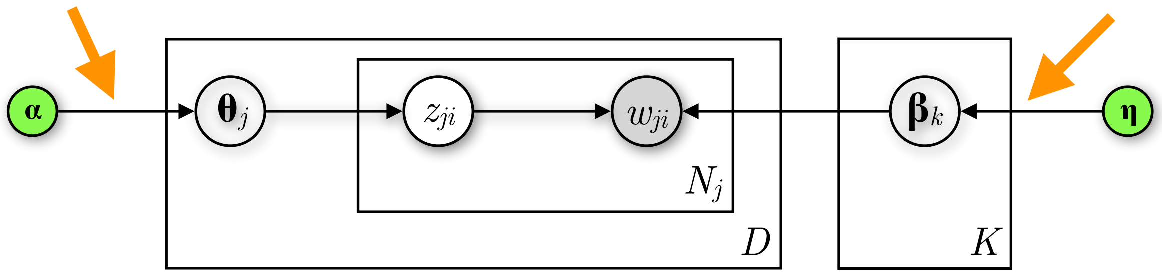


Assume a simplex $\boldsymbol{\theta} = [\theta_1,\ \theta_2,\ \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

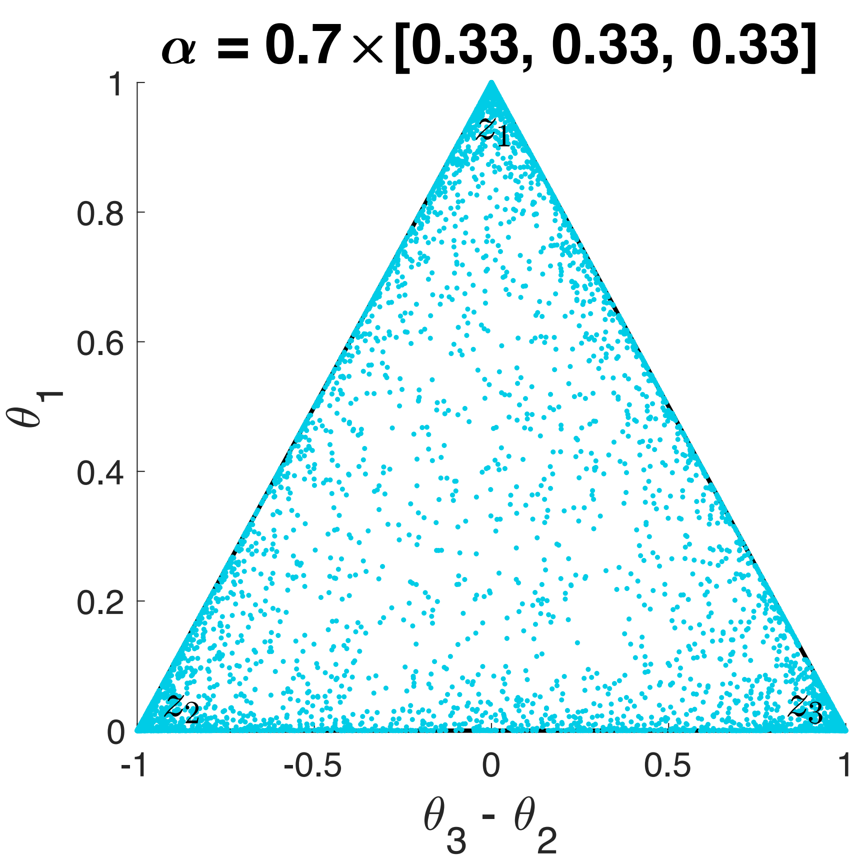


Imbalance in $\boldsymbol{\alpha}$ shapes the focus of the distribution

LDA — Dirichlet distribution (Dir)

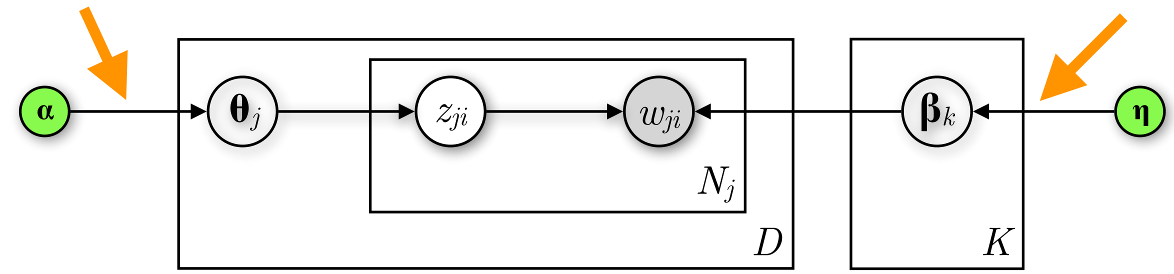


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

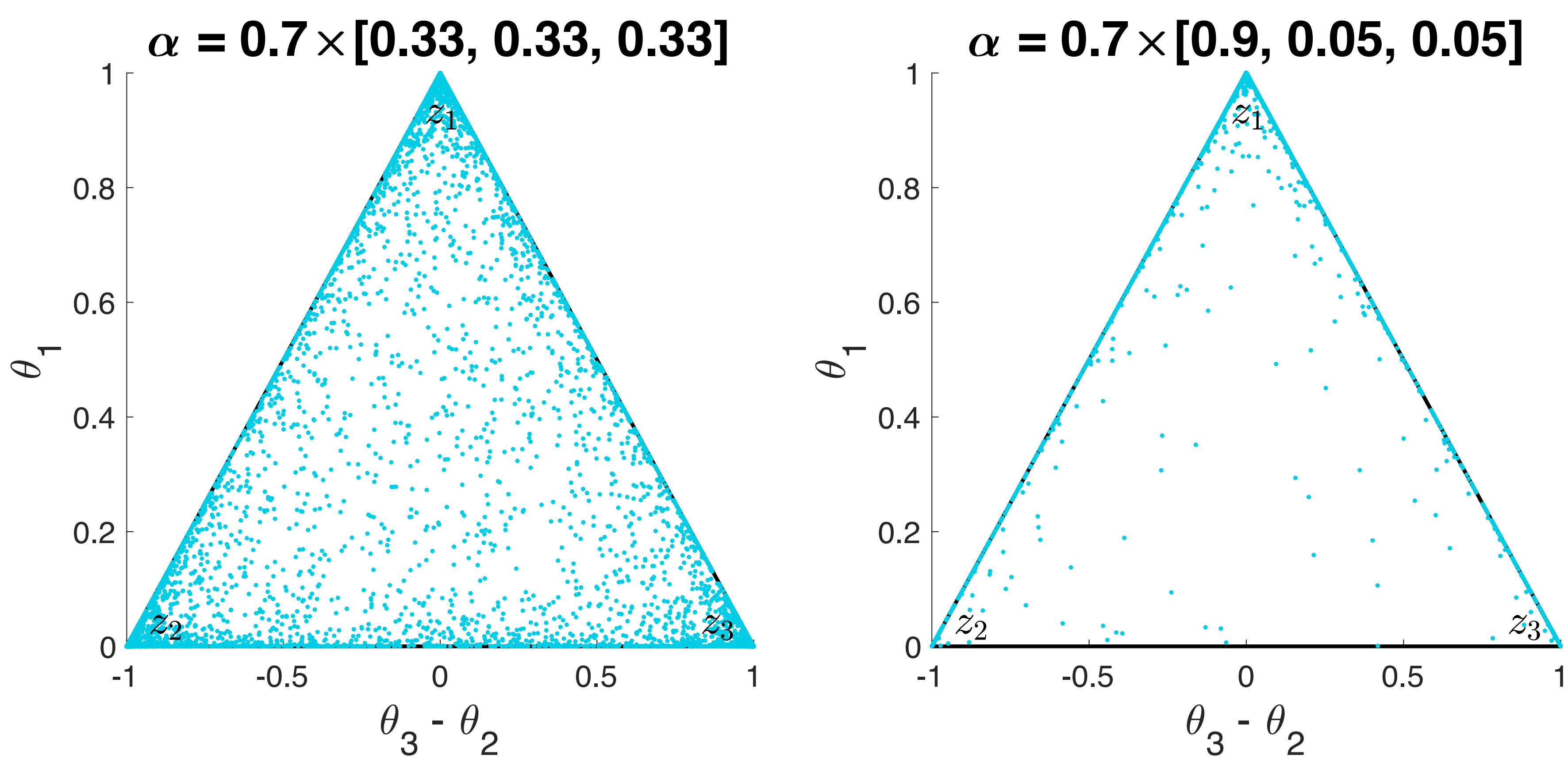


Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

LDA — Dirichlet distribution (Dir)

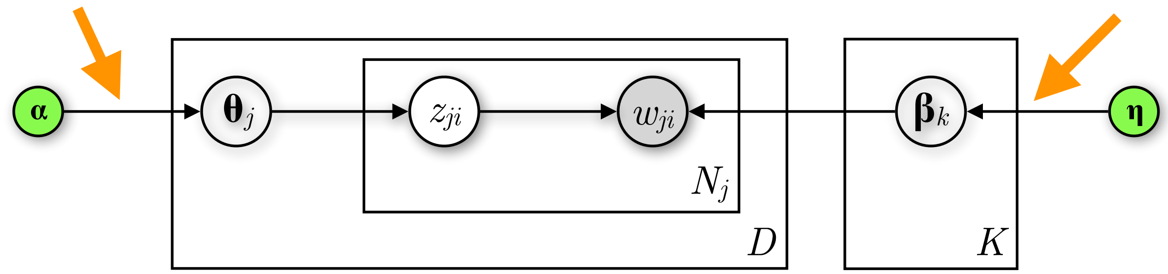


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

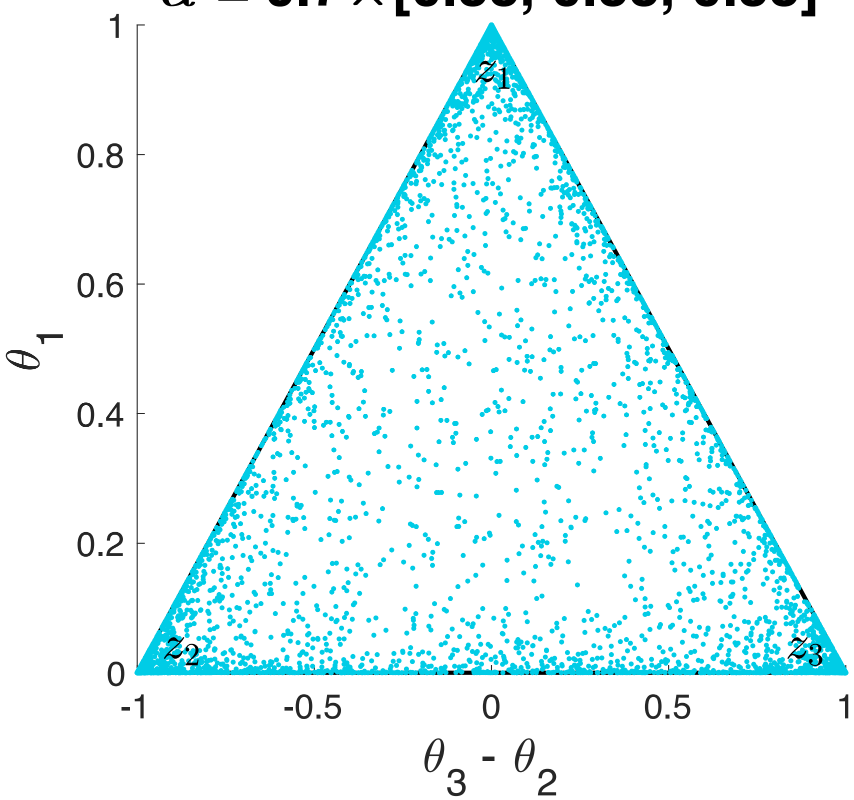


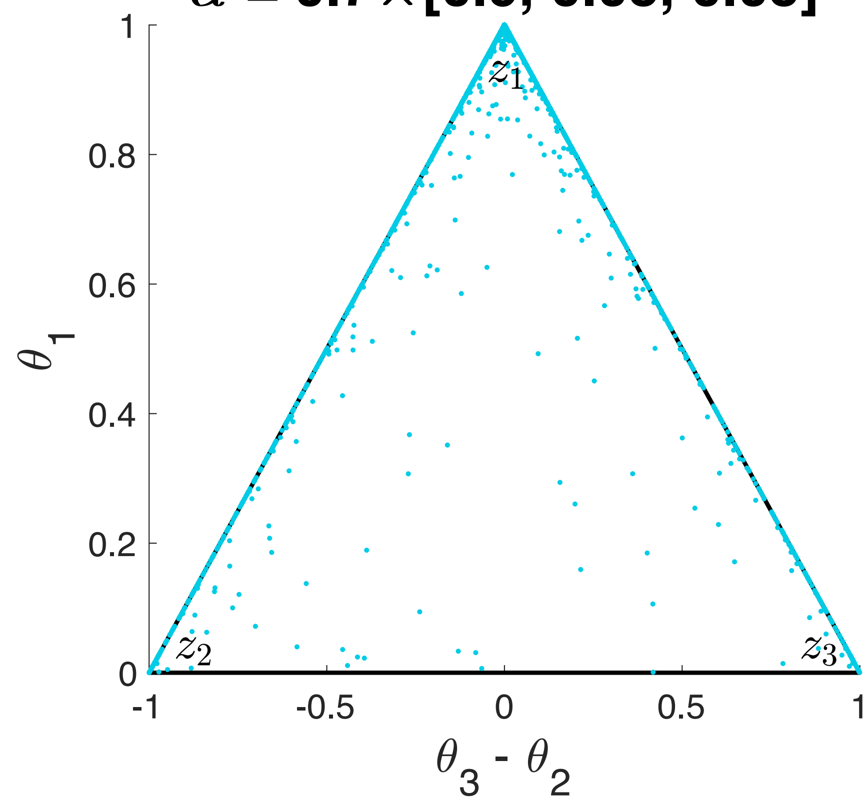
Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

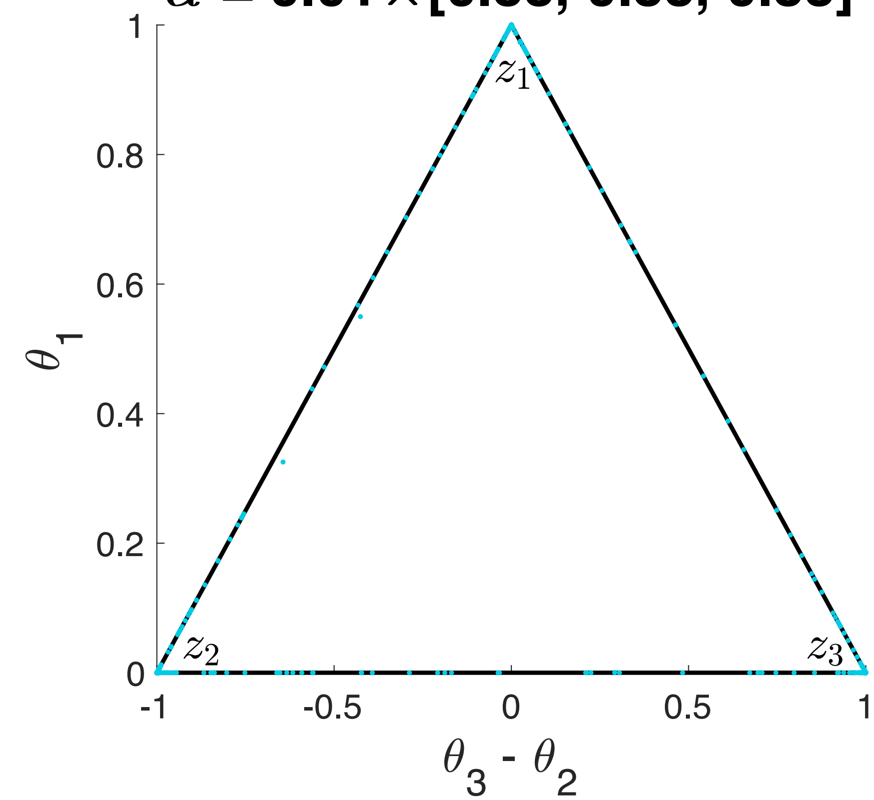
LDA — Dirichlet distribution (Dir)



Assume a simplex $\boldsymbol{\theta} = [\theta_1,\ \theta_2,\ \theta_3]$ across $K = 3$ topics. How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

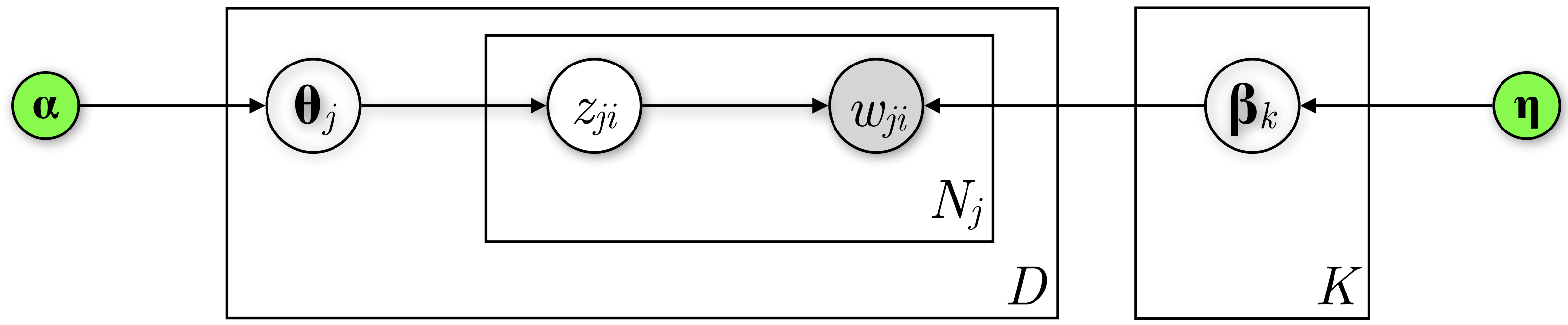






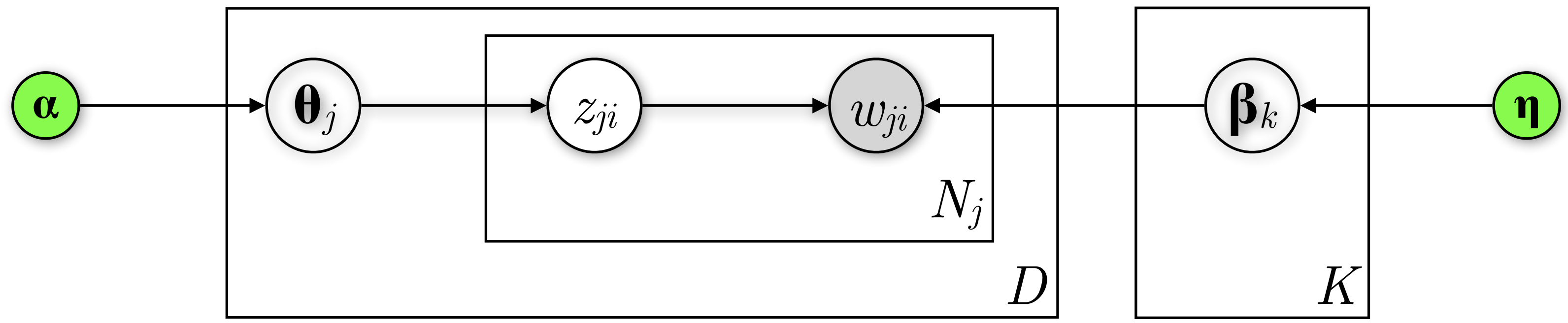
Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

LDA — Why combine Dir and Mult distributions?



- The Dirichlet distribution is conjugate to the Multinomial distribution
- Posterior $p(\boldsymbol{\beta}|\boldsymbol{\eta}, w)$ and prior $p(\boldsymbol{\beta}|\boldsymbol{\eta})$ belong to the same distribution family as the prior (Dirichlet) given that $p(w|\boldsymbol{\beta})$ is a Multinomial and $p(\boldsymbol{\beta}|\boldsymbol{\eta})$ a Dirichlet
- Abstracting the math, observed data (w) are adding to our prior intuition ($\boldsymbol{\eta}$) about how words relate with topics

LDA — Inference



Joint probability distribution

$$p(w, \boldsymbol{\theta}, \boldsymbol{\beta}, z | \boldsymbol{\alpha}, \boldsymbol{\eta}) = \prod_{k=1}^{K} p(\boldsymbol{\beta}_k | \boldsymbol{\eta}) \prod_{j=1}^{D} p(\boldsymbol{\theta}_j | \boldsymbol{\alpha}) \left(\prod_{i=1}^{N_j} p(z_{ji} | \boldsymbol{\theta}_j)\, p(w_{ji} | \boldsymbol{\beta}_{1:K}, z_{ji}) \right)$$

Posterior of the latent variables

$$p(\boldsymbol{\theta}, \boldsymbol{\beta}, z | w, \boldsymbol{\alpha}, \boldsymbol{\eta}) = \frac{p(\boldsymbol{\theta}, \boldsymbol{\beta}, z, w | \boldsymbol{\alpha}, \boldsymbol{\eta})}{\int_{\boldsymbol{\beta}} \int_{\boldsymbol{\theta}} \sum_z p(\boldsymbol{\theta}, \boldsymbol{\beta}, z, w | \boldsymbol{\alpha}, \boldsymbol{\eta})}$$

can't compute $\rightarrow$ approximate inference

LDA — Inference; Gibbs sampling

- Initialise probabilities randomly or uniformly
- In each step, replace the value of one of the variables by a value drawn from the distribution of that variable conditioned on the values of the remaining variables
- Repeat until convergence

Initialise x_i, $i = 1,\ldots,N$
For $t = 1,\ldots,T$:

$$\text{Sample } x_1^{(t+1)} \sim p\left(x_1 \,|\, x_2^{(t)}, \ldots, x_N^{(t)}\right)$$

$$\text{Sample } x_2^{(t+1)} \sim p\left(x_2 \,|\, x_1^{(t+1)}, x_3^{(t)}, \ldots, x_N^{(t)}\right)$$

$\ldots$

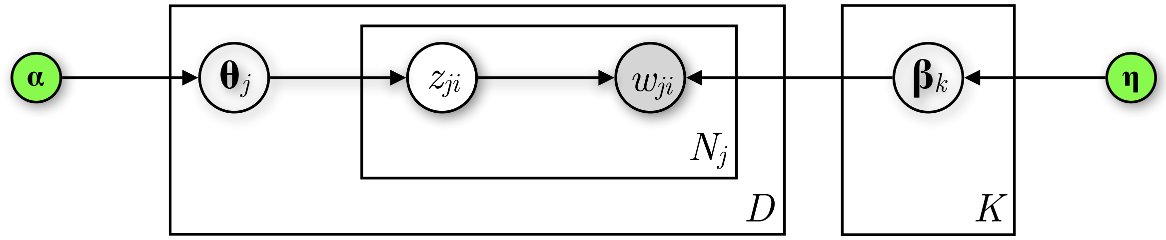
$$\text{Sample } x_j^{(t+1)} \sim p\left(x_j \,|\, x_1^{(t+1)}, x_2^{(t+1)}, \ldots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \ldots, x_N^{(t)}\right)$$

$\ldots$

$$\text{Sample } x_N^{(t+1)} \sim p\left(x_N \,|\, x_1^{(t+1)}, \ldots, x_{N-1}^{(t+1)}\right)$$

LDA — Inference; Gibbs sampling

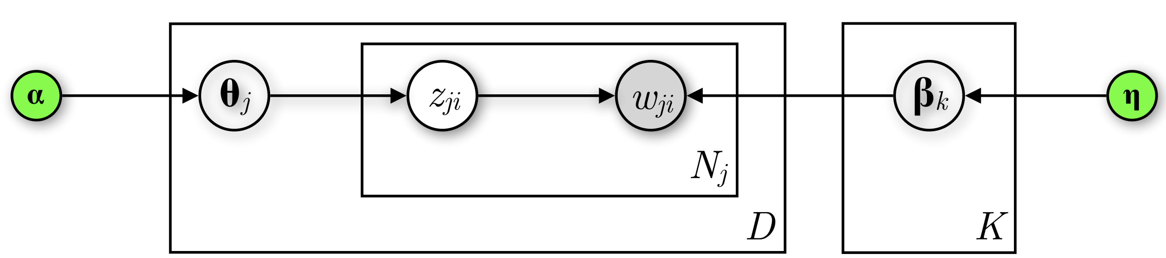
- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*note conventional notation for the exclusion of* w_{ji})



LDA — Inference; Gibbs sampling

- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*note conventional notation for the exclusion of* w_{ji})

$$p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \frac{n_{j,k,-i} + \alpha_k}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}} \cdot \frac{m_{k,w_{ji},-i} + \eta_{w_{ji}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}$$



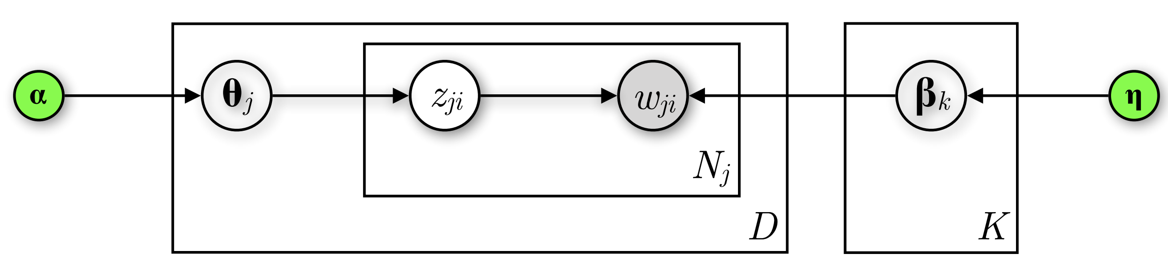
LDA — Inference; Gibbs sampling

- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*note conventional notation for the exclusion of* w_{ji})

$$p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \frac{\boxed{n_{j,k,-i} + \alpha_k}}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}} \cdot \frac{\boxed{m_{k,w_{ji},-i} + \eta_{w_{ji}}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}$$

\# topic k is assigned to a word in document j without counting the current word

\# word w_{ji} is associated with topic k in all documents without counting the current instance of w_{ji}



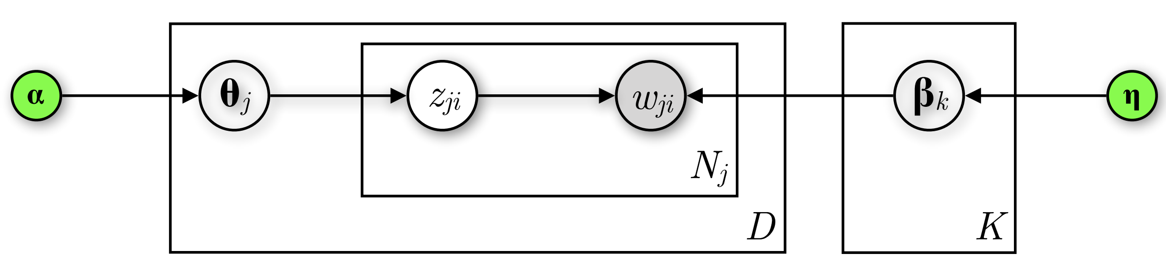
LDA — Inference; Gibbs sampling

- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*note conventional notation for the exclusion of* w_{ji})

$$p(z_{ji} = k|\mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \boxed{\frac{n_{j,k,-i} + \alpha_k}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}}} \cdot \boxed{\frac{m_{k,w_{ji},-i} + \eta_{w_{ji}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}}$$

How much does document j "like" topic k?

How much does topic k "like" word w_{ji}?



LDA — Inference; Gibbs sampling

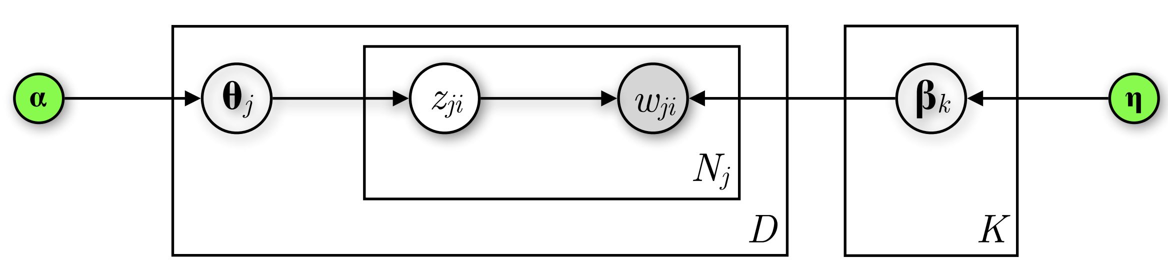
- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*note conventional notation for the ex*

How much does topic k “like” word w_{ji}?

$$p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \boxed{\frac{n_{j,k,-i} + \alpha_k}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}}} \cdot \boxed{\frac{m_{k,w_{ji},-i} + \eta_{w_{ji}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}}$$

How much does document j “like” topic k?

- From the above conditional distribution, sample a topic and set it as the new topic assignment z_{ji} of w_{ji}



LDA — Gibbs sampling; toy example

— **Consider $\boldsymbol{K = 3}$ topics**

LDA — Gibbs sampling; toy example

— Consider $K = 3$ topics

— **Sampling from document** $\boldsymbol{j}$ (word order doesn't matter)

document j

z_{ji}	?	?	?	?	?
w_{ji}	Brexit	deficit	Europe	market	single

LDA — Gibbs sampling; toy example

— Consider $K = 3$ topics

— Sampling from document j (word order doesn't matter)

— **Randomly assign topics to all words in document *j*** (and all other docs)

document j

z_{ji}	**3**	?	?	?	?
w_{ji}	Brexit	deficit	Europe	market	single

LDA — Gibbs sampling; toy example

— Consider $K = 3$ topics

— Sampling from document j (word order doesn't matter)

— **Randomly assign topics to all words in document $\boldsymbol{j}$** (and all other docs)

document j

z_{ji}	3	**2**	?	?	?
w_{ji}	Brexit	deficit	Europe	market	single

LDA — Gibbs sampling; toy example

— Consider $K = 3$ topics

— Sampling from document j (word order doesn't matter)

— **Randomly assign topics to all words in document** $\boldsymbol{j}$ (and all other docs)

document j

z_{ji}	3	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

LDA — Gibbs sampling; toy example

- Consider $K = 3$ topics
- Sampling from document j (word order doesn't matter)
- Randomly assign topics to all words in document j (and all other docs)
- **Update the word-topic counts for all documents**

document j

z_{ji}	3	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

word-topic counts across all documents

words / topics	1	2	3
Brexit	100	30	2
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
...	...	...	...

LDA — Gibbs sampling; toy example

- Consider $K = 3$ topics
- Sampling from document j (word order doesn't matter)
- Randomly assign topics to all words in document j (and all other docs)
- Update the word-topic counts for all documents
- **Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts**

document j

z_{ji}	**3**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

word-topic counts across all documents

words / topics	1	2	3
Brexit	100	30	**2**
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
...	...	...	...

LDA — Gibbs sampling; toy example

— Consider $K = 3$ topics
— Sampling from document j (word order doesn't matter)
— Randomly assign topics to all words in document j (and all other docs)
— Update the word-topic counts for all documents
— **Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts**

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

word-topic counts across all documents

words / topics	1	2	3
Brexit	100	30	**2 - 1**
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
...	...	...	...

LDA — Gibbs sampling; toy example

- Consider $K = 3$ topics
- Sampling from document j (word order doesn't matter)
- Randomly assign topics to all words in document j (and all other docs)
- Update the word-topic counts for all documents
- **Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts**

document j

z_{ji}	?	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

word-topic counts across all documents

words / topics	1	2	3
Brexit	100	30	**1**
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
...	...	...	...

LDA — Gibbs sampling; toy example

- …
- Randomly assign topics to all words in document j (and all other docs)
- Update the word-topic counts for all documents
- Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts
- **What are the *revised* topic proportions in document j?**

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

$$p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \boxed{\frac{n_{j,k,-i} + \alpha_k}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}}} \cdot \frac{m_{k,w_{ji},-i} + \eta_{w_{ji}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}$$

LDA — Gibbs sampling; toy example

- …
- Update the word-topic counts for all documents
- Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts
- What are the *revised* topic proportions in document j?
- **How much does each topic "like" the word "Brexit"?**

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	**1**
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…

$$p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta}) \propto \frac{n_{j,k,-i} + \alpha_k}{\sum_{k'=1}^{K} n_{j,k',-i} + \alpha_{k'}} \cdot \boxed{\frac{m_{k,w_{ji},-i} + \eta_{w_{ji}}}{\sum_{v=1}^{V} m_{k,v,-i} + \eta_v}}$$

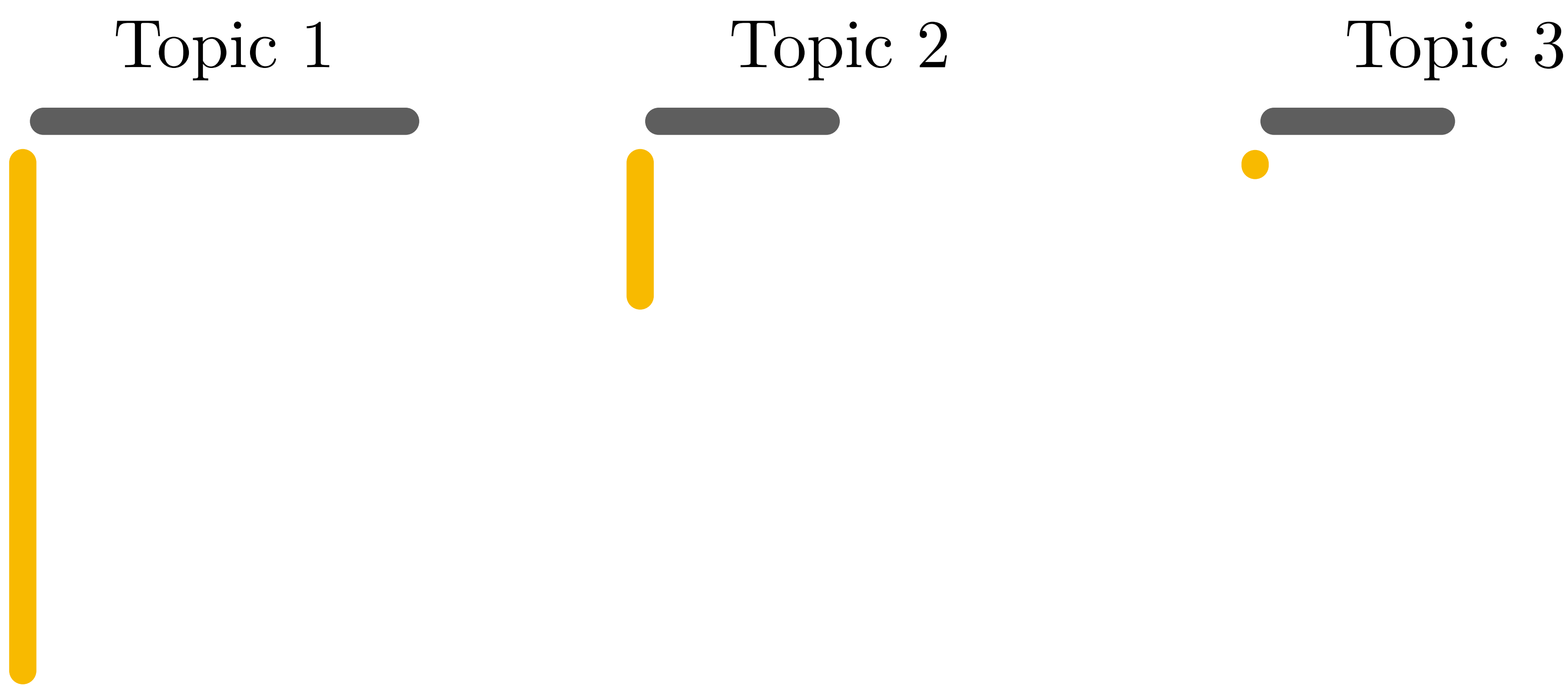
LDA — Gibbs sampling; toy example

— …
— Update the word-topic counts for all documents
— Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts
— What are the *revised* topic proportions in document j?
— **How much does each topic "like" the word "Brexit"?**

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	**1**
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



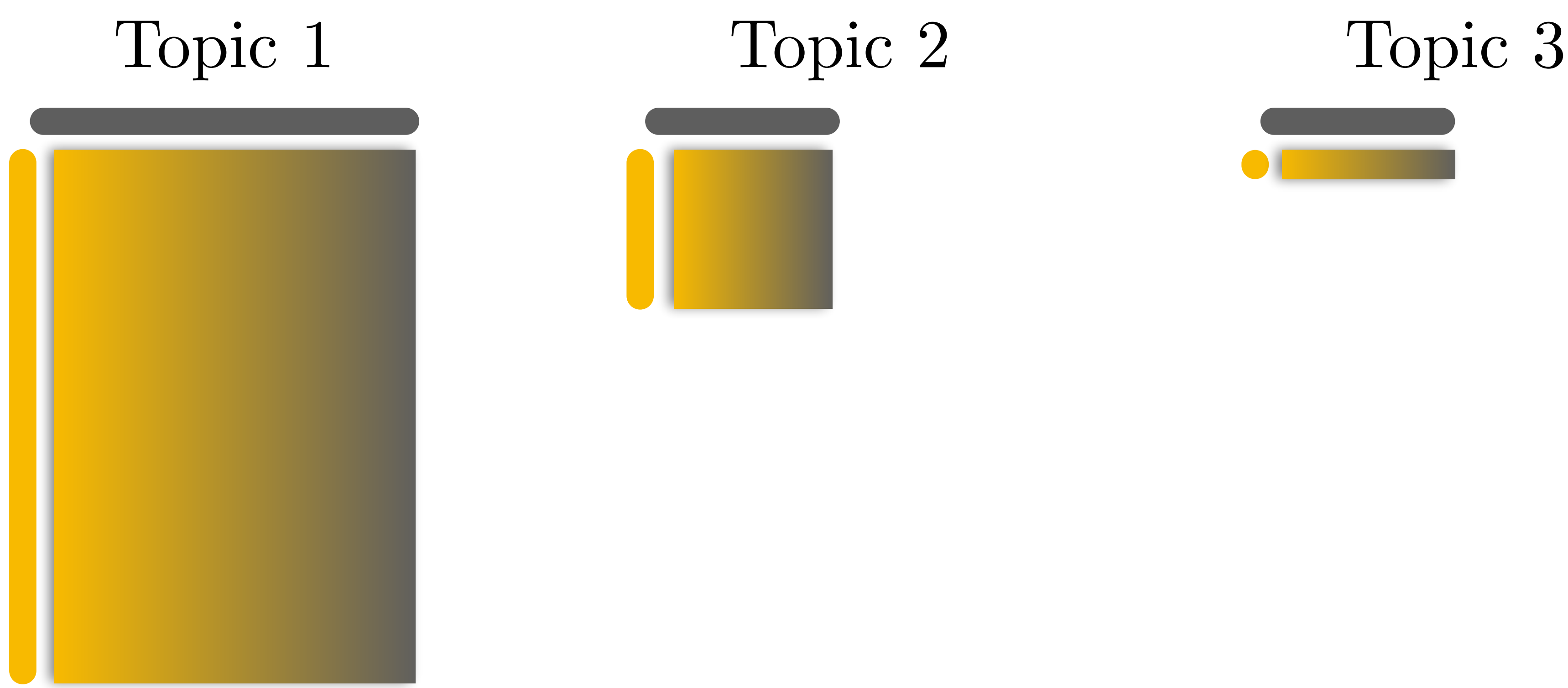
LDA — Gibbs sampling; toy example

— ...
— Update the word-topic counts for all documents
— Sample the first word (“Brexit”) in document j; unassign it from topic 3 and decrement its count in the word-topic counts
— What are the *revised* topic proportions in document j?
— **How much does each topic “like” the word “Brexit”?**

document j

z_{ji}	?	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
...	...	...	...



LDA — Gibbs sampling; toy example

— …

— Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts

— What are the *revised* topic proportions in document j?

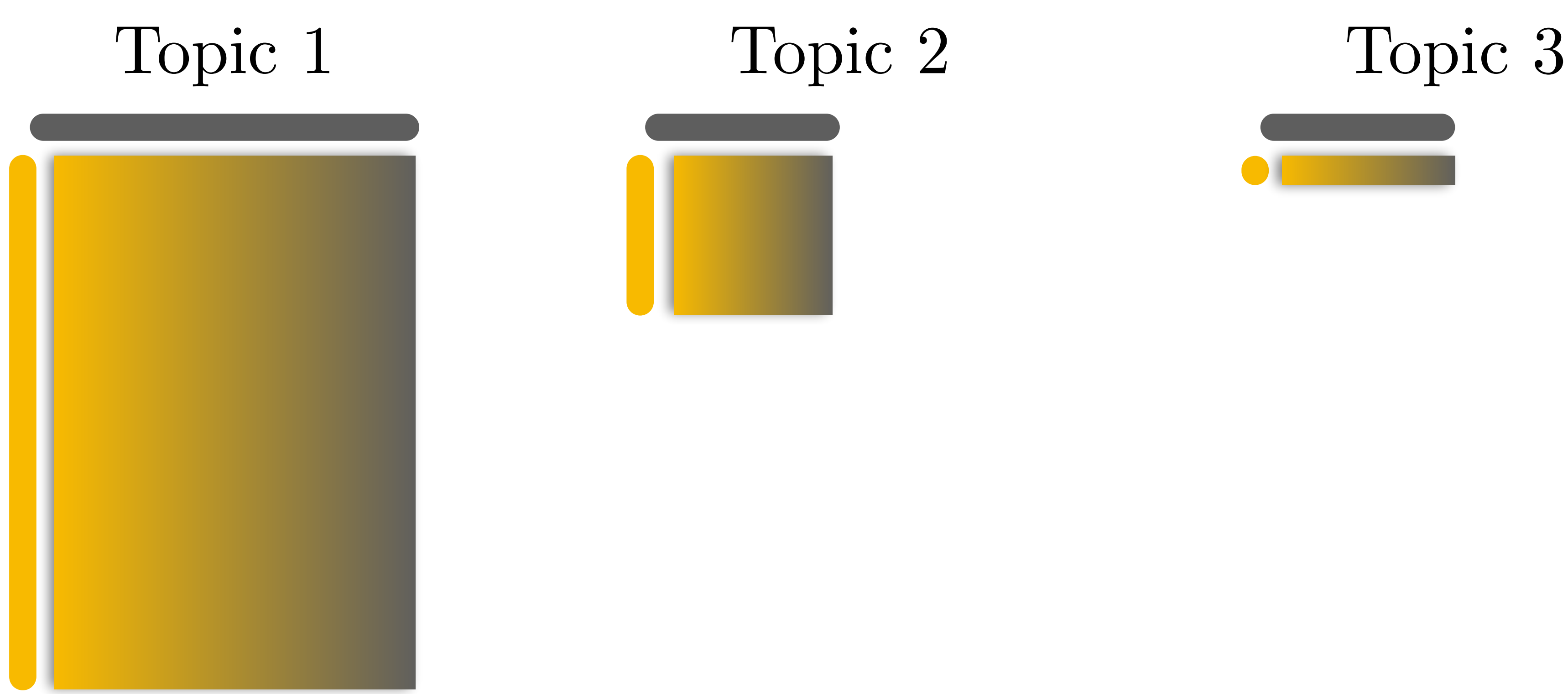
— How much does each topic "like" the word "Brexit"?

— **Sample from the *revised* conditional distribution** $p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta})$

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



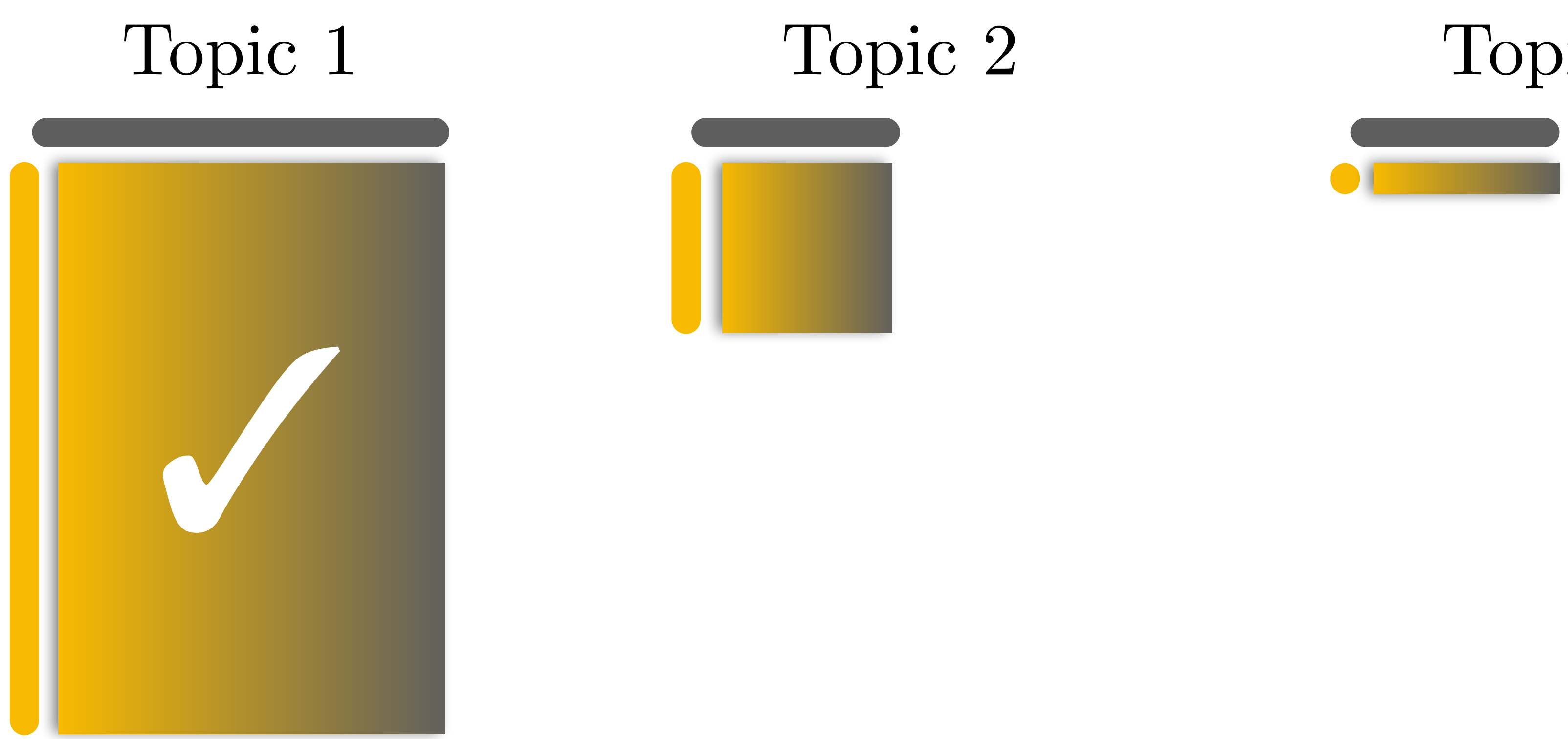
LDA — Gibbs sampling; toy example

- …
- Sample the first word (“Brexit”) in document j; unassign it from topic 3 and decrement its count in the word-topic counts
- What are the *revised* topic proportions in document j?
- How much does each topic “like” the word “Brexit”?
- **Sample from the *revised* conditional distribution** $p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta})$

document j

z_{ji}	**?**	2	3	1	1
w_{ji}	**Brexit**	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



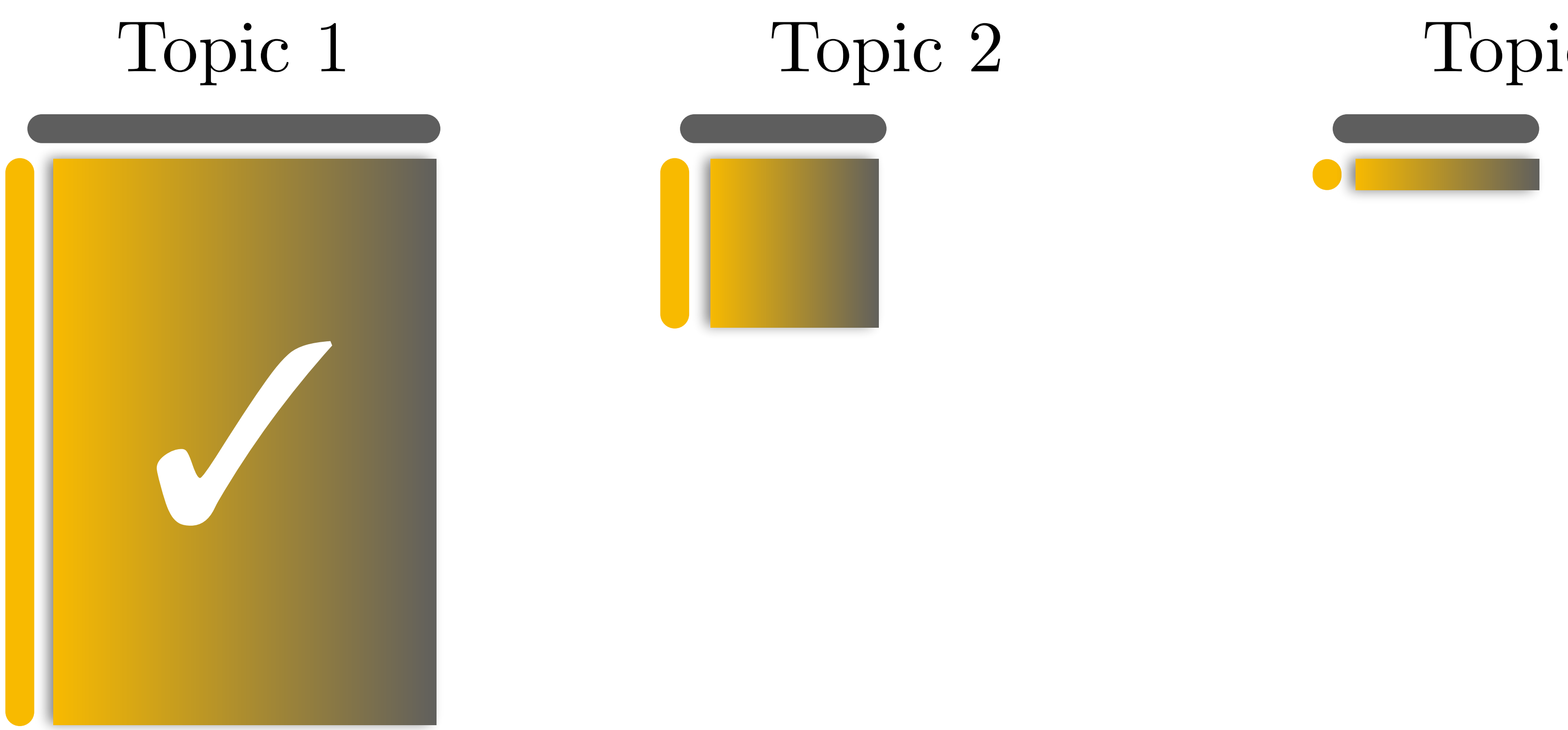
LDA — Gibbs sampling; toy example

— …
— Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts
— What are the *revised* topic proportions in document j?
— How much does each topic "like" the word "Brexit"?
— Sample from the *revised* conditional distribution $p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta})$
— **Assign the sampled topic to the word "Brexit" and update counts**

document j

z_{ji}	?	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

words / topics	1	2	3
Brexit	100	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



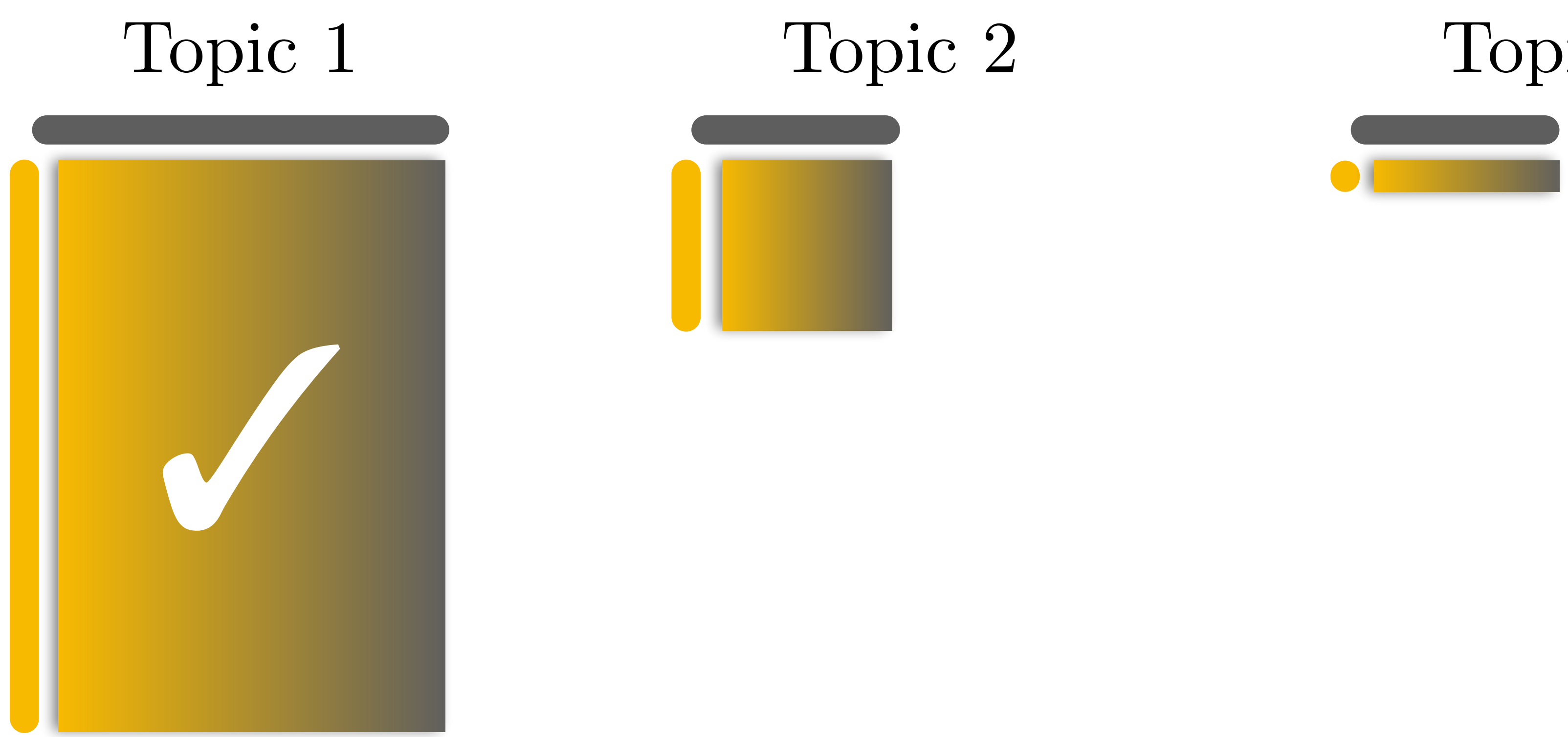
LDA — Gibbs sampling; toy example

— …
— Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts
— What are the *revised* topic proportions in document j?
— How much does each topic "like" the word "Brexit"?
— Sample from the *revised* conditional distribution $p(z_{ji} = k | \mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta})$
— **Assign the sampled topic to the word "Brexit" and update counts**

document j

z_{ji}	1	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

words / topics	1	2	3
Brexit	101	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



LDA — Gibbs sampling; toy example

— …

— Sample the first word ("Brexit") in document j; unassign it from topic 3 and decrement its count in the word-topic counts

— What are the *revised* topic proportions in document j?

— How much does each topic "like" the word "Brexit"?

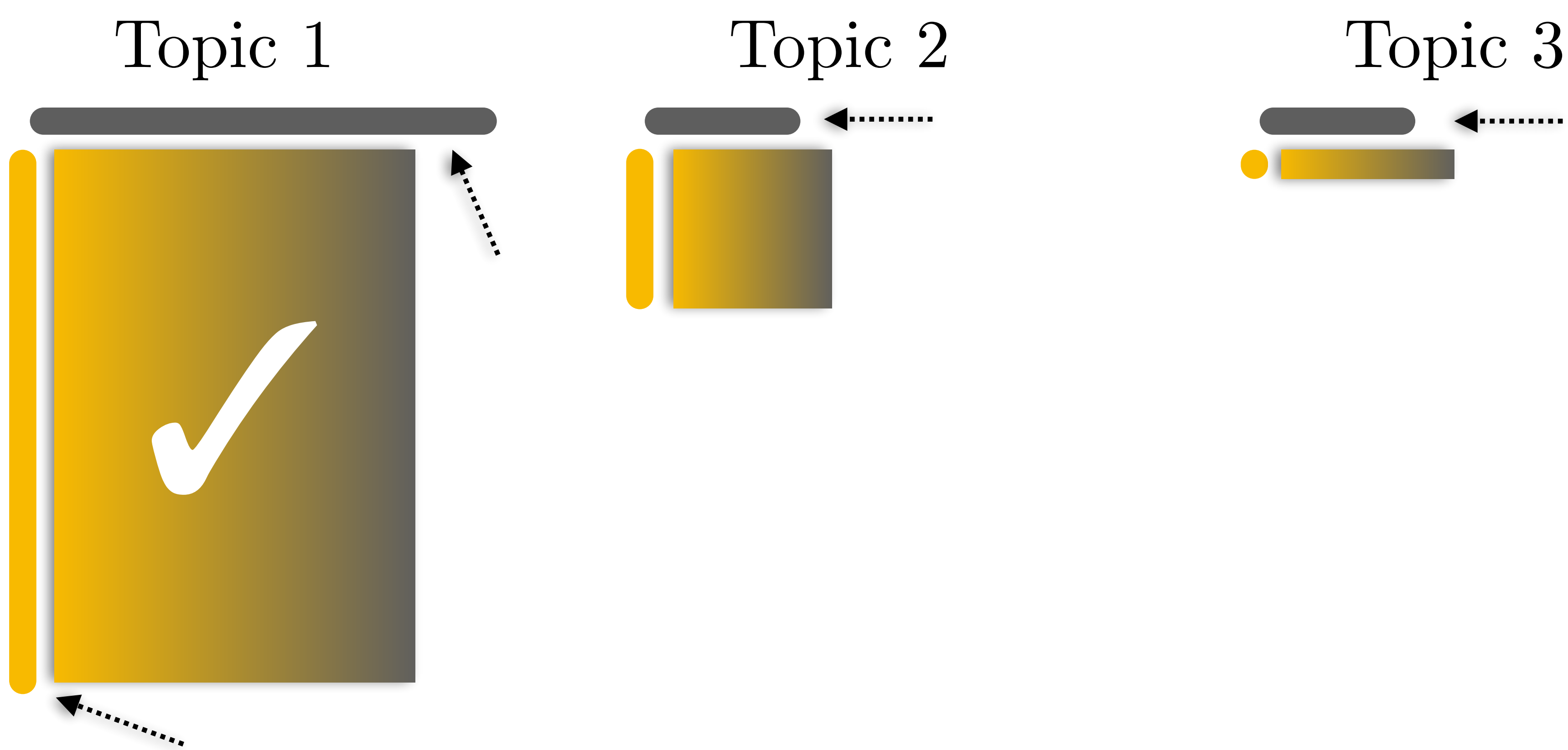
— Sample from the *revised* conditional distribution $p(z_{ji} = k|\mathbf{z}_{j,-i}, \mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\eta})$

— **Assign the sampled topic to the word "Brexit" and update counts**

document j

z_{ji}	1	2	3	1	1
w_{ji}	Brexit	deficit	Europe	market	single

words / topics	1	2	3
Brexit	101	30	1
deficit	10	60	0
Europe	95	5	2
market	50	70	5
single	50	15	90
…	…	…	…



Evaluating topics

- **It depends** on what the topics are for!
- If they are generated for an end task with a measure-able performance, then we it makes sense to use this metric, *i.e.* the **performance of the end task** as a proxy for the value of the topic (Note: LDA tends to underperform in such settings)
- Compute the **probability of generating held-out documents** (*the higher the better*)
- **Word intrusion**: Show words from topics to human judges (*crowdsourcing*) with out-of-topic words inserted (intruders). How often can they identify the word that does not belong?

Part II

Words as vectors

- We've seen that documents can be represented as vectors of word frequencies
- **Words** can also be represented as multi-dimensional **vectors**
- **Property** to exploit: words that occur in similar contexts (co-occur) tend to have similar meanings

"**You shall know a word by the company it keeps**"
J. R. Firth (1957)

Words as vectors

- We've seen that documents can be represented as vectors of word frequencies
- **Words** can also be represented as multi-dimensional **vectors**
- **Property** to exploit: words that occur in similar contexts (co-occur) tend to have similar meanings

> "**You shall know a word by the company it keeps**"
> J. R. Firth (1957)

— My new **W** is much thinner than my previous one.
— I prefer to work from remote locations using a **W**.
— This old **W** has less RAM than my new smartphone.
— With a 15-inch display, it's not a **W** anymore!

Words as vectors

- **Property** to exploit: words that occur in similar contexts (co-occur) tend to have similar meanings

  — My new **W** is much thinner than my previous one.
  — I prefer to work from remote locations using a **W**.
  — This old **W** has less RAM than my new smartphone.
  — With a 15-inch display, it's not a **W** anymore!

- Co-occurs with: "my", "thinner", "remote", "smartphone", "RAM", "display"
- Occurs after: "my", "remote", "display"
- Occurs before: "thinner", "RAM", "smartphone"

Words as vectors

- **Property** to exploit: words that occur in similar contexts (co-occur) tend to have similar meanings

  — My new **W** is much thinner than my previous one.
  — I prefer to work from remote locations using a **W**.
  — This old **W** has less RAM than my new smartphone.
  — With a 15-inch display, it's not a **W** anymore!

- Co-occurs with: "my", "thinner", "remote", "smartphone", "RAM", "display"
- Occurs after: "my", "remote", "display"
- Occurs before: "thinner", "RAM", "smartphone"
- **W** = laptop / notebook

Words as vectors

- Generate a **word-word** matrix
  — a.k.a. **word-context** or **word co-occurrence** matrix
- If the size of our vocabulary (all words) is V, then the size of this matrix is commonly $V \times V$
- Each **cell** of the matrix counts how many times two words **co-occur** within a predefined context
- Possible **contexts**: entire document, a paragraph in a document, a sentence, a number of words (window, commonly $\pm$ 4 words)

… more succinct definition of **computer** science is the study…
… analysis and study of **algorithms**, discipline of computer science…
… the arrival of Japanese **mandarin** oranges signalled the real…
… of pomelo and mandarin, **orange** has genes from both…

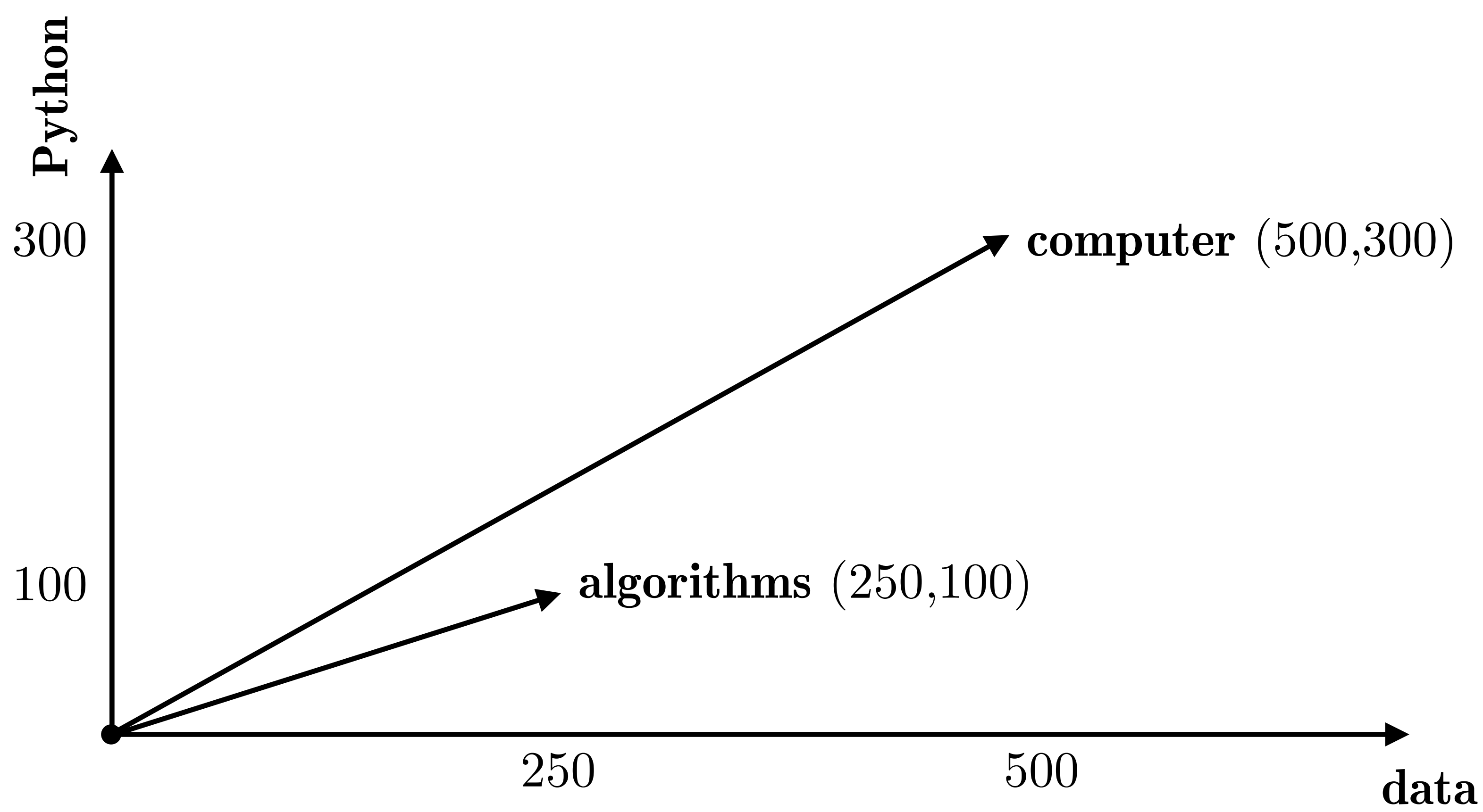
Words as vectors

word-word (word co-occurrence) matrix

	...	**data**	**...**	**fruit**	**...**	**Python**	**...**
...	...	...	...	...	...	...	...
algorithms	...	100	...	2	...	250	...
...	...	...	...	...	...	...	...
computer	...	300	...	5	...	200	...
...	...	...	...	...	...	...	...
mandarin	...	1	...	300	...	0	...
...	...	...	...	...	...	...	...
orange	...	1	...	256	...	10	...
...	...	...	...	...	...	...	...

Words as vectors

	...	data	...	fruit	...	Python	...
...	...	...	...	...	...	...	...
algorithms	...	**250**	...	2	...	**100**	...
...	...	...	...	...	...	...	...
computer	...	**500**	...	5	...	**300**	...
...	...	...	...	...	...	...	...
mandarin	...	1	...	300	...	0	...
...	...	...	...	...	...	...	...
orange	...	1	...	256	...	10	...
...	...	...	...	...	...	...	...



Words as vectors

- Recap: Word-context matrix of size $V \times V$ where V is the length of the vocabulary
- **Large** matrix as V could be even larger than 100,000
- **Sparse** matrix as many entries will be 0
  (not all words co-occur in all contexts)
- Small context window: a more **syntactic** representation
- Longer context window: a more **semantic** representation

Measuring word association — PMI

- Raw word counts are not the best measure for word association — skewed towards frequent/infrequent words, non discriminative
- **Pointwise Mutual Information** (**PMI**) is a measure of how often two events (co-)occur, compared to what we would expect if these events were independent
- Centre (target) word w_i, context word c_j

Measuring word association — PMI

- Raw word counts are not the best measure for word association — skewed towards frequent/infrequent words, non discriminative
- **Pointwise Mutual Information** (**PMI**) is a measure of how often two events (co-)occur, compared to what we would expect if these events were independent
- Centre (target) word w_i, context word c_j

$$\text{PMI}(w_i, c_j) = \log_2 \frac{p(w_i, c_j)}{p(w_i) \cdot p(c_j)}$$

- **Numerator:** How often we have seen the words together
- **Denominator:** How often we expect the words to co-occur, assuming they are independent
- **PMI:** how much more w_i, c_j co-occur than expected by chance

Positive Pointwise Mutual Information (PPMI)

- PMI ranges in $(-\infty, +\infty)$
- Negative PMI values are harder to interpret and evaluate; "relatedness" is easier to evaluate as opposed to "unrelatedness"
- Force positivity — Positive PMI (PPMI)

$$\mathrm{PPMI}(w_i, c_j) = \max\left(\log_2 \frac{p(w_i, c_j)}{p(w_i) \cdot p(c_j)}, 0\right)$$

Computing PPMI

Assume a word-context matrix **A** of size $V \times C$; generalisation of the word-word matrix, where the C contexts may not be identical to the V target words

$$\mathrm{PPMI}(w_i, c_j) = \max\left(\log_2 \frac{p(w_i, c_j)}{p(w_i) \cdot p(c_j)}, 0 \right)$$

$$p(w_i, c_j) = \frac{n_{ij}}{\sum_{i=1}^{V} \left(\sum_{j=1}^{C} n_{ij} \right)}$$

target word w_i co-occurs with context word c_j divided by the total count of word occurrences in the corpus

$$p(w_i) = \frac{\sum_{j=1}^{C} n_{ij}}{\sum_{i=1}^{V} \left(\sum_{j=1}^{C} n_{ij} \right)}$$

target word w_i appears in the corpus (sum of row i of **A**) divided by…

$$p(c_j) = \frac{\sum_{i=1}^{V} n_{ij}}{\sum_{i=1}^{V} \left(\sum_{j=1}^{C} n_{ij} \right)}$$

context word c_j appears in the corpus (sum of column j of **A**) divided by…

Measuring word similarity — Cosine

- Dot product between word vectors w, v: $\quad w^\top v = \sum_{i=1}^{N} w_i \cdot v_i$

Measuring word similarity — Cosine

- Dot product between word vectors w, v: $\quad w^\top v = \sum_{i=1}^{N} w_i \cdot v_i$

- Larger values for longer vectors and for frequent words
- Normalise it by dividing with the length of the vectors! Leads to cosine similarity, *i.e.* the cosine of the angle (ϕ) between the two vectors

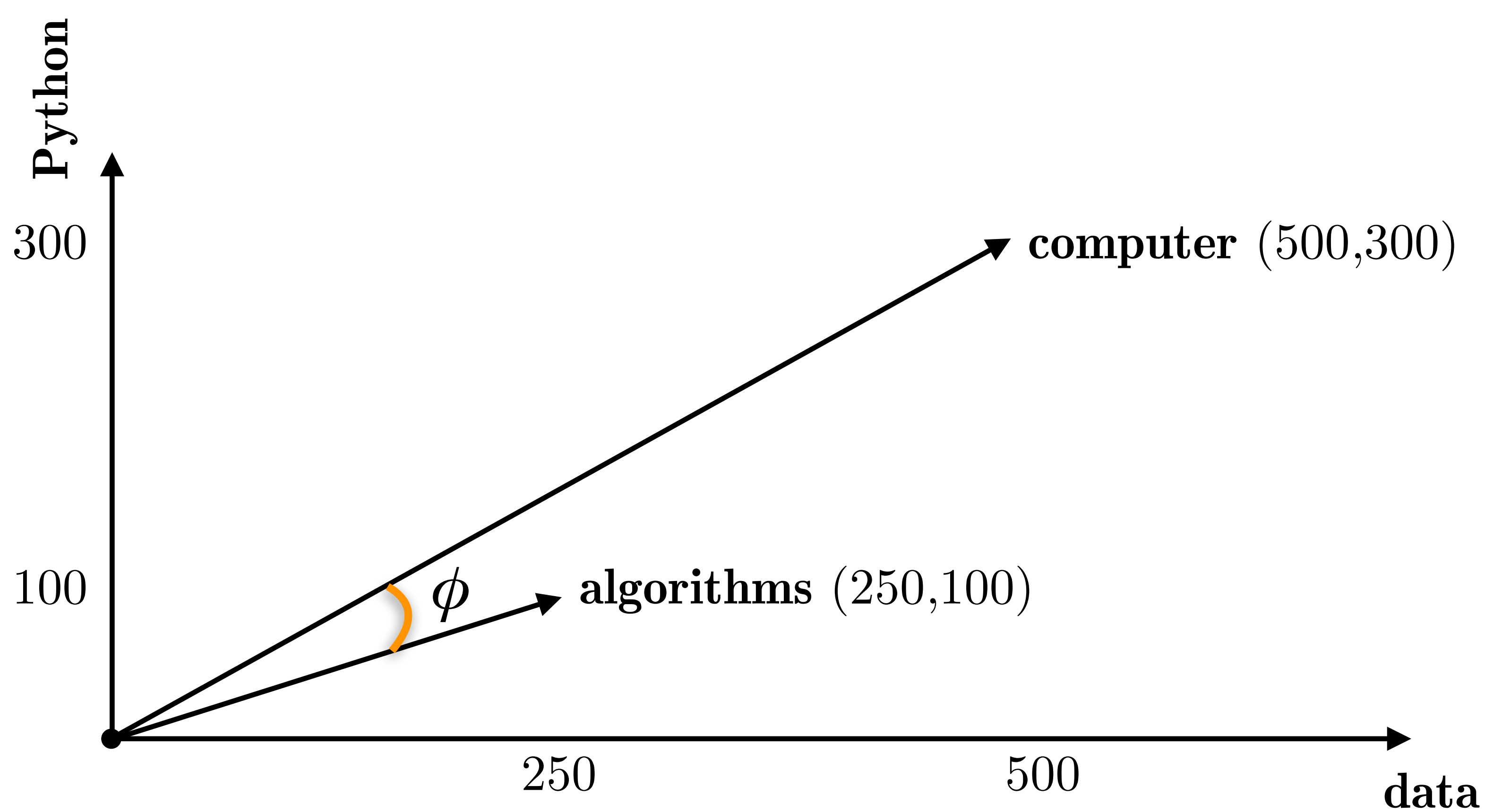
$$\text{cosine-sim}(w, v) = \frac{\sum_{i=1}^{N} w_i \cdot v_i}{\sqrt{\sum_{i=1}^{N} w_i^2} \cdot \sqrt{\sum_{i=1}^{N} v_i^2}} = \frac{w^\top v}{|w||v|} = \cos\phi$$

- Since w and $v > 0$, cosine-sim(w,v) ranges from [0,1]
  — cosine-sim(w,v) $= 0$ means that $\phi = 90°$
  — cosine-sim(w,v) $= 1$ means that $\phi = 0°$

Measuring word similarity — Cosine

$$\text{cosine-sim}(w, v) = \frac{\sum_{i=1}^{N} w_i \cdot v_i}{\sqrt{\sum_{i=1}^{N} w_i^2} \cdot \sqrt{\sum_{i=1}^{N} v_i^2}} = \frac{w^\top v}{|w||v|} = \cos \phi$$

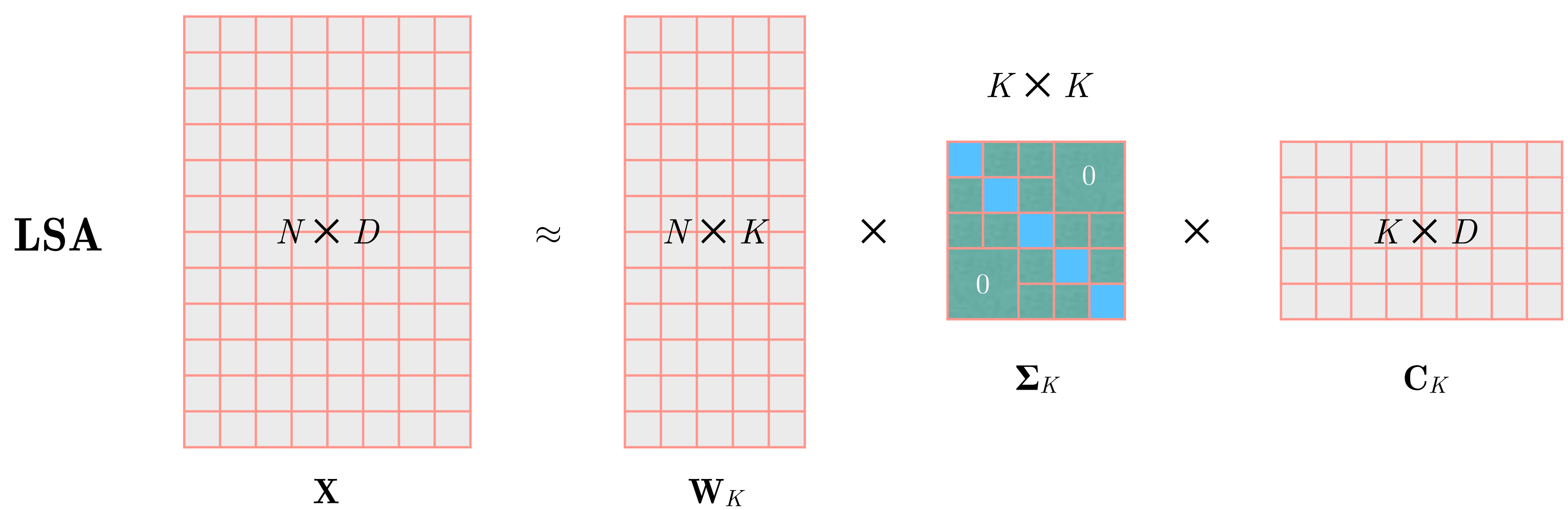
- Since w and $v > 0$, cosine-sim(w,v) ranges from $[0,1]$
  — cosine-sim$(w,v) = 0$ means that $\phi = 90°$
  — cosine-sim$(w,v) = 1$ means that $\phi = 0°$



cosine-sim(computer, algorithms) $= 0.9872$, $\phi = 9.162°$

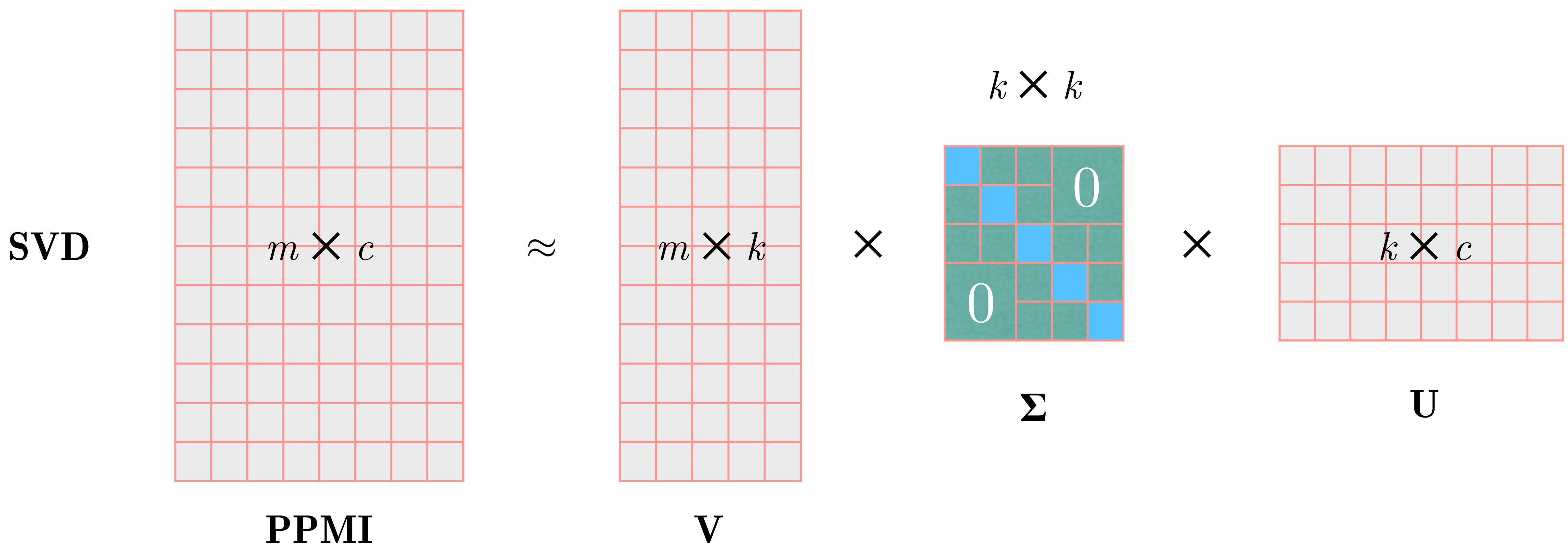
From sparse to dense word vectors

- Previously shown word representations: long (equal to size of the vocabulary V) and sparse (many 0's)
- Short and dense representations have advantages
  — easier to use as features in statistical learning methods
  — capture synonymy better
  — generalise better

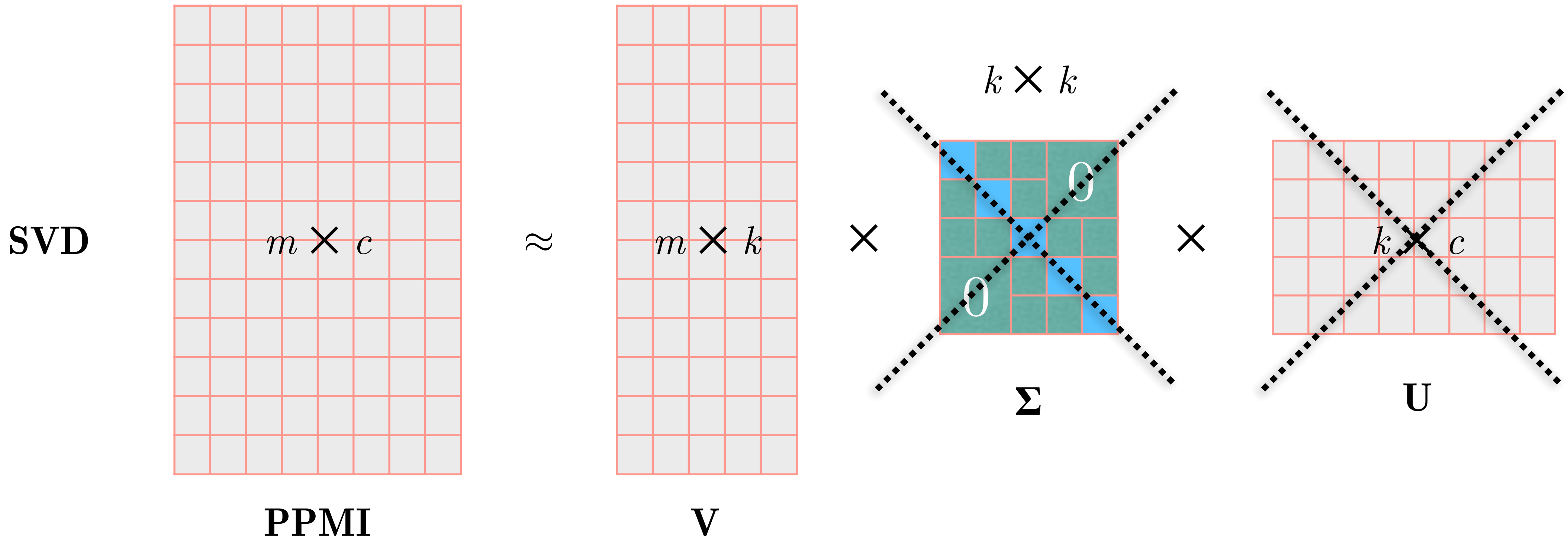


- Recall Latent Semantic Analysis (LSA), *i.e.* SVD on the word-document matrix ($\mathbf{X}$). What if we perform SVD on a word co-occurrence matrix?

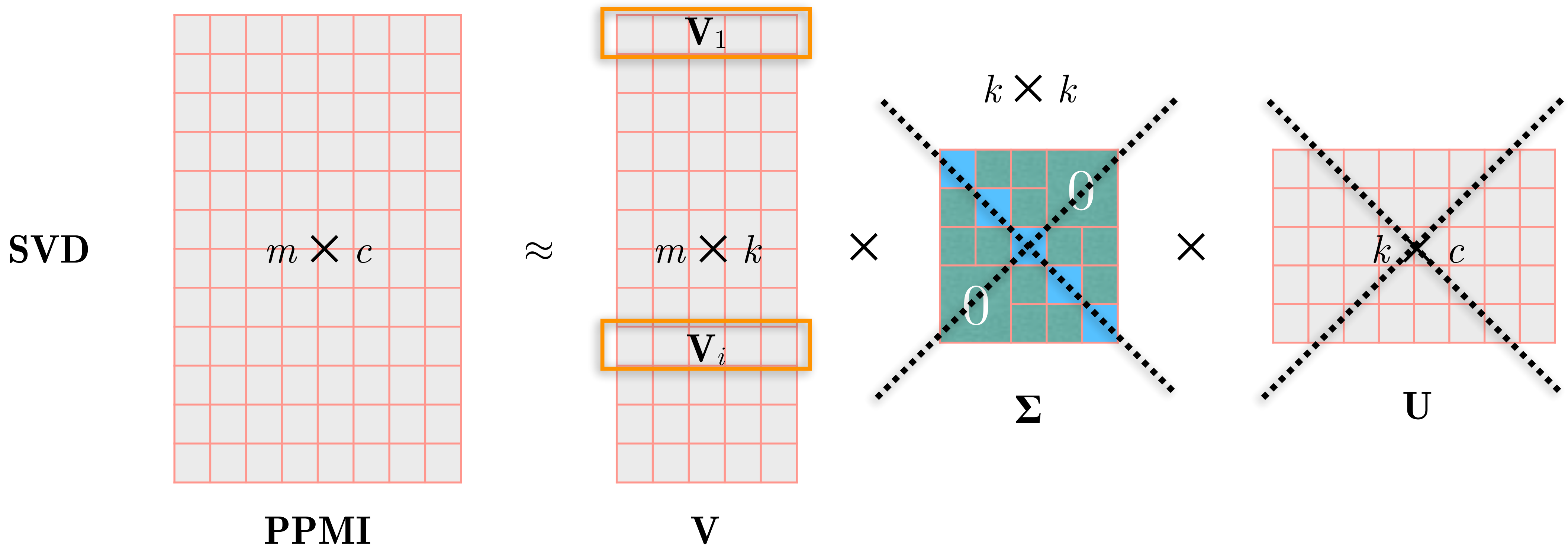
SVD on the PPMI word-context matrix



SVD on the PPMI word-context matrix



SVD on the PPMI word-context matrix



- $\mathbf{V}_i$ is a k-dimensional vector that represents word i in our vocabulary. It is also know as a **word embedding**. Commonly, $k = 300$, i.e. $\mathbf{V}_i$ is short and dense.
- SVD has a significant computational cost $O(mk^2)$.

Word embeddings from prediction

- Same intuition, *different* approach
  — words with similar meanings will co-occur
  — instead of counting co-occurrences, **predict** them

Word embeddings from prediction

- Same intuition, *different* approach
  — words with similar meanings will co-occur
  — instead of counting co-occurrences, **predict** them
- Popular example: **word2vec** — title of the software library, but there is a small family of methods behind it
  - ➡ Algorithms
    - skip-gram: Predict the context (surrounding) words based on a centre word
    - CBOW (continuous bag-of-words): Predict a centre word based on the context words
  - ➡ Training methods
    - Hierarchical softmax
    - Negative sampling

Word embeddings from prediction

- Same intuition, *different* approach
  — words with similar meanings will co-occur
  — instead of counting co-occurrences, **predict** them
- Popular example: **word2vec** — title of the software library, but there is a small family of methods behind it

➡ Algorithms
- **skip-gram:** Predict the context (surrounding) words based on a centre word
- CBOW (continuous bag-of-words): Predict a centre word based on the context words

➡ Training methods
- Hierarchical softmax
- <u>Negative sampling</u>
- **Naïve softmax**

word2vec — skip-gram

… said that "Hey Jude" is Beatles' most famous song, but…

word2vec — skip-gram

… said that "Hey Jude" is **Beatles**' most famous song, but…

↑

centre word

w_t

word2vec — skip-gram

context words

$w_{t-3},\ w_{t-2},\ w_{t-1}$

context words

$w_{t+1},\ w_{t+2},\ w_{t+3}$

… said that “Hey Jude” is **Beatles**’ most famous song, but…

centre word

w_t

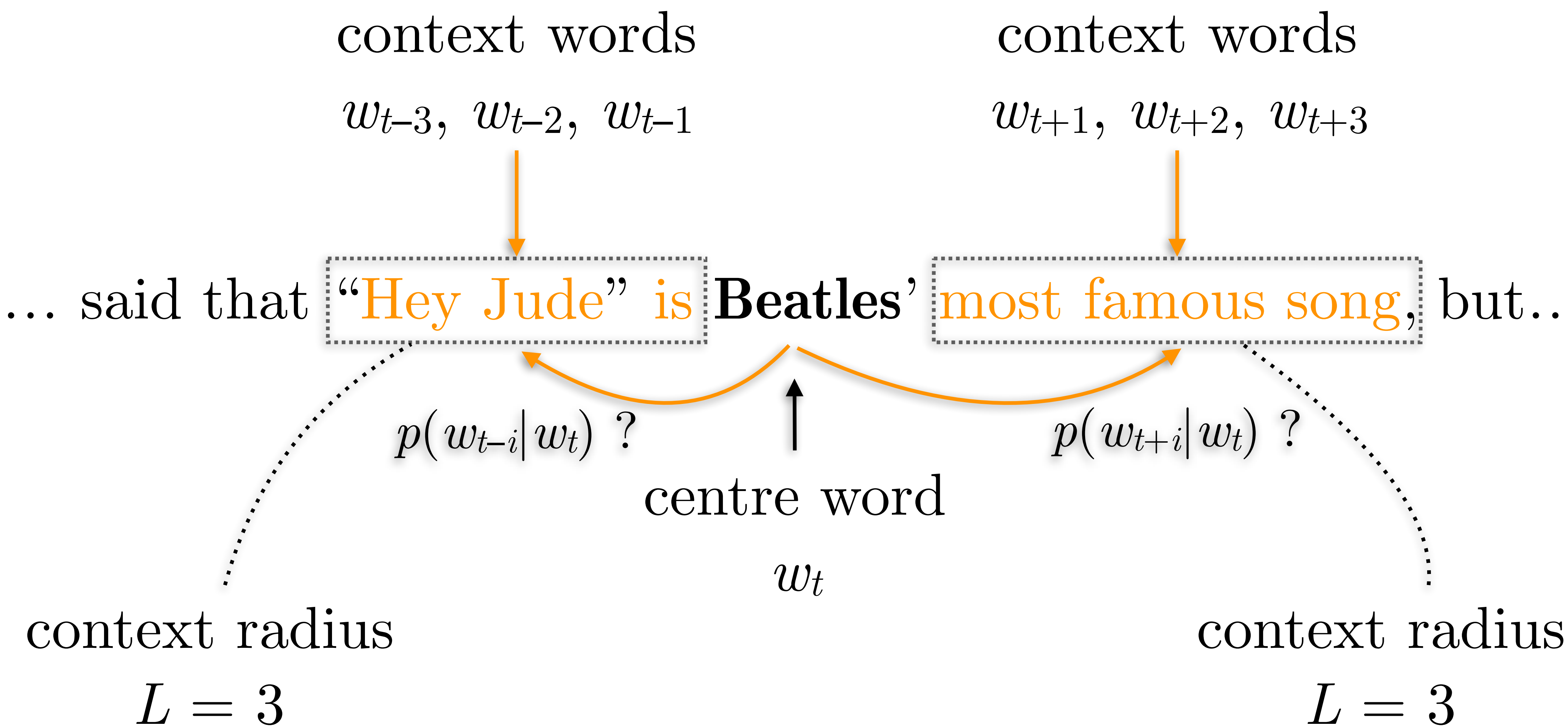
context radius

$L = 3$

context radius

$L = 3$

word2vec — skip-gram



skip-gram — Simplified objective function

For each word position t out of T predict the context words using a fixed radius L (or symmetric window $2L$)

Objective: Maximise the probability of any context word given the current centre word (position of surrounding words does not matter)

$$\max \prod_{t=1}^{T} \prod_{i=-L,\ i\neq 0}^{L} p\left(w_{t+i} \mid w_t\right)$$

Prefer to minimise things, and sums over products

Minimise the mean (across all T samples) negative log likelihood

$$\min \frac{1}{T}\left(-\sum_{t=1}^{T} \sum_{i=-L,\ i\neq 0}^{L} \log\left(p\left(w_{t+i} \mid w_t\right)\right)\right)$$

Aren't we missing something here?

skip-gram — Simplified objective function

For each word position t out of T predict the context words using a fixed radius L (or symmetric window $2L$)

Objective: Minimise the negative log likelihood of the probability of any context word given the current centre word

$$\min \frac{1}{T}\left(-\sum_{t=1}^{T} \sum_{i=-L,\ i\neq 0}^{L} \log\left(p\left(w_{t+i} \mid w_t\right)\right)\right)$$

(a) What exactly are we minimising?

(b) How are we going to minimise it?

skip-gram — Simplified objective function

For each word position t out of T predict the context words using a fixed radius L (or symmetric window $2L$)

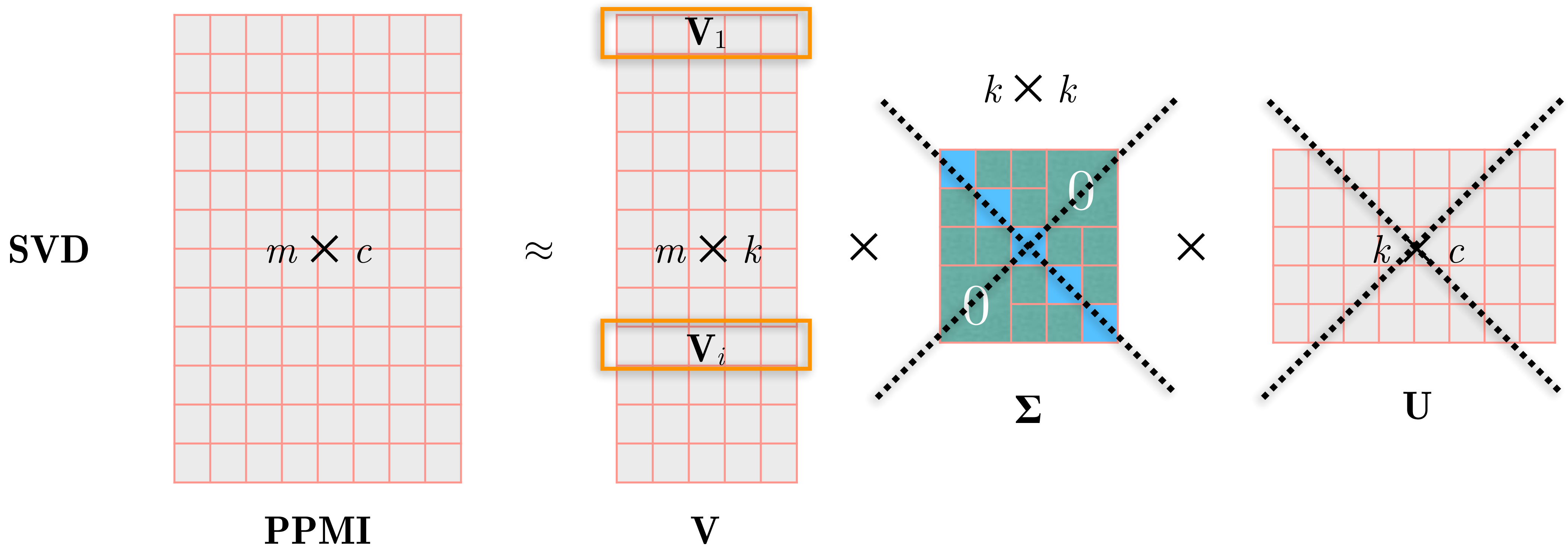
Objective: Minimise the negative log likelihood of the probability of any context word given the current centre word

$$\min \frac{1}{T}\left(-\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L}\log\left(p\left(w_{t+i}|\,w_t\right)\right)\right)$$

— Assume that each centre word (t) has a k-dimensional vector representation $\mathbf{v}_c$; all m words are held in an $k\times m$ matrix $\mathbf{V}$
— Assume that each context word has a k-dimensional vector representation $\mathbf{u}_x$; all m words are help in an $k\times m$ matrix $\mathbf{U}$
— Thus, the model parameters ($=2mk$) are now $\theta = [\mathbf{V}\ \mathbf{U}]$

$$\min_{\theta} \frac{1}{T}\left(-\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L}\log\left(p\left(w_{t+i}|\,w_t\,;\theta\right)\right)\right)$$

SVD on the PPMI word-context matrix



- $\mathbf{V}_i$ is a k-dimensional vector that represents word i in our vocabulary. It is also know as a **word embedding**. Commonly, $k = 300$, i.e. $\mathbf{V}_i$ is short and dense.
- SVD has a significant computational cost $O(mk^2)$.

skip-gram — Simplified objective function

For each word position t out of T predict the context words using a fixed radius L (or symmetric window $2L$)

Objective: Minimise the negative log likelihood of the probability of any context word given the current centre word

$$\min \frac{1}{T}\left(-\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L}\log\left(p\left(w_{t+i}\,|\,w_t\right)\right)\right)$$

— Assume that each centre word (t) has a k-dimensional vector representation $\mathbf{v}_c$; all m words are held in an $k\times m$ matrix $\mathbf{V}$

— Assume that each context word has a k-dimensional vector representation $\mathbf{u}_x$; all m words are help in an $k\times m$ matrix $\mathbf{U}$

— Thus, the model parameters ($=2mk$) are now $\theta = [\mathbf{V}\ \mathbf{U}]$

$$\min_{\theta} \frac{1}{T}\left(-\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L}\log\left(p\left(w_{t+i}\,|\,w_t\,;\theta\right)\right)\right)$$

skip-gram — Simplified objective function

$$\min_{\theta} \frac{1}{T} \left(- \sum_{t=1}^{T} \sum_{i=-L,\ i\neq 0}^{L} \log \left(p \left(w_{t+i} \,|\, w_t \,; \theta \right) \right) \right)$$

We need an estimate of the probability $p(w_{t+i}|w_t)$

Use a (*bad*) measure of similarity (dot product) and normalise it using a common approach in neural networks, the **softmax** function (squashes vector elements to a (0, 1) range)

Assuming a vocabulary of m words, for a centre word c ($\mathbf{v}_c$) and a conte**x**t word x ($\mathbf{u}_x$)

$$p(x \,|\, c) = \frac{\exp\left(\mathbf{u}_x^\top \mathbf{v}_c\right)}{\sum_{w=1}^{m} \exp\left(\mathbf{u}_w^\top \mathbf{v}_c\right)}$$

skip-gram — In practice...

$w_t = [0\,0\,\ldots\,1\,\ldots\,0]^\top$ centre word as an one-hot vector

$v_c = \mathbf{V} \cdot w_t$ get its vector representation (embedding) from the matrix of centre word embeddings

$o = \mathbf{U}^\top \cdot v_c$ dot product with all context word vectors
m (voc. size) × 1

$p_{w_i} = \text{softmax}(o)_i$ compute the softmax of this vector
this is the probability of word i, but we shall focus on the $2L$ context words

e.g. p_w

0.1
0.4
0.01
0.09
0.05
0.25
0.08
0.02

word2vec — skip-gram

context words

$w_{t-3}, \; w_{t-2}, \; w_{t-1}$

context words

$w_{t+1}, \; w_{t+2}, \; w_{t+3}$

… said that "Hey Jude" is **Beatles**' most famous song, but…

$p(w_{t-i} | w_t)$?

$p(w_{t+i} | w_t)$?

centre word

w_t

context radius

$L = 3$

context radius

$L = 3$

skip-gram — In practice…

$w_t = [0\,0\ \dots\ 1\ \dots\ 0]^\top$ centre word as an one-hot vector

$v_c = \mathbf{V} \cdot w_t$ get its vector representation (embedding) from the matrix of centre word embeddings

$o = \mathbf{U}^\top \cdot v_c$ dot product with all context word vectors m (voc. size) $\times$ 1

$p_{w_i} = \mathrm{softmax}(o)_i$ compute the softmax of this vector this is the probability of word i, but we shall focus on the $2L$ context words

$$\text{e.g.}\quad p_w \quad \begin{bmatrix} 0.1 \\ \mathbf{0.4} \\ 0.01 \\ 0.09 \\ 0.05 \\ \mathbf{0.25} \\ 0.08 \\ 0.02 \end{bmatrix} \longrightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \mathbf{1} \\ 0 \\ 0 \end{bmatrix}$$

but we also know the correct answer! In this case, we need to improve our embeddings (**V** and **U**). In neural nets by applying error backpropagation.

skip-gram — Negative sampling

Naïve / inefficient way for parameter inference

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L} \log\Big(p\left(w_{t+i}\,|\,w_t\,;\theta\right)\Big)$$

Gradient descent

$$\theta_{p+1} = \theta_p + \gamma\,\nabla_\theta J(\theta_p)$$

Too slow and computationally expensive. Recall:

The denominator is too expensive to compute (for large vocabularies)

$$p(x\,|\,c) = \frac{\exp\left(\mathbf{u}_x^\top \mathbf{v}_c\right)}{\sum_{w=1}^{m} \exp\left(\mathbf{u}_w^\top \mathbf{v}_c\right)}$$

Negative sampling: For each context word sample non-neighbouring words as "negative" samples

New objective: High dot product with context words and low dot product with "negative" samples

skip-gram — Stochastic gradient descent

Naïve / inefficient way for parameter inference

$$J(\theta) = -\frac{1}{T}\sum_{t=1}^{T}\sum_{i=-L,\ i\neq 0}^{L} \log\Big(p\left(w_{t+i}\,|\,w_t\,;\theta\right)\Big)$$

Gradient descent

$$\theta_{p+1} = \theta_p + \gamma\,\nabla_\theta J(\theta_p)$$

Too slow and computationally expensive.

Apply stochastic gradient descent.

i.e. instead of going through all the data for computing the gradient of $\nabla_\theta J(\theta)$

we use one or small subsets of the data (mini batches) to update the gradient

Word analogies with word embeddings

vector('*king*')–vector('*man*')+vector('*woman*') ≈ vector('*queen*')

More formally:

$$\arg\max_{b \in V} \left(\cos \left(b, a - a_p + b_p \right) \right)$$

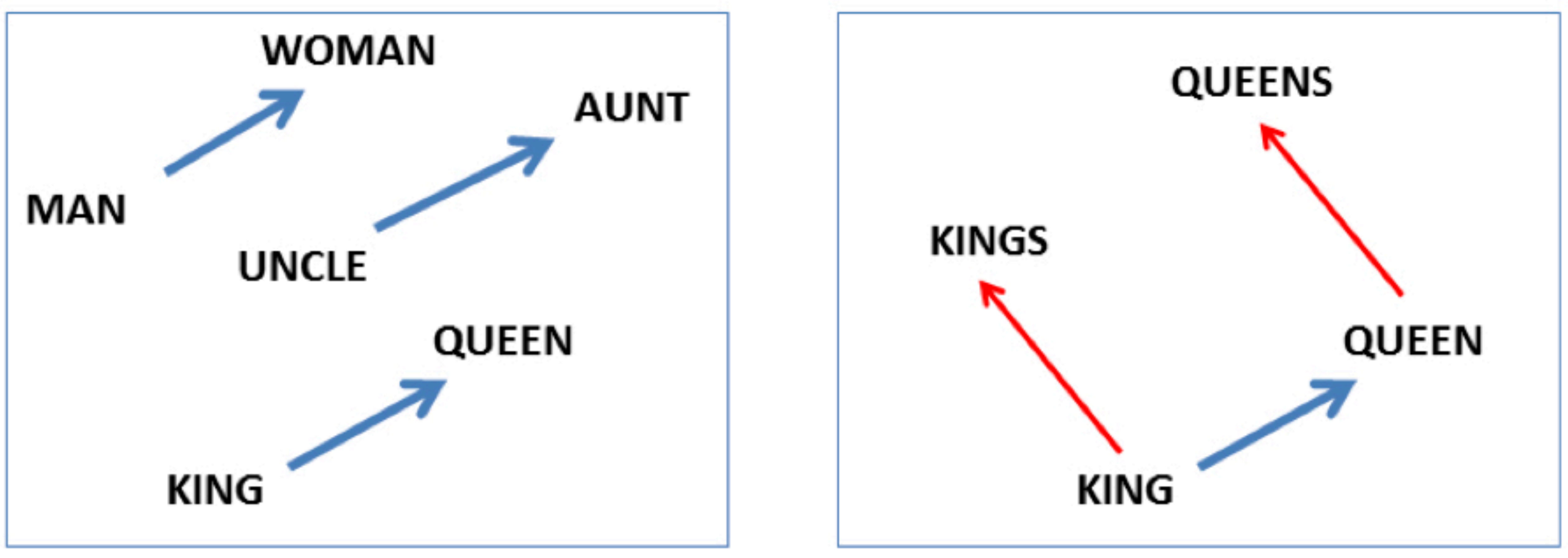
a=vector('*king*'), a_p=vector('*man*'), b_p = vector('*woman*')

If we compute the cosine similarity of $a - a_p + b_p$ with the embeddings of all the V words in our corpus, we expect b=vector('*queen*') to have the greatest one

a_p is for a what b_p is for b

Word analogies with word embeddings

vector('*king*')–vector('*man*')+vector('*woman*') ≈ vector('*queen*')



embeddings of all the V words in our corpus, we expect
b=vector('*queen*') to have the greatest one

a_p is for a what b_p is for b

Word embeddings based on UK Twitter data

— 1.1 billion tweets from 2012 to 2016
— approximately geolocated in the UK
— 512-dimensional embeddings for 470,194 words

Most similar words (top-5) to:

- **Monday:** Tuesday, Thursday, Wednesday, Friday, Sunday
- **January:** February, August, October, March, June
- **red:** yellow, blue, purple, pink, green
- **we:** they, you, we've, our, us
- **espresso:** expresso, cappuccino, macchiato, latte, coffee
- **linux:** Unix, Centos, Debian, Ubuntu, Redhat
- **retweet:** rt, tweet, retweets, retweeting, rewteet
- **democracy:** democratic, dictatorship, democracies, socialism, undemocratic
- **loool:** looool, lool, loooool, looooool, loooooool
- **xxxx:** xxxxx, xxx, xxxxxxxx, xxxxxx, xxxxxxx
- **enviroment:** environment, environments, env, enviro, habitats

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to … [**?**]

‘Rome’ is to ‘Italy’ what ‘London’ is to … [**?**]

‘go’ is for ‘went’ what ‘do’ is to… [**?**]

‘big’ is to ‘bigger’ what ‘small’ is to… [**?**]

‘poet’ is to ‘poem’ what ‘author’ is to… [**?**]

‘Messi’ is to ‘football’ what ‘Lebron’ is to… [**?**]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to… [**?**]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to… [**?**]

‘UK’ is for ‘Farage’ what ‘USA’ is to… [**?**]

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to … [**his**, him, himself]

‘Rome’ is to ‘Italy’ what ‘London’ is to … [**?**]

‘go’ is for ‘went’ what ‘do’ is to… [**?**]

‘big’ is to ‘bigger’ what ‘small’ is to… [**?**]

‘poet’ is to ‘poem’ what ‘author’ is to… [**?**]

‘Messi’ is to ‘football’ what ‘Lebron’ is to… [**?**]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to… [**?**]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to… [**?**]

‘UK’ is for ‘Farage’ what ‘USA’ is to… [**?**]

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to … [**his**, him, himself]

‘Rome’ is to ‘Italy’ what ‘London’ is to … [**UK**, Denmark, Sweden]

‘go’ is for ‘went’ what ‘do’ is to… [**?**]

‘big’ is to ‘bigger’ what ‘small’ is to… [**?**]

‘poet’ is to ‘poem’ what ‘author’ is to… [**?**]

‘Messi’ is to ‘football’ what ‘Lebron’ is to… [**?**]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to… [**?**]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to… [**?**]

‘UK’ is for ‘Farage’ what ‘USA’ is to… [**?**]

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to … [**his**, him, himself]

‘Rome’ is to ‘Italy’ what ‘London’ is to … [**UK**, Denmark, Sweden]

‘go’ is for ‘went’ what ‘do’ is to… [**did**, doing, happened]

‘big’ is to ‘bigger’ what ‘small’ is to… [**?**]

‘poet’ is to ‘poem’ what ‘author’ is to… [**?**]

‘Messi’ is to ‘football’ what ‘Lebron’ is to… [**?**]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to… [**?**]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to… [**?**]

‘UK’ is for ‘Farage’ what ‘USA’ is to… [**?**]

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to … [**his**, him, himself]

‘Rome’ is to ‘Italy’ what ‘London’ is to … [**UK**, Denmark, Sweden]

‘go’ is for ‘went’ what ‘do’ is to… [**did**, doing, happened]

‘big’ is to ‘bigger’ what ‘small’ is to… [**smaller**, larger, tiny]

‘poet’ is to ‘poem’ what ‘author’ is to… [**?**]

‘Messi’ is to ‘football’ what ‘Lebron’ is to… [**?**]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to… [**?**]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to… [**?**]

‘UK’ is for ‘Farage’ what ‘USA’ is to… [**?**]

Word embeddings based on UK Twitter data

download from figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650

‘she’ is to ‘her’ what ‘he’ is to ... [**his**, him, himself]

‘Rome’ is to ‘Italy’ what ‘London’ is to ... [**UK**, Denmark, Sweden]

‘go’ is for ‘went’ what ‘do’ is to... [**did**, doing, happened]

‘big’ is to ‘bigger’ what ‘small’ is to... [**smaller**, larger, tiny]

‘poet’ is to ‘poem’ what ‘author’ is to... [**novel**, excerpt, memoir]

‘Messi’ is to ‘football’ what ‘Lebron’ is to... [**basketball**, bball, NBA]

‘Elvis’ is to ‘Presley’ what ‘Aretha’ is to... [**Franklin**, Ruffin, Vandross]

‘UK’ is for ‘Brexit’ what ‘Greece’ is to... [**Grexit**, Syriza, Tsipras]

‘UK’ is for ‘Farage’ what ‘USA’ is to... [**Trump**, Farrage, Putin]

Part III

Predicting judicial decisions of the ECtHR

- Predict the outcome of a case tried by the European Court of Human Rights (ECtHR), *e.g.* whether an article of the European Convention on Human Rights has been violated

- The observed data is specific parts from the proceedings of a case as recorded by the court. In particular:

  Procedure
  The facts
  — The circumstances of the case
  — Relevant law
  The law
  — Alleged violation of Article X
  —— Parties' submissions
  —— Merits

Case structure at ECtHR

Procedure: This section contains the procedure followed before the Court, from the lodging of the individual application until the judgment was handed down

PROCEDURE

1. The case originated in an application (no. 35355/08) against the Republic of Bulgaria lodged with the Court under Article 34 of the Convention for the Protection of Human Rights and Fundamental Freedoms ("the Convention") by a Bulgarian national, Ms Gana Petkova Velcheva ("the applicant"), on 30 June 2008.
2. The applicant was represented by Mr M. Ekimdzhiev and Ms G. Chernicherska, lawyers practising in Plovdiv. The Bulgarian Government ("the Government") were represented by their Agent, Ms Y. Stoyanova, of the Ministry of Justice.
3. The applicant alleged that the authorities had failed to comply with a final court judgment allowing her claim for restitution of agricultural land.
4. On 7 May 2013 the application was communicated to the Government.

Case structure at ECtHR

Facts → Circumstances of the case: This section comprises all material which is not considered as belonging to points of law, i.e., legal arguments

THE FACTS

I. THE CIRCUMSTANCES OF THE CASE

5. The applicant was born in 1927 and lives in the village of Ribaritsa.

6. Her father, of whom she is the sole heir, owned agricultural land in the area surrounding the village which was incorporated into an agricultural cooperative at the beginning of the 1950s.

7. In 1991, following the adoption of the Agricultural Land Act ("the ALA", see paragraph 17 below), the applicant applied for the land's restitution.

8. By a decision dated 10 March 1999 the land commission dealing with the case refused to restore her rights to two plots of 900 and 2,000 square metres respectively, noting that sheep pens had been built on them by the agricultural cooperative. It held that the applicant was entitled to compensation in lieu of restitution.

Data and textual features

Article	Human Right	Cases
3	Prohibits torture and inhuman and degrading treatment	250
6	Protects the right to a fair trial	80
8	Provides a right to respect for one's "private and family life, his home and his correspondence"	254

- ***n*-grams**
  Use the 2,000 most frequent *n*-grams, where n $= \{1,\ldots, 4\}$
  Different frequencies for different parts of the case
- **Topics**
  — Convert the document (case)-word matrix to a **word-word** matrix using cosine similarity between all pairs of word representations (frequencies) across the documents (cases)
  — Perform **spectral clustering** on the word-word matrix to obtain (hard) word clusters (30)

Prediction accuracy

Feature Type		Article 3	Article 6	Article 8	Average
N-grams	Full	.70 (.10)	.82 (.11)	.72 (.05)	.75
	Procedure	.67 (.09)	.81 (.13)	.71 (.06)	.73
	Circumstances	.68 (.07)	.82 (.14)	.77 (.08)	.76
	Relevant law	.68 (.13)	.78 (.08)	.72 (.11)	.73
	Facts	.70 (.09)	.80 (.14)	.68 (.10)	.73
	Law	.56 (.09)	.68 (.15)	.62 (.05)	.62
Topics		**.78 (.09)**	.81 (.12)	.76 (.09)	.78
Topics and circumstances		.75 (.10)	**.84 (0.11)**	**.78 (0.06)**	**.79**

n-gram features on the "Circumstances" of a case provide a strong performance (76%)

Topics (on the "Full" proceedings) perform better (78%)

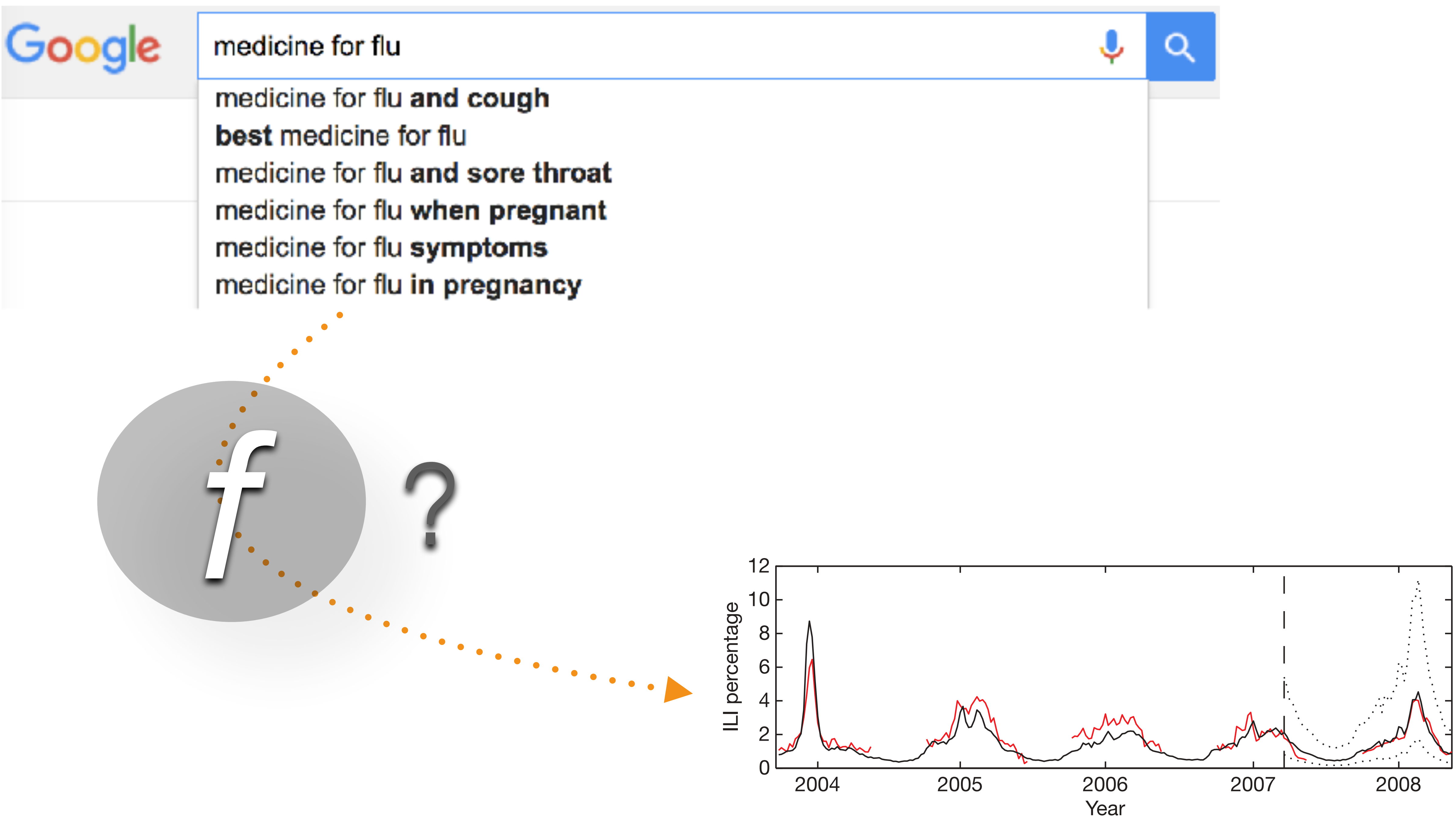
Combining the two categories of features in a linear ensemble yields the overall best performance (79%)

Article 3 — Topic weights

(*prohibits torture and inhuman and degrading treatment*)

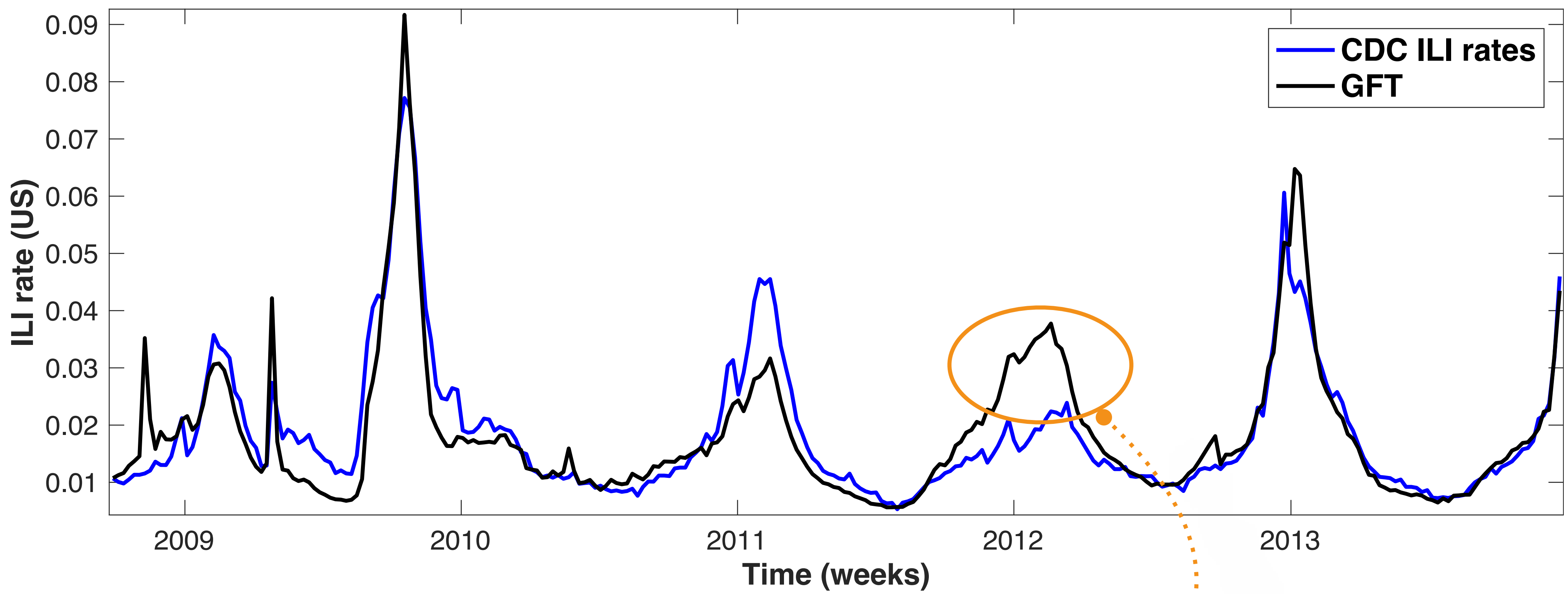
Topic	**Most frequent *n*-grams**	***w***
Positive state obligations	injury, protection, ordered, damage, civil, caused, failed, claim, course, connection	13.5
Detention conditions	prison, detainee, visit, well, regard, cpt, access, food, situation, problem	11.7
Treatment by state officials	police, officer, treatment, police officer, July, ill, force, evidence, ill treatment, arrest	10.2
Prior violation of Article 2	june, statement, three, dated, car, area, jurisdiction, gendarmerie, perpetrator, scene	-12.4
Issues of proof	witness, asked, told, incident, brother, heard, submission, arrived, identity, hand	-15.2
Sentencing	sentence, year, life, circumstance, imprisonment, release, set, president, administration, sentenced	-17.4

Inferring disease rates from Google search



Google proposed an infamous method…

… Google Flu Trends, that made some major mistakes, such as



“rsv” — 25%

“flu symptoms” — 18%

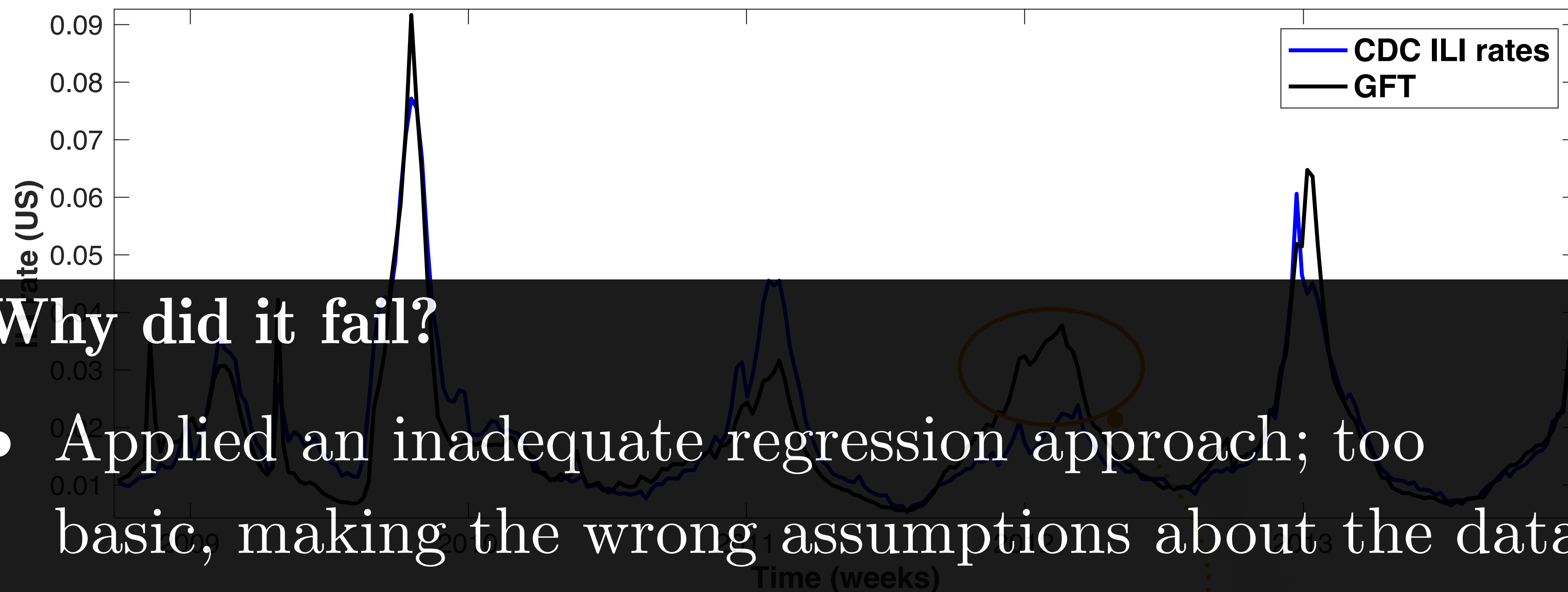
“benzonatate” — 6%

“symptoms of pneumonia” — 6%

“upper respiratory infection” — 4%

Google proposed an infamous method…

… Google Flu Trends, that made some major mistakes, such as



Why did it fail?

- Applied an inadequate regression approach; too basic, making the wrong assumptions about the data
- Did not care to model language at all
- Plus, it was not tested properly!

"rsv" — 25%
"flu symptoms" — 18%
"benzonatate" — 6%
"symptoms of pneumonia" — 6%
"upper respiratory infection" — 4%

A better way to select search queries

1. Learn **word embeddings** using Twitter data
2. **Query embedding** = Average token embedding
3. Derive a **concept** by specifying a **positive** (P) and a **negative** (N) **context** (sets of n-grams)
4. **Rank** all queries using their **similarity score** with this concept

query embedding

$$S(\mathcal{Q}, \mathcal{C}) = \frac{\sum_{i=1}^{k} \cos\left(\mathbf{e}_{\mathcal{Q}}, \mathbf{e}_{P_i}\right)}{\sum_{j=1}^{z} \cos\left(\mathbf{e}_{\mathcal{Q}}, \mathbf{e}_{N_j}\right) + \gamma}$$

embedding of a negative concept n-gram

constant to avoid division by 0

A better way to select search queries

Positive context	Negative context	Most similar queries
#flu fever flu flu medicine gp hospital	bieber ebola wikipedia	cold flu medicine flu aches cold and flu cold flu symptoms colds and flu
flu flu gp flu hospital flu medicine	ebola wikipedia	flu aches flu colds and flu cold and flu cold flu medicine

Hybrid combination with regression techniques

Embedding based feature selection (concept ranking) is an **unsupervised technique**, thus non optimal

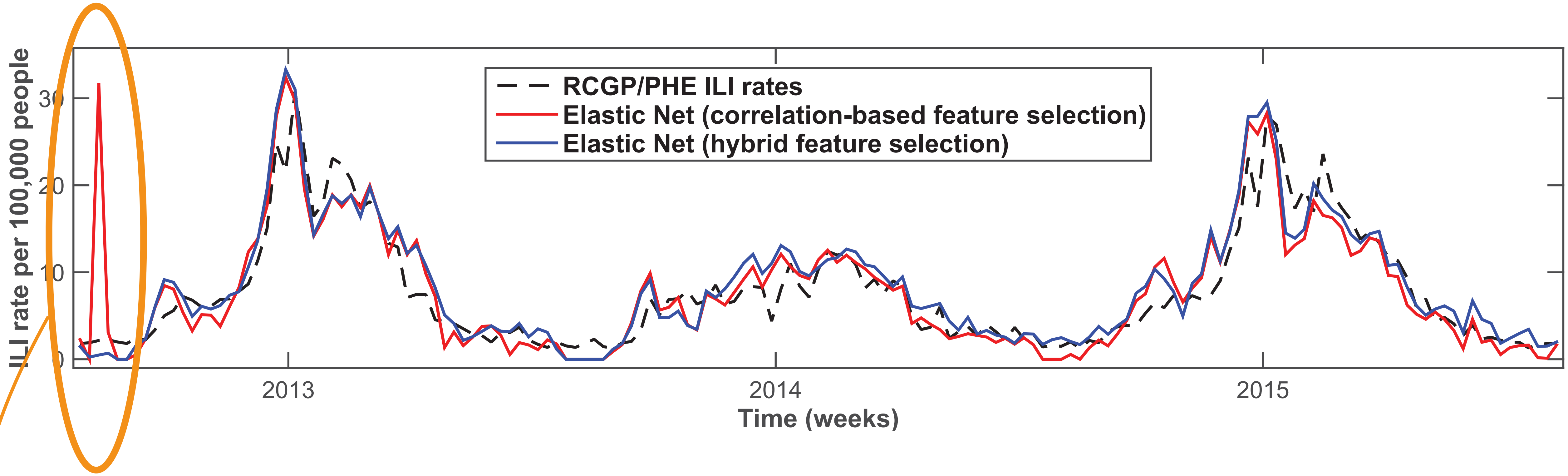
If we combine it with the previous ways for selecting features and state-of-the-art regression approaches, will we obtain **better inference accuracy**?

We test **7 feature selection approaches**:

- concept ranking (CR) → elastic net (**1**)
- correlation → elastic net (**2**) → Gaussian Process (GP) (**3**)
- CR → correlation → elastic net (**4**) → GP (**5**)
- $\boxed{\text{CR} \rightarrow \text{correlation} \rightarrow \text{GP } (\mathbf{6})}$
- correlation → GP (**7**)

Performance improvements

Elastic net with and without word embeddings filtering



ratio over highest weight

prof. *surname* (70.3%), *name surname* (27.2%), heal the world (21.9%), heating oil (21.2%), *name surname* recipes (21%), tlc diet (13.3%), blood game (12.3%), swine flu vaccine side effects (7.2%)

Flu detector

fludetector.cs.ucl.ac.uk

Predicting Twitter user occupation

"Socioeconomic variables are influencing language use."

(Bernstein, 1960; Labov, 1972/2006)

- Validate this hypothesis using a larger sample of humans (social media users)
- Applications
  — research (social sciences, health etc.)
  — commercial

Standard Occupation Classification (SOC)

Major Group 1 (**C1**): Managers, Directors and Senior Officials
Sub-major Group 11: Corporate Managers and Directors
Minor Group 111: Chief Executives and Senior Officials
Unit Group 1115: Chief Executives and Senior Officials
•Job: chief executive, bank manager
Unit Group 1116: Elected Officers and Representatives
Minor Group 112: Production Managers and Directors
Minor Group 113: Functional Managers and Directors
Minor Group 115: Financial Institution Managers and Directors
Minor Group 116: Managers and Directors in Transport and Logistics
Minor Group 117: Senior Officers in Protective Services
Minor Group 118: Health and Social Services Managers and Directors
Minor Group 119: Managers and Directors in Retail and Wholesale
Sub-major Group 12: Other Managers and Proprietors
Major Group (**C2**): Professional Occupations
•Job: mechanical engineer, pediatrist
Major Group (**C3**): Associate Professional and Technical Occupations
•Job: system administrator, dispensing optician
Major Group (**C4**): Administrative and Secretarial Occupations
•Job: legal clerk, company secretary
Major Group (**C5**): Skilled Trades Occupations
•Job: electrical fitter, tailor
Major Group (**C6**): Caring, Leisure and Other Service Occupations
•Job: nursery assistant, hairdresser
Major Group (**C7**): Sales and Customer Service Occupations
•Job: sales assistant, telephonist
Major Group (**C8**): Process, Plant and Machine Operatives
•Job: factory worker, van driver
Major Group (**C9**): Elementary Occupations
•Job: shelf stacker, bartender

provided by the Office for National Statistics (UK)

9 major groups

25 sub-major groups

90 minor groups

369 unit groups

Standard Occupation Classification (SOC)

The 9 major occupational classes (C1-9)

—**C1:** Managers, Directors, Senior Officials (**CEO**, **bank manager**)

—**C2:** Professional Occupations (**postdoc**, **pediatrist**)

—**C3:** Associate Professional, Technical (**sysadmin**, **dispensing optician**)

—**C4:** Administrative, Secretarial (**legal clerk**, **secretary**)

—**C5:** Skilled Trades (**electrical fitter**, **tailor**)

—**C6:** Caring, Leisure, Other Service (**nursery assistant**, **hairdresser**)

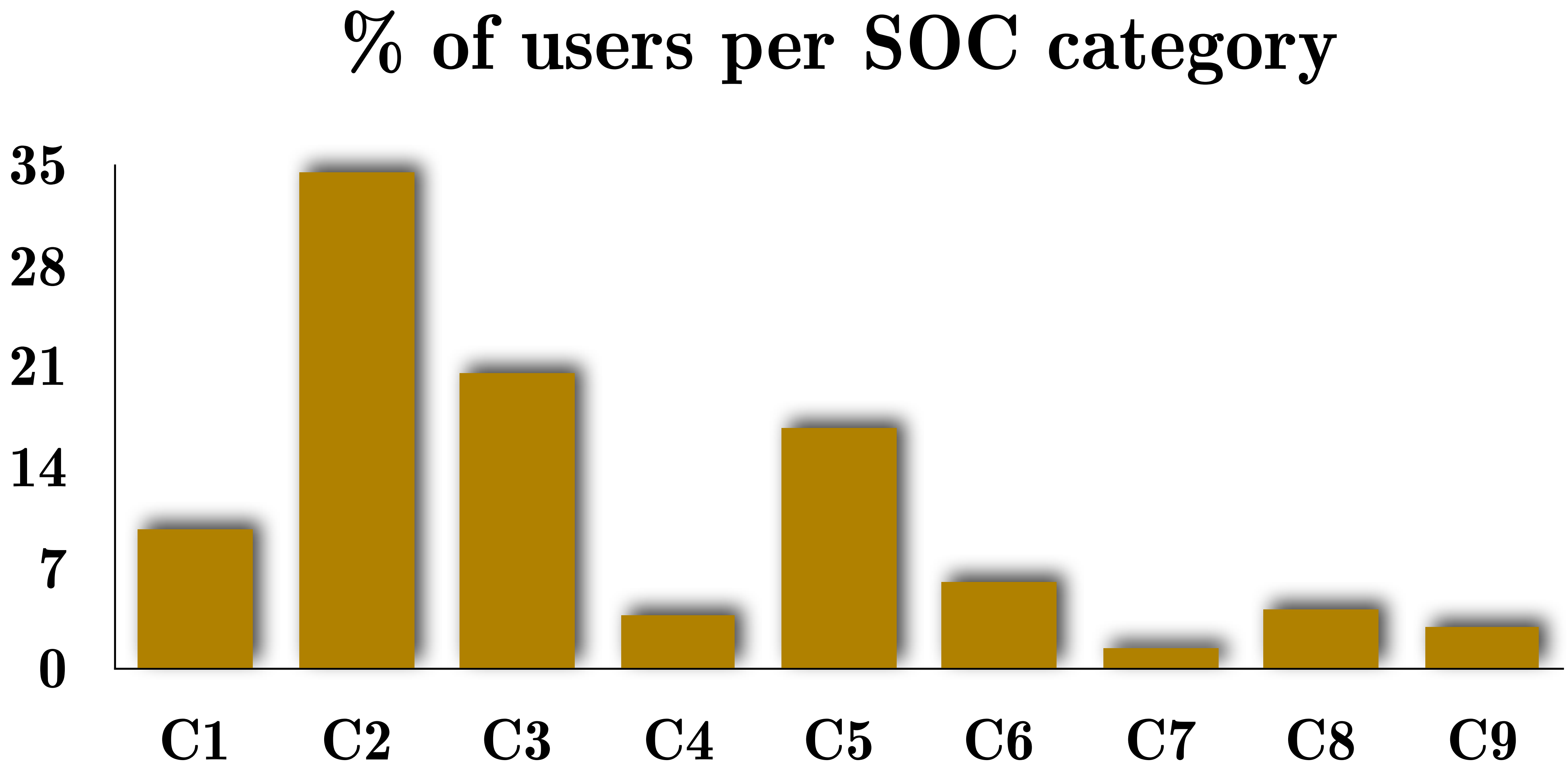
—**C7:** Sales, Customer Service (**sales assistant**, **telephonist**)

—**C8:** Process, Plant, Machine Operatives (**factory worker**, **van driver**)

—**C9:** Elementary (**shelf stacker**, **bartender**)

Twitter data

- **5,191** Twitter users mapped to their occupations, then mapped to one of the 9 SOC categories
- 10 million tweets



Twitter user features

number of

— followers
— friends
— followers/friends (ratio)
— times listed
— tweets
— favourites (likes)
— unique @-mentions
— tweets/day (avg.)
— retweets/tweet (avg.)

proportion of

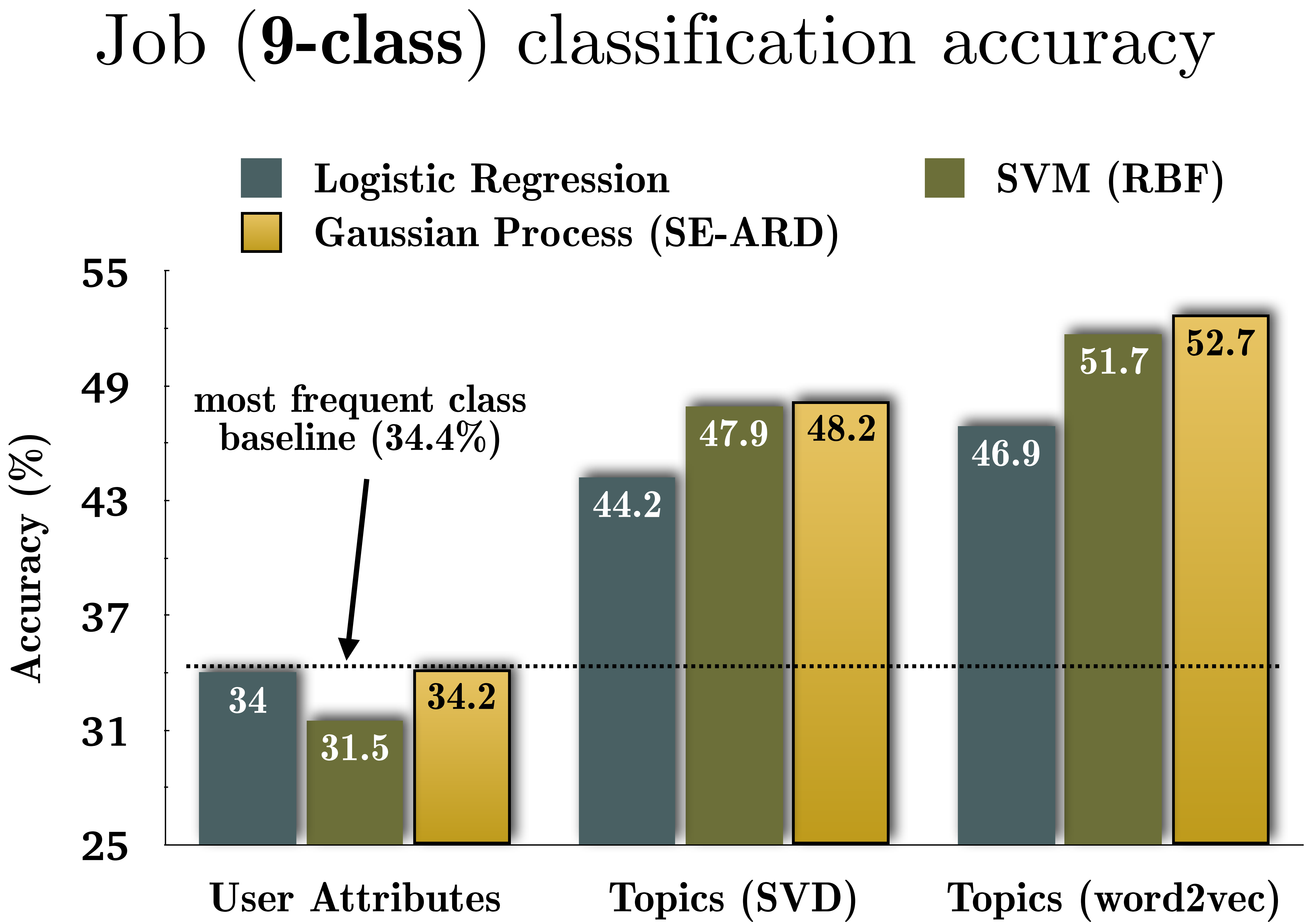
— retweets done
— non duplicate tweets
— retweeted tweets
— hashtags
— tweets with hashtags
— tweets with @-mentions
— @-replies
— tweets with links
— tweets in English

Twitter user features — Topics

Topics — **Word clusters** (#: 30, 50, 100, **200**)

- **SVD** on the graph laplacian of the word by word similarity matrix using **normalised PMI**, *i.e.* a form of spectral clustering

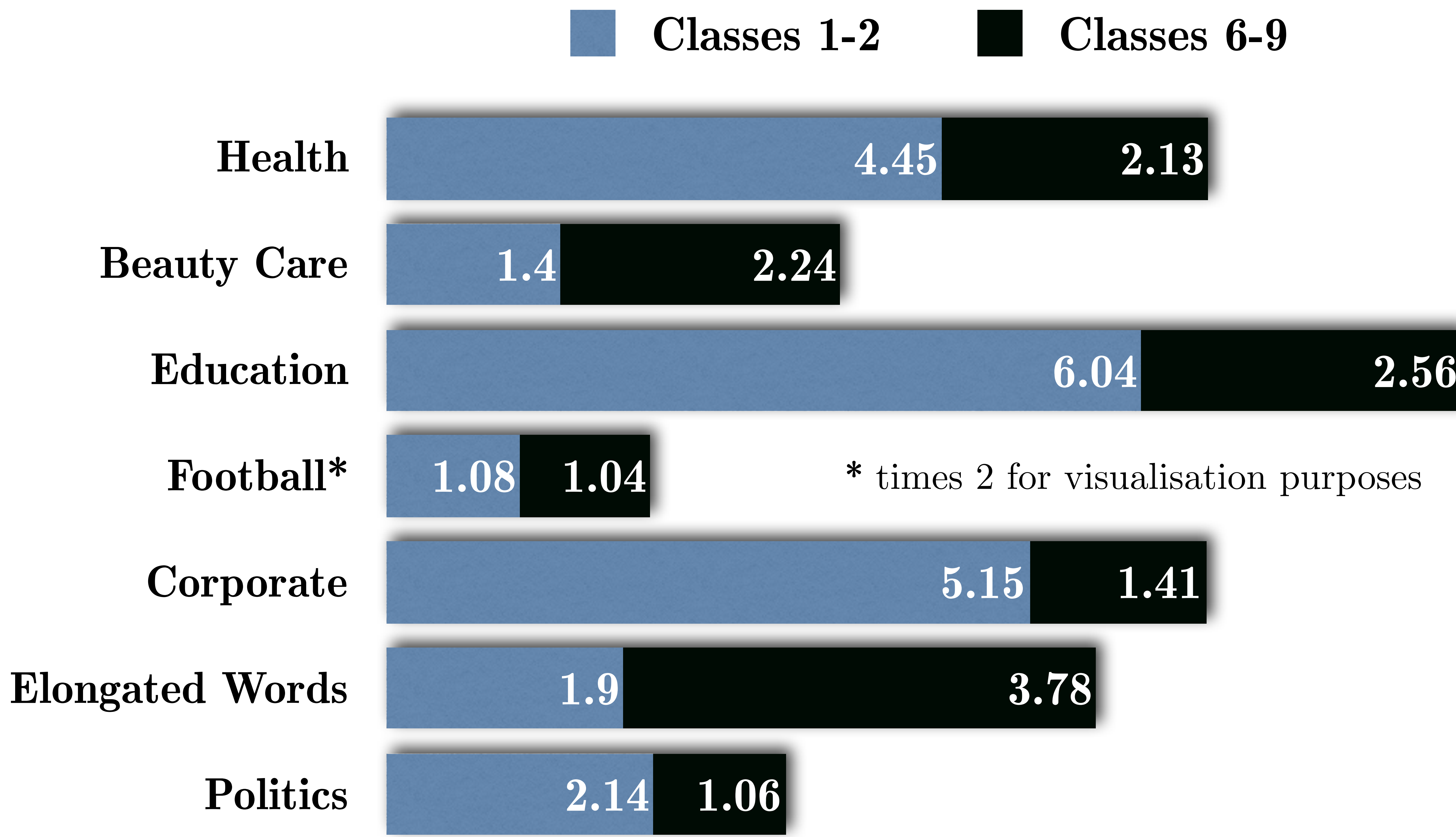
- **`word2vec`** (skip-gram with negative sampling) to learn word embeddings; pairwise **cosine similarity** on the embeddings to derive a word by word similarity matrix; then spectral clustering on the similarity matrix


Job (9-class) classification accuracy
Logistic Regression
SVM (RBF)
Gaussian Process (SE-ARD)
Accuracy (%)
55
49
43
37
31
25
most frequent class baseline (34.4%)
34
31.5
34.2
44.2
47.9
48.2
46.9
51.7
52.7
User Attributes
Topics (SVD)
Topics (word2vec)

Most predictive topics (word2vec)

Topic	**Most central words;** ***Most frequent words***
Arts	archival, stencil, canvas, minimalist; *art, design, print*
Health	chemotherapy, diagnosis, disease; *risk, cancer, mental, stress*
Beauty Care	exfoliating, cleanser, hydrating; *beauty, natural, dry, skin*
Higher Education	undergraduate, doctoral, academic, students, curriculum; *students, research, board, student, college, education, library*
Football	bardsley, etherington, gallas; *van, foster, cole, winger*
Corporate	consortium, institutional, firm's; *patent, industry, reports*
Elongated Words	yaaayy, wooooo, woooo, yayyyyy, yaaaaay, yayayaya, yayy; *wait, till, til, yay, ahhh, hoo, woo, woot, whoop, woohoo*
Politics	religious, colonialism, christianity, judaism, persecution, fascism, marxism; *human, culture, justice, religion, democracy*

Higher vs. lower skilled occupations and topics



Topic scores for occupational class supersets

end_of_lecture

@lampos