

COMP0084 (IRDM)

Topic models, vector semantics and applications

Vasileios Lamos
Computer Science, UCL

Slides (with potential revisions)

lampos.net/slides/irdm2019.pdf

@lampos



lampos.net



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 the word2vec method
- **Applications**
 - Predicting judicial decisions
 - Improving the accuracy of disease models from Web searches
 - Inferring the occupational class of a Twitter user

Material

Books

— Jurafsky and Martin. Speech and Language Processing, 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, videlectures.net/mlss09uk_blei_tm/

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

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

Slides

— Web Search and Data Mining (WSDM) 2014 tutorial on “Multilingual Probabilistic Topic Modelling”, liir.cs.kuleuven.be/tutorial/WSDM2014Tutorial.pdf

Popular software libraries (*might be out-dated by now!*)

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

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

Part I — Topic models

What is a topic model?

- **Informally:**

What is a topic model?

- **Informally:** groupings (or clusters) of words (or n -grams) that are somehow related

What is a topic model?

- **Informally:** groupings (or clusters) of words (or n -grams) 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

What is a topic model?

- **Informally:** groupings (or clusters) of words (or n -grams) 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 often defined a **probabilistic structure** expressing a certain set of assumptions about how the documents in our collection were generated
- **Note:** we can also learn 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!



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 inter-disciplinary interest (newspapers, books, social media, health reports, ...)



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 inter-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 inter-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

[1/6]

“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.

17K articles from Science

[2/6]

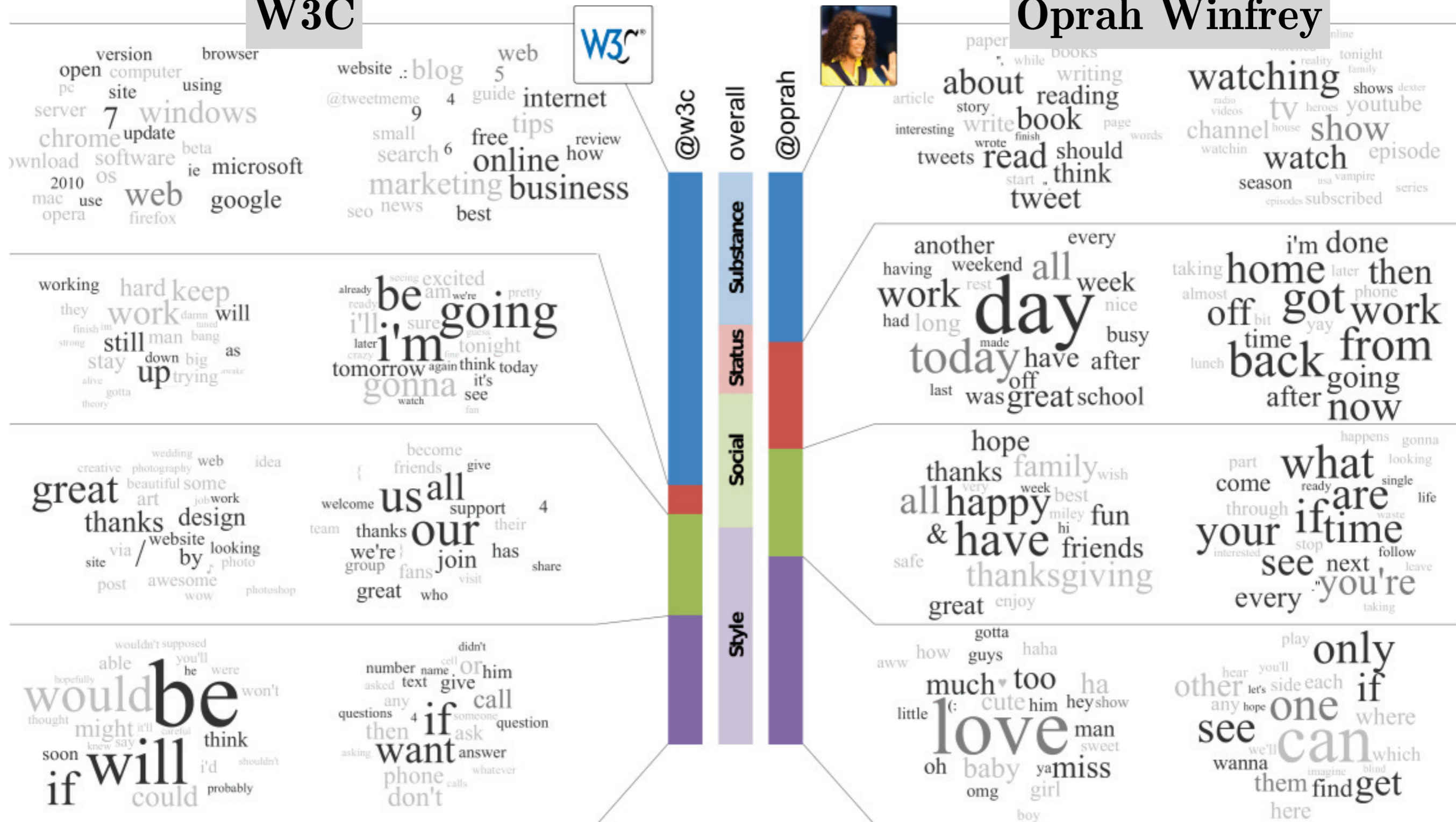
“Genetics”	“Evolution”	“Disease”	“Computers”
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	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

Characterising Twitter users

[3/6]

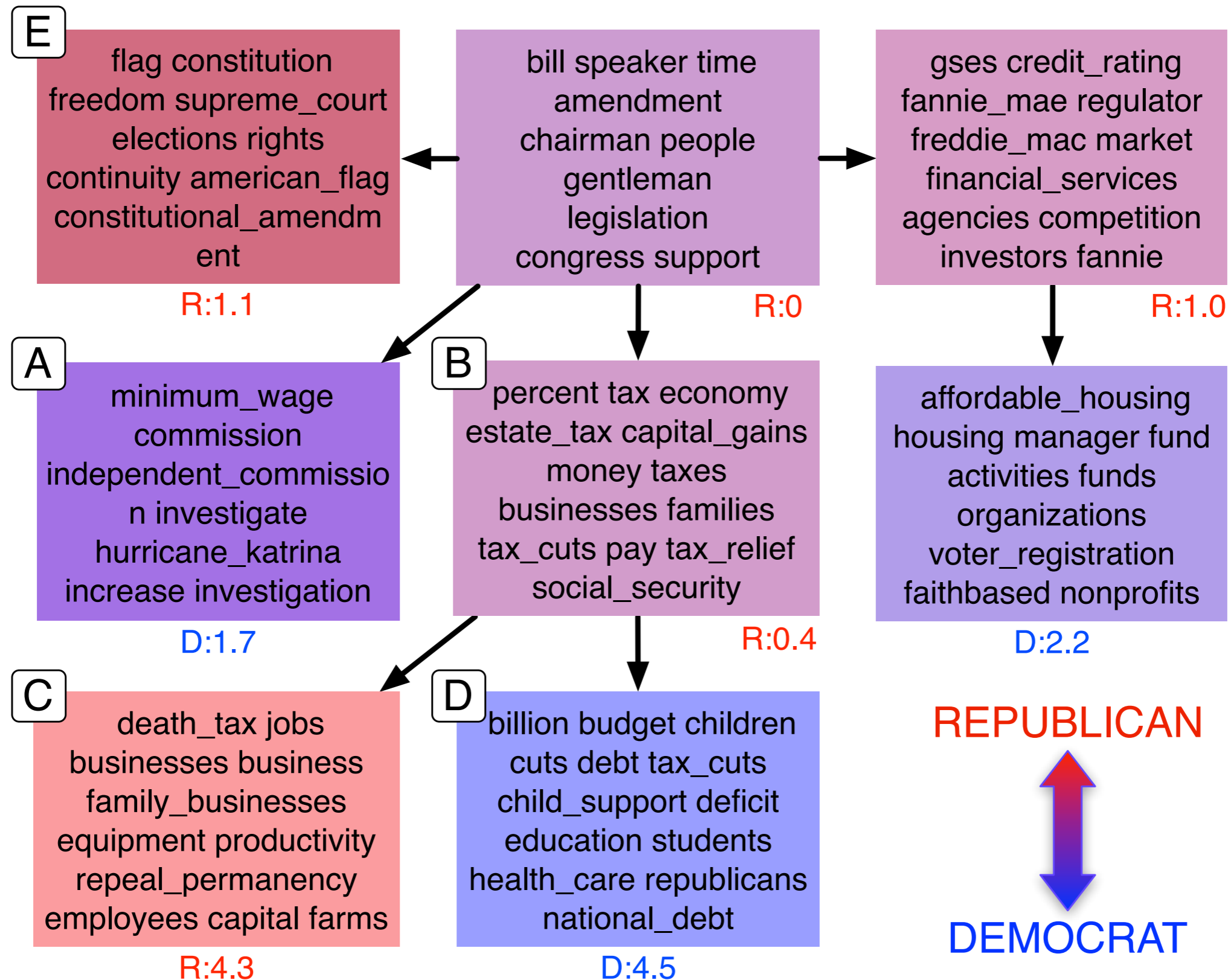
W3C

Oprah Winfrey



Congressional floor debates

[5/6]

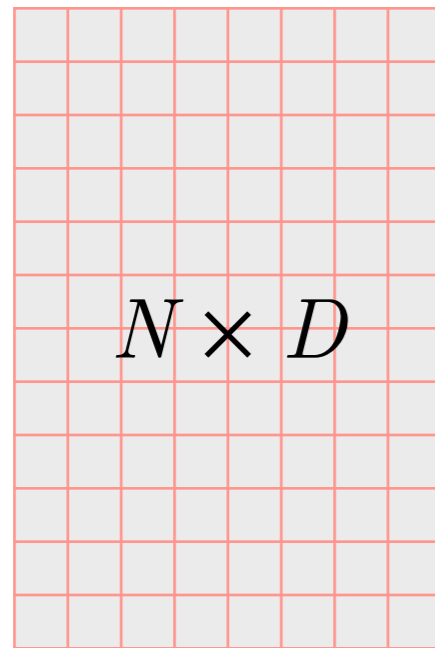


Predicting judicial decisions

[6/6]

Label	Words	Violation of Article 3 that prohibits inhuman treatment	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

Latent Semantic Analysis (or Indexing) — LSA



X

Singular Value Decomposition (SVD; truncated) on the term-document matrix **X** representing the frequency of N terms (words or n -grams) in D documents

Latent Semantic Analysis (or Indexing) — LSA

The diagram shows three matrices represented as grids. The first grid is labeled $N \times D$ and \mathbf{X} . The second grid is labeled $N \times K$ and \mathbf{W}_K . The third grid is labeled $K \times D$ and \mathbf{Y} . An approximation symbol \approx is between the first and second grids, and a multiplication symbol \times is between the second and third grids.

$$\mathbf{X} \approx \mathbf{W}_K \times \mathbf{Y}$$

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

\mathbf{W}_K : each topic's (K) distribution over N terms

Latent Semantic Analysis (or Indexing) — LSA

The diagram shows the matrix decomposition of the term-document matrix \mathbf{X} into the product of the topic distribution matrix \mathbf{W}_K and the topic importance matrix $\mathbf{\Sigma}_K$. The matrix \mathbf{X} is represented as a grid of size $N \times D$. The matrix \mathbf{W}_K is a grid of size $N \times K$. The matrix $\mathbf{\Sigma}_K$ is a $K \times K$ matrix with a diagonal of blue squares and off-diagonal green squares, with the number 0 in the top-right and bottom-left corners. The decomposition is shown as $\mathbf{X} \approx \mathbf{W}_K \times \mathbf{\Sigma}_K$.

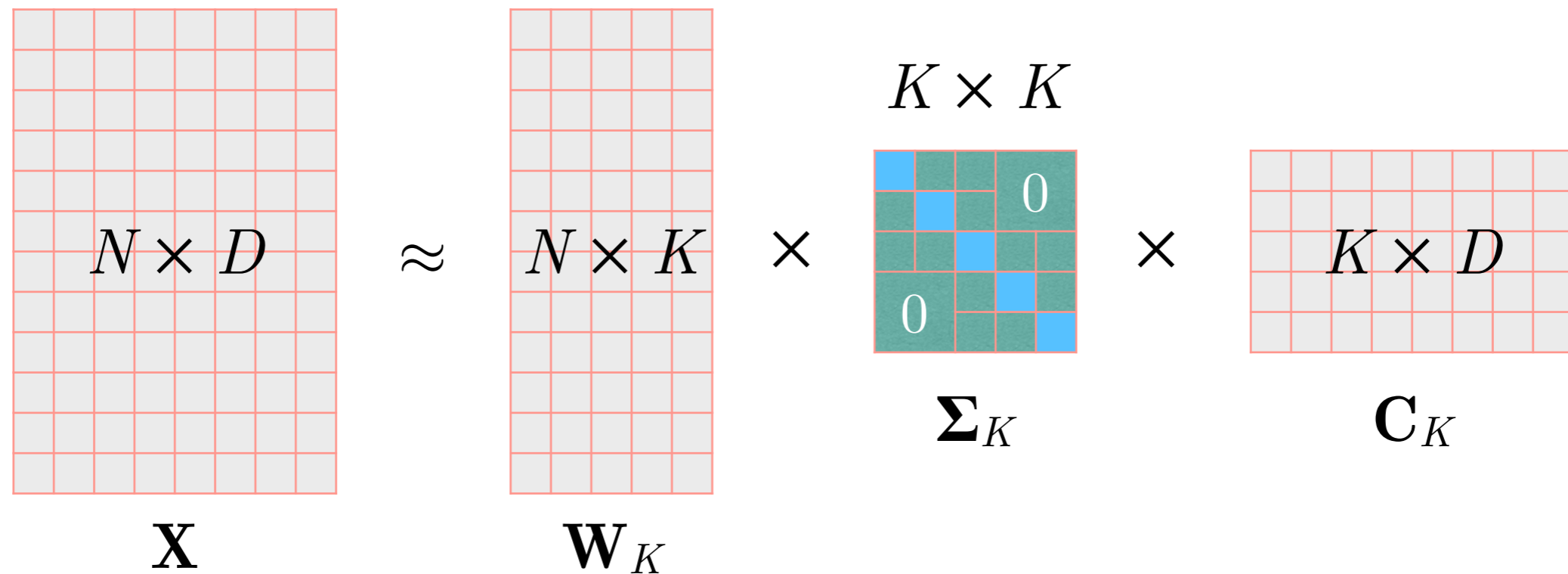
$$\mathbf{X} \approx \mathbf{W}_K \times \mathbf{\Sigma}_K$$

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

\mathbf{W}_K : each topic's (K) distribution over N terms

$\mathbf{\Sigma}_K$: topic importance

Latent Semantic Analysis (or Indexing) — LSA



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

\mathbf{W}_K : each topic's (K) distribution over N terms

Σ_K : topic importance

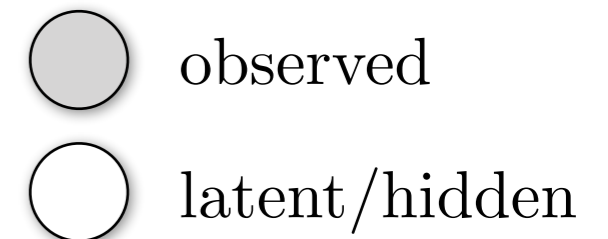
\mathbf{C}_K : each document's (D) distribution over K topics

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 $p(w_i|z_k)$

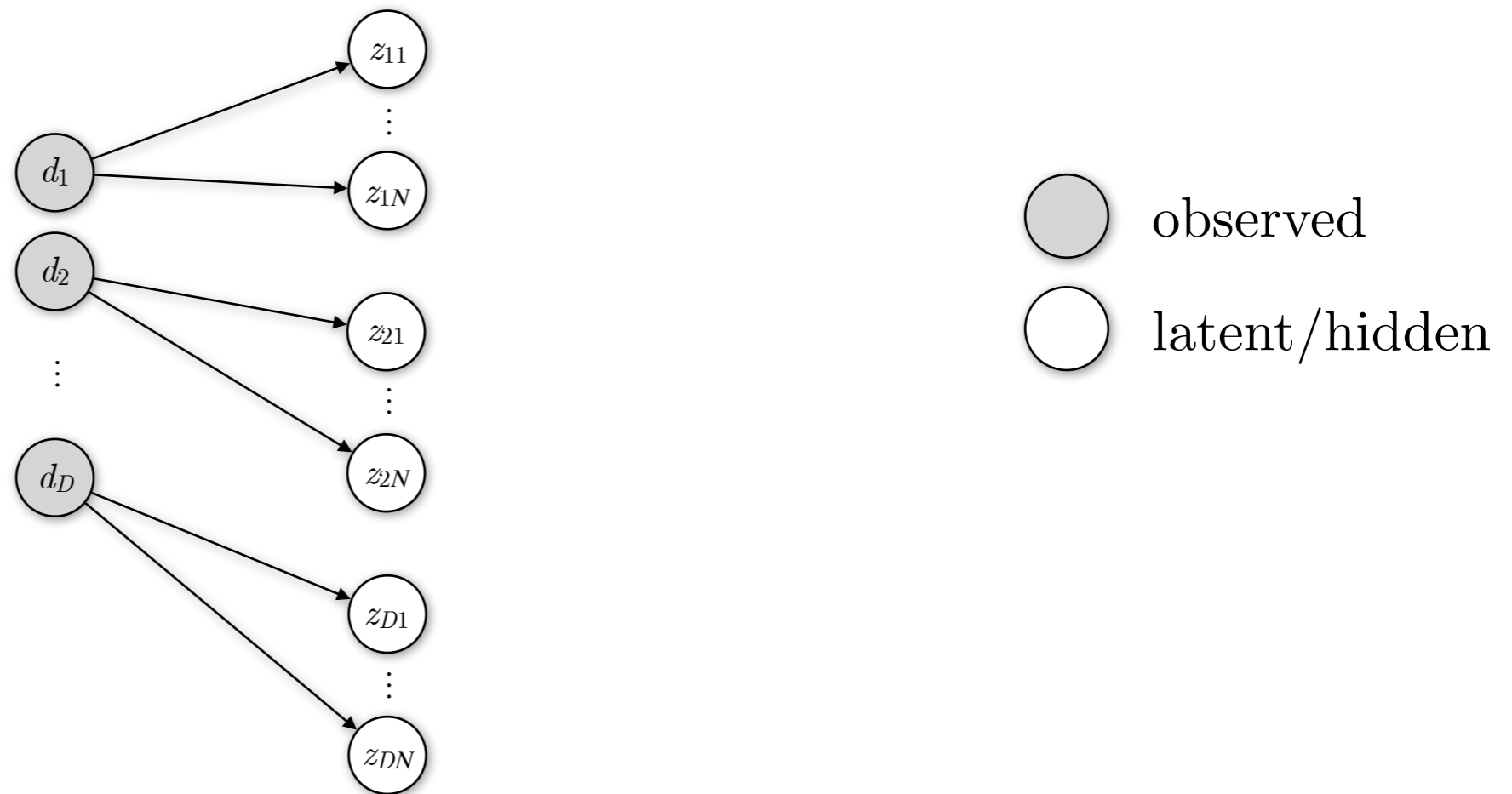
Probabilistic LSA — pLSA



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 $p(w_i|z_k)$

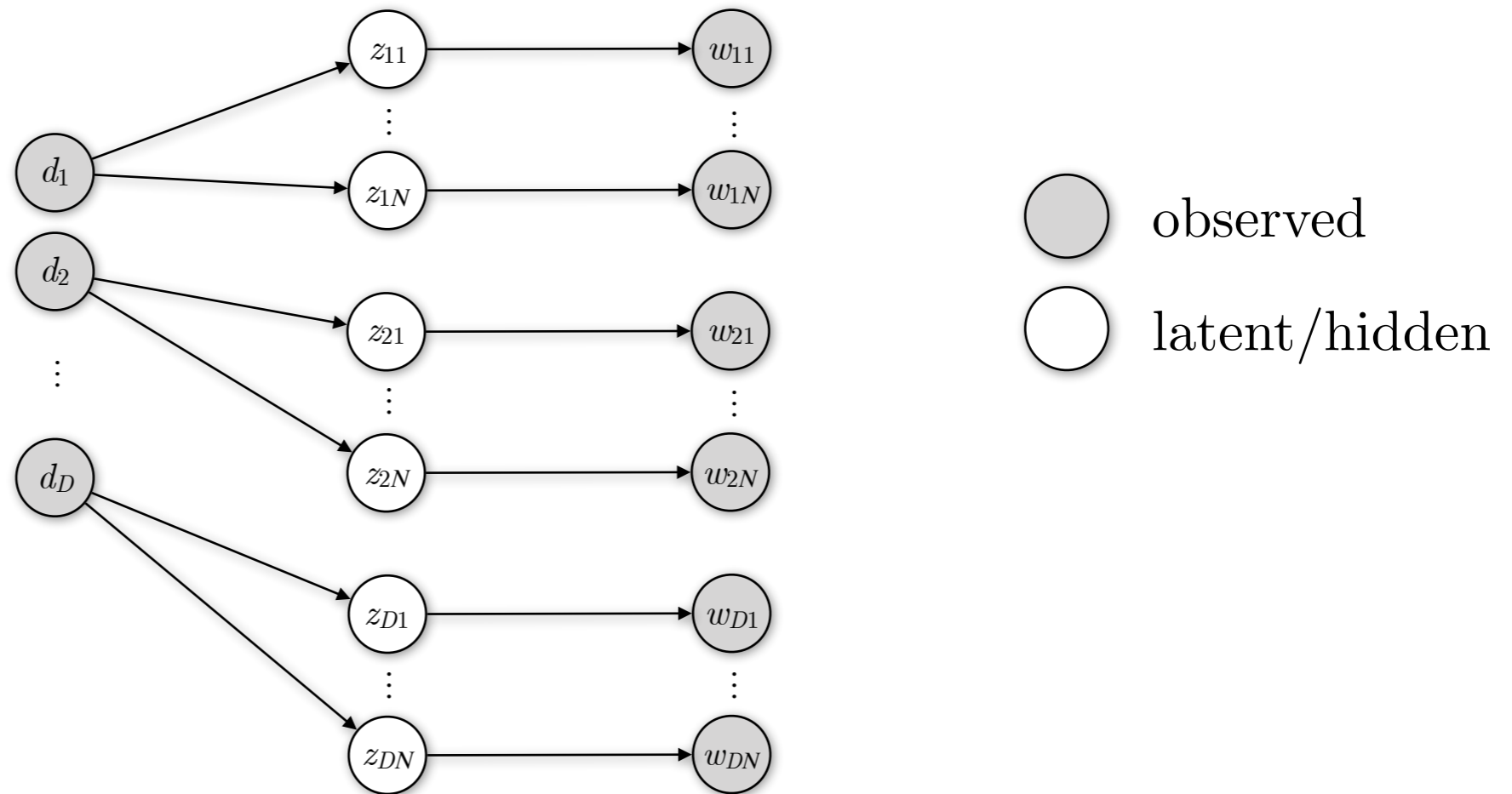
Probabilistic LSA — pLSA



For all j documents (1 to D):

- Select a document d_j with probability $p(d_j)$
- Choose a mixture of K topics θ_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

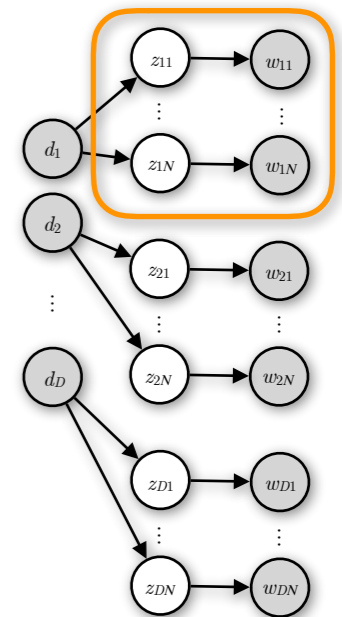
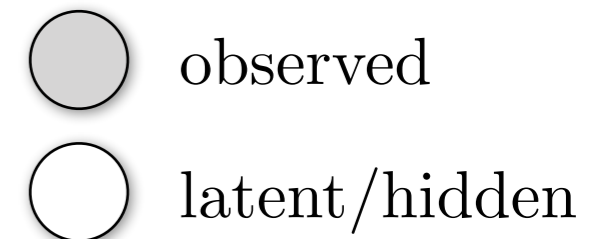
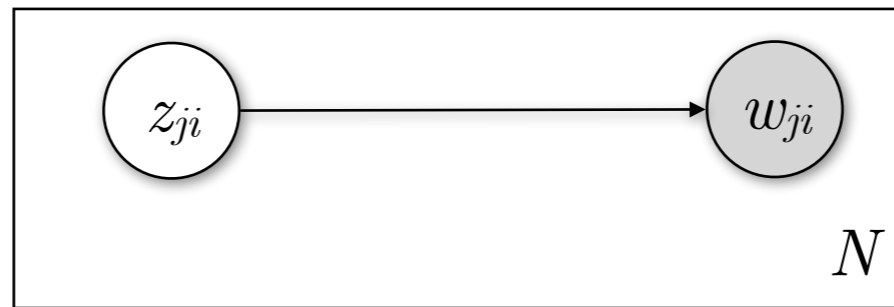


For all j documents (1 to D):

- Select a document d_j with probability $p(d_j)$
- Choose a mixture of K topics θ_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

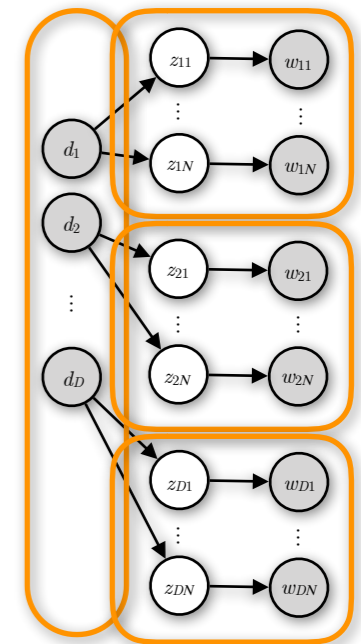
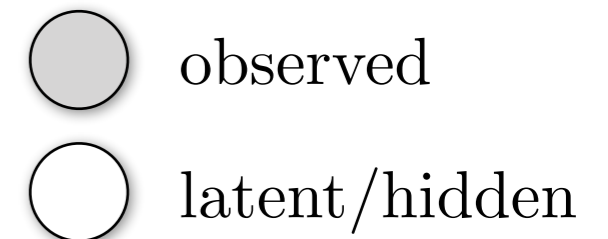
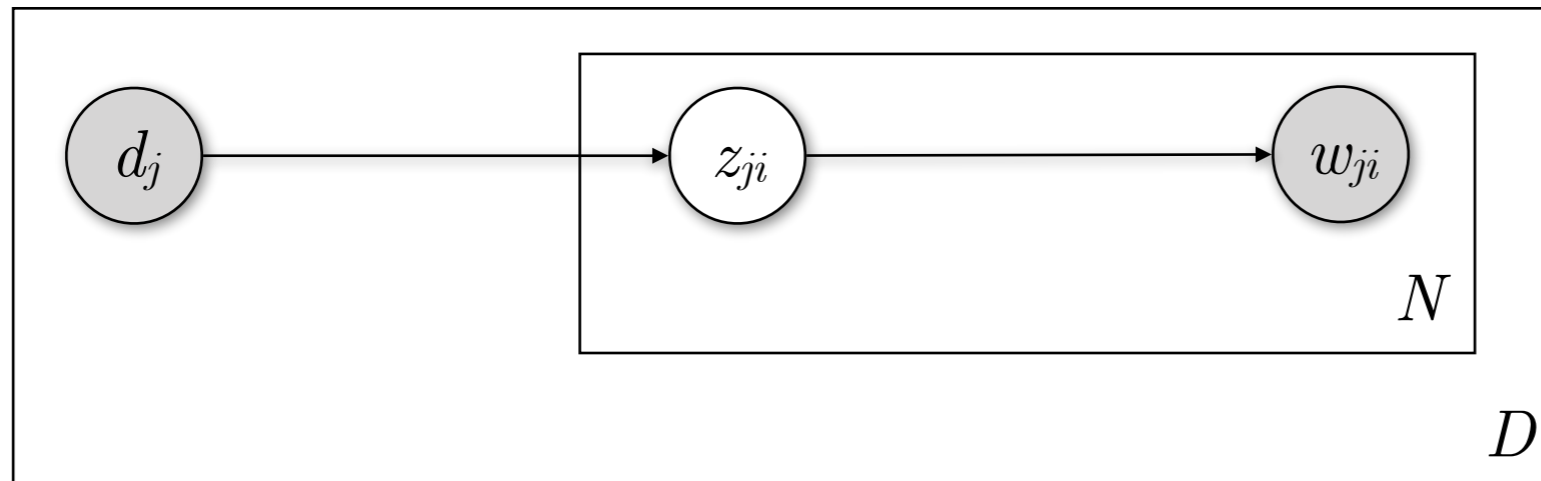


For all j documents (1 to D):

- Select a document d_j with probability $p(d_j)$
- Choose a mixture of K topics θ_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

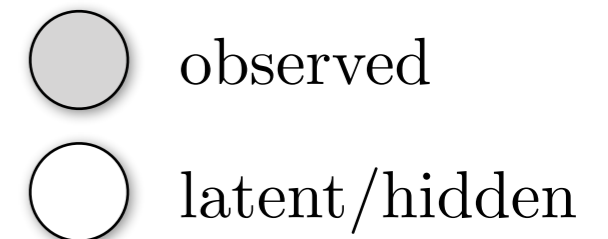
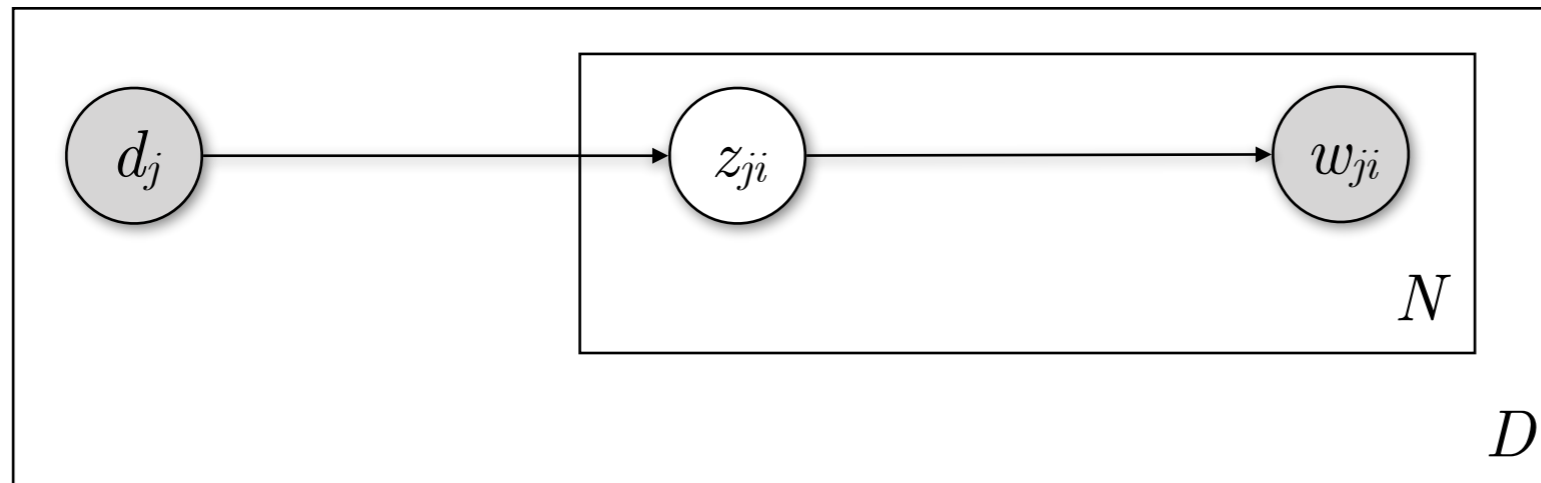


For all j documents (1 to D):

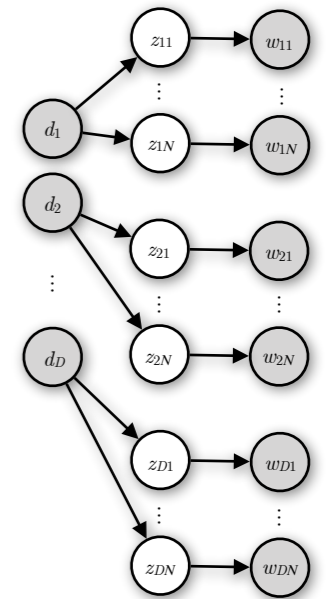
- Select a document d_j with probability $p(d_j)$
- Choose a mixture of K topics θ_j for document d_j
- For each word position i (1 to N) in the document d_j :
 - Choose a topic z_k with probability $p(z_k|d_j)$
 - Choose a word w_i with probability $p(w_i|z_k)$

Probabilistic LSA — pLSA

Plate notation



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

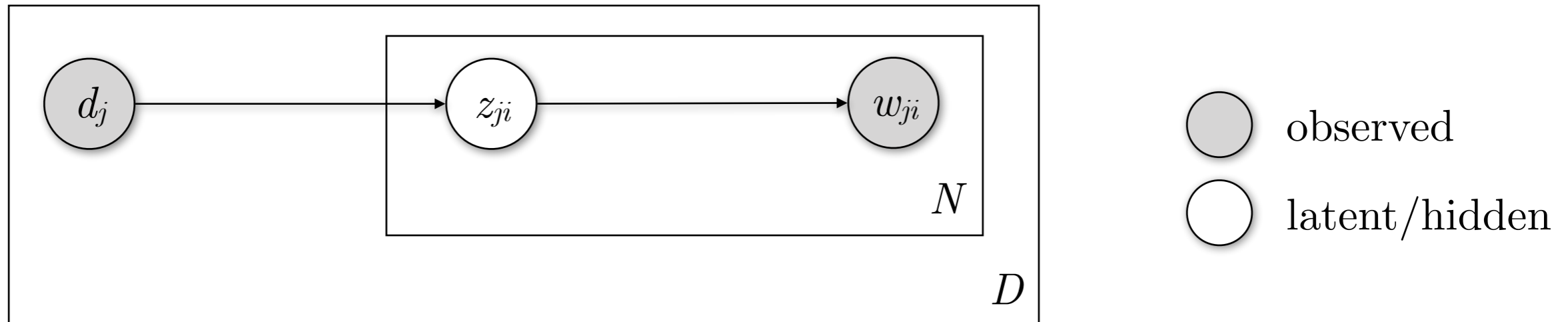


For all j documents (1 to D):

- Select a document d_j with probability $p(d_j)$
- Choose a mixture of K topics θ_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

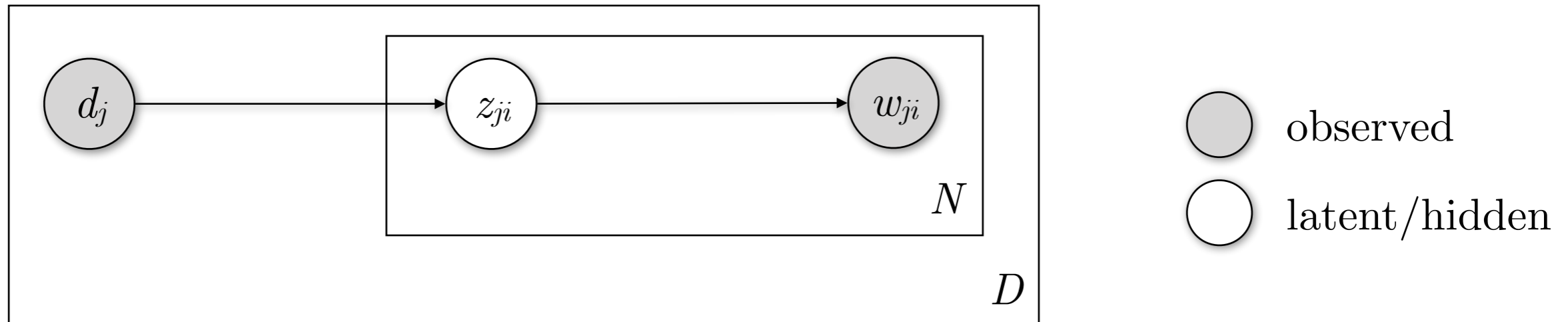


Assumptions: In a document (d_j), a word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

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

Probabilistic LSA — pLSA

Plate notation



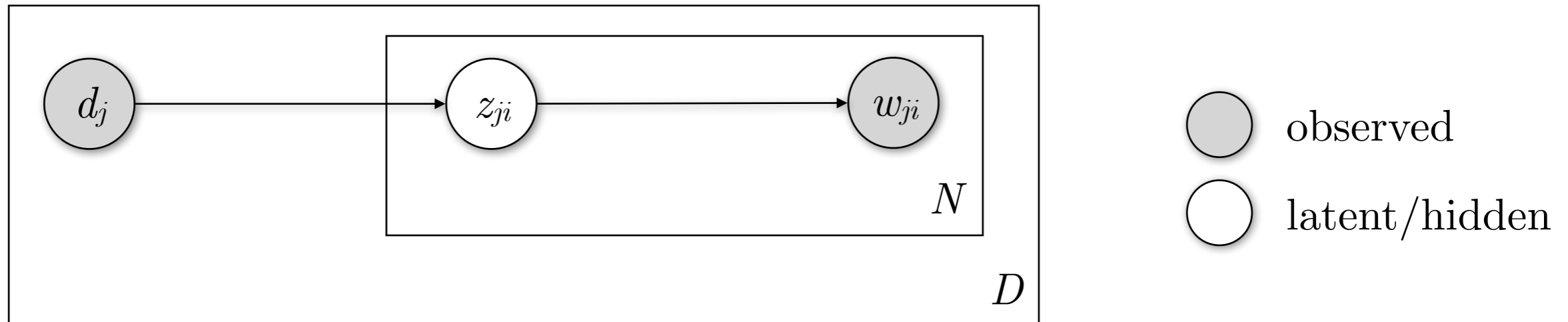
Assumptions: In a document (d_j), a word (w_{ji}) is generated from a single topic (z_{ji}) from the K assumed ones, and given that topic, the word is independent of all of the other words in that document.

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

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

Probabilistic LSA — pLSA

Plate notation



Assumptions: In a document (d_j), a 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.

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

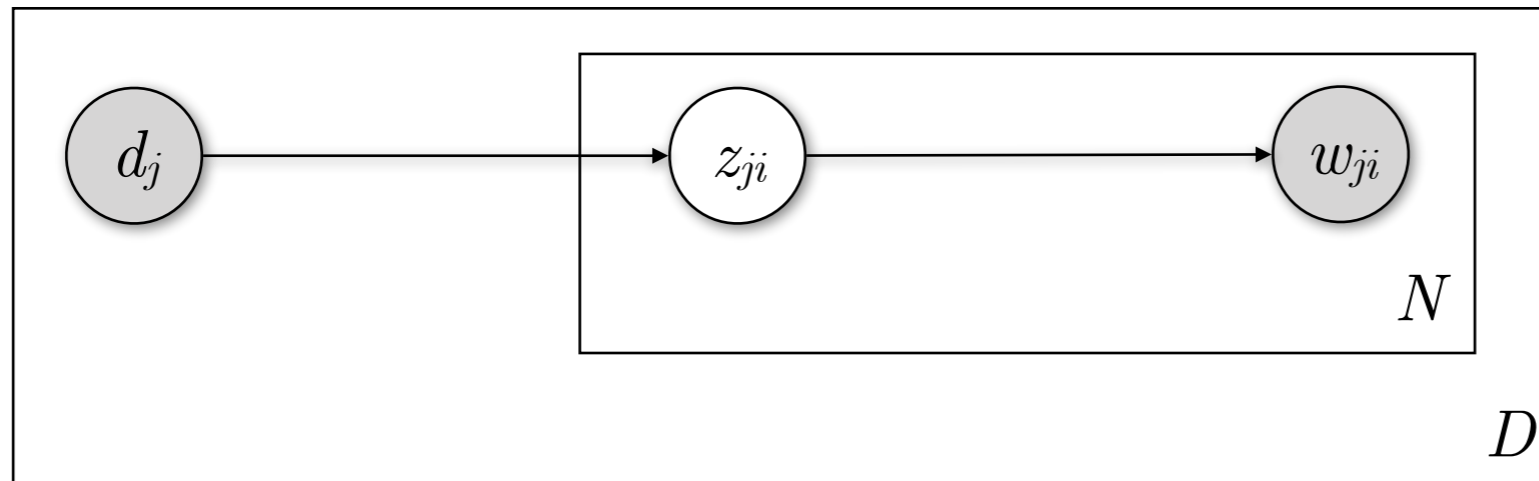
$$p(d_j, w) = p(d_j) \prod_{i=1}^N \sum_{k=1}^K p(z_i = k | d_j) p(w_i | z_i = k) \quad \text{Joint prob. dist. for } d_j \text{ and } w$$

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

Probabilistic LSA — pLSA

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

Plate notation



● observed
○ latent/hidden

Assumptions: In a document (d_j), a 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.

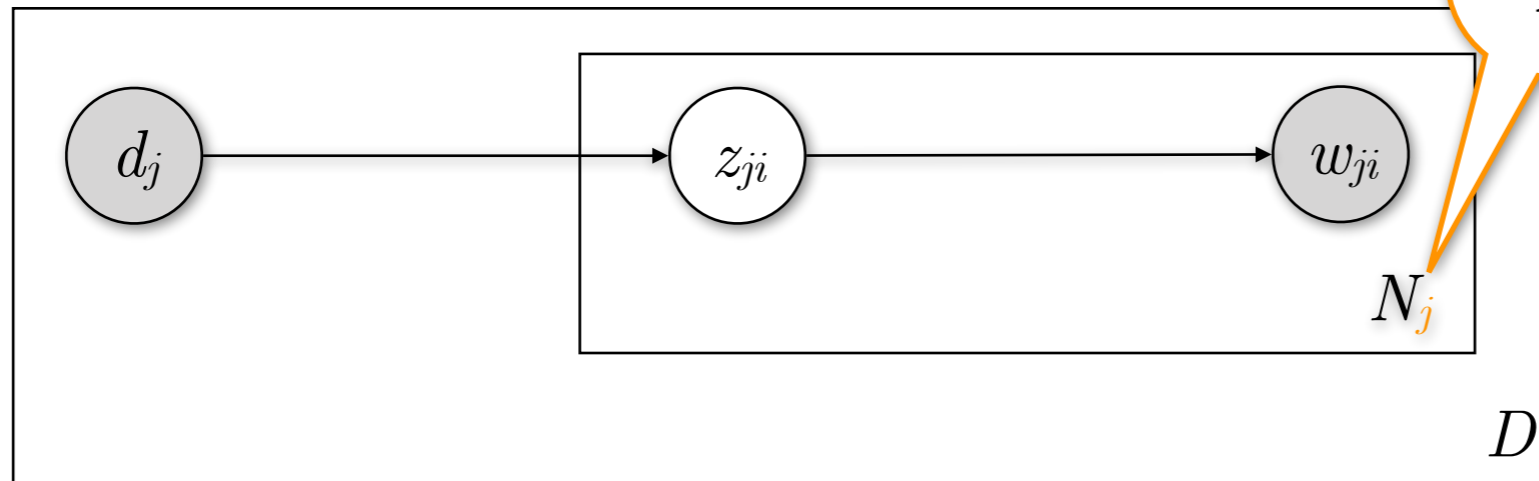
$$p(d_j, w_i) = p(d_j) p(w_i | d_j) = p(d_j) \sum_{k=1}^K p(z = k | d_j) p(w_i | z = k) \quad \text{Joint prob. dist. for } d_j \text{ and } w_i$$

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

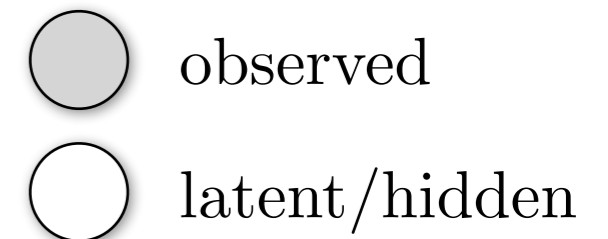
$$p(d, w) = \prod_{j=1}^D p(d_j) \prod_{i=1}^N \sum_{k=1}^K p(z_{ji} = k | d_j) p(w_{ji} | z_{ji} = k) \quad \text{Joint probability distribution}$$

Probabilistic LSA — pLSA

Plate notation



Number of words may not be the same for all documents!




Assumptions: In a document (d_j), a 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.

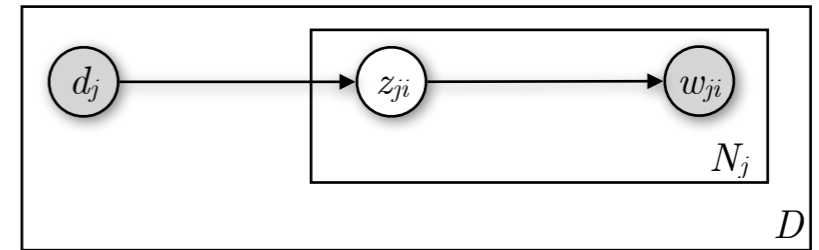
$$p(d_j, w_i) = p(d_j) p(w_i | d_j) = p(d_j) \sum_{k=1}^K p(z = k | d_j) p(w_i | z = k) \quad \text{Joint prob. dist. for } d_j \text{ and } w_i$$

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

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

pLSA — Inference


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


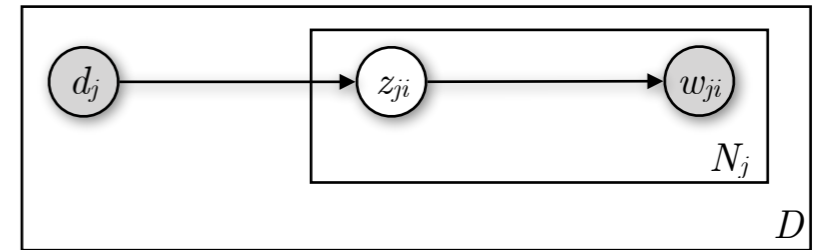


Expectation Maximisation (EM):

- 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

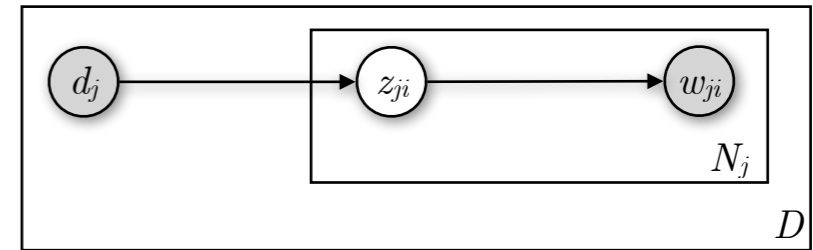
$$p(d, w) = \prod_{j=1}^D p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^K p(z_{ji} = k | d_j) p(w_{ji} | z_{ji} = k)$$




- 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


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

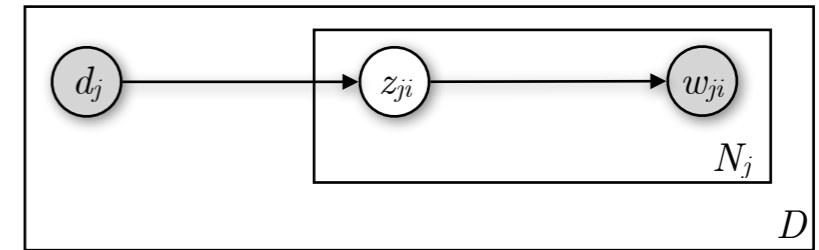


- Initialise $p(z_k | d_j)$ and $p(w_i | z_k)$ to positive quantities
- **E-step:** Estimate the probability of each topic given the words in each document

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

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

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

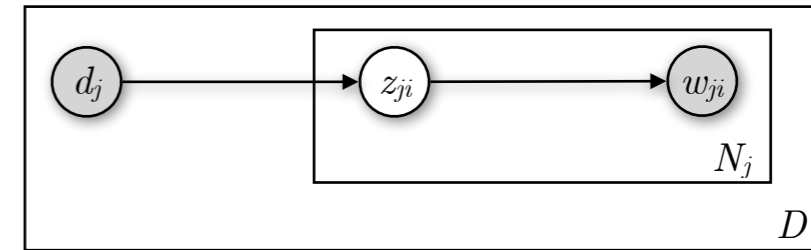
- **M-step:** Re-estimate $p(z_k | d_j)$, $p(w_i | z_k)$ given the revised $p(z_k | d_j, w_i)$

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

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

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 .

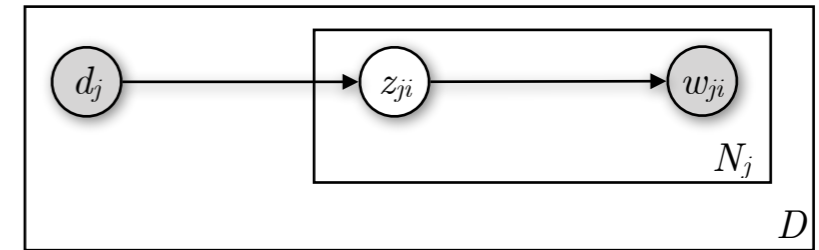
- **M-step:** Re-estimate $p(z_k | d_j)$, $p(w_i | z_k)$ given the revised $p(z_k | d_j, w_i)$

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

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

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

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

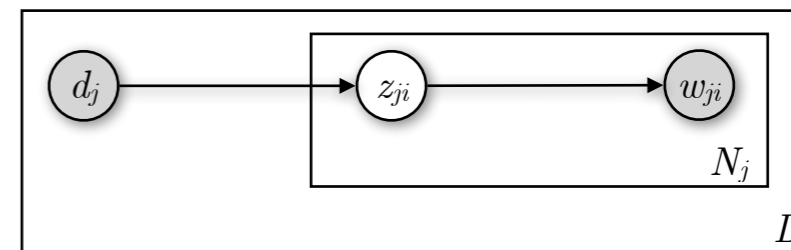
- **M-step:** Re-estimate $p(z_k | d_j)$, $p(w_i | z_k)$ given the revised $p(z_k | d_j, w_i)$

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

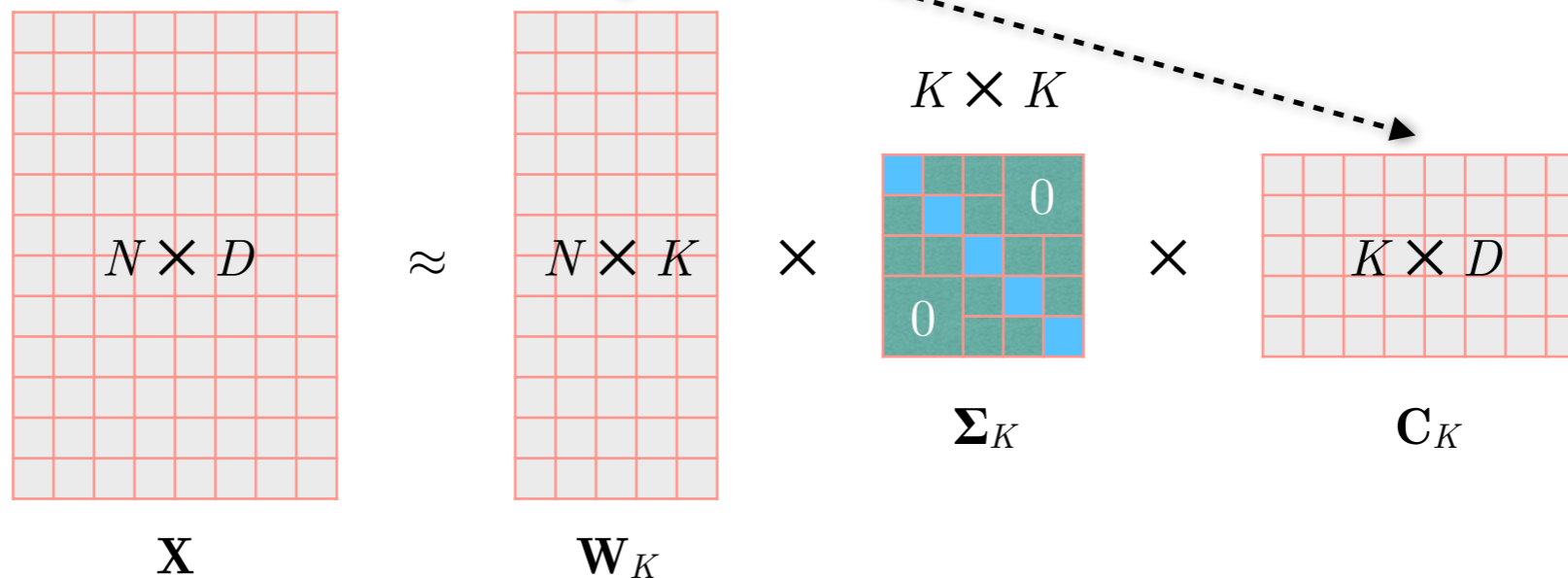
$$p(w_i | z_k) = \frac{\sum_{j=1}^D n(d_j, w_i) p(z_k | d_j, w_i)}{\sum_{j=1}^D \sum_{i'=1}^{N_j} n(d_j, w_{i'}) p(z_k | d_j, w_{i'})}$$

pLSA and LSA

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

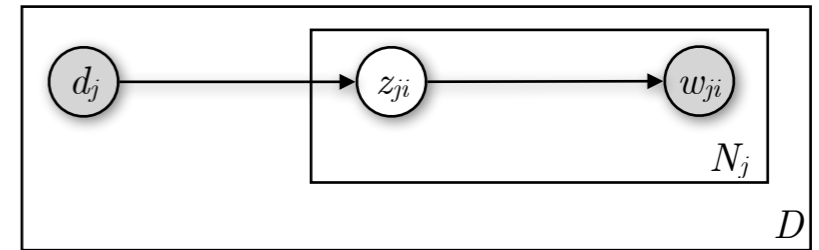


LSA

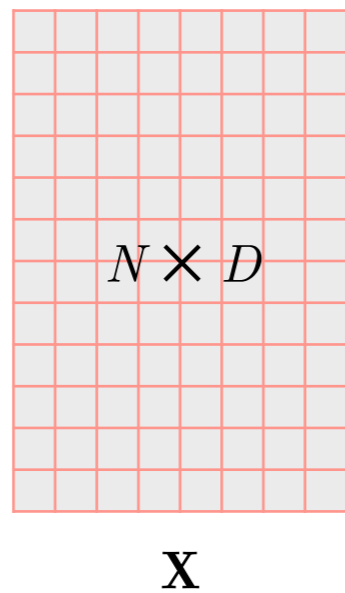


pLSA and LSA

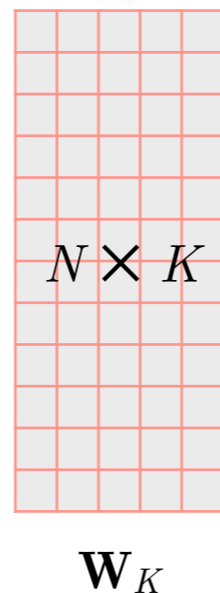
$$p(d, w) = \prod_{j=1}^D p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^K p(z_{ji} = k | d_j) p(w_{ji} | z_{ji} = k)$$



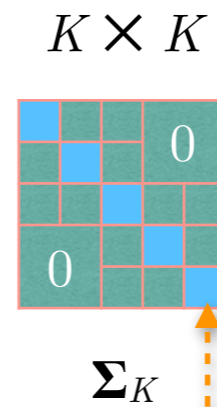
LSA



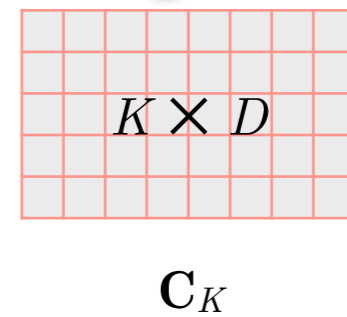
\approx



\times



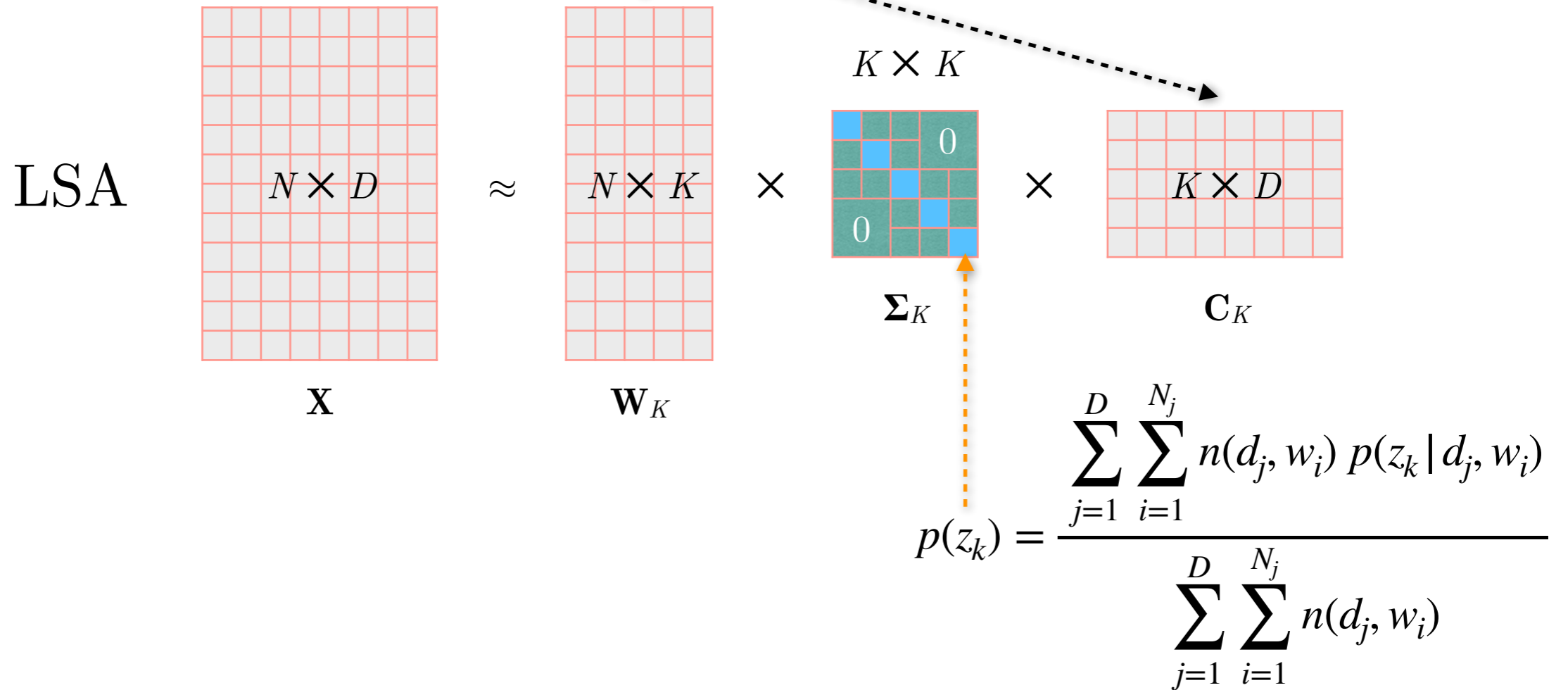
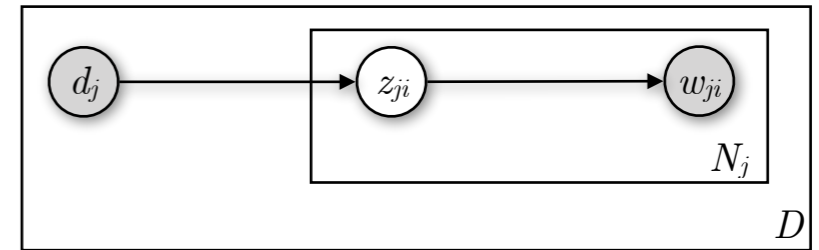
\times



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

pLSA and LSA

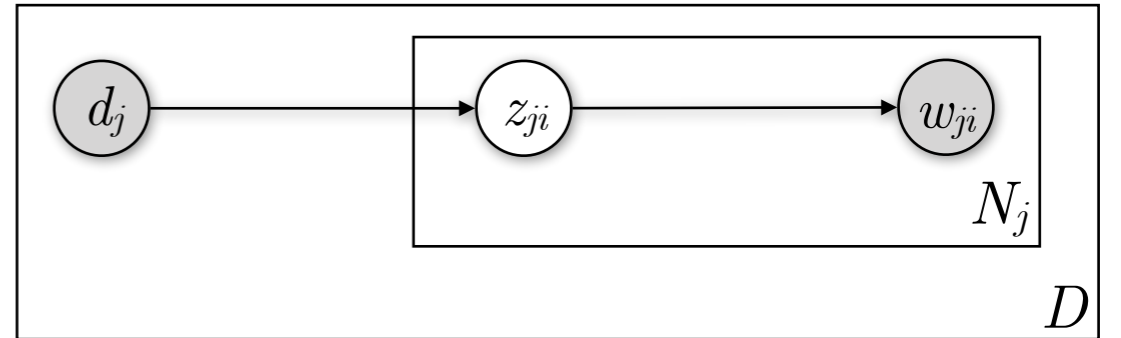
$$p(d, w) = \prod_{j=1}^D p(d_j) \prod_{i=1}^{N_j} \sum_{k=1}^K p(z_{ji} = k | d_j) p(w_{ji} | z_{ji} = k)$$



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

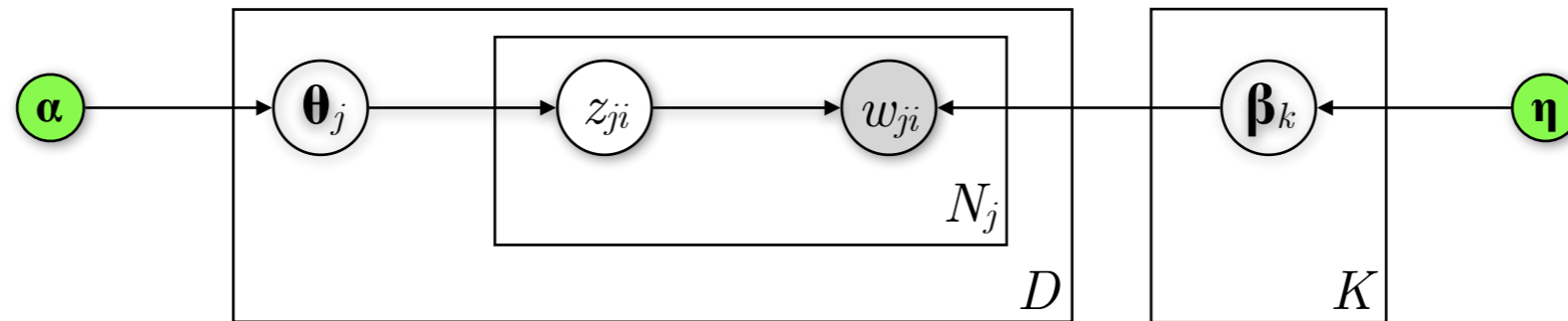
pLSA — Disadvantages

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

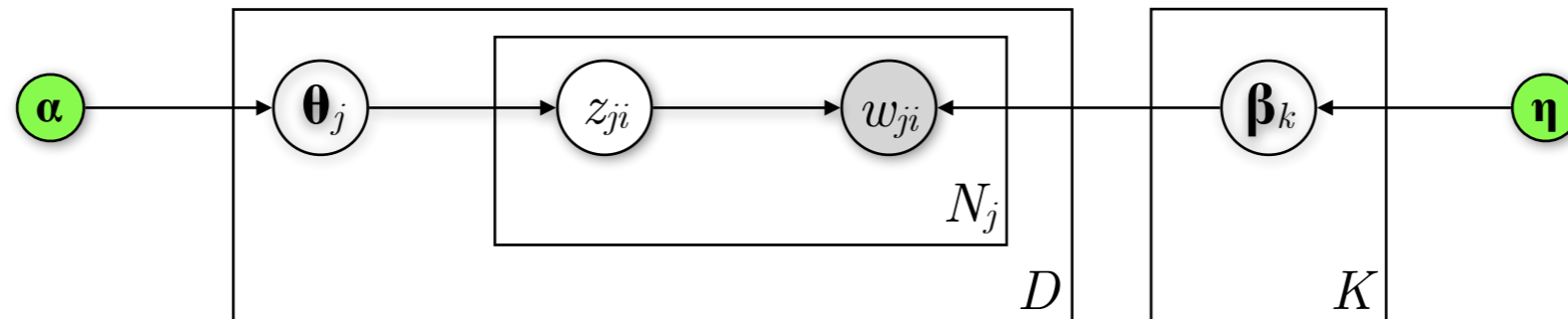


- The number of parameters that we need to infer during training grows linearly with the number of documents (D), which ultimately leads to overfitting.
- pLSA learns $p(z_k | d_j)$ only for the documents it sees during the training phase. To deal with a new document, it needs to repeat EM (retrain).

Latent Dirichlet Allocation (LDA)

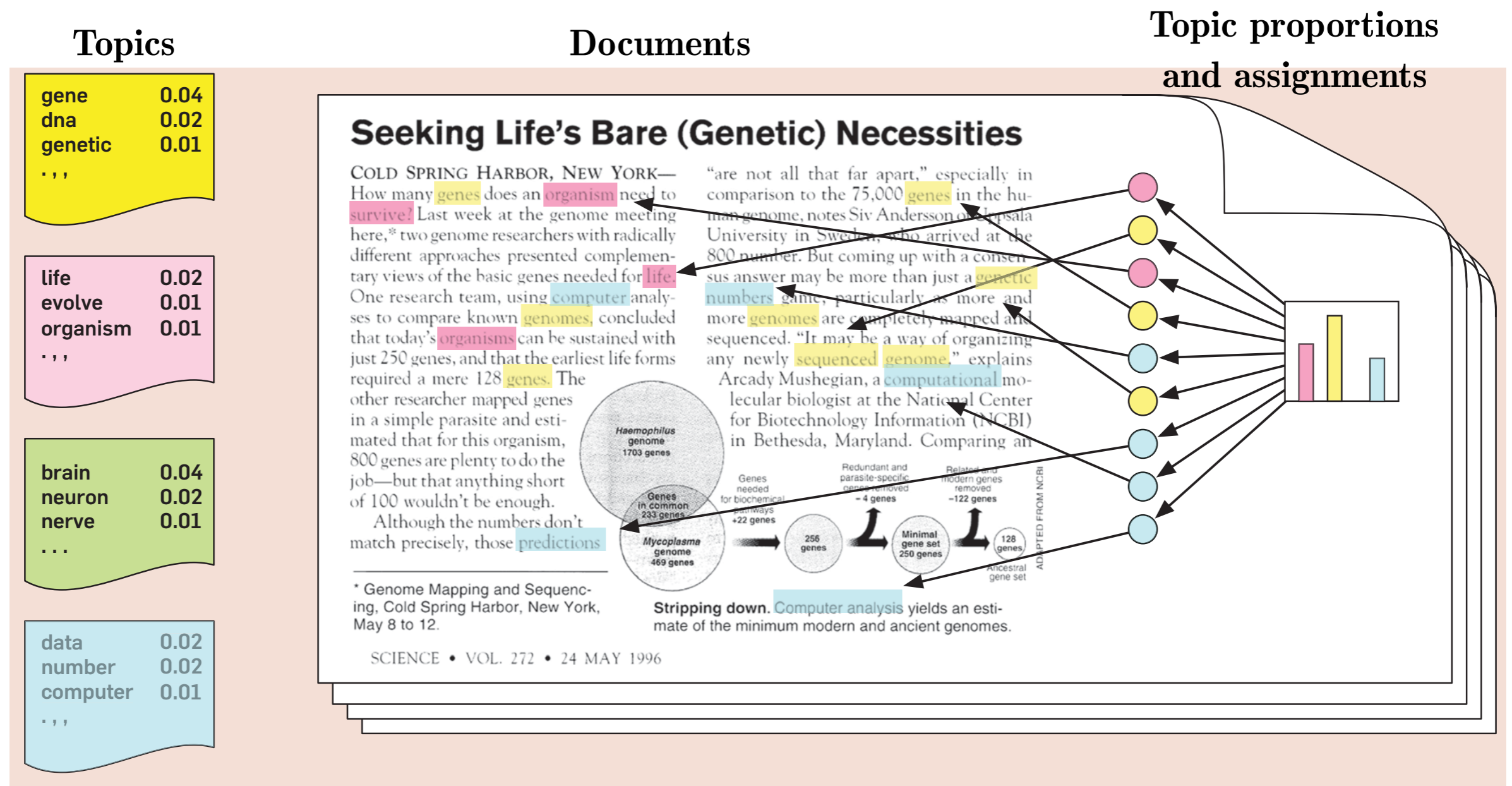


Latent Dirichlet Allocation (LDA)



- For each of the K topics draw a multinomial distribution β_k from a Dirichlet distribution with parameter η
- For each of the D documents draw a multinomial distribution θ_j from a Dirichlet distribution with parameter α
- 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 θ_j
 - Choose the observed word w_{ji} from the multinomial distribution parametrised by $\beta_{z_{ji}}$

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.

LDA — Generative story

Topics

gene	0.04
dna	0.02
genetic	0.01
...	

life	0.02
evolve	0.01
organism	0.01
...	

brain	0.04
neuron	0.02
nerve	0.01
...	

data	0.02
number	0.02
computer	0.01
...	

Documents

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many **genes** does an **organism** need to **survive**? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for **life**. One research team, using **computer** analyses to compare known **genomes**, concluded that today's **organisms** can be sustained with just 250 genes, and that the earliest life forms required a mere 128 **genes**. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough. Although the numbers don't match precisely, those **predictions**

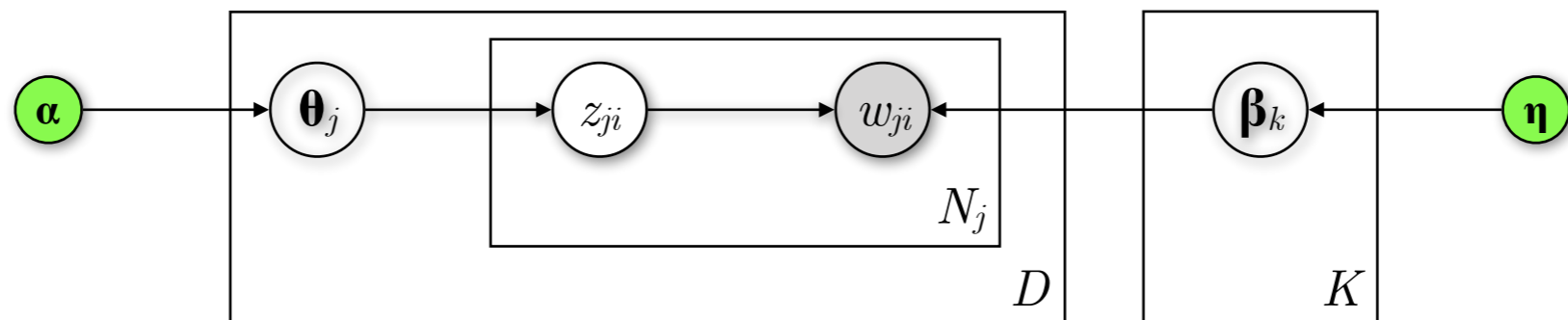
"are not all that far apart," especially in comparison to the 75,000 **genes** in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a **genetic numbers** game, particularly as more and more **genomes** are completely mapped and sequenced. "It may be a way of organizing any newly **sequenced genome**," explains Arcady Mushegian, a **computational** molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

SCIENCE • VOL. 272 • 24 MAY 1996

ADAPTED FROM NCBI

Topic proportions and assignments



LDA — Generative story

Topics

gene 0.04
dna 0.02
genetic 0.01
...

life 0.02
evolve 0.01
organism 0.01
...

brain 0.04
neuron 0.02
nerve 0.01
...

data 0.02
number 0.02
computer 0.01
...

Documents

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

"are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



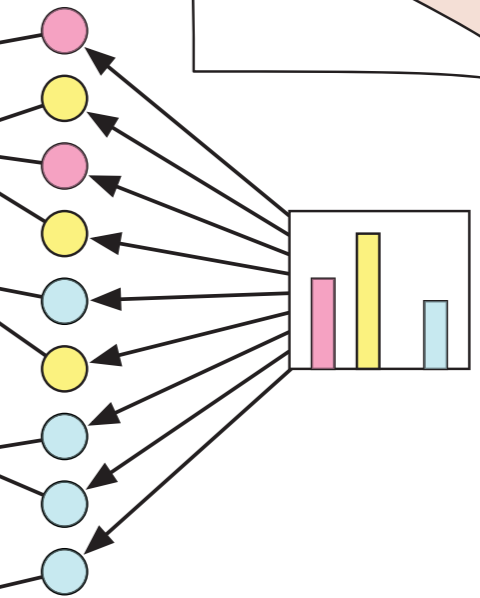
Although the numbers don't match precisely, those predictions

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

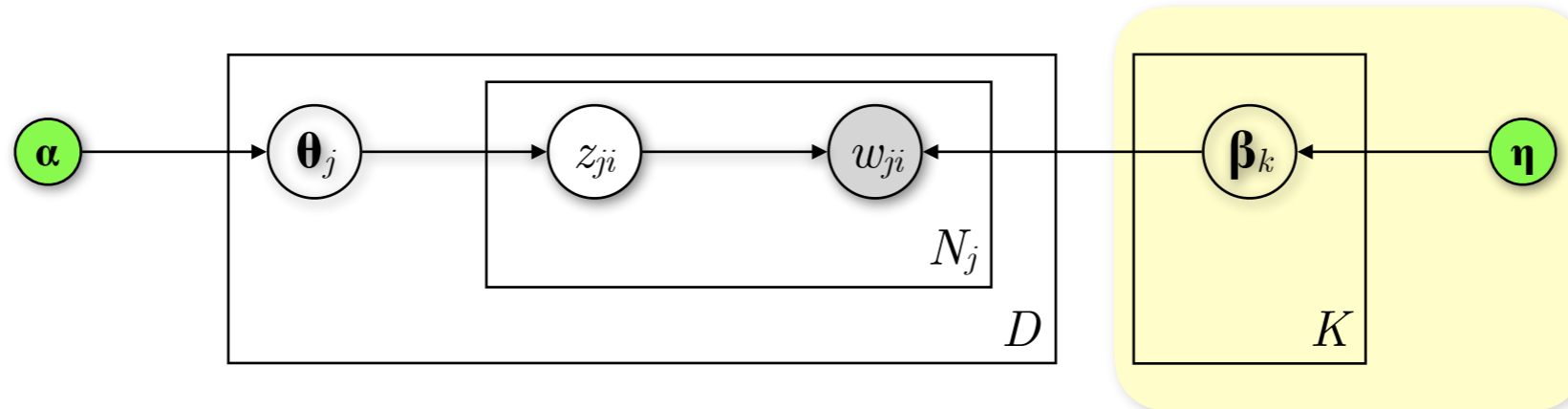
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

SCIENCE • VOL. 272 • 24 MAY 1996

Topic proportions and assignments



For each of the K topics draw a multinomial distribution (over words) β_k from a Dirichlet distribution with parameter η



LDA — Generative story

Topics

gene 0.04
dna 0.02
genetic 0.01
...

life 0.02
evolve 0.01
organism 0.01
...

brain 0.04
neuron 0.02
nerve 0.01
...

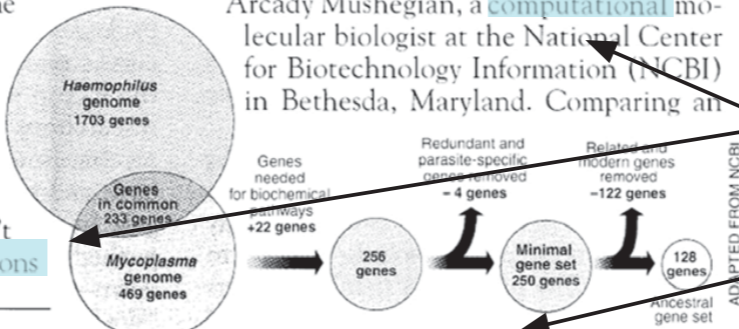
data 0.02
number 0.02
computer 0.01
...

Documents

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

"are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

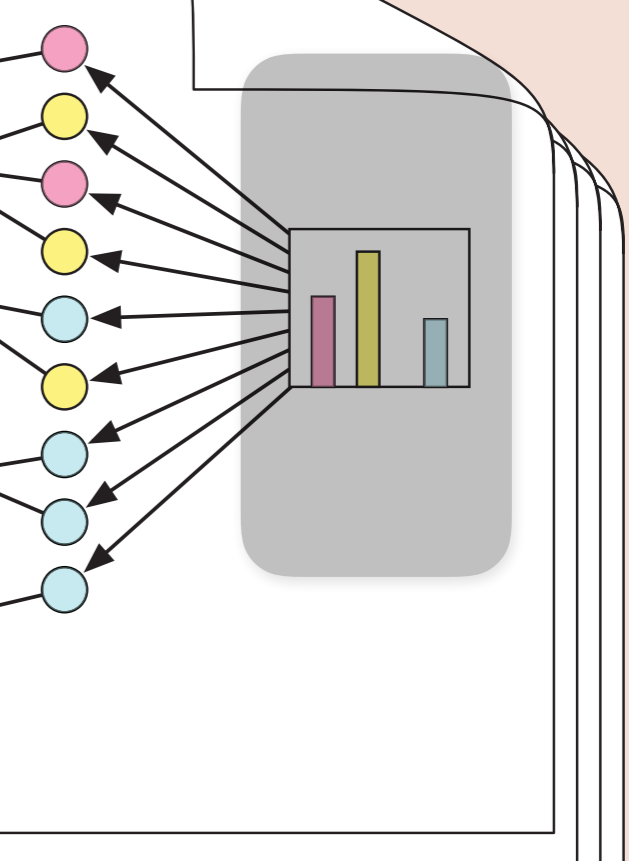


Although the numbers don't match precisely, those predictions

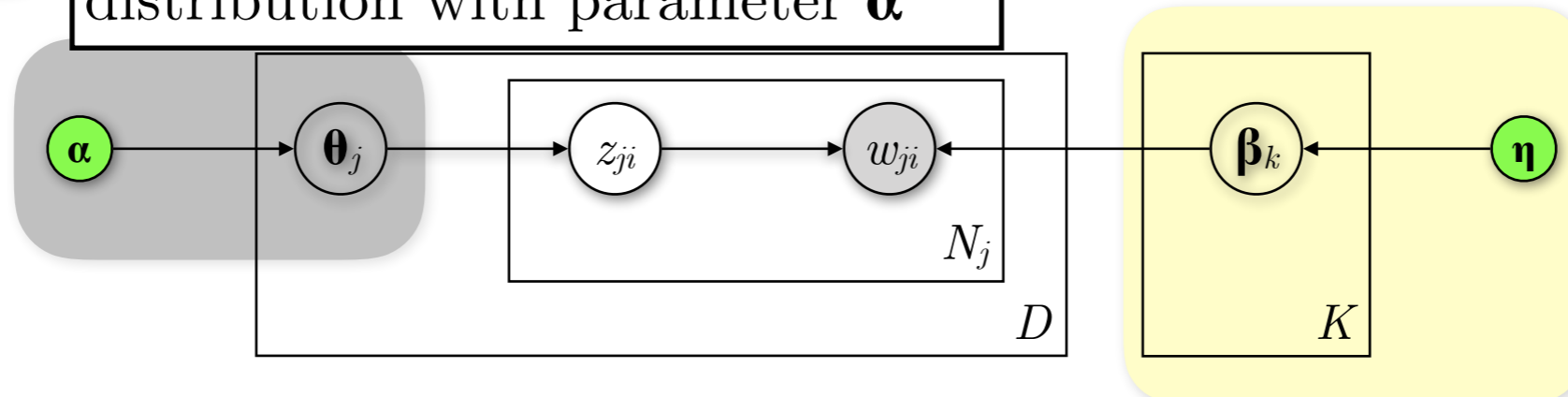
* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

Topic proportions and assignments



For each of the D documents draw a multinomial distribution (over topics) θ_j from a Dirichlet distribution with parameter α



LDA — Generative story

Topics

gene 0.04
dna 0.02
genetic 0.01
...

life 0.02
evolve 0.01
organism 0.01
...

brain 0.04
neuron 0.02
nerve 0.01
...

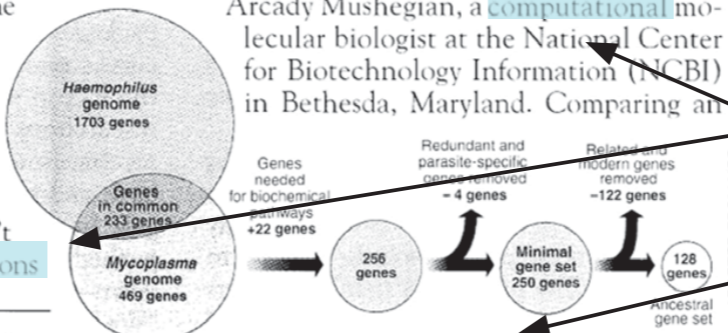
data 0.02
number 0.02
computer 0.01
...

Documents

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

"are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



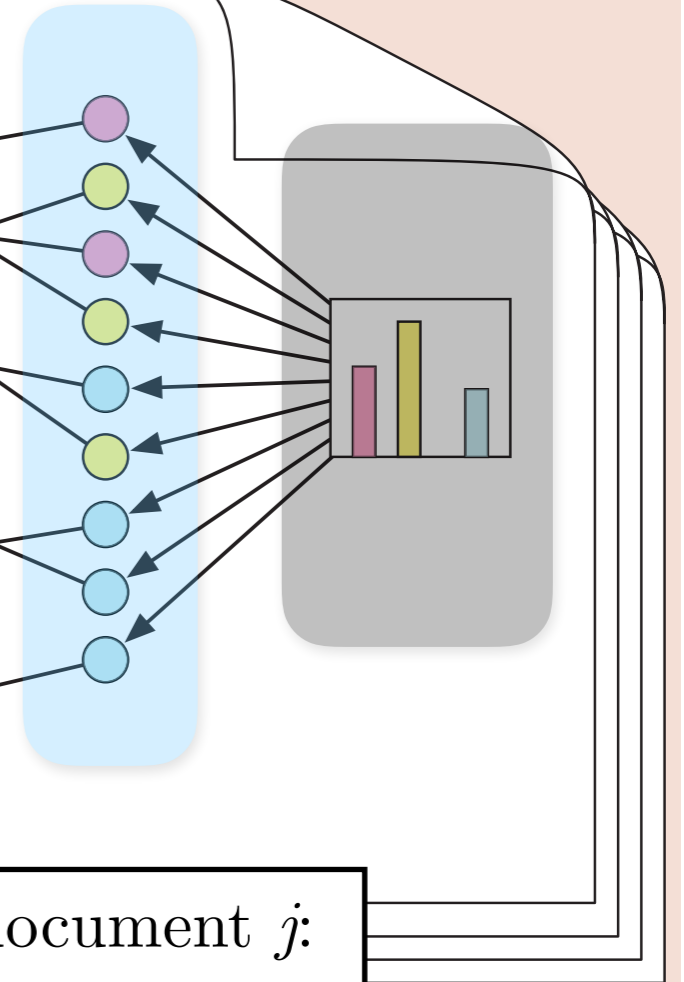
Although the numbers don't match precisely, those predictions

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

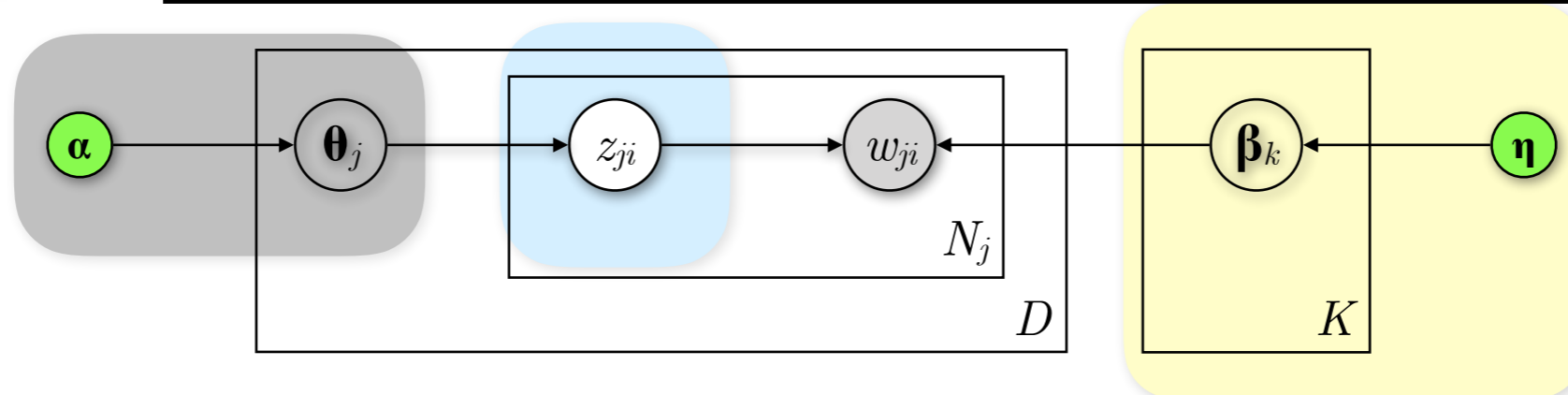
Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

SCIENCE • VOL. 272 • 24 MAY 1996

Topic proportions and assignments



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 θ_j



LDA — Generative story

Topics

gene	0.04
dna	0.02
genetic	0.01
...	

life	0.02
evolve	0.01
organism	0.01
...	

brain	0.04
neuron	0.02
nerve	0.01
...	

data	0.02
number	0.02
computer	0.01
...	

Documents

Seeking Life's Bare (Genetic) Necessities

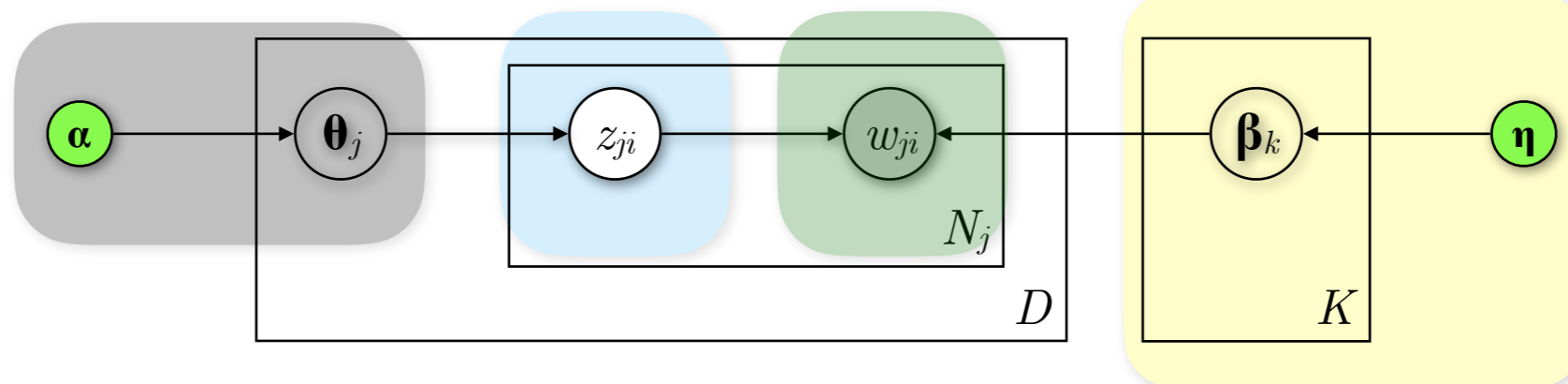
COLD SPRING HARBOR, NEW YORK— How many **genes** does an **organism** need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for **life**. One research team, using **computer** analyses to compare known **genomes**, concluded that today's **organisms** can be sustained with just 250 genes, and that the earliest life forms required a mere 128 **genes**. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough. Although the numbers don't match precisely, those **predictions**

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

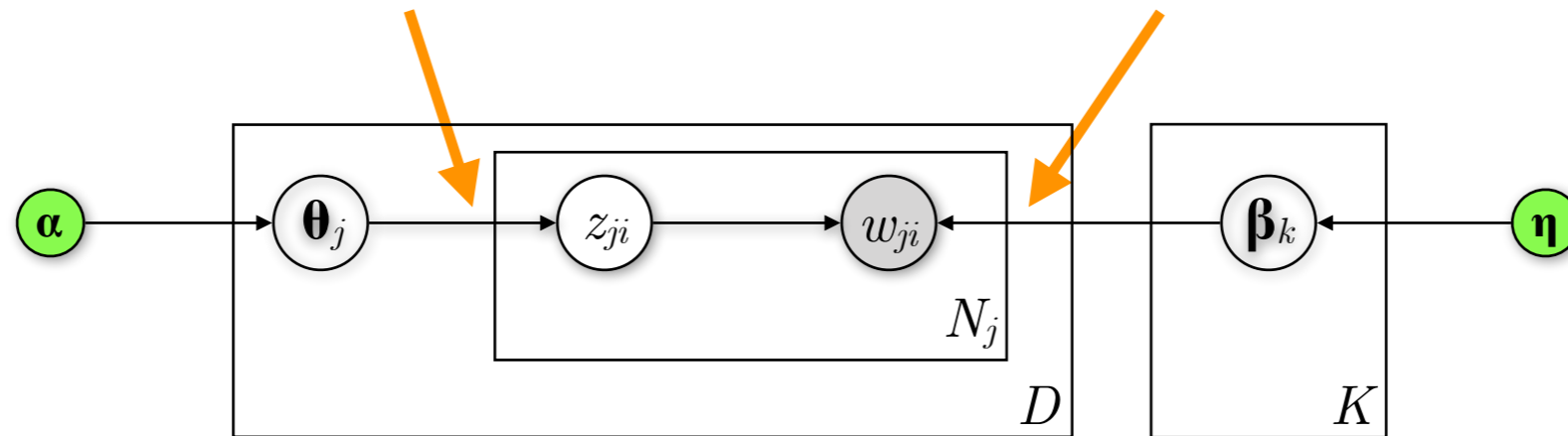
Topic proportions and assignments

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 $\beta^{z_{ji}}$



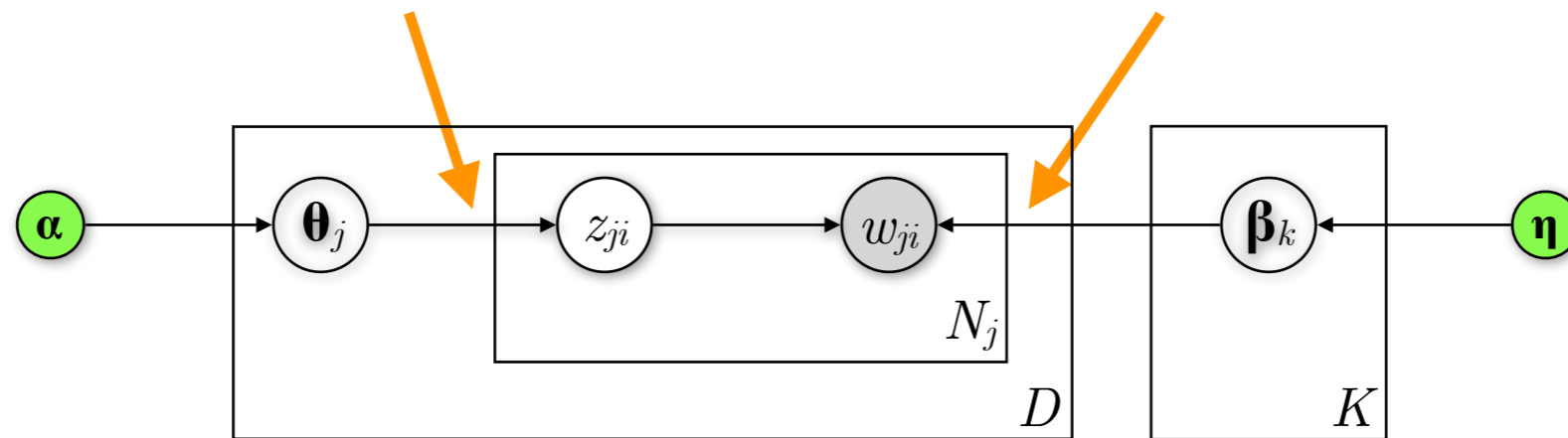
LDA — Multinomial distribution (Mult)



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)



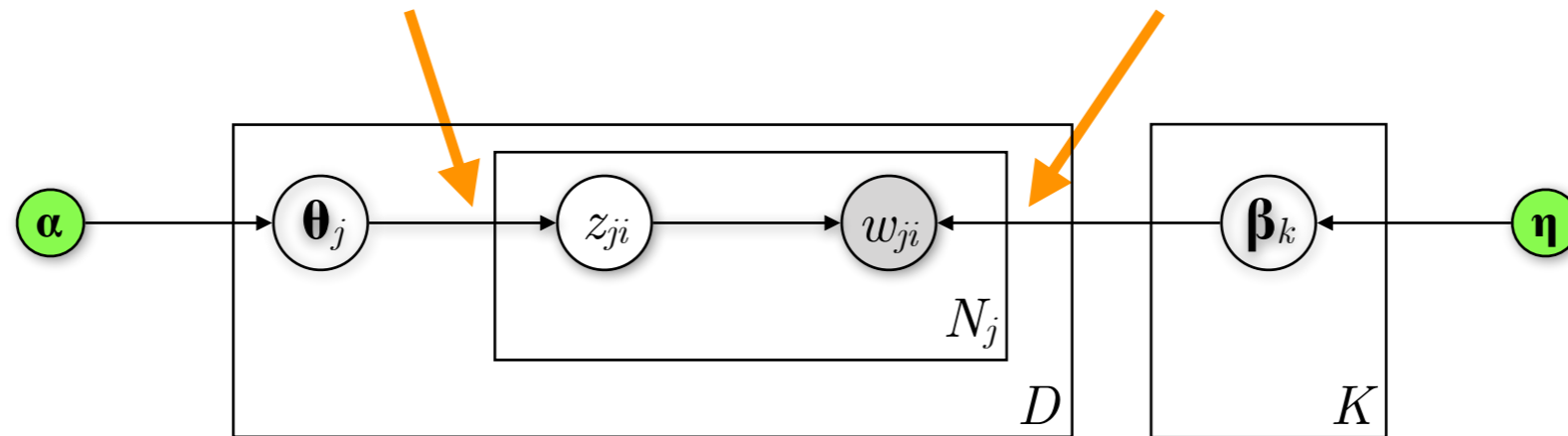
What is the probability of a set of outcomes for an event that has multiple outcomes?

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

$$\frac{5!}{1!4!} \cdot \left(\frac{1}{6}\right) \cdot \left(\frac{1}{6}\right)^4 \approx 0.00064$$

#ways to get 1 “3” and 4 “6”s prob. of 1 “3” prob. of 4 “6”s

LDA — Multinomial distribution (Mult)



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

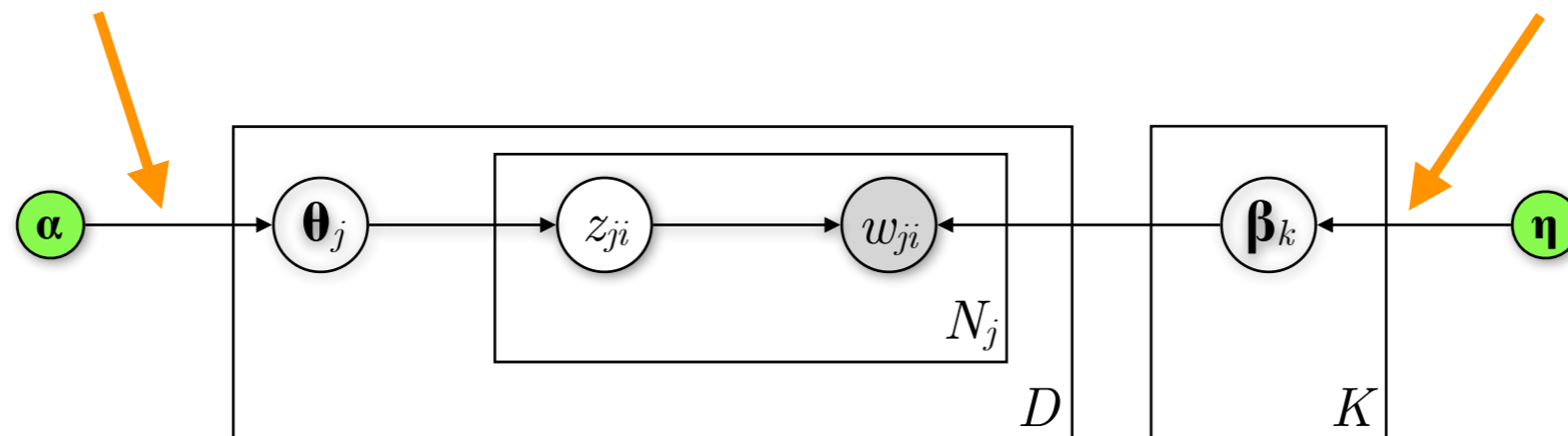
— We roll a 6-sided dice 5 times. What is the probability of getting a “3” 1 time and a “6” 4 times?

$$\frac{5!}{1!4!} \cdot \left(\frac{1}{6}\right) \cdot \left(\frac{1}{6}\right)^4 \approx 0.00064$$

#ways to get 1 “3” and 4 “6”s prob. of 1 “3” prob. of 4 “6”s

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

LDA — Dirichlet distribution (Dir)



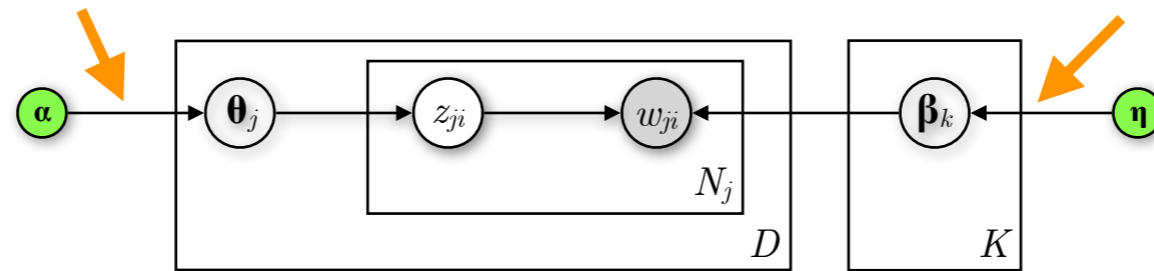
Exponential family distribution over the simplex (= positive vectors with elements that sum up to 1), essentially a distribution over multinomial distributions

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

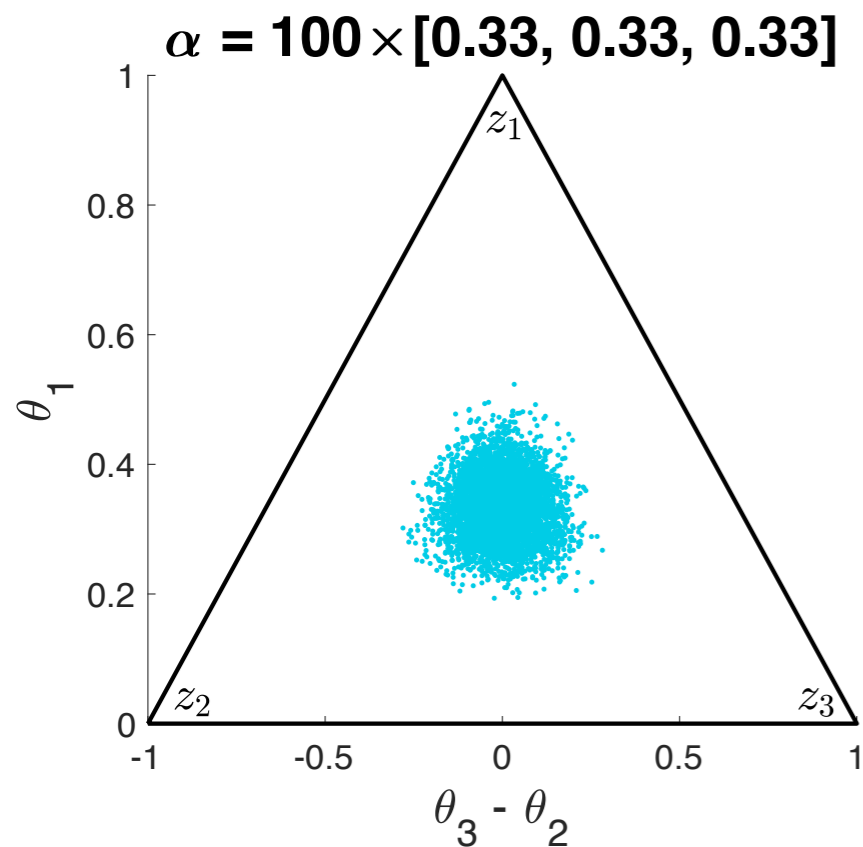
Parameter $\boldsymbol{\alpha}$ controls the mean shape and sparsity of $\boldsymbol{\theta}$ (same for $\boldsymbol{\beta}$)

Note: $\boldsymbol{\alpha}$ is a vector of K (= number of topics) 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)

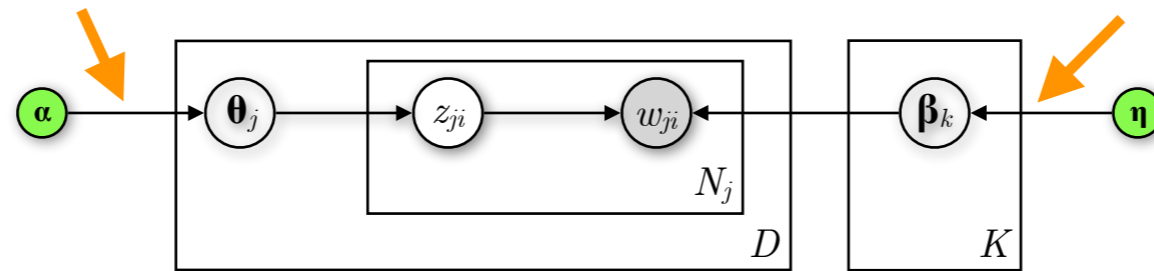


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

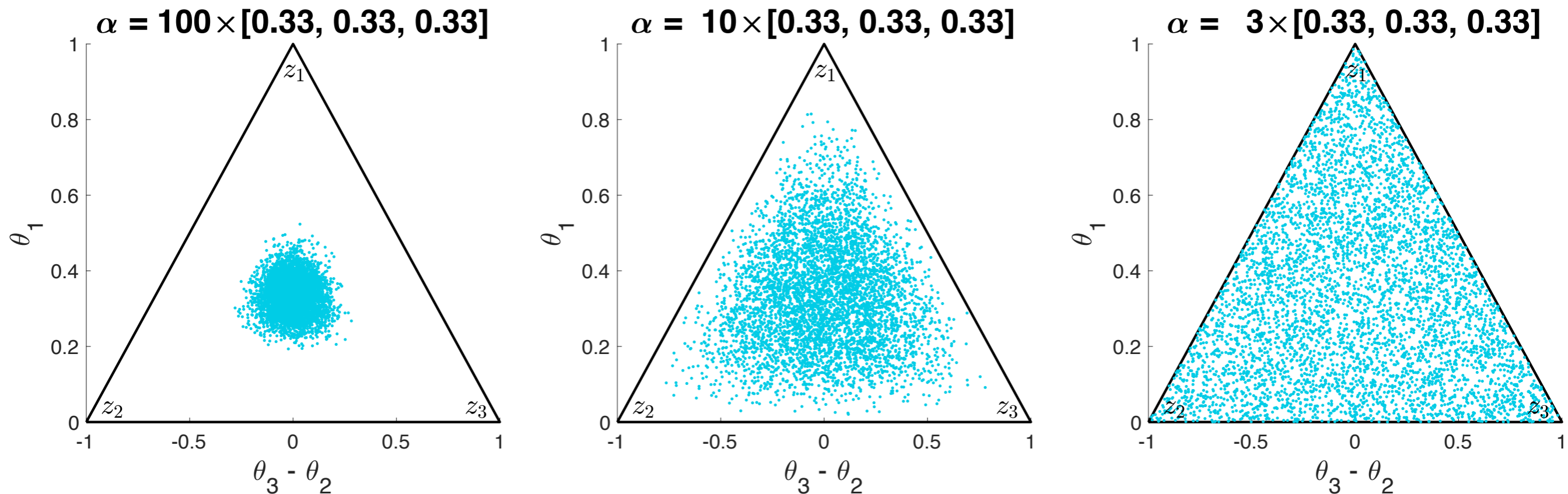


Large values of $\boldsymbol{\alpha}$ lead to more dense $\boldsymbol{\theta}$'s

LDA — Dirichlet distribution (Dir)

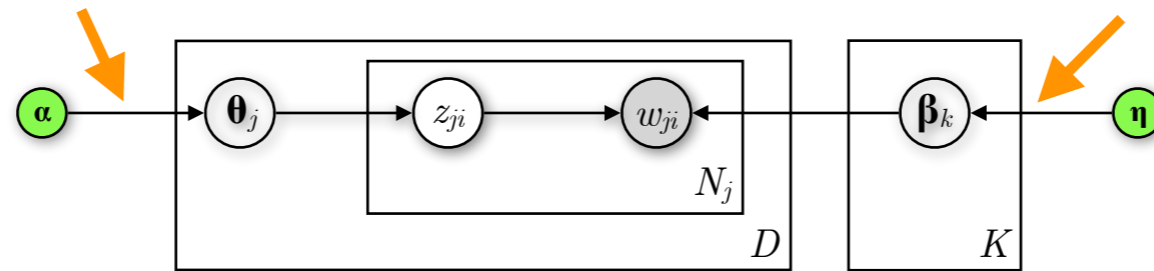


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

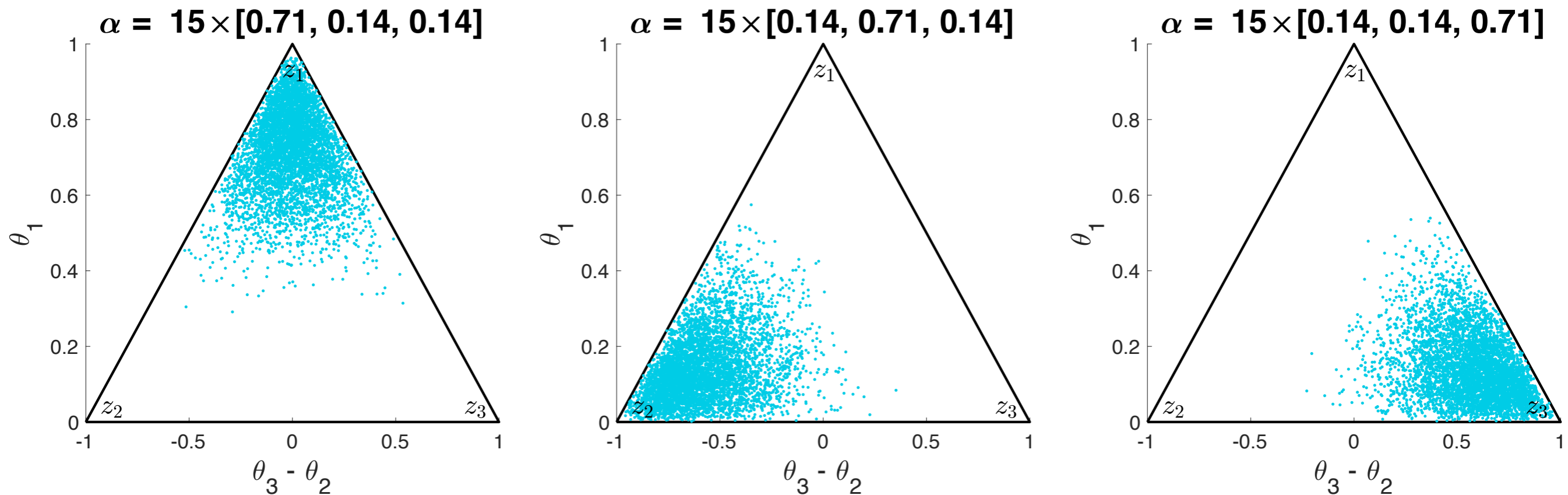


Large values of $\boldsymbol{\alpha}$ lead to more dense $\boldsymbol{\theta}$'s

LDA — Dirichlet distribution (Dir)

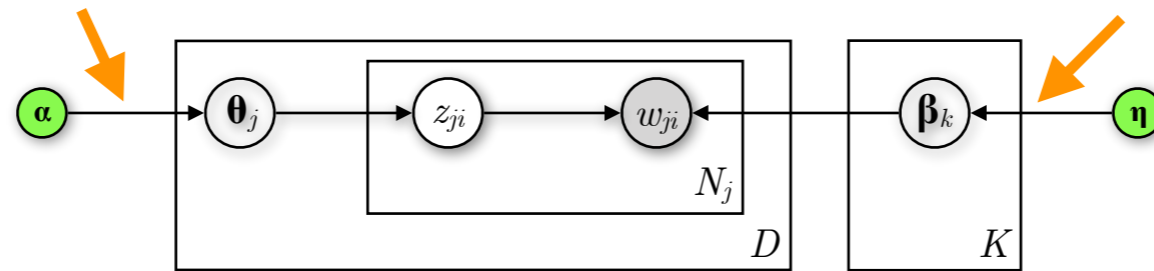


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

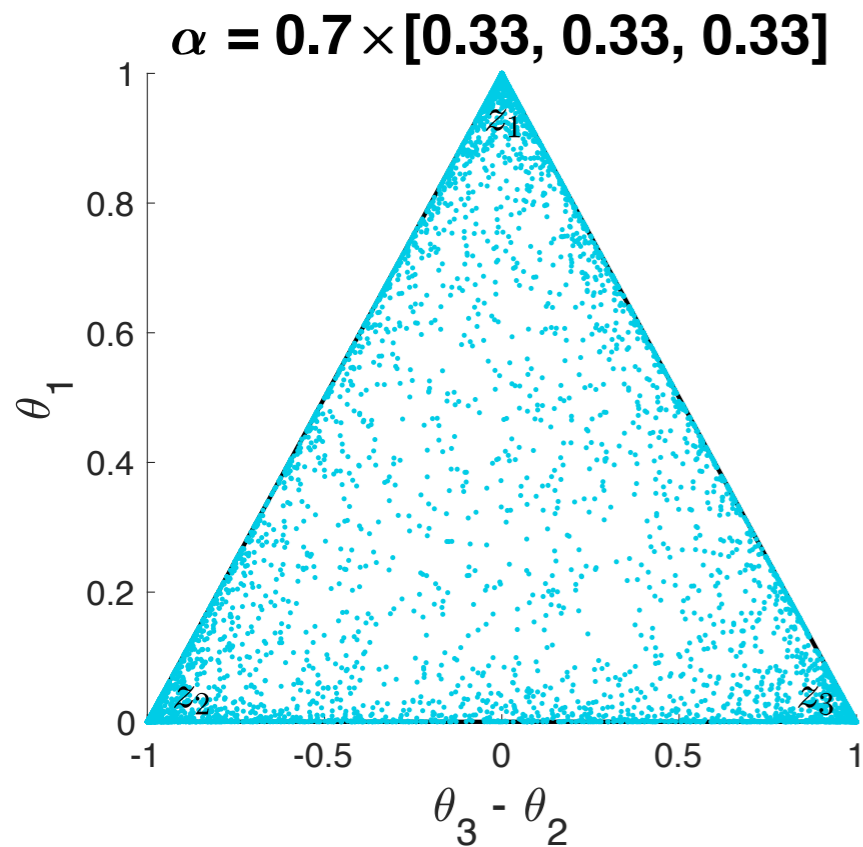


Imbalance in $\boldsymbol{\alpha}$ shapes the focus of the distribution

LDA — Dirichlet distribution (Dir)

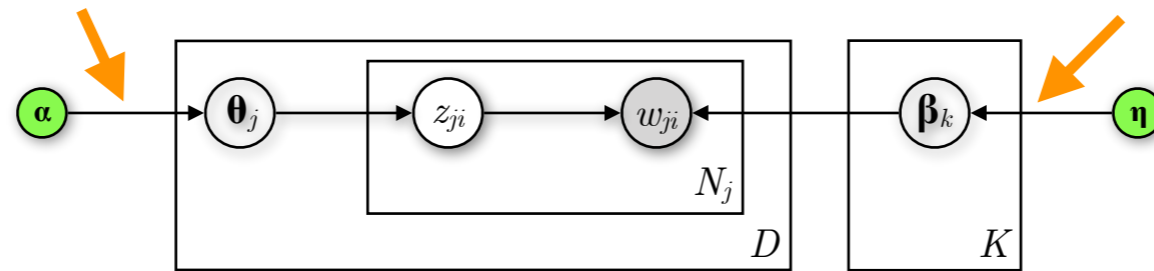


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

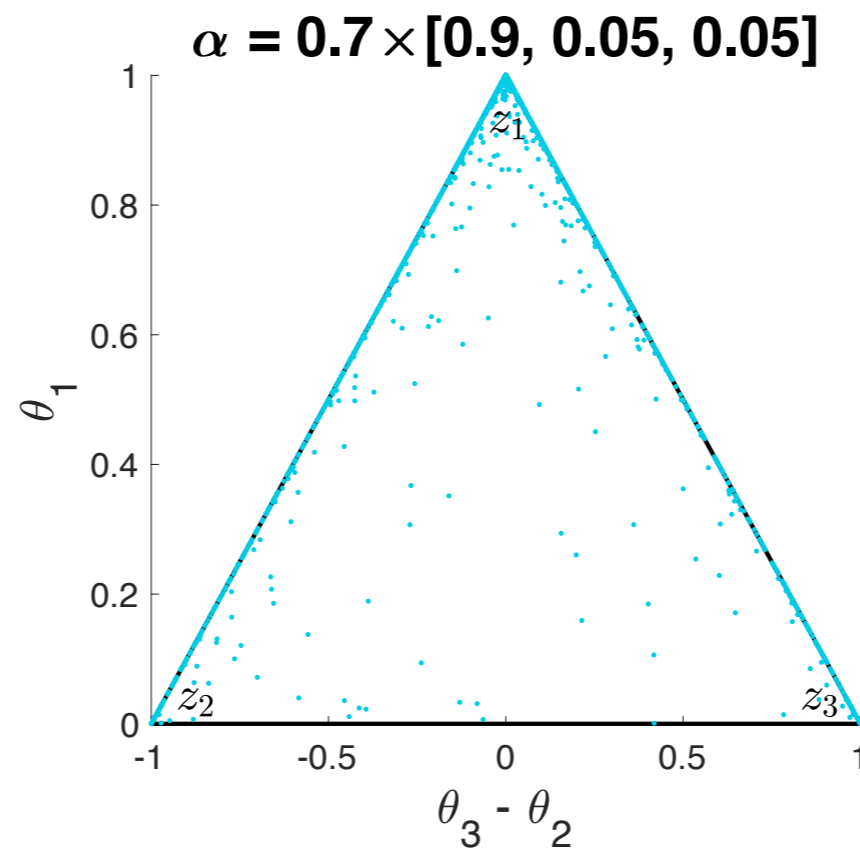
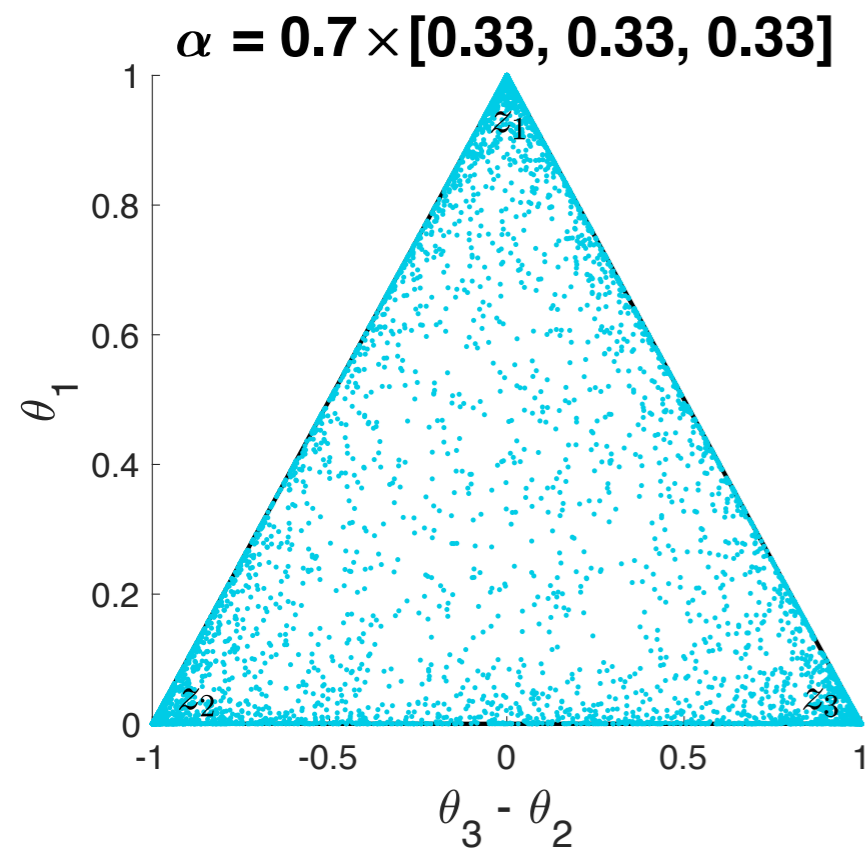


Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

LDA — Dirichlet distribution (Dir)

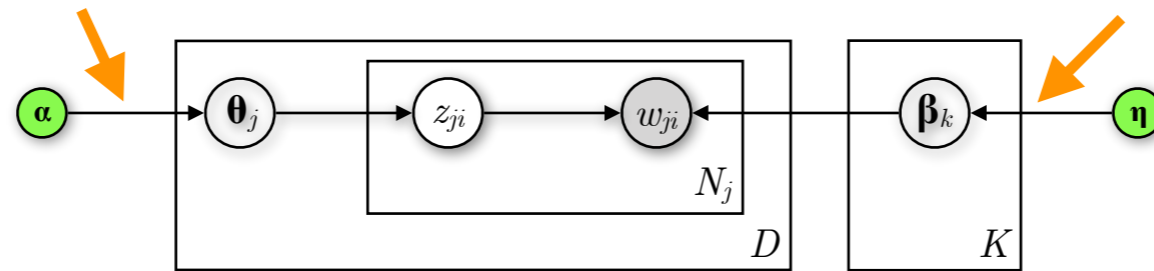


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.

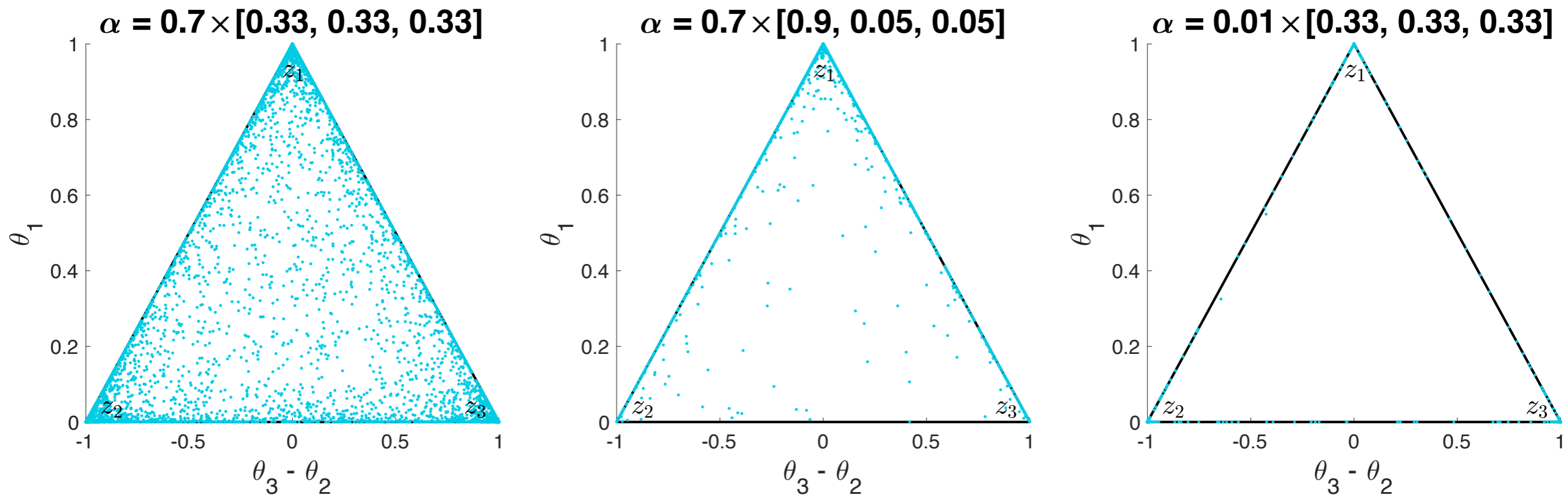


Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

LDA — Dirichlet distribution (Dir)

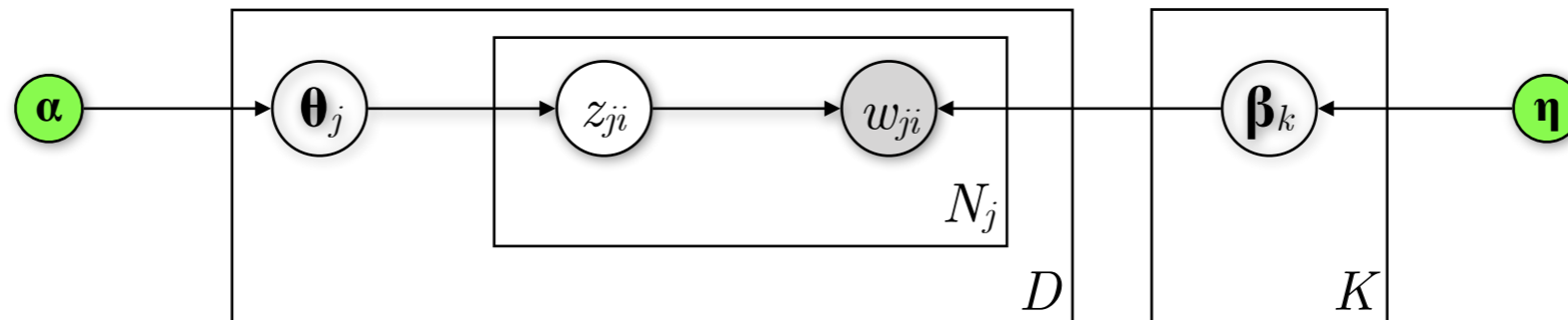


Assume a simplex $\boldsymbol{\theta} = [\theta_1, \theta_2, \theta_3]$ across $K = 3$ topics ($0 \leq \theta_i \leq 1$). How do different values for $\boldsymbol{\alpha}$ affect the $\boldsymbol{\theta}$ produced by the Dirichlet distribution? Let's plot 5,000 samples for different $\boldsymbol{\alpha}$'s.



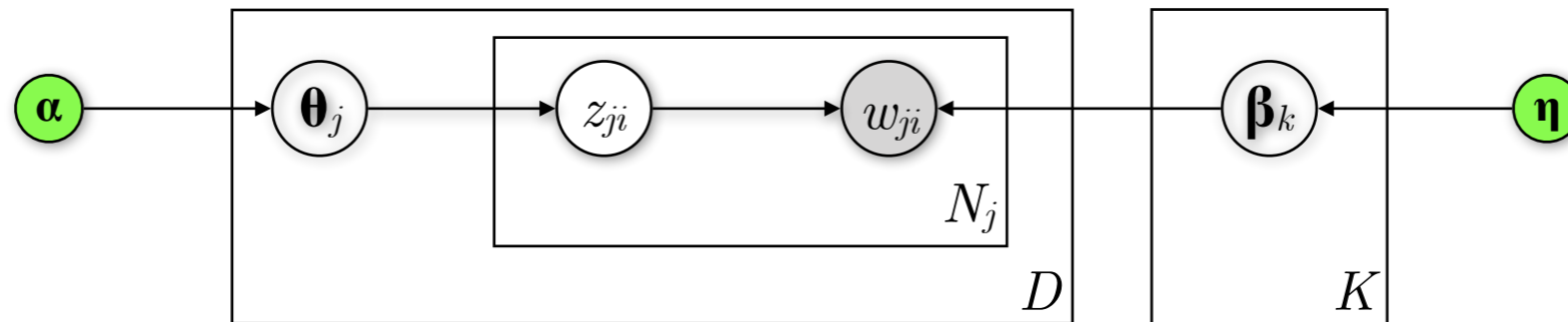
Values of $\boldsymbol{\alpha} < 1$ create increasingly sparse outputs

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

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

We are interested in this posterior

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

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 $x_i, i = 1, \dots, N$

For $t = 1, \dots, T$:

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

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

...

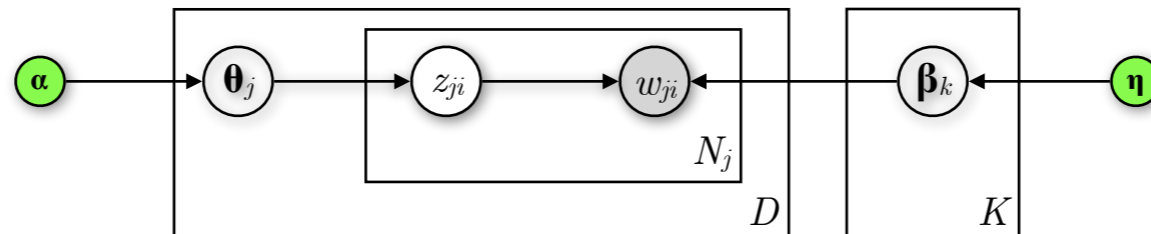
$$\text{Sample } x_j^{(t+1)} \sim p \left(x_j \mid x_1^{(t+1)}, x_2^{(t+1)}, \dots, x_{j-1}^{(t+1)}, x_{j+1}^{(t)}, \dots, x_N^{(t)} \right)$$

...

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

LDA — Inference; Gibbs sampling

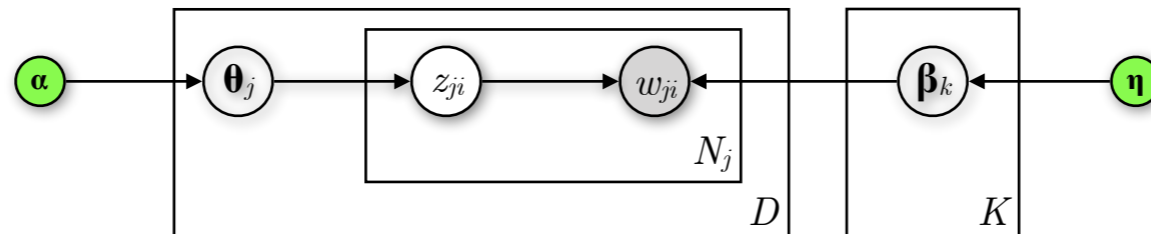
- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*notation indicating the exclusion of w_{ji}*)



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 $w_{j,-i}$ (*notation indicating the exclusion of w_{ji}*)

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



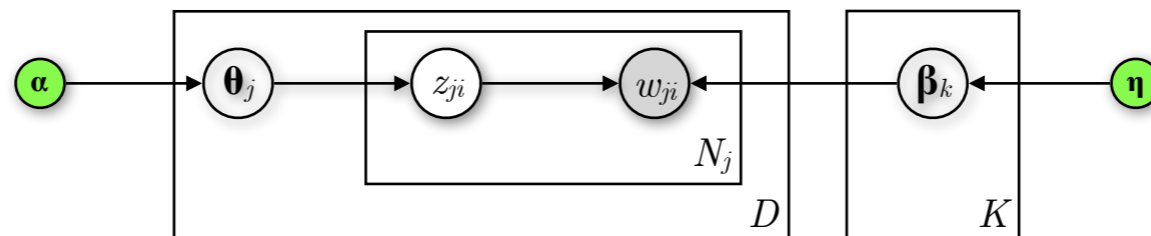
LDA — Inference; Gibbs sampling

- 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 $w_{j,-i}$ (notation indicating the exclusion of w_{ji})

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

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}



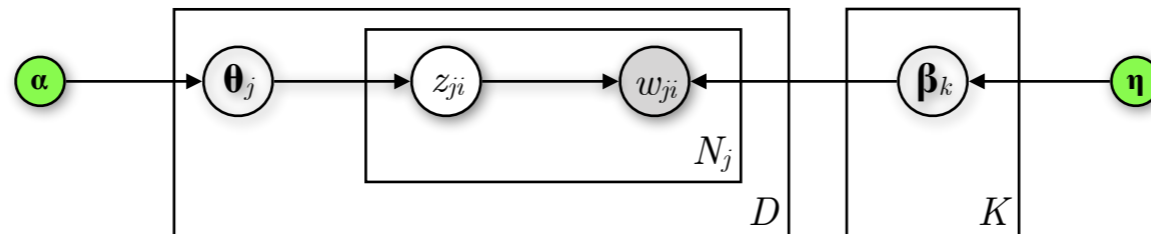
LDA — Inference; Gibbs sampling

- 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 $w_{j,-i}$ (notation indicating the exclusion of w_{ji})

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

How much does document j “like” topic k ?

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



LDA — Inference; Gibbs sampling

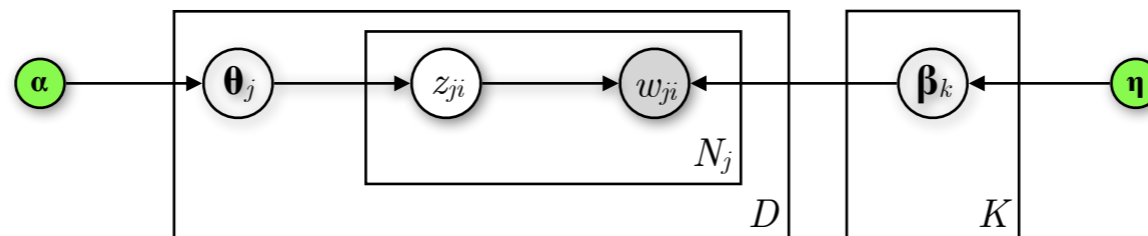
- Initialise probabilities randomly or uniformly
- Go over each word i in every document j (w_{ji})
- Estimate the probability of assigning w_{ji} to each topic, conditioned on the topic assignments ($z_{j,-i}$) of all other words $w_{j,-i}$ (*notation indicating the exclusion of w_{ji}*)

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

How much does document j “like” topic k ?

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

- 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 $K = 3$ topics

LDA — Gibbs sampling; toy example

- Consider $K = 3$ topics
- **Sampling from document 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 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 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
		Topic 1	Topic 2		Topic 3	

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

LDA — 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

Topic 1
Topic 2
Topic 3

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
...

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

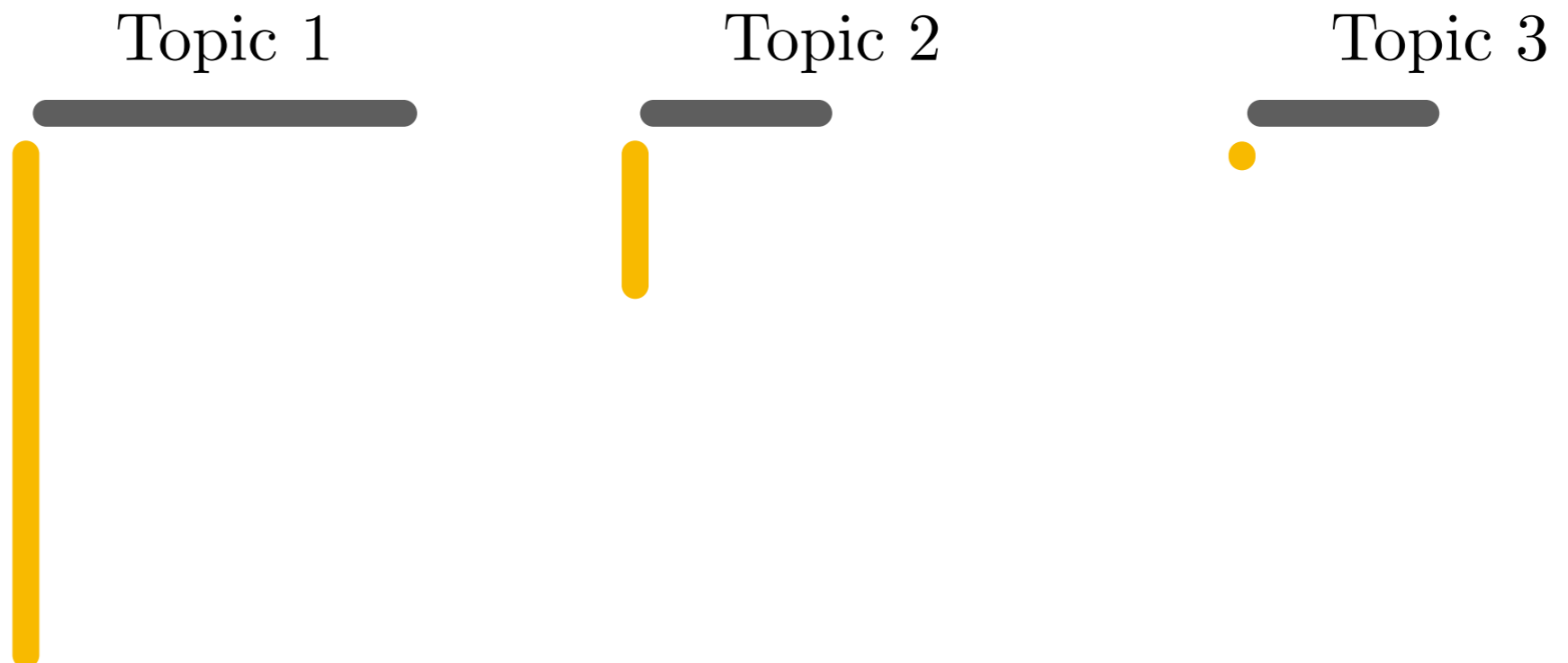
LDA — Gibbs sampling; toy example

- ...
- 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
...



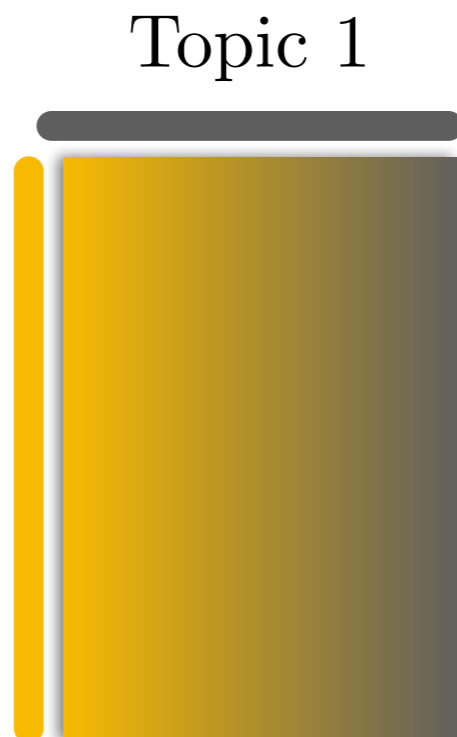
LDA — Gibbs sampling; toy example

- ...
- 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
...



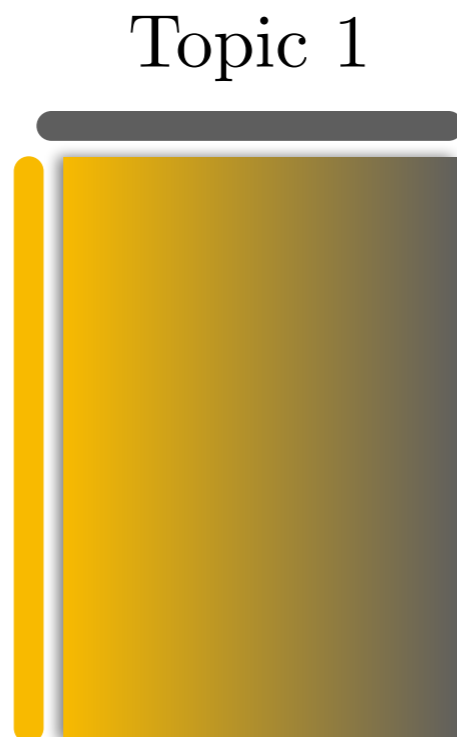
LDA — Gibbs sampling; toy example

- ...
- 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
...

Topic 1



Topic 2



Topic 3



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
...



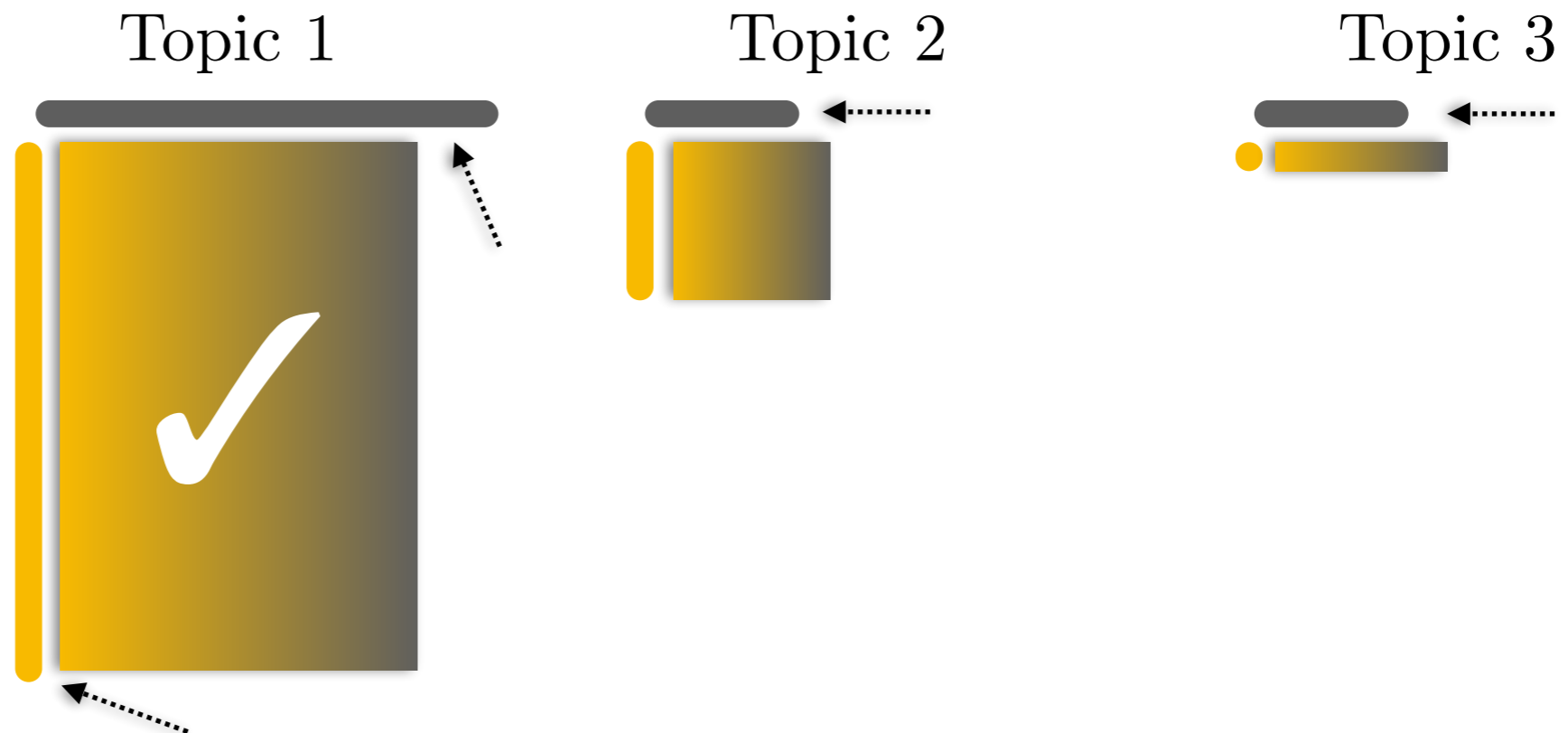
LDA — Gibbs sampling; toy example

- ...
- 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 — Vector semantics

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”

John Rupert (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”

John Rupert (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”, “a”, “new”, “old”, “display”
- Occurs before: “has”, “RAM”, “thinner”

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”, “a”, “new”, “old”, “display”
- Occurs before: “has”, “RAM”, “thinner”
- **W** = laptop / notebook / tablet

Words as vectors

- Generate a **word-word** matrix
 - a.k.a. **word-context** or **word co-occurrence** matrix

Words as vectors

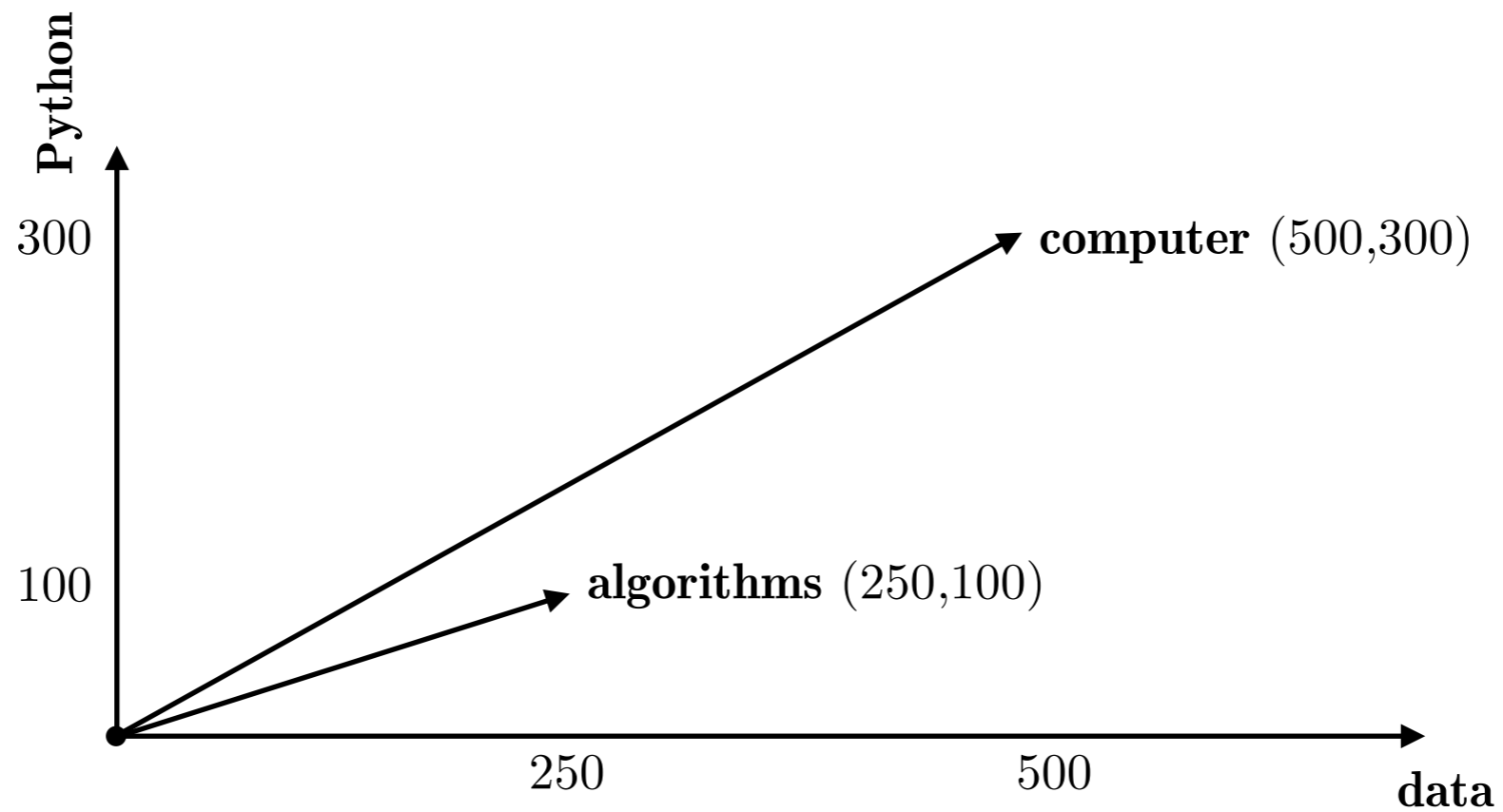
- Generate a **word-word** matrix
 - a.k.a. **word-context** or **word co-occurrence** matrix
- If the size of our vocabulary (all words) is V , then the size of this matrix is commonly $V \times V$
- Each **cell** of the matrix counts how many times two words **co-occur** within a predefined context

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 ± 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

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



Words as large, sparse vectors

- Recap: Word-context matrix of size $V \times V$ where V is the length of the vocabulary
- **Large** matrix as V is often very large ($>100,000$)
- **Sparse** matrix as many entries will be 0
(not all words co-occur in all contexts)

Words as large, sparse vectors

- Recap: Word-context matrix of size $V \times V$ where V is the length of the vocabulary
- **Large** matrix as V is often very large ($>100,000$)
- **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)

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

Computing PPMI

Assume a word-context matrix \mathbf{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. Let's generate a PPMI matrix from that.

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

Computing PPMI

Assume a word-context matrix \mathbf{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. Let's generate a PPMI matrix from that.

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

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

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 \mathbf{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 \mathbf{A}) divided by...

Computing PPMI

Assume a word-context matrix \mathbf{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. Let's generate a PPMI matrix from that.

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

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

target words in context c_j divided by the corpus size

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

target words in row i of \mathbf{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 \mathbf{A}) divided by...

We can use the PPMI matrix to measure how (semantically) similar different words are. We need a **similarity metric** for that.

Measuring word similarity — Cosine

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

Measuring word similarity — Cosine

- Dot product between word vectors w, v : $w^\top v = \sum_{i=1}^N w_i \cdot v_i$
- Larger values for longer vectors and for frequent words

Measuring word similarity — Cosine

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

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

Measuring word similarity — Cosine

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

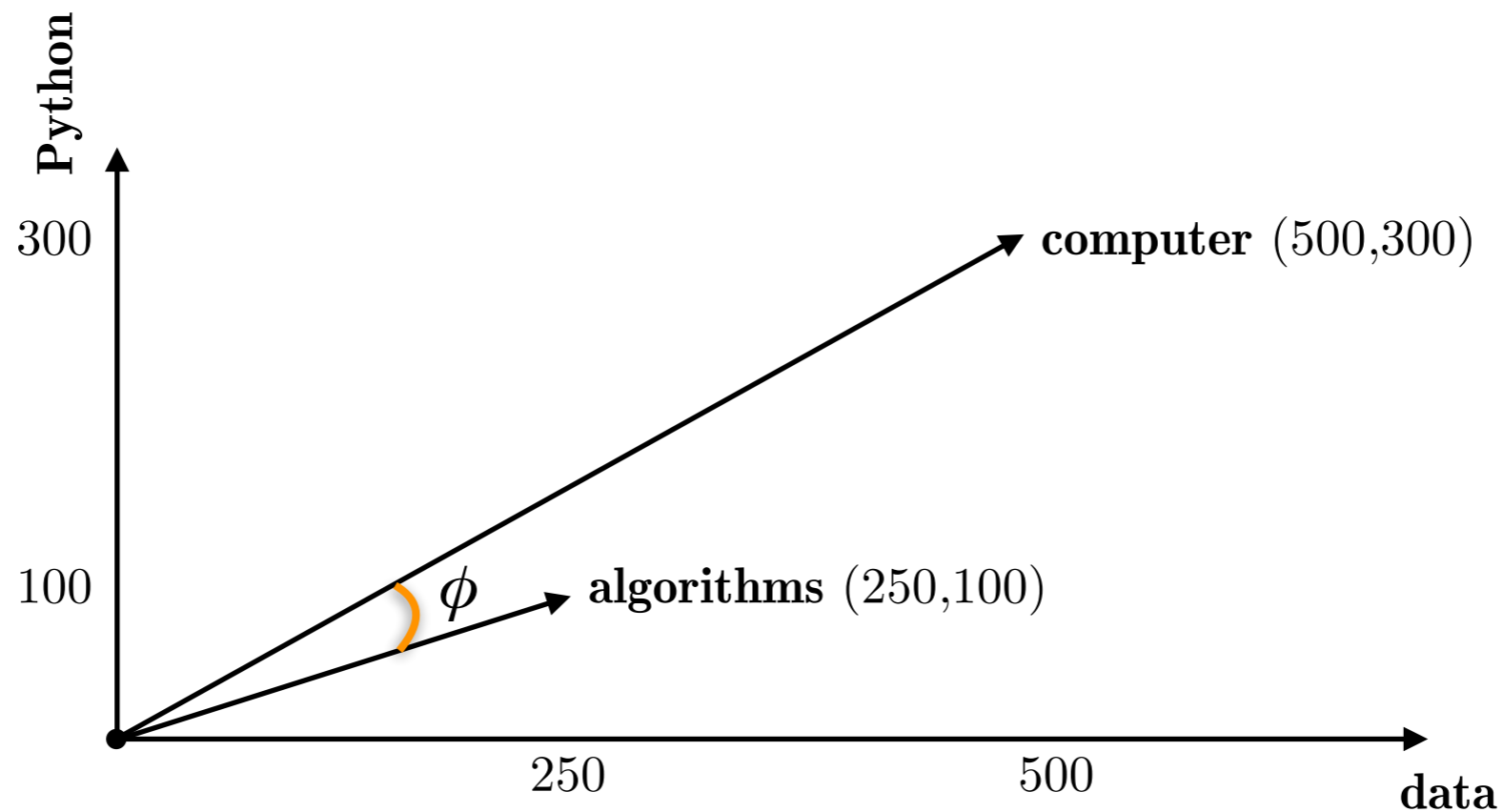
$$\text{cosine-sim}(w, v) = \frac{\sum_{i=1}^N w_i \cdot v_i}{\sqrt{\sum_{i=1}^N w_i^2} \cdot \sqrt{\sum_{i=1}^N v_i^2}} = \frac{w^\top v}{|w| |v|} = \cos \phi$$

- Since w and $v > 0$, $\text{cosine-sim}(w, v)$ ranges from $[0, 1]$
 - $\text{cosine-sim}(w, v) = 0$ means that $\phi = 90^\circ$
 - $\text{cosine-sim}(w, v) = 1$ means that $\phi = 0^\circ$

Measuring word similarity — Cosine

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

- Since w and $v > 0$, $\text{cosine-sim}(w, v)$ ranges from $[0, 1]$
 - $\text{cosine-sim}(w, v) = 0$ means that $\phi = 90^\circ$
 - $\text{cosine-sim}(w, v) = 1$ means that $\phi = 0^\circ$



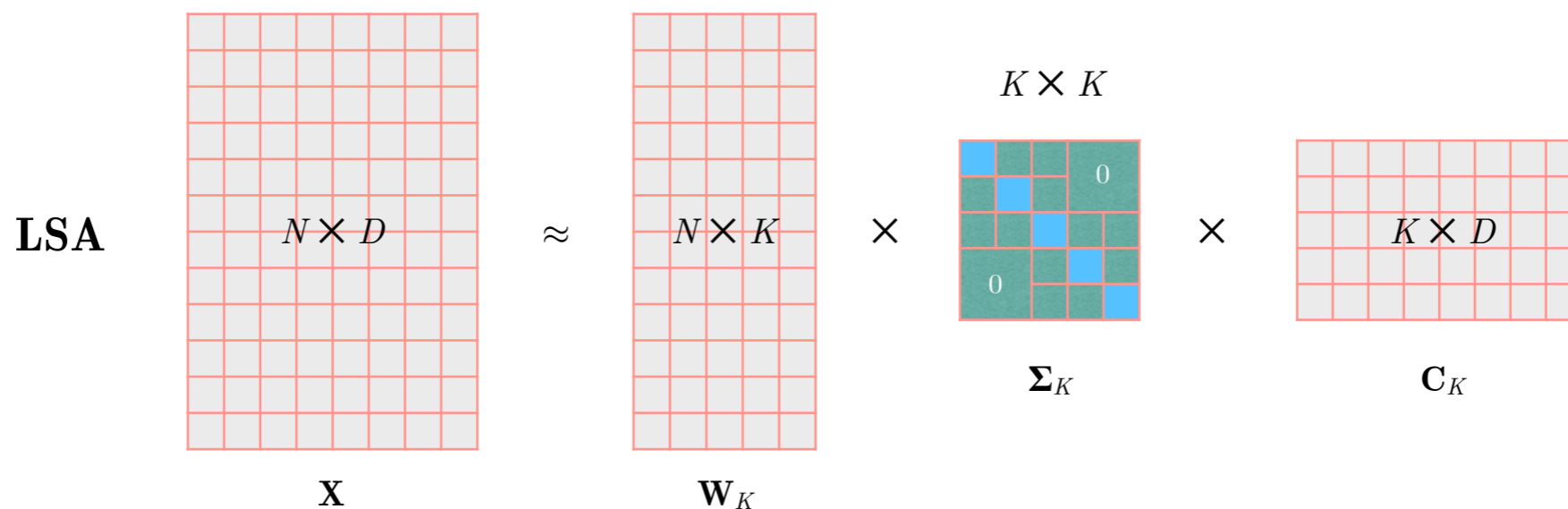
$$\text{cosine-sim}(\text{computer}, \text{algorithms}) = 0.9872, \phi = 9.162^\circ$$

From sparse to dense word vectors

- Previously shown word representations: **long** (equal to size of the vocabulary V) and **sparse** (many 0's)

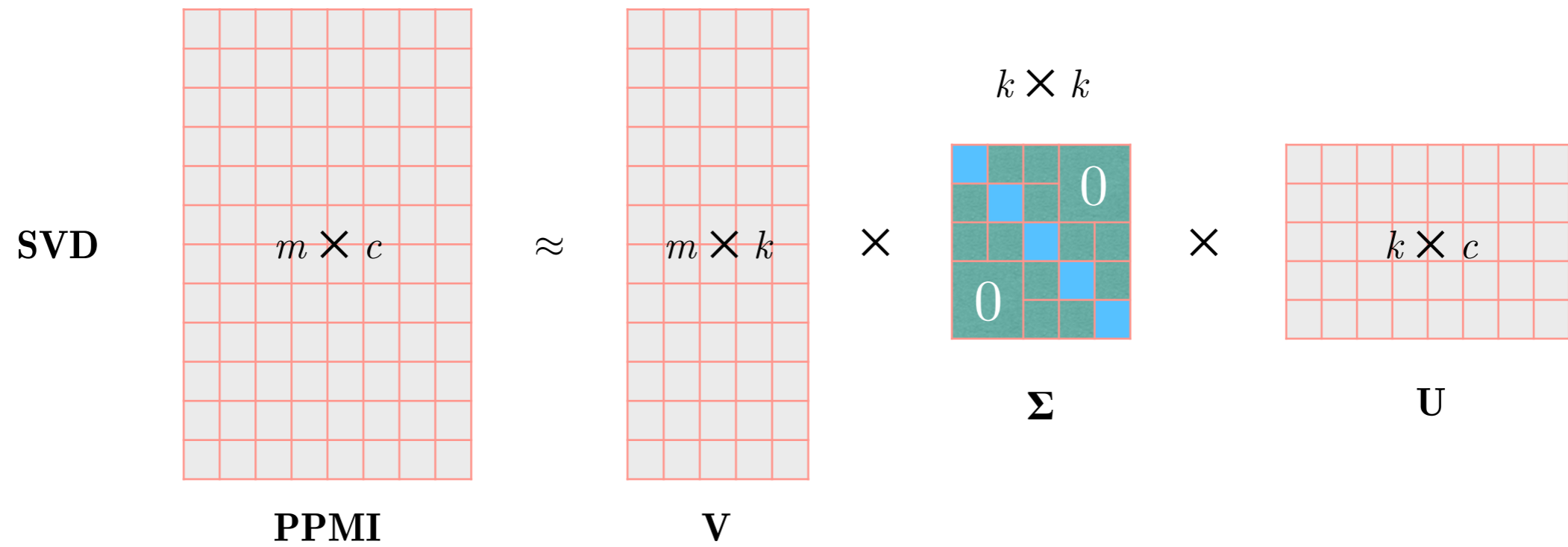
From sparse to dense word vectors

- 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

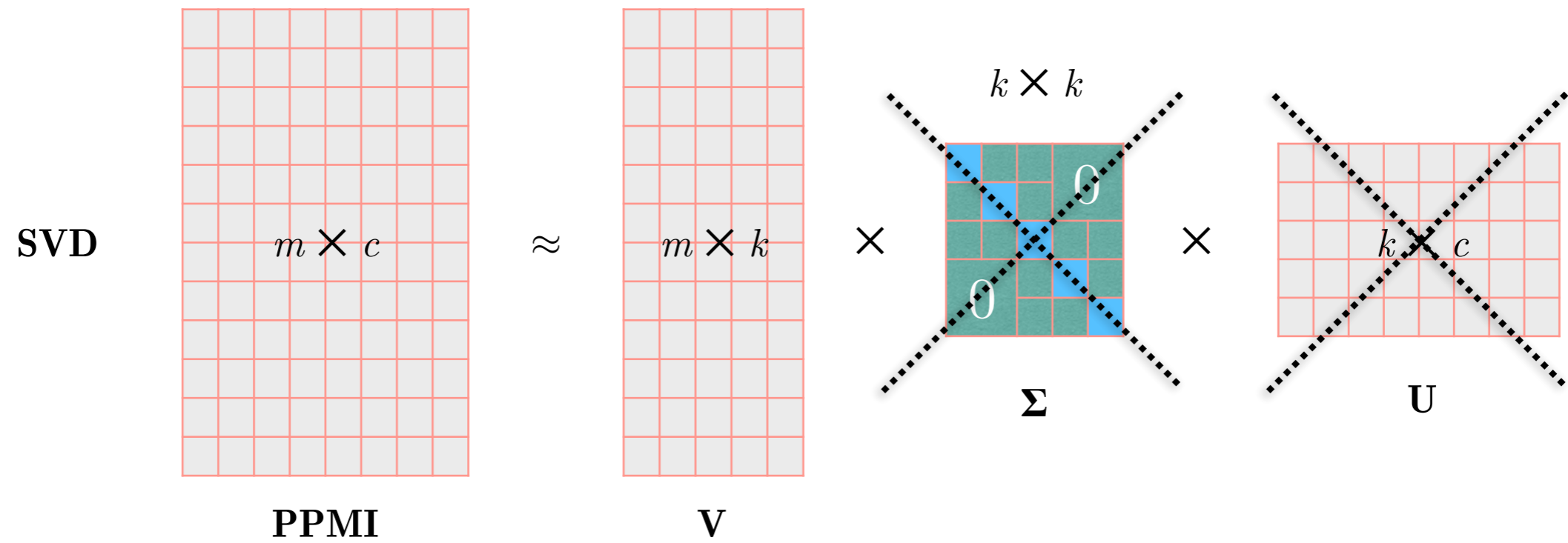


- Recall Latent Semantic Analysis (LSA), *i.e.* SVD on the word-document matrix (\mathbf{X}). What if we perform SVD on a word co-occurrence matrix?

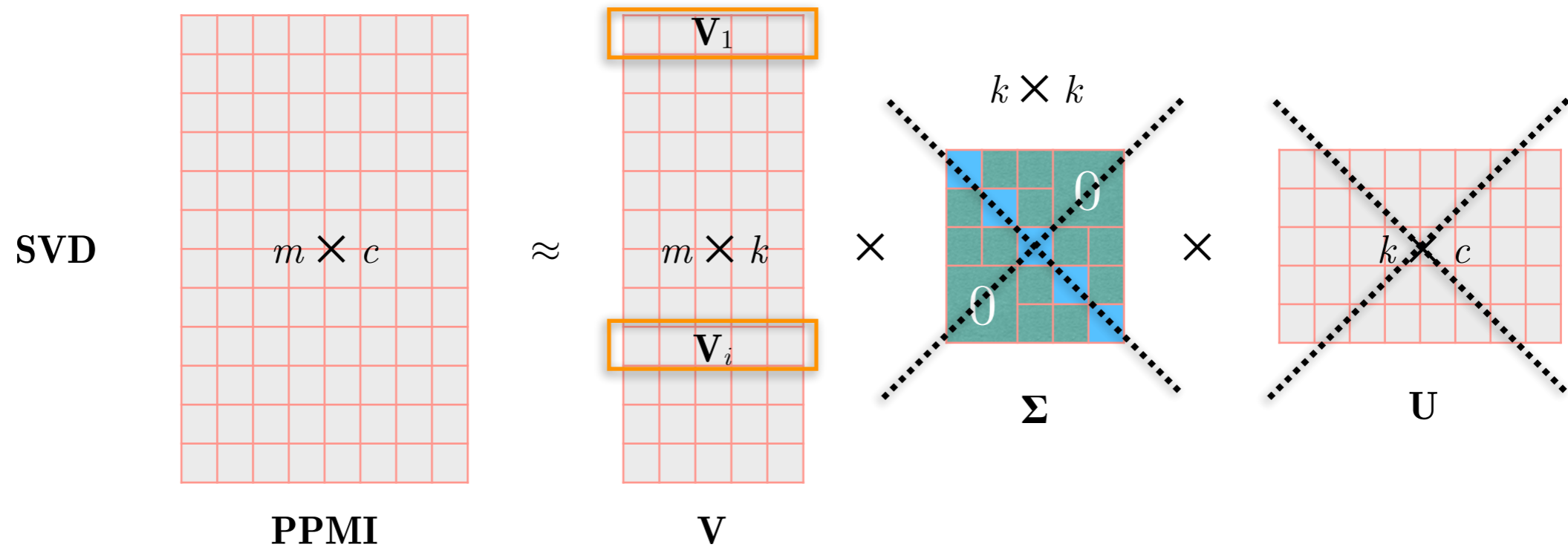
SVD on the PPMI word-context matrix



SVD on the PPMI word-context matrix

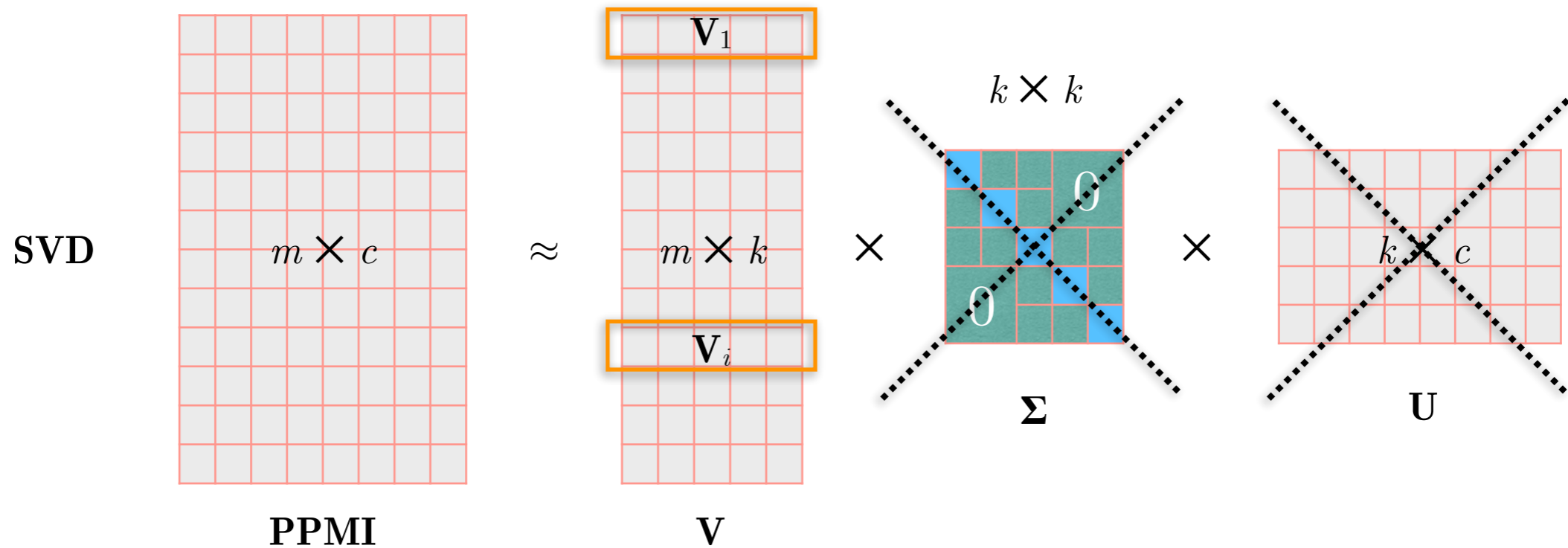


SVD on the PPMI word-context matrix



- V_i is a k -dimensional vector that represents word i in our vocabulary. It is also known as a **word embedding**. Commonly, $k = 300$, i.e. V_i is short and dense.

SVD on the PPMI word-context matrix



- V_i is a k -dimensional vector that represents word i in our vocabulary. It is also known as a **word embedding**. Commonly, $k = 300$, i.e. V_i is short and dense.
- **Downside:** SVD has a significant computational cost $\sim O(mk^2)$

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
- Negative sampling
- **Naïve softmax**

word2vec — skip-gram

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

word2vec — skip-gram

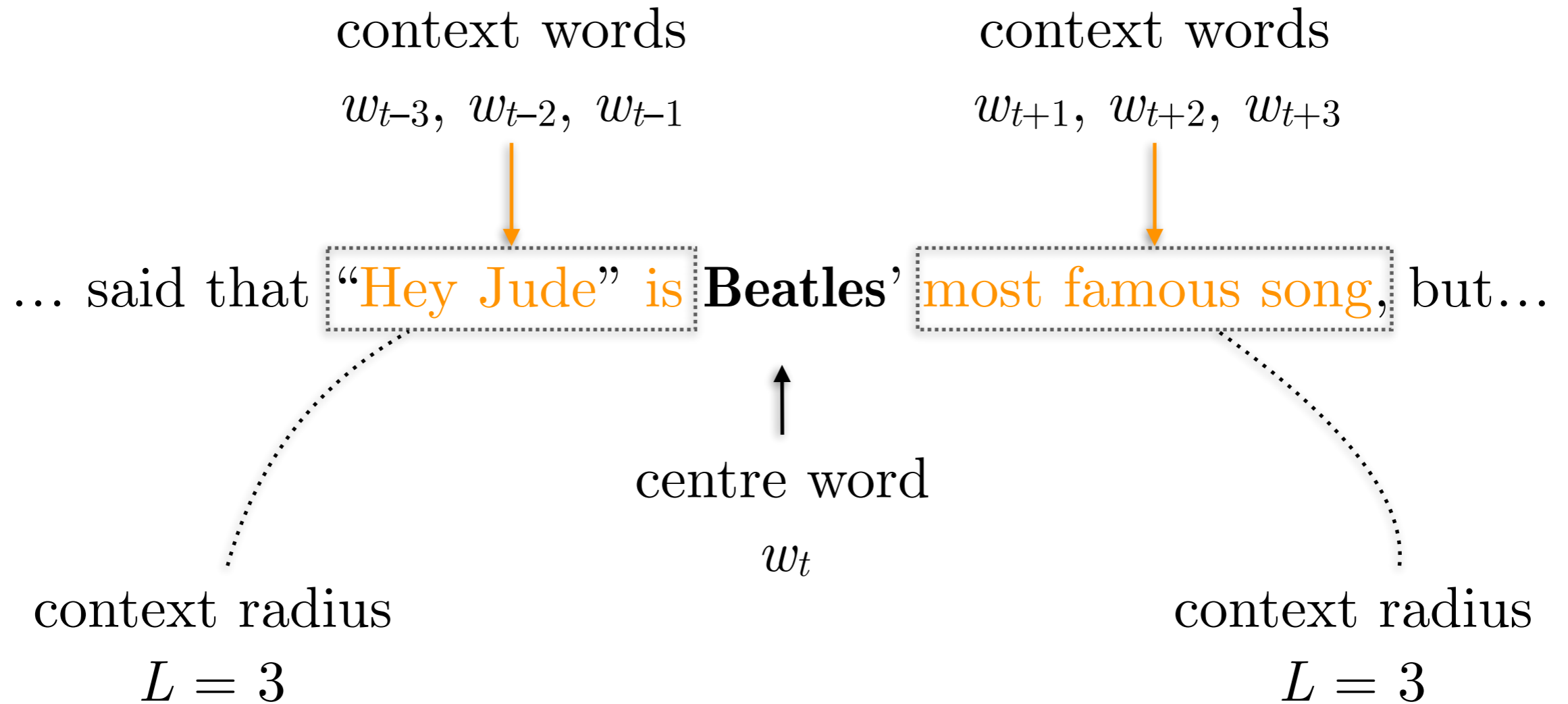
... said that “Hey Jude” is **Beatles**’ most famous song, but...



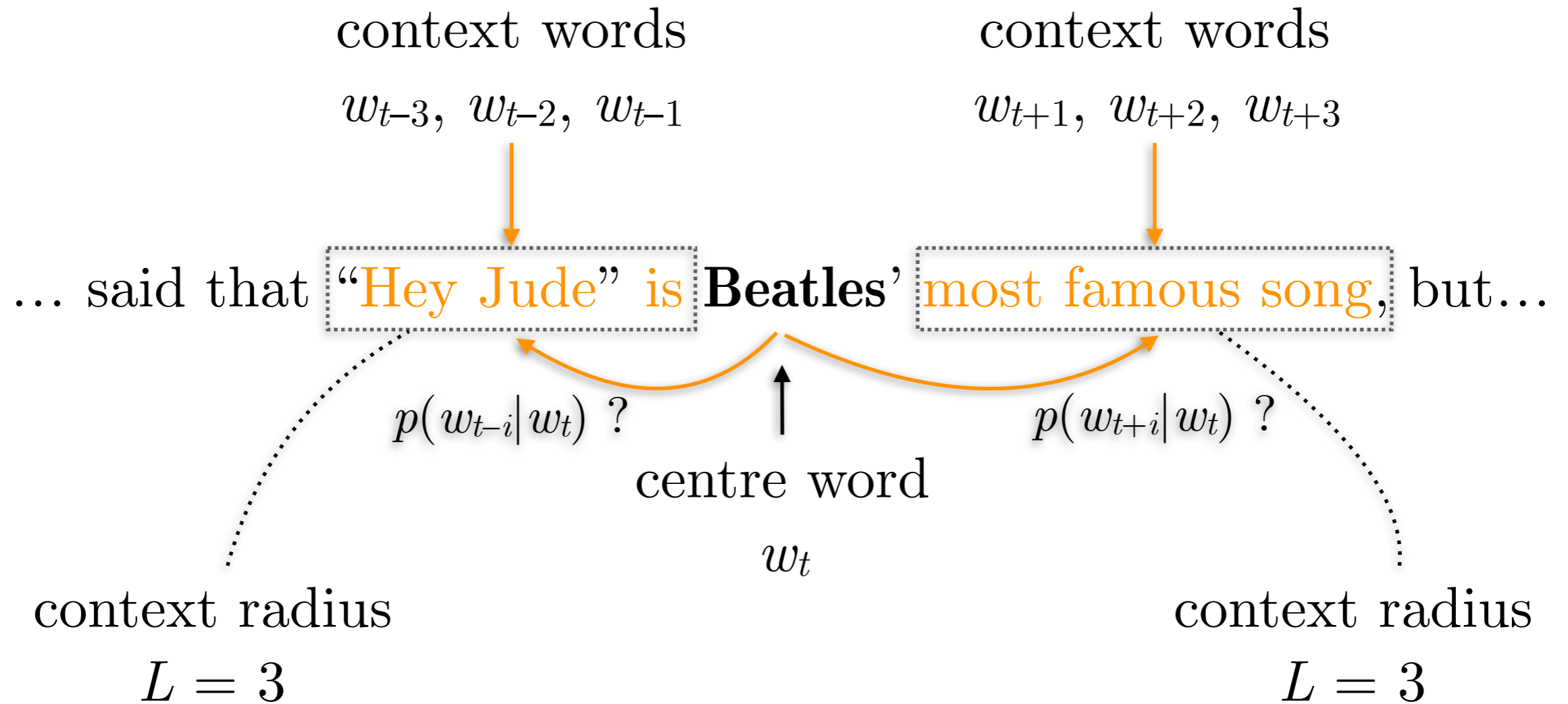
centre word

w_t

word2vec — skip-gram



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)

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

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)

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

Prefer to minimise things, and sums over products

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

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

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

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

How are we going to minimise this?

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

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

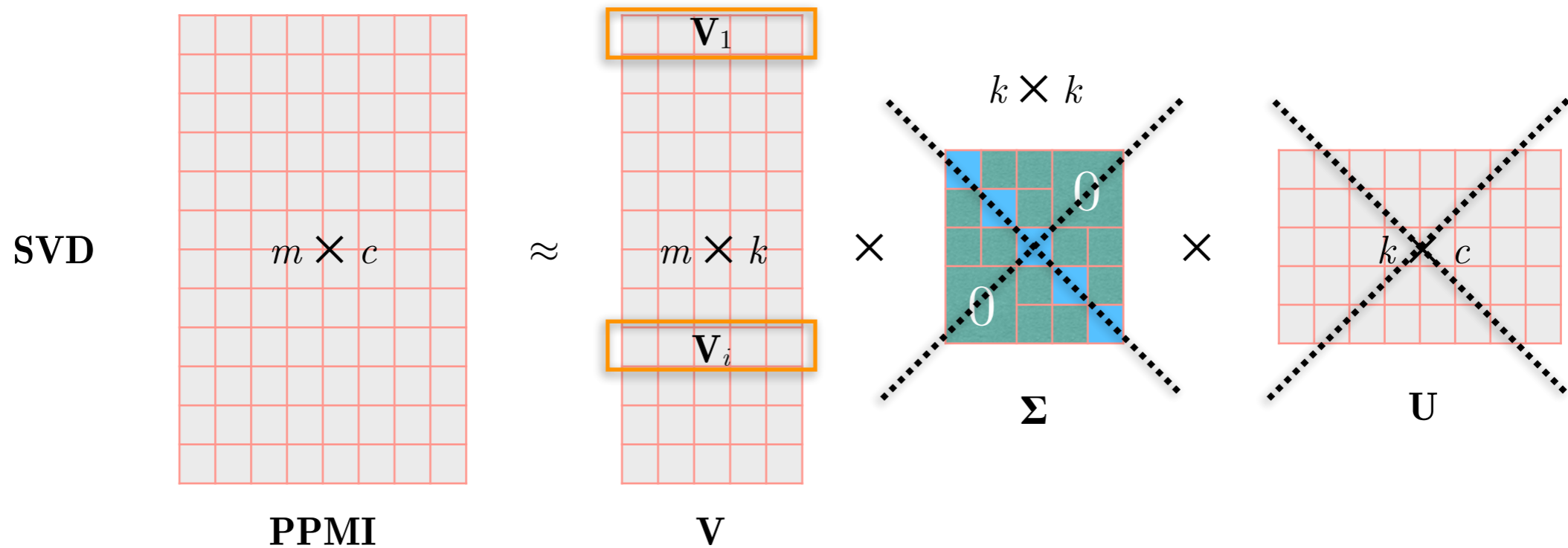
— Assume that each centre word (t) has a k -dimensional vector representation \mathbf{v}_c ; all m words are held in an $k \times m$ matrix \mathbf{V}

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

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

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

SVD on the PPMI word-context matrix



- V_i is a k -dimensional vector that represents word i in our vocabulary. It is also known as a **word embedding**. Commonly, $k = 300$, i.e. 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

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

— 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 held in an $k \times m$ matrix \mathbf{U}

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

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

skip-gram — Simplified objective function

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

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

skip-gram — Simplified objective function

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

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)

skip-gram — Simplified objective function

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

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 context word x (\mathbf{u}_x)

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

skip-gram — In practice...

$w_t = [0 \ 0 \ \dots \ 1 \ \dots \ 0]^T$ 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

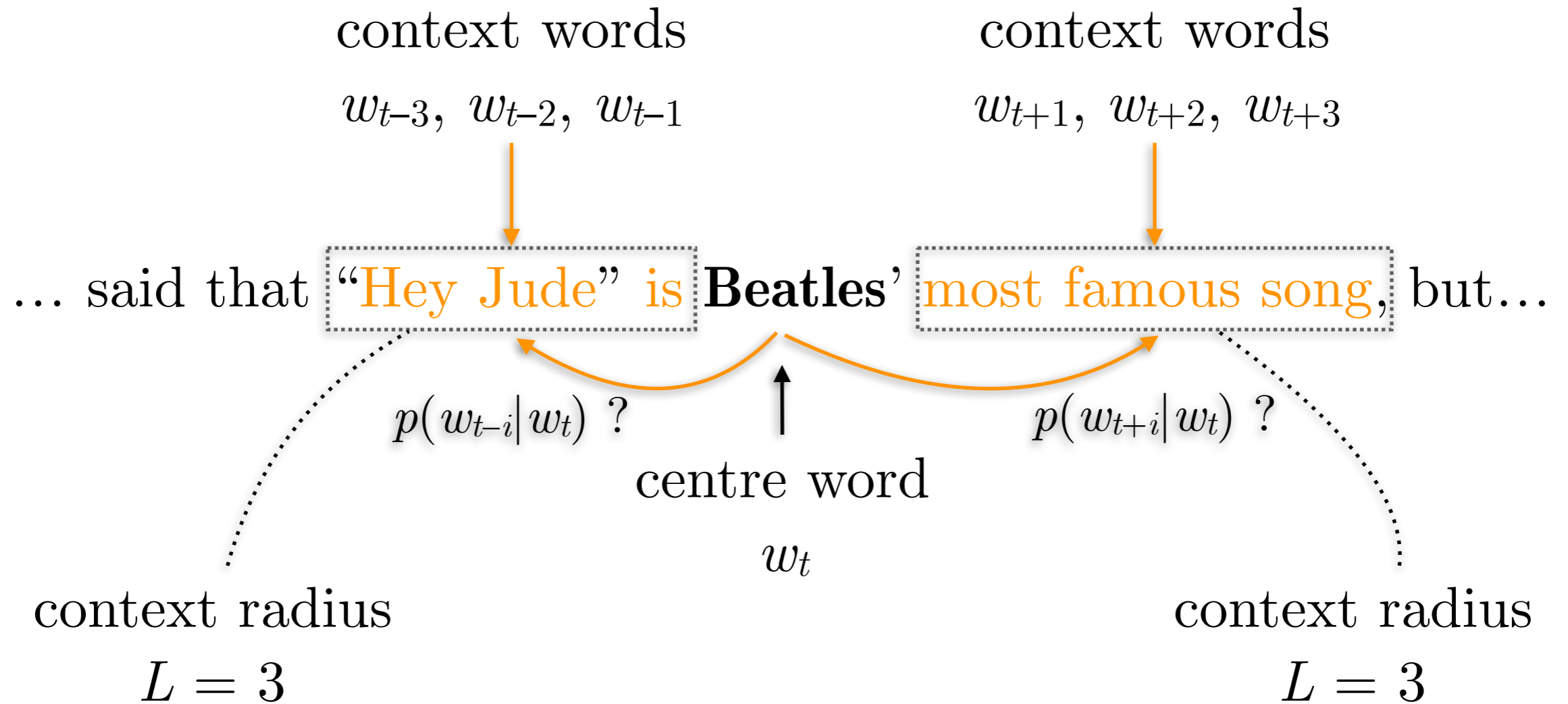
$o = \mathbf{U}^T \cdot v_c$ dot product with all context word vectors
 m (voc. size) \times 1

$p_{w_i} = \text{softmax}(o)_i$ 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



skip-gram — In practice...

$w_t = [0 \ 0 \ \dots \ 1 \ \dots \ 0]^T$ 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

$o = \mathbf{U}^T \cdot v_c$ dot product with all context word vectors
 m (voc. size) \times 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
0.4	0
0.01	0
0.09	0
0.05	0
0.25	1
0.08	0
0.02	0

but we also know the correct answer!
In this case, we need to improve our embeddings (\mathbf{V} and \mathbf{U}).

In neural nets: do error back-propagation.

skip-gram — Negative sampling

Naïve / inefficient way for parameter inference

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

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)

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

skip-gram — Negative sampling

Naïve / inefficient way for parameter inference

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

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)

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

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

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

Gradient descent

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

Too slow and computationally expensive.

skip-gram — Stochastic gradient descent

Naïve / inefficient way for parameter inference

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

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

$$\text{vector}(\textit{king}) - \text{vector}(\textit{man}) + \text{vector}(\textit{woman}) \approx \text{vector}(\textit{queen})$$

Word analogies with word embeddings

$\text{vector}(\text{'king'}) - \text{vector}(\text{'man'}) + \text{vector}(\text{'woman'}) \approx \text{vector}(\text{'queen'})$

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

Word analogies with word embeddings

$$\text{vector}(\textit{king}) - \text{vector}(\textit{man}) + \text{vector}(\textit{woman}) \approx \text{vector}(\textit{queen})$$

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

In this example, if:

$$a = \text{vector}(\textit{king})$$

$$a_p = \text{vector}(\textit{man})$$

$$b_p = \text{vector}(\textit{woman})$$

Then, when we compute the cosine similarity of $(a - a_p + b_p)$ with the embeddings of all the V words in our corpus, we expect $b = \text{vector}(\textit{queen})$ to have the greatest one.

Word analogies with word embeddings

$$\text{vector}(\textit{king}) - \text{vector}(\textit{man}) + \text{vector}(\textit{woman}) \approx \text{vector}(\textit{queen})$$

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

In this example, if:

$$a = \text{vector}(\textit{king})$$

$$a_p = \text{vector}(\textit{man})$$

$$b_p = \text{vector}(\textit{woman})$$

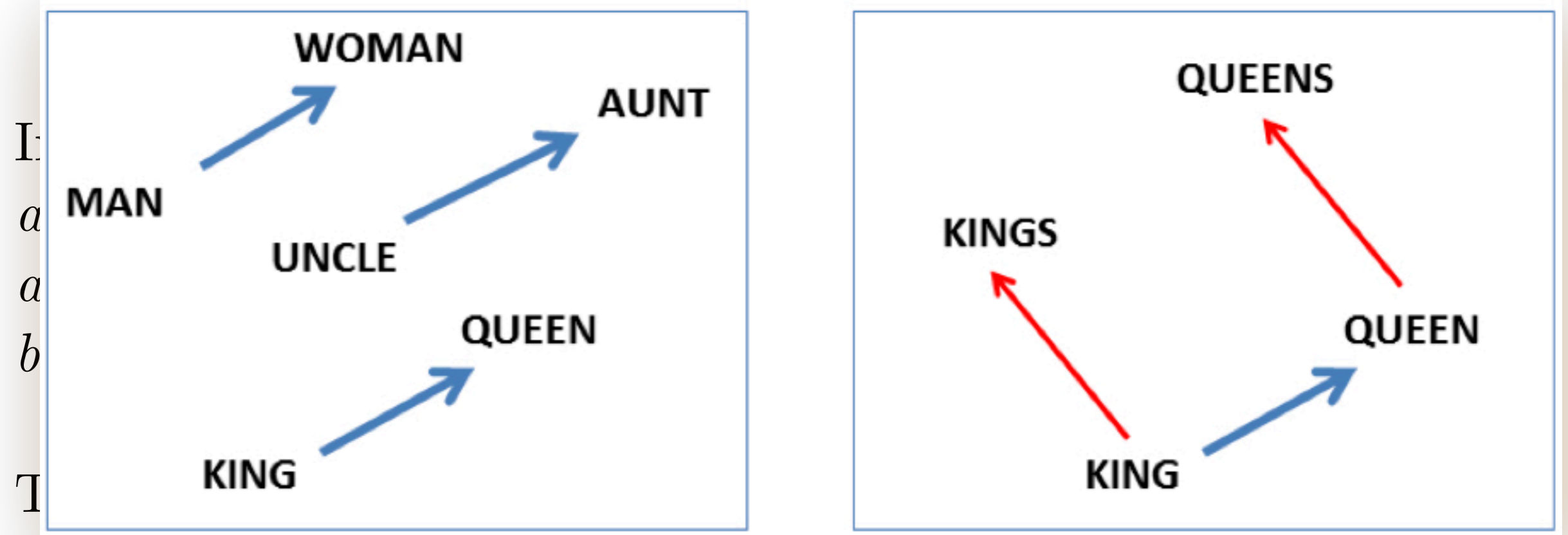
Then, when we compute the cosine similarity of $(a - a_p + b_p)$ with the embeddings of all the V words in our corpus, we expect $b = \text{vector}(\textit{queen})$ to have the greatest one.

This generates the **analogy**: a_p is for a , what b_p is for b
or *man* is for *king*, what *woman* is for *queen*

Word analogies with word embeddings

$\text{vector}('king') - \text{vector}('man') + \text{vector}('woman') \approx \text{vector}('queen')$

More formally: $\arg \max \left(\cos \left(b, a - a_n + b_n \right) \right)$



the embeddings of all the V words in our corpus, we expect $b = \text{vector}('queen')$ to have the greatest one.

This generates the **analogy**: a_p is for a , what b_p is for b
or *man* is for *king*, what *woman* is for *queen*

Word embeddings based on UK Twitter data

word2vec embeddings

- trained on 1.1 billion tweets from 2012 to 2016, geolocated in the UK
- tweets represent current trends, include informal forms of language, and are often topic-consistent
- 512 dimensions, 470,194 words covered
- available online: [figshare.com/articles/UK Twitter word embeddings II /5791650](https://figshare.com/articles/UK_Twitter_word_embeddings_II_/5791650)

Twitter word embeddings — Similarities

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:** espresso, cappuccino, macchiato, latte, coffee
- **linux:** Unix, Centos, Debian, Ubuntu, Redhat
- **retweet:** rt, tweet, retweets, retweeting, rewteet
- **democracy:** democratic, dictatorship, democracies, socialism, undemocratic
- **loool:** loool, lool, looooool, loooooool, loooooool
- **xxxx:** xxxxx, xxx, xxxxxxxxx, xxxxxx, xxxxxxxx
- **enviroment:** environment, environments, env, enviro, habitats

Twitter word embeddings — Analogies

‘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... [?]

Twitter word embeddings — Analogies

‘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... [?]

Twitter word embeddings — Analogies

‘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... [?]

Twitter word embeddings — Analogies

‘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... [?]

Twitter word embeddings — Analogies

‘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... [?]

Twitter word embeddings — Analogies

‘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... [**Grex**it, Syriza, Tsipras]

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

Part III — Applications

N. Aletras, D. Tsarapatsanis, D. Preotiuc-Pietro and V. Lampos.
Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective. PeerJ Computer Science, 2:e93, 2016. [doi:10.7717/peerj-cs.93](https://doi.org/10.7717/peerj-cs.93)



Predicting judicial decisions of the European Court of Human Rights: a Natural Language Processing perspective

Nikolaos Aletras^{1,2}, Dimitrios Tsarapatsanis³, Daniel Preotiuc-Pietro^{4,5} and Vasileios Lampos²

¹ Amazon.com, Cambridge, United Kingdom
² Department of Computer Science, University College London, University of London, London, United Kingdom
³ School of Law, University of Sheffield, Sheffield, United Kingdom
⁴ Positive Psychology Center, University of Pennsylvania, Philadelphia, United States
⁵ Computer & Information Science, University of Pennsylvania, Philadelphia, United States

ABSTRACT

Recent advances in Natural Language Processing and Machine Learning provide us with the tools to build predictive models that can be used to unveil patterns driving judicial decisions. This can be useful, for both lawyers and judges, as an assisting tool to rapidly identify cases and extract patterns which lead to certain decisions. This paper presents the first systematic study on predicting the outcome of cases tried by the European Court of Human Rights based solely on textual content. We formulate a binary classification task where the input of our classifiers is the textual content extracted from a case and the target output is the actual judgment as to whether there has been a violation of an article of the convention of human rights. Textual information is represented using contiguous word sequences, i.e., N-grams, and topics. Our models can predict the court's decisions with a strong accuracy (79% on average). Our empirical analysis indicates that the formal facts of a case are the most important predictive factor. This is consistent with the theory of legal realism suggesting that judicial decision-making is significantly affected by the stimulus of the facts. We also observe that the topical content of a case is another important feature in this classification task and explore this relationship further by conducting a qualitative analysis.

Subjects Artificial Intelligence, Computational Linguistics, Data Mining and Machine Learning, Data Science, Natural Language and Speech
Keywords Natural Language Processing, Text Mining, Legal Science, Machine Learning, Artificial Intelligence, Judicial decisions

INTRODUCTION

In his prescient work on investigating the potential use of information technology in the legal domain, Lawlor surmised that computers would one day become able to analyse and predict the outcomes of judicial decisions (Lawlor, 1963). According to Lawlor, reliable

Submitted 11 May 2016
Accepted 23 September 2016
Published 24 October 2016

Corresponding author
Nikolaos Aletras,
nikos.aletras@gmail.com

Academic editor
Lexing Xie

Additional Information and
Declarations can be found on
page 16

DOI 10.7717/peerj-cs.93

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 $n = \{1, \dots, 4\}$
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

V. Lampos, A. Miller, S. Crossan and C. Stefansen. Advances in nowcasting influenza-like illness rates using search query logs.

Nature Scientific Reports, 5(12760), 2015. [doi:10.1038/srep12760](https://doi.org/10.1038/srep12760)

V. Lampos, B. Zou and I. J. Cox. Enhancing feature selection using word embeddings: The case of flu surveillance.

26th International Conference on World Wide Web, pages 695-704, 2017. [doi:10.1145/3038912.3052622](https://doi.org/10.1145/3038912.3052622)

SCIENTIFIC REPORTS

OPEN

Advances in nowcasting influenza-like illness rates using search query logs

Vasileios Lampos^{1,2}, Andrew C. Miller^{1,3}, Steve Crossan¹ & Christian Stefansen¹

Received: 07 May 2015
Accepted: 06 July 2015
Published: 03 August 2015

User-generated content can assist epidemiological surveillance in the early detection and prevalence estimation of infectious diseases, such as influenza. Google Flu Trends embodies the first public platform for transforming search queries to indications about the current state of flu in various places all over the world. However, the original model significantly mispredicted influenza-like illness rates in the US during the 2012–13 flu season. In this work, we build on the previous modeling attempt, proposing substantial improvements. Firstly, we investigate the performance of a widely used linear regularized regression solver, known as the Elastic Net. Then, we expand on this model by incorporating the queries selected by the Elastic Net into a nonlinear regression framework, based on a composite Gaussian Process. Finally, we augment the query-only predictions with an autoregressive model, injecting prior knowledge about the disease. We assess predictive performance using five consecutive flu seasons spanning from 2008 to 2013 and qualitatively explain certain shortcomings of the previous approach. Our results indicate that a nonlinear query modeling approach delivers the lowest cumulative nowcasting error, and also suggest that query information significantly improves autoregressive inferences, obtaining state-of-the-art performance.

User-generated content published on or submitted to online platforms has been the main source of information in various recent research efforts^{1–3}. It has been shown that large data sets of social media posts and search engine queries contain signals representative of real-life patterns and are therefore indicative of social phenomena in a variety of domains, including politics⁴, finance⁵, commerce⁶, and health^{7,8}. Focusing on health-oriented applications, early research efforts have provided empirical evidence of the informativeness of website⁹ and search engine logs^{10,11} for predicting influenza-like illness (ILI) rates. Google Flu Trends (GFT; <http://www.google.org/flutrends>)¹², in particular, is the first real-time system to apply such methods in practice over a considerable number of countries and a large time span. Similar results were derived through the application of simple^{13,14} or more elaborate^{15–17} natural language processing techniques to content published on the micro-blogging, social networking platform of Twitter.

Data driven estimates can undoubtedly complement current sentinel surveillance systems. One of the original motivations for developing these methods is the intuition that web data could provide timely and less costly information about the prevalence of influenza in a population as opposed to traditional schemes^{18,19}. An important distinction here is that web content can potentially access the bottom and larger part of a disease population pyramid, whereas epidemiological derivations are usually based on the subset of people that actively seek medical attention. Beyond this, places with less established health monitoring systems can greatly benefit from an adaptation of this technology.

Aside from the novelty that GFT introduced, the statistical model behind it has not been tested extensively under practical conditions. Certain works have reported on the mispredictions of GFT through an analysis of its outputs that are published online^{20–22}. Our study proposes and compares several alternative approaches to the original GFT model based on original search query data. We explore three

¹University College London, Department of Computer Science, London, NW1 2FD, UK. ²Google, Flu Trends Team, London, SW1W 9TQ, UK. ³Harvard University, School of Engineering and Applied Sciences, Cambridge, MA 02138, US. Correspondence and requests for materials should be addressed to V.L. (email: v.lampos@ucl.ac.uk)

SCIENTIFIC REPORTS | 5:12760 | DOI: 10.1038/srep12760

Enhancing Feature Selection Using Word Embeddings: The Case of Flu Surveillance

Vasileios Lampos^{*} v.lampos@ucl.ac.uk Bin Zou^{*} bin.zou.14@ucl.ac.uk Ingemar J. Cox^{*,†} i.cox@ucl.ac.uk

^{*} Department of Computer Science, University College London, London WC1E 6BT, United Kingdom
[†] Department of Computer Science, University of Copenhagen, Copenhagen 2200, Denmark

ABSTRACT

Health surveillance systems based on online user-generated content often rely on the identification of textual markers that are related to a target disease. Given the high volume of available data, these systems benefit from an automatic feature selection process. This is accomplished either by applying statistical learning techniques, which do not consider the semantic relationship between the selected features and the inference task, or by developing labour-intensive text classifiers. In this paper, we use neural word embeddings, trained on social media content from Twitter, to determine, in an unsupervised manner, how strongly textual features are semantically linked to an underlying health concept. We then refine conventional feature selection methods by a priori operating on textual variables that are sufficiently close to a target concept. Our experiments focus on the supervised learning problem of estimating influenza-like illness rates from Google search queries. A “flu infection” concept is formulated and used to reduce spurious—and potentially confounding—features that were selected by previously applied approaches. In this way, we also address forms of scepticism regarding the appropriateness of the feature space, alleviating potential cases of overfitting. Ultimately, the proposed hybrid feature selection method creates a more reliable model that, according to our empirical analysis, improves the inference performance (Mean Absolute Error) of linear and nonlinear regressors by 12% and 28.7%, respectively.

Keywords

Computational Health; Syndromic Surveillance; Influenza-Like Illness; User-Generated Content; Search Query Logs; Feature Selection; Word Embeddings; Regularised Regression; Gaussian Processes

1. INTRODUCTION

Online user-generated content (UGC), primarily in the form of social media posts or search query logs, has been the focus of considerable research effort in recent years. It has facilitated methods, interpretations and inferences in various scientific areas, such as Computational Linguistics [19, 47], Behavioural Sciences [28, 55], Computational Social Science [3, 24] and Computational Health [8, 20, 37, 56], among many others.

A common paradigm, evident in many of these works, is the formulation of a supervised learning task based on a textual representation of UGC [10, 53]. This often involves a large number of features, but a moderate number of training samples, encouraging the application of statistical methods that are able to project the data to a lower dimensional space or maintain the most relevant predictors [33, 34, 36, 48]. A valid criticism of such approaches is that some of the selected features may have little or no semantic link to the regression task, increasing the possibility of overfitting, especially in situations where spurious correlations are observed. To alleviate this effect, methods in Natural Language Processing (NLP) have incorporated classification schemes or have routinely used lexical taxonomies, aiming to encourage a relatedness between the input information and the target thematic concept [2, 7, 11, 49]. However, these operations tend to require an extensive human effort, especially in obtaining a sufficient number of labelled outputs, and are limited to a specific task.

In this paper, we take advantage of current developments in statistical NLP and propose a method to address the aforementioned deficiencies. We form general textual concepts by adopting neural word embeddings [44], and then use them in conjunction with conventional feature selection methods to encourage a level of topicality in the selected predictors within a text regression task. This approach can be regarded as an unsupervised classification layer that favours textual features that belong to a target theme of interest. We use this method to improve feature selection for a large-scale, practical, and well-studied text regression task, specifically the inference of influenza-like illness (ILI) rates from time series of search query frequencies [20, 35, 59].

Monitoring disease rates from online activity can complement the existing health surveillance infrastructure, as it provides access to a larger part of the population including

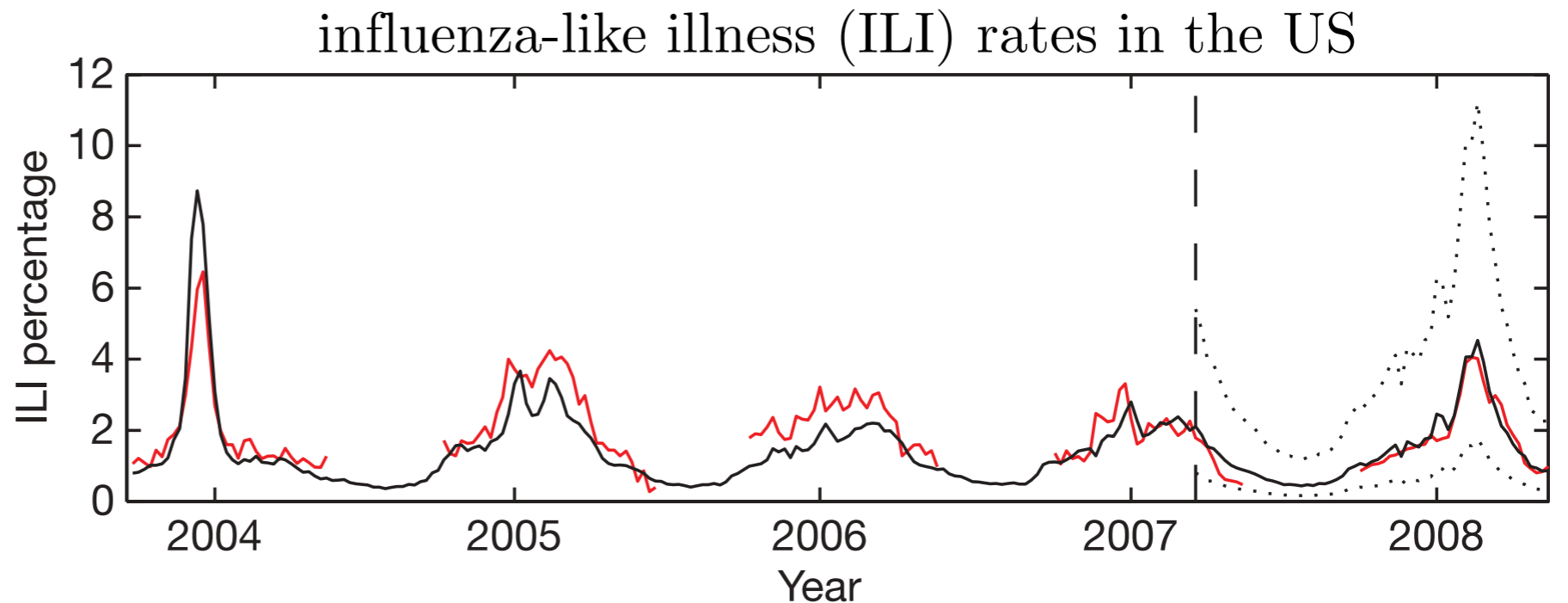
© 2017 International World Wide Web Conference Committee (IW3C2), published under Creative Commons CC BY 4.0 License.
WWW 2017, April 3–7, 2017, Perth, Australia.
ACM 978-1-4503-4913-1/17/04.
DOI: <https://doi.org/10.1145/3038912.3052622>

CC BY

Inferring disease rates from Google search



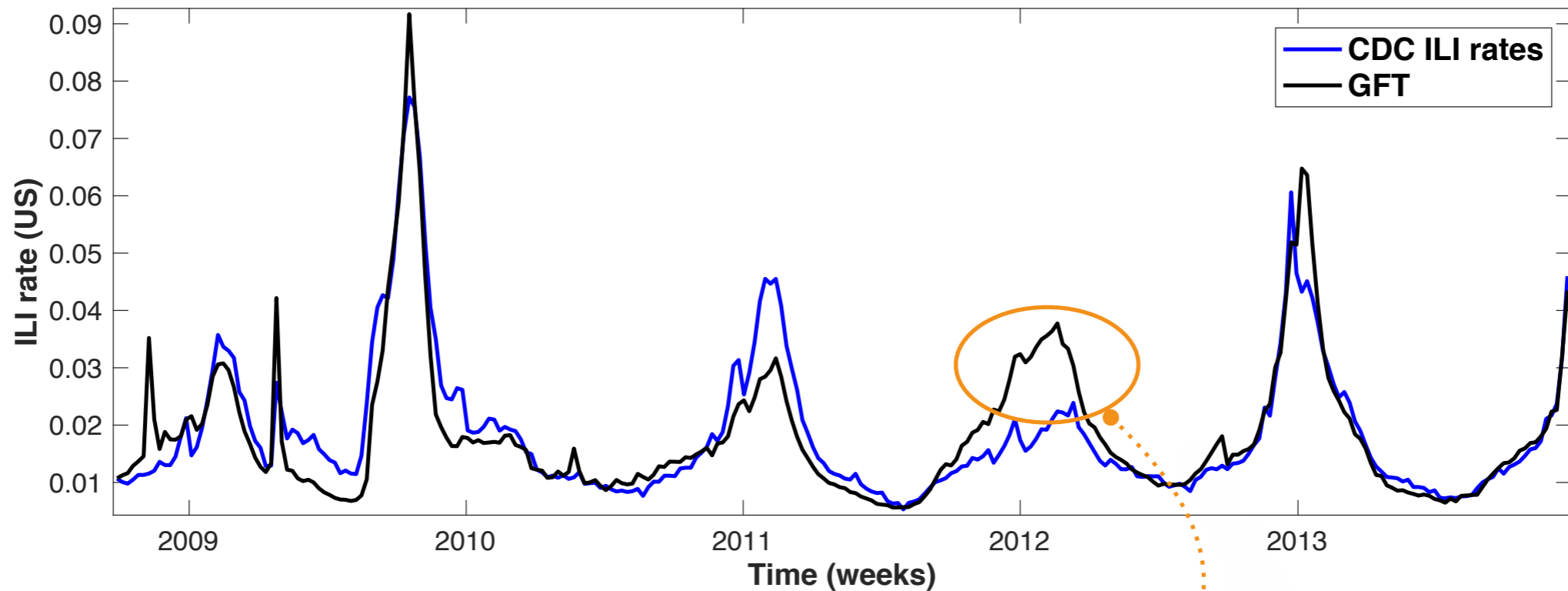
? $f(\text{Google})$



— Centers for Disease Control & Prevention (CDC)
— $f(\text{Google})$

Google proposed an infamous method...

... Google Flu Trends, that made some major mistakes when estimating flu rates in the US, such as



“rsv” — 25%

“flu symptoms” — 18%

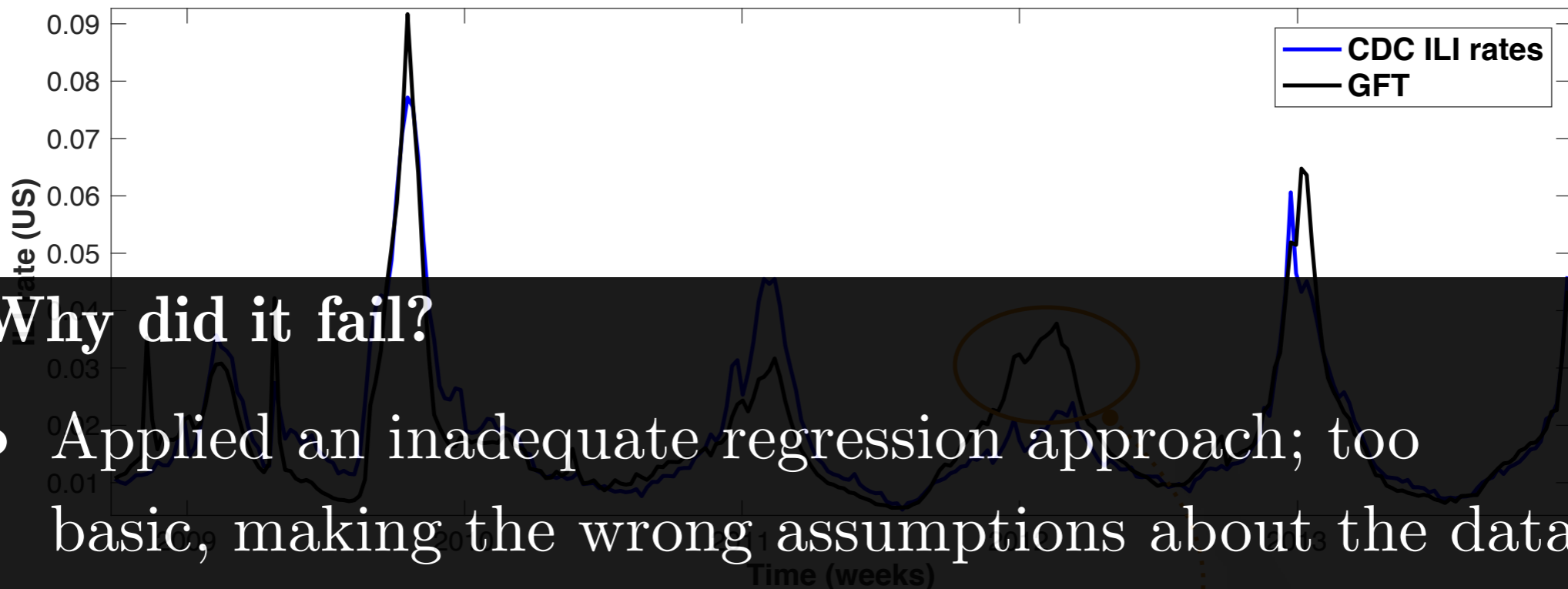
“benzonatate” — 6%

“symptoms of pneumonia” — 6%

“upper respiratory infection” — 4%

Google proposed an infamous method...

... Google Flu Trends, that made some major mistakes when estimating flu rates in the US, 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** by applying word2vec on Twitter data
2. **Search 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

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

query embedding

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

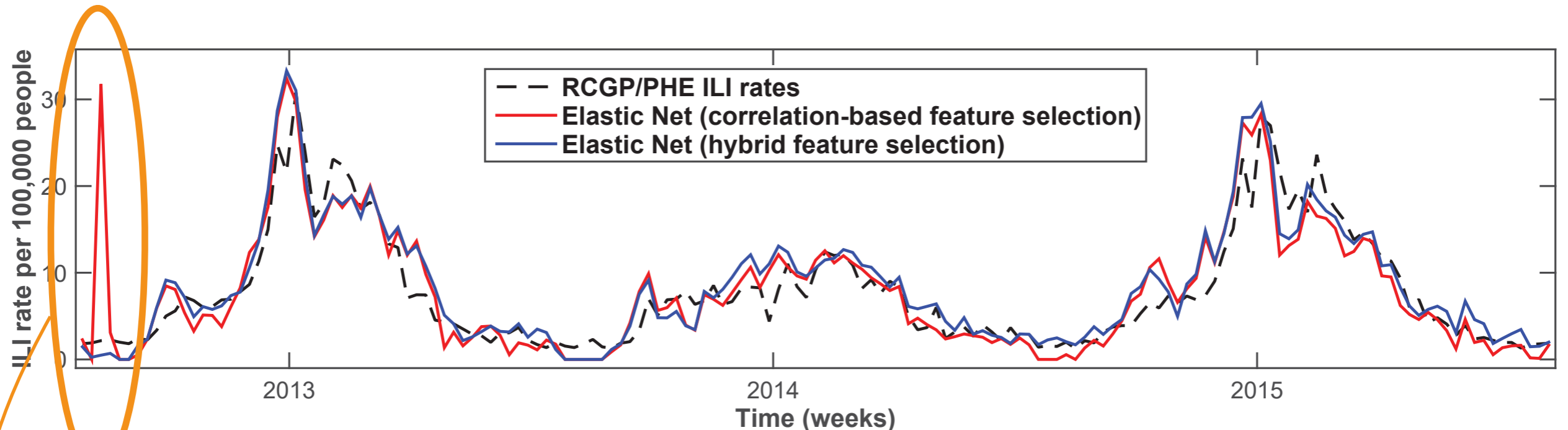
If we combine it with the previous ways for selecting features and state-of-the-art regression approaches, will we obtain **better inference accuracy**?

We test **7 feature selection approaches**:

- concept ranking (CR) \rightarrow elastic net (**1**)
- correlation \rightarrow elastic net (**2**) \rightarrow Gaussian Process (GP) (**3**)
- CR \rightarrow correlation \rightarrow elastic net (**4**) \rightarrow GP (**5**)
- CR \rightarrow correlation \rightarrow GP (**6**)
- correlation \rightarrow 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%)

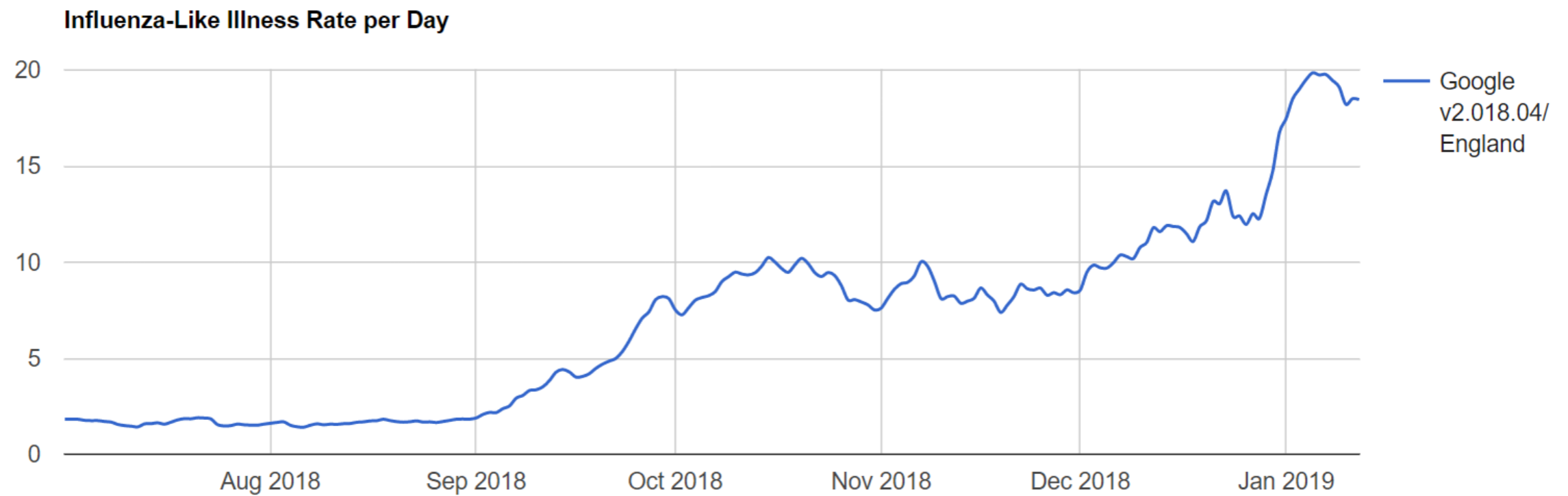
i-sense flu



Home

About

Docs



fludetector.cs.ucl.ac.uk

D. Preotiuc-Pietro, V. Lampos and N. Aletras. **An analysis of the user occupational class through Twitter content.** 53rd Annual Meeting of the Association for Computational Linguistics, pages 1754-1764, 2015. aclweb.org/anthology/P15-1169

An analysis of the user occupational class through Twitter content

Daniel Preotiuc-Pietro¹, Vasileios Lampos² and Nikolaos Aletras²

¹ Computer & Information Science, University of Pennsylvania

² Department of Computer Science, University College London

danielpr@sas.upenn.edu, {v.lampos,n.aletras}@ucl.ac.uk

Abstract

Social media content can be used as a complementary source to the traditional methods for extracting and studying collective social attributes. This study focuses on the prediction of the occupational class for a public user profile. Our analysis is conducted on a new annotated corpus of Twitter users, their respective job titles, posted textual content and platform-related attributes. We frame our task as classification using latent feature representations such as word clusters and embeddings. The employed linear and, especially, non-linear methods can predict a user's occupational class with strong accuracy for the coarsest level of a standard occupation taxonomy which includes nine classes. Combined with a qualitative assessment, the derived results confirm the feasibility of our approach in inferring a new user attribute that can be embedded in a multitude of downstream applications.

1 Introduction

The growth of online social networks provides the opportunity to analyse user text in a broader context (Tumasjan et al., 2010; Bollen et al., 2011; Lampos and Cristianini, 2012). This includes the social network (Sadilek et al., 2012), spatio-temporal information (Lampos and Cristianini, 2010) and personal attributes (Al Zamal et al., 2012). Previous research has analysed language differences in user attributes like location (Cheng et al., 2010), gender (Burger et al., 2011), impact (Lampos et al., 2014) and age (Rao et al., 2010), showing that language use is influenced by them. Therefore, user text allows us to infer these properties. This *user profiling* is important not only for sociolinguistic studies, but also for other applications: recommender systems

to provide targeted advertising, analysts who study different opinions in each social class or integration in text regression tasks such as voting intention (Lampos et al., 2013).

Social status reflected through a person's occupation is a factor which influences language use (Bernstein, 1960; Bernstein, 2003; Labov, 2006). Therefore, our hypothesis is that language use in social media can be indicative of a user's occupational class. For example, executives may write more frequently about business or financial news, while people in manufacturing positions could refer more to their personal interests and less to job related activities. Similarly, we expect some categories of people, like those working in sales and customer services, to be more social or to use more informal language.

Focusing on the microblogging platform of Twitter, we explore our hypothesis by studying the task of predicting a user's occupational class given platform-related attributes and generated content, i.e. tweets. That has direct applicability in a broad range of areas from sociological studies, which analyse the behaviour of different occupations, to recruiting companies that target people for new job opportunities. For this study, we created a publicly available data set of users, including their profile information and historical text content as well as a label to an occupational class from the "Standard Occupational Classification" taxonomy (see Section 2).

We frame our task as classification, aiming to identify the most likely job class for a given user based on profile and a variety of textual features: general word embeddings and clusters (or 'topics'). Both linear and non-linear classification methods are applied with a focus on those that can assist interpretation and offer qualitative insights. We find that text features, especially word clusters, lead to good predictive performance. Accuracy for our best model is well above 50% for 9-way classifi-

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, pediatrician
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)

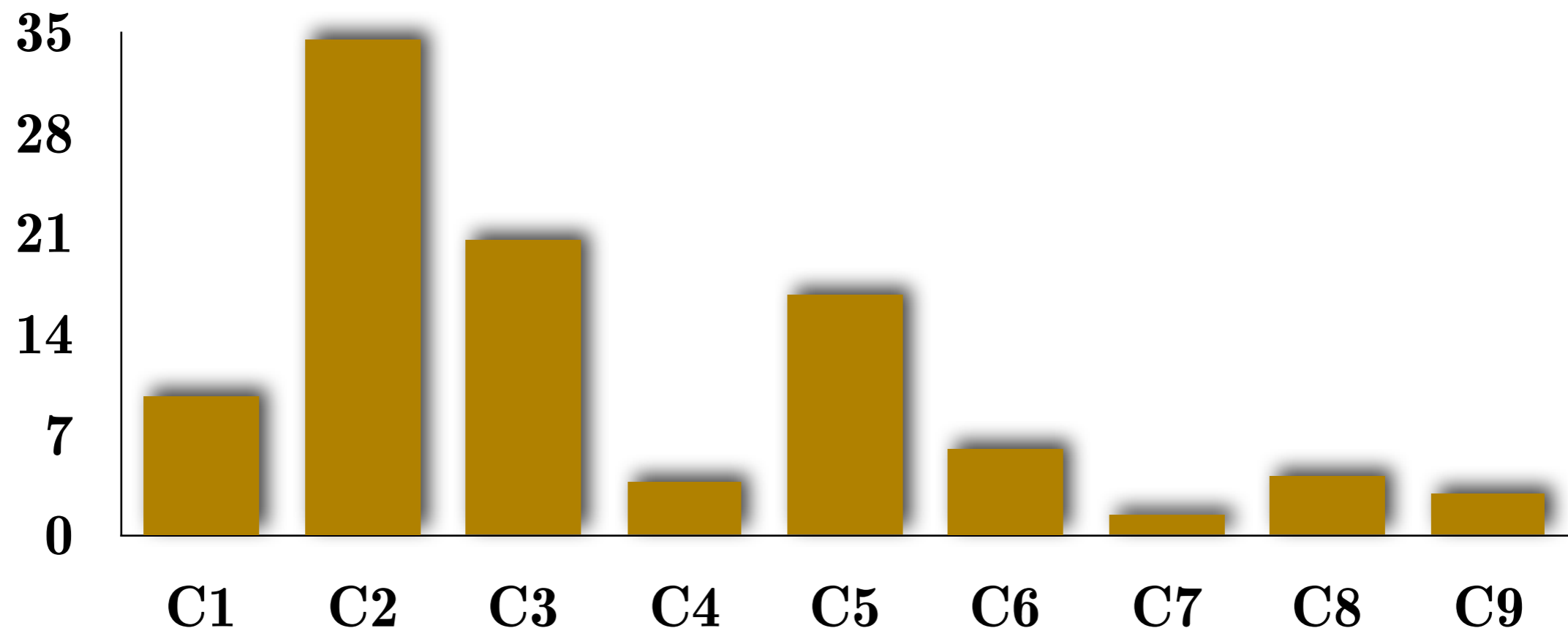
The 9 major occupational classes (C1-9)

- **C1:** Managers, Directors, Senior Officials (**CEO, bank manager**)
- **C2:** Professional Occupations (**postdoc, pediatricist**)
- **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

% of users per SOC category



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

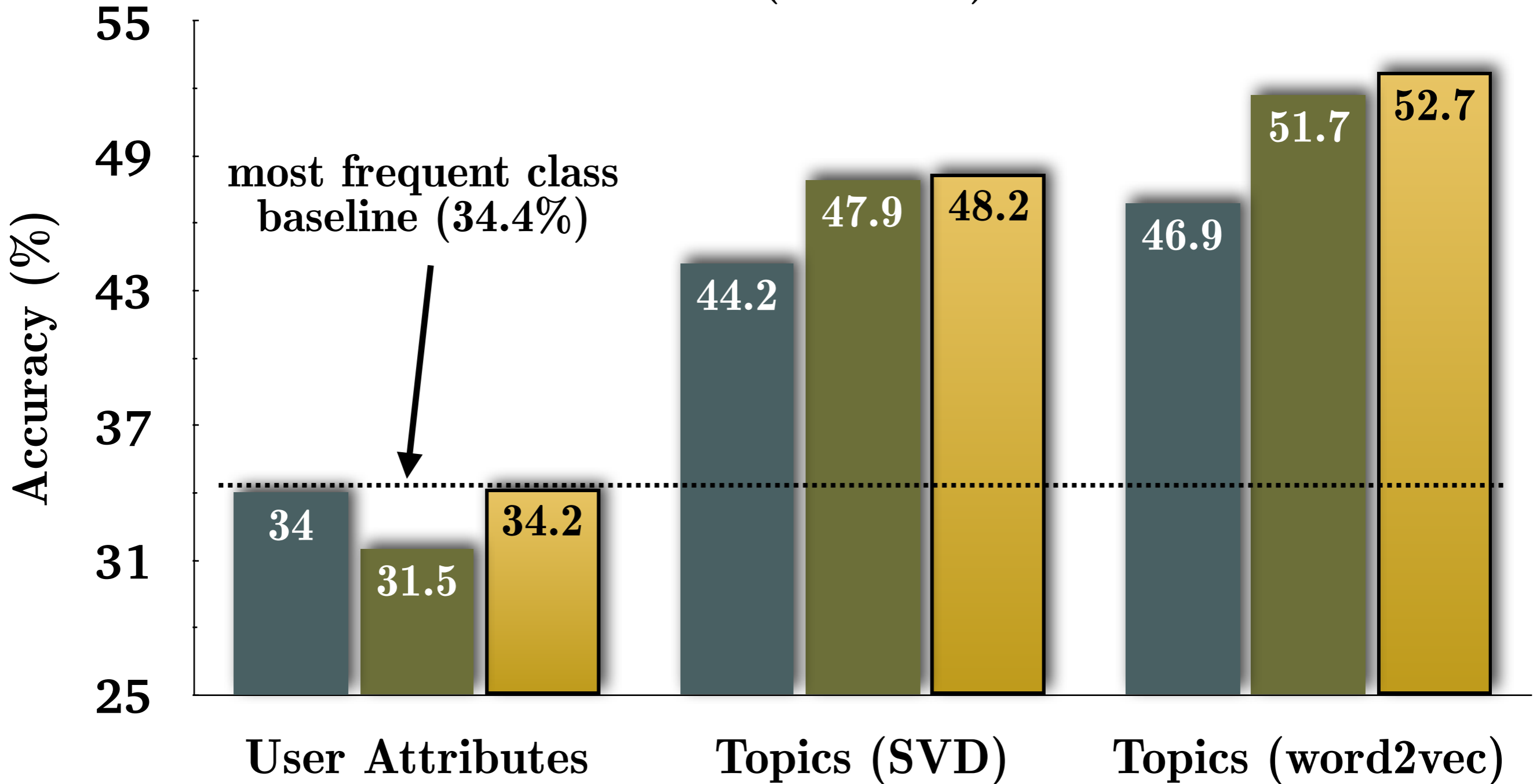
- retweets done
- non duplicate tweets
- retweeted tweets
- hashtags
- tweets with hashtags
- tweets with @-mentions
- @-replies
- tweets with links
- tweets in English

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

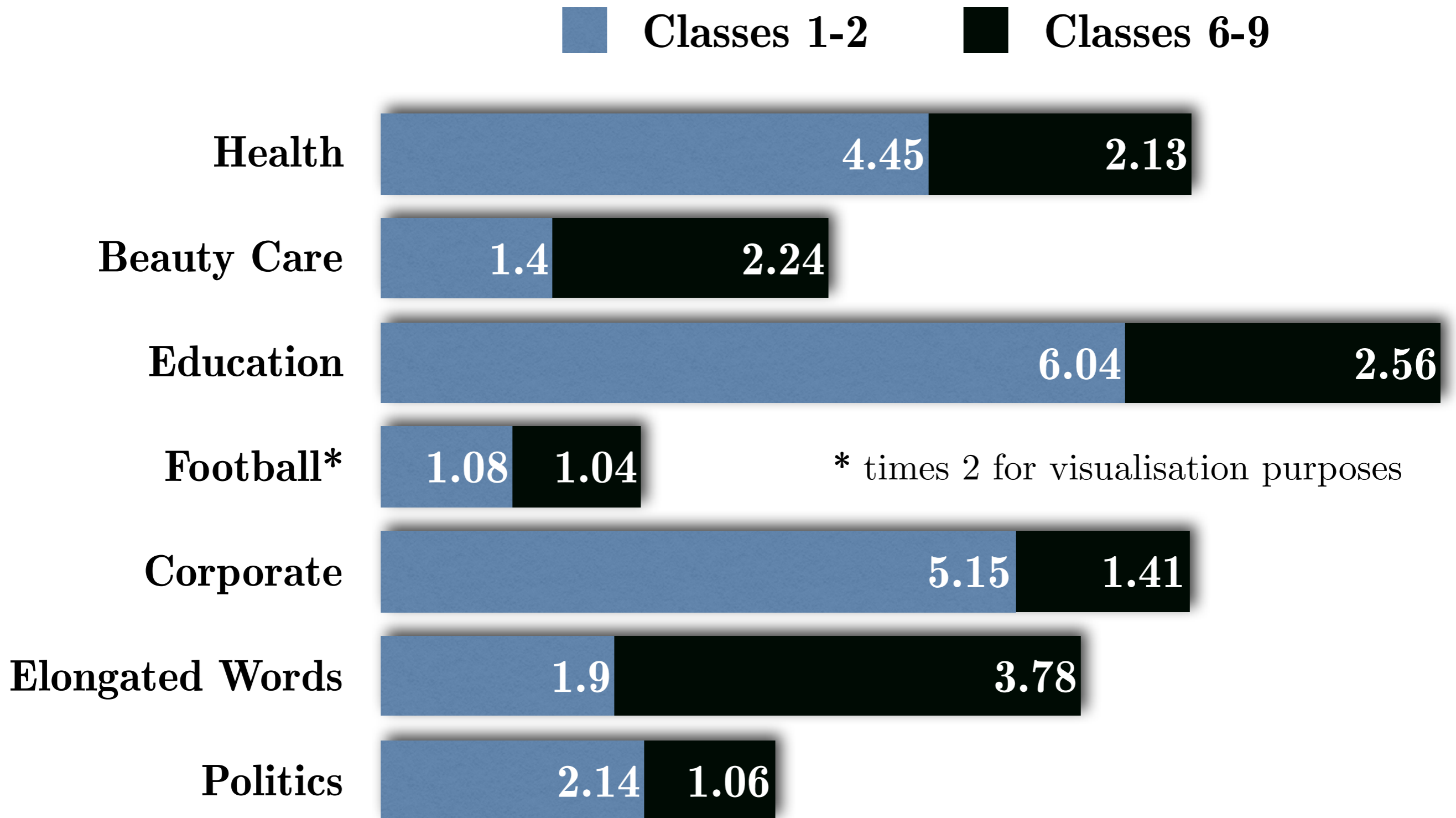
Job (9-class) classification accuracy



Most predictive topics (word2vec)

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

Higher vs. lower skilled occupations and topics



Topic scores for occupational class supersets

Slides (with potential **revisions**)

lampos.net/slides/irdm2019.pdf

end_of_lecture

@lampos



lampos.net

