POLITÉCNICA
"Ingeniamos el futuro"

CAMPUS
DE EXCELENCIA
INTERNACIONAL

Text Mining 2
Unsupervised Methods

Madrid Summer School on
Advanced Statistics and Data Mining

Florian Leitner
Data Catalytics, S.L.
leitner@datacatytics.com

Sentence segmentation

- Sentences are **the** fundamental linguistic unit
  - Sentences are the boundaries or "constraints" for linguistic phenomena.
  - **Collocations** ["United Kingdom", "vice president"], idioms ["drop me a line"], phrases [e.g., the preposition phrase "of great fame"], clauses, statements, … all occur **within** a sentence.
- Rule/pattern-based segmentation
  - Segment sentences if the marker is followed by an upper-case letter
  - Works well for "clean text" (news articles, books, papers, …)
  - **Special cases**: abbreviations, digits, lower-case proper nouns (genes, "amnesty international", …), hyphens, quotation marks, …
- Supervised sentence boundary detection
  - Use some Markov model or a conditional random field to identify possible sentence segmentation tokens
  - Requires labeled examples (segmented sentences)

Punkt Sentence Tokenizer (PST) 1/2

- Unsupervised sentence boundary detection
- $P(\bullet|\mathbf{w}_{-1}) > \mathbf{c}_{cpc}$ — Dr.
  - Determines if a marker • is used as an **abbreviation** marker by comparing the **conditional probability** that the word $\mathbf{w}_{-1}$ before • is followed by the marker against some (high) cutoff probability.
    - $P(\bullet|\mathbf{w}_{-1}) = P(\mathbf{w}_{-1}, \bullet) \div P(\mathbf{w}_{-1})$
    - K&S set $\mathbf{c} = 0.99$
- $P(\mathbf{w}_{+1}|\mathbf{w}_{-1}) > P(\mathbf{w}_{+1})$ — Mrs. Watson
  - Evaluates the likelihood that $\mathbf{w}_{-1}$ and $\mathbf{w}_{+1}$ surrounding the marker • are more commonly collocated than would be expected by chance: • is assumed an **abbreviation** marker ("not independent") if the LHS is greater than the RHS.
- $F_{length}(\mathbf{w}) \times F_{markers}(\mathbf{w}) \times F_{penalty}(\mathbf{w}) \geq \mathbf{c}_{abbr}$ — U.S.A.
  - Evaluates if any of $\mathbf{w}$'s morphology (length of $\mathbf{w}$ w/o marker characters, number of periods inside $\mathbf{w}$ (e.g., ["U.S.A"]), penalized when $\mathbf{w}$ is not followed by a •) makes it more likely that $\mathbf{w}$ is an abbreviation against some (low) cutoff.
- $F_{ortho}(\mathbf{w})$; $P_{sstarter}(\mathbf{w}_{+1}|\bullet)$; … — . Therefore
  - Orthography: lower-, upper-case or capitalized word after a probable • or not
  - Sentence Starter: Probability that $\mathbf{w}$ is found after a •

Punkt Sentence Tokenizer (PST) 2/2

- Unsupervised Multilingual Sentence Boundary Detection
  - Kiss & Strunk, MIT Press 2006.
  - Available from NLTK: `nltk.tokenize.punkt` (http://www.nltk.org/api/nltk.tokenize.html)
- PST is language agnostic
  - Requires that the language uses the sentence segmentation marker as an abbreviation marker
  - Otherwise, the problem PST solves is not present
- PST factors in word length
  - Abbreviations are relatively shorter than regular words
- PST takes "internal" markers into account
  - E.g., "U.S.A"
- Main weakness: long lists of abbreviations
  - E.g., author lists in citations
  - Can be fixed with a pattern-based post-processing strategy
- NB: a marker must be present
  - E.g., chats or fora

From syntactic to semantic similarity

Cosine Similarity, χ^2, Spearman's ρ, LSH, etc. all compare equal tokens.

But what if you are talking about "automobiles" and I am lazy, calling it a "car"?

We can solve this with Latent Semantic Indexing!

Latent Semantic Analysis (LSI 1/3)

- a.k.a. Latent Semantic **Indexing** (in Text Mining): **feature extraction** for **semantic inference**

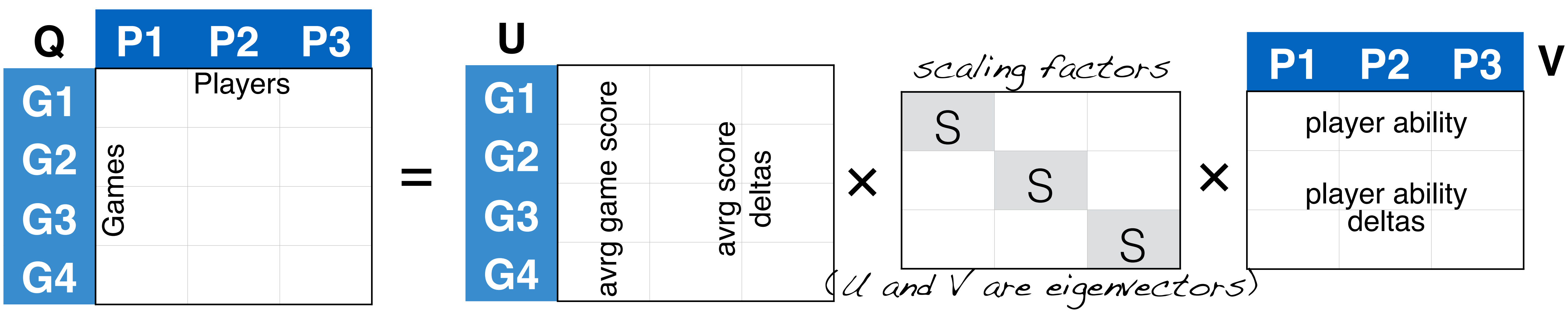
- Linear algebra background

orthonormal factors of Q (QQ^T and Q^TQ)

  - Singular value decomposition of a matrix Q: $\mathbf{Q} = \mathbf{U\Sigma V^T}$

singular values: scaling factor

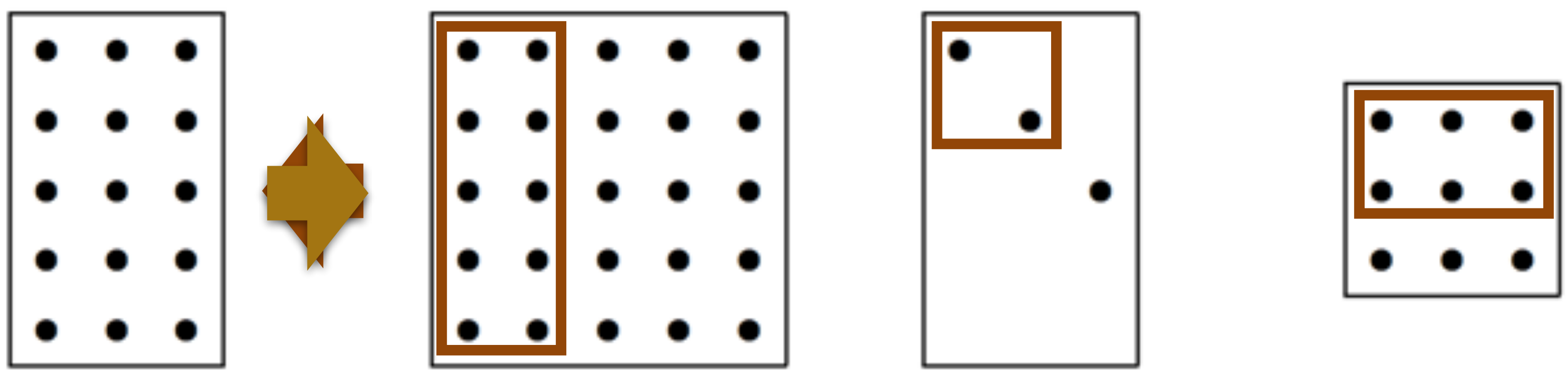
the factors "predict" Q in terms of similarity (Frobenius norm) using as many factors as the lower dimension of Q



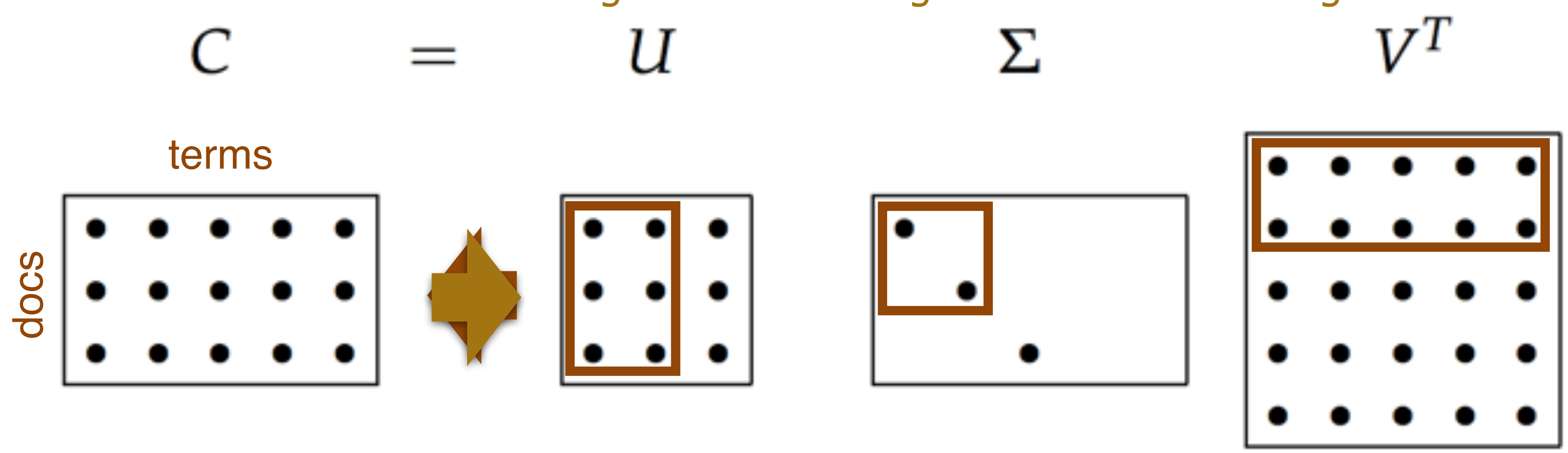
- SVD in text mining
  - Inverted index = doc. eigenvectors × singular values × term eigenvectors

Latent Semantic Analysis (LSI 2/3)

C = ˆFeat. extraction by selecting only the largest n eigenvalues



- Inverted index = doc. eigenvectors × singular values × term eigenvectors



- Image taken from: Manning et al. An Introduction to IR. 2009

Latent Semantic Analysis (LSI 3/3)

[Spearman's] rho(human, user) = -0.38
rho(human, minors) = -0.29

c1: *Human* machine *interface* for ABC *computer* applications
c2: A *survey* of *user* opinion of *computer system response time*
c3: The *EPS user interface* management *system*
c4: *System* and *human system* engineering testing of *EPS*
c5: Relation of *user* perceived *response time* to error measurement

m1: The generation of random, binary, ordered *trees*
m2: The intersection *graph* of paths in *trees*
m3: *Graph minors* IV: Widths of *trees* and well-quasi-ordering
m4: *Graph minors*: A *survey*

C

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

top 2 dim

Ĉ

	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	0.16	0.40	0.38	0.47	0.18	-0.05	-0.12	-0.16	-0.09
interface	0.14	0.37	0.33	0.40	0.16	-0.03	-0.07	-0.10	-0.04
computer	0.15	0.51	0.36	0.41	0.24	0.02	0.06	0.09	0.12
user	0.26	0.84	0.61	0.70	0.39	0.03	0.08	0.12	0.19
system	0.45	1.23	1.05	1.27	0.56	-0.07	-0.15	-0.21	-0.05
response	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
time	0.16	0.58	0.38	0.42	0.28	0.06	0.13	0.19	0.22
EPS	0.22	0.55	0.51	0.63	0.24	-0.07	-0.14	-0.20	-0.11
survey	0.10	0.53	0.23	0.21	0.27	0.14	0.31	0.44	0.42
trees	-0.06	0.23	-0.14	-0.27	0.14	0.24	0.55	0.77	0.66
graph	-0.06	0.34	-0.15	-0.30	0.20	0.31	0.69	0.98	0.85
minors	-0.04	0.25	-0.10	-0.21	0.15	0.22	0.50	0.71	0.62

test # dim to use via synonyms or missing words

From: Landauer et al. An Introduction to LSA. 1998

rho(human, user) = **0.94**
rho(human, minors) = **-0.83**

Principal Component vs. Latent Semantic Analysis

best Frobenius norm: minimize “std. dev.” of matrix
best affine subspace: minimize dimensions while maintaing the form

- **LSA** seeks for the **best linear subspace** in **Frobenius norm**, while **PCA** aims for the **best affine linear subspace**.
- **LSA** (**can**) **use** TF-IDF weighting as **preprocessing** step.
- **PCA requires the** (square) **covariance matrix** of the original matrix as its first step and therefore can only compute term-term or doc-doc similarities.
- **PCA matrices are more dense** (zeros occur only when true independence is detected).

Text Summarization

Russian defense minister Ivanov called on Sunday for the creation of a global front for combating terrorism.

- Extractive summarization
  - Select the most informative sentences.
  - Order sentences (or leave in order).

Ivanov called for a global front combating terrorism.

- Abstractive summarization
  - Generate new text given the input document.
  - Very unique but still rather experimental.

Russia calls for a joint effort against terrorism.

Extractive Summarization with TextRank

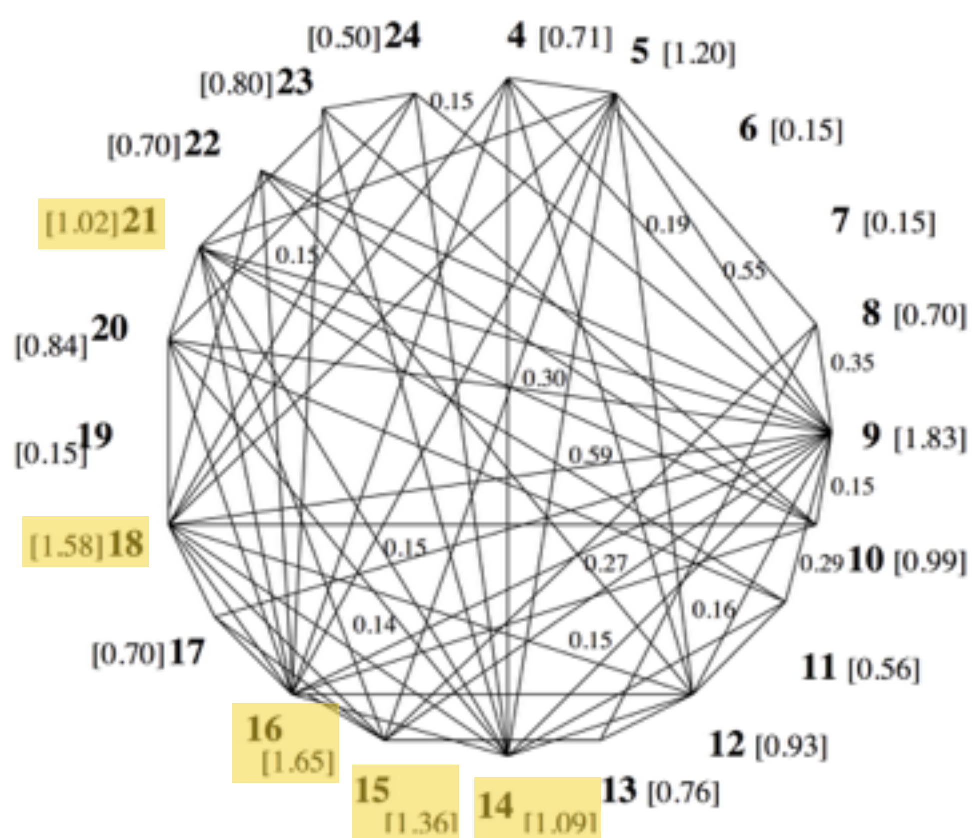
tokens, n-grams or whole sentences

1. Collect all text shingles (→ graph vertices) from the input document[s].

2. Quantify relation strength (→ edges) between those shingles from their context (**co-occurrence**) or content (**TF-IDF**).

3. Iterate a graph ranking algorithm (**PageRank**) to convergence.

4. Sort the vertices on their final score to identify the most informative shingles.



Mihalcea, R., and Tarau, P. (2004). TextRank: Bringing order into texts.

TextRank Summarization with Okapi-BM25 Ranking

2. Quantify relation strength (→ edges) between those shingles from their content.

"classical" TF-IDF

$$TFIDF(D_n, Q) = \sum_{i}^{|Q|} TF(q_i, D_n) \times IDF(q_i)$$

$$TF(q_i, D_n) = log(|q_i \in D_n|)$$

Okapi BM25 "TF modification"

$$BM25(D_n, Q) = \sum_{i}^{|Q|} Okapi(q_i, D_n) \times IDF(q_i)$$

$$Okapi(q_i, D_n) = \frac{TF(q_i, D_n)(k+1)}{TF(q_i, D_n) + k(1 - b + b\frac{|D_n|}{mean(|D|)})}$$

Main difference: the Okapi function flattens out much faster than a log-scaled Term Frequency function (alone).

https://en.wikipedia.org/wiki/Okapi_BM25

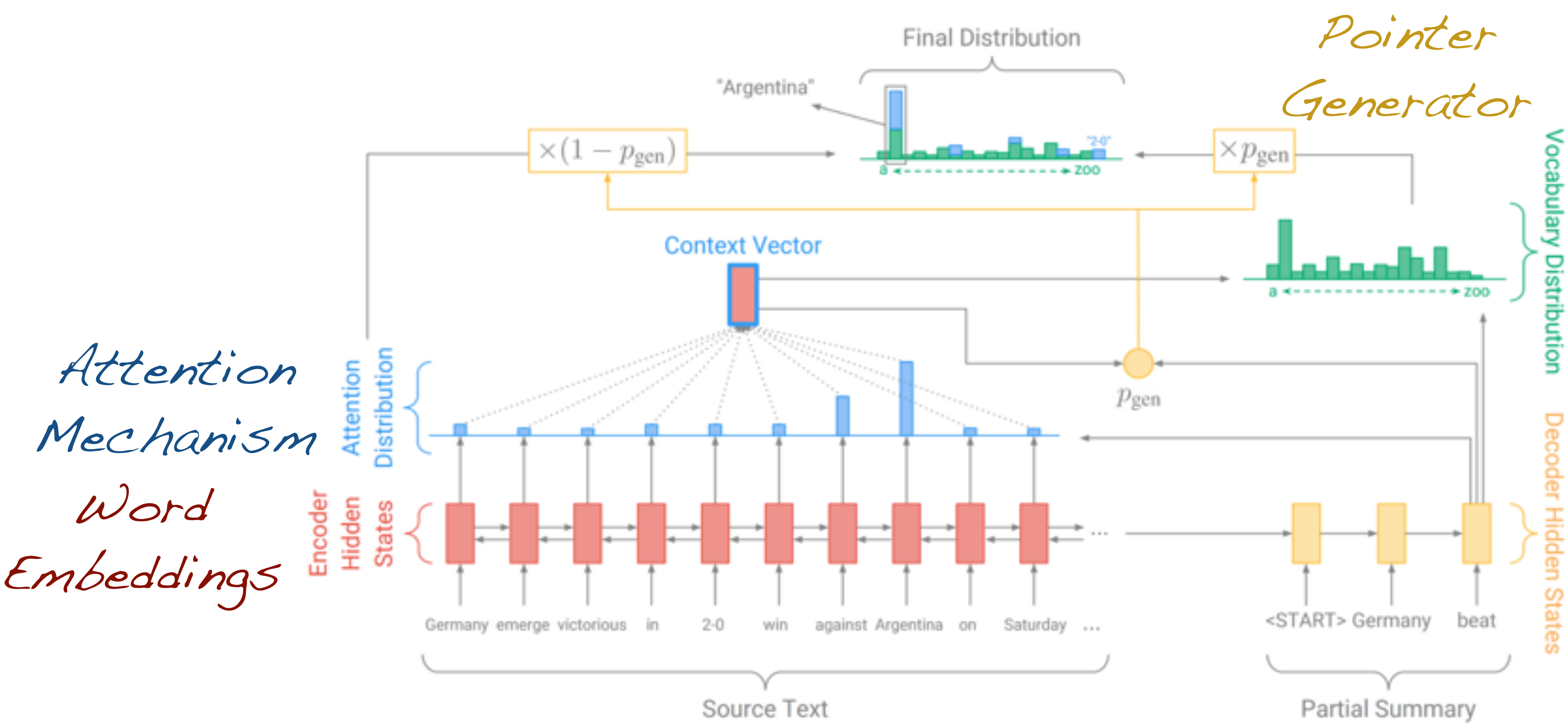
Barrios, F., López, F., Argerich, L., and Wachenchauzer, R. (2016). Variations of the similarity function of TextRank for automated summarization.

LexRank vs. TextRank

- Published simultaneously in 2004 by two independent groups
- Both are based on the same idea (graph similarity ranking)
- **LexRank** is part of a larger supervised summarization system ("MEAD") that uses features like sentence position and length.
- **LexRank** additionally covered a multi-document summarization approach (requiring post-processing; "CSIS")
- The **TextRank** authors expanded their work to keyword extraction

Erkan, G., and Radev, D.R. (2004). LexRank: Graph-based Lexical Centrality as Salience in Text Summarization.

Abstractive Summarization with Recurrent Neural Networks



Generates new text using the full sentence context (attention mechanism) from the current text (word embeddings), while at the same time it can copy facts/words (pointer generator) over to the new text.

See, A., Liu, P.J., and Manning, C.D. (2017). Get To The Point: Summarization with Pointer-Generator Networks.

A first look at probabilistic graphical models

- Latent Dirichlet Allocation: LDA
  - Blei, Ng, and Jordan. Journal of Machine Learning Research 2003
  - For assigning “topics” to “documents” i.e., for text classification
  - An **unsupervised**, **generative** model

Latent Dirichlet Allocation (LDA 1/3)

- Intuition for LDA
  - From: Edwin Chen. Introduction to LDA. 2011
  - “Document Collection”
    - I like to eat broccoli and bananas.
    - I ate a banana and spinach smoothie for breakfast.

      ➡ Topic A
    - Chinchillas and kittens are cute.
    - My sister adopted a kitten yesterday.

      ➡ Topic B
    - Look at this cute hamster munching on a piece of broccoli.

      ➡ Topic 0.6A + 0.4B

Topic A: 30% broccoli, 15% bananas, 10% breakfast, 10% munching, …

Topic B: 20% chinchillas, 20% kittens, 20% cute, 15% hamster, …

The Dirichlet process

A Dirichlet process is like drawing from an (infinite) "bag of dice" (with finite faces).

- A Dirichlet is a [possibly continuos] **distribution over** [discrete/multinomial] **distributions** (probability **masses**).

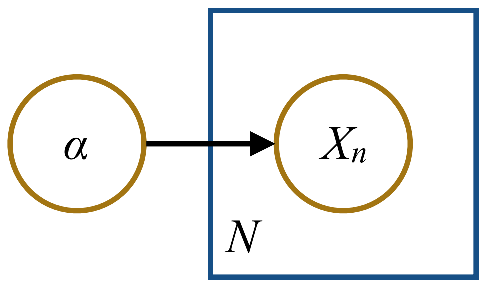
Gamma function -> a "continuous" factorial [!]

$$D(\boldsymbol{\theta}, \boldsymbol{\alpha}) = \frac{\Gamma(\sum \alpha_i)}{\prod \Gamma(\alpha_i)} \prod \theta_i^{\alpha_i - 1}$$

α Dirichlet prior: ∀ $\alpha_i \in \boldsymbol{\alpha}$: $\alpha_i > 0$

$\sum \theta_i = 1$; a Probability Mass Function

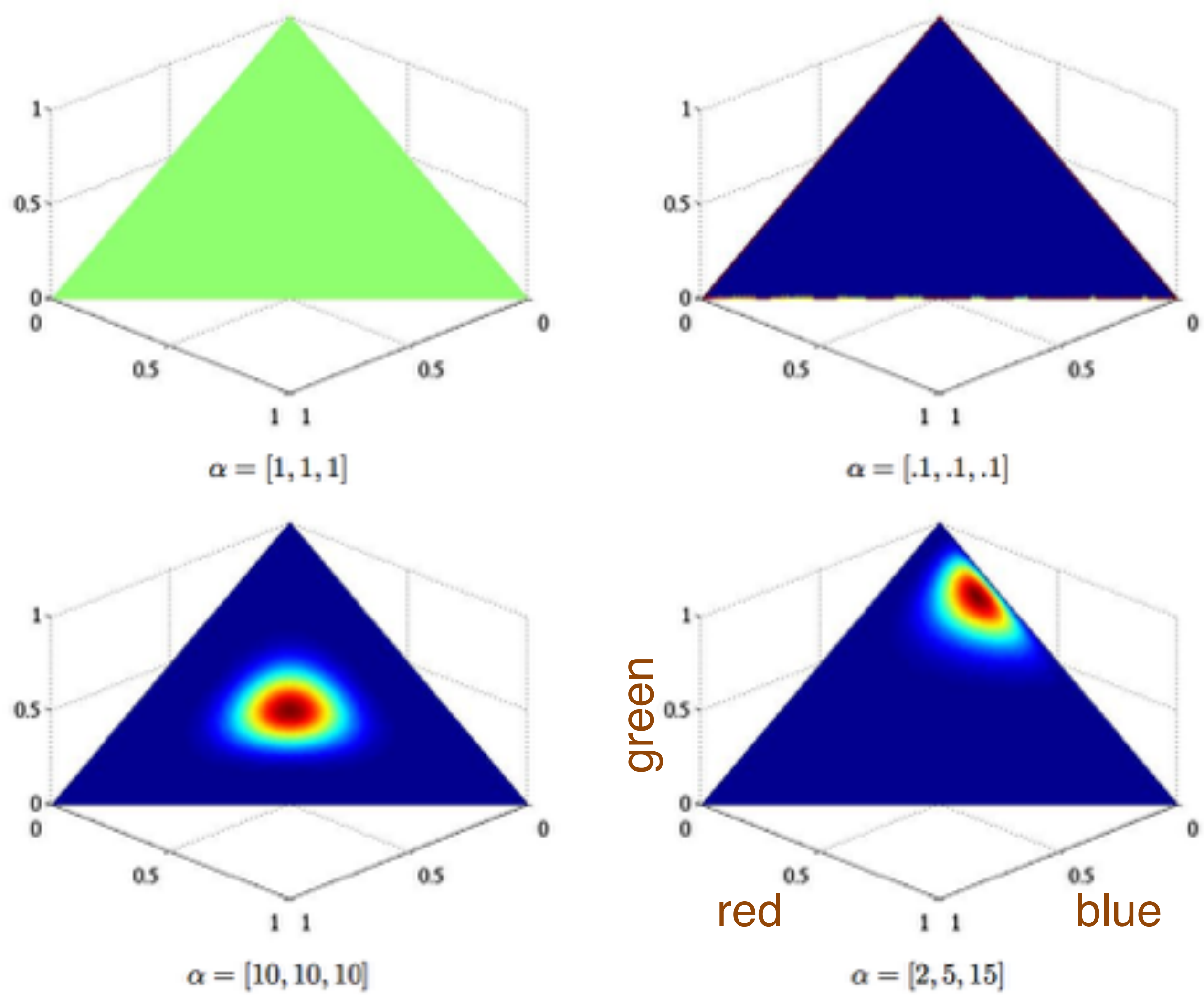
- The **Dirichlet Process samples** multiple independent, discrete **distributions** θ_i with repetition from $\boldsymbol{\theta}$ ("statistical clustering").



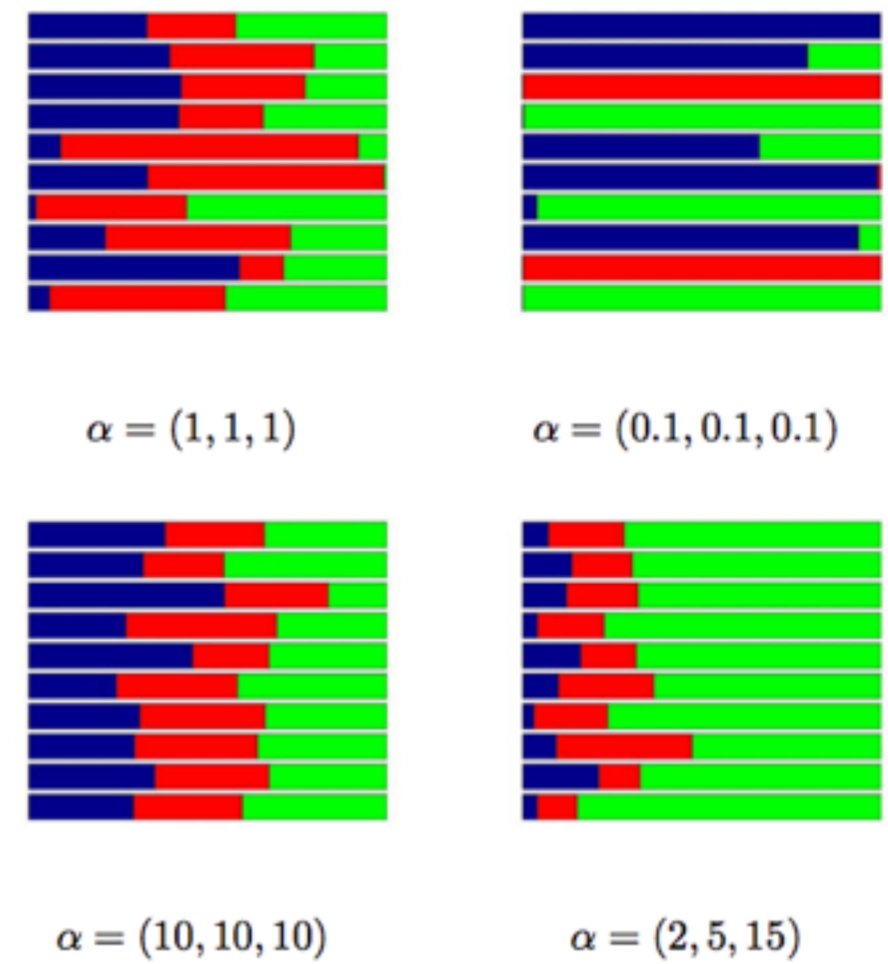
1. Draw a new distribution X from $D(\boldsymbol{\theta}, \boldsymbol{\alpha})$
2. With probability $\alpha \div (\alpha + n - 1)$ draw a new X
   With probability $n \div (\alpha + n - 1)$, (re-)sample an X_i from X

The Dirichlet prior α

"density plots over the probability simplex in R3"



Documents and topic distributions (N=3)

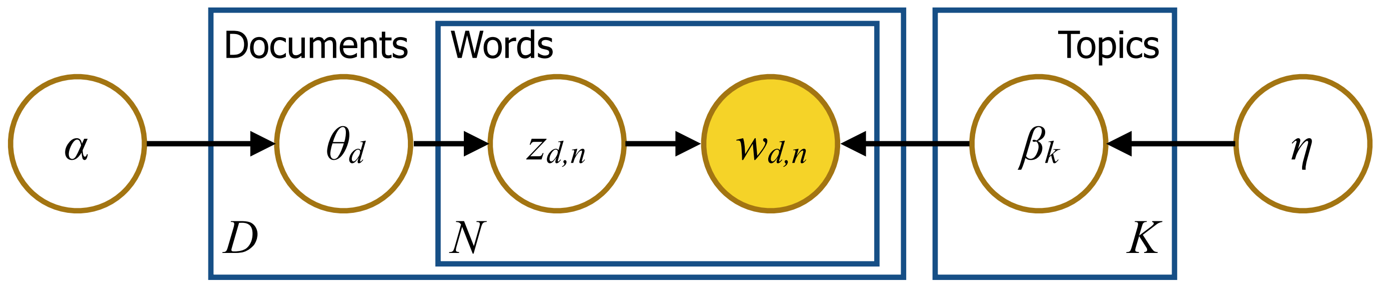


→ equal, =1 ➡ uniform distribution
→ equal, <1 ➡ marginal distrib. ("choose few")
→ equal, >1 ➡ symmetric, mono-modal distrib.
→ not equal, >1 ➡ non-symmetric distribution

Frigyik et al. Introduction to the Dirichlet Distribution and Related Processes. 2010

Latent Dirichlet Allocation (LDA 2/3)

A Document-Topic is the assignment of a Document to some Topic.
A Word-Topic is the assignment of a (non-unique!) Word (in a Document) to some Topic



Joint Probability

P(Document-T.)

P(Word I Topics, Word-T.)

$$P(B,\Theta,Z,W) = \left(\prod_{k}^{K} P(\beta_k|\eta)\right)\left(\prod_{d}^{D} P(\theta_d|\alpha)\prod_{n}^{N} P(z_{d,n}|\theta_d)P(w_{d,n}|\beta_{1:K},z_{d,n})\right)$$

P(Topic)

P(Word-T. I Document-T.)

- α - per-document Dirichlet prior
- θ_d - topic distribution of document d
- $z_{d,n}$ - word-topic assignments
- $w_{d,n}$ - **observed** words
- β_k - word distrib. of topic k
- η - per-topic Dirichlet prior

dampens the topic-specific score of terms assigned to many topics

$$termscore_{k,n} = \hat{\beta}_{k,n}\, \boxed{log\frac{\hat{\beta}_{k,n}}{\left(\prod_j^K \hat{\beta}_{j,n}\right)^{1/K}}}$$

What Topics is a Word assigned to?

Latent Dirichlet Allocation (LDA 3/3)

- LDA sampling/inference in a nutshell
  - Initialization: Choose K, the number of Topics, and randomly assign one out of the K Topics to each of the N Words in each of the D Documents.
    - The **same word** can have different Topics **at different positions** in the Document.
  - Calculate the posterior probability that Topic t generated Word w.
  - Then, for each Topic and for each Word in each Document:
    1. Compute P(Word-Topic | Document): the proportion of [Words assigned to] Topic t in Document d
    2. Compute P(Word | Topics, Word-Topic): the probability a Word w is assigned a Topic t (using the general distribution of Topics and the Document-specific distribution of [Word-] Topics)
       - Note that a Word can be assigned a different Topic each time it appears in a Document.
    3. Given the prior probabilities of a Document's Topics and that of Topics in general, reassign P(Topic | Word) = P(Word-Topic | Document) * P(Word | Topics, Word-Topic)
  - Repeat until P(Topic | Word) stabilizes (e.g., Collapsed Gibbs sampling)
  - Better: Use collapsed **variational inference** (i.e., combining Variational Bayes)

Teh, Newman, Welling (2006). A Collapsed Variational Bayes Inference Algorithm for LDA