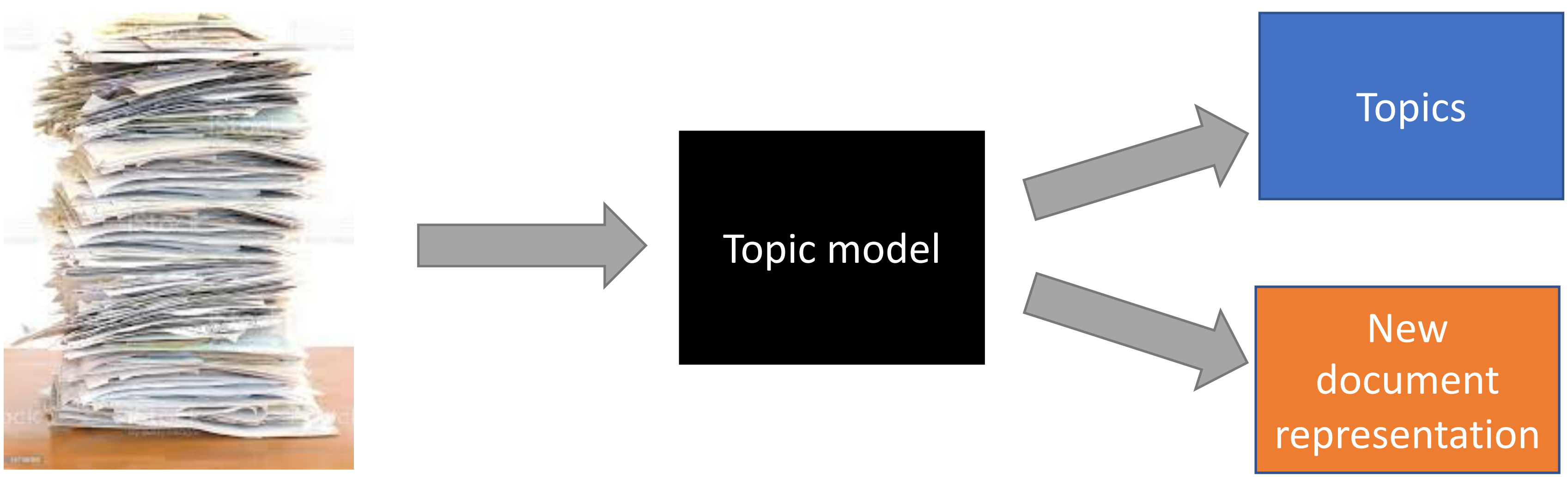
Topic Models

ECE57000: Artificial Intelligence

David I. Inouye

Topic models are unsupervised methods for text data that extract topic and document representations

1. Given a dataset of text documents (often called a **corpus**), what are the main topics or themes?
2. Can you find a compressed semantic representation of each document/instance?



Motivation: Difficult to discover new and relevant information in uncategorized text collections

- Example: New York Times news articles
  - Automatically categorize articles into different themes
  - How do these themes change over time?
  - What specific articles are in each theme?

- Expensive manual option: Employ many humans to carefully read and categorize

- Cheap automatic option: Use topic models!
  - No labels are required! Just raw text

Other examples that could leverage topic models

- Survey responses
- Customer feedback
- Research papers
- Emails

Preliminary: How should a collection of documents be represented?

- Two naïve assumptions

1. Each word is considered a single unit (called **unigram**)

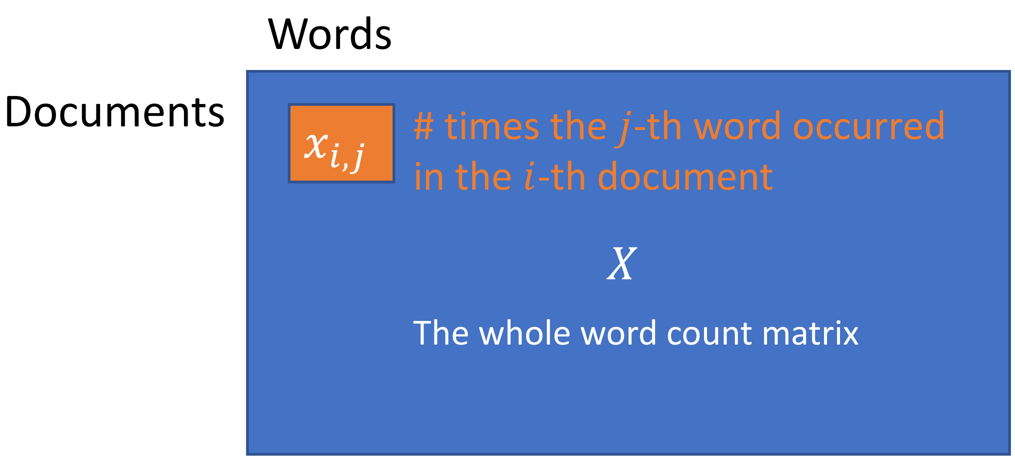
The sun is bright.
The sun is red.
----
2 1 3 4
2 1 2 5

2. Order of words ignored (**Bag-of-words** assumption)

the sun is bright
=
bright sun the is

Preliminary: The document collection can be represented as a word-count matrix

- Each row represents a document
- Each column represents a word
- Each element represents the number of times (i.e., count) that word occurred in the document



Create word-count matrix in scikit-learn: https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html

Example word-count matrix

- This movie is very scary and long
- This movie is not scary and is slow
- This movie is spooky and good

	1 This	2 movie	3 is	4 very	5 scary	6 and	7 long	8 not	9 slow	10 spooky	11 good	Length of the review(in words)
Review 1	1	1	1	1	1	1	1	0	0	0	0	7
Review 2	1	1	2	0	0	1	1	0	1	0	0	8
Review 3	1	1	1	0	0	0	1	0	0	1	1	6

https://www.analyticsvidhya.com/blog/2020/02/quick-introduction-bag-of-words-bow-tf-idf/

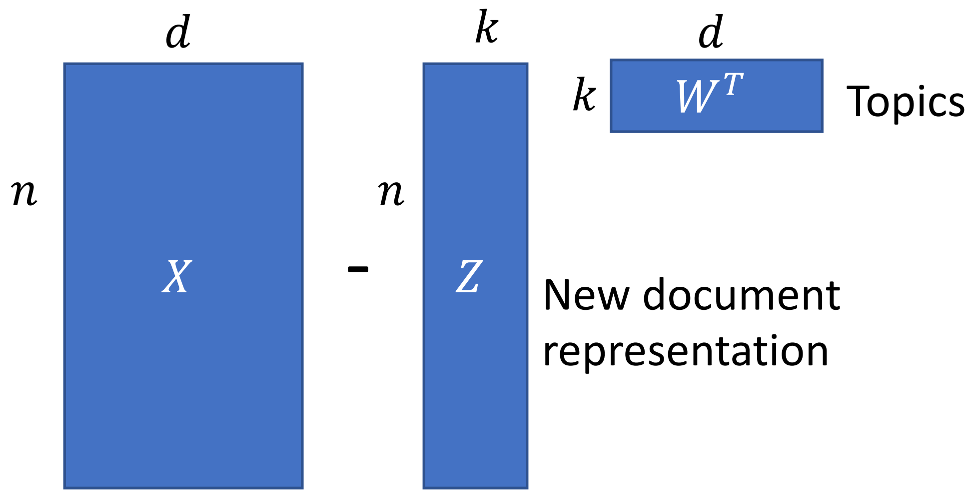
Latent semantic indexing (LSI) is one of the simplest topic models and uses truncated SVD

- Optimization over low rank matrices Z and W

$$Z, W = \min_{Z,W} \|X - ZW^T\|_F^2$$

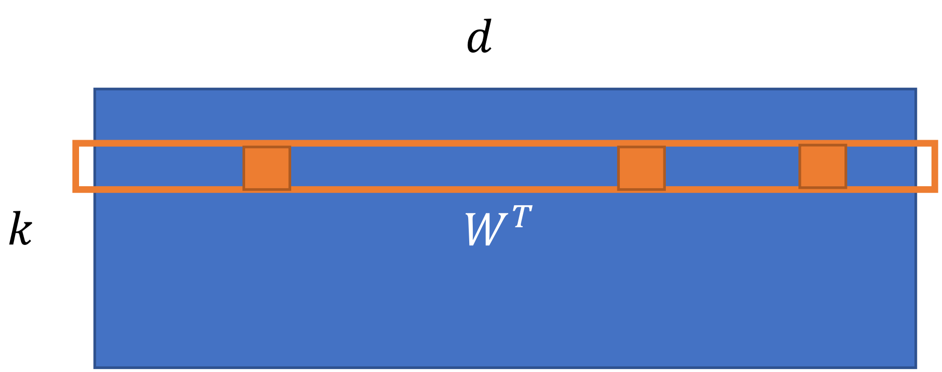
- Solution: Truncated SVD of $X = USV^T$

$$Z = US_k, \qquad W = V_k$$



LSI "topics" can capture <u>synonymy</u> or similarity between words

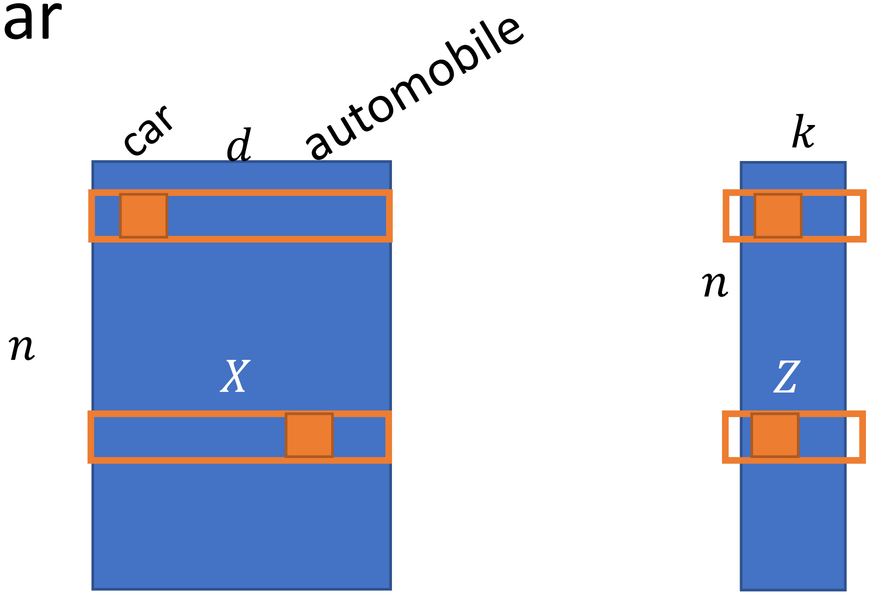
- Examples:
  - "Car" and "automobile" (synonyms)
  - "School" and "education" (related)
- These related words will tend to have high weights in the same row of the topic matrix W^T



"Automotive" topic may have high values on columns for "car", "automobile" and "truck".

LSI document representation groups documents even if their exact words do not overlap

- Example
  - One document only uses the word “car”
  - One document only uses the word “automobile”
  - The documents may have no exact words shared but are similar



LSI problem: Interpretation of topics and representations is challenging since values could be arbitrary

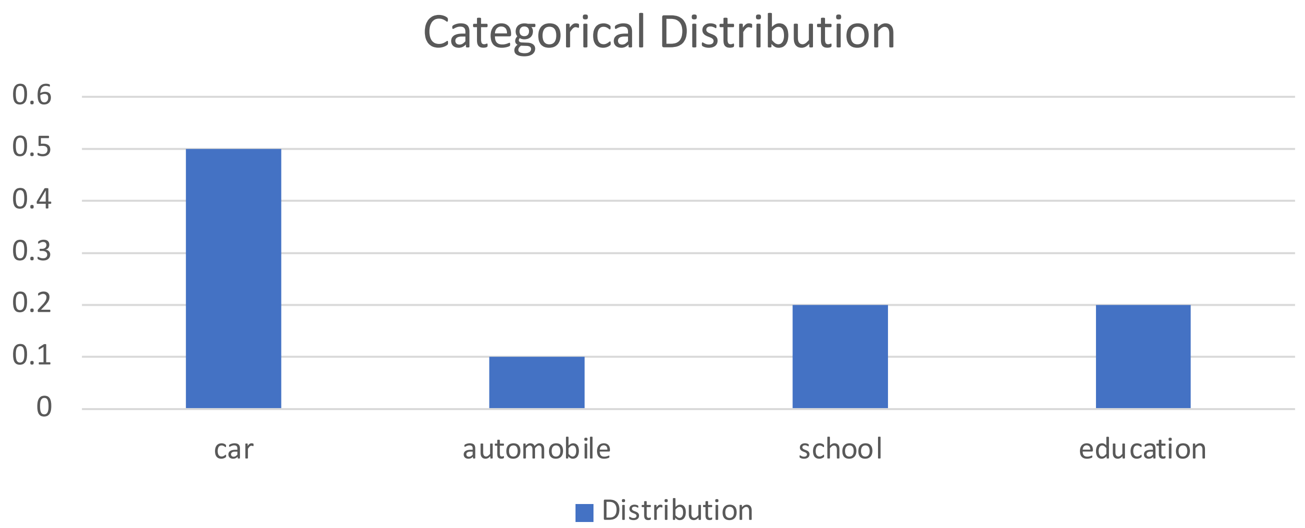
- SVD implicitly assume data is real-valued
  - (e.g., -2.1, 3.5, -1.2, 100.1)
- Yet input word-count matrix is discrete data
  - Non-negative integer values (e.g., 0,1,2,3,etc.)
- What do negative values mean?
(e.g., automobile is 1.1 but school is -0.5)
- What does the scale of these values mean?
(e.g., 4 or 0.2)

LSI problem: No generative model to create new data (less deep understanding)

- Like the difference between AEs and VAEs
  - VAEs provide a way to generate fake new data
- “What I cannot create, I do not understand.” – Richard Feynman
- Previously we’ve considered mostly *continuous* generative models (GANs, VAEs, flows, etc.)
- What about discrete generative models?

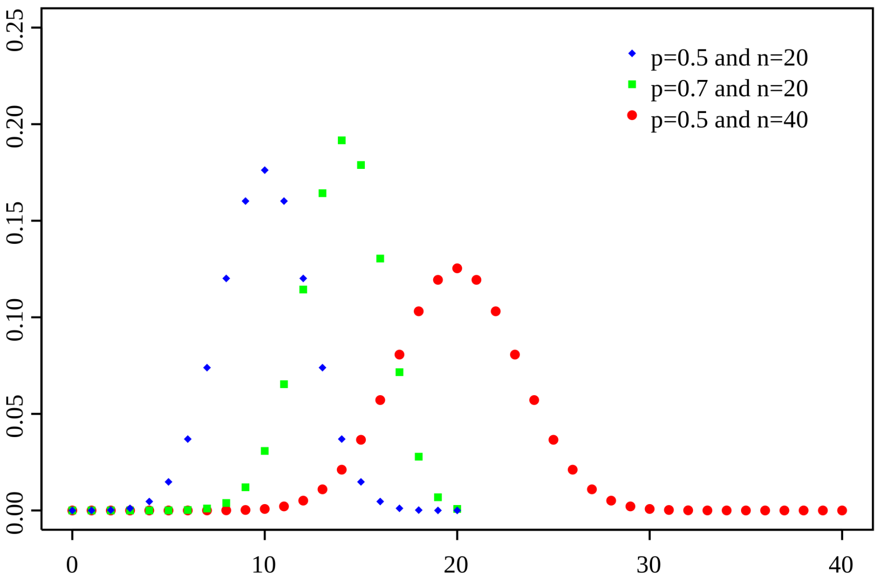
The <u>**categorical distribution**</u> generalizes the Bernoulli (coin flip) distribution to many outcomes

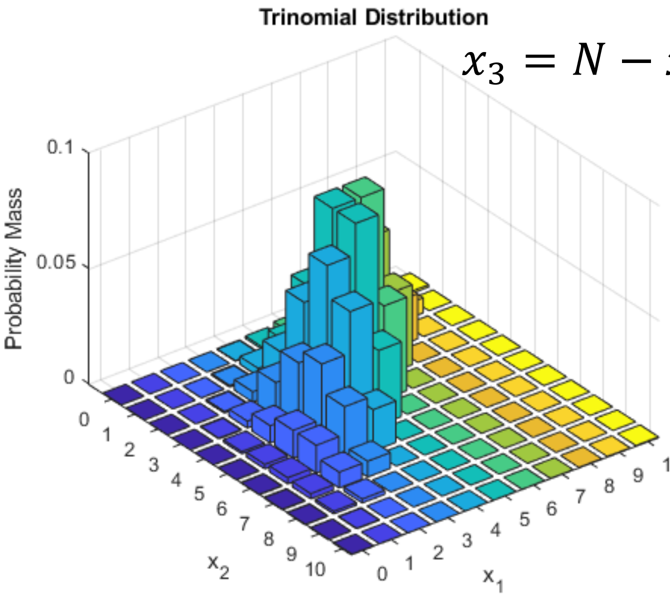
- Intuition, rolling a d-sided dice
- Each side has a probability $p_s = \Pr(x = s)$
- In our case, d is the number of unique words in our corpus



The multinomial distribution is a simple model for count data (the “Ind. Gaussian” for count data)

- Intuition, roll d-sided dice N times and record count for each side
- Example: Flip a biased coin 10 times and count how many are heads and tails





$$x_3 = N - x_1 - x_2$$

The multinomial distribution is a simple model for count data (the "Ind. Gaussian" for count data)

- Word counts can be modeled as
$$x \sim \text{Multinomial}(p; N)$$
  - p is the probability for each word
  - N is the number of words in the document
    - $N = \sum_s x_s = \|x\|_1$
- Log PMF is:

$$\log P_{\text{mult}}(x) = \log \frac{N!}{x_1! \cdots x_d!} \prod_{s=1}^{d} p_s^{x_s} = \sum_{s=1}^{d} x_s \log p_s + c$$

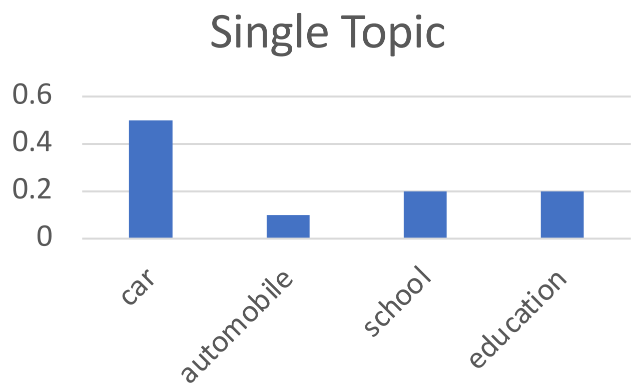
A mixture of multinomials adds complexity like mixture of Gaussians

- Let $x \sim \mathrm{MixtureMult}(\pi, (p_1, \cdots, p_k); N)$
  - π is the mixture weights
  - p_j is the probability vector for the j-th multinomial component distribution
  - N is the number of words in a document
- The log PMF is:

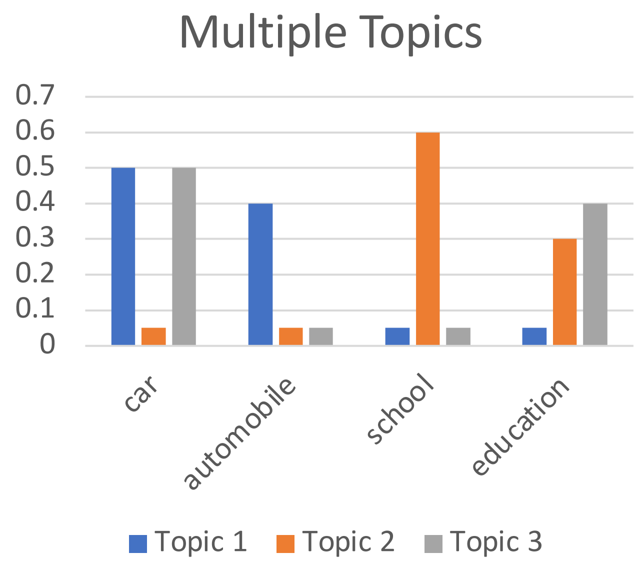
$$\log P_{\mathrm{mult}}(x) = \log \sum_{j=1}^{k} \pi_j P_{\mathrm{mult}}^{j}(x) = \log \sum_{j=1}^{k} \Pr(z = j)\, P_{\mathrm{mult}}^{j}(x)$$

Interpretation of multinomials and mixture of multinomials

- Multinomial distribution
  - Assumes all documents have the same “topic”
  - A topic is the probability for each word

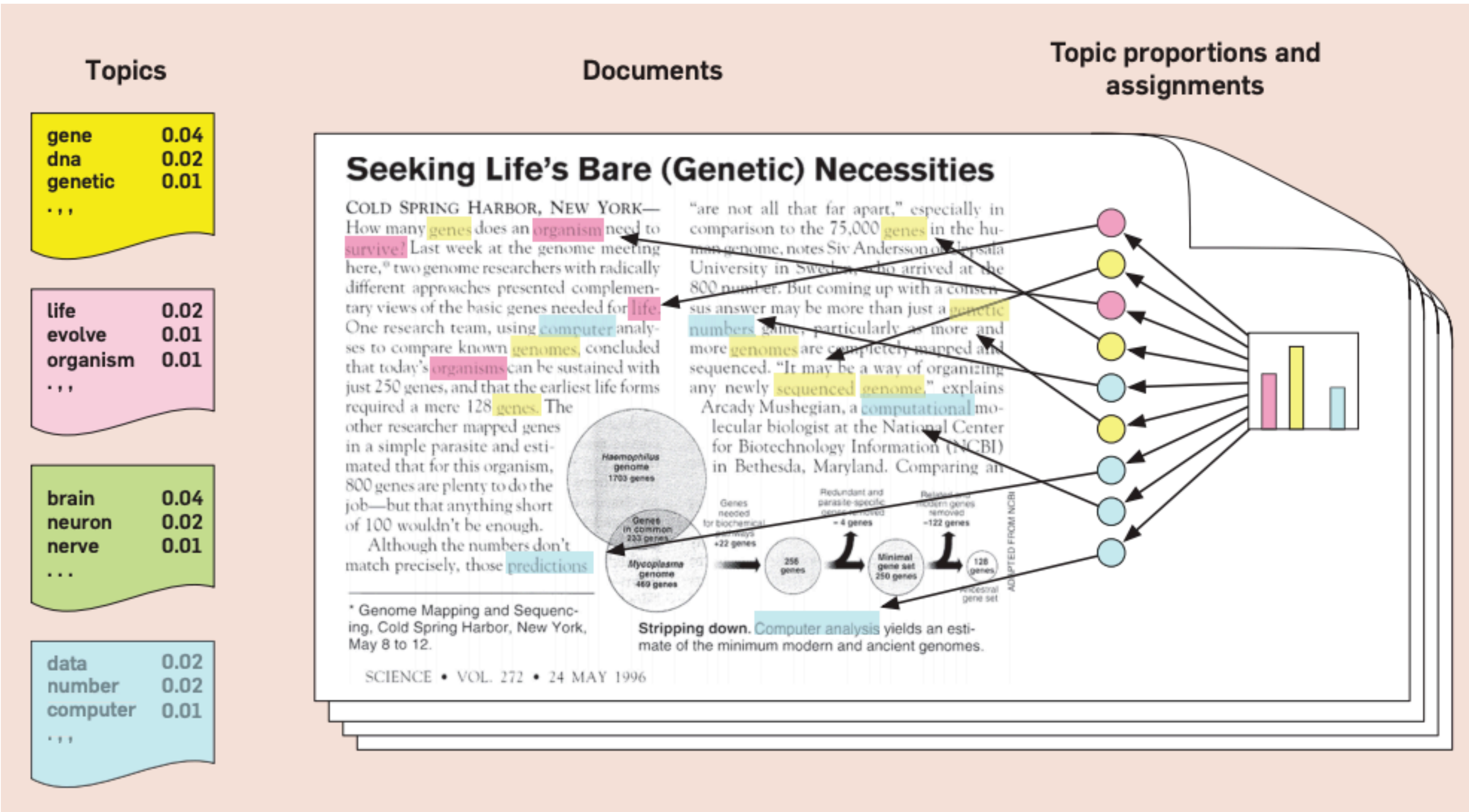


- Multinomial mixture
  - Each component represents a topic
  - Each document only has one topic



- What if each documents have multiple topics?

Latent Dirichlet Allocation (LDA) defines a model where each document can have multiple topics



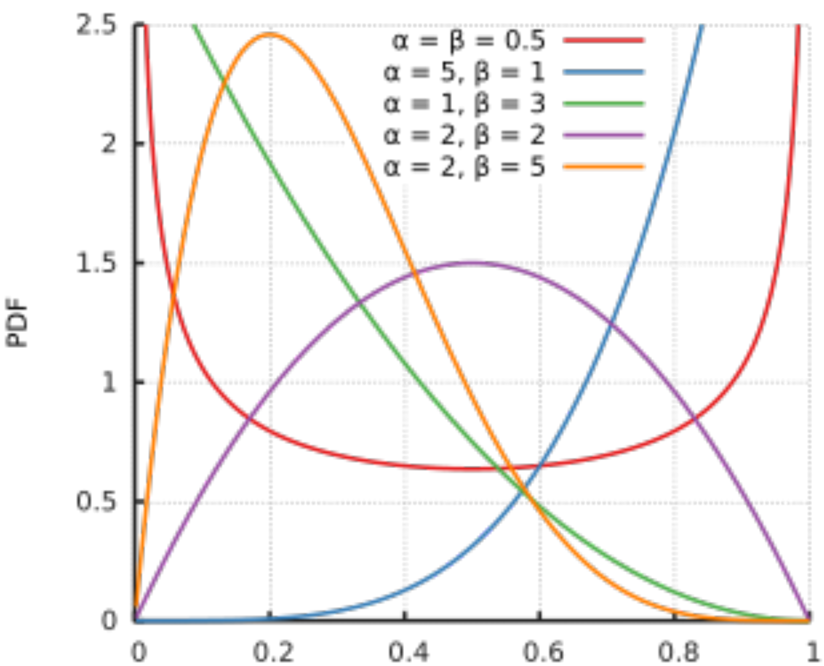
Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77-84.

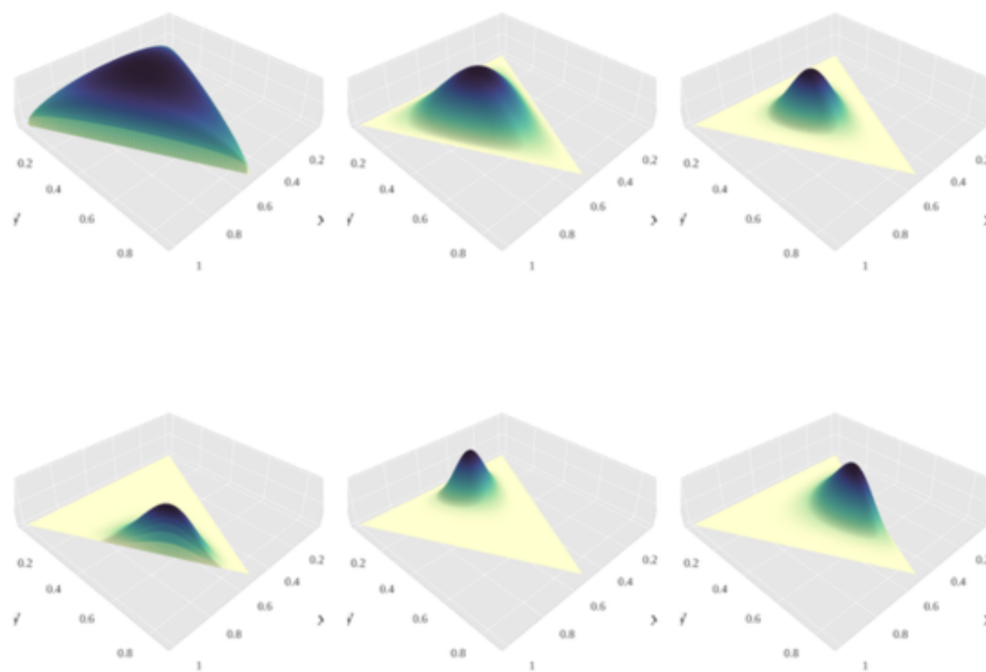
Background: Dirichlet distribution is a distribution over the probability simplex

- The **probability simplex** is the set of vectors that are non-negative and sum to 1

$$\Delta^d = \{x \in [0,1]^d : \sum x_s = 1\}$$

- Dirichlet is simplest distribution on this set

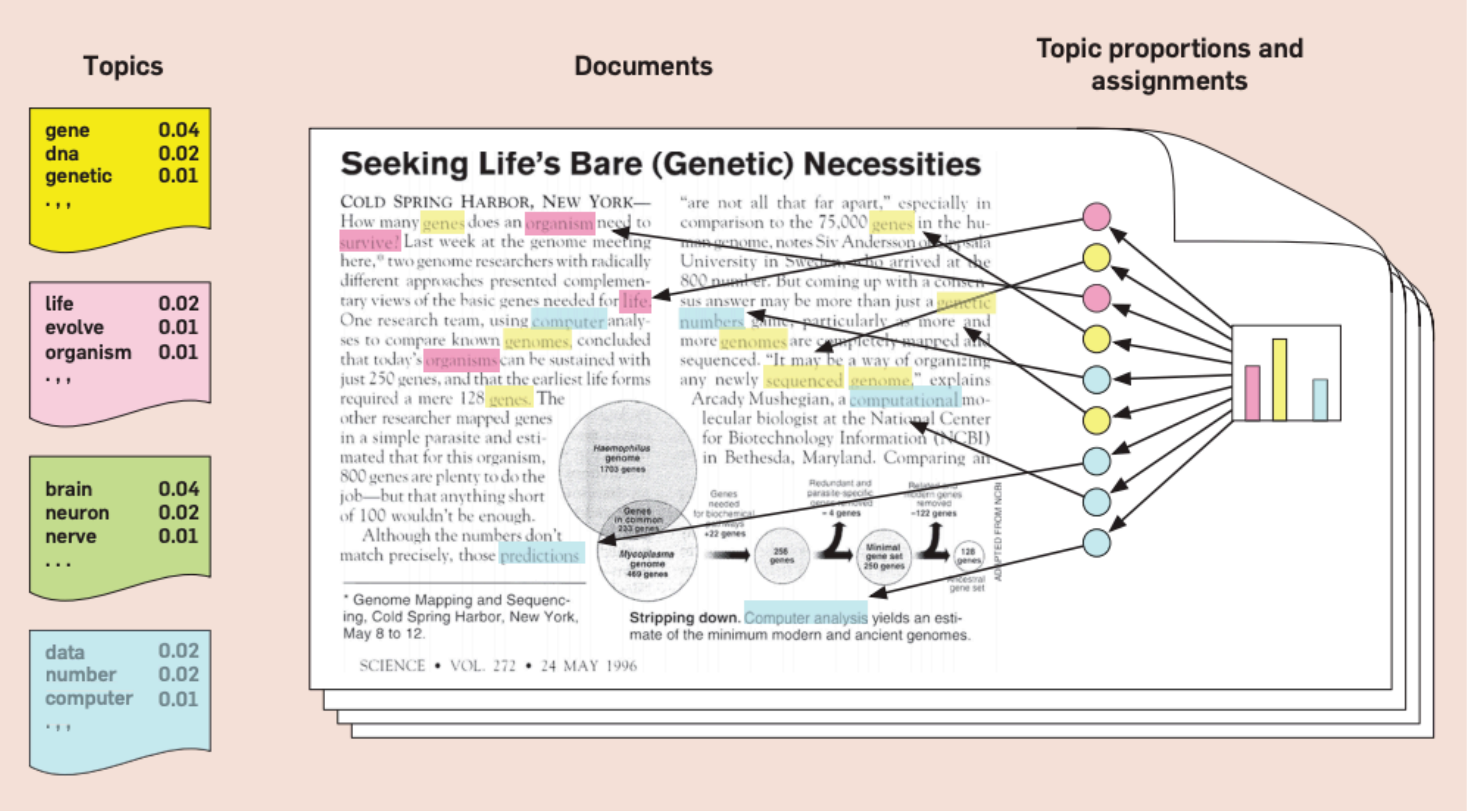




The generative process of LDA is a mixture of mixtures (or admixture)

- Mixture generative process (assume N is fixed)
  - Sample single topic $z \sim \text{Categorical}(\pi)$
  - Repeat $\ell = 1$ to N:
    - Sample individual words $w_\ell \sim \text{Categorical}(p_z)$ (where w_ℓ are one hot vectors)
  - $x = \sum w_\ell$ (equivalent to $x \sim \text{Multinomial}(p_z; N)$)
- LDA generative process (assume N is fixed)
  - Sample mixture over topics $\theta_i \sim \text{Dirichlet}(\alpha)$
  - Repeat $\ell = 1$ to N
    - Sample topic of word $z_\ell \sim \text{Categorical}(\theta_i)$
    - Sample individual words $w_\ell \sim \text{Categorical}\left(p_{z_\ell}\right)$
  - $x = \sum w_\ell$ (equivalent to $x \sim \text{Multinomial}([p_1, \cdots, p_k]\theta_i; N)$)

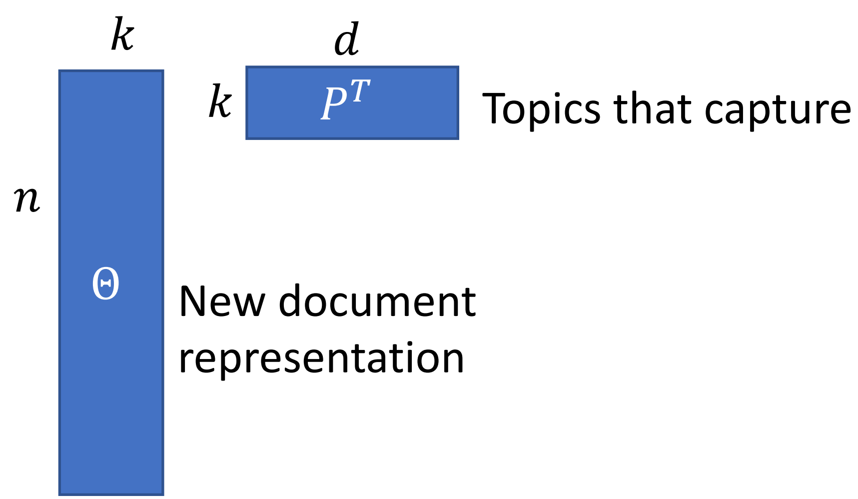
Latent Dirichlet Allocation (LDA) defines a model where each document can have multiple topics



Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77-84.

After training, we can recover more interpretable topics and document representations

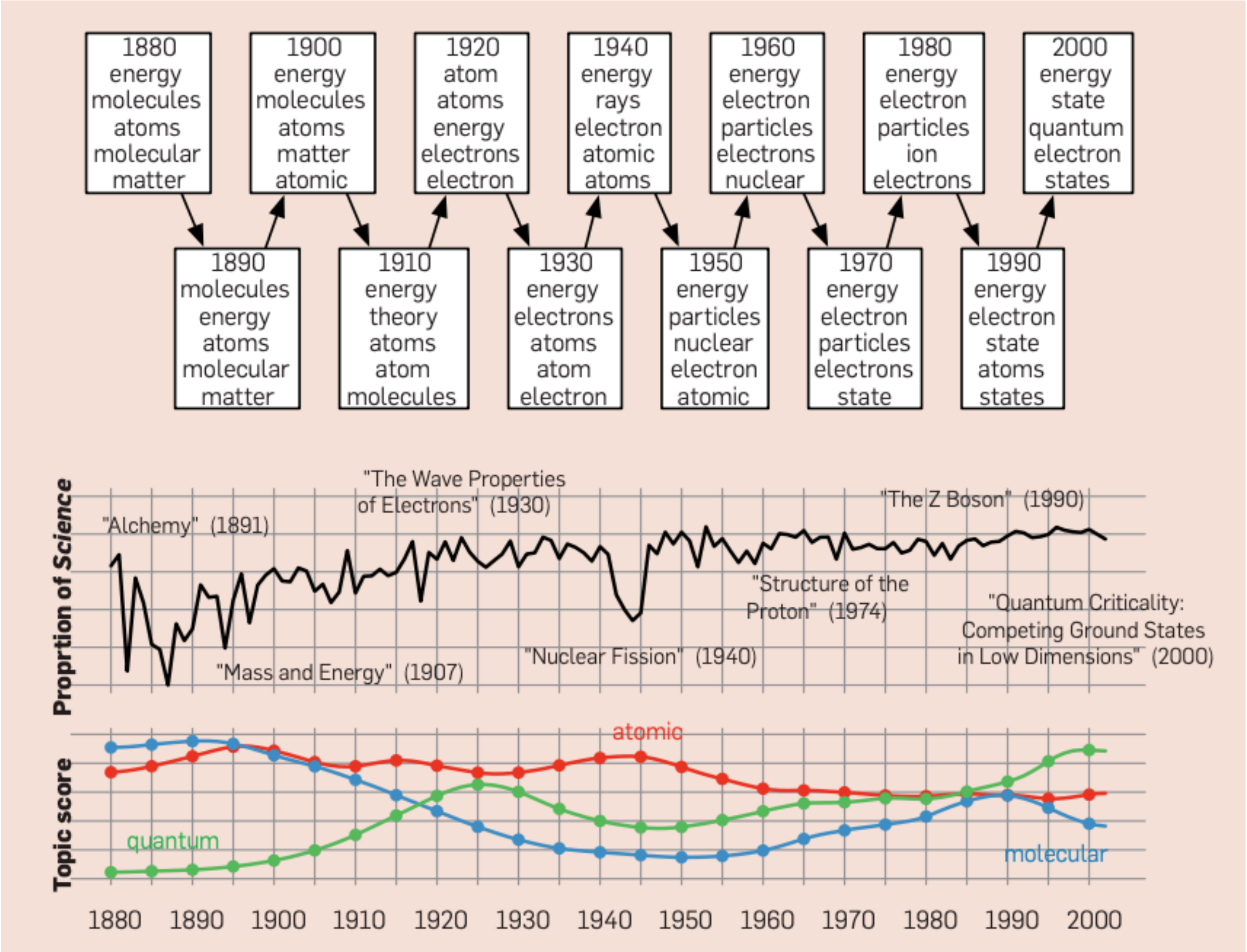
- Each topic is a probability distribution $p_j \in \Delta^d$
- Each document is represented by a probability distribution over topics $\theta_j \in \Delta^k$
- Can be seen as "discrete PCA" method



Estimating these generative models for text data

- Multinomial model
  - MLE has closed form solution (merely empirical frequencies)
- Mixture of multinomials
  - Could use EM algorithm or other mixture-based algorithms
- LDA
  - Variational inference (i.e., use ELBO as in VAEs)
  - MCMC/Gibbs sampling (often performs better)

Dynamic topic models can track topics over time



Blei, D. M. (2012). Probabilistic topic models. *Communications of the ACM*, *55*(4), 77-84.

Additional resources for topic modeling

- Gentle introduction to topic modeling
  http://www.cs.columbia.edu/~blei/papers/Blei2012.pdf

- More resources/tutorials
  http://www.cs.columbia.edu/~blei/topicmodeling.html

- Text analysis with scikit-learn
  https://scikit-learn.org/stable/tutorial/text_analytics/working_with_text_data.html