Text-as-data for Social Science II

KAPS Summer Methods Workshop

Taegyoon Kim | KAIST

Jul 11, 2024

Topic Models

Topic Models: themes

Things to be covered

- What is topic modeling
- Latent Dirichlet Allocation
- Extensions
  - Correlated Topic Model
  - Structural Topic Model
- Model selection and interpretation
- Summary

What is Topic Model

Multi-membership, unsupervised algorithms to discover topics

- Offers an automated method to discover topics in a corpus of documents
  - Used to understand and organize large collections of documents according to the discovered topics
  - Either for exploration or for measurement
- Documents can contain multiple topics ($\Longleftrightarrow$ clustering algorithms)
- Unsupervised learning (no manual labeling of topics)

What is Topic Model

Multi-membership, unsupervised algorithms to discover topics

- Topic: what is being talked about/written about
  - Unclear what topic actually means in theoretical terms
  - Topic models assume an intuitive and abstract notion of a topic
- *"The meaning of a topic in an LDA topic model must be assessed empirically instead and defined against the background of substantive theoretical concepts, such as political issues or frames* (Maier et al. 2018)

What is Topic Model

LDA and its extensions

- Latent Dirichlet Allocation (LDA) is one fundamental approach (Blei et al., 2003)
- Alternative approaches
  - Correlated Topic Model (CTM)
  - Structural Topic Model (STM)
  - Clustering of (neural) embeddings (non-generative)

Topic Model in Context

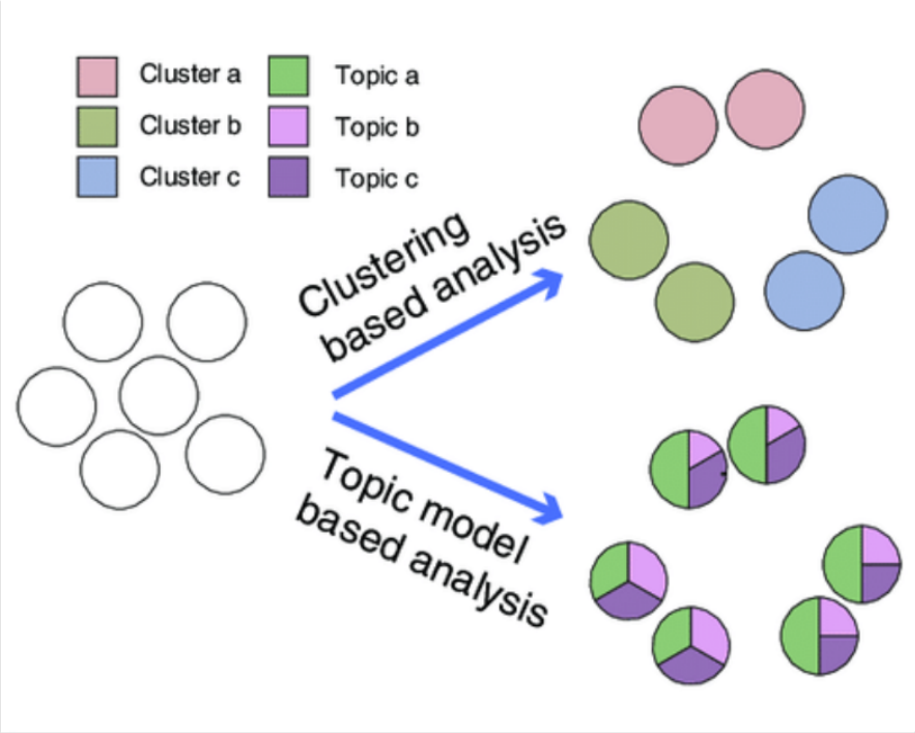
Supervised vs. Unsupervised

	Supervised	**Unsupervised**
Objective	Trained on a labeled data to learn a mapping from input to output	Find patterns or structures within data without labels
Outcome	Pre-defined categories	Not quite pre-defined
Model evaluation	Explicit metrics such as accuracy, precision, recall, or R^2	Can involve qualitative assessment
Examples	Classification/regression for texts	**Topic models**

Topic Model in Context

Clustering algorithms vs. Topic models

- Clustering (e.g., K-means) assumes that each document belongs to one cluster
- Documents can have more than one idea in them (e.g., political speeches, newspaper articles, novels)
- Click for figure source



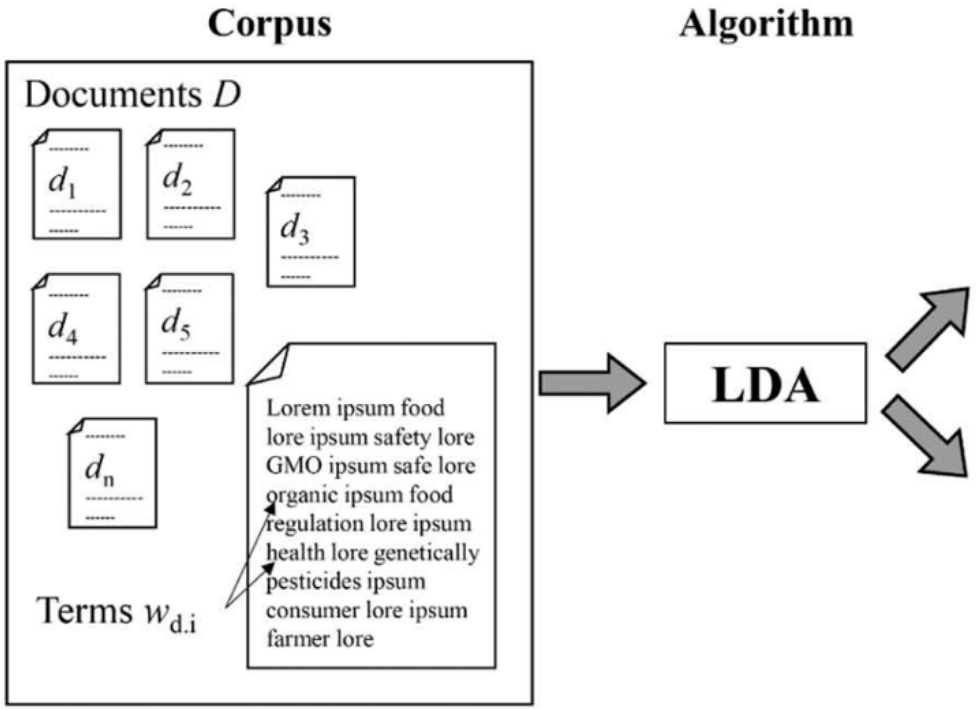
Latent Dirichlet Allocation (LDA)

Used to identify the latent topic structure within a corpus

- One of many statistical algorithms for topic modeling
- Most seminar and widely used model
- Estimated via Bayesian frameworks
- “Documents” are seen as “distributions over topics”
- “Topics” are seen as “distributions over words”
- Make use of the BoW (Bag of Words) assumption

Latent Dirichlet Allocation (LDA)

The framework of LDA (adaped from Maier et al. 2018)



Algorithm

LDA

Results

V	Topic$_1$	Topic$_2$	…	Topic$_K$
food	0.09	0.07	…	0.02
health	0.10	0.05	…	0.01
GMO	0.04	0.01	…	0.20
safety	0.03	0.03	…	0.05
farmer	0.01	0.10	…	0.07
…	…	…	…	…

D	Topic$_1$	Topic$_2$	…	Topic$_K$
d_1	0.10	0.04	…	0.02
d_2	0.05	0.03	…	0.02
d_3	0.02	0.03	…	0.09
d_4	0.07	0.05	…	0.01
…	…	…	…	…
d_D	0.08	0.01	…	0.10

LDA: Key Distributions

Multinomial distribution

- Generalization of binomial distribution (e.g., flipping a coin)
- Probabilities of different outcomes (not just two)
- E.g., rolling a (6-sided) dice
  - Each side (from 1 to 6): discrete outcome
  - The probabilities sum up to 1 ($\frac{1}{6} + \ldots + \frac{1}{6} = 1$)
- E.g., allocating a pie to one of 3 people
  - Each person: discrete outcome
  - The probabilities sum up to 1 ($\frac{1}{4} + \frac{1}{4} + \frac{1}{2} = 1$)

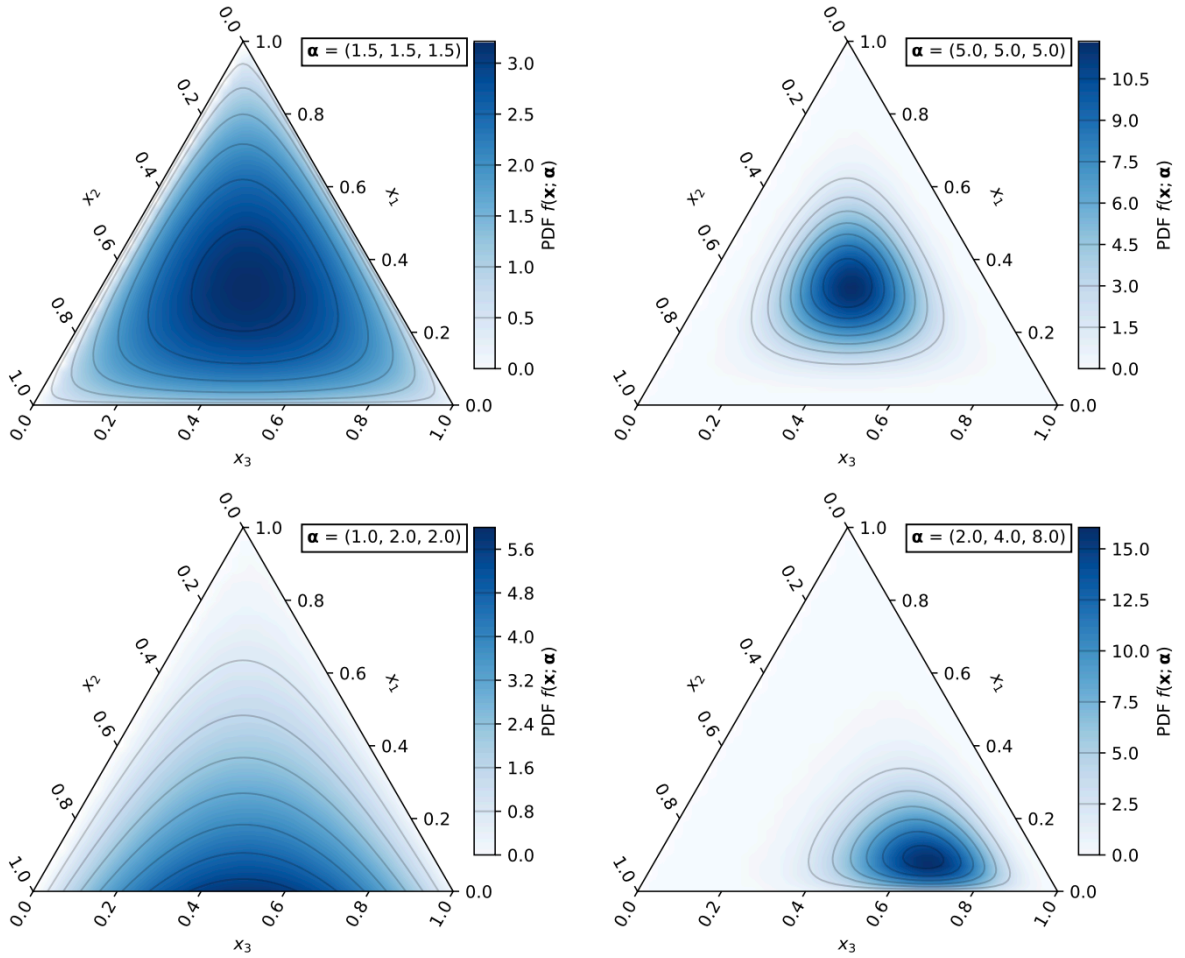
LDA: Key Distributions

Dirichlet distribution

- Provides a way to randomly generate *multinomial distributions*
- The pie split example
  - Imagine if you did not decide ahead of time exactly how to slice the pie
  - Instead, we specify a random process to decide how likely each person gets the pie
  - Equal chances among the three people, or perhaps it's more likely that one person gets it than the others
  - E.g., `[0, 0, 1]`, `[1/3, 1/3, 1/3]`, `[1/2, 1/4, 1/4]`, etc.

LDA: Key Distributions

$Dir(x = 3; \alpha)$



LDA: Generative process

Consider a corpus of D documents, each with N_d words

- Assume a statistical model that generated our documents, then estimate the model to recover latent (unobserved) topics
- **Each document** is seen as **a multinomial distribution over topics**
- **Each topic** is seen as **a multinomial distribution over words**
- E.g., a stylized corpus with D = 5, N = 8 per document (and also 8 unique words in the corpus in total), and K = 3

LDA: Generative process

E.g., $D = 5$, $K = 3$, $N = 8$

- θ_d ~ Dir(3, α): for each d, its **topic distribution** is drawn from a Dirichlet distribution (e.g., θ_1 = [0.1, 0.7, 0.2])
- β_k ~ Dir(8, η): for each k, its **word distribution** is drawn from another Dirichlet distribution (e.g., β_2 = [0, 0, 0.2, 0.1, 0, 0, 0.3, 0.4])
- $z_{d,n}$ ~ Multinomial(θ_d): for each document-word position, its topic is drawn from θ_d (i.e., Topic 1 can be drawn)
- $w_{d,n}$ ~ Multinomial($\beta_{z_{d,n}}$): for each document-word position, its word is drawn from the corresponding topic distribution (i.e., Word 8 can be drawn)

LDA: Generative process

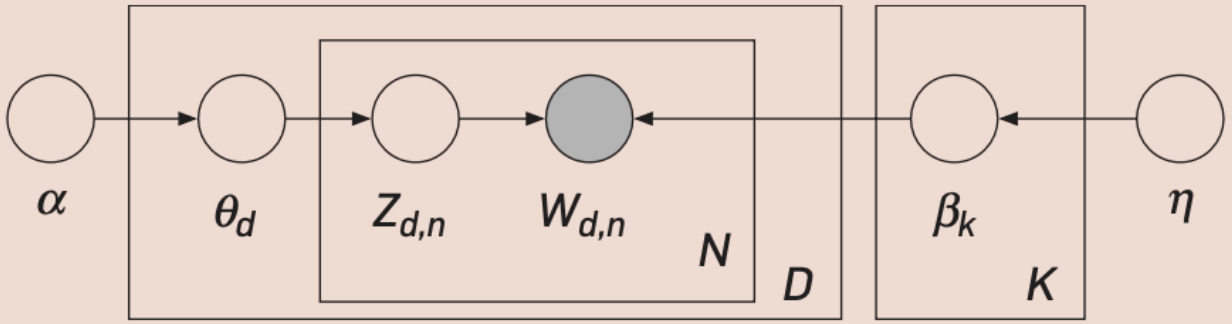
In summary, the joint probability is the following

- θ_d ~ Dir(α) for $d \in \{1, \ldots, D\}$
- β_k ~ Dir(η) for $k \in \{1, \ldots, K\}$
- $z_{d,n}$ ~ Multinomial(θ_d)
- $w_{d,n}$ ~ Multinomial($\beta_{z_{d,n}}$)

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D}) = \prod_{i=1}^{K} p(\beta_i) \prod_{d=1}^{D} p(\theta_d) \left(\prod_{n=1}^{N} p(z_{d,n} | \theta_d) p(w_{d,n} | \beta_{1:K}, z_{d,n}) \right)$$

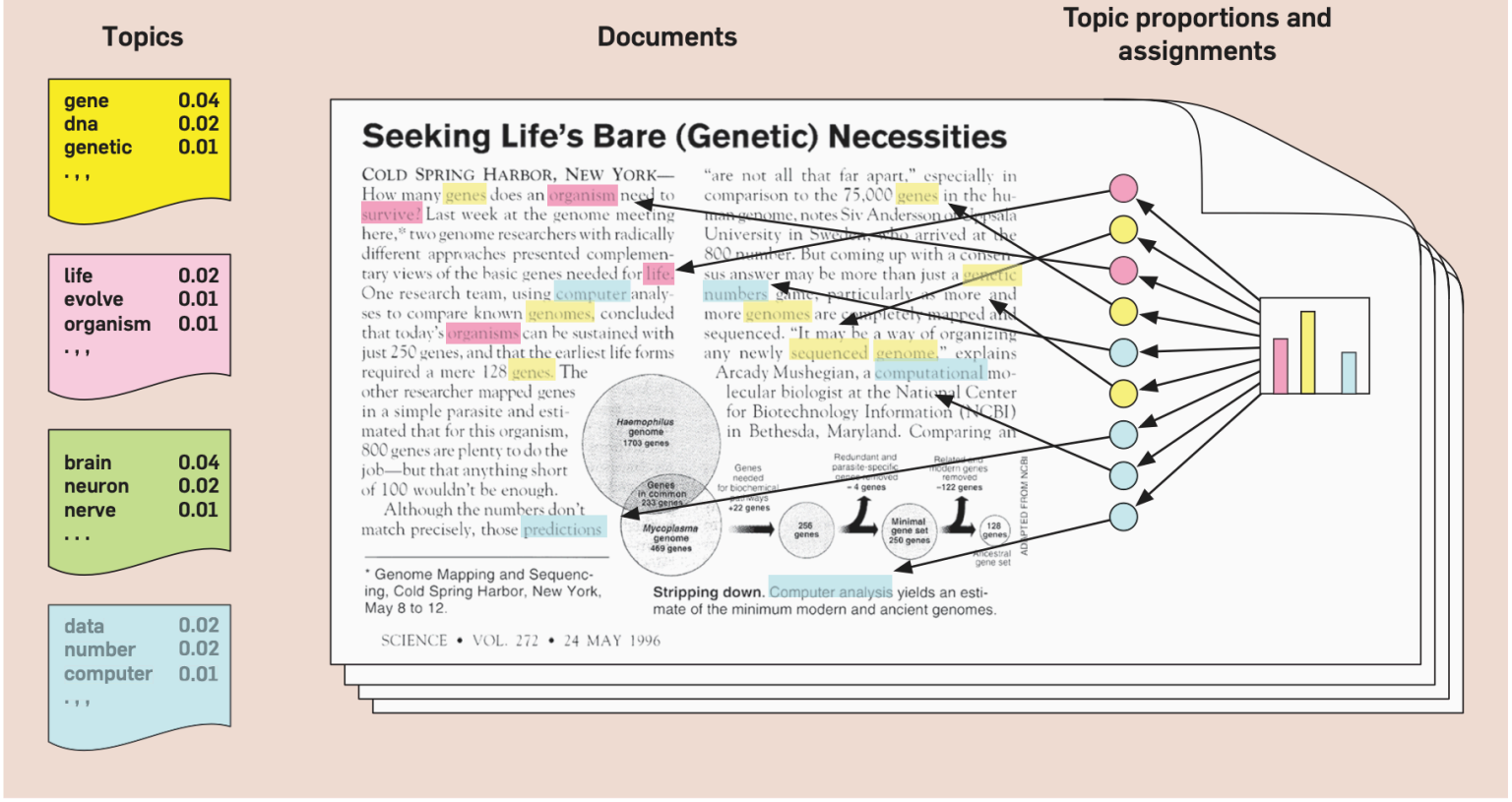
LDA: Generative process

Figure 4. The graphical model for latent Dirichlet allocation. Each node is a random variable and is labeled according to its role in the generative process (see Figure 1). The hidden nodes—the topic proportions, assignments, and topics—are unshaded. The observed nodes—the words of the documents—are shaded. The rectangles are "plate" notation, which denotes replication. The *N* plate denotes the collection words within documents; the *D* plate denotes the collection of documents within the collection.



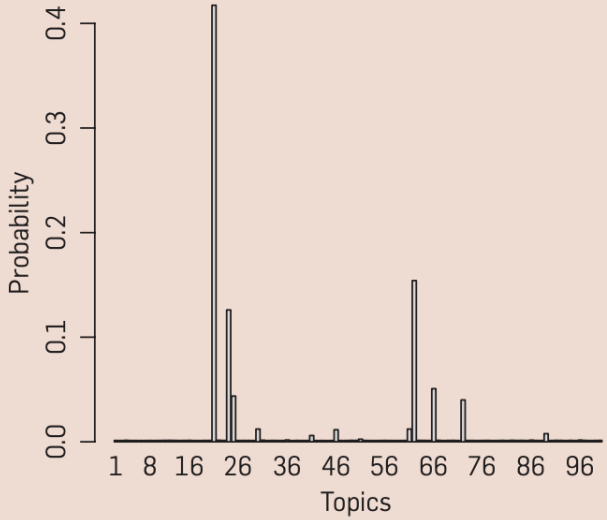
LDA: Generative process

Figure 1. The intuitions behind latent Dirichlet allocation. We assume that some number of "topics," which are distributions over words, exist for the whole collection (far left). Each document is assumed to be generated as follows. First choose a distribution over the topics (the histogram at right); then, for each word, choose a topic assignment (the colored coins) and choose the word from the corresponding topic. The topics and topic assignments in this figure are illustrative—they are not fit from real data. See Figure 2 for topics fit from data.



LDA: Generative process

Figure 2. Real inference with LDA. We fit a 100-topic LDA model to 17,000 articles from the journal *Science*. At left are the inferred topic proportions for the example article in Figure 1. At right are the top 15 most frequent words from the most frequent topics found in this article.



"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

LDA: Estimation

E.g., with $D = 1000$, $|V| = 10000$, and $K = 3$

- LDA estimates θ (distributions over topics) and β (distributions over vocabularly)

$$\underbrace{\theta = \begin{pmatrix} \theta_{1,1} & \theta_{1,2} & \theta_{1,3} \\ \theta_{2,1} & \theta_{2,2} & \theta_{2,3} \\ \dots & \dots & \dots \\ \theta_{1000,1} & \theta_{1000,2} & \theta_{1000,3} \end{pmatrix}}_{1000 \times 3} = \underbrace{\begin{pmatrix} 0.7 & 0.2 & 0.1 \\ 0.3 & 0.1 & 0.6 \\ \dots & \dots & \dots \\ 0.1 & 0.8 & 0.1 \end{pmatrix}}_{1000 \times 3}$$

$$\underbrace{\beta = \begin{pmatrix} \beta_{1,1} & \beta_{1,2} & \dots & \beta_{1,10000} \\ \beta_{2,1} & \beta_{2,2} & \dots & \beta_{2,10000} \\ \beta_{3,1} & \beta_{3,2} & \dots & \beta_{3,10000} \end{pmatrix}}_{3 \times 10000} = \underbrace{\begin{pmatrix} 0.04 & 0.01 & \dots & 0.0001 \\ 0.00002 & 0.001 & \dots & 0.05 \\ 0.00001 & 0.03 & \dots & 0.0001 \end{pmatrix}}_{3 \times 10000}$$

LDA: Estimation

Estimation is done in a Bayesian framework

- Estimation methods (see Blei (2012) for more discussions)
  - Gibbs sampling methods
  - Variational approximations

Model Selection

Determining hyperparameters: K (and α, η)

- A combination of quantitative metrics and human judgement
- The Dirichlet priors; α (the topic distribution prior) and η (the word distribution prior)
  - Often set symmetric
  - Typically both are set at 0.1 or 0.01 (the smaller, the fewer topics/words dominate) but this *can be tuned and affect* model performance
  - See Maier et al. (2018) for more discussions

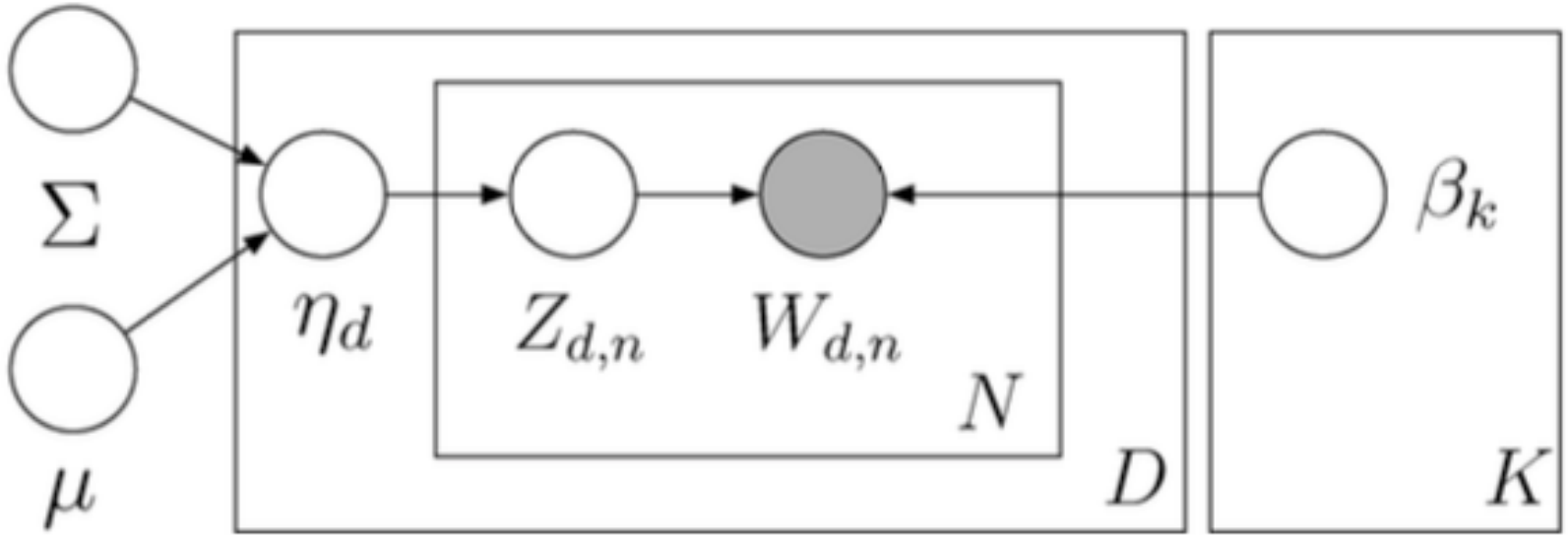
Extensions

Correlated Topic Model (CTM)

- Blei and Lafferty (2005) and (2007)
- A limitation of LDA is the inability to model topic correlation
- “A document about genetics is more likely to also be about disease than X-ray astronomy” (Blei and Lafferty (2007))
- CTM relies on logistic normal distribution, which allows for modeling correlations between topics through a covariance matrix
- The authors report that this can lead to a better model fit (what the model fit means will be discussed soon)

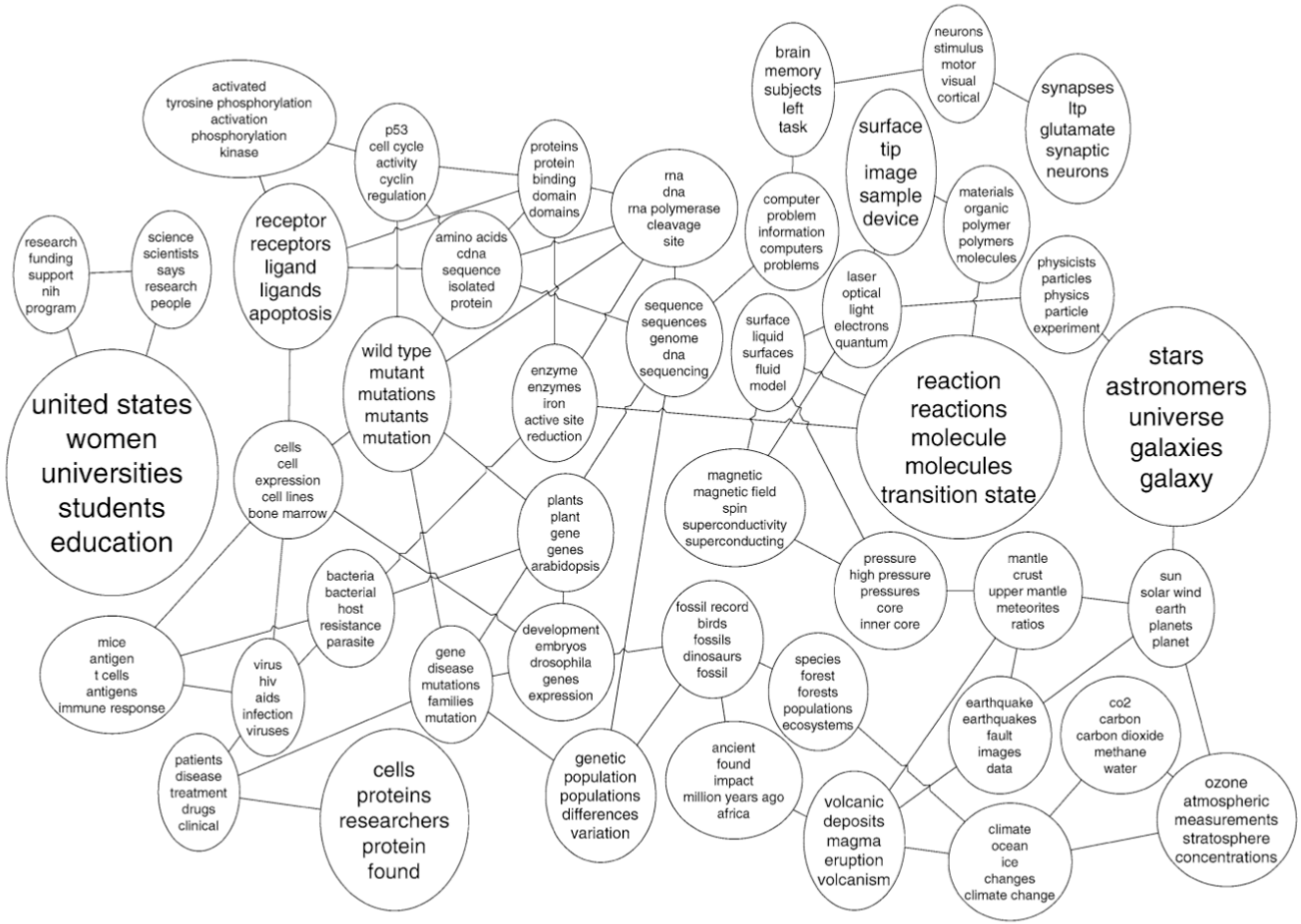
Extensions

Correlated Topic Model (CTM)



Extensions

Correlated Topic Model (CTM)



Extensions

Correlated Topic Model (CTM)

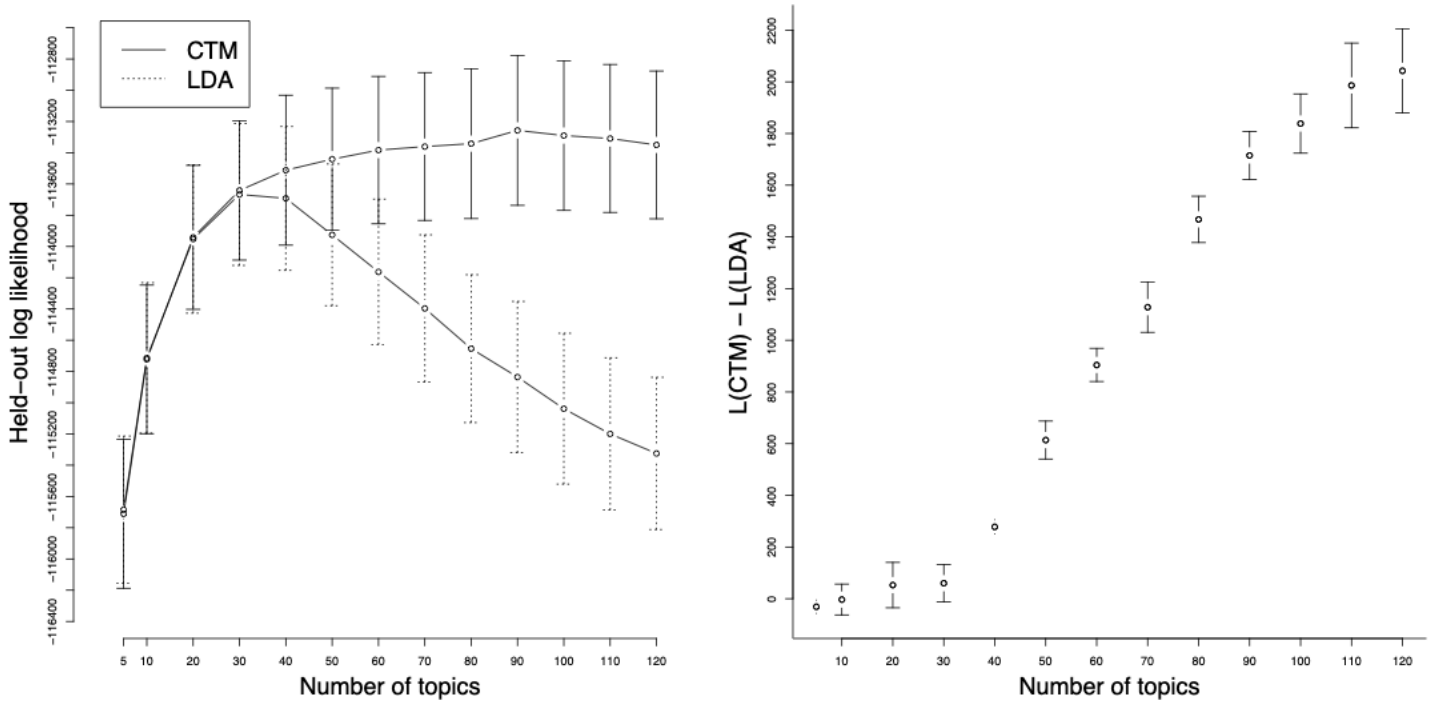


Figure 3: (L) The average held-out probability; CTM supports more topics than LDA. See figure at right for the standard error of the difference. (R) The log odds ratio of the held-out probability. Positive numbers indicate a better fit by the correlated topic model.

Extensions

Structural Topic Model (STM)

- Roberts et al. (2013) and Roberts et al. (2014)
- Key motivation: short documents do not provide the opportunity to observe the correlations between words—key information used to estimate topics
- "Structure": how document-level covariates drives topics
  - E.g., author-level variables (gender, party affiliation, race, etc.)
- Covariates can be modeled to estimate **a) topic prevalence** and **b) topic content**
- Without any covariates, the model reduces to CTM

Extensions

Structural Topic Model (STM)

- Topic prevalence
  - Documents which have similar covariates will tend to talk about the same topics
  - Topic proportions within documents can vary through covariates
  - E.g., social media posts by Republican politicians might have different topic proportions than those posted by Democrats

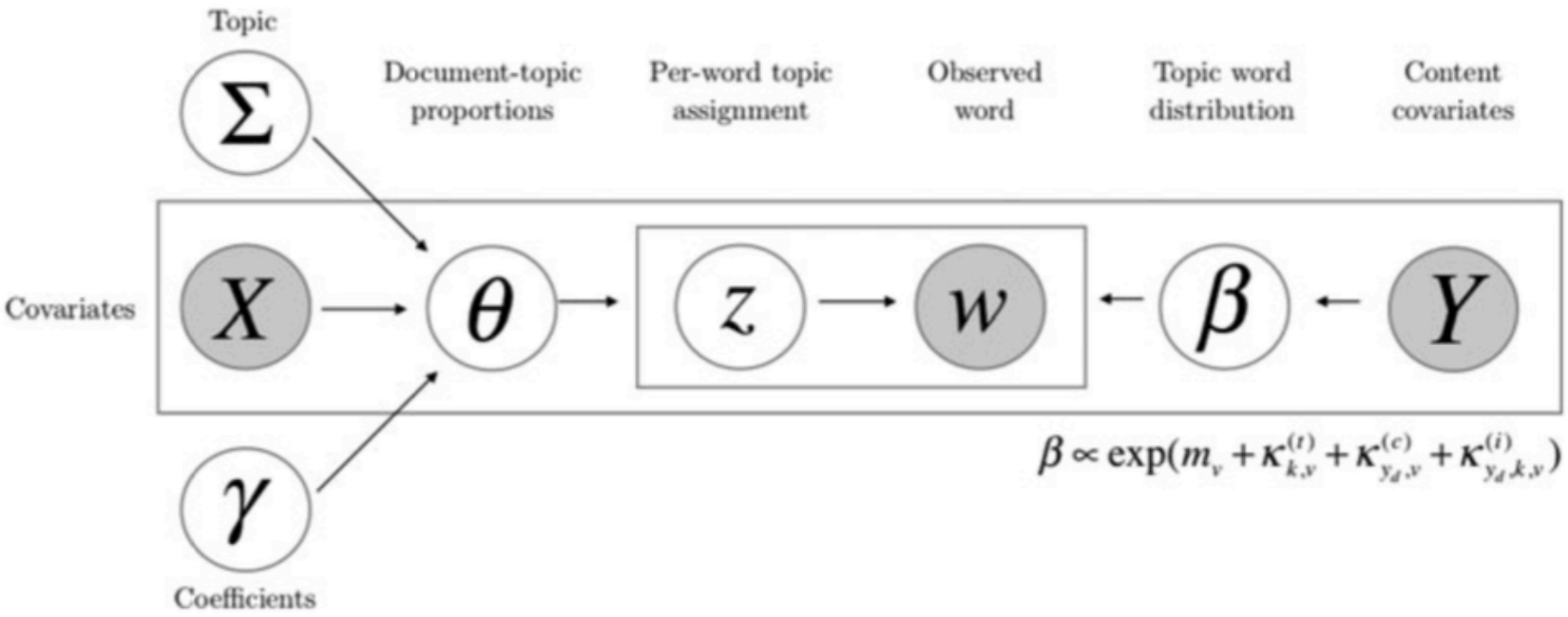
Extensions

Structural Topic Model (STM)

- Topic content
  - Word proportions within topics can vary through covariates
  - Documents which have similar covariates will tend to talk about topics in a similar way
  - E.g., when talking about a health care topic, Republican politicians might use different words than Democrats

Extensions

Structural Topic Model (STM)



Model Selection and Interpretation

Quantitative + qualitative approaches

- Quantitative
  - Perplexity (or held-out likelihood): for some held-out documents, how likely would the model have generated/predicted these documents?
    - Evaluation of predictive power ≠ coherent topics
    - For how they differ, see Chang et al.(2009) and/or this video)
  - Semantic coherence: how likely do the most common words for a topic co-occur in the same documents?
  - Exclusivity: do words with high probability in one topic have low probabilities in others?

Model Selection and Interpretation

Quantitative + qualitative approaches

- Qualitative
  - Examine words that are indicative of a particular topic
    - The most straightforward method for obtaining these words is to select the highest probability words in each topic
  - Careful reading of exemplar texts
    - Select a small subset of the documents with the highest proportion of the document assigned to the particular category under consideration
    - Read those documents to assess their common facets and interrogate whether a particular organization makes sense

Model Selection and Interpretation

Quantitative + qualitative approaches

- No quantitative metric can replace human judgement
- *"The most effective method for assessing model fit is to carefully read documents that are closely associated with particular topics to verify that the semantic concept covered by the topic is reflected in the text."* (Roberts et al. 2016)

Summary

- Topic models help us explore or explicitly measure the thematic structure of a large collection of documents
- Select an algorithm that suits your goal
- Experiment with hyper-parameters, including the number of topics
- Quantitative measures for model selection is useful
- Manual reading plays a crucial role in model seletion and labeling

Guided Coding

- Guided Coding for LDA: LDA with SOTU address data in R
- Materials for STM
  - Official documentation of `stm` package
    - Short demonstration
  - Tutorial with Facebook posts data

Embeddings

Embeddings: themes

Things to be covered

- Word representations
- Word embeddings
- Word2Vec SGNS
- Other models: GloVe, FastText
- Evaluating performance
- Pre-trained vs. self-trained
- Bias reflected in embeddings

Document Representation

We have mostly dealt with document representation

- Document-term matrix (DTM)
  - Count matrix
  - TF-IDF matrix
- Rows represent documents
- Columns represent words (or types)

Document Representation

An example corpus

- Doc 1: “The clever fox cleverly jumps over the lazy dog, showcasing its cleverness.”
- Doc 2: “Magic and mysteries mingle in the wizard’s daily musings, revealing mysteries unknown.”
- Doc 3: “Sunny days bring sunshine and sunsets, making sunny parks the best for sunny strolls.”

Document Representation

An example DFM

Index	clever	jumps	lazy	dog	mysteries	…
Doc 1	3	1	1	1	0	…
Doc 2	0	0	0	0	2	…
Doc 3	0	0	0	0	0	…

Word Representation

How do we represent *words*?

- Vector semantics: a method that represents words in a multi-dimensional space
- The simplest approach: one-hot encoding
  - A vector with one dimension per unique word (i.e., type) in the vocabulary
  - Records 1 for that word and 0 for all the others
  - E.g., `author` = (0, 0, 0, 0, 1, …, 0, 0) (the dimension size is `|V|`)

Word Representation

Limitations

- Semantics
  - Similarity: one-hot(`author`) $\perp$ one-hot(`writer`)
  - Think about the rationale behind lemmatization/stemming: `author` vs. `authors`
- Computation
  - Sparsity (mostly 0s in huge dimensional space: $|V|$)

Word Representation

Term-document matrix (TDM)

- Rows represents words, and columns represent documents
- Similar words have similar vectors because they tend to occur in similar documents (documents are the context)
- E.g., four words in four Shakespeare plays ([JM] Chp. 6)

	As You Like It	Twelfth Night	Julius Caesar	Henry V
battle	1	0	7	13
good	114	80	62	89
fool	36	58	1	4
wit	20	15	2	3

Figure 6.5 The term-document matrix for four words in four Shakespeare plays. The red boxes show that each word is represented as a row vector of length four.

Word Representation

Term-term matrix (TTM)

- Dimension: $|V| \times |V|$
- Each cell records the number of times the row word and the column word co-occur in some context
- Contexts are often a window around the word (e.g., $\pm$ 5)

	aardvark	...	computer	data	result	pie	sugar	...
cherry	0	...	2	8	9	442	25	...
strawberry	0	...	0	0	1	60	19	...
digital	0	...	1670	1683	85	5	4	...
information	0	...	3325	3982	378	5	13	...

Figure 6.6 Co-occurrence vectors for four words in the Wikipedia corpus, showing six of the dimensions (hand-picked for pedagogical purposes). The vector for *digital* is outlined in red. Note that a real vector would have vastly more dimensions and thus be much sparser.

Word Embeddings

What are word embeddings

- *Dense/short* vectors representing word meanings in a multi(low)-dimensional space (d = 50–1000)
  - Word embeddings $\subset$ word vectors
- Words are “embedded” into a common low-dimensional space
- Distributional hypothesis (Joos 1950; Harris 1954)
  - Word that occur in similar contexts tend to have similar meanings
  - “You shall know a word by the company it keeps” (Firth 1957)
- E.g., oculists & eye-doctor: eyes, examine, diagnose, patient, etc.

Word Embeddings

If we have seen

- "… spinach sauteed with garlic over rice …"
- "… chard stems and leaves are delicious …"
- "… collard greens and other salty leafy greens …"

We can guess what `ongchoi` is

- `ongchoi` is delicious sauteed with garlic
- `ongchoi` is superb over rice
- `ongchoi` leaves with salty sauces

Word Embeddings

Why useful?

- Downstream tasks: feature representations
  - Part of speech tagging
  - Named entity recognition
  - Text classification
  - Etc.
- Direct object of interest (to study word usage and meaning)

Word Embeddings

Why useful?

- A measure of word meaning

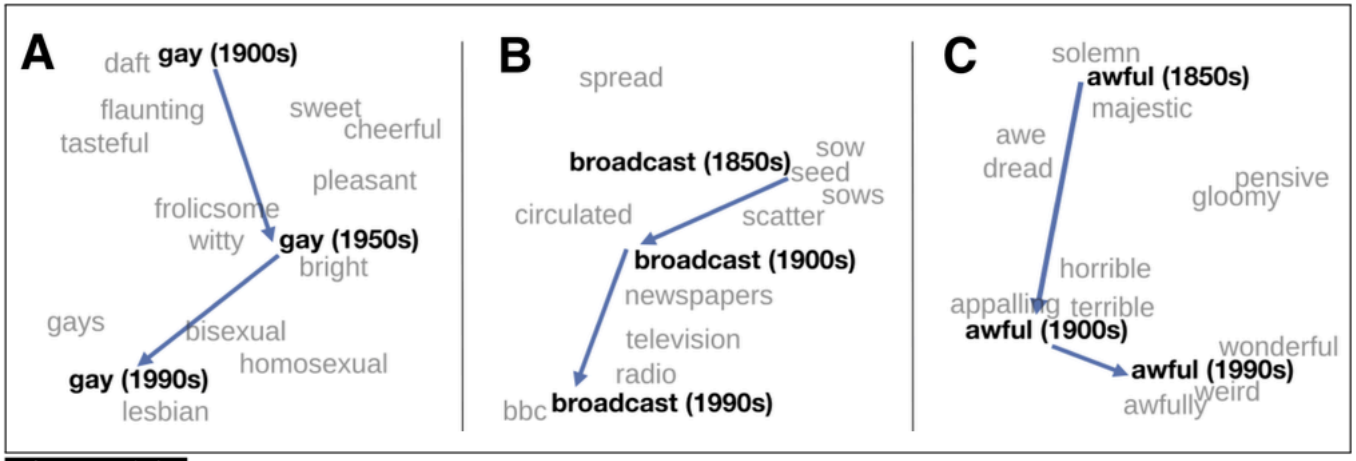
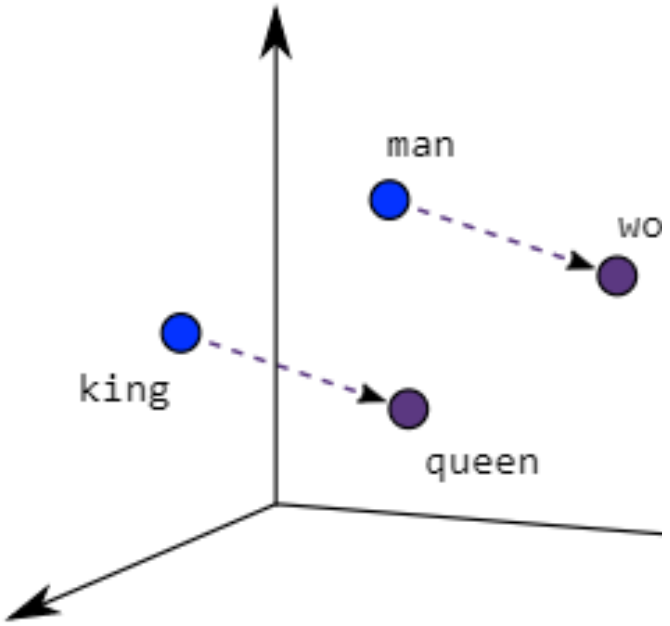


Figure 6.17 A t-SNE visualization of the semantic change of 3 words in English using word2vec vectors. The modern sense of each word, and the grey context words, are computed from the most recent (modern) time-point embedding space. Earlier points are computed from earlier historical embedding spaces. The visualizations show the changes in the word *gay* from meanings related to "cheerful" or "frolicsome" to referring to homosexuality, the development of the modern "transmission" sense of *broadcast* from its original sense of sowing seeds, and the pejoration of the word *awful* as it shifted from meaning "full of awe" to meaning "terrible or appalling" (Hamilton et al., 2016).

Word Embeddings

Why useful?

- Encoding similarity
  - For similar words, their embeddings point in similar directions ($\Longleftrightarrow$ one-hot encodings)
    - E.g., $\vec{e_{author}} \propto \vec{e_{writer}}$
  - Similarity in relations ("vector arithmetic")
    - E.g., `king` - `man` + `woman` $\approx$ `queen` (Mikolov et al. 2013)



Word Embeddings

Why useful?

- Automatic generalization
  - Information retrieval
    - E.g., identifying academic papers about literacy in the digital age
      - Seed keywords: `digital literacy`, `information literacy`, etc.
      - Identifying similar words using word embeddings: `e-literacy`, `technology proficiency`, etc.
  - Dictionaries combined with word embeddings (Garten et al. 2018; Osnabrugge et al. 2021)
    - E.g., keywords for “anger”
    - The centroid of embeddings for terms signalling “anger”
    - The centroid of the embeddings of the words in a document

Estimating Word Embeddings

Word2Vec (Mikolov et al. 2013a; Mikolov et al. 2013b)

- Skipgram and CBOW (Continuous Bag Of Words)
  - Skipgram: given a target word, predict the context words (e.g., $\pm$ 5)
  - CBOW: given the context words, predicts the target word
- SGNS (skip-gram with negative sampling)
  - Given a pair of a target word and another word c, what is the probability of c being the actual context word (c_{pos})?

Estimating Word Embeddings

Word2Vec SGNS

- Self-supervision: “+” if in context, otherwise “-”
  - L: the size of the context window
  - K: the proportion of positive (or context) to (randomly selected) negative examples (recommended K: 2–5 for big, 5–20 for small data)

```
... lemon,  a [tablespoon of apricot jam,      a] pinch ...
               c1         c2    w     c3       c4
```

positive examples +

w	c_{pos}
apricot	tablespoon
apricot	of
apricot	jam
apricot	a

negative examples -

w	c_{neg}	w	c_{neg}
apricot	aardvark	apricot	seven
apricot	my	apricot	forever
apricot	where	apricot	dear
apricot	coaxial	apricot	if

Estimating Word Embeddings

Word2Vec SGNS

- Task
  - Train a binary classifier that computes Pr(+|w, c)
  - Pr(+|w, c) = $\sigma(\overrightarrow{e_w} \cdot \overrightarrow{e_c})$
- Goal
  - Maximize the similarity of the target-context pairs (w, c_{pos})
  - Minimize the similarity of the target-non-context pairs (w, c_{neg})

Estimating Word Embeddings

Word2Vec SGNS

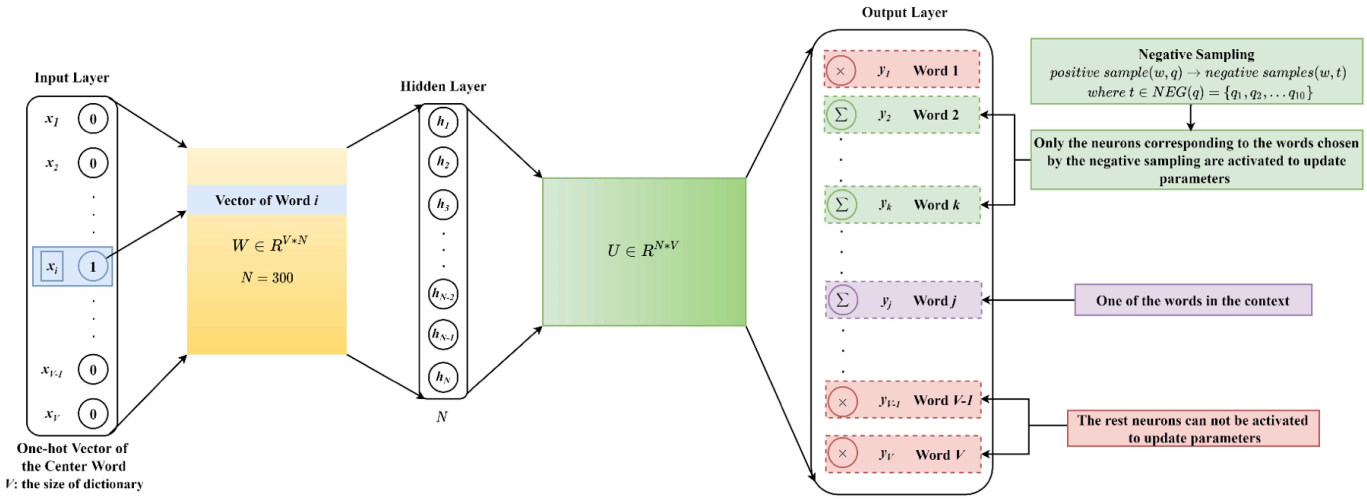
- Optimization: minimize the cross-entropy loss function using (stochastic) gradient descent

$$L_{CE} = -log[P(+|w, c_{pos}) \prod_{i=1}^{k} P(-|w, c_{neg_i})]$$

Estimating Word Embeddings

Word2Vec SGNS

- The neural network for SG(NS) (source: link)



Estimating Word Embeddings

Word2Vec SGNS

- Detailed treatments of SG and SGNS
  - SG: link
  - SGNS: link

Other Approaches

There are many different approaches how one could obtain word embeddings

- GloVe (Global Vectors for Word Representation) (Pennington et al. 2014)
- FastText (Bojanowski et al. 2017)
  - Subword-level model
    - Each word is represented as itself along with a bag of constituent n-grams, with boundary symbols < and >
    - E.g., $\overrightarrow{e_{apple}} = \overrightarrow{e_{<ap}} + \overrightarrow{e_{app}} + \overrightarrow{e_{ppl}} + \overrightarrow{e_{ple}} + \overrightarrow{e_{le>}} + \overrightarrow{e_{<apple>}}$
  - Deals with OOV (out of vocabulary), rare words, and typos (e.g., `appple`) efficiently

Evaluating Performance

How to evaluate word embeddings?

- Extrinsic validation (straightforward)
  - Performance on a downstream NLP task (PoS tagging, NER, etc.)
- Intrinsic validation
  - Whether the embeddings are able to capture similarities between words
  - Computer science as well as social sciences (e.g., Rodriguez and Spirling (2022))

Various Pre-trained Embeddings

General embeddings

- Word2Vec: link ("GoogleNews-vectors-negative300.bin.gz")
- GloVe: link
- FastText: link

(A few examples from many) domain-specific embeddings

- Trained on 19th-century British newspapers: link
- Trained on tweets: link ("glove.twitter.27B.zip")

Pre-trained vs. Self-trained

A few circumstances that require self-training

- Temporal changes in language
  - Known as diachronic/dynamic embeddings (e.g., Kim and Jeon 2023)
- Group-specific language (e.g., Democrats vs. Republicans)
- Domain-specific language (e.g., Case2Vec)
- Low resource languages

Pre-trained vs. Self-trained

Which one captures word similarity better?

- Experiment by Rodriguez and Spirling (2022)
  - Provide crowd workers (human annotators) on MTurk 10 political words and ask them to produce a set of ten nearest neighbors (“human”)
  - They then use these same words to generate machine nearest neighbors by finding the most cosine similar vector using word embeddings (“local” or pre-trained “GloVe”)
  - They then have a separate set of humans (human judge, N = 135) look at a prompt word and two possible nearest neighbors (“human” vs. “local” vs. pre-trained “GloVe”)

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

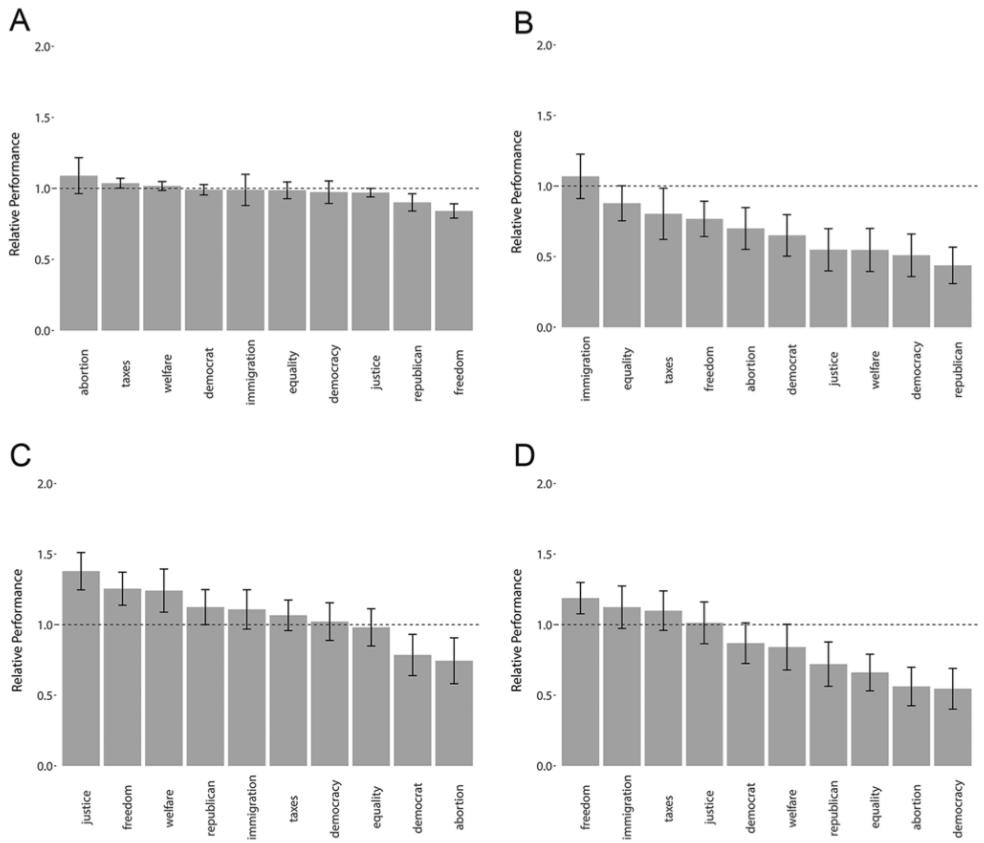


Figure 2. Human preferences: Turing assessment. *A*, Candidate: local 48–300; baseline: local 6–300. *B*, Candidate: local 6–300; baseline: human. *C*, Candidate: GloVe; baseline: local 6–300. *D*, Candidate: GloVe; baseline: human.

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

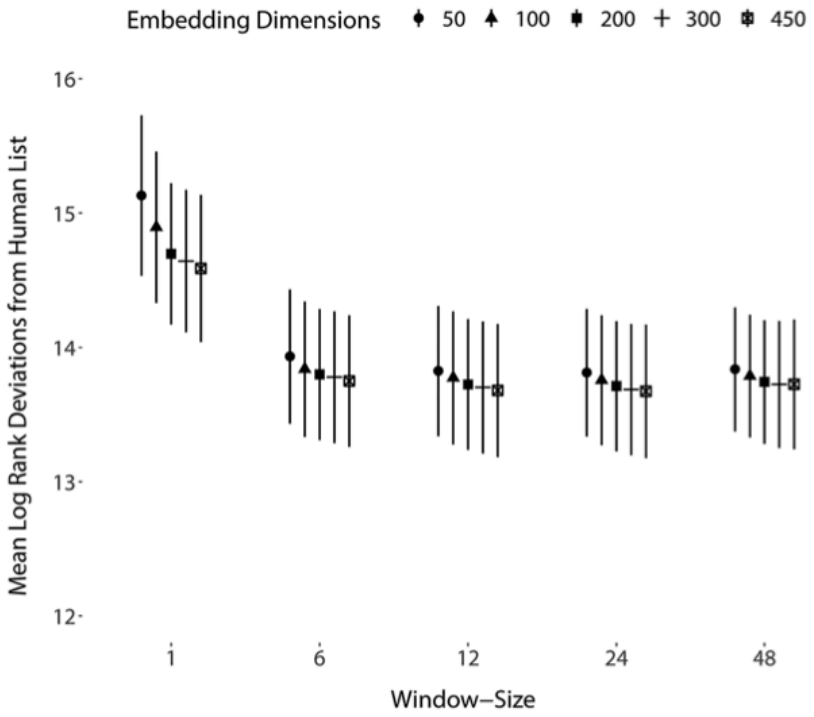


Figure 3. Human preferences: log rank deviations: complex models come closer to "human" assessments, but medium-size models are almost as good as very large ones.

Pre-trained vs. Self-trained

Findings from Rodriguez and Spirling (2022)

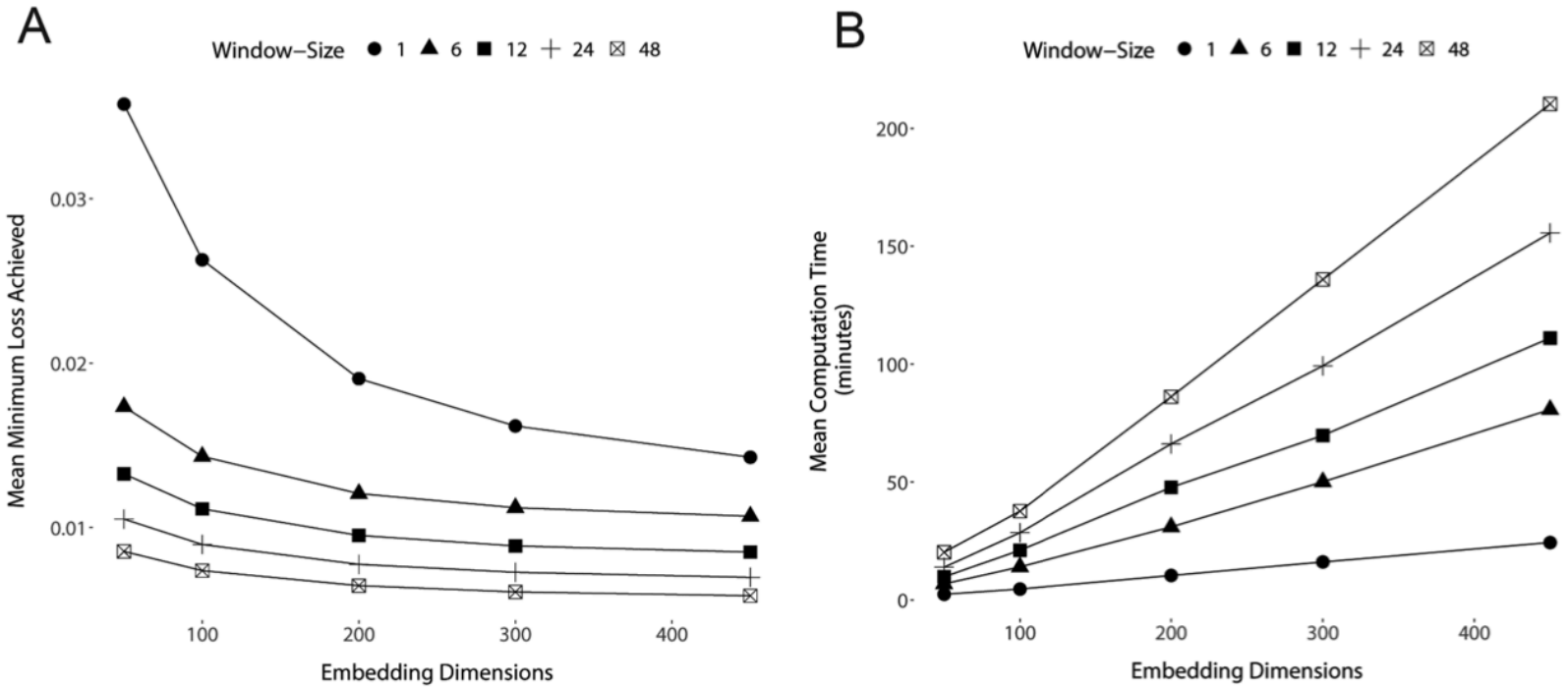


Figure 4. Technical criteria: larger models fit better but take longer to compute. *A*, Mean minimum loss achieved. *B*, Computation time (minutes)

Pre-trained vs. Self-trained

Lessons

- Popular pre-trained word embeddings “perform” at a level close to–or even surpassing–both human annotators and locally fit models with various configurations

Caveat

- A specific meaning of “perform”
- The results are based on a limited set of corpora (political in nature)

Bias Reflected in Human Language

Bolukbasi et al. (2016)

- Pretrained Word2Vec embeddings
  - E.g., ‘computer programmer’ - ‘man’ + ‘woman’ = ‘homemaker’

Gender stereotype *she-he* analogies.

sewing-carpentry	register-nurse-physician	housewife-shopkeeper
nurse-surgeon	interior designer-architect	softball-baseball
blond-burly	feminism-conservatism	cosmetics-pharmaceuticals
giggle-chuckle	vocalist-guitarist	petite-lanky
sassy-snappy	diva-superstar	charming-affable
volleyball-football	cupcakes-pizzas	hairdresser-barber

Gender appropriate *she-he* analogies.

queen-king	sister-brother	mother-father
waitress-waiter	ovarian cancer-prostate cancer	convent-monastery

Figure 2: **Analogy examples**. Examples of automatically generated analogies for the pair *she-he* using the procedure described in text. For example, the first analogy is interpreted as *she*:*sewing* :: *he*:*carpentry* in the original w2vNEWS embedding. Each automatically generated analogy is evaluated by 10 crowd-workers are to whether or not it reflects gender stereotype. Top: illustrative gender stereotypic analogies automatically generated from w2vNEWS, as rated by at least 5 of the 10 crowd-workers. Bottom: illustrative generated gender-appropriate analogies.

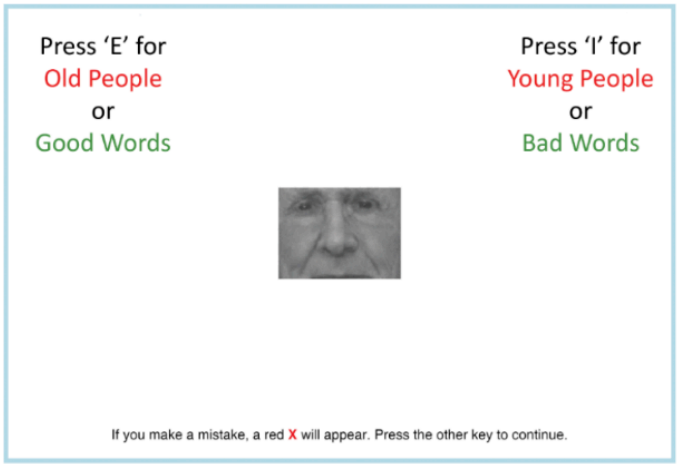
Bias Reflected in Human Language

Let’s try it in Korean: link

Bias Reflected in Human Language

Caliskan et al. (2017)

- Replicated evidence of bias from IATs (Implicit Association Test) using pre-trained GloVe vectors and cosine similarity
- African American (European-American) names have higher cosine similarity with unpleasant (pleasant) words



Summary

- Word embeddings can be used
  - Not only for downstream NLP tasks
  - But also for studying word usage/meanings
- Popular pre-trained embeddings appear to match (or outperform) locally trained embeddings in terms of capturing word similarity
- Word embeddings reflect bias in various aspects

Guided Coding

Training Word2Vec and FastText in Python