Clustering in Natural Language Processing & Text Mining

CpE 6330 Deep Dives

Stuart Miller

Master's Student in Computer Engineering

MISSOURI S&T

Outline

- **Preliminary Steps**
  - Preprocessing
  - Stemming Algorithms
  - Bag of Word Model
- Term Frequency – Inverse Document Frequency
  - Zipf's Law
- Latent Dirichlet Allocation
- Latent Semantic Analysis
- Conclusions

Text Preprocessing

```
38x5 string array

    "a"          "doesn't"    "in"         "she'll"       "we're"
    "about"      "doesnt"     "instead"    "shell"        "we've"
    "above"      "doing"      "into"       "should"       "weve"
    "across"     "done"       "is"         "since"        "were"
    "after"      "don't"      "isn't"      "so"           "what"
    "all"        "dont"       "isnt"       "some"         "what's"
    "along"      "during"     "it"         "such"         "whats"
    "also"       "each"       "it'll"      "than"         "when"
    "am"         "either"     "itll"       "that"         "when's"
    "an"         "for"        "it's"       "the"          "whens"
    "and"        "from"       "its"        "their"        "where"
    "any"        "given"      "let's"      "them"         "whether"
    "are"        "had"        "lets"       "then"         "which"
    "aren't"     "has"        "may"        "there"        "while"
    "arent"      "have"       "me"         "therefore"    "who"
    "as"         "having"     "more"       "these"        "who'll"
    "at"         "he"         "most"       "they"         "wholl"
    "be"         "he'd"       "much"       "this"         "who's"
    "because"    "hed"        "must"       "those"        "whos"
    "been"       "he'll"      "my"         "through"      "who've"
    "before"     "her"        "no"         "to"           "whove"
    "being"      "here"       "not"        "too"          "will"
    "between"    "hers"       "now"        "towards"      "with"
    "both"       "him"        "of"         "under"        "within"
    "but"        "himself"    "on"         "until"        "without"
    "by"         "his"        "one"        "us"           "won't"
    "can"        "how"        "only"       "use"          "would"
    "can't"      "how's"      "or"         "used"         "wouldn't"
    "cant"       "hows"       "other"      "uses"         "you"
    "cannot"     "however"    "our"        "using"        "you'd"
    "could"      "i"          "out"        "very"         "youd"
    "couldn't"   "i'd"        "over"       "want"         "you'll"
    "couldnt"    "i'll"       "said"       "was"          "youll"
    "did"        "i'm"        "says"       "wasn't"       "you're"
    "didn't"     "im"         "see"        "wasnt"        "youre"
    "didnt"      "i've"       "she"        "we"           "you've"
    "do"         "ive"        "she'd"      "we'd"         "youve"
    "does"       "if"         "shed"       "we'll"        "your"
```

MATLAB's default stopwords

- Capitalization
  - Change all letters to lowercase
- Misc. Removal
  - Remove numbers, punctuation, symbols, and various other unnecessary entities
- Stopwords
  - Remove all insignificant words
- Tokenization
  - Split into a list of individual words
- Stemming

Stemming Algorithms

- Aims to reduce all similar words to a common form
  - Ex. “revival” -> “reviv”, “defensible" -> “defens”
- First proposed as an aid for a library cataloguing system at MIT
  - J. B. Lovins, “Development of a Stemming Algorithm”, Mechanical Translation and Computational Linguistics, vol. 11, nos. 1 and 2, pp. 22-31, 1968
- Porter stemmer is the current standard
  - M. F. Porter, “An Algorithm for Suffix Stripping”, Program, vol. 14, no. 3, pp. 130-137, 1980
  - Freely available at https://tartarus.org/martin/PorterStemmer/
  - Open source encoding for most programming languages

MISSOURI S&T

Examples of Stemming Issues

> Complexities of the English language propose many problems

produc | er : product | ion	invert | ed : invers | ion
induc | ed : induct | ion	adher | e : adhes | ion
induct | ed : induct | ion	register | ing : registr | ation
consum | ed : consumpt | ion	resolv | ed : resolut | ion
absorb | ing : absorpt | ion	admitt | ed : admiss | ion
attend | ing : attent | ion	circl | e : circul | ar
expand | ing : expans | ion	matrix | : matric | es
respond | : respons | ive	lattic | e : lattic | es
exclud | e : exclus | ion	index | : indic | es
collid | ing : collis | ion	hypothes | ized : hypothet | ical
analys | is : analyt | ic	

> A stemming algorithm essentially just becomes a set of rules to accommodate all these exceptions

MISSOURI S&T

Lovins Stemming Algorithm

Condition Codes for Context-sensitive Rules Associated with Certain Endings

- A . . . No restrictions on stem
- B . . . Minimum stem length = 3
- C . . . Minimum stem length = 4
- D . . . Minimum stem length = 5
- E . . . Do not remove ending after *e*
- F . . . Minimum stem length = 3 and do not remove ending after *e*
- G . . . Minimum stem length = 3 and remove ending only after *f*
- H . . . Remove stem ending only after *t* or *ll*
- I . . . Do not remove ending after *o* or *e*
- J . . . Do not remove ending after *a* or *e*
- K . . . Minimum stem length = 3 and remove ending only after *l*, *i*, or *uαe* (where *α* stands for any letter)
- L . . . Do not remove ending after *u*, *x*, or *s*, unless *s* follows *o*
- M . . . Do not remove ending after *a*, *c*, *e*, or *m*
- N . . . Minimum stem length = 4 after *sαα*, elsewhere = 3
- O . . . Remove ending only after *l* or *i*
- P . . . Do not remove ending after *c*
- Q . . . Minimum stem length = 3 and do not remove ending after *l* or *n*
- R . . . Remove ending only after *n* or *r*
- S . . . Remove ending only after *dr* or *t*, unless *t* follows *t*
- T . . . Remove ending only after *s* or *t*, unless *t* follows *o*
- U . . . Remove ending only after *l*, *m*, *n*, or *r*
- V . . . Remove ending only after *c*
- W . . . Do not remove ending after *s* or *u*
- X . . . Remove ending only after *l*, *i*, or *uαe*
- Y . . . Remove ending only after *in*
- Z . . . Do not remove ending after *f*
- AA . . . Remove ending only after *d*, *f*, *ph*, *th*, *l*, *er*, *or*, *es*, or *t*
- BB . . . Minimum stem length = 3 and do not remove ending after *met* or *ryst*
- CC . . . Remove ending only after *l*

Transformational Rules Used in Recoding Stem Terminations

- 1 . . . Remove one of double *b*, *d*, *g*, *l*, *m*, *n*, *p*, *r*, *s*, *t*
- 2 . . . iev → ief
- 3 . . . uct → uc
- 4 . . . umpt → um
- 5 . . . rpt → rb
- 6 . . . urs → ur
- 7 . . . istr → ister
- 7*a* . . metr → meter
- 8 . . . olv → olut
- 9 . . . ul → l except following *a*, *i*, *o*
- 10 . . . bex → bic
- 11 . . . dex → dic
- 12 . . . pex → pic
- 13 . . . tex → tic
- 14 . . . ax → ac
- 15 . . . ex → ec
- 16 . . . ix → ic
- 17 . . . lux → luc
- 18 . . . uad → uas
- 19 . . . vad → vas
- 20 . . . cid → cis
- 21 . . . lid → lis
- 22 . . . erid → eris
- 23 . . . pand → pans
- 24 . . . end → ens except following *s*
- 25 . . . ond → ons
- 26 . . . lud → lus
- 27 . . . rud → rus
- 28 . . . her → hes except following *p*, *t*
- 29 . . . mit → mis
- 30 . . . end → ens except following *m*
- 31 . . . ert → ers
- 32 . . . et → es except following *n*
- 33 . . . yt → ys
- 34 . . . yz → ys

MISSOURI S&T

Porter Stemming Algorithm

```
Step 1a

    SSES -> SS                          caresses  ->  caress
    IES  -> I                           ponies    ->  poni
                                        ties      ->  ti
    SS   -> SS                          caress    ->  caress
    S    ->                             cats      ->  cat

Step 1b

    (m>0) EED -> EE                     feed      ->  feed
                                        agreed    ->  agree
    (*v*) ED  ->                        plastered ->  plaster
                                        bled      ->  bled
    (*v*) ING ->                        motoring  ->  motor
                                        sing      ->  sing
```

```
Step 2

    (m>0) ATIONAL ->  ATE           relational     ->  relate
    (m>0) TIONAL  ->  TION          conditional    ->  condition
                                    rational       ->  rational
    (m>0) ENCI    ->  ENCE          valenci        ->  valence
    (m>0) ANCI    ->  ANCE          hesitanci      ->  hesitance
    (m>0) IZER    ->  IZE           digitizer      ->  digitize
    (m>0) ABLI    ->  ABLE          conformabli    ->  conformable
    (m>0) ALLI    ->  AL            radicalli      ->  radical
    (m>0) ENTLI   ->  ENT           differentli    ->  different
    (m>0) ELI     ->  E             vileli        - >  vile
    (m>0) OUSLI   ->  OUS           analogousli    ->  analogous
    (m>0) IZATION ->  IZE           vietnamization ->  vietnamize
    (m>0) ATION   ->  ATE           predication    ->  predicate
    (m>0) ATOR    ->  ATE           operator       ->  operate
    (m>0) ALISM   ->  AL            feudalism      ->  feudal
    (m>0) IVENESS ->  IVE           decisiveness   ->  decisive
    (m>0) FULNESS ->  FUL           hopefulness    ->  hopeful
    (m>0) OUSNESS ->  OUS           callousness    ->  callous
    (m>0) ALITI   ->  AL            formaliti      ->  formal
    (m>0) IVITI   ->  IVE           sensitiviti    ->  sensitive
    (m>0) BILITI  ->  BLE           sensibiliti    ->  sensible
```

```
Step 3

    (m>0) ICATE ->  IC              triplicate     ->  triplic
    (m>0) ATIVE ->                  formative      ->  form
    (m>0) ALIZE ->  AL              formalize      ->  formal
    (m>0) ICITI ->  IC              electriciti    ->  electric
    (m>0) ICAL  ->  IC              electrical     ->  electric
    (m>0) FUL   ->                  hopeful        ->  hope
    (m>0) NESS  ->                  goodness       ->  good
```

```
Step 4

    (m>1) AL    ->                  revival        ->  reviv
    (m>1) ANCE  ->                  allowance      ->  allow
    (m>1) ENCE  ->                  inference      ->  infer
    (m>1) ER    ->                  airliner       ->  airlin
    (m>1) IC    ->                  gyroscopic     ->  gyroscop
    (m>1) ABLE  ->                  adjustable     ->  adjust
    (m>1) IBLE  ->                  defensible     ->  defens
    (m>1) ANT   ->                  irritant       ->  irrit
    (m>1) EMENT ->                  replacement    ->  replac
    (m>1) MENT  ->                  adjustment     ->  adjust
    (m>1) ENT   ->                  dependent      ->  depend
    (m>1 and (*S or *T)) ION ->     adoption       ->  adopt
    (m>1) OU    ->                  homologou      ->  homolog
    (m>1) ISM   ->                  communism      ->  commun
    (m>1) ATE   ->                  activate       ->  activ
    (m>1) ITI   ->                  angulariti     ->  angular
    (m>1) OUS   ->                  homologous     ->  homolog
    (m>1) IVE   ->                  effective      ->  effect
    (m>1) IZE   ->                  bowdlerize     ->  bowdler
```

```
Step 5a

    (m>1) E     ->                  probate        ->  probat
                                    rate           ->  rate
    (m=1 and not *o) E ->           cease          ->  ceas

Step 5b

    (m > 1 and *d and *L) -> single letter
                                    controll       ->  control
                                    roll           ->  roll
```

MISSOURI S&T

Bag-of-Words Model

- A unifying data structure used across natural language processing to represent textual information
- First referred to as a ”bag” in 1954
  - Z. S. Harris, “Distributional Structure”, Word, vol. 10, issues 2 and 3, pp. 146-162, 1954
- Consists of…
  - Vocabulary (list of words)
  - Frequency list
- Features…
  - Orderless
  - Easy to count, sort, and filter frequencies

“It was the best of times,
it was the worst of times,
it was the age of wisdom,
it was the age of foolishness,”

it	was	the	best	of	times	worst	age	wisdom	foolishness
4	4	4	1	4	2	1	2	1	1

MISSOURI S&T

Outline

- Preliminary Steps
  - Preprocessing
  - Stemming Algorithms
  - Bag of Word Model
- **Term Frequency – Inverse Document Frequency**
  - Zipf’s Law
- Latent Dirichlet Allocation
- Latent Semantic Analysis
- Conclusions

Term Frequency-Inverse Document Frequency

- Proposed in 1972 as part of work done at the Cambridge Computing Laboratory
  - K. S. Jones, “A Statistical Interpretation Of Term Specificity And Its Application In Retrieval”, Journal of Documentation, vol. 28, issue 1, pp. 11-21, 1972
- Reflects the importance of a term within a collection of documents
- Heavily used by Internet search engines
- $tfidf = tf(t,d) \cdot idf(t,d)$

MISSOURI S&T

Term Frequency

- $tf(t,d) = \frac{t \in d}{\forall t \in d}$
  - number of times a word appears over total words in document
- Often scaled proportionally

Variants of term frequency (TF) weight

weighting scheme	TF weight
binary	$0, 1$
raw count	$f_{t,d}$
term frequency	$f_{t,d} \Big/ \sum_{t' \in d} f_{t',d}$
log normalization	$1 + \log(f_{t,d})$
double normalization 0.5	$0.5 + 0.5 \cdot \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$
double normalization K	$K + (1 - K) \frac{f_{t,d}}{\max_{\{t' \in d\}} f_{t',d}}$

MISSOURI S&T

Inverse Document Frequency

- $idf(t,d) = \log \frac{D}{\{d \in D: t \in d\}}$
- De-emphasizes unimportant words (i.e. “the”)
- The commonality of a term within all documents
- Usually the logarithm of documents over documents containing the term

Variants of inverse document frequency (IDF) weight

weighting scheme	IDF weight ($n_t = \|\{d \in D : t \in d\}\|$)
unary	1
inverse document frequency	$\log \frac{N}{n_t} = -\log \frac{n_t}{N}$
inverse document frequency smooth	$\log \left(1 + \frac{N}{n_t}\right)$
inverse document frequency max	$\log \left(\frac{\max_{\{t' \in d\}} n_{t'}}{1 + n_t}\right)$
probabilistic inverse document frequency	$\log \frac{N - n_t}{n_t}$

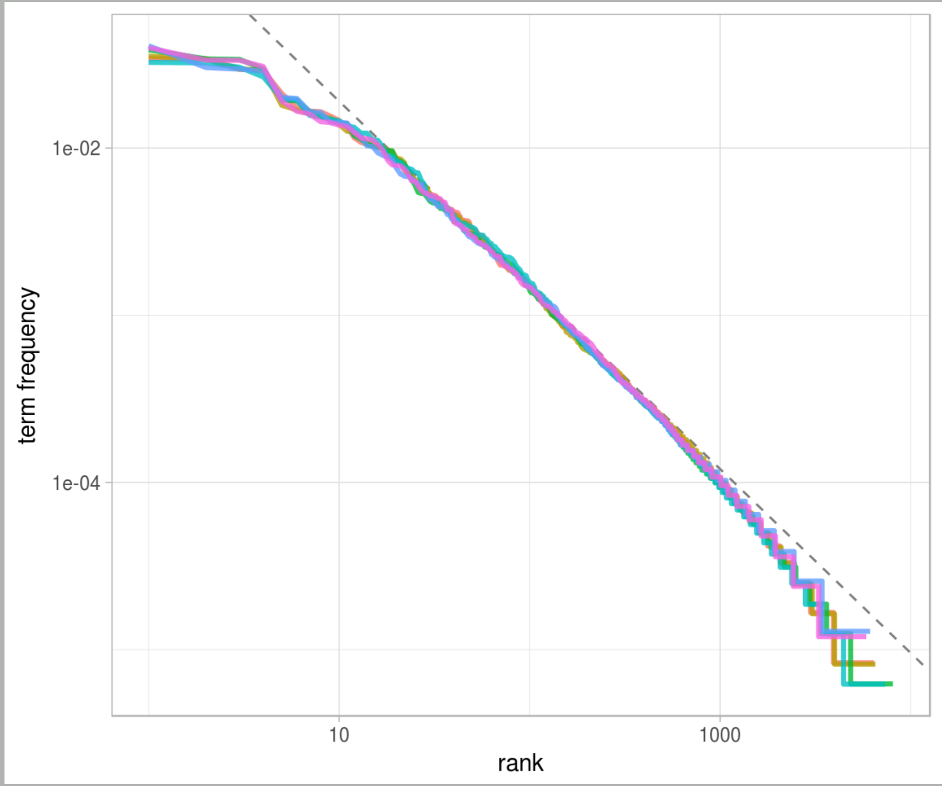
MISSOURI S&T

Zipf’s Law and Term Frequency

- $freq \propto \frac{1}{rank}$
- Frequency that a word appears is inversely proportional to its rank
- Justification for
- Zipf’s work as a linguist summarized into **Zipf’s Law** postmortem
  - D. M. W. Powers, “Applications and explanations of Zipf's law”, Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning, vol. 28, issue 1, pp. 151-160, 1998
- Holds true for almost any collection of text, in any language

MISSOURI S&T

Zipf’s Law Illustrated



Source: https://www.tidytextmining.com/tfidf.html#zipfs-law

How do we use all this?

- Quick and dirty method for NLP
- Can easily group or partition documents relevant to a term
  - Term relevance = one dimension, easy to analyze
  - Limited to one term at a time
  - Not too useful for looking for overall document similarity
- Will make a return later
  - Stay tuned for it’s use in **latent semantic analysis**!

Outline

- Preliminary Steps
  - Preprocessing
  - Stemming Algorithms
  - Bag of Word Model
- Term Frequency – Inverse Document Frequency
  - Zipf’s Law
- **Latent Dirichlet Allocation**
- Latent Semantic Analysis
- Conclusions

MISSOURI S&T

Latent Dirichlet Allocation

- Independently discovered by two groups
  - D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent Dirichlet Allocation,” Journal of Machine Learning Research, vol. 3, pp. 993–1022, Jan. 2003.
  - J. K. Pritchard, M. Stephens, and P. Donnelly, “Inference of population structure using multilocus genotype data,” Genetics, vol. 155, no. 2, pp. 945–959, Jun. 2000.
- Aims to assign documents to **topics**
  - Need to decide on number of topics ahead of time
  - Topics not necessarily strongly defined
- Addresses shortcomings of TF-IDF
  - TF-IDF fails to reveal “big picture” trends
  - TF-IDF limited to just singular words

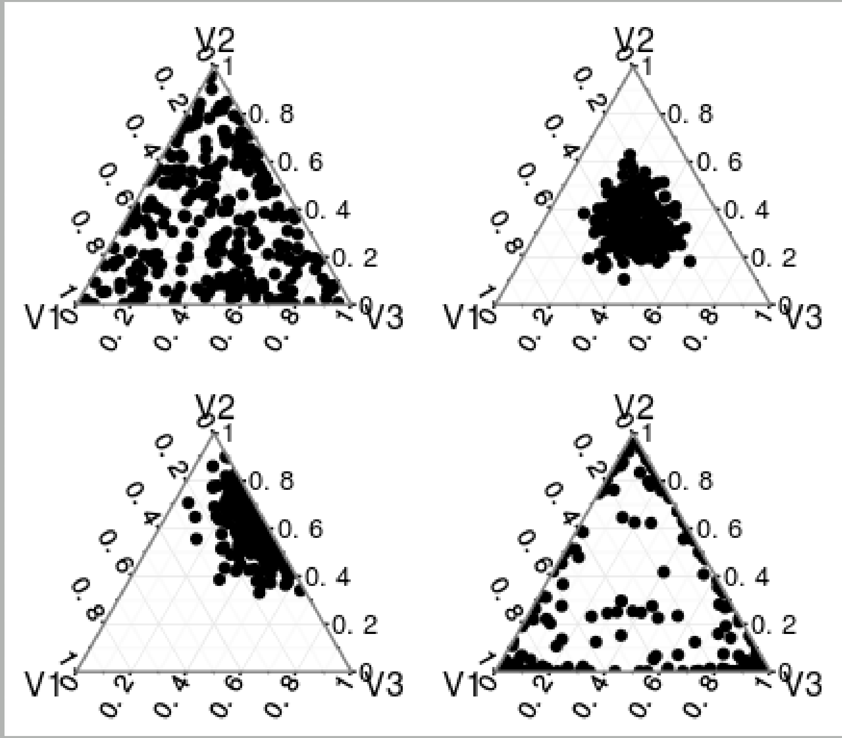
Dirichlet Distribution

- $$P(p|\alpha) = \underbrace{\frac{\Gamma(\sum_{k=0}^{K-1} \alpha_k)}{\prod_{k=0}^{K-1} \Gamma(\alpha_k)}}_{\text{normalizing constant}} \underbrace{\prod_{k=0}^{K-1} p_k^{\alpha_k - 1}}_{\text{probability density function}}$$
  - Where k is the number of topics
  - Where $p_0 \ldots p_k$ is the multinomial distribution
  - Where $\alpha_0 \ldots a_k$ is the Dirichlet concentration parameter
    - $\alpha \geq 1$: concentrated at center
    - $\alpha = 1$: uniform distribution
    - $\alpha \leq 1$: concentrated at edges
- Samples from a **probability simplex**
  - Space of number that add up to one (ex. 0.1, 0.1, 0.8)
  - A probability distribution over all possible probability distributions

Dirichlet Distributions with $k = 3$

$\alpha_1 = 1$
$\alpha_2 = 1$
$\alpha_3 = 1$

$\alpha_1 = 10$
$\alpha_2 = 10$
$\alpha_3 = 10$



$\alpha_1 = 1$
$\alpha_2 = 10$
$\alpha_3 = 5$

$\alpha_1 = 0.2$
$\alpha_2 = 0.2$
$\alpha_3 = 0.2$

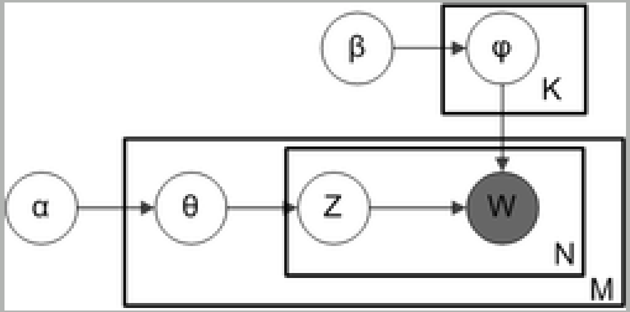
Source: https://stats.stackexchange.com/a/244946

How Does this Relate to Text Analysis?

- Observable variable is w; the word frequency
- Other variables are **latent** (inferred by algorithm)
- Output is a distribution of topics across a document
- **Basic Goal:** To continuously tune the Dirichlet concentration parameters in Θ and Φ varying the assignments of words to topics until a **steady state** is achieved
  - **Steady state:** if word a and word b are both assigned to topic x, then word a and word b will frequently appear together

LDA Mechanics

- Plate model →



Source: Blei "Latent Dirichlet Allocation"

- Variables
  - α is the starting Dirichlet parameter for the per-document topic distribution
  - β is the starting Dirichlet parameter for the per-topic word distribution
  - Θ_m is the topic distribution for document m
    - $\Theta_1 \ldots \Theta_M$ are M-dimensional vectors containing the Dirichlet parameter for each
  - Φ_k is the word distribution for topic k
    - $\Phi_1 \ldots \Phi_K$ are W-dimensional vectors containing the Dirichlet parameter for each
  - z_{mn} is the topic for the n^{th} word in document m
  - w_{mn} is the specific word

Bayesian Inference

- Statistical learning method
  - Update the probability for a hypothesis as more evidence or information becomes available
- $P(H|E) = \frac{P(E|H) \cdot P(H)}{P(E)}$
  - Probability of **H**ypothesis given **E**vidence
- Methods
  - Variational Bayesian sampling of the Posterior Distribution
    - Used in Blei's original paper
  - Gibbs Sampling
    - Fairly common
  - Collapsed Gibbs Sampling
    - Faster version for large data sets

MISSOURI S&T

Collapsed Gibbs Sampling

> $p(z = t|w) \propto \frac{\alpha_t \beta}{n_{\cdot|t} + V\beta} + \frac{n_{t|d}\ \beta}{n_{\cdot|t} + V\beta} + \frac{n_{w|t}\ (\alpha_t + n_{t|d})}{n_{\cdot|t} + V\beta}$

- Where $P(z = t|w)$ refers to the probability of topic z in document d given word w is of topic t
- Where α is the vector of all prior **topic** distribution parameters of a particular **document**
- Where β is the vector of all prior **word** distribution parameters of a particular **topic**
- Where V is the number of words in the vocabulary
- Where n is a counting variable (i.e. $n_{\cdot|t}$ is all words that fit topic t)

> L. Yao, D. Mimno, and A. McCallum, "Efficient Methods for Topic Model Inference on Streaming Document Collections", Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 937-946, 2009

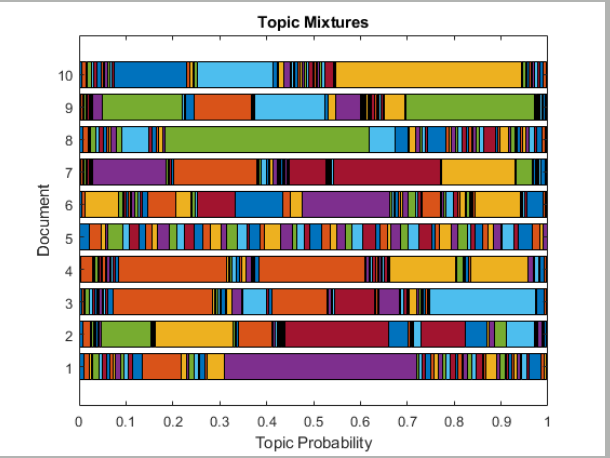
Collapsed Gibbs Sampling

> $p(z=t|w) \propto \underbrace{\frac{\alpha_t \beta}{n_{\cdot|t}+V\beta}}_{A} + \underbrace{\frac{n_{t|d}\,\beta}{n_{\cdot|t}+V\beta}}_{B} + \underbrace{\frac{n_{w|t}\;(\alpha_t+n_{t|d})}{n_{\cdot|t}+V\beta}}_{C}$

> Summing A, B, and C across all documents gives some insight…
  - A represents smoothing, merely calculating Dirichlet parameter for the next step
  - B represents the topics k that appear in document d
  - C represents other topics assigned to word w across all documents

Output

- Loosely defined topics
- Words assigned to topics by percentages
  - Can extend to summing over all words in the document to give the document a topic percentage
  - Preprocessing to remove extraneous words helps accuracy here
- Must decide on an appropriate number of topics



Source: https://www.mathworks.com/help/textanalytics/examples/analyze-text-data-using-topic-models.html

MISSOURI S&T

Outline

- > Preliminary Steps
  - – Preprocessing
  - – Stemming Algorithms
  - – Bag of Word Model
- > Term Frequency – Inverse Document Frequency
  - – Zipf's Law
- > Latent Dirichlet Allocation
- > **Latent Semantic Analysis**
- > Conclusions

Latent Semantic Analysis

- Patented in 1988, now-expired US patent 4,839,853
  - S. C. Deerwester, S. T. Dumais, G. W. Furnace, R. A. Harshman, T. K. Landauer, K. E. Lochbaum, and L. A. Streeter, “Computer information retrieval using latent semantic structure ,” 13-Jun-1989.
- Relies on singular value decomposition of a term-document matrix
  - Strictly matrix math; no learning algorithms used here
- Also addresses shortcomings of simple keyword searches
  - Overrides synonymy (different words with similar meanings)
  - Overrides polysemy (similar words with different meanings)

Latent Semantic Analysis Algorithm

1. Construct a term-document matrix M [m words * n documents]
2. Take the **singular value decomposition** of the matrix $M = U\Sigma V^*$
   - Where U is an $m * m$ unitary matrix over $\mathbb{R}$
     - > (i.e. the columns of U are made up of the eigenvectors of MM^*)
   - Where V is an $n * n$ unitary matrix over $\mathbb{R}$
     - > (i.e. the columns of V are made up of the eigenvectors of M^*M)
   - Where S is a diagonal $m * n$ matrix containing the singular values of M
     - (i.e. the square roots of eigen values from either M^*M or MM^*)
3. Zero out all but the top K values of the singular matrix
   - Where K is the desired number of topics
4. The remaining matrix represents vectors in dimensional space
   - Closer vectors represent more similar documents

MISSOURI S&T

Example

	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
voyage	1	0	0	1	1	0
trip	0	0	0	1	0	1

Term-Document Matrix

U Matrix

	1	2	3	4	5
ship	−0.44	−0.30	0.57	0.58	0.25
boat	−0.13	−0.33	−0.59	0.00	0.73
ocean	−0.48	−0.51	−0.37	0.00	−0.61
voyage	−0.70	0.35	0.15	−0.58	0.16
trip	−0.26	0.65	−0.41	0.58	−0.09

	1	2	3	4	5
ship	2.16	0.00	0.00	0.00	0.00
boat	0.00	1.59	0.00	0.00	0.00
ocean	0.00	0.00	1.28	0.00	0.00
voyage	0.00	0.00	0.00	1.00	0.00
trip	0.00	0.00	0.00	0.00	0.39

S Matrix

V^* Matrix

	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.28	−0.75	0.45	−0.20	0.12	−0.33
4	0.00	0.00	0.58	0.00	−0.58	0.58
5	−0.53	0.29	0.63	0.19	0.41	−0.22

Taken from: C. D. Manning, H. Schütze, and P. Raghavan, *An introduction to Information Retrieval*. Cambridge, England: Cambridge University Press, 2008.

MISSOURI S&T

Example

	d_1	d_2	d_3	d_4	d_5	d_6
ship	1	0	1	0	0	0
boat	0	1	0	0	0	0
ocean	1	1	0	0	0	0
voyage	1	0	0	1	1	0
trip	0	0	0	1	0	1

Term-Document Matrix

U Matrix

	1	2	3	4	5
ship	−0.44	−0.30	0.57	0.58	0.25
boat	−0.13	−0.33	−0.59	0.00	0.73
ocean	−0.48	−0.51	−0.37	0.00	−0.61
voyage	−0.70	0.35	0.15	−0.58	0.16
trip	−0.26	0.65	−0.41	0.58	−0.09

	1	2	3	4	5
ship	2.16	0.00	0.00	0.00	0.00
boat	0.00	1.59	0.00	0.00	0.00
ocean	0.00	0.00	0.00	0.00	0.00
voyage	0.00	0.00	0.00	0.00	0.00
trip	0.00	0.00	0.00	0.00	0.00

Truncated
S Matrix

V^* Matrix

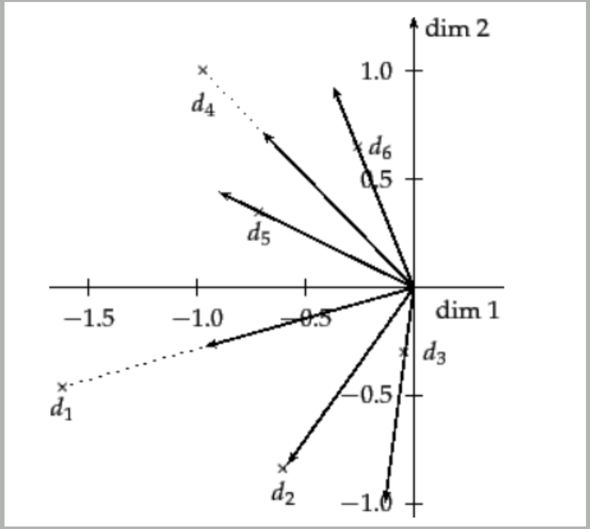
	d_1	d_2	d_3	d_4	d_5	d_6
1	−0.75	−0.28	−0.20	−0.45	−0.33	−0.12
2	−0.29	−0.53	−0.19	0.63	0.22	0.41
3	0.28	−0.75	0.45	−0.20	0.12	−0.33
4	0.00	0.00	0.58	0.00	−0.58	0.58
5	−0.53	0.29	0.63	0.19	0.41	−0.22

Taken from: C. D. Manning, H. Schütze, and P. Raghavan, *An introduction to Information Retrieval*. Cambridge, England: Cambridge University Press, 2008.

Example

	d_1	d_2	d_3	d_4	d_5	d_6
1	−1.62	−0.60	−0.44	−0.97	−0.70	−0.26
2	−0.46	−0.84	−0.30	1.00	0.35	0.65
3	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.00	0.00

Final Singular Value Decomposed Matrix



Topic Vectors Visualized

Taken from: C. D. Manning, H. Schütze, and P. Raghavan, *An introduction to Information Retrieval*. Cambridge, England: Cambridge University Press, 2008.

MISSOURI S&T

Outline

- Preliminary Steps
  - Preprocessing
  - Stemming Algorithms
  - Bag of Word Model
- Term Frequency – Inverse Document Frequency
  - Zipf's Law
- Latent Dirichlet Allocation
- Latent Semantic Analysis
- **Conclusions**

MISSOURI S&T

In Summary

- Primary models for NLP/text mining
  - Term Frequency-Inverse Document Frequency
    - Gives the relevance score for each document in regards to a term
  - Latent Dirichlet Allocation
    - Learning model, uses Bayesian inference
    - Gives a finite topic distribution per document
  - Latent Semantic Analysis
    - Mathematical, matrix-based model
    - Results in topic-vectors that can be analyzed for similarity

Questions?