Natural Language Processing (CSE 517): Cotext Models

Noah Smith

University of Washington
nasmith@cs.washington.edu

April 13, 2018

Thanks to David Mimno for comments.

Quick Review

A language model is a probability distribution over $\mathcal{V}^\dagger$.

Typically p decomposes into probabilities $p(x_i \mid \boldsymbol{h}_i)$.

- We considered n-gram, class-based, log-linear, and neural language models.

Today: probabilistic models that relate a word and its **cotext** (the linguistic environment of the word).

- This might help us learn to represent words, contexts, or both.

Three Kinds of Cotext

If we consider a word token at a particular position i in text to be the observed value of a random variable X_i, what other random variables are predictive of/related to X_i?

Three Kinds of Cotext

If we consider a word token at a particular position i in text to be the observed value of a random variable X_i, what other random variables are predictive of/related to X_i?

1. the document containing i (a moderate-to-large collection of other words)

Three Kinds of Cotext

If we consider a word token at a particular position i in text to be the observed value of a random variable X_i, what other random variables are predictive of/related to X_i?

1. the document containing i (a moderate-to-large collection of other words)
2. the words that occur within a small "window" around i (e.g., x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}, or maybe the sentence containing i)

Three Kinds of Cotext

If we consider a word token at a particular position i in text to be the observed value of a random variable X_i, what other random variables are predictive of/related to X_i?

1. the document containing i (a moderate-to-large collection of other words)
2. the words that occur within a small "window" around i (e.g., x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}, or maybe the sentence containing i)
3. a sentence known to be a translation of the one containing i

Three Kinds of Cotext

If we consider a word token at a particular position i in text to be the observed value of a random variable X_i, what other random variables are predictive of/related to X_i?

1. the document containing i (a moderate-to-large collection of other words) $\longrightarrow$ topic models
2. the words that occur within a small "window" around i (e.g., x_{i-2}, x_{i-1}, x_{i+1}, x_{i+2}, or maybe the sentence containing i) $\longrightarrow$ distributional semantics
3. a sentence known to be a translation of the one containing i $\longrightarrow$ translation models

Topic Models

- Words are not IID!
  - Predictable given history: n-gram/Markov models
  - Predictable given other words in the document: topic models

Topic Models

- Words are not IID!
  - Predictable given history: n-gram/Markov models
  - Predictable given other words in the document: topic models
- Let $\mathcal{Z} = \{1, \ldots, k\}$ be a set of "topics" or "themes" that will help us capture the interdependence of words in a document.
  - Usually these are not named or characterized in advance; they are just k different values with no *a priori* meaning.

Topic Models

- Words are not IID!
  - Predictable given history: n-gram/Markov models
  - Predictable given other words in the document: topic models
- Let $\mathcal{Z} = \{1, \ldots, k\}$ be a set of "topics" or "themes" that will help us capture the interdependence of words in a document.
  - Usually these are not named or characterized in advance; they are just k different values with no *a priori* meaning.
- We'll start with a classical topic model, then turn to probabilistic ones.

The Term-Document Matrix

Let $\mathbf{A} \in \mathbb{R}^{V \times C}$ contain statistics of association between words in $\mathcal{V}$ and C documents.

N is the total number of word tokens.

Tiny example, three documents:

- `yes , we have no bananas`
- `say yes for bananas`
- `no bananas , we say`

	1	2	3
`,`	1	0	1
`bananas`	1	1	1
`for`	0	1	0
`have`	1	0	0
`no`	1	0	1
`say`	0	1	1
`we`	1	0	1
`yes`	1	1	0

Count matrix: $[\mathbf{A}]_{v,c} = c_{\boldsymbol{x}_c}(v)$

Association Score

What we really want here is some way to get at "surprise."

Association Score

What we really want here is some way to get at "surprise."

One way to think about this is, is the occurrence of word v in document c surprisingly high (or low), given what we'd expect due to chance?

Association Score

What we really want here is some way to get at "surprise."

One way to think about this is, is the occurrence of word v in document c surprisingly high (or low), given what we'd expect due to chance?

Chance would be $\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N}$ words out of the ℓ_c words in document c.

Association Score

What we really want here is some way to get at "surprise."

One way to think about this is, is the occurrence of word v in document c surprisingly high (or low), given what we'd expect due to chance?

Chance would be $\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N}$ words out of the ℓ_c words in document c.

Intuition: consider the ratio of *observed* frequency ($c_{\boldsymbol{x}_c}(v)$) to "chance" under independence ($\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c$).

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

From our example:

$$[\mathbf{A}]_{\text{bananas},1} = \log \frac{15 \cdot 1}{3 \cdot 6} \approx -0.18 \to 0$$

$$[\mathbf{A}]_{\text{for},2} = \log \frac{15 \cdot 1}{1 \cdot 4} \approx 1.32$$

	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

A Nod to Information Theory

Pointwise mutual information for two random variables A and B:

$$\begin{aligned}
\mathsf{PMI}(a, b) &= \log \frac{p(A = a, B = b)}{p(A = a) \cdot p(B = b)} \\
&= \log \frac{p(A = a \mid B = b)}{p(A = a)} \\
&= \log \frac{p(B = b \mid A = a)}{p(B = b)}
\end{aligned}$$

A Nod to Information Theory

Pointwise mutual information for two random variables A and B:

$$\text{PMI}(a, b) = \log \frac{p(A = a, B = b)}{p(A = a) \cdot p(B = b)}$$

The **average mutual information** is given by:

$$\text{MI}(A, B) = \sum_{a,b} p(A = a, B = b) \cdot \mathrm{PMI}(a, b)$$

This comes from information theory; it is the amount of information each r.v. offers about the other.

(Recall Shannon entropy; that's the amount of information in a single random variable.)

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

Notes:

- If a word v appears with nearly the same frequency in every document, its row $[\mathbf{A}]_{v,*}$ will be all nearly zero.

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

Notes:

- If a word v appears with nearly the same frequency in every document, its row $[\mathbf{A}]_{v,*}$ will be all nearly zero.
- If a word v occurs *only* in document c, PMI will be large and positive.

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

Notes:

- If a word v appears with nearly the same frequency in every document, its row $[\mathbf{A}]_{v,*}$ will be all nearly zero.
- If a word v occurs *only* in document c, PMI will be large and positive.
- PMI is very sensitive to rare occurrences; usually we smooth the frequencies and filter rare words.

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

Notes:

- If a word v appears with nearly the same frequency in every document, its row $[\mathbf{A}]_{v,*}$ will be all nearly zero.
- If a word v occurs *only* in document c, PMI will be large and positive.
- PMI is very sensitive to rare occurrences; usually we smooth the frequencies and filter rare words.
- One way to think about PMI: it's telling us where a unigram model is most wrong.

Pointwise Mutual Information

A common starting point is positive **pointwise mutual information**:

$$[\mathbf{A}]_{v,c} = \left[\log \frac{c_{\boldsymbol{x}_c}(v)}{\frac{c_{\boldsymbol{x}_{1:C}}(v)}{N} \cdot \ell_c}\right]_+ = \left[\log \frac{N \cdot c_{\boldsymbol{x}_c}(v)}{c_{\boldsymbol{x}_{1:C}}(v) \cdot \ell_c}\right]_+$$

Notes:

- If a word v appears with nearly the same frequency in every document, its row $[\mathbf{A}]_{v,*}$ will be all nearly zero.
- If a word v occurs *only* in document c, PMI will be large and positive.
- PMI is very sensitive to rare occurrences; usually we smooth the frequencies and filter rare words.
- One way to think about PMI: it's telling us where a unigram model is most wrong.
- We could use $\mathbf{A}$ as $\mathbf{M}$ (though d is usually much smaller than C) ...

Topic Models: Latent Semantic Indexing/Analysis
(Deerwester et al., 1990)

LSI/A seeks to solve:

$$\underset{V \times C}{\mathbf{A}} \approx \hat{\mathbf{A}} = \underset{V \times d}{\mathbf{M}} \times \underset{d \times d}{\operatorname{diag}(\mathbf{s})} \times \underset{d \times C}{\mathbf{C}}^\top$$

where $\mathbf{M}$ contains embeddings of words, $\mathbf{C}$ contains embeddings of documents.

$$[\mathbf{A}]_{v,c} \approx \sum_{i=1}^{d} [\mathbf{v}_v]_i \cdot [\mathbf{s}]_i \cdot [\mathbf{c}_c]_i$$

Topic Models: Latent Semantic Indexing/Analysis

(Deerwester et al., 1990)

LSI/A seeks to solve:

$$\underset{V \times C}{\mathbf{A}} \approx \hat{\mathbf{A}} = \underset{V \times d}{\mathbf{M}} \times \underset{d \times d}{\operatorname{diag}(\mathbf{s})} \times \underset{d \times C}{\mathbf{C}}^{\top}$$

where $\mathbf{M}$ contains embeddings of words, $\mathbf{C}$ contains embeddings of documents.

$$[\mathbf{A}]_{v,c} \approx \sum_{i=1}^{d} [\mathbf{v}_v]_i \cdot [\mathbf{s}]_i \cdot [\mathbf{c}_c]_i$$

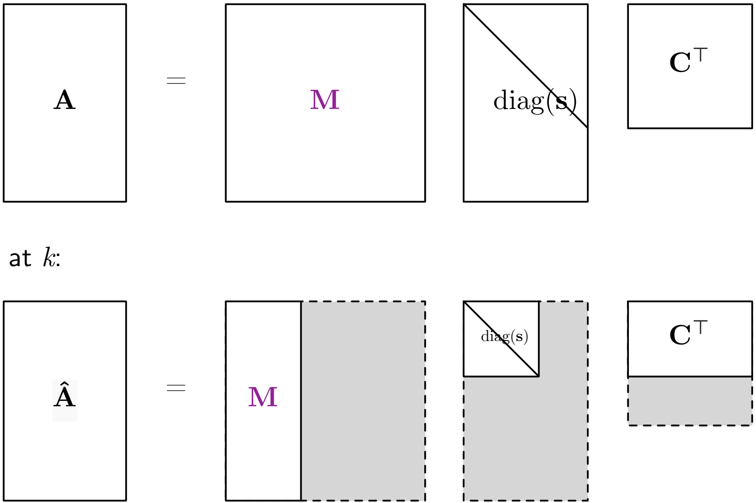
This can be solved by applying singular value decomposition to $\mathbf{A}$, then truncating to d dimensions.

- $\mathbf{M}$ contains left singular vectors of $\mathbf{A}$
- $\mathbf{C}$ contains right singular vectors of $\mathbf{A}$
- $\mathbf{s}$ are singular values of $\mathbf{A}$; they are nonnegative and conventionally organized in decreasing order.

Truncated Singular Value Decomposition

SVD:

truncated at k:



A Nod to Linear Algebra

For (not truncated) singular value decomposition $\mathbf{A} = \mathbf{M} \times \mathrm{diag}(\mathbf{s}) \times \mathbf{C}^\top$:

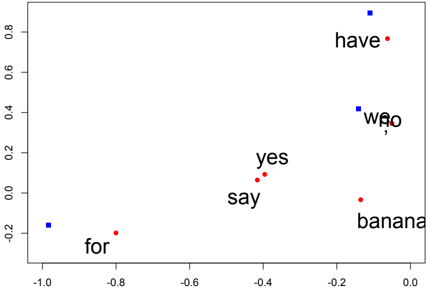
- The columns of $\mathbf{M}$ form an orthonormal basis, $\mathbf{M}$ are eigenvectors of $\mathbf{A}\mathbf{A}^\top$, with eigenvalues $\mathbf{s}^2$.
- The columns of $\mathbf{C}$ form an orthonormal basis, $\mathbf{C}$ are eigenvectors of $\mathbf{A}^\top\mathbf{A}$, with eigenvalues $\mathbf{s}^2$.

If some elements of $\mathbf{s}$ are zero, then $\mathbf{A}$ is "low rank."

Approximating $\mathbf{A}$ by truncating $\mathbf{s}$ equates to a "low rank approximation."

LSI/A Example

$d = 2$

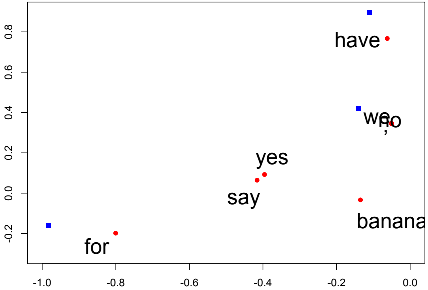


	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

Words and documents in two dimensions.

LSI/A Example

$d = 2$



	1	2	3
,	1	0	1
bananas	1	1	1
for	0	1	0
have	1	0	0
no	1	0	1
say	0	1	1
we	1	0	1
yes	1	1	0

Words and documents in two dimensions.

Note how `no`, `we`, and `,` are all in the exact same spot. Why?

Understanding LSI/A

- Mapping words and documents into the same k-dimensional space.
- Bag of words assumption (Salton et al., 1975): a document is nothing more than the distribution of words it contains.
- Distributional hypothesis (Harris, 1954; Firth, 1957): words are nothing more than the distribution of contexts (here, documents) they occur in. Words that occur in similar contexts have similar meanings.
- $\mathbf{A}$ is sparse and noisy; LSI/A "fills in" the zeroes and tries to eliminate the noise.
  - It finds the best rank-k approximation to $\mathbf{A}$.

Probabilistic Topic Models

As a language model, LSI/A is kind of broken.

- It assumes the elements of $\mathbf{A}$ are the result of Gaussian noise.

Hofmann (1999) proposed instead to model the probability distribution $p(\boldsymbol{X}_c = \boldsymbol{x}_c \mid c)$, for each document c in the corpus $\mathcal{C}$.

- This is a particular kind of *conditional* language model.

Probabilistic Latent Semantic Analysis

(Hofmann, 1999)

Given a corpus $\mathcal{C}$, for every $c \in \mathcal{C}$:

$$p(\boldsymbol{x} \mid c) = \sum_{\boldsymbol{z} \in \{1,\ldots,k\}^\ell} p(\boldsymbol{x}, \boldsymbol{z} \mid c)$$

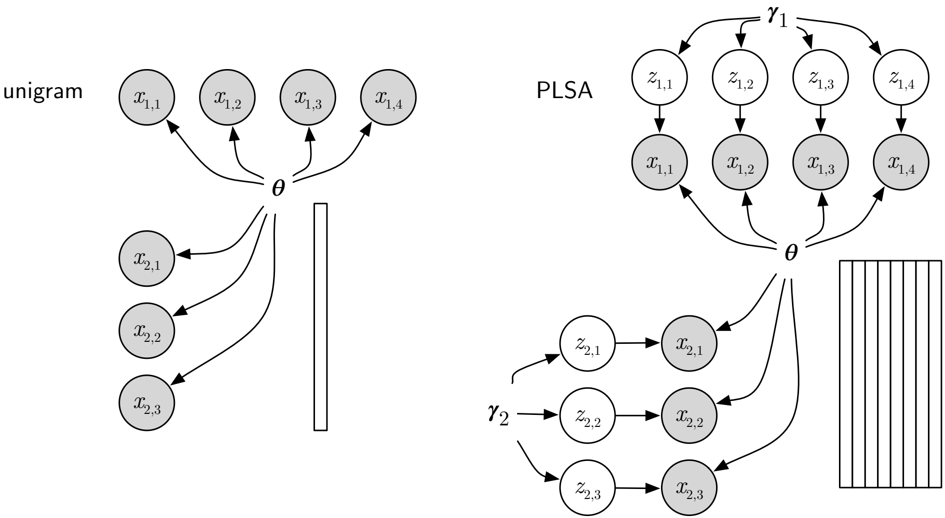
$$\begin{aligned} p(\boldsymbol{x}, \boldsymbol{z} \mid c) &= \prod_{i=1}^{\ell} p(z_i \mid c) \cdot p(x_i \mid z_i) \\ &= \prod_{i=1}^{\ell} \gamma_{z_i|c} \, \theta_{x_i|z_i} \end{aligned}$$

Parameters:

- $\gamma_{z|c}$, $\forall z \in \{1, \ldots, k\}, \forall c \in \mathcal{C}$
- $\theta_{v|z}$, $\forall v \in \mathcal{V}, \forall z \in \{1, \ldots, k\}$

There is no closed form for the MLE!

“Graphical Model” Depiction of PLSA



A Chicken/Egg Problem

If we knew which topic each word token belonged to (i.e., which unigram distribution generated it), we could use relative frequency estimation.

If we knew the parameters γ and $\boldsymbol{\theta}$, we could infer the topic of each word (i.e., which unigram distribution generated it).

"Soft Counts"

Assume for the moment a single document c of length ℓ.

When we estimated unigram language models, everything relied on *counts* of words.

Here, if we knew the counts of every word in every topic in every document, then we'd have a closed form MLE.

$$\hat{\gamma}_{z|c} = \frac{c(z,*)}{\ell}$$

$$\hat{\theta}_{v|z} = \frac{c(z,v)}{c(z,*)}$$

"Soft Counts"

Assume for the moment a single document c of length ℓ.

When we estimated unigram language models, everything relied on *counts* of words.

Here, if we knew the counts of every word in every topic in every document, then we'd have a closed form MLE.

$$\hat{\gamma}_{z|c} = \frac{c(z,*)}{\ell}$$

$$\hat{\theta}_{v|z} = \frac{c(z,v)}{c(z,*)}$$

Instead, we will replace counts with "soft counts."

$$\hat{\gamma}_{z|c} = \frac{\tilde{c}(z,*)}{\ell}$$

$$\hat{\theta}_{v|z} = \frac{\tilde{c}(z,v)}{\tilde{c}(z,*)}$$

Expectation Maximization

Many ways to understand it. Today, we'll stick with a simple one.

Start with arbitrary (e.g., random) parameter values. Alternate between two steps:

- E step: calculate the posterior distribution over each latent variable.
- M step: treat the posteriors as soft counts, and re-estimate the model.

Doing this is a kind of hill-climbing on the likelihood of the *observed* data.

PLSA: M Step

Each word x_i is fractionally assigned to every topic z with value $\tilde{c}_c(z, x_i)$.

$$\hat{\gamma}_{z|c} = \frac{\tilde{c}_c(z)}{\ell_c} = \frac{\sum_{v \in \mathcal{V}} \tilde{c}_c(z, v)}{\ell_c}$$

$$\hat{\theta}_{v|z} = \frac{\sum_{c \in \mathcal{C}} \tilde{c}_c(z, v)}{\sum_{c \in \mathcal{C}} \tilde{c}_c(z)} = \frac{\sum_{c \in \mathcal{C}} \tilde{c}_c(z, v)}{\sum_{c \in \mathcal{C}} \sum_{v \in \mathcal{V}} \tilde{c}_c(z, v)}$$

Note that the $\boldsymbol{\theta}$ parameters are shared across $\mathcal{C}$; all of the documents influence our beliefs about the others through $\boldsymbol{\theta}$.

PLSA: E Step

Assume we have the parameters:

- $\gamma_{z|c}$, $\forall z \in \{1, \ldots, k\}, \forall c \in \mathcal{C}$
- $\theta_{v|z}$, $\forall v \in \mathcal{V}, \forall z \in \{1, \ldots, k\}$

Calculate, for every $c \in \mathcal{C}$, for every word x_i in c, its "membership" to every topic:

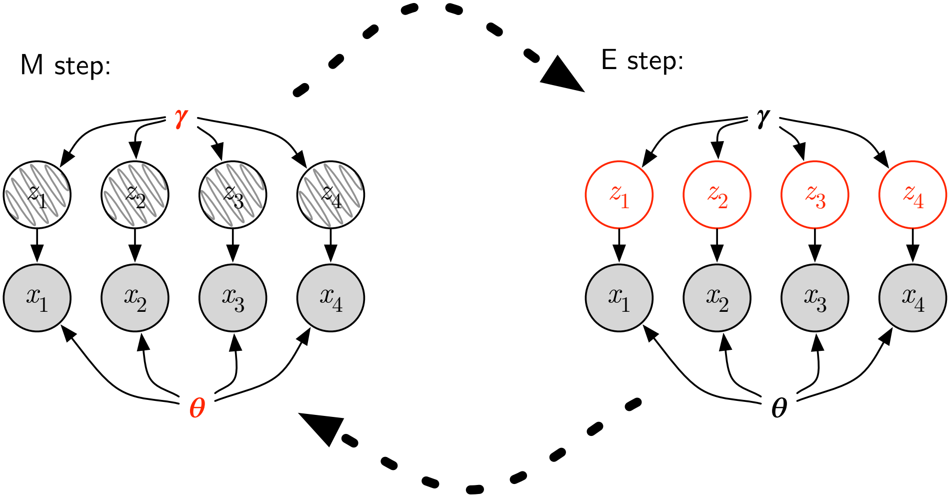
$$
\begin{aligned}
p(Z_i = z \mid x_i, c) &= \frac{p(x_i, z \mid c)}{\sum_{z'} p(x_i, z' \mid c)} \\
&= \frac{p(z \mid c) \cdot p(x_i \mid z)}{\sum_{z'} p(z' \mid c) \cdot p(x_i \mid z')} \\
&= \frac{\gamma_{z|c} \cdot \theta_{x_i|z}}{\sum_{z'} \gamma_{z'|c} \cdot \theta_{x_i|z'}}
\end{aligned}
$$

Each word gets to vote on topics; it can spread its vote fractionally across $\mathcal{Z}$, but the votes sum to 1.

These get summed into soft counts:

$$
\tilde{c}_c(z, v) = \sum_{i : x_i = v} p(Z_i = z \mid x_i, c)
$$

EM for PLSA



Red indicates what is operated on in each step; everything else is held fixed.

Expectation Maximization

Very general technique for learning with *incomplete data*. It's been invented over and over in different fields.

Requires that you specify a generative model with two kinds of variables: **observed** (here, documents and words in each document), and **latent** (here, topic for each word).

Like gradient ascent for neural networks, we are (usually) optimizing a non-convex function. Many tricks exist to try to cope with that.

In NLP, often associated with unsupervised learning. We will see it again!

Remarks

- Like LSI/A, PLSA "squeezes" the relationship between words and contexts (documents) through topics.
- A document is now characterized as a *mixture* of corpus-universal topics (each of which is a unigram model).
- Topic mixtures can be incorporated into language models; see Iyer and Ostendorf (1999), for example.
- Compared to LSI/A: PLSA is more interpretable (e.g., LSI/A can give negative values!).
- PLSA cannot assign probability to a text not in $\mathcal{C}$; it only defines conditional distributions over words given texts in $\mathcal{C}$.
- The next model overcomes this problem by adding another level of randomness: γ becomes a random variable, not a parameter.

Latent Dirichlet Allocation

(Blei et al., 2003)

Widely used today.

$$p(\boldsymbol{x}) = \int_{\boldsymbol{\gamma}} \sum_{\boldsymbol{z} \in \{1,\ldots,k\}^{\ell}} p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\gamma}) \; d\boldsymbol{\gamma}$$

$$p(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\gamma}) = \mathrm{Dir}_{\boldsymbol{\alpha}}(\boldsymbol{\gamma}) \prod_{i=1}^{\ell} \gamma_{z_i} \; \theta_{x_i|z_i}$$

Parameters:

- $\boldsymbol{\alpha} \in \mathbb{R}^k_{>0}$
- $\boldsymbol{\theta}_{*|z} \in \triangle^V$, $\forall z \in \{1, \ldots, k\}$

There is no closed form for the MLE!

"Being Bayesian"

This is another topic that could warrant an entire quarter (e.g., `http://homepages.inf.ed.ac.uk/scohen/bayesian`)

A summary of the Bayesian philosophy in NLP:

- Because we have finite data, we should be uncertain about every estimated model parameter.
- Bayes' rule gives us a way to manage that uncertainty, if we can define a **prior** distribution over model parameters.
- Inference is a "simple matter" of estimating posterior distributions.
  - But exact inference is almost never tractable, so we need approximations.
  - There are many of these, and they tend to be expensive.
  - Some of them look like EM, some don't.

References I

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391–407, 1990.

J. R. Firth. A synopsis of linguistic theory 1930–1955. In *Studies in Linguistic Analysis*, pages 1–32. Blackwell, 1957.

Zellig Harris. Distributional structure. *Word*, 10(23):146–162, 1954.

Thomas Hofmann. Probabilistic latent semantic indexing. In *Proc. of SIGIR*, 1999.

Rukmini M. Iyer and Mari Ostendorf. Modeling long distance dependence in language: Topic mixtures versus dynamic cache models. *Speech and Audio Processing, IEEE Transactions on*, 7(1):30–39, 1999.

Gerard Salton, Anita Wong, and Chung-Shu Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.