# Accounting for Burstiness in Topic Models

Charles Elkan

University of California, San Diego

July 3, 2009

# Text mining

What is text mining?

Working answer: Learning to classify and organize documents.

Example application: Recognizing "helpful" answers to Google/Yahoo questions.

Three central questions:
(1) how to represent a document?
(2) how to model a set of closely related documents?
(3) how to model a set of distantly related documents?

# Mindsets and issues

Mindsets:

- Linguistics vs databases, mathematics vs computation
- Probability vs linear algebra, frequentist vs Bayesian
- Single topic per document vs multiple
- Choosing the right basic model

What issues are important? From most to least interesting :-)

- Sequencing of words:
  "apple pie market share" vs "apple share of market pie"
- Burstiness of words
- Structure of documents
- Repetitions across documents, within documents

# With thanks to ...

Gabe Doyle (UCSD Linguistics)

David Kauchak (UCSD, Adchemy, Pomona College)

Rasmus Madsen (TU Denmark, UCSD)

Three central questions:

(1) how to represent a document?

(2) how to model documents from one class?

(3) how to model documents from multiple classes?

Answers:

(1) "bag of words"

(2) Dirichlet compound multinomial (DCM) distribution

(3) DCM-based topic model (DCMLDA)

# The "bag of words" representation

Let $V$ be a fixed vocabulary (reference set of words).
Write $m = |V|$.

Each document is a vector $x$ of length $m$.

$x_j$ is the number of appearances of word $j$ in the document.

The length of the document is $n = \sum_{j=1}^{m} x_j$.

For typical documents, $n \ll m$ and $x_j = 0$ for most words $j$.

# Standard model for a class of documents

Let $\varphi$ be the parameter vector of a multinomial distribution
i.e. a fixed probability for each word.

The probability of document $x$ is

$$p(x|\varphi) = \Big( \frac{n!}{\prod_{j=1}^m x_j!} \Big) \Big( \prod_{j=1}^m \varphi_j^{x_j} \Big).$$

Each appearance of a word $j$ always has the same probability $\varphi_j$.

Remember $n \ll m$. Computing $p(x|\varphi)$ needs only $O(n)$ time.

# The phenomenon of burstiness

In reality, additional appearances of the same word are less surprising, i.e. they have higher probability. Example:

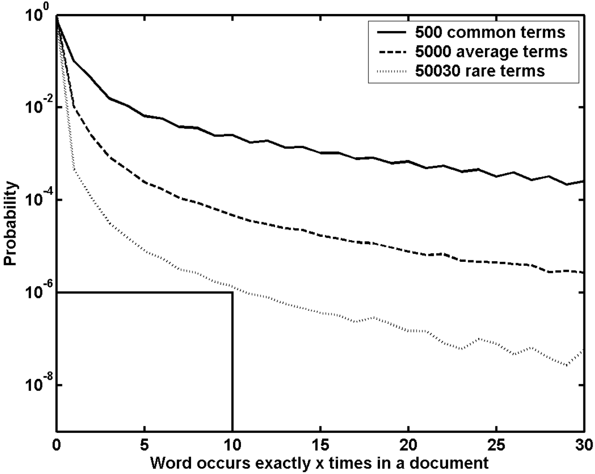
> *Toyota Motor Corp. is expected to announce a major overhaul. Yoshi Inaba, a former senior Toyota executive, was formally asked by Toyota this week to oversee the U.S. business. Mr. Inaba is currently head of an international airport close to Toyota's headquarters in Japan.*
>
> *Toyota's U.S. operations now are suffering from plunging sales. Mr. Inaba was credited with laying the groundwork for Toyota's fast growth in the U.S. before he left the company.*
>
> *Recently, Toyota has had to idle U.S. assembly lines and offer a limited number of voluntary buyouts. Toyota now employs 36,000 in the U.S.*

# Consider the query "apple pie iphone"

Word counts are significant, but how should they be used?

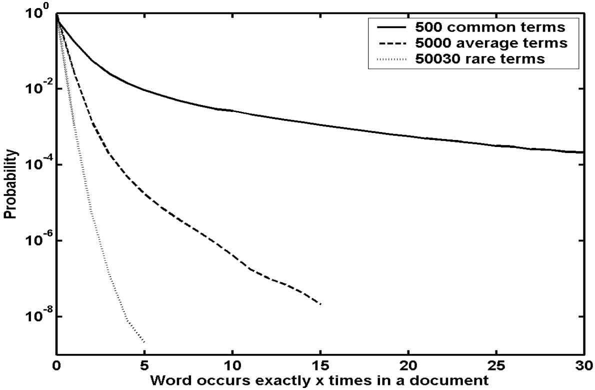
> _The Apple iPhone is becoming as popular as apple pie. Much of the appeal of the iPhone is due to the numerous apps available. How does one approach the development of an iPhone app? We have invited Christopher Allen, co-author of the definitive iPhone App book, "iPhone in Action: Web and SDK Development" to moderate a panel to explore the economic and technical issues._
>
> _* iPhone App Store Strategy: Design, Category, Pricing_
> _* iPhone Analytics: Web Traffic and SDK Install_
> _* iPhone Intelligence: Usage and Behavior_
> _* iPhone Monetization Options_
> _* iPhone Developer Challenges: How to sustain income?_

# Empirical evidence of burstiness



Explanation: The chance that a given rare word occurs 10 times in a document is $10^{-6}$. The chance that it occurs 20 times is $10^{-6.5}$.

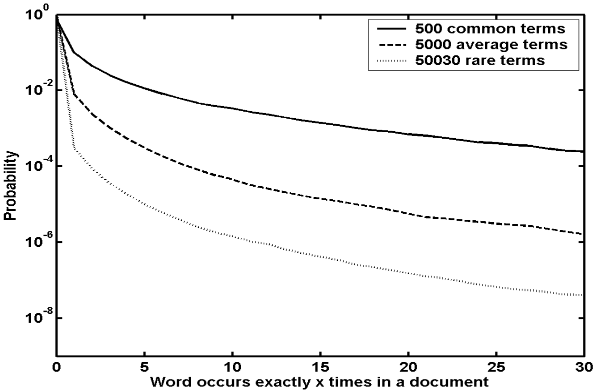
# Multinomial models fail



A multinomial is appropriate only for modeling common words.

Burstiness and informativeness are correlated: more diagnostic words are also more bursty.

# DCM models succeed



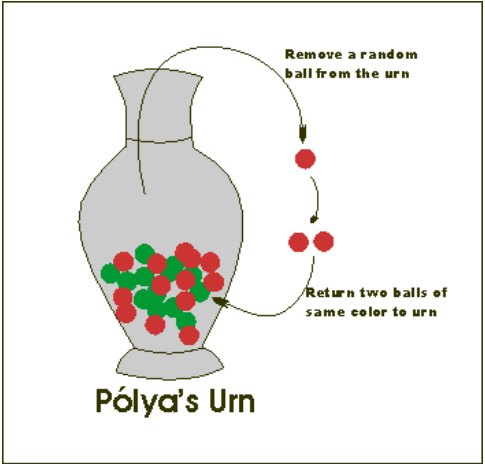
A trained DCM model gives correct probabilities for all counts of all types of words.

# The Polya urn

Consider a bucket with balls of $m = |V|$ different colors.

Each time a ball is selected randomly, it is replaced *and one more ball of the same color is added.*

Let the initial number of balls with color $j$ be $\beta_j$, which can be non-integer.


Remove a random ball from the urn
Return two balls of same color to urn
Pólya's Urn

# The bag-of-bag-of-words process

Let $\varphi$ be the parameter vector of a multinomial.

Let $\mathrm{Dir}(\beta)$ be a Dirichlet distribution over $\varphi$.

To generate a document:
(1) draw document-specific multinomial $\varphi \sim \mathrm{Dir}(\beta)$
(2) draw $n$ words $w \sim \mathrm{Mult}(\varphi)$.

Each document consists of words drawn from a multinomial that is fixed for that document, but different for other documents.

Remarkably, the Polya urn and the bag-of-bag-of-words process yield the same probability distribution over documents.

# How is burstiness captured?

A multinomial parameter vector $\varphi$ has length $|V|$ and is constrained: $\sum_{j=1}^{m} \varphi_j = 1$.

A DCM parameter vector $\beta$ has the same length $|V|$ but is unconstrained.

The one extra degree of freedom allows the DCM to discount multiple observations of the same word, in an adjustable way.

The smaller the sum $s = \sum_{j=1}^{m} \beta_j$, the more words are bursty.

# Moving forward ...

Three central questions:
(1) how to represent documents?
(2) how to model closely related documents?
(3) how to model distantly related documents?

A DCM is a good model of documents that all share a single theme.

$\beta$ represents the central theme; for each document $\varphi$ represents its variation on this theme.

By combining DCMs with latent Dirichlet allocation (LDA), we answer (3).

# Clustering considered harmful

A single DCM models a space of closely related subtopics.

In 2006 we extended the DCM model to a mixture of DCM distributions.

But a mixture distribution (i.e. clustering) assumes that each document arises from a single component.

We want to allow multiple components within one document.

Also: We want to avoid local optima.

# Text mining: Not just for text anymore

Goal: Find companies whose stock prices tend to move together.

Example: { IBM+, MSFT+, AAPL- } means IBM and Microsoft often rise, and Apple falls, on the same days.

Let each day be a document containing words like IBM+.

Each word is a stock symbol and a direction (+ or -). Each day has one copy of the word for each 1% change in the stock price.

Let a co-moving group of stocks be a topic. Each day is a union of multiple topics.

# Examples of discovered topics

| “Computer Related” | | “Real Estate” | |
|---|---|---|---|
| symbol | company | symbol | company |
| NVDA+ | Nvidia | SPG+ | Simon Properties |
| SNDK+ | SanDisk | AIV+ | Apt. Investment |
| BRCM+ | Broadcom | KIM+ | Kimco Realty |
| JBL+ | Jabil Circuit | AVB+ | AvalonBay |
| KLAC+ | KLA-Tencor | DDR+ | Developers |
| NSM+ | Nat’l Semicond. | EQR+ | Equity Residential |

The dataset contains 501 days of transactions between January 2007 and September 2008.

# DCMLDA advantages

Unlike a mixture model, a topic model allows many topics to occur in each document.

DCMLDA allows the same topic to occur with different words in different documents.

Consider a “sports” topic. Suppose “rugby” and “hockey” are equally common. But within each document, seeing “rugby” makes seeing “rugby” again more likely than seeing “hockey.”

A standard topic model cannot represent this burstiness, unless the words “rugby” and “hockey” are spread across two topics.

# Hypothesis

"A DCMLDA model with a few topics can fit a corpus as well as an LDA model with many topics."

Motivation: A single DCMLDA topic can explain related aspects of documents more effectively than a single LDA topic.

The hypothesis is confirmed by the experimental results below.

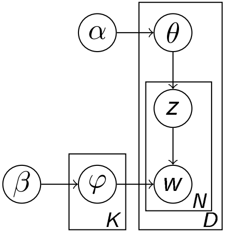
Well-known fact: Topic models find sets of topics that are too flat.

# Latent Dirichlet Allocation (LDA)

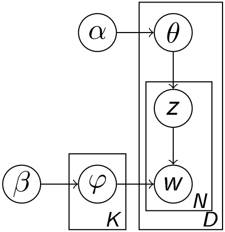
LDA is a generative model:

For each of $K$ topics, draw a multinomial to describe it.

For each of $D$ documents:
(1) Determine the probability of each of $K$ topics in this document.
(2) For each of $N$ words:
first draw a topic, then draw a word based on that topic.



# Graphical model



The fixed parameters of the model are $\alpha$ and $\beta$.

$$\begin{aligned}\varphi &\sim \text{Dirichlet}(\beta)\\ \theta &\sim \text{Dirichlet}(\alpha)\\ z &\sim \text{Multinomial}(\theta)\\ w &\sim \text{Multinomial}(\varphi)\end{aligned}$$

# Using LDA for text mining

Training finds maximum-likelihood values for $\varphi$ for each topic, and for $\theta$ for each document.

For each topic, $\varphi$ is a vector of word probabilities indicating the content of that topic.

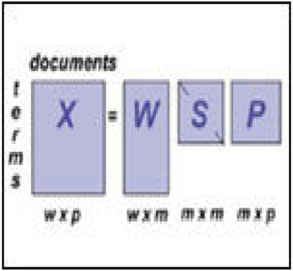
The distribution $\theta$ of each document is a reduced-dimensionality representation. It is useful as:

- a higher level representation for documents
- detecting similarity between words
- more.

# From latent semantic analysis (LSA) to DCMLDA

- LSA = SVD applied to bag-of-words matrix.
- Probabilistic LSA (PLSA) = statistical interpretation of LSA.
- LDA = PLSA extended to model out-of-sample documents.
- DCMLDA = LDA revised to capture burstiness.

All four models have essentially the same interpretation as a matrix factorization.



# Extending LDA to DCMLDA

Goal: Allow multiple topics in a single document, while making subtopics be document-specific.

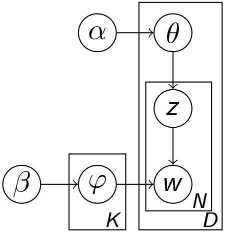
In DCMLDA, for each topic $k$ and each document $d$ a fresh multinomial word distribution is drawn.

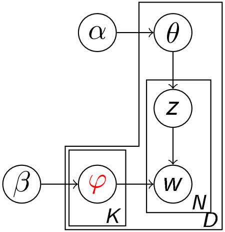
This parameter vector is $\varphi_{kd}$ not $\varphi_k$.

For each topic $k$, these multinomials are drawn from the same Dirichlet $\beta_k$, so all versions of the same topic are linked.

Per-document instances of each topic allow for burstiness.

# Graphical models compared





# DCMLDA generative process

```
for document $d \in \{1, \dots, D\}$ do
    draw topic distribution $\theta_d \sim \mathrm{Dir}(\alpha)$
    for topic $k \in \{1, \dots, K\}$ do
        draw topic-word distribution $\varphi_{kd} \sim \mathrm{Dir}(\beta_k)$
    end for
    for word $n \in \{1, \dots, N_d\}$ do
        draw topic $z_{d,n} \sim \theta_d$
        draw word $w_{d,n} \sim \varphi_{z_{d,n}d}$
    end for
end for
```

# Meaning of $\alpha$ and $\beta$

When applying LDA to text, it is not necessary to learn $\alpha$ and $\beta$.

Steyvers and Griffiths recommend fixed uniform values:
$\alpha = 50/K$ and $\beta = .01$ where $K$ is the number of topics.

But, the information in LDA $\varphi$ values is in DCMLDA $\beta$ values.

Without an effective method to learn the hyperparameters, the DCMLDA model is not useful.

No need to average over hyperparameters, or learn them precisely!

# Training

Given a training set of documents, alternate:
(a) optimize parameters $\varphi$, $\theta$, and $z$ given hyperparameters,
(b) optimize hyperparameters $\alpha$, $\beta$ given document parameters.

For fixed $\alpha$ and $\beta$, do collapsed Gibbs sampling to find the distribution of $z$.

Given a $z$ sample, find $\alpha$ and $\beta$ by Monte Carlo expectation-maximization.

When desired, compute $\varphi$ and $\theta$ from samples of $z$.

# Gibbs sampling

Gibbs sampling for DCMLDA is similar to the method for LDA.

Start by factoring the complete likelihood of the model:

$$p(w, z|\alpha, \beta) = p(w|z, \beta)p(z|\alpha).$$

DCMLDA and LDA are identical over the $\alpha$-to-$z$ pathway, so $p(z|\alpha)$ in DCMLDA is the same as for LDA:

$$p(z|\alpha) = \prod_d \frac{B(n_{\cdot\cdot d} + \alpha)}{B(\alpha)}.$$

$B(\cdot)$ is the Beta function, and $n_{tkd}$ is how many times word $t$ has topic $k$ in document $d$.

To get $p(w|z,\beta)$, average over all possible $\varphi$ distributions:

$$\begin{aligned} p(w|z,\beta) &= \int_\varphi p(z|\varphi)p(\varphi|\beta)d\varphi \\ &= \int_\varphi p(\varphi|\beta)\prod_d \prod_{n=1}^{N_d} \varphi_{w_{d,n}z_{d,n}d}\, d\varphi \\ &= \int_\varphi p(\varphi|\beta) \prod_{d,k,t} (\varphi_{tkd})^{n_{tkd}}\, d\varphi. \end{aligned}$$

Expand $p(\varphi|\beta)$ as a Dirichlet distribution:

$$\begin{aligned} p(w|z,\beta) = \int_\varphi \left[\prod_{d,k} \frac{1}{B(\beta_{\cdot k})} \prod_t (\varphi_{tkd})^{\beta_{tk}-1}\right] \left[\prod_{d,k,t} (\varphi_{tkd})^{n_{tkd}}\right] d\varphi \\ = \prod_{d,k} \int_\varphi \prod_t (\varphi_{tkd})^{\beta_{tk}-1+n_{tkd}}\, d\varphi \ = \ \prod_{d,k} \frac{B(n_{\cdot kd}+\beta_{\cdot k})}{B(\beta_{\cdot k})}. \end{aligned}$$

# Gibbs sampling cont.

Combining equations, the complete likelihood is

$$p(w, z|\alpha, \beta) = \prod_d \left[ \frac{B(n_{\cdot\cdot d} + \alpha)}{B(\alpha)} \prod_k \frac{B(n_{\cdot kd} + \beta_{\cdot k})}{B(\beta_{\cdot k})} \right].$$

# Finding optimal $\alpha$ and $\beta$

Optimal $\alpha$ and $\beta$ values maximize $p(w|\alpha, \beta)$. Unfortunately, this likelihood is intractable.

The complete likelihood $p(w, z|\alpha, \beta)$ is tractable. Based on it, we use single-sample Monte Carlo EM.

Run Gibbs sampling for a burn-in period, with guesses for $\alpha$ and $\beta$.

Then draw a topic assignment $z$ for each word of each document. Use this vector in the M-step to estimate new values for $\alpha$ and $\beta$.

Run Gibbs sampling for more iterations, to let topic assignments stabilize based on the new $\alpha$ and $\beta$ values.

Then repeat.

# Training algorithm

```
Start with initial $\alpha$ and $\beta$
repeat
    Run Gibbs sampling to approximate steady state
    Choose a topic assignment for each word
    Choose $\alpha$ and $\beta$ to maximize complete likelihood $p(w, z|\alpha, \beta)$
until convergence of $\alpha$ and $\beta$
```

# $\alpha$ and $\beta$ to maximize complete likelihood

Log complete likelihood is

$$
\begin{aligned}
L(\alpha, \beta; w, z) = & \sum_{d,k} [\log \Gamma(n_{\cdot kd} + \alpha_k) - \log \Gamma(\alpha_k)] \\
& + \sum_{d} [\log \Gamma(\sum_{k} \alpha_k) - \log \Gamma(\sum_{k} n_{\cdot kd} + \alpha_k)] \\
& + \sum_{d,k,t} [\log \Gamma(n_{tkd} + \beta_{tk}) - \log \Gamma(\beta_{tk})] \\
& + \sum_{d,k} [\log \Gamma(\sum_{t} \beta_{tk}) - \log \Gamma(\sum_{t} n_{tkd} + \beta_{tk})].
\end{aligned}
$$

The first two lines depend only on $\alpha$, and the second two on $\beta$. Furthermore, $\beta_{tk}$ can be independently maximized for each $k$.

We get $K+1$ equations to maximize:

$$\begin{aligned}\alpha' &= \operatorname{argmax} \sum_{d,k} (\log \Gamma(n_{\cdot kd} + \alpha_k) - \log \Gamma(\alpha_k)) \\ &\quad + \sum_d [\log \Gamma(\sum_k \alpha_k) - \log \Gamma(\sum_k n_{\cdot kd} + \alpha_k)] \\ \beta'_{\cdot k} &= \operatorname{argmax} \sum_{d,t} (\log \Gamma(n_{tkd} + \beta_{tk}) - \log \Gamma(\beta_{tk})) \\ &\quad + \sum_d [\log \Gamma(\sum_t \beta_{tk}) - \log \Gamma(\sum_t n_{tkd} + \beta_{tk})]\end{aligned}$$

Each equation defines a vector, either $\{\alpha_k\}_k$ or $\{\beta_{tk}\}_t$.

With a carefully coded Matlab implementation of L-BFGS, one iteration of EM takes about 100 seconds on sample data.

# Non-uniform $\alpha$ and $\beta$

Implementations of DCMLDA must allow the $\alpha$ vector and $\beta$ array to be non-uniform.

In DCMLDA, $\beta$ carries the information that $\varphi$ carries in LDA.

$\alpha$ could be uniform in DCMLDA, but ...

Learning non-uniform values allows certain topics to have higher overall probability than others.

# Experimental design

Question: Does handling burstiness in DCMLDA yield a better topic model than LDA?

Compare DCMLDA only with LDA for two reasons:
(1) Comparable conceptual complexity.
(2) DCMLDA is not in competition with more complex topic models, since those can be modified to include DCM topics.

# Comparison method

Given a test set of documents not used for training,
estimate likelihood $p(w|\alpha, \beta)$ for LDA and DCMLDA models.

For DCMLDA, use trained $\alpha$ and $\beta$.

For LDA, use scalar means of DCMLDA values: $\alpha = \bar{\alpha}$ and $\beta = \bar{\bar{\beta}}$.

Also compare to LDA with heuristic values $\beta = .01$ and $\alpha = 50/K$, where $K$ is the number of topics.

# Datasets

Compare LDA and DCMLDA on text and financial data.

Text dataset is 390 papers from NIPS 2002 and 2003, $m = |V| = 6871$, average length $n = 1336$.

S&P 500 dataset contains 501 days of stock price changes from January 2007 to September 2008. $m = 1000$.

# Digression: Computing likelihood

Incomplete likelihood $p(w|\alpha, \beta)$ is intractable for topic models.

Complete likelihood $p(w, z|\alpha, \beta)$ is tractable, so previous work has averaged it over $z$, but this approach is unreliable.

Another possibility is to measure classification accuracy.

But, many datasets do not have obvious classification schemes. Also, topics may be more accurate than predefined classes.

# Empirical likelihood (EL)

EL is a proxy for true incomplete likelihood.

(1) Train the intractable model.

(2) Generate many pseudo documents from the trained model.

(3) Use pseudo documents to train a tractable model (mixture of multinomials).

(4) Estimate likelihood of genuine test documents as their likelihood under the tractable model.

# Digression$^2$: Stability of EL

Investigate stability by running EL multiple times for the same DCMLDA model

Train three independent 20-topic DCMLDA models on the S&P500 dataset, and run EL five times for each model.

Mean absolute difference of EL values for the same model is 0.08%.

Mean absolute difference between EL values for separately trained DCMLDA models is 0.11%.

Conclusion: Out-of-sample EL likelihood values are reproducible.

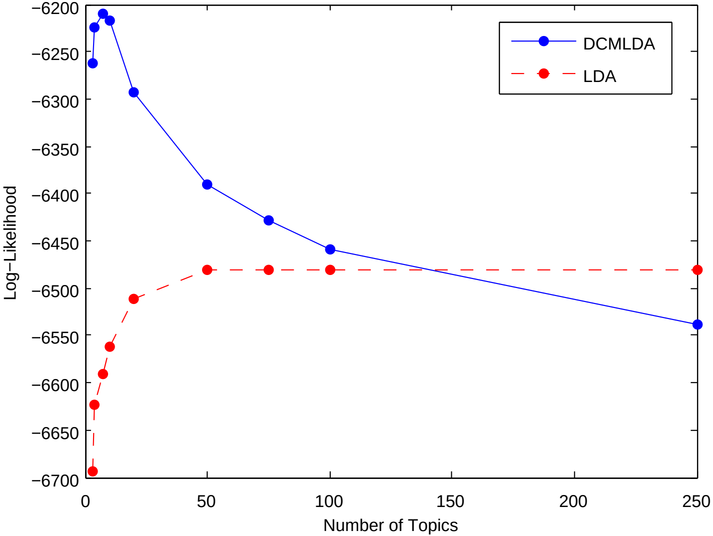
# Cross-validation

Perform five 5-fold cross-validation trials for each number of topics and each dataset.

First train a DCMLDA model, then create two LDA models.
“Fitted LDA” uses the means of the DCMLDA hyperparameters.
“Heuristic LDA” uses fixed parameter values.

Results: For both datasets, DCMLDA is better than fitted LDA, which is better than heuristic LDA.

Mean log-likelihood on the S&P500 dataset. Heuristic model likelihood is too low to show. Max. standard error is 11.2.



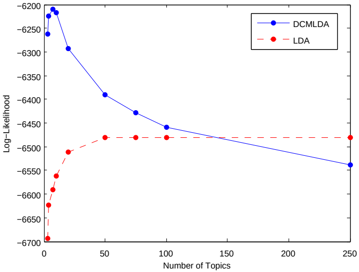
# S&P500 discussion

The best model is DCMLDA with seven topics.

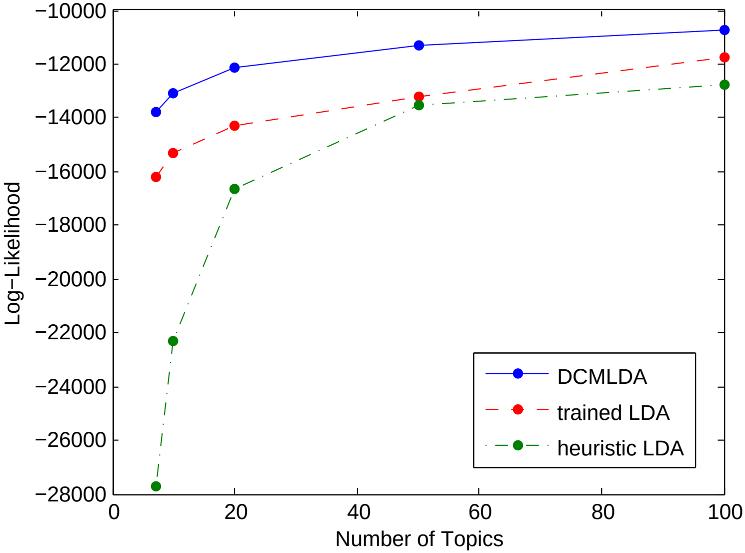
DCMLDA with few topics fits better than LDA with many topics.

Above seven topics, DCMLDA likelihood drops. Plausible explanation is overfitting.

LDA cannot generalize well regardless of how many topics are used.



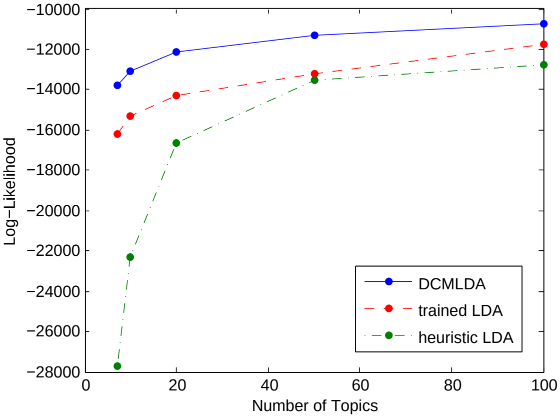
Mean log-likelihood on the NIPS dataset. Max. standard error 21.5.



# NIPS discussion

DCMLDA outperforms LDA model at every number of topics.

LDA with fixed hyperparameters performs worse than fitted LDA, except with 50 topics.



# Alternative heuristic values for hyperparameters

Learning $\alpha$ and $\beta$ is beneficial, both for LDA and DCMLDA models.

Optimal values are significantly different from previously suggested heuristic values.

Best $\alpha$ values around 0.7 seem independent of the number $K$ of topics, smaller than the suggested value $50/K$ for $K \leq 70$.

Smaller $\alpha$ means less concentration of topics.

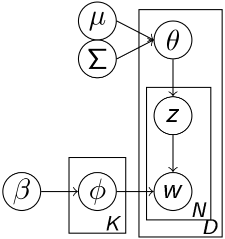
# Newer topic models

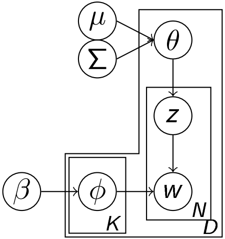
Variants include the Correlated Topic Model (CTM) and the Pachinko Allocation Model (PAM). These outperform LDA on many tasks.

However, DCMLDA competes only with LDA. The LDA core in other models can be replaced by DCMLDA to improve their performance.

DCMLDA and complex topic models are complementary.

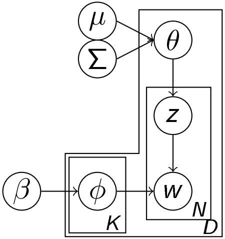
# A bursty correlated topic model





$\mu, \Sigma$ represent a multivariate Gaussian that makes topics correlated.

# A bursty correlated topic model



Difficulties:
(1) Covariance matrix $\Sigma$ has $O(K^2)$ parameters: far too many.
(2) Gaussians don't generate multinomial parameter vectors.
(3) Words are correlated within topics, e.g. Honda and Toyota.

Solution: Use generalized Dirichlet distributions for $\alpha$ and $\beta$.

Can still use stochastic EM with Gibbs sampling and L-BFGS for training.

# Conclusions

The ability of the DCMLDA model to account for burstiness gives major improvement in out-of-sample likelihood over LDA.

The burstiness of words, and of some non-text data, is an important phenomenon to capture in topic modeling.

In fact, burstiness can be important to model throughout supervised and unsupervised learning–see TFIDF.