# Understanding and Using Topic Modeling

## Using inferred document clusters

Philip Robinson

Presented to itds
NASA - Jet Propulsion Lab

September 7, 2018

Jet Propulsion Laboratory
California Institute of Technology

# Presentation Overview

# Intern - Philip (1762)

Computer Science MSc at Oregon Health and Science University.

Thats me!

# What is Topic Modeling

## Our goal: mathematically model topics from a corpus

*Topic modeling is a text processing technique for automatically grouping documents by topics. This is usually used as a strategy to describe documents in low dimensional space or an exploratory tool for document collections.*

# Examples

**In practice, this requires many more documents**

The Tourist huddles in the station While slowly night gives way to dawn ; He finds a certain fascination In knowing all the trains are gone.

The Governess up in the attic Attempts to make a cup of tea ; Her mind grows daily more erratic From cold and hunger and ennui.

The Journalist surveys the slaughter, The best in years without a doubt; He pours himself a gin and water and wonders how it came about.

- Food
- Travel
- Time

From this annotation we know that Document 2 and 3 are about Food and Time

# What can I solve?

- Find similar document pairs
- Cluster documents into groups with similar content
- Find relevant documents to a query or user's interests
- Explore shape of document collection

*Topic modeling can usually be extended to address many other problems, and document embeddings can be used to inform downstream models.*

# Applications of Topic Modeling

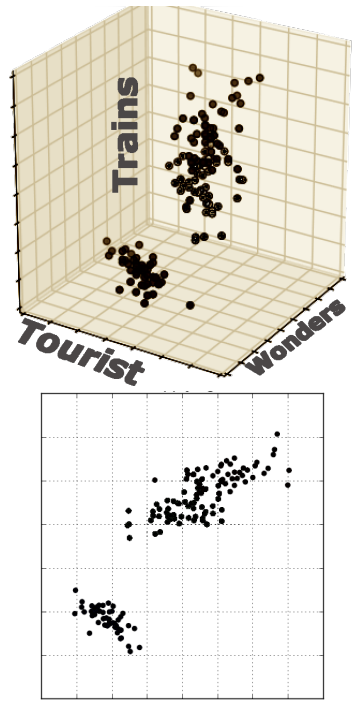



Figure: Dimmentionality Reduction

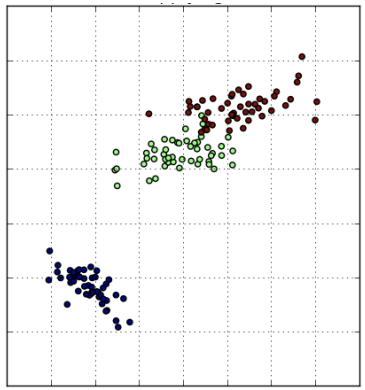


Figure: Cluster Points

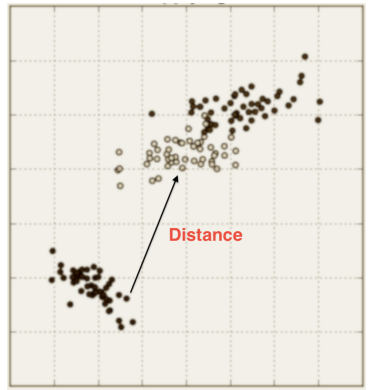



Figure: Exploratory Data Analysis

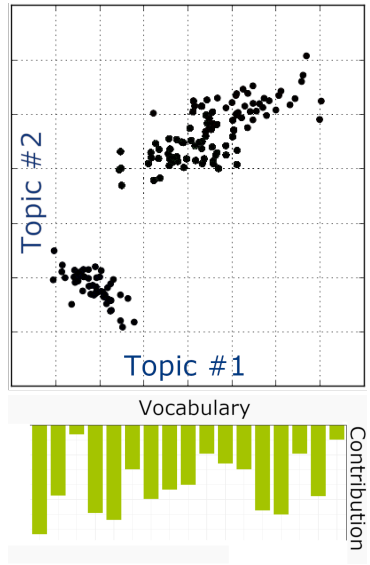



Figure: Analysis of Topics

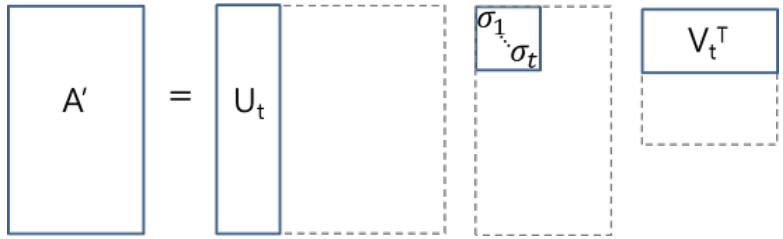
# Latent Semantic Analysis

## SVD on Vocabulary x Document matrix

**Given:** $D$ documents covering $W$ words

- Create $A_{DxW}$ counting or `tfidf`[1] matrix
- Compute Singular Value Decomposition
- Select the number of description topics $t$

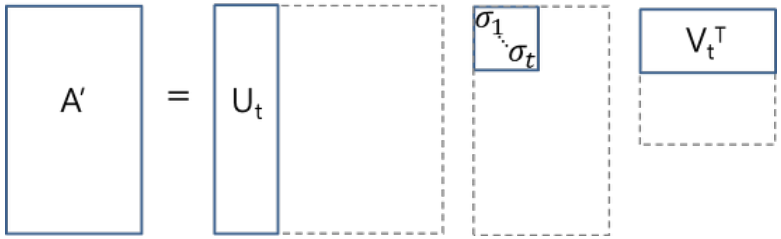
$$A' \approx U_t S_t V_t^T$$




[1] $a_{i,j} = tf_{i,j} \times log \frac{D}{df_i}$

# Understand Latent Semantic Analysis

**Topics are principle components of entire document collection**




**U:** document-topic matrix, topic contributes to document

**S:** singular values

**V:** word-topic matrix, topic contribute to words

- Overfit as consequence of topics strict mathematical definition
- Topics are better interpreted as mathematical than intuitive
- Cost of finding one topic is the same as finding all possible topics

# Goals of generative models

## A generative model

- Assume/Generalize how data could have been generated
- Fit distributions that describe generalization
- Ask questions about the generalization in relation to data
- Ask questions about data in relation to the generalization

*Generative models are much easier to extend, because they abstract the model from it's linear algebra dependencies.*

*Topic modeling generalizes how a document is generated by claiming that words come from topics, and documents have multiple topics.*[2]

[2] this is not a language model

# Probabilistic Latent Semantic Analysis

## Generative model for SVD

$$P(d, w) :\rightarrow \texttt{document-term matrix}$$

- $P(z|d)$ is the probability $z$ contributes to $d$
- $P(w|z)$ is the probability $w$ contributes to $z$

$$P(D, W) = P(D) \sum_{Z} P(Z|D)P(W|Z)$$

$$P(Z|D) \texttt{ and } P(W|Z) \sim \texttt{Multinomial}$$

# Understand Probabilistic Latent Semantic Analysis

## A mapping to SVD

$$P(D, W) = P(D) \sum_Z P(Z|D)P(W|Z)$$

$$= \sum_Z P(Z)P(D|Z)P(W|Z)$$

`remembering`

$$A \approx U_t S_t V_t^T$$

First generate the topic $Z$ then generate the word $W$

- P(D) is not parameterized, we don't observe new documents
- Tends to be softer than LSA, but still overfits (grows with D)
- No longer use `tfidf` best replaced with `stopwords`[3]

[3] Usually top .5 – 2% of vocab

# Latend Dirichlet Allocation

## Bayesian extension to PLSA

- Represent document as Bag-of-Words[4]
- Model/Fit topics as mixture of words
- Documents are projected into or sampled from topic-space-distribution

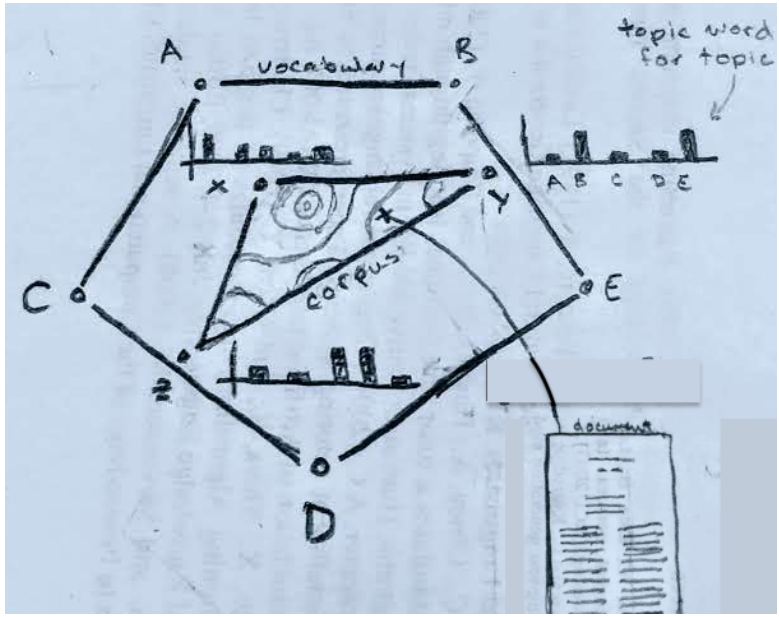



Figure: Latent Dirichlet Allocation

*Enormous body of work extending this model to address more specific problems.*

[4] equivalent to multinomial over the vocabulary

# Dirichlet Distribution

**In this case the Topic-Space is our Dirichlet Distribution**

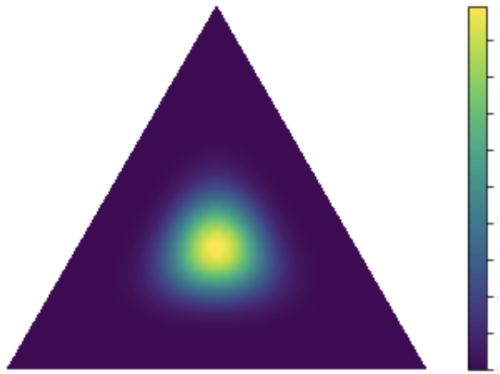


Figure: Non-Informative

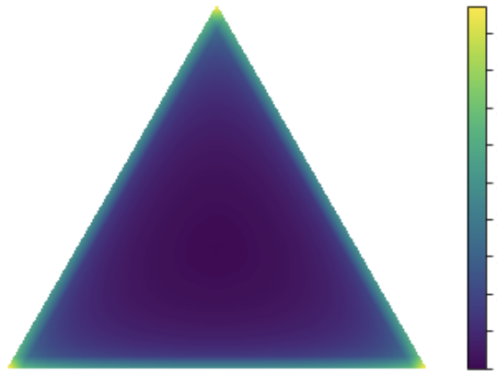


Figure: Little In Common

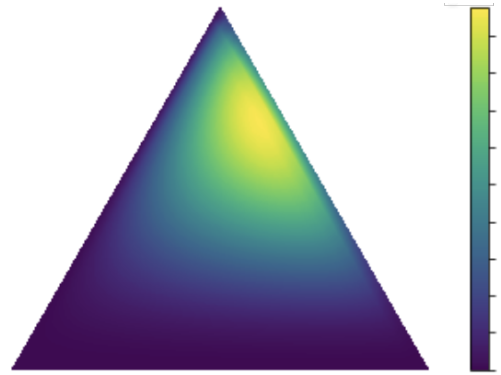


Figure: Shared Topics

# Using this in Python[5]

```
from nltk.corpus import brown

data = []

for fileid in brown.fileids():
    document = ' '.join(brown.words(fileid))
    data.append(document)

NO_DOCUMENTS = len(data)
print(NO_DOCUMENTS)
print(data[:5])

from sklearn.decomposition import LatentDirichletAllocation
from sklearn.feature_extraction.text import CountVectorizer

NUM_TOPICS = 10

vectorizer = CountVectorizer(min_df=5, max_df=0.9,
                             stop_words='english', lowercase=True,
                             token_pattern='[a-zA-Z\-][a-zA-Z\-]{2,}')
data_vectorized = vectorizer.fit_transform(data)

# Build a Latent Dirichlet Allocation Model
lda_model = LatentDirichletAllocation(n_topics=NUM_TOPICS, max_iter=10, learning_method='online')
lda_Z = lda_model.fit_transform(data_vectorized)

text = "The economy is working better than ever"
x = lda_model.transform(vectorizer.transform([text]))[0]
```

[5] example in `scikit-learn`, I used `gensim`

# Looking at top words

## Mitigating apophenia is hard, topics difficult to interpret

Topic #1

- server
- connected
- access
- workstation
- outage
- user

Topic #2

- mode
- instrument
- safe
- spacecraft
- anomaly
- recovery

Topic #3

- uplink
- station
- dsn
- spacecraft
- lock
- ace

*Although the model better describes our generation process, from the perspective of topics, it can be difficult to know what these topics actually represent.*[6] *This may require experts who are immune to apophenia.*

[6] Supervised LDA attempts to addresses this concern, also applies to sentiment analysis

# Evaluation

## Does our fit dirichlet distribution describe our data or our understanding

- perplexity
- coherence
- visualization
- predictive power

# Perplexity vs Coherence

**perplexity for prediction, coherence for EDA**[8]

*Perplexity measures how poorly the model describes the data.*

$$Perplexity(q) = b^{-\frac{1}{N}\sum_{x \in X} log_b q(x)}$$

*Topic coherence measures take the set of N top words from a topic and sums a* `confirmation measure`[7] *over the word pairs. Probabilities are estimated from sliding window over train and test corpora.*

$$C_{Irvine} = \frac{2}{N \cdot N - 1} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} PMI(w_i, w_j)$$

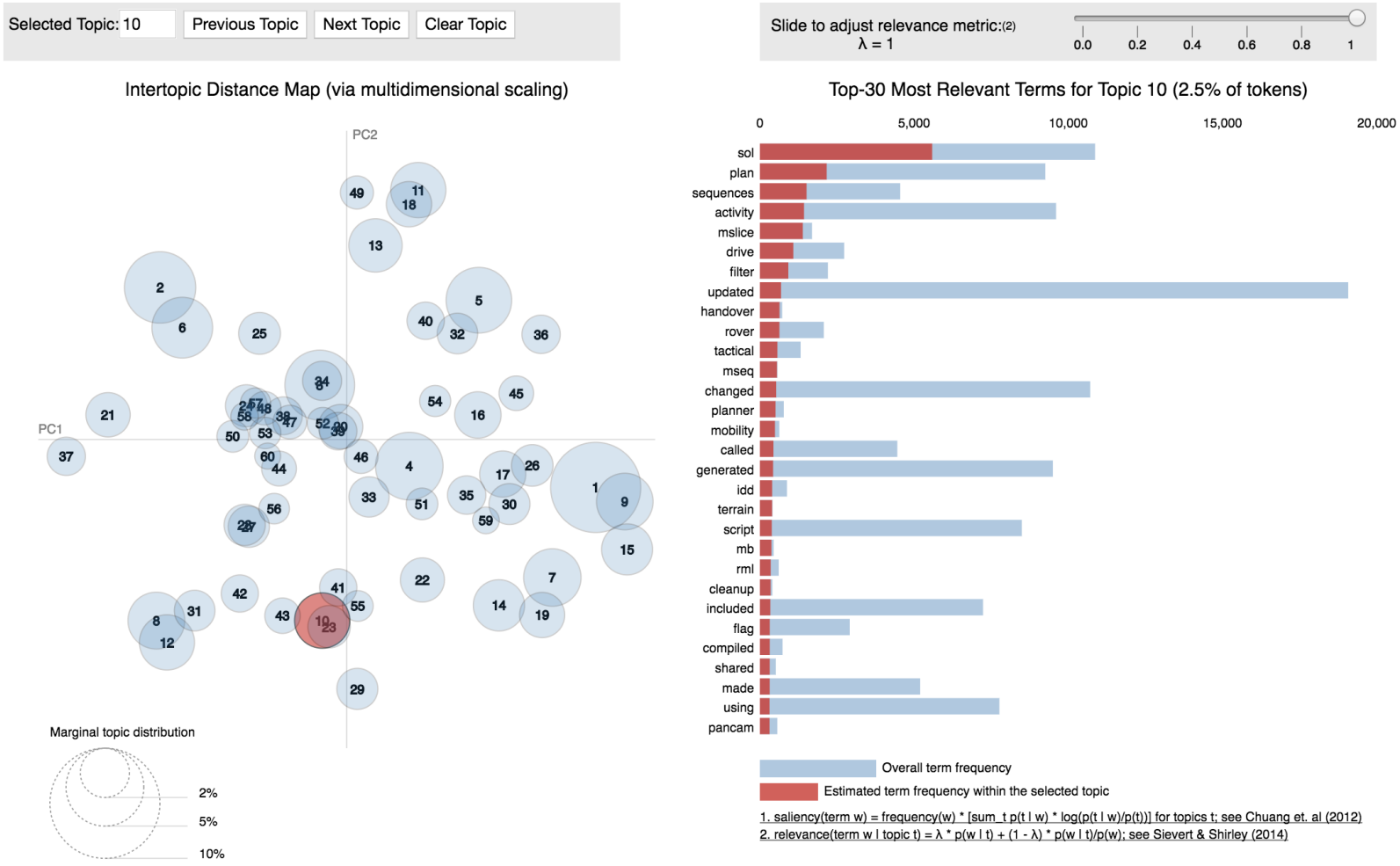
$$PMI(w_i, w_j) = log(\frac{P(w_i, w_j)}{P(w_i) \cdot P(w_j)})$$

[7] like pointwize mutual information (PMI)

[8] exploratory data analysis

# Interactive Visualization[9]

## Breakout to jupyter




[9] used `pyLDAvis`

# Predictive Power

## Depends on your downstream applications

*Does your client end up benefiting from the tool. This could be many measures like recall or clickthrough.*

# Extensions

- Correlated Topic Model
- → Author Topic Model
- Biterm Topic Model
- Hierarchical Dirichlet Process

# Distance Measures

## Testing distances is cheaper than understanding them

*Hellinger distance is a distance between probability distributions. The domain of the Dirichlet distribution can be thought of as a simplex over multinomial distributions.*[10]

[10] Most online examples use cosine distance, without justification

# Text Pre-Processing

## Cleaning applies to most 'simple' NLP problems

*Text normalization is custom to your corpus. Many of the steps are the same, but their application changes with the type of documents.*

- Normalize text
  - Lowercase text
  - ⋆ Remove non-informative text patterns
- Tokenization & (Stemming — Lemmatization)
  - ⋆ pick a stemmer
  - Stem `(applies, applying, apply) -> (appli)`
  - ⋆ Un-Stem `(appli) -> (apply)`
- Focus corpus (remove "stop words")
  - drop most frequent words
  - `nltk` english stop-words
  - Remove extremely rare words

# Non-Informative Delinquent Cases

## Evaluation metrics are only informative given reasonable parameters

*You can often reduce perplexity by having fewer topics. Maximizing coherence is more resilient to this effect.*

# Verifying your intent

## You may not need interpretable topics!

*Base LDA, on it's own, isn't that great. Understanding LDA allows you to understand the extensions, which are pretty cool.*

*Not all evaluation metrics have been written for the extensions, so you may have to come up with proxies.*

# Stability of Topics

*If you are extremely dependent on understandability, then you may need to incorporate model stability.*[11]

[11] http://doi.acm.org/10.1145/2954002

# Take Away

- Perform text level EDA to customize cleaning processing
- Pick a model type
- Evaluation takes care
  - Identify a model-fit measure
  - Identify a performance strategy

*Simply put, LDA attempts to generalize truncated SVD with a generative bias to how we write papers.*

# Conclusion & Questions

*I used the Author Topic Model during my internship to automatically assign tickets to subject matter experts, for the Office of Safety and Mission Success (5x)*