Feature LDA: a Supervised Topic Model for Automatic Detection of Web API Documentations from the Web

Chenghua Lin, Yulan He, Carlos Pedrinaci, and John Domingue
Knowledge Media Institute, The Open University
Milton Keynes, United Kingdom

11th International Semantic Web Conference. Boston, USA, 2012

Outline

- Introduction
- The Feature LDA model
- Data
- Experimental Results
- Conclusion

Introduction

- What are Web APIs?
  - New Web Services based on a simple stack of technologies, ≈ "URL+HTTP+XML/JSON"
  - Known as RESTful service when conforming the REST principles
- Why Web APIs?
  - Light technology stack VS. "classical" Web services (WSDL, SOAP, WS-*)
  - Enable easy access and aggregation of collection of resources
  - Widely used and reused

Finding a Web API

- Dedicated registries, e.g. ProgrammableWeb
  - Contain out of date or incorrect information, e.g. invalid pages or incorrect links to APIs documentation pages
  - Only a limited number of Web APIs listed, left out a large number of third party Web APIs
- General search engine, e.g. Google
  - Not optimized for Web API discovery
  - Mix up with pages that are not (so) relevant, e.g. blogs and advertisement about Web APIs.

Welcome
Register
SUBSCRIBER SERVICES
Profiles
Statistics
Data
FREE DATA SERVICES
Sentiment

Developer Center

Sentiment Analysis API Methods

Viralheat is a social media measurement platform that allows you to track topics, brands, television shows, and movies across many different mediums all in one place using the best technology. The users of Viralheat choose to share a subset of their profiles with the world. Sentiment Analysis API is freely available to everyone with a developer account.

Available Methods

Get Sentiment | Train the Sentiment Analysis Engine | Check Sentiment API usage quota

Method 1: Get Sentiment

URL (JSON): `http://www.viralheat.com/api/sentiment/review.json?api_key=API_KEY&text=i%20dont%20like`

URL (XML): `http://www.viralheat.com/api/sentiment/review.xml?api_key=API_KEY&text=i%20dont%20like`

Format: **XML, JSON**

HTTP Method: **GET**

Requires Authentication: **Yes**

Parameters:

Name	Description
api_key	The authenticated account key.
text	Text for which you want the sentiment. The limit on the text is 360 characters.

Return Values:

Name	Description
prob	The probability with which the system thinks the given text has the output sentiment.
mood	Sentiment output for the given text.
text	Input text for which the sentiment was generated.

(http://www.clarabridge.com/), Lymbix (http://www.lymbix.com/), etc.

Our Goal …

- **Goal**: To build a customized search engine for detecting third party Web APIs on the Web scale
  - Assume every Web API provides public documentation page(s)
  - These pages provide the most relevant information for developers
  - Approached as a binary classification problem, i.e. distinguishing API documentation VS. normal pages

Easy Task?

- **Issues**
  - No simple way to effectively and uniquely identify Web APIs
  - described in plain and unstructured HTML highly heterogeneous in format and contents, i.e. NO Gold standard
    - People hardly follow even there is !!!
  - More than 99% of pages on the Web are **NOT** relevant to Web API
    - Need a high precision classifier yet maintaining good accuracy

Our Approach

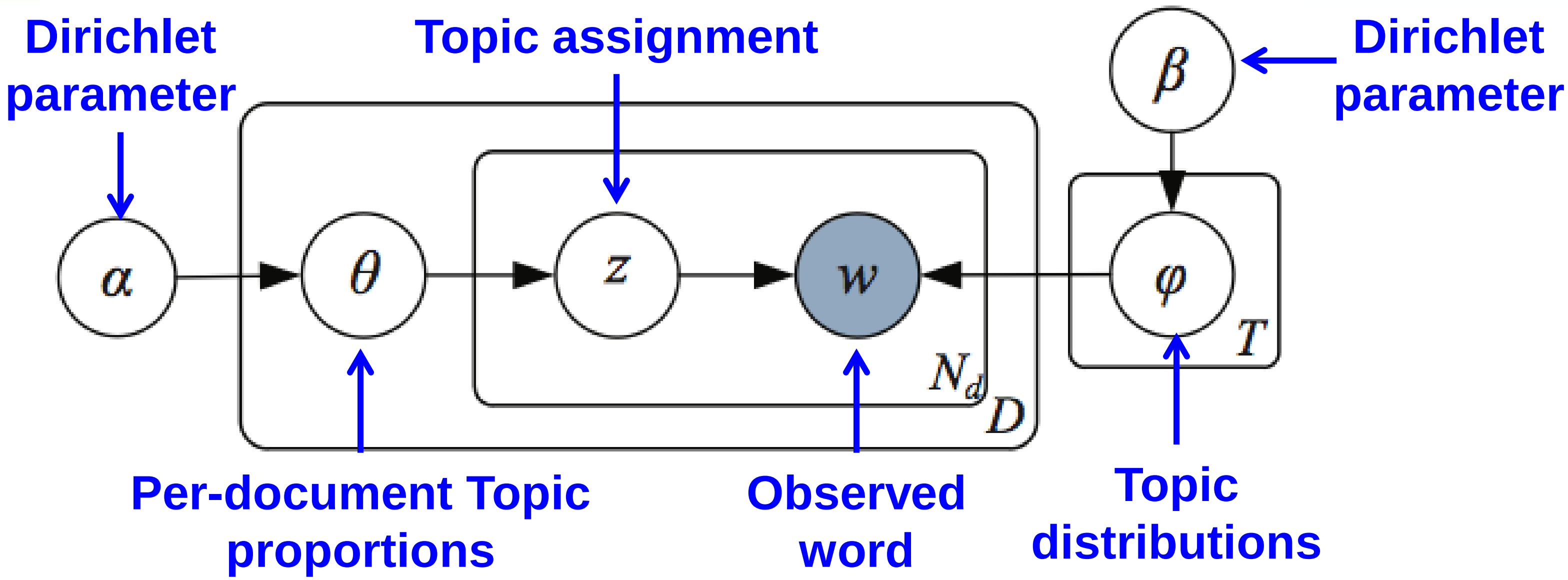
The Feature LDA model

Latent Dirichlet Allocation (LDA)

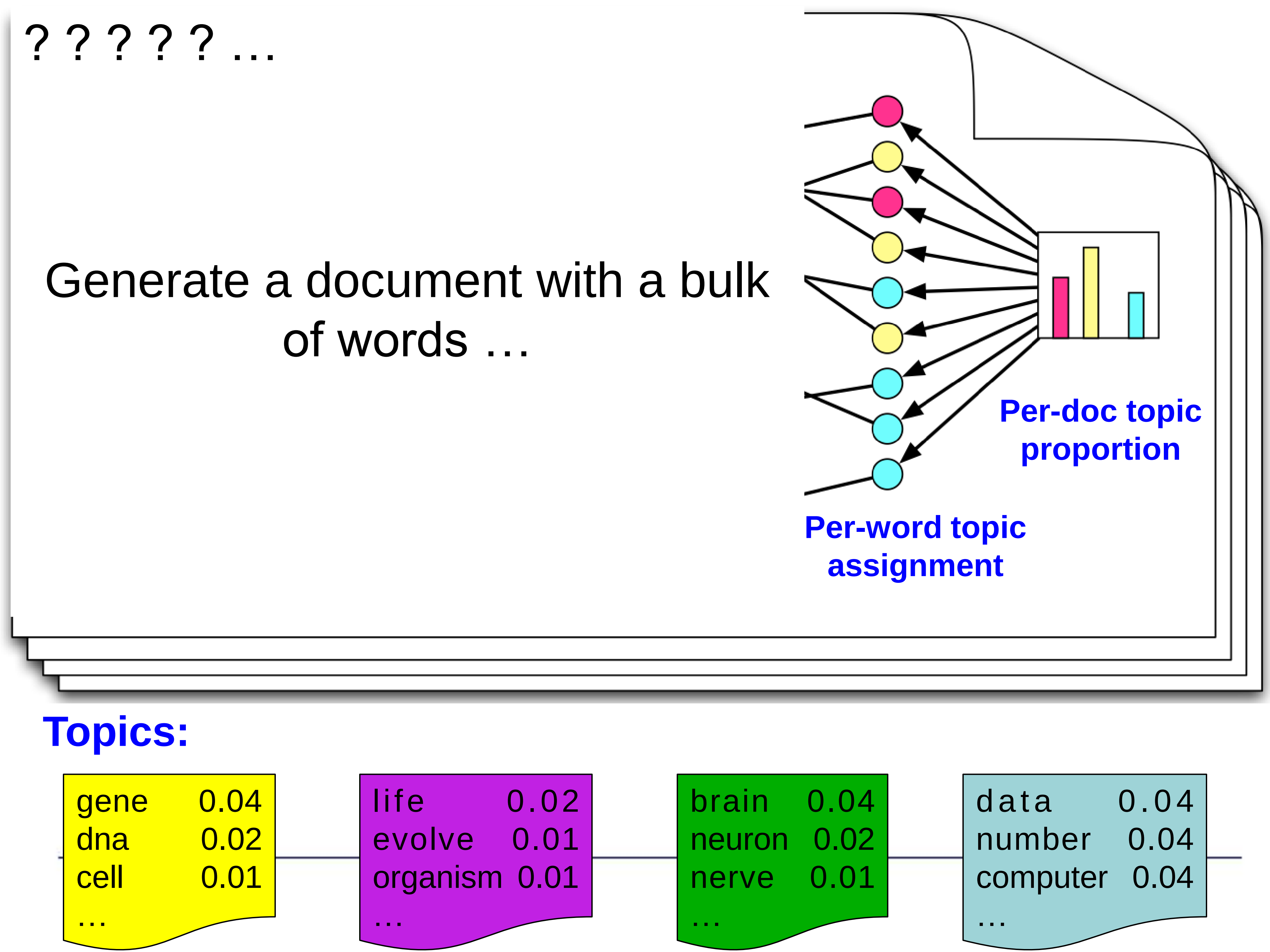
- LDA: the simplest form of topic models
  - A fully unsupervised Bayesian model
  - Assumes that documents exhibit multiple to known as "**theme**" or "**gist**")
  - Each topic is a distribution over words whic tight semantic relation with one another

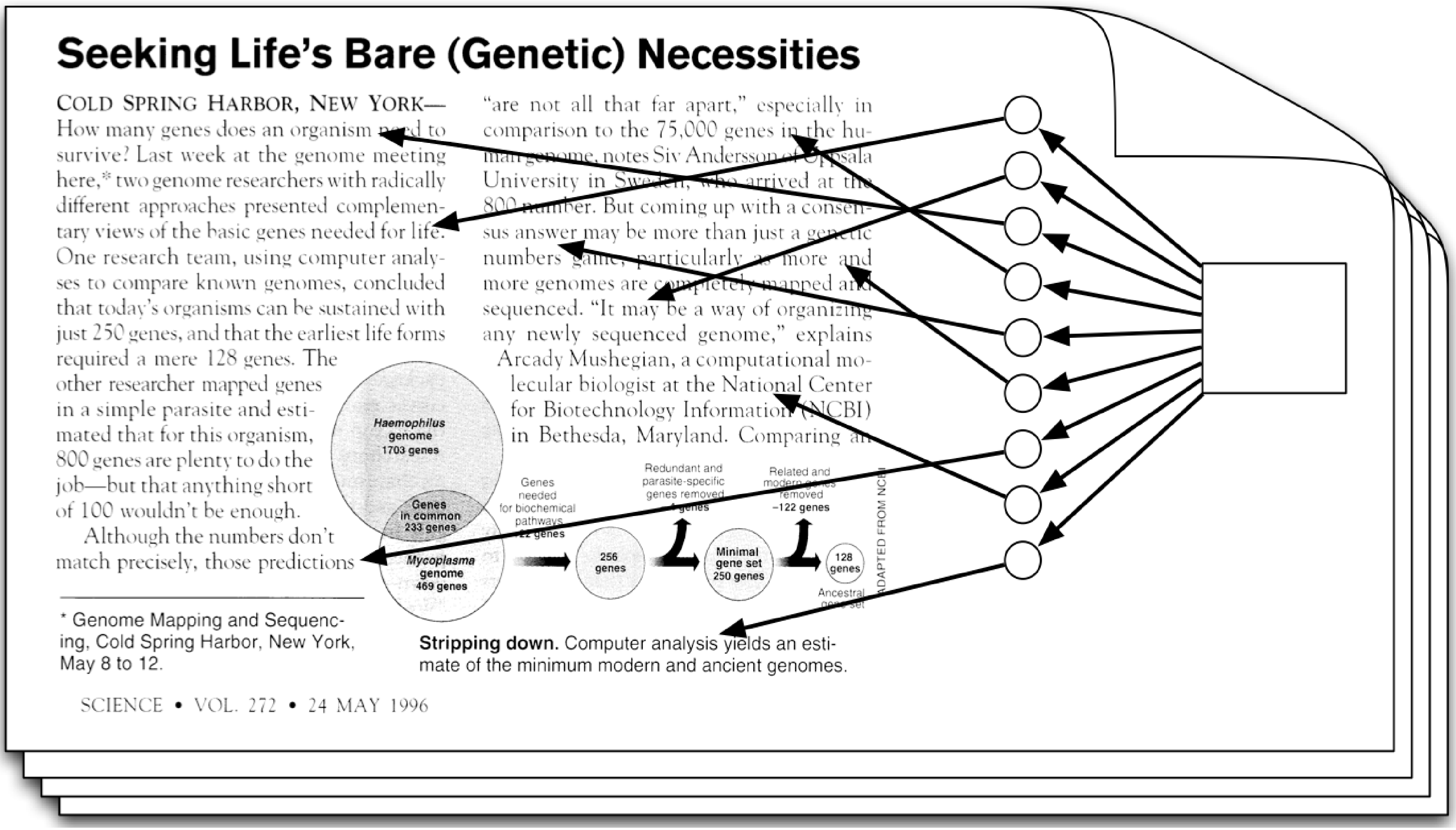
gene
dna
cell
sequence
genetics
mapping
human
molecular
…

Graphical Model



- Intuition:
  - Each document exhibit multiple topics
  - Each topic is a distribution over words
  - Each word is drawn from one of those topics


? ? ? ? ? ...
Generate a document with a bulk of words …
Per-doc topic proportion
Per-word topic assignment
Topics:
gene 0.04
dna 0.02
cell 0.01
…
life 0.02
evolve 0.01
organism 0.01
…
brain 0.04
neuron 0.02
nerve 0.01
…
data 0.04
number 0.04
computer 0.04
…



Seeking Life's Bare (Genetic) Necessities

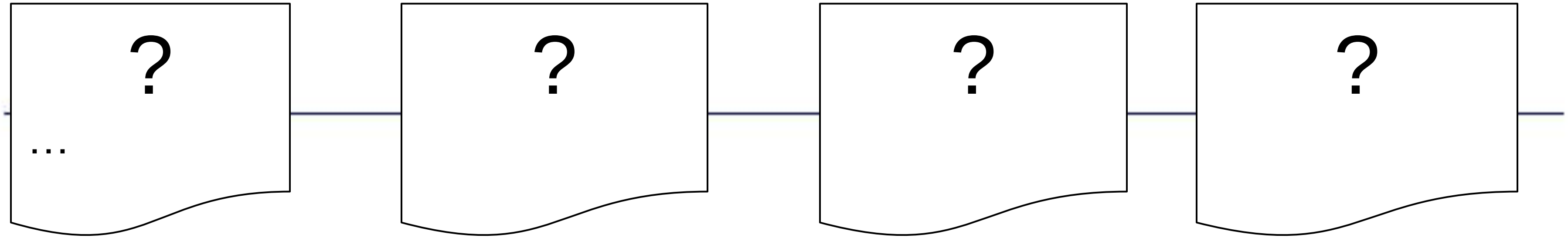
COLD SPRING HARBOR, NEW YORK— How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an

Stripping down. Computer analysis yields an estimate of the minimum modern and ancient genomes.

* Genome Mapping and Sequencing, Cold Spring Harbor, New York, May 8 to 12.

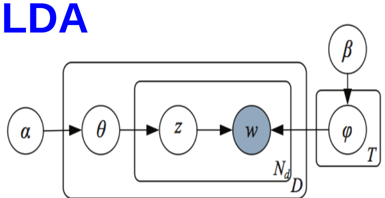
Topics:



Applications

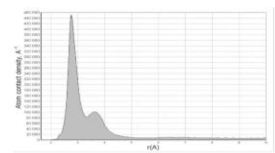
$$p(\beta,\theta,\mathbf{z},\mathbf{w}) = \left(\prod_{i=1}^{K} p(\beta_i \mid \eta)\right)\left(\prod_{d=1}^{D} p(\theta_d \mid \alpha)\prod_{n=1}^{N} p(z_{d,n} \mid \theta_d)\, p(w_{d,n} \mid \beta_{1:K}, z_{d,n})\right)$$

LDA



Data

Posterior inference



Classification

Information retrieval

Google

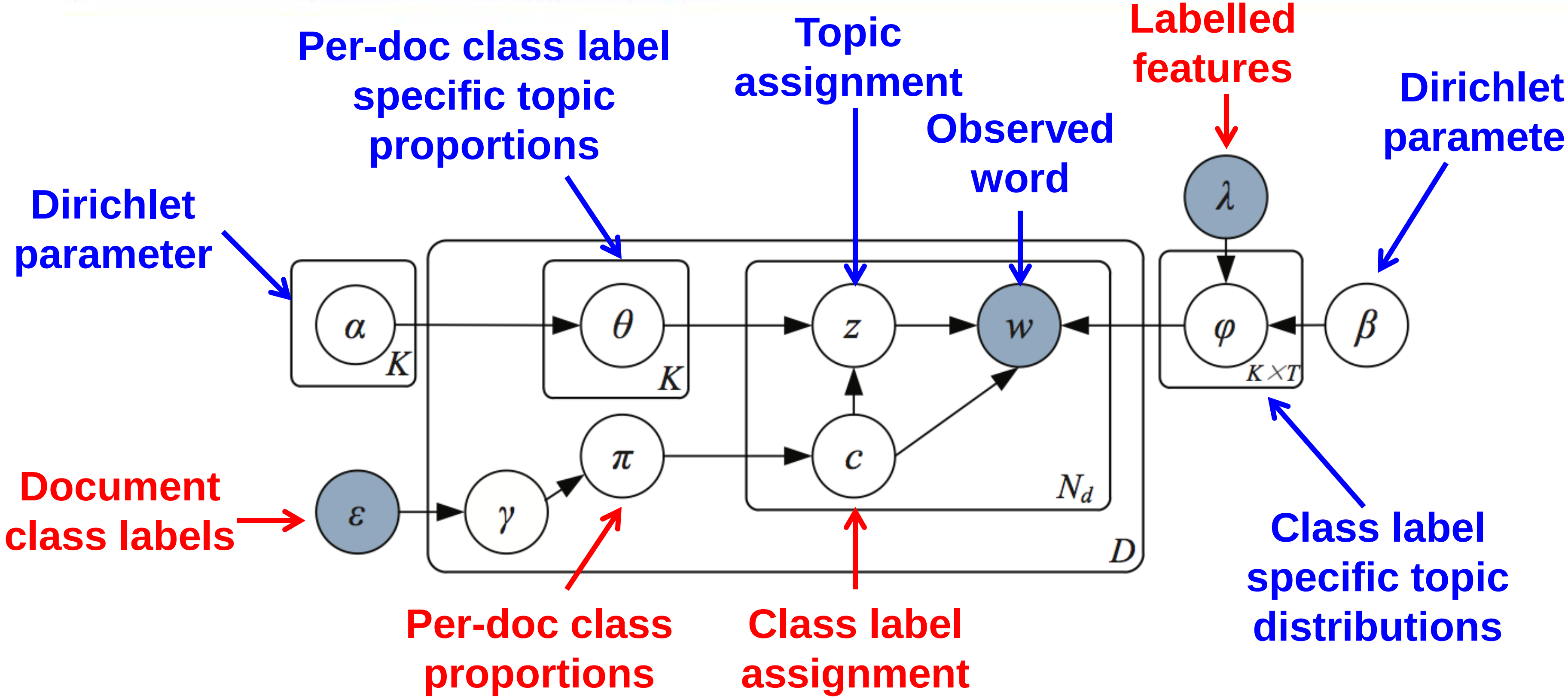
Recommendation

Data exploration

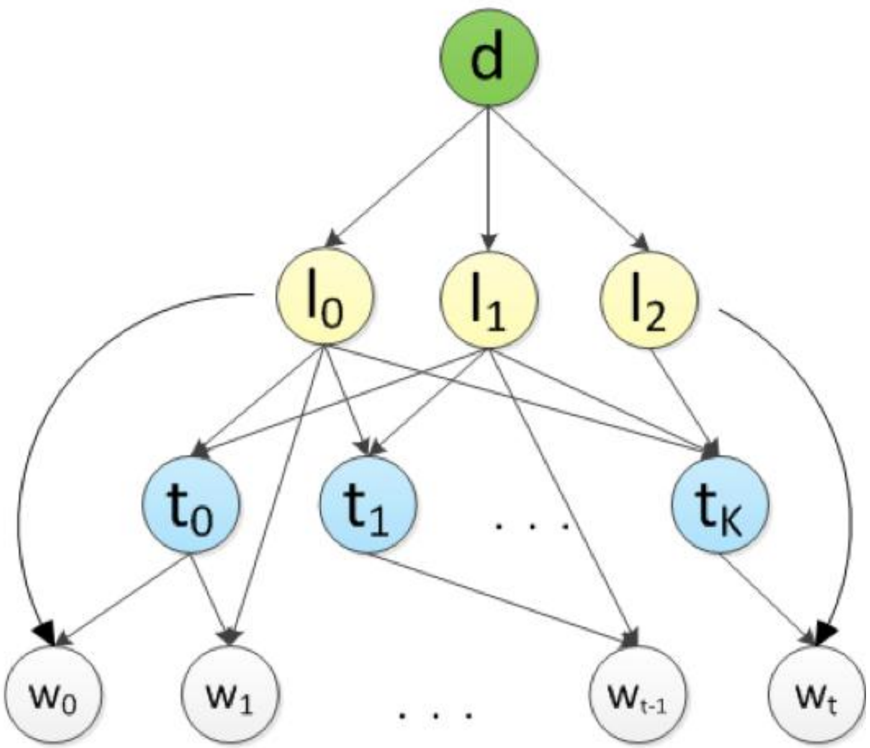
Feature LDA

- **Feature LDA model**: a generic probabilistic framework for text classification.
  - A supervised four-layer hierarchical Bayesian model
  - Accommodate supervisions from both labelled instance and labelled features for training
  - Able to extract meaningful class specific topics
- Labelled features
  - In *baseball* vs. *hockey* text classification
    - pitcher → *baseball*, puck → *hockey*
  - learned automatically from training data using any feature selection method, e.g. Info Gain

Feature LDA Graphical Model



Generative Process



For each document *d*

- Draw $\pi_d \sim Dir(\gamma \times \varepsilon_d)$
- For each class label *k*, draw $\theta_{d,k} \sim Dir(\alpha_k)$

For each word *w* in d

- Draw a class label $l_i \sim Mult(\pi_d)$
- Draw a topic $z_i \sim Mult(\Theta_{d,li})$
- Draw a word $w_i \sim Mult(\Phi_{li,zi})$

Inference

- Collapse Gibbs sampling for model posterior estimation

$$P(z_t = j, c_t = k | \mathbf{w}, \mathbf{z}^{-t}, \mathbf{c}^{-t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \propto \frac{N^{-t}_{k,j,w_t} + \beta_{k,j,t}}{N^{-t}_{k,j} + \sum_i \beta_{k,j,i}} \cdot \frac{N^{-t}_{d,k,j} + \alpha_{k,j}}{N^{-t}_{d,k} + \sum_j \alpha_{k,j}} \cdot \frac{N^{-t}_{d,k} + \gamma_k}{N^{-t}_d + \sum_k \gamma_k}.$$

- Approximating model parameters

$$\varphi_{k,j,i} = \frac{N_{k,j,i} + \beta_{k,j,i}}{N_{k,j} + \sum_i \beta_{k,j,i}} \qquad \theta_{d,k,j} = \frac{N_{d,k,j} + \alpha_{k,j}}{N_{d,k} + \sum_j \alpha_{k,j}} \qquad \pi_{d,k} = \frac{N_{d,k} + \gamma_k}{N_d + \sum_k \gamma_k}$$

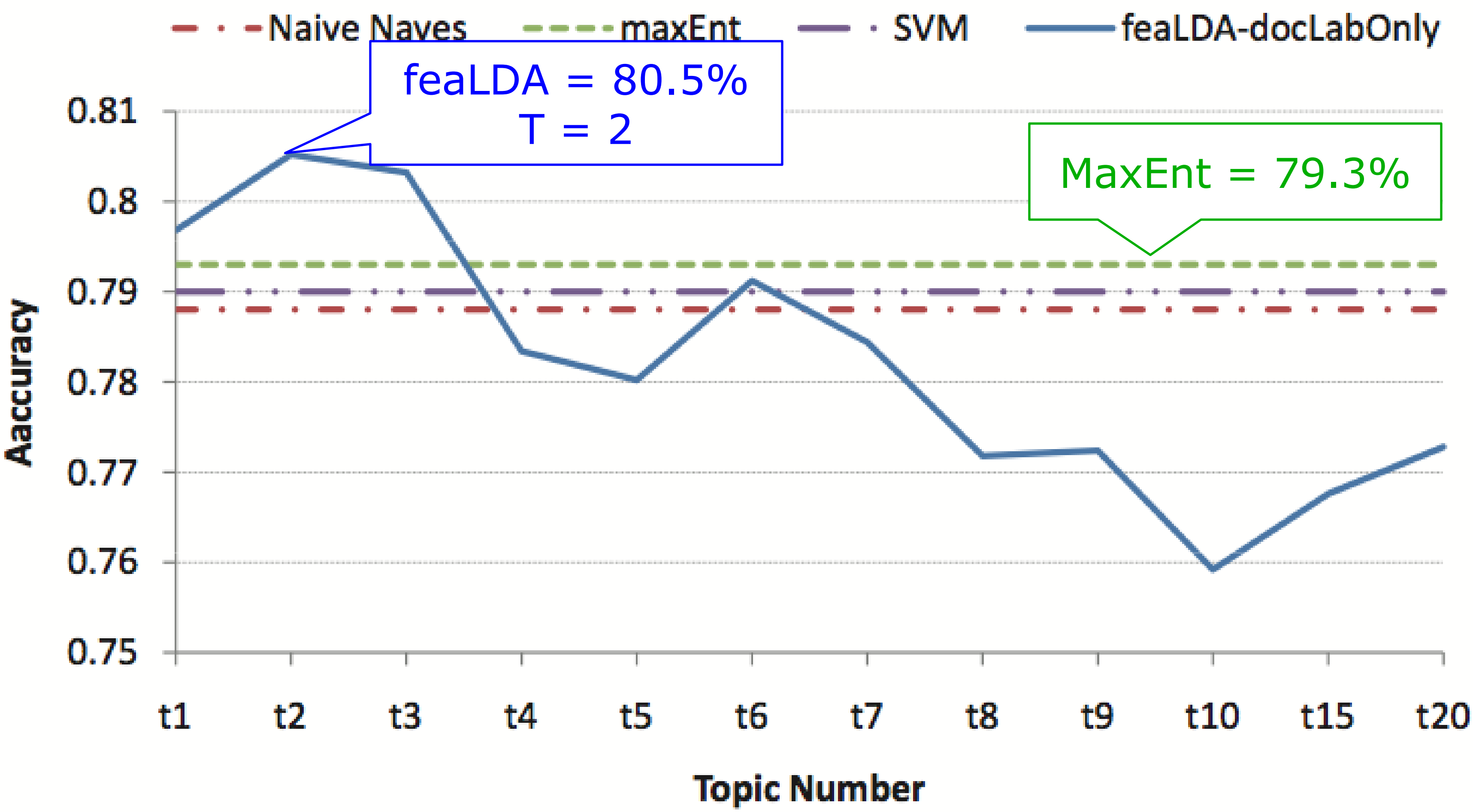
Data and Setup

- The API dataset: **1,547** Web pages crawled from the API Home URLs of ProgrammableWeb (manually labelled, training/testing split: 80%-20%)
  - **622** pages are API documentations
  - **925** pages are normal Web pages
- Preprocessing
  - Extract content from HTMLs by discarding tags and java scripts that are not relevant to classification
  - remove wildcards, non-alphanumeric characters and stop-words, followed by Porter stemming.
- Setup
  - Class label k = 2
  - Topic number T=1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20
  - 29,000 labelled features (Info Gain)

Experimental Results

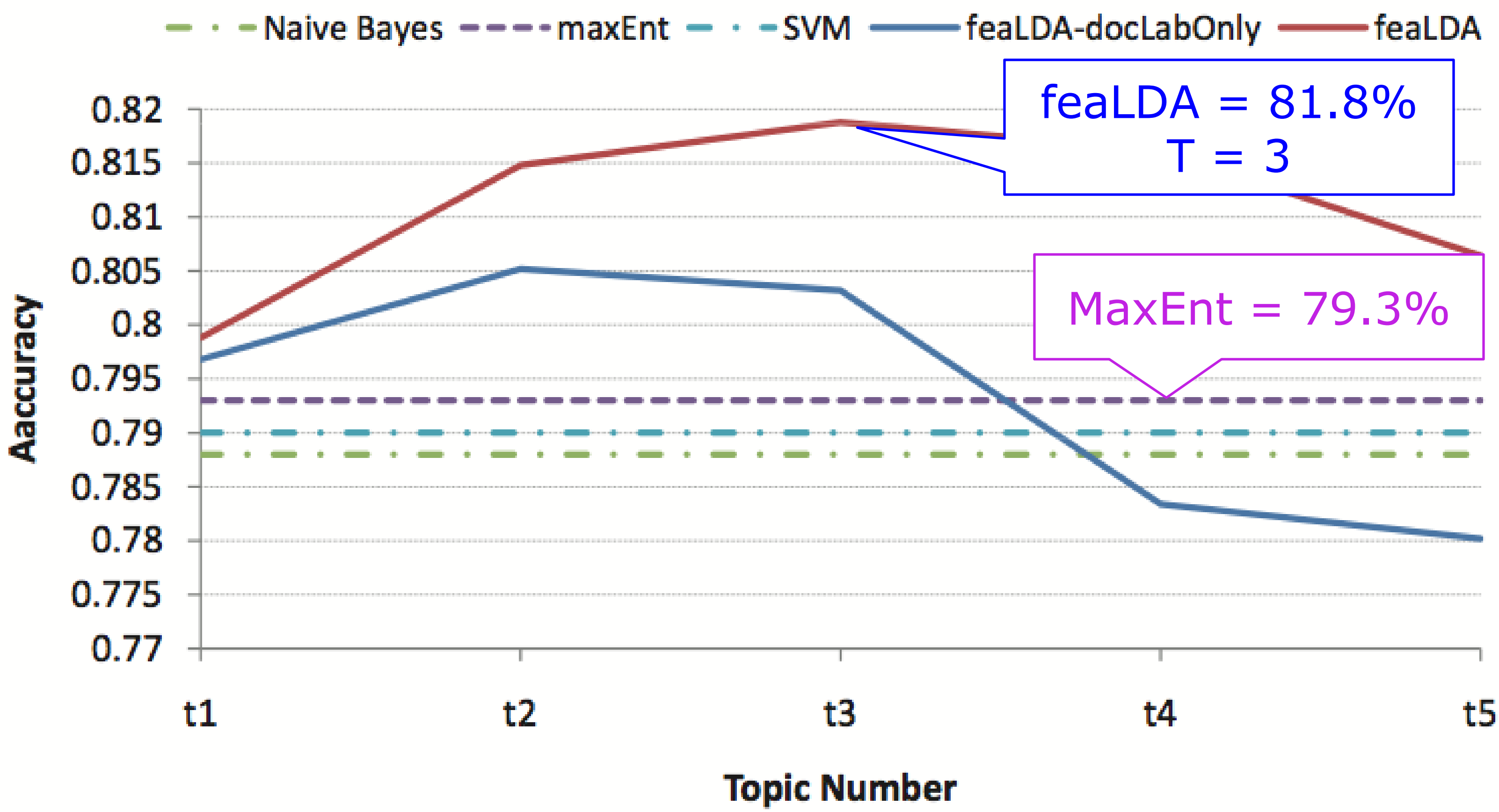
- We report feaLDA classification results on the API dataset with different model settings:
  - Training with labelled instances
  - Training with both labelled instances and labelled features
  - feaLDA performance feature selection on labeled features
  - feaLDA vs. baselines (NB, SVM, MaxEnt) and other supervised topic models (labelLDA, pLDA)
  - Topic extraction

Training with labelled Instances

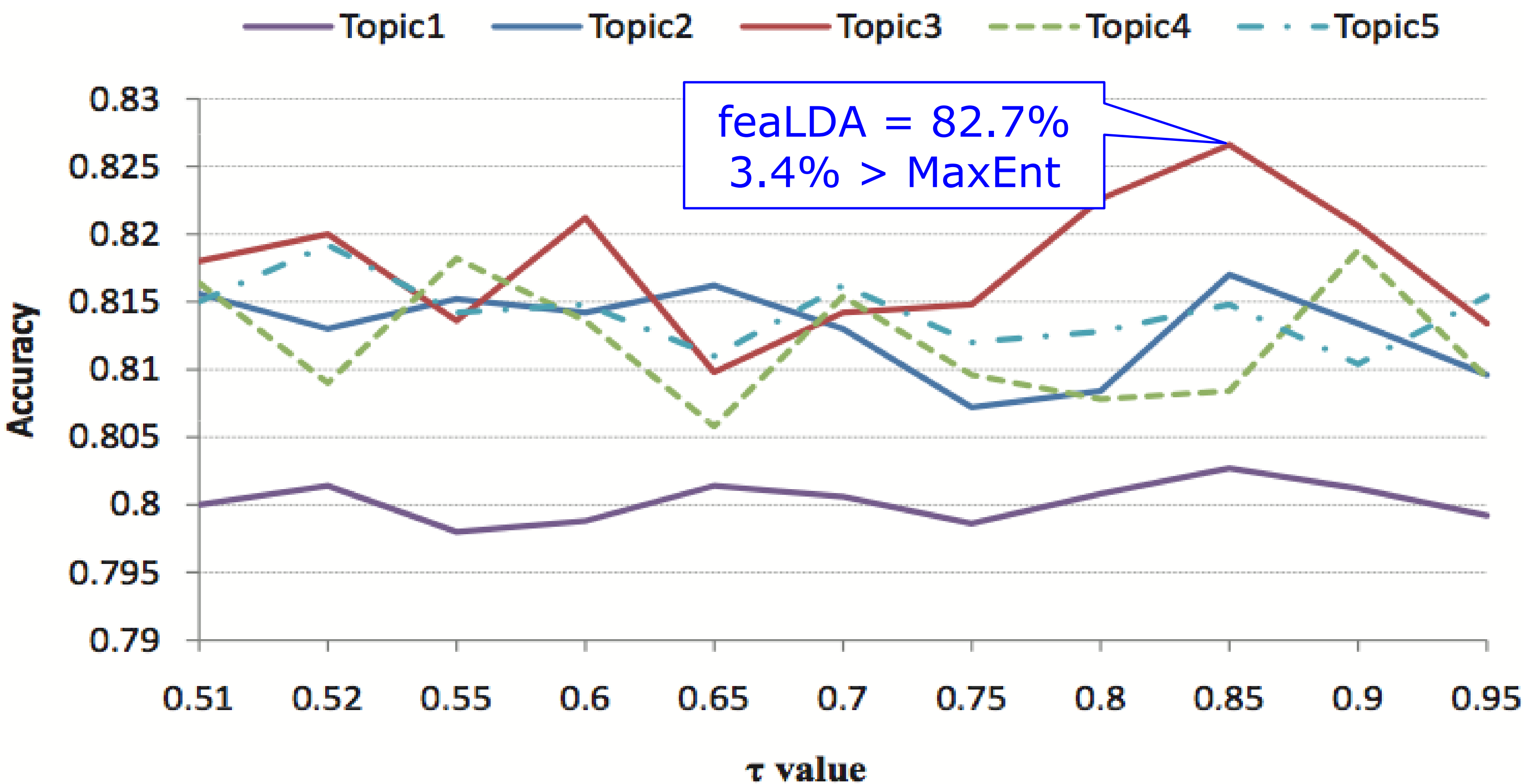


-

Labelled instances + labelled features



Feature Selection



- feaLDA classification accuracy vs. different feature class probability threshold τ.

Overall comparison

Table 2: Comparing feaLDA with existing supervised approaches.

	Naive Bayes	SVM	maxEnt	labeled LDA	pLDA	feaLDA
Recall	**79.2**	70.8	69.3	59.8	65.9	68.8
Precision	71.0	75.4	77.4	85.1	82.1	**85.2**
F1	74.8	73.1	73	70.2	73.1	**76**
Accuracy	78.6	79	79.3	79.8	80.5	**82.7**

- feaLDA vs. the state-of-the-art models
  - outperforms three strong supervised baselines
  - better than labeledLDA and pLDA for more than 3% in accuracy
  - gives very high precision: essential for reducing false positives when mining from the Web

Topic Extraction

Table 3: Topics extracted by feaLDA with $K = 2, T = 3$.

Positive	T1: nbsp quot gt lt http api amp type code format valu json statu paramet element
	T2: lt gt id type http px com true url xml integ string fond color titl date
	T3: api http user get request url return string id data servic kei list page paramet
Negative	T1: px color font background pad margin left imag size border width height text div thread
	T2: servic api site develop data web user applic http get amp email contact support custom
	T3: obj park flight min type citi air fizbber airlin stream school die content airport garag

- Topics with true API label
  - Terms are fairly technical, e.g. *json*, *statu*, *paramet*, *element*, *valu*, *request* and *string*, etc.
- Topics with false API label
  - Terms are less technical and more diverse, e.g. *contact*, *support*, *custom*, *flight*, *school*

Conclusions

- Discovering Web APIs is becoming increasingly important and existing support is not optimal
- Treat Web API discovery as a classification problem
- Presented a supervised topic model called feaLDA
  - offers a generic framework for text classification
  - Capable to encode supervision from both labelled instance and labelled features
  - Offers very high precision which is crucial for reducing false positive when mining from the Web
  - Able to extract class label specific topics

Questions?

Email: chenghua.lin@open.ac.uk