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, MK7 6AA, UK
{c.lin,y.he,c.pedrinaci,j.b.domingue}@open.ac.uk

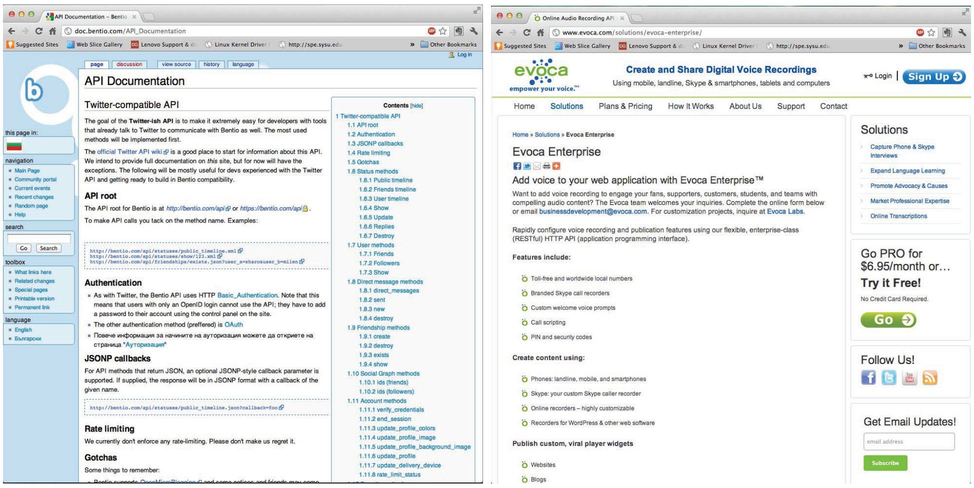
Abstract. Web APIs have gained increasing popularity in recent Web service technology development owing to its simplicity of technology stack and the proliferation of mashups. However, efficiently discovering Web APIs and the relevant documentations on the Web is still a challenging task even with the best resources available on the Web. In this paper we cast the problem of detecting the Web API documentations as a text classification problem of classifying a given Web page as Web API associated or not. We propose a supervised generative topic model called feature latent Dirichlet allocation (feaLDA) which offers a generic probabilistic framework for automatic detection of Web APIs. feaLDA not only captures the correspondence between data and the associated class labels, but also provides a mechanism for incorporating side information such as labelled features automatically learned from data that can effectively help improving classification performance. Extensive experiments on our Web APIs documentation dataset shows that the feaLDA model outperforms three strong supervised baselines including naive Bayes, support vector machines, and the maximum entropy model, by over 3% in classification accuracy. In addition, feaLDA also gives superior performance when compared against other existing supervised topic models.

1 Introduction

On the Web, service technologies are currently marked by the proliferation of Web APIs, also called RESTful services when they conform to REST principles. Major Web sites such as Facebook, Flickr, Salesforce or Amazon provide access to their data and functionality through Web APIs. To a large extent this trend is impelled by the simplicity of the technology stack, compared to WSDL and SOAP based Web services, as well as by the simplicity with which such APIs can be offered over preexisting Web site infrastructures [15].

When building a new service-oriented application, a fundamental step is discovering existing services or APIs. Main means used nowadays by developers for locating Web APIs are searching through dedicated registries like ProgrammableWeb[1] which are manually populated or to use traditional search en-

[1] http://www.programmableweb.com/



API Documentation

Twitter-compatible API

The goal of the **Twitter-ish API** is to make it extremely easy for developers with tools that already talk to Twitter to communicate with Bentio as well. The most used methods will be implemented first.

The official Twitter API wiki is a good place to start for information about this API. We intend to provide full documentation on *this* site, but for now will have the exceptions. The following will be mostly useful for devs experienced with the Twitter API and getting ready to build in Bentio compatibility.

API root

The API root for Bentio is at *http://bentio.com/api* or *https://bentio.com/api*.

To make API calls you tack on the method name. Examples:

```
http://bentio.com/api/statuses/public_timeline.xml
http://bentio.com/api/statuses/show/123.xml
http://bentio.com/api/friendships/exists.json?user_a=sharo&user_b=milen
```

Authentication

- As with Twitter, the Bentio API uses HTTP Basic_Authentication. Note that this means that users with only an OpenID login cannot use the API; they have to add a password to their account using the control panel on the site.
- The other authentication method (preffered) is OAuth
- Повече информация за начините на ауторизация можете да откриете на страница "Ауторизация"

JSONP callbacks

For API methods that return JSON, an optional JSONP-style callback parameter is supported. If supplied, the response will be in JSONP format with a callback of the given name.

```
http://bentio.com/api/statuses/public_timeline.json?callback=foo
```

Rate limiting

We currently don't enforce any rate-limiting. Please don't make us regret it.

Gotchas

Some things to remember:

(a) True API documentation.

Create and Share Digital Voice Recordings

Using mobile, landline, Skype & smartphones, tablets and computers

Evoca Enterprise

Add voice to your web application with Evoca Enterprise™

Want to add voice recording to engage your fans, supporters, customers, students, and teams with compelling audio content? The Evoca team welcomes your inquiries. Complete the online form below or email businessdevelopment@evoca.com. For customization projects, inquire at Evoca Labs.

Rapidly configure voice recording and publication features using our flexible, enterprise-class (RESTful) HTTP API (application programming interface).

Features include:

- Toll-free and worldwide local numbers
- Branded Skype call recorders
- Custom welcome voice prompts
- Call scripting
- PIN and security codes

Create content using:

- Phones: landline, mobile, and smartphones
- Skype: your custom Skype caller recorder
- Online recorders – highly customizable
- Recorders for WordPress & other web software

Publish custom, viral player widgets

- Websites
- Blogs

Solutions

- Capture Phone & Skype Interviews
- Expand Language Learning
- Promote Advocacy & Causes
- Market Professional Expertise
- Online Transcriptions

Go PRO for $6.95/month or…

Try it Free!

No Credit Card Required.

(b) False API documentation.

Fig. 1: Examples of Web pages documenting and not documenting Web APIs.

gines like Google. While public API registries provide highly valuable information, there are also some noticeable issues. First, more often than not, these registries contain out of date information (e.g. closed APIs are still listed) or even provide incorrect links to APIs documentation pages (e.g. the home page of the company is given instead). Indeed, the manual nature of the data acquisition in APIs registries aggravates these problems as new APIs appear, disappear or change. Automatically detecting the incorrect information will help registry operator better maintain their registry quality and provide better services to developers. Second, partly due to the manual submission mechanism, APIs listed in the public registries are still limited where a large number of valuable third party Web APIs may not be included. In this case, the alternative approach is to resort to Web search engine. However, general purpose search engines are not optimised for this type of activity and often mix relevant pages documenting Web APIs with general pages e.g., blogs and advertisement. Figure 1 shows both a Web pages documenting an API and one that is not that relevant but would still be presented in the results returned by search engines.

Motivated by the above observations, in our ongoing work on iServe (a public platform for service publication and discovery), we are building an automatic Web APIs search engine for detecting third party Web APIs on the Web scale. Particularly, we assume that Web pages documenting APIs are good identifiers for the detection as whenever we use an API the first referring point is likely to be the related documentation. While identifying WSDL services are relatively easy by detecting the WSDL documentation which has a standard format, detecting Web APIs documentation raises more challenges. This is due to the fact that Web APIs are generally described in plain and unstructured HTML which are only readable by human being; and to make it worse, the format of documenting a Web API is highly heterogeneous, so as its content and level of details [11]. Therefore, a prerequisite to the success of our Web APIs search engine is to construct a classifier which can offer high performance in identifying Web pages documenting Web APIs.

In this paper, we propose a novel supervised topic model called feature latent Dirichlet allocation (feaLDA) for text classification by formulating the generative process that topics are draw dependent on document class labels and words are draw conditioned on the document label-topic pairs. Particularly, feaLDA is distinguished from other related supervised topic models in its capability of accommodating different types of supervision. In particular, while supervised topic models such as labeled LDA and partial labeled LDA (pLDA) [19, 20] can only model the correspondence between class labels and documents, feaLDA is able to incorporate supervision from both document labels and labelled features for effectively improving classification performance, where the labelled features can be learned automatically from training data.

We tested feaLDA on a Web APIs dataset consisting of 622 Web pages documenting Web APIs and 925 normal Web pages crawled from ProgrammingWeb. Results from extensive experiments show that the proposed feaLDA model can achieve a very high precision of 85.2%, and it significantly outperforms three strong supervised baselines (i.e. naive Bayes, maximum entropy and SVM) as well as two closed related supervised topic models (i.e. labeled LDA and pLDA) by over 3% in accuracy. Aside from text classification, feaLDA can also extract meaningful topics with clear class label associations as illustrated by some topic examples extracted from the Web APIs dataset.

The rest of the paper is organised as follows. Section 2 reviews the previous work on Web APIs detection and the supervised topic models that are closely related to feaLDA. Section 3 presents the feaLDA model and the model inference procedure. Experimental setup and results on the Web APIs dataset are discussed in Sections 4 and 5, respectively. Finally, Section 6 concludes the paper and outlines the future work.

2 Related Work

Web Service Discovery Service discovery has been the subject of much research and development. The most renown work is perhaps Universal Description Discovery and Integration (UDDI) [3], while nowadays Seekda[2] provides the largest public index with about 29,000 WSDL Web services. The adoption of these registries has, however, been limited [3, 18]. Centred around WSDL, UDDI and related service technologies, research on semantic Web services has generated a number of ontologies, semantic discovery engines, and further supporting infrastructure aiming at improving the level of automation and accuracy that can be obtained throughout the life-cycle of service-oriented application, see [17]

[2] http://webservices.seekda.com/

for an extensive survey. Despite these advances, the majority of these initiatives are predicated upon the use of WSDL Web services, which have turned out not to be prevalent on the Web where Web APIs are increasingly favoured [15].

A fundamental characteristic of Web APIs is the fact that, despite a number of proposals [6, 7], there is no widely adopted means for publishing these services nor for describing their functionality in a way such that machines could automatically locate these APIs and understand the functionality and data they offer. Instead, Web APIs are solely accompanied by highly heterogeneous HTML pages providing documentation for developers. As a consequence, there has not been much progress on supporting the automated discovery of Web APIs. Perhaps the most popular directory of Web APIs is ProgrammableWeb which, as of June 2012, lists about 6,200 APIs and provides rather simple search mechanisms based on keywords, tags, or a simple prefixed categorisation. Based on the data provided by ProgrammableWeb, APIHut [5] increases the accuracy of keyword-based search of APIs compared to ProgrammableWeb or plain Google search. A fundamental drawback of ProgrammableWeb and by extension of APIHut is that they rely on the manual registration of APIs by users. This data tends to be out of date (e.g., discontinued APIs are still listed) and often provide pointers to generic Web pages (e.g., the home page of the company offering the API) which are not particularly useful for supporting the discovery and use of the related APIs. Finally, iServe [15] enables the application of advanced (semantic) discovery algorithms for Web API discovery but, thus far, it is limited by the fact that it relies on the presence of hRESTS annotations in Web pages which are still seldom available.

Therefore, despite the increasing relevance of Web APIs, there is hardly any system available nowadays that is able to adequately support their discovery. The first and main obstacle in this regard concerns the automated location of Web APIs, which is the main focus of this paper. In this regard, to the best of our knowledge, we are only aware of two previous initiatives. One was carried out by Steinmetz et al. [22], whose initial experiments are, according to the authors, not sufficiently performant and require further refinement. The second approach [16] is our initial work in this area which we herein expand and enhance.

Topic Models As shown in previous work [4, 12, 21, 25], topic models constructed for purpose-specific applications often involve incorporating side information or supervised information as prior knowledge for model learning, which in general can be categorised into two types depending on how the side information are incorporated [13]. One type is the so called *downstream* topic models, where both words and document metadata such as author, publication date, publication venue, etc. are generated simultaneously conditioned on the topic assignment of the document. Examples of this type include the mixed membership model [4] and the Group Topic (GT) model [25]. The *upstream* topic models, by contrast, start the generative process with the observed side information, and represent the topic distributions as a mixture of distributions conditioned on the side information elements. Examples of this type are the Author-Topic (AT) model [21] and the joint sentiment-topic (JST) model [9, 10]. Although

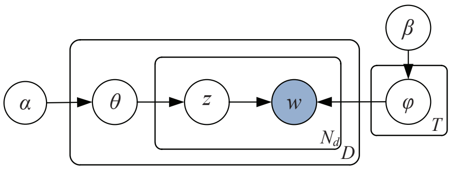
JST can detect sentiment and topic simultaneously from text by incorporating prior information to modify the Dirichlet priors of the topic-word distribution, it is still a weakly-supervised model as no mechanism is provided to incorporate document class label for model inference.

For both downstream and upstream models, most of the models are customised for a special type of side information, lacking the capability to accommodate data type beyond their original intention. This limitation has thus motivated work on developing a generalised framework for incorporating side information into topic models [2, 13]. The supervised latent Dirichlet allocation (sLDA) model [2] addresses the prediction problem of review ratings by inferring the most predictive latent topics of document labels. Mimno and McCallum [13] proposed the Dirichlet-multinomial regression (DMR) topic model which includes a log-linear prior on the document-topic distributions, where the prior is a function of the observed document features. The intrinsic difference between DMR and its complement model sLDA lies in that, while sLDA treats observed features as generated variables, DMR considers the observed features as a set of conditioned variables. Therefore, while incorporating complex features may result in increasingly intractable inference in sLDA, the inference in DMR can remain relatively simple by accounting for all the observed side information in the document-specific Dirichlet parameters.

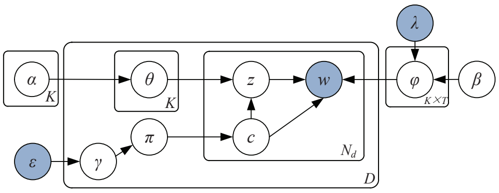
Closely related to our work are the supervised topic models incorporating document class labels. DiscLDA [8] and labeled LDA [19] apply a transformation matrix on document class labels to modify Dirichlet priors of the LDA-like models. While labeled LDA simply defines a one-to-one correspondence between LDA's latent topics and observed document labels and hence does not support latent topics within a give document label, Partially Labeled LDA (pLDA) extends labeled LDA to incorporate per-label latent topics [20]. Different from the previous work where only document labels are incorporated as prior knowledge into model learning, we propose a novel feature LDA (feaLDA) model which is capable of incorporating supervised information derive from both the document labels and the labelled features learned from data to constrain the model learning process.

3 The Feature LDA (feaLDA) Model

The feaLDA model is a supervised generative topic model for text classification by extending latent Dirichlet allocation (LDA) [1] as shown in Figure 2a. feaLDA accounts for document labels during the generative process, where each document can associate with a single class label or multiple class labels. In contrast to most of the existing supervised topic models [8,19,20], feaLDA not only accounts for the correspondence between class labels and data, but can also incorporate side information from labelled features to constrain the Dirichlet prior of topic-word distributions for effectively improving classification performance. Here the labelled features can be learned automatically from training data using any feature selection method such as information gain.



(a)



(b)

Fig. 2: (a) LDA model; (b) feaLDA model.

The graphical model of feaLDA is shown in Figure 2b. Assume that we have a corpus with a document collection $D = \{d_1, d_2, ..., d_D\}$; each document in the corpus is a sequence of N_d words denoted by $d = (w_1, w_2, ..., w_{N_d})$, and each word in the document is an item from a vocabulary index with V distinct terms. Also, letting K be the number of class labels, and T be the total number of topics, the complete procedure for generating a word w_i in feaLDA is as follows:

- For each class label $k \in \{1, ..., K\}$
  - For each topic $j \in \{1, ..., T\}$, draw $\boldsymbol{\varphi}_{kj} \sim \text{Dir}(\boldsymbol{\beta}_{kj})$
- For each document $d \in \{1, ..., D\}$,
  - draw $\boldsymbol{\pi}_d \sim \text{Dir}(\boldsymbol{\gamma} \times \boldsymbol{\epsilon}_d)$
  - For each class label k, draw $\boldsymbol{\theta}_{d,k} \sim \text{Dir}(\boldsymbol{\alpha}_k)$
- For each word w_i in document d
  - Draw a class label $c_i \sim \text{Mult}(\boldsymbol{\pi}_d)$
  - Draw a topic $z_i \sim \text{Mult}(\boldsymbol{\theta}_{d,c_i})$
  - Draw a word $w_i \sim \text{Mult}(\boldsymbol{\varphi}_{c_i,z_i})$

First, one draws a class label c from the per-document class label proportion $\boldsymbol{\pi}_d$. Following that, one draws a topic z from the per-document topic proportion $\boldsymbol{\theta}_{d,c}$ conditioned on the sampled class label c. Finally, one draws a word from the per-corpus word distribution $\boldsymbol{\varphi}_{z,c}$ conditioned on both topic z and class label c.

It is worth noting that if we assume that the class distribution $\boldsymbol{\pi}$ of the training data is observed and the number of topics is set to 1, then our feaLDA model is reduced to labeled LDA [19] where during training, words can only be assigned to the observed class labels in the document. If we allow multiple topics to be modelled under each class label, but don't incorporate the labelled feature constraints, then our feaLDA model is reduced to pLDA [20]. Both labelled LDA and pLDA actually imply a different generative process where class distribution for each document is observed, whereas our feaLDA model incorporates supervised information in a more principled way by introducing the transformation matrices $\boldsymbol{\lambda}$ and $\boldsymbol{\epsilon}$ for encoding the prior knowledge derived from both document labels and labelled features to modify the Dirichlet priors of document specific class distributions and topic-word distributions. A detailed discussion on how this can be done is presented subsequently.

3.1 Incorporating Supervised Information

Incorporating Document Class Labels: feaLDA incorporates the supervised information from document class labels by constraining that a training document can only be generated from the topic set with class labels correspond to the document's observed label set. This is achieved by introducing a dependency link from the document label matrix $\boldsymbol{\epsilon}$ to the Dirichlet prior $\boldsymbol{\gamma}$. Suppose a corpus has 2 unique labels denoted by $\boldsymbol{C} = \{c_1, c_2\}$ and for each label c_k there are 5 topics denoted by $\boldsymbol{\theta}_{c_k} = \{z_{1,c_k}, ...z_{5,c_k}\}$. Given document d's observed label vector $\boldsymbol{\epsilon}_d = \{1, 0\}$ which indicates that d is associated with class label c_1, we can encode the label information into feaLDA as

$$\boldsymbol{\gamma}_d = \boldsymbol{\epsilon}_d^T \times \boldsymbol{\gamma}. \tag{1}$$

where $\boldsymbol{\gamma} = \{\gamma_1, \gamma_2\}$ is the Dirichlet prior for the per-document class proportion $\boldsymbol{\pi}_d$ and $\boldsymbol{\gamma}_d = \{\gamma_1, 0\}$ is the modified Dirichlet prior for document d after encoding the class label information. This ensures that d can only be generated from topics associated with class label c_1 restricted by γ_1.

Incorporating Labelled Features: The second type of supervision that feaLDA accommodates is the labelled features automatically learned from the training data. This is motivated by the observation that LDA and existing supervised topic models usually set the Dirichlet prior of topic word distribution $\boldsymbol{\beta}$ to a symmetric value, which assumes each term in the corpus vocabulary is equally important before having observed any actual data. However, from a classification point of view, this is clearly not the case. For instance, words such as "endpoint", "delete" and "post" are more likely to appear in Web API documentations, whereas words like "money", "shop" and "chart" are more related to a document describing shopping. Hence, some words are more important to discriminate one class from the others. Therefore, we hypothesise that the word-class association probabilities or labelled features could be incorporated into model learning and potentially improve the model classification performance.

We encode the labelled features into feaLDA by adding an additional dependency link of $\boldsymbol{\varphi}$ (i.e., the topic-word distribution) on the word-class association probability matrix $\boldsymbol{\lambda}$ with dimension $C \times V'$, where V' denotes the labelled feature size and $V' <= V$. For word w_i, its class association probability vector is $\boldsymbol{\lambda}_{w_i} = (\lambda_{c_1,w_i}, ..., \lambda_{c_K,w_i})$, where $\sum_{c_k=1}^{K} \lambda_{c_k,w_i} = 1$. For example, the word "delete" in the API dataset with index w_t in the vocabulary has a corresponding class association probability vector $\lambda_{w_t} = (0.3, 0.7)$, indicating that "delete" has a probability of 0.3 associating with the non-API class and a probability of 0.7 with the API class. For each $w \in V$, if $w \in V'$, we can then incorporate labelled features into feaLDA by setting $\beta_{cw} = \lambda_{cw}$, otherwise the corresponding component of β is kept unchanged. In this way, feaLDA can ensure that labelled features such as "delete" have higher probability of being drawn from topics associated with the API class.

3.2 Model Inference

From the feaLDA graphical model depicted in Figure 2b, we can write the joint distribution of all observed and hidden variables which can be factored into three terms:

$$P(\mathbf{w},\mathbf{z},\mathbf{c}) = P(\mathbf{w}|\mathbf{z},\mathbf{c})P(\mathbf{z},\mathbf{c}) = P(\mathbf{w}|\mathbf{z},\mathbf{c})P(\mathbf{z}|\mathbf{c})P(\mathbf{c}) \tag{2}$$

$$= \int P(\mathbf{w}|\mathbf{z},\mathbf{c},\boldsymbol{\varPhi})P(\boldsymbol{\varPhi}|\boldsymbol{\beta})\, d\boldsymbol{\varPhi} \cdot \int P(\mathbf{z}|\mathbf{c},\boldsymbol{\varTheta})\, P(\boldsymbol{\varTheta}|\boldsymbol{\alpha})\, d\boldsymbol{\varTheta} \cdot \int P(\mathbf{c}|\boldsymbol{\varPi})\, P(\boldsymbol{\varPi}|\boldsymbol{\gamma})\, d\boldsymbol{\varPi}. \tag{3}$$

By integrating out $\boldsymbol{\varPhi}$, $\boldsymbol{\theta}$ and $\boldsymbol{\varPi}$ in the first, second and third term of Equation 3 respectively, we can obtain

$$P(\mathbf{w}|\mathbf{z},\mathbf{c}) = \left(\frac{\varGamma(\sum_{i=1}^{V}\beta_{k,j,i})}{\prod_{i=1}^{V}\varGamma(\beta_{k,j,i})}\right)^{C\times T} \prod_{k}\prod_{j}\frac{\prod_i \varGamma(N_{k,j,i}+\beta_{k,j,i})}{\varGamma(N_{k,j}+\sum_i \beta_{k,j,i})} \tag{4}$$

$$P(\mathbf{z}|\mathbf{c}) = \left(\frac{\varGamma(\sum_{j=1}^{T}\alpha_{k,j})}{\prod_{j=1}^{T}\varGamma(\alpha_{k,j})}\right)^{D\times C} \prod_{d}\prod_{k}\frac{\prod_j \varGamma(N_{d,k,j}+\alpha_{k,j})}{\varGamma(N_{d,k}+\sum_j \alpha_{k,j})} \tag{5}$$

$$P(\mathbf{c}) = \left(\frac{\varGamma(\sum_{k=1}^{C}\gamma_k)}{\prod_{k=1}^{C}\varGamma(\gamma_k)}\right)^{D} \prod_{d}\frac{\prod_k \varGamma(N_{d,k}+\gamma_k)}{\varGamma(N_d+\sum_k \gamma_k)}, \tag{6}$$

where $N_{k,j,i}$ is the number of times word i appeared in topic j with class label k, $N_{k,j}$ is the number of times words are assigned to topic j and class label k, $N_{d,k,j}$ is the number of times a word from document d is associated with topic j and class label k, $N_{d,k}$ is the number of times class label k is assigned to some word tokens in document d, N_d is the total number of words in document d and $\varGamma$ is the gamma function.

The main objective of inference in feaLDA is then to find a set of model parameters that can best explain the observed data, namely, the per-document class proportion $\boldsymbol{\pi}$, the per-document class label specific topic proportion $\boldsymbol{\theta}$, and the per-corpus word distribution $\boldsymbol{\varphi}$. To compute these target distributions, we need to know the posterior distribution $P(\boldsymbol{z},\boldsymbol{c}|\boldsymbol{w})$, i.e., the assignments of topic and class labels to the word tokens. However, exact inference in feaLDA is intractable, so we appeal to Gibbs sampler to approximate the posterior based on the full conditional distribution for a word token.

For a word token at position t, its full conditional distribution can be written as $P(z_t = j, c_t = k|\mathbf{w}, \mathbf{z}^{-t}, \mathbf{c}^{-t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma})$, where $\mathbf{z}^{-t}$ and $\mathbf{c}^{-t}$ are vectors of assignments of topics and class labels for all the words in the collection except for the word at position t in document d. By evaluating the model joint distribution in

Table 1: Web APIs dataset statistics.

Num. of Documents	Corpus size	Vocab. size	Avg. doc. length
1,547	1,096,245	35,427	708

Equation 3, we can yield the full conditional distribution as follows

$$P(z_t = j, c_t = k|\mathbf{w}, \mathbf{z}^{-t}, \mathbf{c}^{-t}, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}) \propto \frac{N^{-t}_{k,j,w_t} + \beta_{k,j,t}}{N^{-t}_{k,j} + \sum_i \beta_{k,j,i}} \cdot \frac{N^{-t}_{d,k,j} + \alpha_{k,j}}{N^{-t}_{d,k} + \sum_j \alpha_{k,j}} \cdot \frac{N^{-t}_{d,k} + \gamma_k}{N^{-t}_{d} + \sum_k \gamma_k}. \tag{7}$$

Using Equation 7, the Gibbs sampling procedure can be run until a stationary state of the Markov chain has been reached. Samples obtained from the Markov chain are then used to estimate the model parameters according to the expectation of Dirichlet distribution, yielding the approximated per-corpus topic word distribution $\varphi_{k,j,i} = \frac{N_{k,j,i}+\beta_{k,j,i}}{N_{k,j}+\sum_i \beta_{k,j,i}}$, the approximated per-document class label specific topic proportion $\theta_{d,k,j} = \frac{N_{d,k,j}+\alpha_{k,j}}{N_{d,k}+\sum_j \alpha_{k,j}}$, and finally the approximated per-document class label distribution $\pi_{d,k} = \frac{N_{d,k}+\gamma_k}{N_d+\sum_k \gamma_k}$.

3.3 Hyperparameter Settings

For the feaLDA model hyperparameters, we estimate $\boldsymbol{\alpha}$ from data using maximum-likelihood estimation and fix the values of $\boldsymbol{\beta}$ and $\boldsymbol{\gamma}$.

Setting $\boldsymbol{\alpha}$ A common practice for topic model implementation is to use symmetric Dirichlet hyperparameters. However, it was reported that using an asymmetric Dirichlet prior over the per-document topic proportions has substantial advantages over a symmetric prior [24]. We initialise the asymmetric $\boldsymbol{\alpha} = (0.1 \times L)/(K \times T)$, where L is the average document length and the value of 0.1 on average allocates 10% of probability mass for mixing. Afterwards for every 25 Gibbs sampling iterations, $\boldsymbol{\alpha}$ is learned directly from data using maximum-likelihood estimation [14, 24]

$$\Psi(\alpha_{c,z}) = \Psi(\sum_{z=1}^{T} \alpha^{old}_{c,z}) + \log \bar{\boldsymbol{\theta}}_{c,z}, \tag{8}$$

where $\log \bar{\boldsymbol{\theta}}_{c,z} = \frac{1}{D} \sum_{d=1}^{D} \log \theta_{d,c,z}$ and Ψ is the digamma function.

Setting $\boldsymbol{\beta}$ The Dirichlet prior $\boldsymbol{\beta}$ is first initialised with a symmetric value of 0.01 [23], and then modified by a transformation matrix $\boldsymbol{\lambda}$ which encodes the supervised information from the labelled feature learned from the training data.

Setting $\boldsymbol{\gamma}$ We initialise the Dirichlet prior $\gamma = (0.1 \times L)/K$, and then modify it by the document label matrix $\boldsymbol{\epsilon}$.

4 Experimental Setup

The Web APIs Dataset We evaluate the feaLDA model on the Web APIs dataset by crawling the Web pages from the API Home URLs of 1,553 Web

APIs registered in ProgrammableWeb. After discarding the URLs which are out of date, we end up with 1,547 Web pages, out of which 622 Web pages are Web API documentations and the remaining 925 Web pages are not Web API documentations.

Preprocessing The original dataset is in the HTML format. In the preprocessing, we first clean up the HTML pages using the HTML Tidy Library[3] to fix any mistakes in the Web page source. An HTML parser is subsequently used to extract contents from the HTML pages by discarding tags and the contents associating with the `<\script>` tag as these scripts are not relevant to classification. In the second step, we further remove wildcards, word tokens with non-alphanumeric characters and lower-case all word tokens in the dataset, followed by stop word removal and Porter stemming. The dataset statistics are summarised in Table 1.

Classifying a Document In the feaLDA model, the class label of a test document is determined based on $P(\mathbf{c}|d)$, i.e., the probability of a class label given a document as specified in the per-document class label proportion $\boldsymbol{\pi}_d$. So given a learned model, we classify a document d by $\hat{c}_k = \text{argmax}_{c_k} P(c_k|d)$.

5 Experimental Results

In this section, we present the classification results of feaLDA on classifying a Web page as positive class (API documentation) or negative class (not API documentation) and compare against three supervised baselines, naive Bayes (NB), maximum entropy (MaxEnt), and Support Vector Machines (SVMs). We also evaluate the impact of incorporating labelled features on the classification performance by varying the proportion of labelled features used. Finally we compare feaLDA with some of the existing supervised topic models. All the results reported here are averaged over 5 trials where for each trial the dataset was randomly split into 80-20 for training and testing. We train feaLDA with a total number of 1000 Gibbs sampling iterations.

5.1 feaLDA Classification Results without Labelled Features

As the Web APIs dataset only contains two classes, positive or negative, we set the class number $K = 2$ in feaLDA. In this section, we only incorporate supervised information from the document class labels of the training set. In order to explore how feaLDA behaves with different topic settings, we experimented with topic number $T \in \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 15, 20\}$. It is worth noting that in feaLDA there are T topics associated with each class label. So for a setting of 2 class labels and 5 topics, feaLDA essentially models a total number of 10 topic mixtures.

Figure 3 shows the classification accuracy of feaLDA and three supervised baselines, namely, NB, MaxEnt and SVM. As can be seen from the figure, all

[3] `http://tidy.sourceforge.net/`

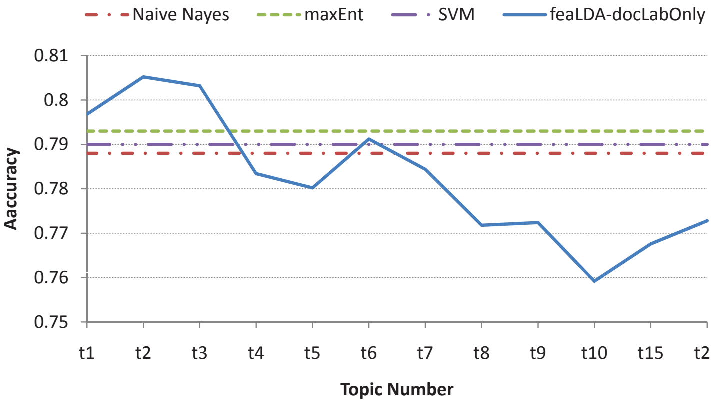


Fig. 3: feaLDA classification accuracy vs. different number of topics by incorporating supervision from class labels only.

the three supervised baselines achieve around 79% accuracy, with maxEnt giving a slightly higher accuracy of 79.3%. By incorporating the same supervision from document class labels, feaLDA outperforms all the three strong supervised baselines, giving the best accuracy of 80.5% at $T = 2$.

In terms of the impact of topic number on the model performance, it is observed that feaLDA performed the best around the topic setting $T = \{2, 3\}$. The classification accuracy drops and slightly fluctuates as the topic number increases. When the topic number is set to 1, feaLDA essentially becomes the labelled LDA model with two labelled topics being modelled corresponding to the two class labels. We see that the single topic setting actually yields worse result (i.e., 79.6% accuracy) than multiple topic settings, which shows the effectiveness of feaLDA over labelled LDA.

5.2 feaLDA Classification Results Incorporating Labelled Features

While feaLDA can achieve competitive performance by incorporating supervision from document labels alone, we additionally incorporated supervision from labelled features to evaluate whether a further gain in performance can be achieved. We extracted labelled features from the training data using information gain and discarded the features which have equal probability of both classes, resulting in a total of 29,000 features. In this experiment, we ran the feaLDA model with $T \in \{1, 2, 3, 4, 5\}$ as previous results show that large topic numbers do not yield good performance.

As observed in Figure 4, after incorporating both the document labels and labelled features, feaDLA has an substantial improvement over the model incorporating document labels only, regardless of the topic number setting. Particularly, feaLDA gives the best accuracy of 81.8% at $T = 3$, a clear 2.5% improvement over the best supervised baseline. It is also noted that when topic number is

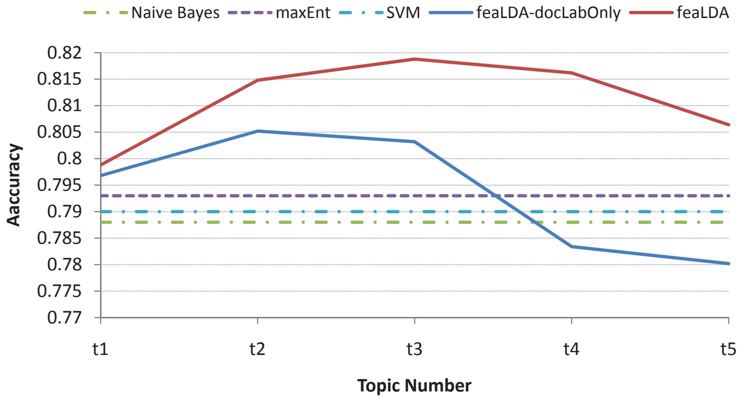


Fig. 4: feaLDA classification accuracy vs. different number of topics by incorporating supervision from both document class labels and labelled features.

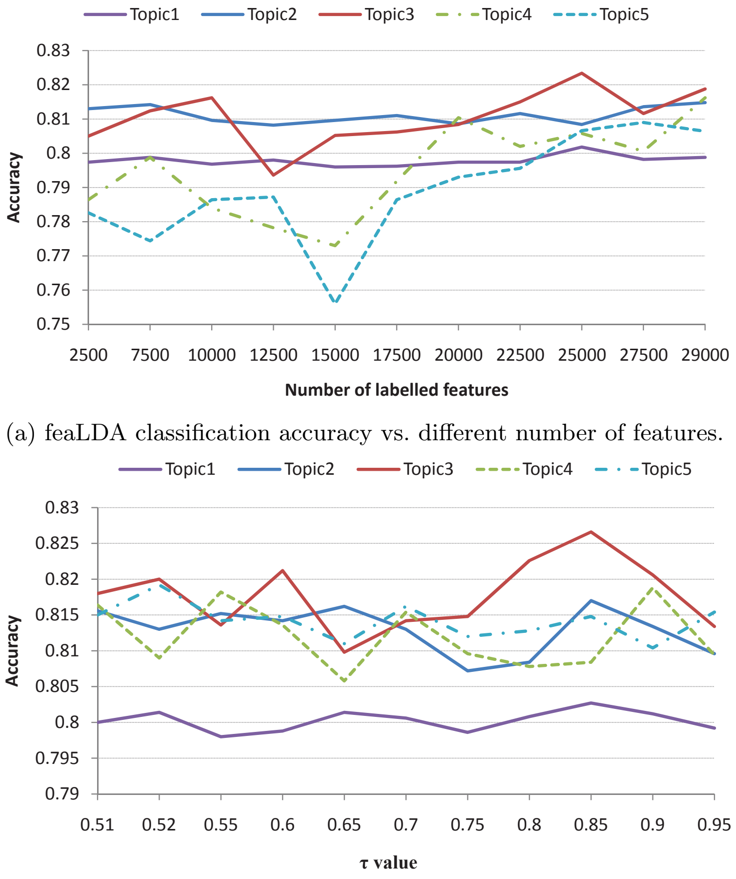
relatively large (i.e. $T = \{4, 5\}$), a significant performance drop is observed for feaLDA which only incorporates document labels; whereas feaLDA is less sensitive to topic number setting and can give fairly stable performance.

5.3 feaLDA Performance vs. Different Feature Selection Strategies

In the previous section, we directly incorporated all the labelled features into the feaLDA model. We hypothesise that using appropriate feature selection strategies to incorporate the most informative feature subset may further boost the model performance. In this section, we explore two feature selection strategies: (1) incorporate the top M features based on their information gain values; and (2) incorporate feature f if its highest class association probability is greater than a predefined threshold τ, i.e, $\text{argmax}_{c_k} P(c_k|f) > \tau$.

Figure 5a shows the classification accuracy of feaLDA by incorporating different number of most informative labelled features ranked by the information gain values. With topic setting $T = \{1, 2\}$, classification accuracy is fairly stable regardless of the number of features selected. However, with larger number of topics, incorporating more labelled features generally yields better classification accuracy. feaLDA with 3 topics achieves the best accuracy of 82.3% by incorporating the top 25,000 features, slightly outperforming the model with all features incorporated by 0.5%.

On the other hand, incorporating labelled features filtered by some predefined threshold could also result in the improvement of classification performance. As can be seen from Figure 5b, similar accuracy curves are observed for feaLDA with topic setting $T = \{1, 2, 3\}$, where they all achieved the best performance when $\tau = 0.85$. Setting higher threshold value, i.e. beyond 0.85, results in performance drop for most of the models as the number of filtered features becomes relatively small. In consistent with the previous results, feaLDA with 3 topics



(a) feaLDA classification accuracy vs. different number of features.

(b) feaLDA classification accuracy vs. different feature class probability threshold τ.

Fig. 5: feaLDA performance vs. different feature selection strategies.

still outperforms the other topic settings giving the best accuracy of 82.7%, about 1% higher than the result incorporating all the features and 3.4% higher than the best supervised baseline model MaxEnt. From the above observations, we conclude that 3 topics and a feature-class association threshold $\tau = 0.85$ are the optimal model settings for feaLDA in our experiments.

5.4 Comparing feaLDA with Existing Supervised Topic Models

In this section, we compare the overall performance of feaLDA with two supervised topic models (i.e. labelled LDA and pLDA) as well as three supervised baseline models on the APIs dataset. Apart from classification accuracy, we also

Table 2: Comparing feaLDA with existing supervised approaches. (Unit in %, numbers in bold face denote the best result in their respective row.)

	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**

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

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

report the recall, precision and F1 score for the positive class (true API label), which are summarised in Table 2.

It can be seen from Table 2 that although both feaLDA and labeled LDA give similar precision values, feaLDA outperforms labeled LDA in recall by almost 10%. Overall, feaLDA significantly outperforms labeled LDA by 6% in F1 score and 3% in accuracy. While labeled LDA simply defines a one-to-one correspondence between LDA's latent topics and document labels, pLDA extended labelled LDA by allowing multiple topics being modelled under each class label. Although pLDA (with the optimal topic setting $T = 2$) improves upon labeled LDA, it is still worse than feaLDA with its F-measure nearly 3% lower and accuracy over 2% lower compared to feaLDA. This demonstrates the effectiveness of feaLDA in incorporating labelled features learned from the training data into model learning.

When compared to the supervised baseline models, feaLDA outperforms the supervised baselines in all types of performance measure except recall. Here we would like to emphasise that one of our goals is to develop a Web APIs discovery engine on the Web scale. So considering the fact that the majority of the Web pages are not related to Web API documentation, applying a classifier such as feaLDA that can offer high precision while maintaining reasonable recall is crucial to our application.

5.5 Topic Extraction

Finally, we show some topic examples extracted by feaLDA with 2 class label and 3 topics. As listed in Table 3, the 3 topics in the top half of the table were generated from the positive API class and the remaining topics were generated from the negative class, with each topic represented by the top 15 topic words.

By inspecting the topics extracted by feaLDA, it is revealed that, most of the words appear in the topics with true API label (positive class) are fairly technical such as *json*, *statu*, *paramet*, *element*, *valu*, *request* and *string*, etc. In contrast, topics under the negative class contain many words that are not likely to appear in an API documentation, such as *contact*, *support*, *custom*, *flight*, *school*, etc. This illustrates the effectiveness of feaLDA in extracting class-associated topics from text, which can potentially be used for Web service annotation in the future extension of our search engine.

6 Conclusions

In this paper, we presented a supervised topic model called feature LDA (feaLDA) which offers a generic framework for text classification. While most of the supervised topic models [2, 19, 20] can only encode supervision from document labels for model learning, feaLDA is capable to incorporate two different types of supervision from both document label and labelled features for effectively improving classification performance. Specifically, the labelled features can be learned automatically from training data and are used to constrain the asymmetric Dirichlet prior of topic distributions. Results from extensive experiments show that, the proposed feaLDA model significantly outperforms three strong supervised baselines (i.e. NB, SVM and MaxEnt) as well as two closely related supervised topic models (i.e. labeled LDA and pLDA) for more than 3% in accuracy. More importantly, feaLDA offers very high precision performance (more than 85%), which is crucial to our Web APIs search engine to maintain a low false positive rate as majority pages on the Web are not related to APIs documentation.

In the future, we plan to develop a self-training framework where unseen data labelled with high confidence by feaLDA are added to the training pool for iteratively retraining the feaLDA model with potentially further performance improvement. Another direction we would like to pursue is to extend feaLDA for multiple class classification and evaluate it on datasets from different domains.

Acknowledgements

This work was partly funded by the EU project VPH-Share (FP7-269978).

References

1. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.
2. D.M. Blei and J.D. McAuliffe. Supervised topic models. *Arxiv preprint arXiv:1003.0783*, 2010.
3. Thomas Erl. *SOA Principles of Service Design*. The Prentice Hall Service-Oriented Computing Series. Prentice Hall, 2007.
4. E. Erosheva, S. Fienberg, and J. Lafferty. Mixed-membership models of scientific publications. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5220, 2004.

5. Karthik Gomadam, Ajith Ranabahu, Meenakshi Nagarajan, Amit P. Sheth, and Kunal Verma. A faceted classification based approach to search and rank web apis. In *Proceedings of ICWS*, pages 177–184, 2008.
6. Marc Hadley. Web Application Description Language. Member submission, W3C, 2009.
7. Jacek Kopecký, Karthik Gomadam, and Tomas Vitvar. hRESTS: an HTML Microformat for Describing RESTful Web Services. In *Proceedings of the International Conference on Web Intelligence*, 2008.
8. S. Lacoste-Julien, F. Sha, and M.I. Jordan. DiscLDA: Discriminative learning for dimensionality reduction and classification. *in NIPS*, 21, 2008.
9. C. Lin and Y. He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of Conference on Information and knowledge management (CIKM)*, 2009.
10. C. Lin, Y. He, R. Everson, and S. Rüger. Weakly-Supervised Joint Sentiment-Topic Detection from Text. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 2011.
11. M. Maleshkova, C. Pedrinaci, and J. Domingue. Investigating web apis on the world wide web. In *European Conference on Web Services*, pages 107–114, 2010.
12. A. McCallum, A. Corrada-Emmanuel, and X. Wang. Topic and role discovery in social networks. In *Proceedings of IJCAI*, pages 786–791, 2005.
13. D. Mimno and A. McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *Uncertainty in Artificial Intelligence*, pages 411–418. Citeseer, 2008.
14. T. Minka. Estimating a Dirichlet distribution. Technical report, MIT, 2003.
15. C. Pedrinaci and J. Domingue. Toward the next wave of services: linked services for the web of data. *Journal of Universal Computer Science*, 16(13):1694–1719, 2010.
16. C. Pedrinaci, D. Liu, C. Lin, and J. Domingue. Harnessing the crowds for automating the identification of web apis. In *Intelligent Web Services Meet Social Computing at AAAI Spring Symposium*, 2012.
17. Carlos Pedrinaci, John Domingue, and Amit Sheth. *Handbook on Semantic Web Technologies*, chapter Semantic Web Services. Springer, 2010.
18. Thomi Pilioura and Aphrodite Tsalgatidou. Unified Publication and Discovery of Semantic Web Services. *ACM Trans. Web*, 3(3):1–44, 2009.
19. D. Ramage, D. Hall, R. Nallapati, and C.D. Manning. Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of EMNLP*, pages 248–256, 2009.
20. D. Ramage, C.D. Manning, and S. Dumais. Partially labeled topic models for interpretable text mining. In *Proceedings of KDD*, pages 457–465, 2011.
21. M. Rosen-Zvi, T. Griffiths, M. Steyvers, and P. Smyth. The author-topic model for authors and documents. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 487–494, 2004.
22. Nathalie Steinmetz, Holger Lausen, and Manuel Brunner. Web service search on large scale. In *Proceedings of the International Joint Conference on Service-Oriented Computing*, 2009.
23. M. Steyvers and T. Griffiths. Probabilistic topic models. *Handbook of latent semantic analysis*, 427, 2007.
24. H. Wallach, D. Mimno, and A. McCallum. Rethinking lda: Why priors matter. volume 22, pages 1973–1981, 2009.
25. X. Wang, N. Mohanty, and A. McCallum. Group and topic discovery from relations and text. In *Proceedings of Intl. Workshop on Link Discovery*, pages 28–35, 2005.