Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections

Xuan-Hieu Phan
GSIS, Tohoku University
Aobayama 6-3-09
Sendai, 980-8579, Japan
hieuxuan@ecei.tohoku.ac.jp

Le-Minh Nguyen
GSIS, JAIST
1-1 Asahidai, Nomi
Ishikawa, 923-1292, Japan
nguyenml@jaist.ac.jp

Susumu Horiguchi
GSIS, Tohoku University
Aobayama 6-3-09
Sendai, 980-8579, Japan
susumu@ecei.tohoku.ac.jp

ABSTRACT

This paper presents a general framework for building classifiers that deal with short and sparse text & Web segments by making the most of hidden topics discovered from large-scale data collections. The main motivation of this work is that many classification tasks working with short segments of text & Web, such as search snippets, forum & chat messages, blog & news feeds, product reviews, and book & movie summaries, fail to achieve high accuracy due to the data sparseness. We, therefore, come up with an idea of gaining external knowledge to make the data more related as well as expand the coverage of classifiers to handle future data better. The underlying idea of the framework is that for each classification task, we collect a large-scale external data collection called "universal dataset", and then build a classifier on both a (small) set of labeled training data and a rich set of hidden topics discovered from that data collection. The framework is general enough to be applied to different data domains and genres ranging from Web search results to medical text. We did a careful evaluation on several hundred megabytes of Wikipedia (30M words) and MEDLINE (18M words) with two tasks: "Web search domain disambiguation" and "disease categorization for medical text", and achieved significant quality enhancement.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*Text analysis*; I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms, Experimentation

Keywords

Web data analysis/classification, sparse text, topic analysis

1. INTRODUCTION

Learning to classify text and Web documents has been intensively studied during the past decade. Many learning methods, such as k nearest neighbors (k-NN), Naive Bayes, maximum entropy, and support vector machines (SVMs), have been applied to a lot of classification problems with different benchmark collections (Reuters-21578, 20Newsgroups, WebKB, etc.) and achieved satisfactory results [3, 32].

WWW 2008, April 21–25, 2008, Beijing, China.
ACM 978-1-60558-085-2/08/04.

With the explosion of e-commerce and online communication & publishing, texts become available in a variety of genres like Web search snippets, forum & chat messages, blog & news feeds, book & movie summaries, product descriptions, and customer reviews. Successfully processing them, therefore, becomes increasingly important in many Web and IR applications. However, matching, classifying, and clustering these sorts of text & Web data pose new challenges. Unlike normal documents, these text & Web segments are usually noisier, less topic-focused, and much shorter, that is, they consist of from a dozen words to a few sentences. Because of the short length, they do not provide enough word co-occurrence or shared context for a good similarity measure. Therefore, normal machine learning methods usually fail to achieve desire accuracy due to the data sparseness.

There have been several studies that attempted to overcome the data sparseness to get a better (semantic) similarity. One way is to employ search engines in order to expand and enrich the context of data [10, 30, 34]. For each pair of short texts, they do statistics on the results returned by a search engine (e.g., Google) in order to decide the similarity score. A disadvantage is that repeatedly querying search engines is quite time-consuming and not suitable for real-time applications. Another way is to utilize online data repositories, such as Wikipedia or Open Directory Project[1], as external knowledge sources [4, 31]. These researches have shown positive improvement though they only used the user-defined categories and concepts in those repositories.

Inspired by the idea of using external data sources mentioned above, we present a general framework for building classifiers with hidden topics discovered from large-scale data collections that can deal successfully with short and sparse text & Web segments. The underlying idea of the framework is that for each classification task, we collect a very large external data collection called "universal dataset", and then build a classification model on both a small set of labeled training data and a rich set of hidden topics discovered from that data collection. The framework is mainly based on recent successful latent topic analysis models, such as pLSA [22] and LDA [8], and powerful machine learning methods like maximum entropy and SVMs. The main advantages of the framework include the following points:

• Reducing data sparseness: while uncommon words preserve the distinctiveness among training examples, hidden topics do make those examples more related than the original. Including hidden topics in training data helps both reduce the sparseness and make the data more topic-focused.

[1]Open Directory Project: http://www.dmoz.org

• Expanding the coverage of the classifier: topics coming from external data cover a lot of terms/words that do not exist in a (small) labeled training dataset. This is extremely useful to deal with future data, especially Web segments, that usually contain a lot of previously unseen features.

• Flexible semi-supervised learning: this framework can also be seen as a semi-supervised method because it can utilize unlabeled data to improve the classifier. However, unlike traditional semi-supervised learning algorithms [11, 29], the universal data and the training/test data are not required to have the same format. In addition, once estimated, a topic model can be applied to more than one classification problems provided that they are consistent.

• Easy to implement: the framework is simple to implement. Given a classification task, all we need to prepare is to collect a large-scale data collection to serve as the universal data and annotate a small set of training examples.

Also, the framework is general enough to be applied to different text domains and genres ranging from Web search results to medical text. We performed a careful evaluation for our method with several hundred megabytes of Wikipedia and MEDLINE data on two classification tasks: "Web search domain classification" and "disease categorization for medical text", and achieved impressive quality enhancement.

2. RELATED WORK

"Text categorization by boosting automatically extracted concepts" by Cai & Hoftmann 2003 [12] is probably the study most related to our framework. This attempts to analyze topics from data using pLSA and uses both the original data and resulting topics to train two different weak classifiers for boosting. The difference is that they extracted topics only from the training and test data while we discover hidden topics from external large-scale data collections. In addition, we aim at dealing with short and sparse text and Web segments rather than normal text documents. Another related work is the use of topic features to improve the word sense disambiguation by Cai et al. 2007 [13].

The second group of studies focused on the similarity between very short texts. Bollegala et al. 2007 [10] use search engines to get the semantic relatedness between words. Sahami & Heilman 2006 [30] also measure the relatedness between text snippets by using search engines and a similarity kernel function. Metzeler et al. 2007 [27] evaluated a wide range of similarity measures for short queries from Web search logs. Yih & Meek 2007 [34] considered this problem by improving Web-relevance similarity and the method in [30]. Gabrilovich & Markovitch 2007 [17] computing semantic relatedness for texts using Wikipedia concepts.

Prior to recent topic analysis models, word clustering algorithms were introduced to improve text categorization in different ways. Baker & McCallum 1998 [2] attempted to reduce dimensionality by class distribution-based clustering. Bekkerman et al. 2003 [6] combined distributional clustering of words and SVMs. And Dhillon & Modha 2001 [16] introduced spherical k-means for clustering sparse text data.

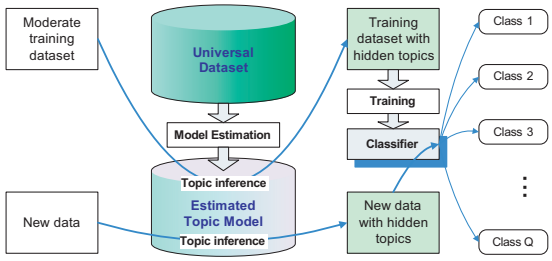
Clustering Web search have been becoming an active research topic during the past decade. Many clustering techniques were proposed to place search results into topic–oriented clusters [25, 35, 36]. This research trend has achieved great successes in which Vivisimo[2] is probably one of the most successful search clustering engines on the Web.

[2]Vivisimo: http://vivisimo.com

3. THE GENERAL FRAMEWORK

In this section, we present the proposed framework that aims at building text & Web classifiers with hidden topics from large-scale data collections. The framework is depicted in Figure 1 and consists of the following sub-problems.

Figure 1: The general framework of learning to classify short and sparse text & Web with hidden topics



(a) *Choosing an appropriate "universal dataset"*
(b) *Doing topic analysis for the universal dataset*
(c) *Building a moderate size labeled training dataset*
(d) *Doing topic inference for training and future data*
(e) *Building the classifier*

Among the five steps, choosing a right universal dataset (a) is probably the most important. First, the universal dataset, as its name implies, must be large and rich enough to cover a lot of words, concepts, and topics that are relevant to the classification problem. Second, this dataset should be consistent with the training and future unseen data that the classifier will deal with. This means that the nature of universal data (e.g., patterns, statistics, and co-occurrence of them) should be observed by humans to determine whether or not the potential topics analyzed from this data can help to make the classifier more discriminative. This will be discussed more in Section 5 where we analyze two large-scale text & Web collections for solving two classification problems. The step (b), doing topic analysis for the universal dataset, is performed by using one of the well-known hidden topic analysis models such as pLSA or LDA. We chose LDA because this model has a more complete document generation assumption. LDA will be briefly introduced in Section 4. And the analysis process of two typical universal datasets, Wikipedia & MEDLINE, is described in detail in Section 5.

In general, building a large amount of labeled training data for text classification is a labor-intensive and time-consuming task. Our framework can avoid this by requiring a moderate size or even small size of labeled data (c). One thing needs to pay more attention is that words/terms in this dataset should be relevant to as many hidden topics as possible. This is to ensure that almost hidden topics are incorporated into the classifier. Therefore, in spite of small size, the labeled training data should be balanced among topics. The experiments in Section 7 will show how well the framework can work with small size of labeled training data.

Topic inference for training and future unseen data (d) is another important issue. This depends on not only LDA but also which machine learning technique we choose to train the classifier. This will be discussed more detail in Section 6.2. Building the classifier (e) is the final procedure. After doing topic inference for training data, this step is similar to any other training process to build a text classifier. Section 6 will give a more detailed discussion about this.

4. HIDDEN TOPIC ANALYSIS MODELS

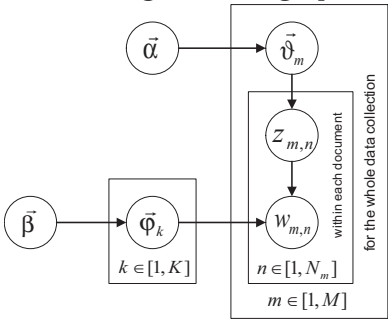
Latent Dirichlet Allocation (LDA), first introduced by Blei et al. [8], is a probabilistic generative model that can be used to estimate the multinomial observations by unsupervised learning. With respect to topic modeling, LDA is a method to perform so-called latent semantic analysis (LSA). The intuition behind LSA is to find the latent structure of "topics" or "concepts" in a text corpus. The term LSA has been coined by Deerwester et al. [14] who empirically showed that the co–occurrence (both direct and indirect) of terms in text documents can be used to recover this latent topic structure. In turn, latent-topic representations of text allowing modeling of linguistic phenomena like synonymy and polysemy. This allows IR systems to represent text in a way suitable for matching user queries on a semantic level rather than by lexical occurrence.

LDA is closely related to the probabilistic latent semantic analysis (pLSA) by Hofmann [22], a probabilistic formulation of LSA. However, it has been pointed out that LDA is more complete than pLSA in such a way that it follows a full generation process for document collection [8, 19, 21].

Models like pLSA, LDA, and their variants have been having more successful applications in document & topic modeling [8, 19], dimensionality reduction for text categorization [8], collaborative filtering [23], ad hoc IR [33], entity resolution [7], and digital library [9], and many more.

4.1 Latent Dirichlet Allocation (LDA)

Figure 2: LDA: a generative graphical model



LDA is a generative graphical model as shown in Figure 2. It can be used to model and discover underlying topic structures of any kind of discrete data in which text is a typical example. LDA was developed based on an assumption of document generation process depicted in both Figure 2 and Table 1. This process can be interpreted as follows.

In LDA, a document $\overrightarrow{w}_m = \{w_{m,n}\}_{n=1}^{N_m}$ is generated by first picking a distribution over topics $\overrightarrow{\vartheta}_m$ from a Dirichlet distribution ($Dir(\overrightarrow{\alpha})$), which determines topic assignment for words in that document. Then the topic assignment for each word placeholder $[m, n]$ is performed by sampling a particular topic $z_{m,n}$ from multinomial distribution $Mult(\overrightarrow{\vartheta}_m)$. And finally, a particular word $w_{m,n}$ is generated for the word placeholder $[m, n]$ by sampling from multinomial distribution $Mult(\overrightarrow{\varphi}_{z_{m,n}})$.

From the generative graphical model depicted in Figure 2, we can write the joint distribution of all known and hidden variables given the Dirichlet parameters as follows.

Table 1: Generation process for LDA

for all topics $k \in [1, K]$ **do**
  sample mixture components $\overrightarrow{\varphi}_k \sim Dir(\overrightarrow{\beta})$
end for
for all documents $m \in [1, M]$ **do**
  sample mixture proportion $\overrightarrow{\vartheta}_m \sim Dir(\overrightarrow{\alpha})$
  sample document length $N_m \sim Poiss(\xi)$
  for all words $n \in [1, N_m]$ **do**
    sample topic index $z_{m,n} \sim Mult(\overrightarrow{\vartheta}_m)$
    sample term for word $w_{m,n} \sim Mult(\overrightarrow{\varphi}_{z_{m,n}})$
  end for
end for

Parameters and variables:
- M: the total number of documents
- K: the number of (hidden/latent) topics
- V: vocabulary size
- $\overrightarrow{\alpha}$, $\overrightarrow{\beta}$: Dirichlet parameters
- $\overrightarrow{\vartheta}_m$: topic distribution for document m
- $\Theta = \{\overrightarrow{\vartheta}_m\}_{m=1}^{M}$: a $M \times K$ matrix
- $\overrightarrow{\varphi}_k$: word distribution for topic k
- $\Phi = \{\overrightarrow{\varphi}_k\}_{k=1}^{K}$: a $K \times V$ matrix
- N_m: the length of document m
- $z_{m,n}$: topic index of nth word in document m
- $w_{m,n}$: a particular word for word placeholder [m, n]

$$p(\overrightarrow{w}_m, \overrightarrow{z}_m, \overrightarrow{\vartheta}_m, \Phi | \overrightarrow{\alpha}, \overrightarrow{\beta}) = p(\Phi | \overrightarrow{\beta}) \prod_{n=1}^{N_m} p(w_{m,n} | \overrightarrow{\varphi}_{z_{m,n}}) p(z_{m,n} | \overrightarrow{\vartheta}_m) p(\overrightarrow{\vartheta}_m | \overrightarrow{\alpha})$$

And the likelihood of a document $\overrightarrow{w}_m$ is obtained by integrating over $\overrightarrow{\vartheta}_m$, Φ and summing over $\overrightarrow{z}_m$ as follows.

$$p(\overrightarrow{w}_m | \overrightarrow{\alpha}, \overrightarrow{\beta}) = \int\int p(\overrightarrow{\vartheta}_m | \overrightarrow{\alpha}) p(\Phi | \overrightarrow{\beta}) \cdot \prod_{n=1}^{N_m} p(w_{m,n} | \overrightarrow{\vartheta}_m, \Phi) d\Phi d\overrightarrow{\vartheta}_m$$

Finally, the likelihood of the whole data collection $\mathcal{W} = \{\overrightarrow{w}_m\}_{m=1}^{M}$ is product of the likelihoods of all documents:

$$p(\mathcal{W} | \overrightarrow{\alpha}, \overrightarrow{\beta}) = \prod_{m=1}^{M} p(\overrightarrow{w}_m | \overrightarrow{\alpha}, \overrightarrow{\beta}) \quad (1)$$

4.2 LDA Estimation with Gibbs Sampling

Estimating parameters for LDA by directly and exactly maximizing the likelihood of the whole data collection in (1) is intractable. The solution to this is to use approximate estimation methods like Variational Methods [8], Expectation–propagation [28], and Gibbs Sampling [19]. Gibbs Sampling is a special case of Markov-chain Monte Carlo (MCMC) [18] and often yields relatively simple algorithms for approximate inference in high-dimensional models such as LDA [21].

The first use of Gibbs Sampling for estimating LDA is reported in [19] and a more comprehensive description of this method is from the technical report [21]. One can refer to these papers for a better understanding of this sampling technique. Here, we only show the most important formula that is used for topic sampling for words. Let $\overrightarrow{w}$ and $\overrightarrow{z}$ be the vectors of all words and their topic assignment of the whole data collection W. The topic assignment for a particular word depends on the current topic assignment of all the other word positions. More specifically, the topic assignment of a particular word t is sampled from the following multinomial distribution.

$$p(z_i = k|\overrightarrow{z}_{\neg i}, \overrightarrow{w}) = \frac{n^{(t)}_{k,\neg i} + \beta_t}{[\sum_{v=1}^{V} n^{(v)}_k + \beta_v] - 1} \frac{n^{(k)}_{m,\neg i} + \alpha_k}{[\sum_{j=1}^{K} n^{(j)}_m + \alpha_j] - 1} \quad (2)$$

where $n^{(t)}_{k,\neg i}$ is the number of times the word t is assigned to topic k except the current assignment; $\sum_{v=1}^{V} n^{(v)}_k - 1$ is the total number of words assigned to topic k except the current assignment; $n^{(k)}_{m,\neg i}$ is the number of words in document m assigned to topic k except the current assignment; and $\sum_{j=1}^{K} n^{(j)}_m - 1$ is the total number of words in document m except the current word t. In normal cases, Dirichlet parameters $\overrightarrow{\alpha}$, and $\overrightarrow{\beta}$ are symmetric, that is, all α_k $(k = 1..K)$ are the same, and similarly for β_v $(v = 1..V)$.

After finishing Gibbs Sampling, two matrices Φ and Θ are computed as follows.

$$\varphi_{k,t} = \frac{n^{(t)}_k + \beta_t}{\sum_{v=1}^{V} n^{(v)}_k + \beta_v} \quad (3)$$

$$\vartheta_{m,k} = \frac{n^{(k)}_m + \alpha_k}{\sum_{j=1}^{K} n^{(j)}_m + \alpha_j} \quad (4)$$

5. LARGE-SCALE TEXT & WEB COLLECTIONS AS UNIVERSAL DATASETS

Choosing an appropriate universal dataset for a given classification problem is extremely important. This is because topics analyzed from this data directly influence the learning and classifying performance of the classifier. There are two main conditions that should be followed to build a right universal dataset. First, the data is large enough and should have balanced distributions over words and topics (observed by humans) in order to cover the training data, and more importantly, deal well with the diversity of future unseen data. Second, hidden topic analysis models work independently and their outputs (model parameters, topics) reflect the underlying statistics of the data. Therefore, the nature of universal data (patterns/statistics and co–occurrence) should be consistent with the classification problem.

In this section, we investigate hidden topic analysis of two large-scale data collections, Wikipedia and MEDLINE, that will be used for the evaluation in Section 7.

5.1 Hidden Topic Analysis of Wikipedia Data

Today, Wikipedia has been known as the richest online encyclopedia written collaboratively by a large number of contributors around the world. A huge number of documents available in various languages and placed in a nice structure (with consistent formats and category labels) do inspire the WWW, IR, and NLP research communities to think of using it as a huge corpus [15]. Actually, some previous researches have utilized it for named entity recognition, parallel corpus, and text categorization [4, 17, 31].

5.1.1 Data Preparation

Since Wikipedia covers a lot of concepts and domains, it is reasonable to use it as a universal dataset in our framework for classifying and clustering short and sparse text/Web. To collect the data, we prepared various *seed* crawling keywords coming from different domains as shown in the following table. For each seed keyword, we ran JWikiDocs[3] to download the corresponding Wikipedia page and crawl relevant pages by following outgoing hyperlinks. Each crawling transaction is limited by the total number of download pages or the maximum depth of hyperlink (usually 4).

Topic-oriented keywords for crawling Wikipedia
Arts architecture, fine art, dancing, fashion, film, music ...
Business advertising, e-commerce, finance, investment ...
Computers hardware, software, database, multimedia ...
Education course, graduate, professor, university ...
Engineering automobile, telecommunication, civil eng. ...
Entertainment book, music, movie, painting, photos ...
Health diet, therapy, healthcare, treatment, nutrition ...
Mass-media news, newspaper, journal, television ...
Politics government, legislation, party, regime, military ...
Science biology, physics, chemistry, ecology, laboratory ...
Sports baseball, cricket, football, tennis, olympic games ...
Misc. association, development, environment ...
Statistics of the crawled Wikipedia data
Raw data: 3.5GB; |docs| = 471,177
Preprocessing: removing duplicate docs, HTML tags, navigation links, stop and rare words
Final data: 240MB; |docs| = 71,986; |paragraphs| = 882,376; |vocabulary| = 60,649; |total words| = 30,492,305

After crawling, we got a total of 3.5GB with more than 470,000 Wikipedia documents. Because the outputs of different crawling transactions share a lot of common pages, we removed these duplicates and obtained more than 70,000 documents. And after removing HTML tags, noisy text and links, rare (threshold = 30) and stop words, we obtained the final data as reported in the table above.

5.1.2 Analysis and Outputs

We estimated many LDA models for the Wikipedia data using GibbsLDA++[4], our C/C++ implementation of LDA using Gibbs Sampling. The number of topics ranges from 10, 20 ... to 100, 150, and 200. The hyperparameters alpha and beta were set to 0.5 and 0.1, respectively. Some sample topics from the model of 200 topics are shown in Figure 3. We observed that the analysis outputs (topic-document and topic-word distributions) are impressive and satisfy our expectation. These LDA models will be used for topic inference to build Web search domain classifiers in Section 7.

5.2 Hidden Topic Analysis of MEDLINE Data

Another data collection that we performed topic analysis is Ohsumed/MEDLINE. Unlike Wikipedia, Ohsumed only includes medical abstracts. We analyzed this dataset to aim at building classifiers in the medicine domain.

5.2.1 Data Preparation

Ohsumed[5] is a test collection that was created to assist IR research. It is a clinically-oriented MEDLINE subset, that consists of 348,566 references (out of a total of over 7 million), covering all references from 270 medical journals over a five-year period (1987-1991). The collection is about 380MB including both content and meta data like field tags. After eliminating too short abstracts, meta data, rare and stop words, we finally obtained 233,442 abstracts (156MB).

[3]JWikiDocs: http://jwebpro.sourceforge.net

[4]GibbsLDA++: http://gibbslda.sourceforge.net

[5]Ohsumed: ftp://medir.ohsu.edu/pub

Figure 3: Most likely words of some sample topics of Wikipedia data. See the complete results online at: http://gibbslda.sourceforge.net/wikipedia-topics.txt

T0: medical health medicine care practice patient training treatment patients hospital surgery clinical physicians physician hospitals doctors therapy physical nursing doctor ...
T1: memory intel processor instruction processors cpu performance instructions architecture hardware data address core cache computer processing operating program ...
T4: signal radio frequency signals digital transmission channel antenna frequencies receiver communication transmitter analog modulation transmitted mhz data channels ...
T10: theory quantum universe physics mechanics particles matter particle relativity einstein model space physical light classical field theories principle energy fundamental ...
T18: economic trade economy world government growth countries country industry foreign production sector gdp development domestic billion industrial market policy nations ...
T19: film films movie production movies director cinema studio hollywood released pictures picture studios directed motion release shot sound scene actors ...
T20: party election vote elections parties voting votes candidate candidates majority political voters seats electoral democratic elected opposition coalition government ballot ...
T22: tax income taxes pay paid rate revenue taxation government benefit plan sales benefits rates value plans money cost property federal ...
T27: philosophy philosophers world philosophical knowledge mind reality aristotle existence nature plato ideas experience philosopher view consciousness kant physical idea ...
T28: space function functions vector linear theory geometry matrix equations mathematics equation field theorem algebra mathematical spaces differential product continuous ...
T33: insurance debt risk rate credit bonds pay loss loan cash policy payment bond money paid rates loans cost payments financial ...
T34: university college degree students universities school research academic student degrees campus colleges education graduate professor master institute institutions ...
T38: law act rights laws court constitution federal united legal government supreme legislation amendment civil constitutional congress public process justice power ...
T45: network networks protocol server data internet client ip nodes node connection servers protocols address packet layer connections service routing link ...
T55: government house parliament minister prime president power executive elected office council constitution assembly appointed powers head cabinet parliamentary ...
T57: cell cells protein proteins membrane molecules amino enzymes enzyme structure binding acids process bacteria acid cellular receptor antibodies receptors atp ...
T60: radio television tv stations broadcast channel news network station cable broadcasting bbc satellite programming channels service media networks broadcasts program ...
T62: music jazz dance folk blues songs musicians style musical styles traditional american song rhythm country pop performers artists played dances ...
T64: gold currency dollar coins silver value money coin issued exchange euro inflation monetary rate pound currencies paper standard dollars mint ...
T73: internet online users site com content sites community web website user virtual information websites people software media personal forums yahoo ...
T81: art artists painting paintings artist style arts movement artistic sculpture museum painted aesthetic abstract visual painters figures architecture beauty gallery ...
T84: race sports sport racing olympic events world event competition races games team golf course olympics track international championship teams formula ...
T93: military army service officers forces force officer rank training command war armed united personnel units air soldiers ranks corps navy ...
T98: bc ancient egyptian egypt civilization period culture bronze bce age king city maya archaeological stone cities egyptians temple millennium discovered ...
T101: magic harry potter magical house witch book witchcraft wizard witches magician books people spell wizards hogwarts rowling black paranormal phoenix ...
T103: card cards stores store chain department items retail customer customers shopping credit chains service retailers cash item shop merchant target ...
T104: software windows file microsoft operating version user files os applications linux source system mac versions application users released code release ...
T107: market price stock value exchange trading markets prices sell options buy spread index stocks risk selling trade features shares contracts ...
T137: bank money banks account credit financial banking central accounts reserve balance funds federal savings services deposit loans transactions deposits commercial ...
T152: economics economic value market theory price demand production capital economy cost economists costs prices marginal utility money output labor inflation ...
T199: distribution probability test random sample variables statistical variable data error analysis function value mean tests inverse statistics values hypothesis correlation ...

5.2.2 *Analysis and Outputs*

We estimated several LDA models with different numbers of topics. Alpha and beta were the same as those for Wikipedia data. We used the analysis outputs to build disease classifiers reported in Section 7.2. Some sample topics are shown in Figure 4. The sample topics in the figure are put into groups that are most relevant to each disease for more readable. We observed that the topics produced by Ohsumed are more unpredictable than those from Wikipedia data. This is because the topics in this data collection are more specific and one of them is usually coupled with a particular medical phenomenon, symptom, or disease that are only understood by domain experts.

6. BUILDING CLASSIFIERS WITH HIDDEN TOPICS

Building a classifier after topic analysis for the universal dataset includes three main steps. First, we choose one from different learning methods, such as Naive Bayes, maximum entropy (MaxEnt), SVMs, etc. Second, we integrate hidden topics into the training, test, or future data according to the data representation of the chosen learning technique. Finally, we train the classifier on the integrated training data.

6.1 Choosing Machine Learning Method

Many traditional classification methods, such as k–NN, Decision Tree, Naive Bayes, and more recent advanced models like MaxEnt, SVMs can be used in our framework. Among them, we chose MaxEnt [5] because of two main reasons. First, MaxEnt is robust and has been applied successfully to a wide range of NLP tasks, such as POS tagging, NER, parsing, etc. It even performs better than SVMs and others in some special cases, such as classifying sparse data. Second, it is very fast in both training and inference. SVMs is also a good choice because it is powerful. However, the learning and inference speed of SVMs are still a challenge to applied to almost real–time applications.

6.2 Topic Inference and Integration into Data

Given a set of new documents $\underline{W} = \{\overrightarrow{\underline{w}}_{\underline{m}}\}_{\underline{m}=1}^{\underline{M}}$. Keep in mind that $\underline{W}$ is different from the *universal dataset* W. For example W is a collection of Wikipedia documents while $\underline{W}$ is a set of Web search snippets that we need to classify. $\underline{W}$ can be the training, test, or future data. Topic inference for documents in $\underline{W}$ also needs to perform Gibbs Sampling. However, the number of sampling iterations for inference is much smaller than that for the parameter estimation. We observed that about 20 or 30 iterations are enough.

Let $\overrightarrow{w}$ and $\overrightarrow{z}$ be the vectors of all words and their topic assignment of the whole universal dataset W. And $\overrightarrow{\underline{w}}$ and $\overrightarrow{\underline{z}}$ are the vectors of all words and their topic assignment of the new dataset $\underline{W}$. The topic assignment for a particular word t in $\overrightarrow{\underline{w}}$ depends on the current topics of all the other words in $\overrightarrow{\underline{w}}$ and the topics of all words in $\overrightarrow{w}$ as follows.

$$p(\underline{z}_i = k | \overrightarrow{\underline{z}}_{\neg i}, \overrightarrow{\underline{w}}, \overrightarrow{z}, \overrightarrow{w}) = \frac{n_k^{(t)} + \underline{n}_{k,\neg i}^{(t)} + \beta_t}{[\sum_{v=1}^{V} n_k^{(v)} + \underline{n}_k^{(v)} + \beta_v] - 1} \frac{n_{\underline{m},\neg i}^{(k)} + \alpha_k}{[\sum_{j=1}^{K} n_{\underline{m}}^{(j)} + \alpha_j] - 1} \quad (5)$$

where $\underline{n}_{k,\neg i}^{(t)}$ is the number of times the current word t is assigned to topic k within $\overrightarrow{\underline{W}}$ except the current assignment; $\sum_{v=1}^{V} \underline{n}_k^{(v)} - 1$ is the number of words in $\overrightarrow{\underline{W}}$ that are assigned to topic k except the current assignment; $n_{\underline{m},\neg i}^{(k)}$ is the number of words in document $\underline{m}$ assigned to topic k except the current assignment; and $\sum_{j=1}^{K} n_{\underline{m}}^{(j)} - 1$ is the total of words in document $\underline{m}$ except the current word t.

After performing topic sampling, the topic distribution of a new document $\overrightarrow{\underline{w}}_{\underline{m}}$ is $\overrightarrow{\vartheta}_{\underline{m}} = \{\vartheta_{\underline{m},1}, \ldots, \vartheta_{\underline{m},k}, \ldots, \vartheta_{\underline{m},K}\}$ where each distribution component is computed as follows.

Figure 4: Some sample topics from of the Ohsumed-MEDLINE data. See the complete results online at: http://gibbslda.sourceforge.net/ohsumed-topics.txt

Topics most related to *Neoplasms*:
T182: cancer breast cancers women er screening pr mammography carcinoma mammary mastectomy colorectal axillary tamoxifen estrogen tumor incidence ...
T149: chemotherapy median toxicity treatment therapy survival treated combination mg /m cisplatin regimen study remission methotrexate cancer doxorubicin...
T193: carcinoma cell malignant melanoma carcinomas squamous tumour tumours gland tumor adenocarcinoma tumors salivary metastatic lesions benign glands ...
T151: biopsy specimens diagnosis biopsies aspiration needle examination cervical specimen cytology histologic cytologic positive diagnostic findings lesions ...
Topics most related to *Digestive System Diseases*:
T60: liver hepatic portal cirrhosis hepatocytes hepatocellular shunt varices cirrhotic chronic livers alcoholic bleeding sclerotherapy function hepatitis hepatocyte ...
T102: gastric ulcer duodenal gastrointestinal mucosal ulcers mucosa stomach cimetidine acid pylori bleeding endoscopic endoscopy gastritis gastrin ranitidine ...
T74: bile pancreatic biliary duct pancreatitis gallbladder pancreas endoscopic cholecystectomy bilirubin obstruction gallstones cholangitis jaundice ducts amylase ...
T141: esophageal reflux gastric esophagus emptying motility transit sphincter ph oesophageal meal dysphagia motor gastroesophageal esophagitis contractions ...
Topics most related to *Cardiovascular Diseases*:
T164: myocardial infarction coronary acute angina ischemia artery cardiac ischemic ami infarct depression unstable cad segment ecg heart mi events silent pectoris ...
T180: pressure blood hypertension hg hypertensive systolic diastolic pressures bp arterial normotensive antihypertensive shr mean spontaneously mmhg wky ...
T114: coronary artery angioplasty stenosis balloon arteries bypass angiography percutaneous occlusion ptca transluminal diameter vessel angiographic restenosis ...
T119: heart cardiac pressure output rate hemodynamic resistance arterial vascular systemic pulmonary effects mean index congestive cardiovascular hg peripheral ...
Topics most related to *Immunologic Diseases*:
T137: hiv aids immunodeficiency infection virus human acquired syndrome infected risk seropositive transmission htlv -i immune drug sexual zidovudine hiv -infected...
T140: asthma histamine airway mast bronchial fev asthmatic inhaled responsiveness airways subjects methacholine inhalation pd bronchoconstriction function ...
T20: antibodies igg antibody sera serum iga lupus immune igm systemic assay erythematosus elisa sle antigen autoantibodies immunoglobulin autoimmune ...
T129: reactions ige allergic skin reaction contact dermatitis hypersensitivity test allergy patch positive atopic tests allergens allergen rhinitis sensitization testing ...
Topics most related to *Pathological Conditions, Signs & Symptoms*:
T54: symptoms clinical diagnosis signs pain history symptomatic asymptomatic examination findings physical presentation fever recurrent laboratory symptom ...
T162: surgery cent postoperative surgical operation preoperative complications procedures operative operations intraoperative postoperatively resection procedure ...
T70: complications abdominal surgical fistula drainage management abscess complication perforation surgery fistulas treated splenic treatment repair laparotomy...
T195: syndrome abnormalities congenital abnormal clinical defect defects abnormality features disorder findings disorders severe idiopathic anomalies association ...

$$\vartheta_{\underline{m},k} = \frac{n_{\underline{m}}^{(k)} + \alpha_k}{\sum_{j=1}^{K} n_{\underline{m}}^{(j)} + \alpha_j} \quad (6)$$

After doing topic inference, we will integrate the topic distribution $\overrightarrow{\vartheta}_{\underline{m}} = \{\vartheta_{\underline{m},1}, \ldots, \vartheta_{\underline{m},k}, \ldots, \vartheta_{\underline{m},K}\}$ and the original document $\overrightarrow{w}_{\underline{m}} = \{w_{\underline{m},1}, w_{\underline{m},2}, \ldots, w_{\underline{m},N_{\underline{m}}}\}$ in order that the resulting vector is suitable for the chosen learning technique. This combination is non-trivial because the first vector is a probabilistic distribution while the second is a bag-of-word vector and their importance weights are different. This integration directly influences the learning and classification performance.

Here we describe how we integrate $\overrightarrow{\vartheta}_{\underline{m}}$ into $\overrightarrow{w}_{\underline{m}}$ to be suitable for building the classifier using MaxEnt. Because MaxEnt requires discrete feature attributes, it is necessary to discretize the probability values in $\overrightarrow{\vartheta}_{\underline{m}}$ to obtain topic names. The name of a topic appears once or several times depending on the probability of that topic. For example, a topic with probability in interval [0.05, 0.10) will appear 4 times (denote [0.05, 0.10):4). Here is an example of integrating the topic distribution into its bag-of-word vector to obtain the **snippet1** as shown in Figure 5.

- $\overrightarrow{w}_{\underline{m}}$ = {online poker tilt poker money ... card}
- $\overrightarrow{\vartheta}_{\underline{m}} = \{\ldots, \vartheta_{\underline{m},70}=0.0208, \ldots, \vartheta_{\underline{m},103}=0.1125, \ldots, \vartheta_{\underline{m},137}=0.0375, \ldots, \vartheta_{\underline{m},188}=0.0125, \ldots\}$
- Applying discretization intervals
- $\overrightarrow{w}_{\underline{m}} \cup \overrightarrow{\vartheta}_{\underline{m}}$ = **snippet1**, shown in Figure 5

The top part in Figure 5 shows an example of 9 Web search search snippets after doing topic inference and integration. Those snippets will be used with a MaxEnt classifier. For other learning techniques like SVMs, we need another integration because SVMs work with numerical vectors.

Inferred hidden topics really make the data more related. This is demonstrated by the middle and the bottom parts in Figure 5. The middle part shows the sparseness among 9 Web snippets in which only a small fraction of words are shared by two or three different snippets. Even some common words, such as "search", "online", and "compare", are not useful (noisy) because they are not related to *business* domain of the 9 snippets. The bottom part visualizes the topics shared among snippets after doing inference and integration. Most shared topics, such as "T22", "T33", "T64", "T73", "T103", "T107", "T152", and specially "T137" make the snippets more related in a semantic way. Refer to Figure 3 to see what these topics are about.

6.3 Training the Classifier

We train the MaxEnt classifier on the integrated data by using limited memory optimization (L-BFGS) [26]. As shown in recent studies, training using L-BFGS gives high performance in terms of speed and classification accuracy. All MaxEnt classifiers were trained using the same parameter setting. Those context predicates (words and topics) whose frequency in the whole training data is smaller than 3 will be eliminated, and those features (a pair of a context predicate and a class label) whose frequency is smaller than 2 will also be cut off. A Gaussian prior over feature weights with variance σ^2 of 100 was chosen for all classifiers.

7. EVALUATION

To evaluate our framework, we performed two classification tasks: "Domain disambiguation for Web search results" and "Disease classification for medical abstracts". The former attempts to classify search snippets of a Web search transaction into different domains, such as *Business*, *Computers*, *Health*, etc. And the latter classifies each medical abstract into one of five disease categories that are related to *neoplasms*, *digestive system*, etc. Search snippets are very short in comparison with normal documents. each includes a URL, a very short title, and a short text description. They are also noisy and less topic-focused. Medical abstracts are a bit longer. Each consists of several sentences and describes a particular disease. Both search snippets and medical abstracts are short, sparse, and hard-to-classify enough for verifying our framework.

Figure 5: Top: sample Google search snippets (including Wikipedia topics after inference); Middle: visualization of snippet-word co–occurrences; Bottom: visualization of shared topics among snippets after inference

(snippet1) online poker tilt poker money payment processing deposit money tilt poker account visa mastercard credit card atm check debit card *topic:70* *topic:103* *topic:103* *topic:103* *topic:103* *topic:137* *topic:137* *topic:188*

(snippet2) money payment proof broker payment online payment e-gold ebullion liberty reserve web money edinar wire transfer *topic:33* *topic:33* *topic:68* *topic:69* *topic:73* *topic:103* *topic:133* *topic:137* *topic:151*

(snippet3) savings accounts isas investments compare savings isa accounts cash isas access savings investment bonds moneysupermarket com *topic:1* *topic:22* *topic:33* *topic:45* *topic:64* *topic:73* *topic:117* *topic:137* *topic:137* *topic:138* *topic:152* *topic:153* *topic:179*

(snippet4) savings accounts online banking rate apy compare online banking rates savings account features rates apy help online savings *topic:22* *topic:32* *topic:64* *topic:73* *topic:89* *topic:107* *topic:137* *topic:137* *topic:137* *topic:199*

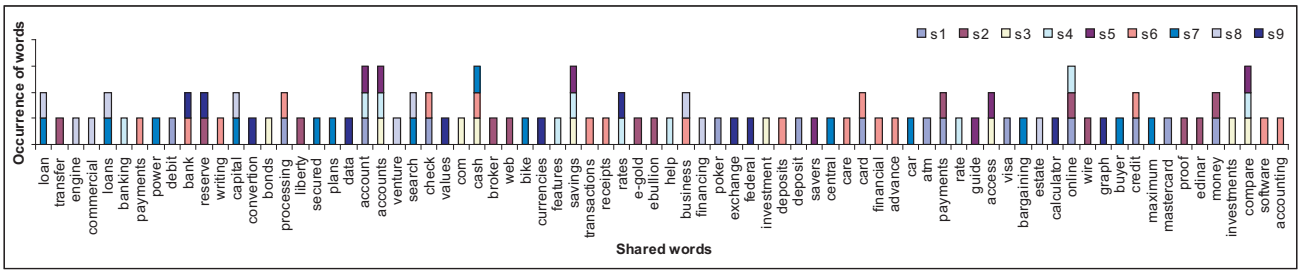
(snippet5) compare savings accounts savings accounts compare savings account savings account savings account guide compare access savers *topic:20* *topic:39* *topic:43* *topic:72* *topic:137* *topic:137* *topic:137* *topic:157* *topic:186*

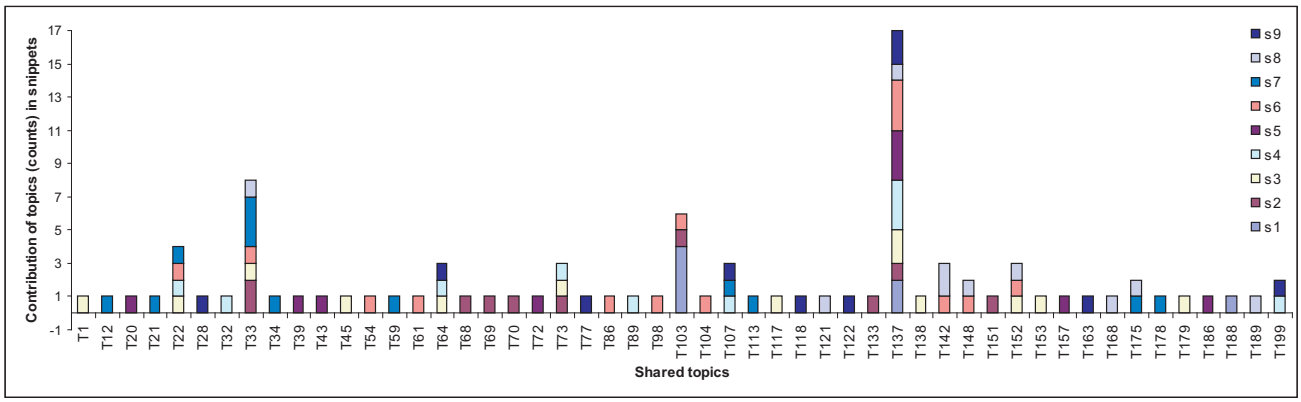
(snippet6) bank transactions sap business accounting software sap business care financial processing cash receipts check writing deposits advance payments credit card payments *topic:22* *topic:33* *topic:54* *topic:61* *topic:86* *topic:98* *topic:103* *topic:104* *topic:137* *topic:137* *topic:137* *topic:142* *topic:148* *topic:152*

(snippet7) secured loans central capital loans car loan van loan bike loan ll search secured loan plans maximum bargaining power ll cash buyer *topic:12* *topic:21* *topic:22* *topic:33* *topic:33* *topic:33* *topic:34* *topic:59* *topic:107* *topic:113* *topic:175* *topic:178*

(snippet8) search business loan capital business capital search engine business loans venture capital commercial estate financing *topic:33* *topic:121* *topic:137* *topic:142* *topic:142* *topic:148* *topic:152* *topic:168* *topic:175* *topic:189*

(snippet9) exchange rates currencies convertion calculator exchange rates graph values data federal reserve bank *topic:28* *topic:64* *topic:77* *topic:107* *topic:118* *topic:122* *topic:137* *topic:137* *topic:163* *topic:199*





7.1 Domain Disambiguation for Web Search Results with Wikipedia Topics

Clustering Web search results have been an active research topic during the past decade. Many clustering techniques were proposed to place search snippets into topic– or aspect–oriented clusters [25, 35, 36]. This research trend has achieved great successes in which Vivisimo is one of the most successful search clustering engines on the Web.

Web search domain disambiguation is different from clustering in that it attempts to put search snippets into one of predefined domains as in Table 2. In this task, hidden topics were discovered from Wikipedia data as described in Section 5.1. Both labeled training and testing data were retrieved from Google search using JWebPro[6]. The topic inference for data is as described in Section 6.2 and demonstrated in Figure 5. All the classifiers were built by using JMaxEnt[7].

[6]JWebPro: http://jwebpro.sourceforge.net
[7]JMaxEnt (in JTextPro): http://jtextpro.sourceforge.net

7.1.1 *Experimental Data*

Table 2: Google snippets as training & test data

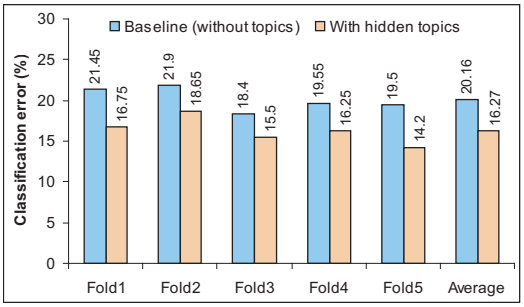
Search phrases for training & test data are exclusive				
Domain	**Training data**		**Test data**	
	#Phrs.	#Snip.	#Phrs.	#Snip.
Business	60	1,200	10	300
Computers	60	1,200	10	300
Culture-Arts-Ent.	94	1,880	11	330
Education-Science	118	2,360	10	300
Engineering	11	220	5	150
Health	44	880	10	300
Politics-Society	60	1,200	10	300
Sports	56	1,120	10	300
Total		10,060		2,280

To prepare the labeled training and test data, we performed Web search transactions using various phrases belonging to different domains. For each search transaction, we selected top 20 or 30 snippets from the results to en-

sure that most of them belong to the same domain. For example, for domain *Business*, we searched 60 phrases and selected top 20 snippets for each, and got a total of 1,200 snippets. Note that our search phrases for training and test data are *totally exclusive* to make sure that test data are really difficult to classify. The data statistics are shown in Table 2. The training and test data are available online[8].

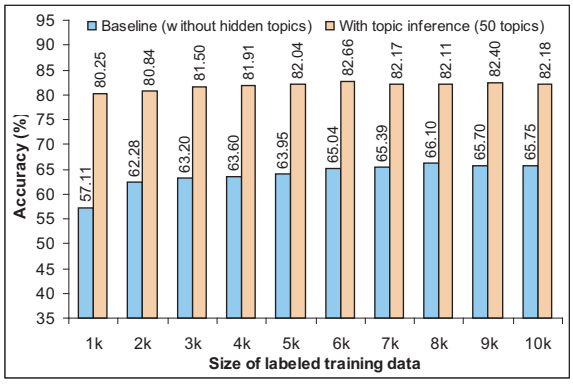
7.1.2 Results and Analysis

Figure 6: 5-fold CV evaluation on the training set



In order to examine the classification accuracy within the training data, we randomly divided the training set into five equal partitions and performed five-fold cross validation. For each fold, we ran experiments to measure the classification error of the baseline model (i.e., without hidden topics) and the model that was built according to the framework with 50 Wikipedia topics. The comparison of error is shown in Figure 6. The last two columns show the average error reduction over the five folds. As in the figure, we can reduce the error from 20.16% to 16.27% (removing 19% of error), i.e., increasing the classification accuracy from 79.84% to 83.73%. This means that even within the training data with a certain level of words shared among the snippets, our method is still able to improve the accuracy significantly.

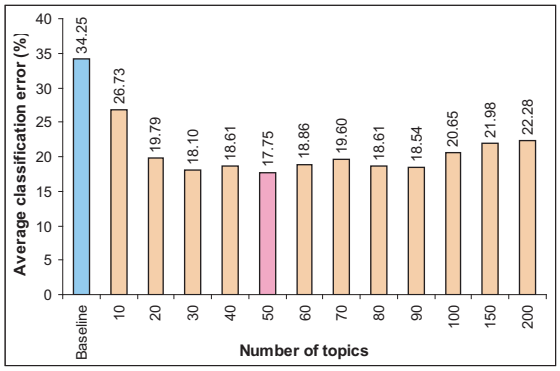
Figure 7: Test-out-of-train evaluation with different sizes of labeled training data



We did a more important experiment by training many classifiers on different sizes of the training data ranging from 1,000 to 10,000 labeled snippets, and measured the accuracy on the test set. Keep in mind that the search phrases for test data and training data are *totally exclusive* so that their snippets share very few common words. This makes the test data is really difficult to predict correctly if using traditional classifiers. The results of this experiment are shown in Figure 7. This figure highlights two points. First, the proposed method can achieve an impressive improvement of accuracy when classifying new data, that is, increasing from an accuracy of 65.75% of the baseline to 82.18% (i.e., eliminating more than 52% of error). This means that the method deals very well with sparse and previously unseen data. Second, we can achieve a high accuracy with even a small amount of labeled training data. When the size of training changes from 1,000 to 10,000 snippets, the accuracy with hidden topics changes slightly from 80.25% to 82.18% (while the baseline accuracy increases nearly 10%, from 57.11% to 65.75%).

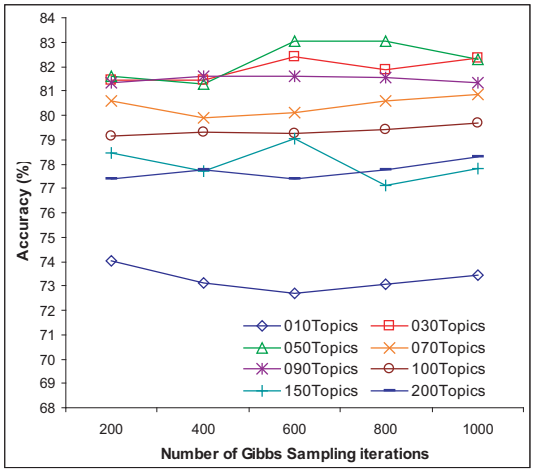
[8]http://jwebpro.sourceforge.net/data-web-snippets.tar.gz

Figure 8: Classification error reduction changing according to the number of topics



The next experiment is to see how the classification accuracy (and error) changes if we change the number of hidden topics of Wikipedia. We estimated many LDA models for the Wikipedia data with different numbers of topics (from 10 to 100, 150 and 200). After doing topic inference, 12 MaxEnt classifiers were built on the training data according to different numbers of topics. All of them, and a baseline classifier, were evaluated on the test data, and the classification error was measured. The change of classification error is depicted in Figure 8. We can see that the error reduces gradually with 10, 20 topics, reduces most around 50 topics, and then increase gradually. The error changes slightly from 20 to 100 topics. This means that the accuracy is quite stable with respect to the number of topics.

Figure 9: The accuracy changes according to the #topics and the #Gibbs Sampling iterations



The last experiment with Web search snippets is to examine how Gibbs Sampling influences the classification accuracy. We estimated different LDA models on the Wikipedia data with with different numbers of topics (K = 10, 30, ..., 100, 150, 200). To estimated parameters of each model,

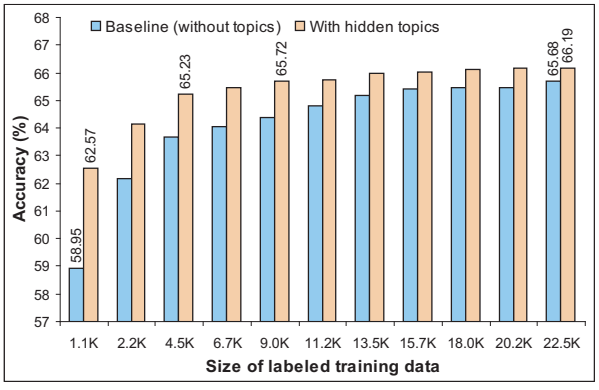
we ran 1,000 Gibbs Sampling iterations, and saved the estimated model at every 200 iterations. At these saving points, we performed topic inference for training and test data, building MaxEnt classifiers on the training data, and then measured the accuracy on the test set. The results are shown in Figure 9. As depicted in the figure, for those numbers of topics that give high performance (e.g., 30, 50 70, 90 topics), the accuracy changes slightly with respect to the different numbers of Gibbs Sampling iterations. Although it is hard to control the convergence of Gibbs Sampling, we observed that it is quite fast and yields stable results after the "burn-in" period (about 200 iterations).

7.2 Disease Classification for Medical Abstracts with MEDLINE Topics

Disease name	ID
Neoplasms	C04
Digestive System Diseases	C06
Cardiovascular Diseases	C14
Immunologic Diseases	C20
Pathological Conditions, Signs and Symptoms	C23

In order to examine how our framework work with different kinds of text domains and genres, we performed experiments with "disease classification" on medical abstracts. The original task is to assign each abstract to one of 23 MeSH disease categories[9]. For the sake of simplicity, we restricted to only 5 diseases listed above, that are similar to the experiment reported in Joachims 1998 [20]. This is a hard classification problem in comparison with "Web search domain disambiguation" because all abstracts of the same domain and abstracts of different diseases also share a lot of common medical terms. This means that the data are much less discriminative. We apply our framework to this problem to see whether or not the topics analyzed from Ohsumed/MEDLINE can make the disease abstracts more discriminative, and at which level the hidden topics can help to improve the task.

Figure 10: The average classification accuracy of the 5-fold CV tests on the Ohsumed dataset



7.2.1 Experimental Data

The evaluation data[10] include medical abstracts of the year 1991 of MEDLINE. The data consist of more than 50,000 annotated abstracts that belong to 23 MeSH disease categories. We filtered to select those abstracts that belong to the five disease categories mentioned above. After discarding blank abstracts (those only have a title) and removing stop words, we got a total of 28,145 abstracts. This subset was divided into five partitions for 5-fold CV tests.

7.2.2 Results and Analysis

For each fold we increased the training data from 1,100, 2,200 to 22,500 abstracts. We evaluated with different sizes of labeled training data in order to determine how well our framework can improve classification accuracy if we have only a few training examples. For each fold, we performed topic inference for the test data and the training data of different sizes, and then built corresponding MaxEnt classifiers. Then we took the average accuracy over the five folds for each size of training data. The average accuracies of the baseline and with hidden topics are reported in Figure 10.

As we can see, the accuracy improvement in this task is not as impressive as in Web snippet classification. This is because the texts describing diseases are much more ambiguous. From 4,500 training examples the accuracy of the baseline only increases slightly if we add more training data. However, with small size of training data, our method still improves the accuracy significantly. For example, we need only 4,500 training examples with hidden topics to attain 65.23% of accuracy, almost the same as the baseline accuracy (65.68%) that used up to 22,500 training examples.

8. DISCUSSION

We have presented a general framework to build classifiers for short and sparse text & Web data by making the most of hidden topics discovered from huge text & Web collections. We again highlight and sum up important points that are discussed elsewhere throughout the paper.

• Dealing well with short and sparse text & Web: This is the major target of the framework. Figure 5 visualizes how inferred hidden topics from the universal dataset can make the data denser and more topic–focused. Actually, the idea of using external sources of data to improve a classifier is not very new. The most significant achievement here is that all components, a huge data collection, a topic analysis model, a topic inference procedure, a small set of labeled training data, and a learning technique, are combined together in a consistent way to deal with short and sparse text & Web.

• Flexible semi-supervised learning: This framework can also be viewed as a semi-supervised learning technique. Moreover, it is flexible in that the universal data (i.e., unlabeled data) are not necessary to have the same format as the labeled training or future unseen data. Web search domain disambiguation is a typical example in which Wikipedia documents are much different from Web snippets. Also, a universal dataset, once analyzed, can be utilized to build different classification tasks provided that they are consistent. For instance, the Wikipedia data can also be used to build classifier for email or news & feed messages.

• Easy to implement: In spite of combining from different components, the framework is really easy to implement. We think of keeping the framework as simple as possible to build practical applications rather than making it more complex than necessary. All we need to prepare to develop a classifier is to collect a large data collection serving as the universal dataset and a small labeled training data.

• Computational time vs. performance: This is always an important aspect that needs to be considered when developing practical applications. Estimating the topic model for

[9]http://en.wikipedia.org/wiki/Medical_Subject_Headings
[10]http://dit.unitn.it/∼moschitt/corpora.htm

a large universal dataset is quite time-consuming. For example, estimating the Wikipedia data using GibbsLDA++ with 50 topics and 2000 Gibbs Sampling iterations took about 8 hours on a 2GHz processor. Thus it would take several days if we estimate again and again with different input parameter values to find a desire model. However, we saved time since we estimated all the models on different nodes of a cluster system in parallel. Moreover, by using our framework we only need to train classifiers on a small size of labeled training data. This is quite significant if we use SVMs or any other computation-intensive learning method.

• Data consistency: This is probably the biggest difficulty when learning with external data sources in general and applying our framework in particular. That is discovered hidden topics totally depend on the nature of the universal dataset. This is the way a topic analysis model like pLSA or LDA works. For some classification tasks, finding or collecting a universal dataset of which the underlying phenomena (e.g., patterns, their co-occurrence, or their distribution) are suitable to produce desire hidden topics are not easy. How much our framework can improve the classification accuracy depends greatly on the consistency between the universal data and the data the classifier deals with.

There are some other issues that are missing in this paper. The future work will explore more tasks for our framework, such as clustering and topic-oriented ranking in IR. Also, how to build an appropriate universal dataset for a given task is an important question that is worth considering.

Acknowledgements This work is fully supported by the research grant No.P06366 from Japan Society for the Promotion of Science (JSPS). We would also like to thank the anonymous reviewers for their valuable comments.

9. REFERENCES

[1] C. Andrieu, N. Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 50:5–43, 2003.

[2] L. Baker and A. McCallum. Distributional clustering of words for text classification. *Proc. ACM SIGIR*, 1998.

[3] P. Baldi, P. Frasconi, and P. Smyth. *Modeling the Internet & the Web: probabilistic methods & algorithms*. Wiley, 2003.

[4] S. Banerjee, K. Ramanathan, and A. Gupta. Clustering short texts using Wikipedia. *Proc. ACM SIGIR*, 2007.

[5] A. Berger, A. Pietra, and J. Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996.

[6] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *JMLR*, 3:1183–1208, 2003.

[7] I. Bhattacharya and L. Getoor. A latent Dirichlet model for unsupervised entity resolution. *Proc. SIAM SDM*, 2006.

[8] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet Allocation. *JMLR*, 3:993–1022, 2003.

[9] D. Blei and J. Lafferty. A correlated topic model of *Science*. *The Annals of Applied Statistics*, 1(1):17-35, 2007.

[10] D. Bollegala, Y. Matsuo, and M. Ishizuka. Measuring semantic similarity between words using Web search engines. *Proc. WWW*, 2007.

[11] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co–training. *Proc. COLT*, 1998.

[12] L. Cai and T. Hofmann. Text categorization by boosting automatically extracted concepts. *Proc. ACM SIGIR*, 2003.

[13] J. Cai, W. Lee, and Y. Teh. Improving WSD using topic features. *Proc. EMNLP-CoNLL*, 2007.

[14] S. Deerwester, G. Furnas, and T. Landauer. Indexing by latent semantic analysis. *Journal of the American Society for Info. Science*, 41(6):391–407, 1990.

[15] L. Denoyer and P. Gallinari. The Wikipedia XML Corpus. *ACM SIGIR Forum*, 2006.

[16] I. Dhillon and D. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 29(2–3):103–130, 2001.

[17] E. Gabrilovich and S. Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. *Proc. IJCAI*, 2007.

[18] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE PAMI*, 6:721–741, 1984.

[19] T. Griffiths and M. Steyvers. Finding scientific topics. *The National Academy of Sciences*, 101:5228–5235, 2004.

[20] T. Joachims. Text categorization with SVMs: learning with many relevant features. *Proc. ECML*, 1998.

[21] G. Heinrich. Parameter estimation for text analysis. *Technical report*, 2005.

[22] T. Hofmann. Probabilistic LSA. *Proc. UAI*, 1999.

[23] T. Hofmann. Latent semantic models for collaborative filtering. *ACM TOIS*, 22(1):89–115, 2004.

[24] F. Keller, M. Lapata, and O. Ourioupina. Using the Web to overcome data sparseness. *Proc. EMNLP*, 2002.

[25] K. Kummamuru, R. Lotlikar, S. Roy, K. Singal, and R. Krishnapuram. A hierarchical monothetic document clustering algorithm for summarization and browsing search results. *Proc. WWW*, 2004.

[26] D. Liu and J. Nocedal. On the limited memory BFGS method for large-scale optimization. *Mathematical Programming*, 45:503–528, 1989.

[27] D. Metzler, S. Dumais, and C. Meek. Similarity measures for short segments of text. *Proc. ECIR*, 2007.

[28] T. Minka and J. Lafferty. Expectation-propagation for the generative aspect model. *Proc. UAI*, 2002.

[29] K. Nigram, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2–3):103–134, 2000.

[30] M. Sahami and T. Heilman. A Webŋbased kernel function for measuring the similarity of short text snippets. *Proc. WWW*, 2006.

[31] P. Schonhofen. Identifying document topics using the Wikipedia category network. *Proc. the IEEE/WIC/ACM International Conference on Web Intelligence*, 2006.

[32] F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[33] X. Wei and W. Croft. LDA-based document models for ad-hoc retrieval. *Proc. ACM SIGIR*, 2006.

[34] W. Yih and C. Meek. Improving similarity measures for short segments of text. *Proc. AAAI*, 2007.

[35] O. Zamir and O. Etzioni. Grouper: a dynamic clustering interface to Web search results. *Proc. WWW*, 1999.

[36] H. Zeng, Q. He, Z. Chen, W. Ma, and J. Ma. Learning to cluster Web search results. *Proc. ACM SIGIR*, 2004.