Interpretation of User Comments for Detection of Malicious Websites

Mehrbod Sharifi

CMU-LTI-12-016

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Jaime Carbonell, Co-chair
Eugene Fink, Co-chair
Scott Fahlman
Gordon Cormack (University of Waterloo)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Abstract

Automated understanding of natural language is a challenging problem, which has remained open for decades. We have investigated its special case, focused on identifying relevant concepts in natural-language text in the context of a specific given task. We have developed a set of general-purpose language interpretation techniques and applied them to the task of detecting malicious websites by analyzing comments of website visitors. In this context, concepts are related to behavior or contents of websites, such as presence of pop-ups and false testimonials.

The developed algorithms are based on probabilistic topic models and other dimensionality reduction techniques applied to a special case of multi-label text classification, where concepts are output labels. We integrate information about the target task with other relevant information, including relations among concepts and external knowledge sources using a concept graph. The system iterates between training a topic model on the partially labeled data and optimizing the parameters and the label assignments. We analyze several alternative versions of this mechanism, such as one that measures the quality of separation among topics and eliminates words that are not discriminative.

For the task of detecting malicious websites, we have developed an approach that applies machine-learning techniques to the automatically collected data about websites and achieves 98% precision and 95% recall. We present a crowdsourcing system for collecting multiple-choice and free-text comments from website visitors, which is especially useful when other sources of information are insufficient or unreliable. We improve detection performance by considering the text features in the comments about the website. This performance gain is greater when using unstructured free-text comments than using multiple-choice comments. Finally, we have evaluated the performance of our language interpretation framework, and shown that the performance gain from the extracted concepts is related to the popularity of the website and task-based concepts are complementary to text features for obscure websites.

Acknowledgement

First and foremost, I would like to thank my advisors Jaime Carbonell and Eugene Fink for their support, encouragement and patience every step of the way. Eugene has been an amazing mentor to me. He taught me how to succinctly talk about ideas and properly convey them in writing, and most importantly, how to analyze them from novel perspectives which I always neglected to consider. I am grateful to Jaime for helping me develop a vision to choose among many possible research directions and supporting me to follow the ones I enjoyed. I learned from him how to focus on the most important aspects of the problems and every one of our meetings was filled with valuable feedback and insightful examples from his amazing and diverse research experience.

I am thankful to my thesis committee members: Scott Fahlman and Gordon Cormack. Scott has provided me with many valuable advices since I started at CMU. I am indebted to him for teaching me practical considerations of a language understanding systems. Gordon gave me my many valuable comments regarding web security research and his research work on web spam was my inspiration to develop the approach in this thesis on detection of malicious website.

I have greatly benefited from discussions with many people at CMU. William Cohen for his our collaborations in text mining sentiment analysis projects; Bhiksha Raj, Yiming Yang, Eric Xing and Christos Faloustos for discussions on how I should design the probabilistic models I presented in this thesis; Lorrie Cranor, Benoit Morel, Niki Kittur for discussions about my crowdsourcing solution to scam detection; Carlos Guestrin, for teaching me the techniques in the Machine Learning course in my first semester and the helpful discussions on topic modeling; Tom Mitchell for the Read-the-web project which was one of my motivations behind developing the language interpretation framework.

I am thankful to many students for helping me through the PhD years with their support: Narges Sharif Razavian, Manas Pathak, Abhay Harpale, Shilpa Arora, Amr Ahmed, Andrew Carlson, Yi Zhang, and many others.

During my two internships at Google, I collaborated with many people who helped me define and complete projects in close relationship with my thesis work. I am thankful to Natalie Glance, Kamal Nigam, Andrew Moore, Jian Huang, Mike Graham, Brady Hunsaker and many others.

I would like to thank my family who has always provided me with their unconditional support for anything I tried to achieve in my life. My parents, especially my mother, whose love for science I have inherited, were the main motivators for my shift from industry work to research. And my sister who has helped me in many ways, including verifying that my thesis made some sense to people outside of computer science community.

Finally, I am incredibly indebted to my wife Niloufar Behrouz, for her infinite love, compassion and consistent positive energy. She has been the best friend and partner one can wish for in this long journey. I would not have been able to produce this work without her support and sacrifices, from leaving her job when we moved to Pittsburgh, to taking the bigger share in taking care of our son.

Table of Content

Chapter 1

Introduction

We begin with a motivating example about deciding the trust in websites, and then provide an overview of our approaches. We explain how understanding of the natural language can help with this task, and more generally with other tasks where we have access to relevant textual information for a task.

1.1 Motivation

Imagine you are looking to buy a handmade antique or a rare stamp on the web. You perform a web search and find what you want, but it is only available on a website that you have never heard of before. At this point, you are unsure whether you can trust this website with your credit card information. A similar situation may occur when looking for specific information such as advice on treating a medical condition. Many scam websites provide false medical information and go to extremes to deceive users; for example, by posting false testimonial videos recorded by actors posed as doctors. The question is how to decide whether you can trust the information, products or services offered on a website.

We propose two approaches to address this question. First, the users can try to ask people they know. While such communications have become easier due to the prevalence of social networks, it can also be construed as spam, especially if used frequently. Moreover, there is little chance that user's friends know of a random obscure website or have the time to evaluate it, and there may also be an issue of privacy in some cases. Finally, often the delay in obtaining such responses is unacceptably long.

Another approach is to use information available on the web regarding the website and the company or individuals running it. The user can perform a manual "background check" by searching website's domain name in search engines and process the relevant information manually. We divide the information obtained in this fashion in two groups: first group is information related to the identity and reputation of the given website, such as how many visitors the website had during the last month or which country their server is located. Second group is

the comments provided by other users sharing their opinions and experiences. In the above mentioned example, other customers of the website may have shared that they have received products from this merchant without any issues.

We summarize how we have designed and evaluated these two approaches in Section 1.2. We also explain why analyzing the textual information is useful for this task and, in Section 1.3, we describe how we have attacked the problem of natural language understanding and used it to improve our performance on the target task. The developed language processing framework uses latent variable models and other dimensionality reduction methods and performs a special projection from natural language text to task-based language concepts. We believe the developed system is general and applicable to other areas beyond web security. We present the thesis statement in Section 1.4 and summarize the contributions in Section 1.5.

1.2 Detection of Malicious Websites

Most of the existing cybersecurity tools are aimed at the traditional security threats, such as viruses; however, there is only limited research on the threats caused by user naïveté. An example of such threats is Internet scam, that is, a type of security threat in which a website makes false or intentionally misleading claims, usually with the purpose of tricking its visitors into paying for fake product offers or providing sensitive information. Examples include fraudulent promises to deliver large sum of inherited money, help find work at home and cure various diseases. The detection of such scams is difficult because fraudulent sites usually use effective deception techniques that make them appear legitimate, such as fake testimonials and success stories, as well as genuine references to trusted sources, such as Wikipedia, specific scientific publications, and patents. In most cases, due to legal reasons, search engines are reluctant to block scammers unless they have specific strong proof of fraudulent activity, such as confirmed instances of malware distribution.

The defense against such threats is fundamentally different from the defense against the traditional threats. In Chapter 3, we explain what we mean by malicious websites and then provide our analysis of the different types of malicious activities in terms of how they reach victims. We then define and evaluate our machine-learning approach to address this problem which is summarized next.

1.2.1 Approach and Results

The developed system starts by collecting information about websites and then creates a scam detector classifier based on this data. We call each piece of information collected a feature. Features describe information about various aspects of a website such as its operator, identity, and behavior. Features are collected from various online sources. For example, inclusion in blacklists is a group of features. We designed and implemented a Host Analyzers web service that performs the feature collection for a given domain name. It currently extracts 43 features from 11 online sources. Some feature values may need to be transformed before they can be used in most machine learning algorithms. For example, the country where the server is located is obtained based on its IP address and it is a nominal value, that is, it is the country code of one of the countries in the world. Depending on the algorithm we use, we may need to transform nominal features, for example, to a set of Boolean features, one for each country code.

We take a supervised approach and the next step is to have a set of websites labeled as positive (scam) and negative (non-scam). The details about creation of the datasets are presented in Section 3.2.2. It involved both automated methods of using certain attributes or sources, and manual approach of asking human to label the websites. After this process, we have a dataset of 837 websites with their scam/non-scam labels and their feature values.

Finally, we apply classification algorithms to these datasets. Specifically, we used L1-regularized logistic regression and linear SVM and compared the results and analyzed the effect of the parameters. We have been able to achieve 98% precision and 95% recall for detecting scam websites.

1.2.2 Discussion

In Section 3.2.5, we discuss the strengths and weaknesses of approaches that rely on the collecting and analyzing the features about websites for detecting malicious activities. We motivate the need for using additional information when features are unavailable or unreliable. Most new websites or websites that changed ownership fall in this category.

We have designed a crowdsourcing solution to address this problem. Specifically, users provide their input in the form of structured ratings and free-text comments, and then we apply task-based language understanding to analyze and integrate their comments. We have

investigated several approaches to using unstructured text and evaluated their performance in improving the quality of the prediction.

1.3 "Understanding" Natural Language

Language is an essential component of human intelligence. Despite many years of research, there has been limited success in creating computer algorithms that can understand natural language. We follow the approach of *shallow* semantics or information extraction, where the goal is to understand parts of what is communicated. For example, consider the sentence "example.com is a safe website". We may only want to understand that this sentence refers to the entity "`example.com`"; this task is called named-entity recognition [Nadeau and Sekine, 2007]. Alternatively, in sentiment-analysis task [Pang et al., 2002], we are interested to detect that the author of this sentence had a positive opinion about this website. These shallow semantic approaches are contrasted with *deep* semantic approaches, in which the goal is to understand text completely. For example, semantic parsing converts a sentence to a first-order logic representation [Zettlemoyer and Collins, 2005]. The meaning of the example sentence can be represented as two predicates: `website("example.com")` and `safe("example.com")`. These methods often make strong assumptions, such as grammaticality of the input, and they currently have practical limitations, such as high time complexity. Additionally, deep language understanding is often impaired by the lack of background knowledge and lack of expressiveness of the knowledge representation.

To avoid confusion with the deep language understanding, we use term *language interpretation* to refer to our specific approach in shallow language understanding. We further restrict our problem by taking a goal-oriented approach, where the shallow language understanding is performed with the purpose of improving the performance on a specific *target task*, which is often framed as a classification task. We believe that the human understanding of the language is motivated by the same reward system. Humans frequently acquire additional focus information about given tasks with the goal of performing them better. One source of extra information is what other people have communicated through natural language.

The proposed task-specific language interpretation framework provides a clear way to evaluate our success based on the performance on a given target task. Since our ultimate goal is to improve task-specific performance, we focus on gathering only relevant information. That is,

we collect natural language text with an explicit instruction to the contributor that it should help with the performance of a target task, or perform filtering if significant portion of the data may be irrelevant. To increase the applicability of the framework, we assume that the input text is noisy and may contain typographical or grammatical errors. With this definition, a significant amount of the user contributed textual information on the Internet can be considered input to our framework. While we mainly focus on detection of malicious websites, we expect that our approach is applicable to other tasks that have relevant textual information. For example, user reviews for products are textual information that can be used to improve the target task of making better buying decisions.

We present our approach in two chapters: in Chapter 4, we outline our concept extraction framework with focus on types of datasets that are related to our ultimate goal. In Chapter 5, we describe experiments using the developed framework on the scam detection task.

1.3.1 Approach and Results

We present the problem of shallow language understanding as a projection from words to task-specific concepts. We have developed two groups of algorithms: the traditional dimensionality reduction approaches and probabilistic latent variable topic models. For traditional dimensionality reduction method, we compared common approaches such as PCA and *k*-mean, as well as hashing algorithms, which are projections used in approximate nearest neighbor problems. For the topic models, we present our algorithm for detection of the concepts from unstructured text. We analyze desired characteristics of the topic models and present algorithms that improve results by iteratively optimizing the model parameters. The developed approach produces more coherent and well separated topics, which are what we call concepts.

We started by analyzing the sensitivity of the prediction performance on the input parameters. For example, we observed that for the hard classes the initial random seed has a bigger impact and also more topics may be needed. Then we evaluate the prediction performance of several approaches and their dependence on the properties of the output space. We found that when the numbers of labels increase, the basic topic models do not perform as well. We also experimented with changing the number of labeled documents for a supervised topic model and noticed some performance drop when the topic constraints created by the labels are increased. Next, we evaluated the newly introduced score for determining the quality of topic separations in the topic models and showed that the score can improve the interpretability of topics when

applied iteratively. We then provided an evaluation of the topic labeling approaches and concluded selecting documents at random perform reasonable well; however, there is advantage in biasing toward the projected document entropy approaches for the easy classes versus the uncertainty-based approaches for the hard classes. Next, we perform evaluation of the newly developed Adaptive LDA with various approaches in allocating the best number of topics per class, and have shown the prediction performance improvement in the case of large number of classes. Finally, we evaluate the convergence measures and the approaches in updating the parameter of the model.

1.3.2 Discussion

We apply the language interpretation framework on the website user comments for improving the task of detecting malicious websites. There are other considerations for designing a safe-browsing system based on interpretation of user comments. For example, we believe that we can expect further performance improvements by considering more information about the author of comments such as their affiliation and expertise. We outline some of the related issues and potential future directions in Section 6.2.

1.4 Thesis Statement

In this work, we evaluated the following three research hypotheses:

1. We can develop a data mining system that automatically detects malicious websites based on publicly available data, with precision and recall of at least 95%.
2. We can develop algorithms for automated identification of concepts in natural language text that are relevant to specific given tasks.
3. We can improve the accuracy of detecting malicious websites by including the textual information from users, particularly after identifying the task-based concepts.

1.5 Contributions

We have developed a novel approach to extracting task-related concepts from natural language text and applied it to the problem of detecting malicious websites. We now provide a list of main contributions of this work:

- An approach for detection of malicious websites by applying machine learning on automatically collected website identity and reputation information. Our approach removes the biases in traditional blacklist approach.
- A natural language processing framework which maps text to a set of task-related concepts.
- Improvements of multi-label text classification approach in particular setting where the label space is large, redundant and has highly skewed distribution. In additions to improving the general approach, our algorithms can use additional information from the task, the concept relations, external knowledge either automatically acquired or by human through the labeling of instances or topics.
- Improvements to topic quality produced from probabilistic topic models. Improved topics are those that are more semantically coherent, well separated from other topics, and result in greater improvements in the task performance.
- Improvement of scam detection accuracy by integrating the textual user comments and also applying our language interpretation framework.

Figure 1 visually summarizes the various aspects of our work. The left box presents different names for the dimensions of the target space for the language interpretation. To improve the concept extraction and make it related to the task, we combine approaches from Language Technology and Crowdsourcing as shown on the right side.

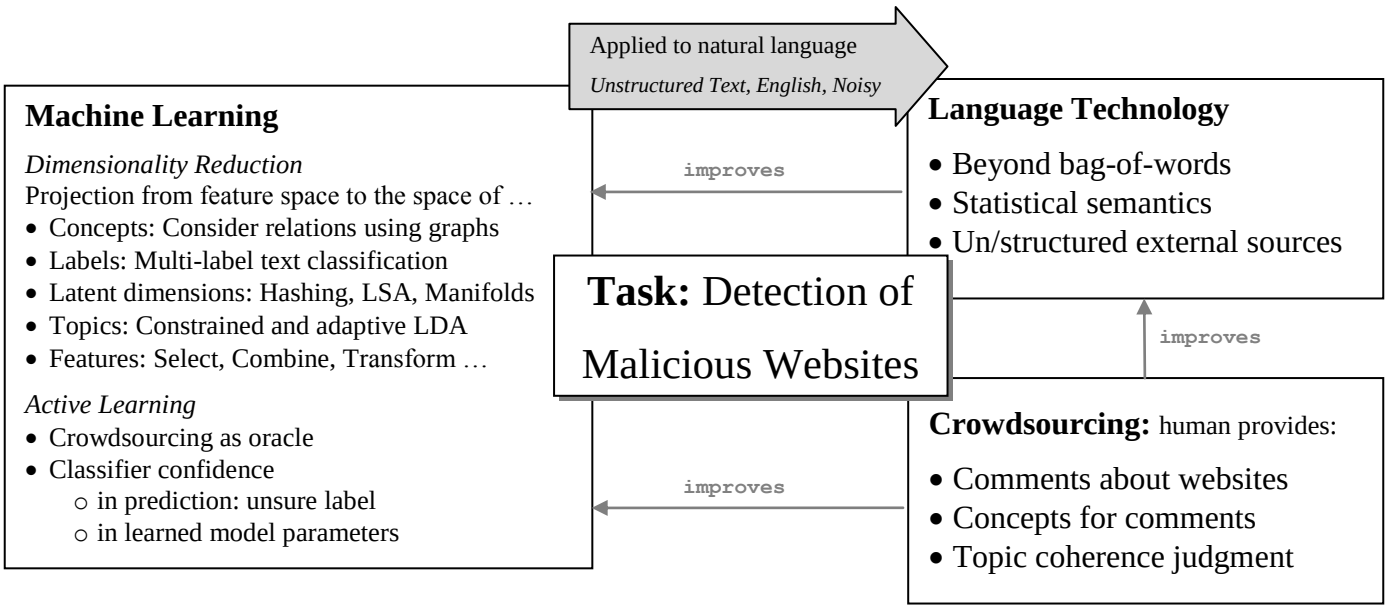


Figure 1. Summary of the methods used.

Chapter 2

Related Work

We apply machine learning to the problem of detecting malicious websites, more specifically, web scam. The novel focus is enabling the interpretation of natural language user comments with the purpose to improve the detection accuracy. There is a large body of research related to various aspects of our work, which we summarize concisely.

2.1 Multi-Label Text Classification

Text classification task is usually defined as finding a binary decision function that takes text as input, called a *document*, and determines whether it belongs to a certain class. Each class has a *label*. For example, a webpage can be labeled as "Sports". Classes can represent any concepts but they are usually categories. Some classifiers output a score, called a decision value, which represents the classifier's confidence in the predicted label. In this case, the classifier decision is obtained by thresholding the decision value. For some classifiers, such as logistic regression, the decision value is the probability of the input belonging to the class and the decision is positive when the value is greater than 0.5.

We formulated our problem of language understanding as a text classification problem where output classes are task-related concepts. A text domain can contain a large number of concepts (Chapter 4). When the number of classes is more than one, there are two possible scenarios:

1. *Multi-class* text classification: Each document belongs to exactly one class chosen from the set of available classes. In other words, the classes form a disjoint partition.
2. *Multi-label* text classification: A document may belong to none, one or multiple classes.

We focus on the multi-label case. There are two groups of approaches for extending the binary text classification to multi-label case:

1. *Ensemble Classifiers* combine the decisions from independently trained binary classifiers.
2. *Multi-label Classifiers* consider multiple labels simultaneously. This formulation has been developed for some classifiers, such as logistic regression [Carlson et al., 2010] and *k*-nearest neighbor [Zhang and Zhou, 2005].

In both cases, we obtain a binary prediction for each class and depending on the classifier, a decision score. We can often improve the performance of the classifier by tuning the decision value threshold on a validation set [Yang, 2001; Fan and Lin, 2007].

The simplest and most common ensemble classifier is one-versus-all where we train one binary classifier for each class and then apply them independently [Fan and Lin, 2007; Yang, 2001]. One way this method has been improved is using error-correcting output codes (ECOC) [Dietterich and Bakiri, 1995; Ghani, 2000; Rennie and Rifkin, 2001], where additional classifiers are trained to help the performance by separating confusing classes. Classifiers predict a subset of classes as 1 and the rest as 0, which is then combined into a binary *code* with the length equal to the number of classifiers. A new instance is classified based the Hamming distance of the code predicted by multiple classifiers and the binary code of each class. An alternative to partitioning classes as done in ECOC is to consider classifiers for all pairs of classes and combine their decisions [Hastie and Tibshirani, 1998; Liu et al., 2003]. We apply the idea of *overlapping* classifiers to our topic model approach. We also use the notion of the classifier confusion to determine the parameters of model for the classes and forming a graph of concepts.

Another approach for building ensemble classifiers is combining the classifiers by applying some weighing scheme, such as a linear combination of classifier votes in boosting [Freund et al., 1997]. A related approach is employed in random forests [Breiman, 2001], where a large number of decision trees created and combined by partitioning the features and the dataset [Ting et al., 2011; Ho, 1998]. More elaborate schemes of classifier combination such as prediction markets [Lay and Barbu, 2010], consider classifier interactions as well. We propose investigation of these directions as part of the future work.

2.1.1 Label Space Assumptions

The output for our language interpretation framework is task-related concepts (Chapter 4). The space of concepts has special properties that limit the applicability of common text classification algorithms. We review the existing work and discuss the assumptions made about the properties of the label space.

Large-scale text classification has been explored in the context where output label belongs to a large existing taxonomies existing, such as classification of webpages [Liu et al., 2005a; Liu et al., 2005b; Cai and Hofmann, 2004]. In this case, the hierarchical relationship between the labels is used to divide the training data at each node of the taxonomy tree. We consider relations

among the concepts and also organize them into a tree; however, we are also interested in other types of relation among the concepts, such as co-occurrence and *has-a* relationship in addition to *is-a* relation in the taxonomy.

The idea of using co-occurrence information among labels has been considered in previous work, such as extracting shared structure through subspacing [Ji et al., 2008], maximum margin problem [Kazawa et al., 2005], and mixture modeling [Ueda and Saito, 2002]. Label relations are especially important when finding instances of concepts in a large ontology from the web [Mitchell et al., 2009], because failing to consider certain relations can result in a major precision loss. For example, the word "BMW" can be the company name or the product name. One approach is to manually encode mutual exclusivity relations among concepts in the classifier objective function [Carlson et al., 2010]. In another related work [Yang and Gopal, 2011], the relation among classes are captured by *meta-level* features, which measure the distance of each instance to their nearest neighbor class instances and the class centroids.

Structured prediction [Taskar et al., 2005] is another classification setting where large output space is considered. The classifier maps unstructured text to a certain value assignment of discrete variables or ranks a certain number of possible value assignments. An example instance of this problem is phrase structure parsing. The possible number of outputs in this case, that is the number possible trees, is exponential in the size of the tree. We can model our problem as a structured prediction problem that maps input text to a graph where nodes are concepts and the edges are correlation between concepts; however, we did not use this formulation because the space of concepts has different properties than the structured prediction output space (Chapter 4).

Large label sets that are not defined by experts are noisy. Many labels are defined and assigned incorrectly, and for most labels there are many related labels that can be interchangeably used. We improve the performance of the model in the presence of label noise using regularized models to reduce the overfitting [Zhu and Wu, 2004; Sculley and Cormack, 2008].

2.2 Probabilistic Text Modeling

One way to characterize the classification models, such as those described in the previous section, is based on how a model is trained. Discriminative approaches model the conditional probability of label given the instance $P(y|x)$ as in logistic regression classifier, and generative

approaches model the joint probability distribution of labels and instance $P(x, y)$ as in naïve Bayes classifier. The properties of these approaches have been studied extensively [Mitchell, 2006; Andrew Y. Ng, 2002]. For text classification, the discriminative approaches such as SVM often perform better than their generative counterparts [Joachims, 2002]; however, generative models are more interpretable and can easily be modified to take into account additional information.

Initial generative text models were based on *n*-grams, which consisted of counting the occurrence of all *n*, usually consecutive, tokens in the input text and normalizing the counts to obtain a maximum likelihood estimate of the parameters for a multinomial distribution. This estimate can be poor due to lack of data and therefore various smoothing techniques were employed [Chen and Goodman, 1996]. *n*-grams models were successfully used in speech recognition [Bahl et al., 1983] and then extended to other domains [Church, 1988]. They have been particularly successful in information retrieval [Croft and Liu, 2005]. There are many approaches to prune *n*-grams to include in a language model [Banerjee and Pedersen, 2003]. We consider the *n*-grams in the context of topic models, which we will introduce later in this section.

2.2.1 Latent Variable Models

A natural extension of the *n*-gram language models is to consider multiple multinomial distributions and assume that each distribution only models the documents from a single category. Specifically, mixture of *n*-gram model [Nigam et al., 2000] introduces one *latent* variable for each document and its value is the category index of the given document.

Another approach to define latent dimensions for text modeling was inspired by the work in the human psychology [Landauer et al., 1998]. Latent semantic indexing/analysis (LSA) [Deerwester et al., 1990] involved application of singular value decomposition (SVD) of the document-term frequency matrix and effectively projecting it to a *semantic* space spanned by its eigenvectors, that is, each document is represented by new latent dimensions. LSA is a non-probabilistic method and relative to other methods introduced so far, has a higher time and space complexity; however it has been successfully used in many application areas such as word sense disambiguation [Sharifi, 2009].

2.2.2 Probabilistic Topic Models

The initial version of the topic model was a probabilistic version of LSA, called PLSA [Hofmann, 1999]. The latent dimensions were called *aspects* and the model was that documents are a mixture of aspects by assigning each word to an aspect. The model was trained using a special version of EM to avoid local optima [Hofmann, 1999].

PLSA suffered from overfitting due to using a large number of parameters. More crucially, it was unclear how the aspect proportions of new documents should be determined. These issues were addressed using heuristics in the original work. Latent Dirichlet Allocation (LDA) [Blei et al., 2003; Griffiths and Steyvers, 2004] defined a Dirichlet prior on the latent variables. Many extensions of this approach have been proposed. We categorize them based on how they related to our work; see [Sharifi, 2009] for a more detailed review:

1. **Supervised:** sLDA [Blei and McAuliffe, 2007] considers the label variable as another random variable in the graphical model that is inferred by the topic allocation of each word. DiscLDA [Lacoste-Julien et al., 2009] defines an optimization problem for the projection of the topics to labels. L-LDA [Ramage et al., 2009; Ramage et al., 2011] performs multi-label classification by constraining topics to labels. We improve on this model by making these constraints more flexible. DependencyLDA [Rubin et al., 2011] is another topic model multi-label classifier that uses another LDA over the labels as the prior over the topics. Another approach to consider relation between the topic and therefore labels is Correlated Topic Model [Blei and Lafferty, 2006], which puts a logistic normal prior over the document topic assignment instead of the Dirichlet prior. We performed experiments with these alternative approaches as well.
2. **Beyond bag of words:** Topic models that consider the order of word and *n*-gram has been extensively researched and we experimented with some of them [Wallach, 2006; Wang et al., 2007; Blei and Lafferty, 2009; Johri et al., 2010].
3. **Modeling side information:** Modeling side information is done by addition of new variables to the model. It has been extensively researched for various types of data and tasks. Examples of the side information include named entities [Newman et al., 2006], images and their tags [Blei and Jordan, 2003], authors of the documents [Rosen-Zvi et al., 2004], and sentiment and product aspects in user reviews [Titov and Mcdonald, 2008b; Lakkaraju et al.,

2011]. In Chapter 5, we describe our topic model that integrates a discriminative classifier to create the task-specific topics.

4. **Regularization:** The effect of the Dirichlet priors in LDA is similar to the regularization and it is not straightforward to formulate additional regularizations. There are, however, several formulations for regularizing the PLSA [Cai et al., 2009; Cai et al., 2008; Huh and Fienberg, 2010]. We provide more details about these models in the experiments of Chapter 4 (Section 4.2.2).
5. **Topic labeling:** We use the term *topic labels* to refer to the assignments of topics to the concepts. Topics, which are usually multinomial word distributions, are usually shown to users by listing the top *n* most probable words within the topic. Several approaches have been proposed to generate better *topic titles*, such as ranking possible titles generated from the top *n* words and Wikipedia titles using a supervised method [Lau et al., 2011], extracting of candidate titles from the corpus using techniques such as statistical significance tests on *n*-grams after applying unigram topic model [Blei and Lafferty, 2009], and optimizing the KL divergence between the topic distributions and the label distribution [Mei et al., 2007]. Any of these approaches may potentially be used with our framework. We use the top *n* word representation because it is more expressive than the alternatives.

Concept Topic Model (CTM) and Hierarchical CTM [Chemudugunta et al., 2008a] are closely related to our work because they are using the human-defined concepts. This model belongs to the category of modeling side information and works similar to Statistical Topic Model [Newman et al., 2006], which was proposed for named entity extraction. In our approach, we bias the model to push the topics toward the labels rather than modeling them in parallel because we do not have the human-defined concepts.

Another closely related work is interactive topic modeling [Andrzejewski et al., 2009], which incorporates the domain knowledge using a structured prior called *Dirichlet Forest*. This work is similar to ours in terms of using an iterative approach to set the priors using three operations called must-link, cannot-link and split; however, we use the regular Dirichlet priors and regularize them using hard constraints based on our newly defined score. Human interaction is not required for setting the priors in our work; however, we can incorporate the additional constraints defined by human. We also define another form of interaction with the user for obtaining the topic and document labels (Section 2.4).

2.3 Dimensionality Reduction

Probabilistic topic models can be considered dimensionality reduction methods because the documents are projected onto the latent dimensions and represented by the document-topic proportions. LSA [Deerwester et al., 1990] is also essentially the same technique as the most common dimensionality reduction method Principal Component Analysis (PCA) [Jolliffe and MyiLibrary, 2002]. There is a large number dimensionality reduction approaches, which can be grouped based on characteristics such as whether they are probabilistic and whether they use labels [Van Der Maaten et al., 2007]. Some dimensionality reduction techniques consider aspects of the input space that can give them advantage over the topic model. For example, locally-linear embedding [Van Der Maaten et al., 2007] models the document manifolds but LDA cannot [Huh and Fienberg, 2010].

We are particularly interested in a class of dimensionality reduction methods that are used for the approximate nearest neighbor problem, such as finding similar videos or document in a large collection. Locality Sensitive Hashing (LSH) [Indyk and Motwani, 1998] is a common approach in this domain, where a set of *hash* functions project all input instances to a binary hash code with this property: instances similar in the original space will have similar hash codes. The output code similarity is using Hamming distance, which can be implemented very efficiently through *bitcounting*. Different variations of this algorithm maintain different input space distances, such as cosine [Charikar, 2002] and L_2 [Andoni and Indyk, 2006].

There has been a considerable advance in hashing methods in recent years due to the demand of large media collections. The initial use in the language processing applications was for clustering noun phrases [Ravichandran et al., 2005]. For the purpose of our language interpretation task, we focus on a special class of hashing called *semantic hashing* [Salakhutdinov and Hinton, 2009], which are compared against LSA and LDA model. We performed experiments with several approaches including the spectral hashing [Weiss et al., 2009], and provide some directions for future work on improving these methods.

2.4 Active Learning

The goal of active learning [Settles, 2012] is to select unlabeled data to be labeled by an oracle, in such a way as to reduce the number of required labeled examples to achieve the same

prediction performance of the classifier. We use active learning techniques in two parts of our work (Chapter 4):

1. We minimize the number of labeled documents needed by comparing several active learning approaches such as uncertainty sampling [Tong and Koller, 2002].
2. We introduce the idea of active learning on topics in topic model. We provide the initial work on algorithms for selection of the topics and documents to be labeled by the oracle. We are not aware of any previous work in this area; however, the idea of human passive evaluation of the topics has been studied [Boyd-Graber et al., 2009; Newman et al., 2010; Mimno et al., 2011]. Most active learning approaches assume the oracle provides document labels. Some recent work used the oracles that can provide feature labels as well [Raghavan et al., 2006; Druck et al., 2008; Settles, 2011].

Part of our concept extraction algorithm is inspired by the *self-training* algorithm [Yarowsky, 1995], where the classifier evaluates its performance on the unlabeled data and also learns from the instances it has confidently labeled. A related procedure is used in *transductive* learning [Joachims, 1999]; however, the unlabeled data is added to the optimization problem of the SVM rather than an iterative approach.

2.5 Language Understanding

Understanding language has been a goal of the artificial intelligence research for decades. Initial attempts showed some success on simple and restricted domains. For example SHRDLU [Winograd, 1980] understood simple instructions for manipulating objects in a limited virtual world; however, these systems failed to perform reasonably on real world language understanding tasks.

Research on language understanding in recent years is focused on more focused goals. Examples include semantic parsing [Zettlemoyer and Collins, 2005], which maps natural language sentences to a first-order logic representation, textual entailment [de Marneffe et al., 2006], which answers a specific question about a given sentence, and understanding the language of instructions [Richard et al., 2001; Branavan et al., 2009; Branavan et al., 2010; Bordes et al., 2010]. Another group of language understanding systems create and populate knowledge-bases [Weld et al., 2008; Mitchell et al., 2009]. The systems use combination of symbolic and statistical techniques to extract knowledge from unstructured text and store it in predefined or

generated ontologies [Snow et al., 2006; Liang et al., 2009]. Our approach to language understanding uses shallow semantics (Section 1.3). As part of future work, our language interpretation framework can be applied to the above tasks.

2.6 Crowdsourcing

Crowdsourcing is a relatively new research area in human-computer interaction, which primarily emerged as a result of popularity of the web and the increase in contributions from the web users. Many successful systems have been implemented based on crowdsourcing, especially using Amazon Mechanical Turk (`mturk.com`) service. For example, Soylent [Bernstein et al., 2010] is a word processor add-on that enables the crowd to collaborate on difficult tasks such as proof-reading a document and rewriting shortening a document by rewrites. Another example of a widely used system, not based on the Mechanical Turk, is reCAPTCHA [Von Ahn et al., 2008], which addresses the problem of distinguishing human from software agents by challenging them to transcribe illegible words from scanned documents.

Designing crowdsourcing systems have many challenges, such as increasing the work quality of the contributors and reducing the delay in getting responses [Bernstein et al., 2011; Bernstein et al., 2012]. To the best of our knowledge, our work is the first in applying crowdsourcing approach to the problem of detecting malicious websites.

2.7 Web Security

The research on detecting web threats is mainly focused on attack types caused by security flaws in applications and protocols. Examples include drive-by downloads [Cova et al., 2010], where an attacker exploits the web browser's vulnerability to install malicious code; and cross-site request forgery [Barth et al., 2008], where an attacker uses a trusted browser session to steal or affect sensitive information. Most techniques are not based on machine learning or statistical analysis of the data; exceptions are the area of network intrusion detection [Laskov et al., 2005; Sun et al., 2008] and malware detection based on the behavioral analysis [Bayer et al., 2009] and the analysis of the malware signature [Rieck et al., 2008].

Our focus is on the web threats that exploit users' naïveté, such as in phishing or other social engineering attacks [Sheng et al., 2010]. This area of web-security is often referred to as *safe-browsing*, which is concerned with creating tools to make web browsing safer. Application of

machine learning techniques in this area is in preliminary stage [Sharifi et al., 2010]. Most of research in this area has been focused on detection of phishing attacks. Example approaches include extracting features from the phishing URLs [Garera et al., 2007; Ma et al., 2009a; Ma et al., 2009b], combining URL features with blacklists [Prakash et al., 2010], analyzing spam email and phishing website in parallel [Moore et al., 2009], analyzing website contents [Zhang et al., 2007b; Xiang and Hong, 2009], analyzing website contents with rich DHTML features that are robust to obfuscation and transformations [Hou et al., 2010], analyzing the network of how spammer are connected to each other [Ramachandran and Feamster, 2006], and analyzing the domain registration behavior pattern [McGrath and Gupta, 2008].

We discussed the task of web scam detection in Section 1.2. While the research in phishing detection has some overlap with our techniques, web scam detection has different challenges. In addition to implementing the feature extraction approaches, our technique enables integration of the human feedback to address the complexity of the web scam detection. We are not aware of any existing work that directly addresses scam detection problem. Several closely related work [Ma et al., 2009a; Choi et al., 2011] extend their work on phishing detection to cover various types of attacks simultaneously and also combine several class of features similar to our settings. The novel aspect of our work are the following:

1. Our dataset contains scam-related websites obtained by using search queries (see Section 3.2.2).
2. Our features are related to website reputation, which are designed to capture scam behavior. Moreover, our features are often more robust to manipulation than features such as link popularity [Choi et al., 2011] (see Section 3.2.1)
3. We present a crowdsourcing solution, and consider textual features extracted from user comments (see Chapter 5).

The problem of web scam is related to spam email detection, which is a well-researched area with high detection accuracy [Cormack, 2007b]. Some spam emails contain links to scam websites [Anderson et al., 2007], a fact that we use to build part of our dataset. The topics of spam emails have considerable overlap with the topics of web scams, which we will describe in Section 3; however clearly the mode of interaction, possible contents and therefore deception techniques in email spam are different than scam websites. For example, techniques for detecting

spam on social networks use many of the successful techniques such as honeypots and profile features from email spam detection techniques adapted to this new problem [Lee et al., 2010].

Another problem related to safe-browsing is the web spam [Gyongyi and Molina, 2005; Castillo et al., 2006], where attackers trick search engines and receives an undeservedly high rank for their web pages, with the purpose of attracting more visitors. There are two types of approaches to detection of web spam, called content-based [Ntoulas et al., 2006; Cormack, 2007a] and link-based [Becchetti et al., 2008] such as graph clustering methods [Castillo et al., 2007], analysis of link farms [Gyöngyi and Garcia-Molina, 2005], and spam link generators [Chung et al., 2010]. Currently, the content-based approaches tend to be more effective. The underlying techniques are similar to those used for detecting email spam [Cormack, 2007a; Cormack et al., 2010]. While web spam targets search mechanisms, web scam is aimed directly at deceiving the web users. Attackers use the search engines as one their distribution techniques and exploit the sense of trust users have developed for search engine results, which often makes web spam more successful for attackers than email spam or advertisements.

To conclude, our approach to detection of web scam is a combination of machine learning and crowdsourcing techniques applied to automatically-collected reputation-related features about websites as well as users' unstructured textual comments, as illustrated in Figure 1 (page 7). We collect the features from multiple heterogeneous online sources. When such information is unreliable, we use our task-based language understanding to improve the overall detection performance.

Chapter 3

Detection of Malicious Websites

The primary output of our work is helping the user avoid website that should not be trusted by processing information we can collect about the website. We explain the details and challenges of this task and then present our approaches. We present the approaches that use any form of textual information in Chapter 5 after we have discussed our language interpretation techniques in Chapter 4.

3.1 Internet Scam

Internet scam is a type of security threat in which a website makes false or intentionally misleading claims, usually with the purpose to trick its visitors into paying for fake product offers or providing sensitive identity information such as their social security number. Examples include fraudulent promises to help find work at home and cure various diseases. The cost of the Internet scam for the web users can be significantly higher than other forms of security threat. Internet scam can result in direct financial loss or even identity theft, which may be far worse than even a successful virus attack.

The detection of such scams is difficult because fraudulent sites usually use effective deception techniques that make them appear legitimate, such as fake testimonials and success stories, as well as genuine references to trusted sources, such as Wikipedia, specific scientific publications, and patents.

3.1.1 Common Types

We have identified the following methods of scam distribution:

- Spam emails.
- User-generated contents of generally benign multi-user posting websites, such as blog comments and user reviews.
- Online advertisements: Ad networks with banners or textual ads; classified ads on Craigslist, Ebay, Amazon, and other similar websites.

- Outside of Internet: Hard mail; TV and radio ads.

We have used these sources to create our datasets; in particular, we have manually analyzed 200 randomly selected spam emails to create this preliminary categorization of Internet scam types, which is probably not exhaustive:

1. **Medical:** Fake disease cure, longevity, weight loss, performance enhancer drugs.
2. **Phishing:** Pretending to be from a well known legitimate company, such as PayPal, and requesting a user action. The more recent variations include notifications of failed UPS and FedEx shipments, and gift cards from department stores.
3. **Advance payout:** Requests to make a payment in expectation of a large future gain, such as lottery prize or inheritance.
4. **False deals:** Fake offers of products at unusually steep discounts, such as cheap medications, insurance, and software.
5. **Other:** False promises of online degrees, work at home, dating, and other highly desirable opportunities.

3.1.2 Detection Approaches

The most common approach to fighting Internet scam is blacklisting. Many public services, such as *hosts-file.net* and *spamcop.net*, maintain a list of suspicious websites, compiled through user reports. The Web of Trust (`mywot.com`) has several million users, who rank websites on vendor reliability, trustworthiness, privacy, and child safety.

The blacklist approach however has several important limitations. First, a list may not include recently created or moved scam websites (false negatives). Second, it can mistakenly include legitimate websites because of inaccurate or intentionally biased reports (false positives). Scammers may intentionally bias blacklists in both directions. Some advanced blacklists, called *predictive* blacklists [Zhang et al., 2008b; Ma et al., 2009a] try to address this update latency, however they are focused on the problem of phishing detection (see Section 2.7 for more details).

3.2 Scam Detection using Website Reputation Data

We introduce our approach that reduces the omissions and biases in blacklists by integrating information from various heterogeneous sources, particularly focusing on the quantitative

measurements that are hard to manipulate, such as the web traffic rank and the number of search engine results pointing to a website. We have applied a logistic regression algorithm to this set of automatically collected statistics. This machine learning approach is supervised, which means that we need a dataset of known scam and legitimate websites in order to train the system. We describe how we have created such labeled datasets after we provide a preliminary taxonomy of web scam types. Then we describe the core engine of our system, which collects necessary information for the scam detection algorithm. Finally, we present an empirical evaluation of the developed technique [Sharifi et al., 2011a].

3.2.1 Host Analyzer

HostAnalyzer is a web service that we created for collecting the reputation data about websites from online sources. It is implemented as a collection of *wrappers*, which are small programs for extracting useful parts of a webpage using regular expression patterns. Each pieces of the extracted data is called *feature*. For example, `traffic_rank` of a website is an integer-valued feature obtained from the online source `alexa.com`. We currently collect 43 features from 11 online sources, which are listed with their definition in Table 1.

Most features are integers or real numbers and they are used as is. `traffic_rank` and `sites_linking_in` features are replaced by the value of $log(feature) + 1$. Boolean features are converted to {0,1}. Nominal features, such as the `country_code`, are converted Boolean features, one for each possible value.

The feature vector of a website may have many missing values, which are assigned a -1 value. For example, to extract features from Wikipedia, we need to identify the company name corresponding to a website. If Host Analyzer cannot automatically find the company name, all features obtained from Wikipedia are missing.

Feature Name	Description
Source: Google (google.com)	
search_result_count	Number of search result for the domain name.
Source: Alexa (alexa.com)	
reviews_count	Number of user comments.
Rating	Overall website rating.
traffic_rank	Worldwide traffic rank.
us_traffic_rank	US traffic rank.
sites_linking_in	Number of linking to site.
Source: IP Info (ipinfodb.com)	
latitude, longitude	Server coordinates.
country_code	Country of the server.
ip_count	Number of IP addresses for this domain name.
Source: Wikipedia (wikipedia.org)	
years_in_business, company_revenue, employees	Years since the company is established, annual revenue, number of employees.
Source: Whois (internic.net/whois.html)	
country_code	Country of the registrar.
created_days, updated_days, expires_days	Days since/to creation, update, or expiration of the domain name.
Source: Google Finance (google.com/finance)	
market_cap	Company's market capitalization.
Source: Google Safebrowsing (code.google.com/apis/safebrowsing)	
safe	Whether the website is safe.
Source: Truste (truste.com)	
member	Whether website is a member of Truste
Source: Compete (compete.com)	
unique_monthly_visitors	Unique number of user visits.
monthly_visit	Total number of user visits.
traffic_rank	Worldwide traffic rank.
Source: Web of Trust (mywot.com)	
rank	Rank based the total score.
popularity	Rank based on the traffic.
trustworthiness, trustworthiness_conf, vendor_reliability, vendor_reliability_conf, privacy, privacy_conf, child_safety, child_safety_conf	Score given by the community to each of these aspects and the number of votes.
total_positive_comments, total_negative_comments	Total number of comments for this websites.
Source: McAfee SiteAdvisor (siteadvisor.com)	
site_is_good, site_is_spam, malware, pop_ups, scam, bad_shopping_experience, browser_exploits	Number of votes for each of category.
total_comments	Total number of votes.

Table 1. The website features collected by Host Analyzer.

3.2.2 Datasets

We created five labeled datasets of scam and legitimate websites (Table 2). For each of these websites, we collect the features using Host Analyzer which is described in Section 3.2.1.

1. **Scam queries:** Certain search queries are especially likely to return scam websites. We have issued the following three queries to Google and recorded the top 500 results for each: "cancer treatments", "work at home", and "mortgage loans". After removing duplicates and non-HTML links, such as PDFs and videos, we have randomly picked 100 domain names for each query and submitted them to the Amazon Mechanical Turk for manual labeling as scam or non-scam. For each link, we have collected three human opinions. We have asked the Turk "workers" to review each website, decide whether it is scam, and indicate their level of confidence (highly confident, somewhat confident, or not confident). We marked a website

scam or non-scam if this decision was unanimous among the three highly confident annotators, which was true for %38 percent of all responses.

2. **Web of Trust:** To compile a set of popular links, we have collected the 200 most recent threads on the "Comments on websites" forum of Web of Trust (mywot.com). We have extracted all domain names mentioned in these threads, thus obtaining 159 unique names. To increase diversity of this dataset, we have added the websites ranked 1–200, 1000–1100, 10000–10100, and 15000–15100 on alexa.com. We have eliminated any websites with fewer than five comments on Web of Trust. We have then sorted the remaining 456 websites by the human-provided scam score available on Web of Trust. We have used the top 150 websites as negative and the bottom 150 websites as positive instances of scam.
3. **Spam emails:** We have obtained a dataset of 1551 spam emails from a corporate email system, detected with high confidence by McAfee AntiSpam in November 2010. We have extracted 11825 web links from those emails. Since spam emails often provide links to legitimate high-ranked websites to trick the spam filters, we have eliminated all links that are listed among the top one million websites on alexa.com. We have then identified the links that occurred at least ten times and used them as scam examples. We manually evaluated about 50 of the links (20% of the data) and they were all positive scam websites, mostly instances of phishing attack.
4. **hpHosts:** To test the effectiveness of our approach on new threats, we have taken the 100 most recent reported websites on the blacklist at hosts-file.net.
5. **Top websites:** This set is the top 100 websites according to the ranking on alexa.com. We have used them as examples of non-scam.

Dataset	Scam	Non-scam	Total
Scam Queries	33	63	96
Web of Trust	150	150	300
Spam Emails	241	none	241
hpHosts	100	none	100
Top Websites	none	100	100
All datasets	524	313	837

Table 2. Labeled datasets for scam detection.

3.2.3 Scam Detection Approach

We apply a supervised machine learning approach to scam detection. Specifically, we used a logistic regression classifier with L_1 and L_2 regularization, and a SVM classifier with linear, polynomial ($d = 2$) and Gaussian kernels on datasets labeled with scam websites. We represent each dataset of websites as D_k which is a set of feature vectors x_i.

$$D_k = \{x_i = (1, x_{i1}, x_{i2}, \dots, x_{im})\}_{i=1}^{n}$$

The features are collected using the HostAnalyzer (Table 2) and in the current version, $m = 43$ and x_{ij} is the values of feature j. Let $y_i \in \{-1 \text{ (not scam)}, 1 \text{ (scam)}\}$ be the labels for the website represented by feature vector x_i . In logistic regression, the probability of x_i being scam is:

$$P(y_i = 1 | x_i, w) = \frac{1}{1 + exp(w.x_i)}$$

$$\text{where} \quad w.x = \sum_j x_{ij} w_j$$

The training procedure learns w by maximizing the following convex log-likelihood function with λ being the regularization parameter and $\|w\|_1 = \sum_j |w_j|$ is the L1-norm of the weight vector:

$$l(w) = \sum_i log(P(y_i | x_i, w)) - \lambda \|w\|_1$$

The second classifier that we used is SVM [Joachims 2002]. Training procedure for SVM minimizes the objective function of the following form. $\boldsymbol{w}, \boldsymbol{x_i}$ and $\boldsymbol{y_i}$ are introduced above. There is one constraint per training instance and ξ_i are slack variables which enable some training instances to be classified incorrectly but the total of these variable are penalized by C which specifies our tolerance for this error.

$$min\left\{\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{n}\xi_i\right\} \quad s.t.$$

$$y_i(\boldsymbol{w}.\boldsymbol{x_i} - b) \geq 1 - \xi_i$$

Both λ and C are set by maximizing the AUC in a 5-fold cross validation on the training part of the data. We provide more detailed results for the sensitivity of λ in the experiments.

3.2.4 Experiments

The classification performance is measured with four measures. For all measure, the higher numbers are better.

1. *Precision:* ratio of correctly detected scam websites (true positives) within all those websites detected as scam (all positives). It is affected by incorrect prediction of a legitimate website as scam (false positives):

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \mathbf{false\ positives}}$$

2. *Recall:* ratio of correctly detected scam websites (true positives) within all the scam websites within the specific dataset. It is affected by incorrect prediction of a scam website as legitimate (false negatives):

$$Recall = \frac{\text{true positives}}{\text{true positives} + \mathbf{false\ negatives}}$$

3. *F1-measure:* harmonic mean of precision and recall.

$$\mathrm{F_1} = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$$

4. *AUC:* the area under the Receiver Operating Characteristics (ROC) curve, which is false positives rate as x-axis (false positives divided by negatives) and true positive rate as y-axis (true positives divided by positives). A uniform random binary classifier on a balance set produces the diagonal line with AUC of 0.5.

To measure the statistical significance of our results in this thesis, we considered two tests as shown to be accurate in previous evaluations [Smucker et al., 2007]: two-sided student's paired *t*-test and the permutation test. Permutation test, also known as randomization test, is an improvement to *t*-test and it does not make the same normality assumption, however it is computationally intensive to calculate it for large number of comparison values. Let us assume that we have obtained k pairs of results from these two system and the average difference of A

has been greater than B by x, and the null hypothesis is that the two systems A and B are identical. The permutation test randomly labels each value in pairs as coming from system A or B, then the two-sided *p*-value is the number of time that the difference based on the given label has been greater than x divided by all the 2^k possible label assignments. In practice, smaller number of permutation is considered as an approximation [Smucker et al., 2007]. We observed that the result of permutation test has always been similar or better (i.e., lower *p*-value estimate) in comparison with the more conventional two-sided paired *t*-test, and therefore we only reported the *t*-test. *p*-value bounds reported, mostly in figure captions, are using 10-fold cross validation and comparing the result of each system on the test portion of the given fold. For smaller datasets, we also tested with larger number of folds and the results were similar.

We show the 10-fold cross validation results for each subset of our dataset in Table 3. The results from of various classifier options are summarized in Table 4. We compare these results to three baselines in Figure 2: majority class baseline and individual classifiers trained on two most relevant features which are the number of search results from Google and the traffic rank from *alexa.com*. For the single feature baselines, the performance is measured by treating the feature values as the classifier output and only for the instances where the feature values were available. Some features have missing values. Google search-result-count feature is defined for 70% of our dataset and Alexa traffic-rank feature is defined for 61% of the dataset.

We also provide a comparison between web scam detection and the state-of-the-art results on two related tasks: email spam detection [Cormack, 2007a] and web spam detection [Cormack, 2010].

Dataset Name	Precision	Recall	F1	AUC
Scam Queries	0.9833	0.9667	0.9749	0.9667
Web of Trust	0.9923	0.9923	0.9923	0.9990
Spam Email	1.0000	0.9627	0.9809	-
hpHosts	1.0000	0.9200	0.9583	-
Top Websites	1.0000	0.9600	0.9795	-
All	**0.9795**	**0.9813**	**0.9803**	**0.9858**

Table 3. Summary of scam detection results.

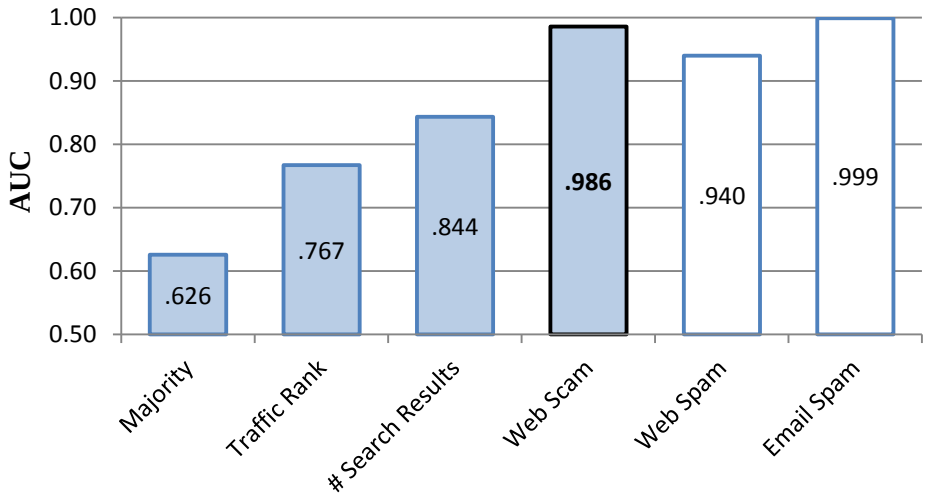


Figure 2. Comparison of detection performance to baselines (three solid bars on the left) and related tasks (two white bars on the right). AUC of web scam result compared to the number search results (bar on the left) is $p<0.004$.

Classifier	Precision	Recall	F1	AUC
LR L1 $\lambda = 1$	**0.985**	**0.953**	**0.968**	**0.986**
LR L1 $\lambda = 10^2$ 26 non-zero features out of 43	0.958	0.9349	0.946	0.9343
LR L1 $\lambda = 10^4$ 13 non-zero features out of 43	0.891	0.830	0.858	0.886
LR L2	0.874	0.906	0.889	0.928
SVM L2	0.887	0.901	0.893	0.939
SVM RBF	0.872	0.930	0.899	0.947
SVM Linear	0.886	0.896	0.890	0.922

Table 4. Scam detection performance of various classifiers on dataset of website reputation features. 10-fold cross-validation is used. LR is logistic regression. L1 and L2 are the regularization methods used. RBF is the radial basis function kernel for SVM. AUC of LR L1 in compare with SVM RBF is $p<0.01$.

Regularization parameter λ is the weight of the L1 penalty in the logistic regression log-likelihood objective function. Higher values of λ forces more feature weights to become zero. We can eliminate the zero-weight features from the data collection to improve the efficiency. In Figure 3, we show that AUC is not very sensitive to the choice of λ and it is near its optimal with only around half of the original 43 features.

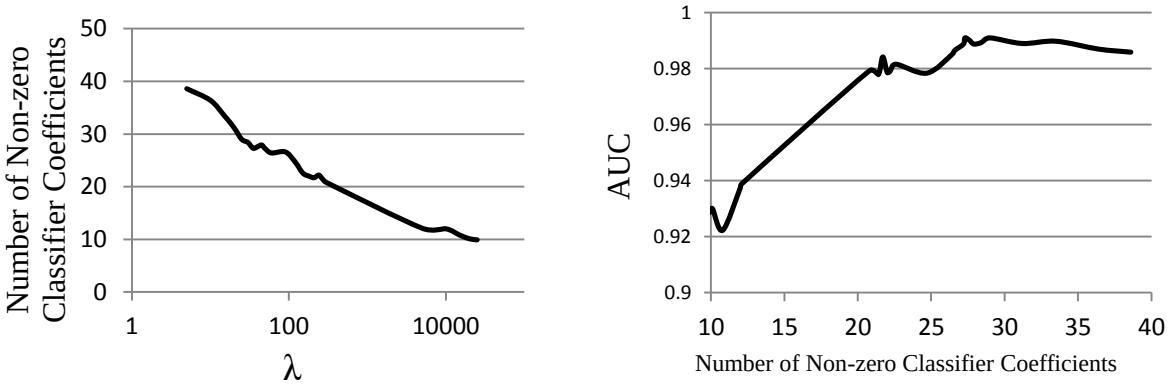


Figure 3. Effect of L_1 regularization. Classifier coefficients are **w** vector learned in logistic regression training. Higher values of λ result in fewer non-zero coefficients at the cost of some accuracy. The performance is above 0.92 even with only 10 features active.

3.2.5 Discussions

We have shown the effectiveness of the developed method in detecting malicious websites. We rely on our ability to collect and analyze the features about websites; however in some cases there are issues with this approach. We identify the related issues and address some of them in later sections:

1. *Website contents:* Malicious activity of a website is determined by its content. For example, a website selling fake medicine or hosting a virus infected file are both considered malicious, however, detecting the former is often a harder problem because it requires more advanced content analysis which is sometimes challenging even for human, as opposed to virus detection for which there are well established automated techniques.
2. *Website content ownership:* The questions of who maintains the content on a website and how much control exist on what is shared. Increasingly, the website owners allow their partners and users to provide contents for their visitors. For example, a filesharing website usually has very limited control over the contents posted. Social networking websites have some controls over the posted contents, and websites such as Wikipedia and Quora, have a thorough process for controlling the accuracy of their contents.
3. *Evolution of the websites:* The content and behavior of websites usually change over time and major changes can happen when the owners change. Such changes violate the

stationary world assumption, that is, we no longer can judge the trustworthiness of websites based on historical information. It is often not feasible to update the information quickly or even determine when such change in content or behavior has occurred.

Changes to the website availability and contents also cause problems with the historical data and learning from them. Such changes are particularly more frequent for malicious websites because they are adapting and avoiding the new techniques to discover them. For example, during the period of several months of revising our feature extraction algorithms, we faced some continuity issue when about 20% of the websites we longer available or significantly changed.

Finally, website changes, particularly for those used as online sources for the features, cause implementation issues as the wrappers have to adapt quickly and prevent degradation of input feature quality to the algorithms.

4. *Borderline websites:* Decision on what is considered malicious website or not can sometimes be difficult. An example is "who's who" type organizations (http://en.wikipedia.org/wiki/Who's_Who) that claim certain benefits with their memberships but fail to fulfill them completely. However, there is no consensus for the degree they fail and whether the issue is subjective and relates to expectation of the members or there is a malicious intent. Another example is when a website provides a treatment to a medical condition. Expensive medical experimentations may be needed to provide whether the treatment has any effect on the given medical condition.

Figure 4. Names of URL parts.

Finding suitable automated methods to address above issues is mostly for the future work, however we experimented with several potential solutions. In particular, we attempted to address the issue of the content ownership but it turns out to be a hard problem because of how the Internet domain name ownership has been designed which we will briefly explain. Figure 4

shows names of different parts of a URL. The Internet Assigned Numbers Authority (IANA) is the organization who regulates the domain name registrations. Top level Domain names are assigned based on the industry sectors such as .com and .edu, and based on countries such as .co.uk, but sometimes their usage is allowed regardless of the company's mission or location. The domain name registrars, accredited by IANA, are required to identify the individuals or companies registering the domain name, but verification can be minimal or even the registration information can be completely hidden. Often multiple domain names are operated by the same company such as `google.com` and `google.co.uk` or less obviously in the case of `amazon.com` and `imdb.com`. There are no regulations or conventions for other parts of the URL. Website owners may allow various degree of flexibility in the contents created by user; they can range from simple comments to an entire website launched under subdomains, as allowed on websites such as `wordpress.com`. In our work, we make a simplifying assumption that each unique domain name is independent and operated by a single entity which is correct for the majority of the cases. The creation of the equivalent class of the domain names is performed in Host Analyzer which we introduced earlier.

We believe that the only viable approach to adequately address some of these complex issues discussed is to combine human feedback with the automated approach. Every website has a number of visitors with varying amount of expertise in detecting malicious activities. We discovered significant amount of information related to the above issues provided by the website users. For example, users may report, sometimes in real-time on websites such as Twitter, when they are unhappy with certain product vendor.

One approach to involving human is to ask experts to label positive and negative instances of the above cases and then train a classifier based on this data; however this approach is expensive and does not scale. Therefore, we developed a crowdsourcing solution where the users report information about the various aspects of the website. We provided a mechanism for the user to inquire such information. As part of future work, expert users can be integrated to help with the more difficult cases of scam through proactive learning technique approaches [Donmez and Carbonell, 2008]. Next, we will describe SmartNotes, our crowdsourcing platform for detecting malicious websites.

3.3 Scam Detection using Crowdsourcing

We have developed an approach to collect human input and use it for deciding whether a website performs malicious activity. The tool is implemented as a browser extension, called SmartNotes, using which the users can exchange their opinion about websites or inquire about it. We provide a brief introduction to this system and will provide more information in Chapter 6.

3.3.1 Introduction

The problems with using a completely automated approach in detecting malicious websites are outlined in the Section 3.2.5. We have taken a crowdsourcing approach to address these problems. We encourage users to identify and report security threats, and then we apply machine learning to integrate their responses. This idea is analogous to user-review mechanisms, where people share their experiences with specific products or services.

3.3.2 SmartNotes

SmartNotes is our crowdsourcing platform for detecting web threat implemented as a browser add-on, currently for Google Chrome (cyberpsa.com). This platform enables users to share comments regarding websites, in ways similar to traditional social bookmarking, but the comments are focused on the trustworthiness of websites. Figure 6 shows user interface of this system. The users invoke this dialog box by clicking on a browser button and then can provide a rating for the website and some text notes describing what they think about the website content and behavior, similar to how they would do this for bookmarking. Notes can be made private as well.

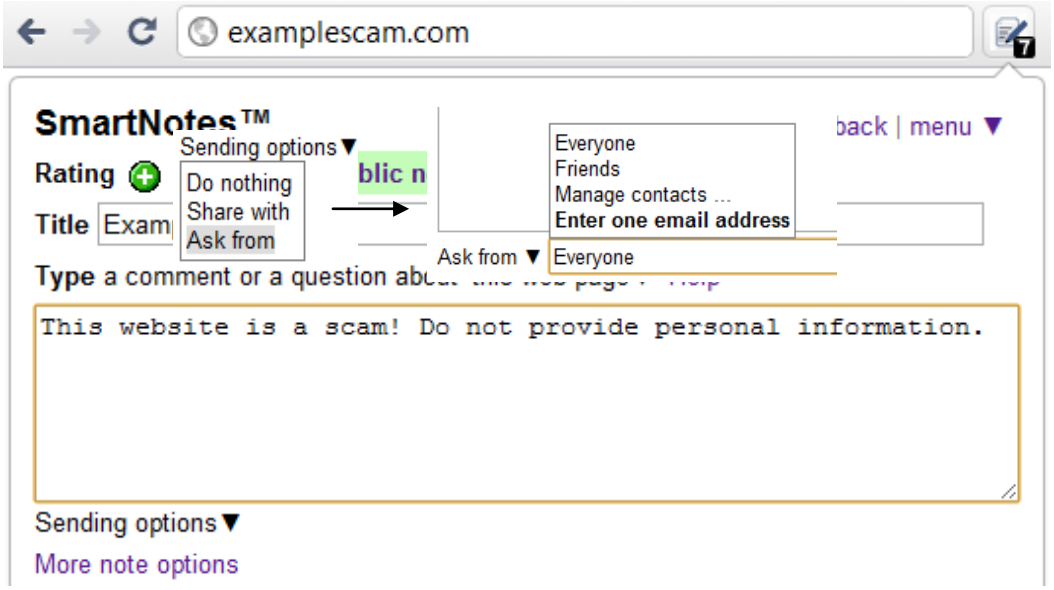


Figure 5. Question-answering feature of SmartNotes that enables users to communicate with each other about the websites.

The question-answering system within SmartNotes, shown in Figure 5, is for asking questions about specific websites. It encourages conversations among users and enables novice users to ask expert users about potential threats. While experts are often unwilling to invest their time into identifying malicious websites, they may contribute if related inquiries come from their friends through an easy-to-use mechanism. There is also more motivation for experts when they have the option to share their response with a larger group of audience. In addition to obtaining expert feedback, we gather data on the pattern of communications among users, which helps to detect biases in user opinions.

The structured data that this system collects, namely the website rating, is similar to the other reputation features that Host Analyzer (Section 3.2.1) is automatically collecting from online sources; however, the textual comments need different treatment. In Chapter 5, we experiment with various approaches of adding the user comments but we first introduce a framework that can process the comments and extract the important concepts communicated by the users about the website.

Chapter 4

Extraction of Task-Based Concepts from Text

A large portion of user-generated contents on the web is in the form of textual comments. Examples include product reviews, merchant reviews, blog comments, and website feedback. While spam and useless comments are quite common in all these instances, there is still a large number of comments with insightful information and many interested audience. For example, an electronic product manufacturer may be interested to learn about the pros and cons of their product being discussed and consumers are usually using these comments to make a better purchasing decision based on the experiences shared by other users. We use the term *language interpretation* to refer to the shallow language understanding approach that extracts "useful information" from natural language text, more specifically, from user-provided noisy contents. We proceed by defining the useful information to be the information relevant to a task, which we refer to as *task-based concepts*. We provide our motivation for this specific definition of natural language understanding and focus on addressing specific challenges.

We first define the language interpretation task as a projection from text features to concepts. The projection is modeled as a special case of multi-label text classification that is guided by a given task. We initially describe an abstract algorithm that allows for various approaches be plugged in. We then provide specific algorithms for each step and perform experiments on several datasets. In Chapter 5, we provide details on how the described concept extraction technique is used for the task of scam detection.

4.1 Language Interpretation

General natural language understanding is a challenging research. Most successful attempts have taken the approach of reducing the general problem to more specific easier problems. A group of these approaches are known as shallow language understanding, where the goal is understanding some aspects of the text. An example is the task of named entity recognition, where we are interested to identify spans of text that refers to entities such as people, locations and

organizations. Shallow understanding approaches are contrasted with deep understanding approaches, where the entire text is processed, such as various forms of parsing.

We introduce our framework that takes a goal-oriented approach to reduce the scope of the general language understanding problem. We assume that a reader of a given text has a specific task in mind and is interested to identify relevant information related to this task. For example, a physician who reads the symptoms described by a patient looks for certain phrases indicating a medical condition such as “Chest pain”. Another example is the reader of product review who usually looks for information about product features such as “Good battery life” when buying a smart phone. In Chapter 5, we describe our application area, detection of malicious websites and the reader of website comments looks for information about the website contents and behavior such as existence of false testimonials.

Many of the described techniques are general and can be applied across multiple languages; however, we have only experimented with English language text in our work. We assumed that the input text is noisy, that is, it may not be gramatically correct and may also contain misspellings. We discuss possible approaches for dealing with some of the related issues.

4.2 Task-Based Concepts

We have motivated the language interprettation as a simplified language understanding approach in which we are interested to extract task-related information from natural language text. We now describe what specifically is being extracted as concepts and compare concept extraction with other related tasks.

We define the term concept as a reference to any abstract or real object or idea, which is recognized by a group of people. This definition is very general and it can indeed refer to anything. For example, noun phrases, such as “Computer” and “Web browser”, usually refer to basic concepts. We can represent a concept using a textual description, such the ones appear in a dictionary or an encylopedia such as Wikipedia. For example, the concept “Web browser” can be represented as “a computer program to view websites on the Internet”. This representation is usually suitable for human, but it has gained some interest in recent language understanding research as better techniques are available to process the concept definition text.

Another approach to represent a concept is using a set of related words or phrases and possibly assigning some numerical weights to show their relatedness to the concept. This is the

approach that we will use and it has been traditionally known in language research as *distributional semantic*s. The idea dates back at least to 1957 and the famous quote from John Rupert Firth: "You shall know a word by the company it keeps". For example, for the concept "web privacy", we expect to see words such as "tracking" and "cookies" in the context (See Table 31 in Section 5.5.5 for more information). Representation of concepts with related words provides a straightforward setting for designing our algorithms as we will show later in this chapter.

Our general definition of concepts allows for a degenerate concept extraction algorithm. Since most words in any language refer to concepts, a text tokenizer, which converts a piece of text to its individual words, can be considered a concept extraction algorithm. We note that such trivial mapping from text to concepts may not be useful as we often are interested to know about a more complex high level idea infered by comining the words. For example, if we see a comment from a user about a website that states: "This website shares information about their users with third-party companies", we are potentially interested to know that it refers to the concept of "web privacy" rather than the concepts references by the individual words in this comment. Therefore we constrain what is admissible as concepts in our application by introducing the notion of tasks, as we eluded to earlier in this chapter. We define the *task-based concepts* as those concepts that can help us improve our performance in a task. For example, in Chapter 3 we introduced the task of detection of malicious websites. We would like to improve the detection performance by including of the textual information from the user (Chapter 5). The "web privacy" concept that we mentioned is a useful concept to know about and it potentially helps with the performance of the classification task of website into malicious or not.

What we called concepts are sometimes refered by other names and we now describe the similarities and distinctions. First, we compare with the document categorization, which is the task of assigning one or more category labels to a given document that summarize the contents of the document from various perspectives. For example, we can assign "Medical" label to a document talking about a treatment of a health condition. We can also usually find various *related* category labels. For example, many "Medical" documents can also be labeled as "Health". Finally, these related category labels can be in various degrees of specificy or granularity, for example the "Health" document may be labeled more specifically as "Heart Disease". These labels are assigned with the purpose of organizating the documents in groups,

which historically made browsing or retrieving them easier for human and therefore they are usually simple concepts. For instance, continuing our medical example, International Classification of Diseases (ICD) is a categorization scheme created to cover most health problems and is now required to be assigned to any document of medical diagnosis for the purpose of generating the statistics. Another example of category labels is *social tags*, which are words and phrases for categorizing web contents such as websites or blog articles. Concepts generalize the idea of category labels and social tags and they can be defined for any task and not just the browsing task. The concepts can be considered as answers to questions that can help with a task. In the example of the medical diagnosis from the patient description of symptoms, a question may be "where the patient is feeling any pain or discomfort?". The advantage of our task-based formulation is the guidance it provides for the concept detection task. We will describe later that our techniques are closely related to the text classification techniques that are commonly used in document categorization.

The term *attribute* often refers to the property of an object, such as the color of a bird in an image. Concepts are exactly equivalent to attributes for text. We discussed earlier that words, phrases or more generally textual *features* are simple concepts, however we usually prefer higher level and more complex ideas than what is extracted from text as features. It is unusual but possible to regard text classification or information techniques as complex feature extraction algorithms. Finally, concepts may also be called *topics* as defined in topic models (Section 2.2). The main distinction is the quality constraints that we consider for a topic and of course the task-based formulation. We use topic models as one of the basic building blocks of our techniques. The developed technique essentially improves the quality of the topics and pushes them to represent concepts of interest.

4.2.1 Characteristics of Concept Space

We outline three characteristics of the concept space that make the learning of the mapping function difficult with the traditional classifiers or dimensionality reduction techniques:

1. **Large size:** The concept space is considerably larger than typical label space of multi-label text classification algorithms. This property causes significant drop in the performance of the existing algorithms. Existing work in this area is for hierarchical text classification (Section 2.1.3), which is different than our problem setting. The set of concepts are not predefined and fixed; it is easy to generate more concepts as needed in various level of granularity. In

contrast, label sets in hierarchical text classification literature such as medical diagnosis codes and Open Directory Project for web page classification, are human curated sets and well defined. The problem with the large space of concepts is also explored in the case populating ontologies, which is the most similar to our setting.

2. **Frequency distribution skew:** An implicit assumption of most existing methods is that the label frequency is not too skewed. The experiments are usually performed by eliminating labels with frequency less than a threshold. In the case of highly skewed label frequency distributions, this procedure removes large number of labels. We measure the fit of the distribution to power law distribution, which is considered highly skewed. Power law distribution has been used to model the frequency of the words in a corpus, and we assume that the frequency of concepts has a similar distribution.
3. **Concept relations:** Concepts are often related to one another. It is again helpful to think of this relationship as the relation between words: concepts can have relations such as synonymy, high correlation, and mutually exclusion. Some type of relations can be relatively easy to detect, such as plural vs. singular and misspellings. In Section 4.2.3, we describe how we create a *concept graph*, which is our approach to consider concept relations in our model.

Dataset Name	# Docs	# Labels	Avg # Labels/Doc
20 Newsgroup	19,890	20	1
20NG Easy	1,183	2	1
20NG Hard	856	2	1
Yahoo (11 sets)			
Art	7,484	26	2.50
Business	11,214	30	1.87
New York Times (NYT)	19,340	3,031	3.13
Delicious	62,757	34,655	9.79

Table 5. Summary of the datasets for the language concept extraction experiments.

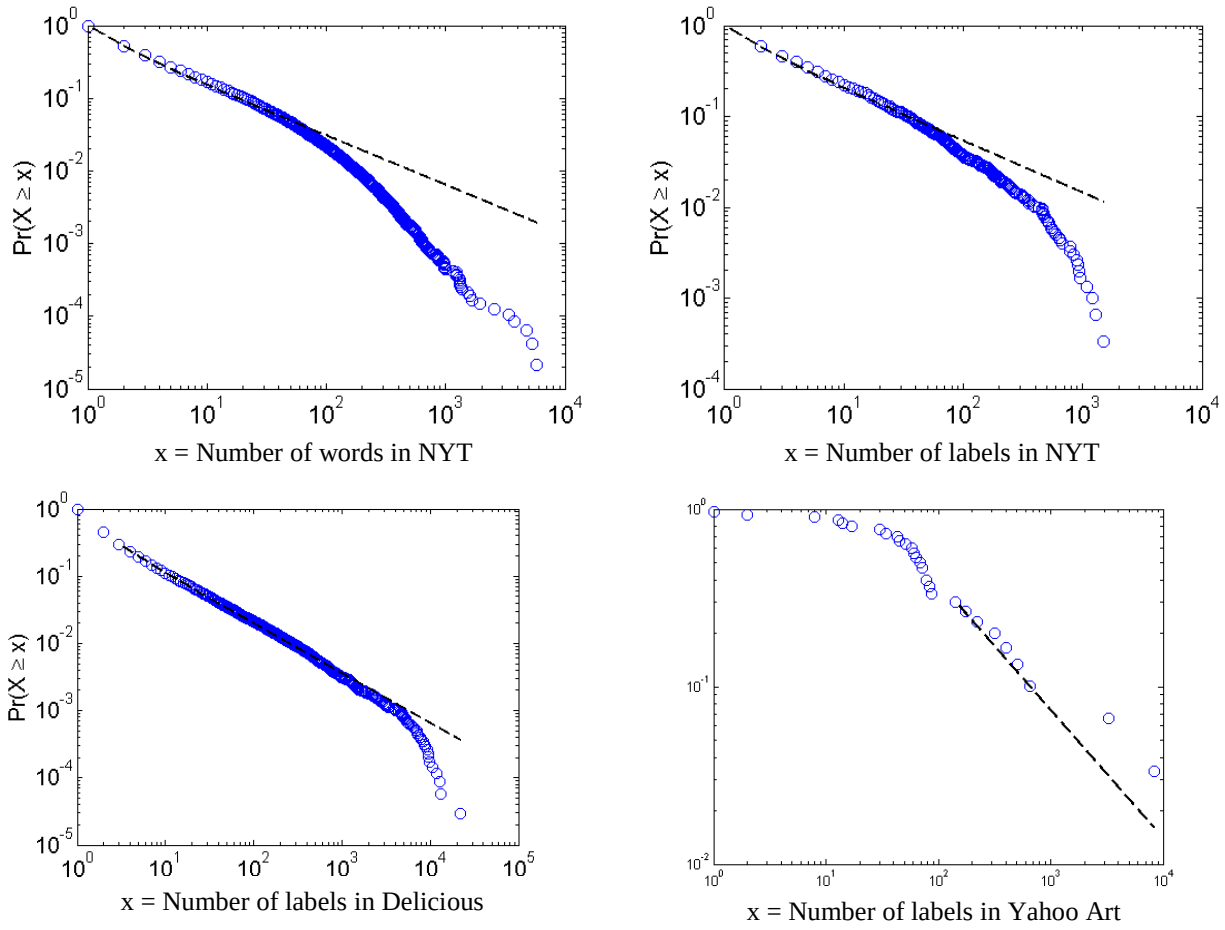


Figure 7. Power-law distribution fit of number of words and labels in several datasets. x-axis is the log number of count of tokens. y-axis is the probability of seeing a token with higher counts than the given x value. Intuitively, the faster this probability drops, skew of the distribution is higher (absolute value for line slope is the power-law parameter α).

4.2.2 Datasets

We have used four datasets to evaluate our approaches. Our focus is on the datasets that their label space is similar to the concept space with respect to the characteristics we outlined in Section 4.2.1: large, skewed frequency distribution and relations among labels. Table 5 shows a summary of the dataset statistics. To measure the skew of the label frequency distribution, we show the fit to power law distribution in compare to the word frequency distribution from one corpus for comparison.

1. **20 Newsgroups** (20NG) `people.csail.mit.edu/jrennie/20Newsgroups`: Benchmark dataset for the multi-class text classification. Class labels are 20 newsgroup topics such as Christianity and baseball. Each class has roughly 1000 written by web users. This dataset is widely used in text classification and we used it to our results reproducible and comparable to previous work.

   We frequently refer to two 2-class subsets of this data that are representative of extremes in separability of the data points estimated by the supervised classifier performance as shown in Table 6. We believe this approach to be a more realistic evaluation of our methods than synthetically constructed datasets.

   a. *20NG Easy:* 1,183 documents, 13,642 unique words. Training instances from class 4 (PC Hardware, 587 documents) and class 9 (Motorcycle, 596 documents).
   b. *20NG Hard:* 856 documents, 14,454 unique words. Training instances from class 1 (alt.atheism): 480 and class 20 (talk.religion.misc, 376 documents).

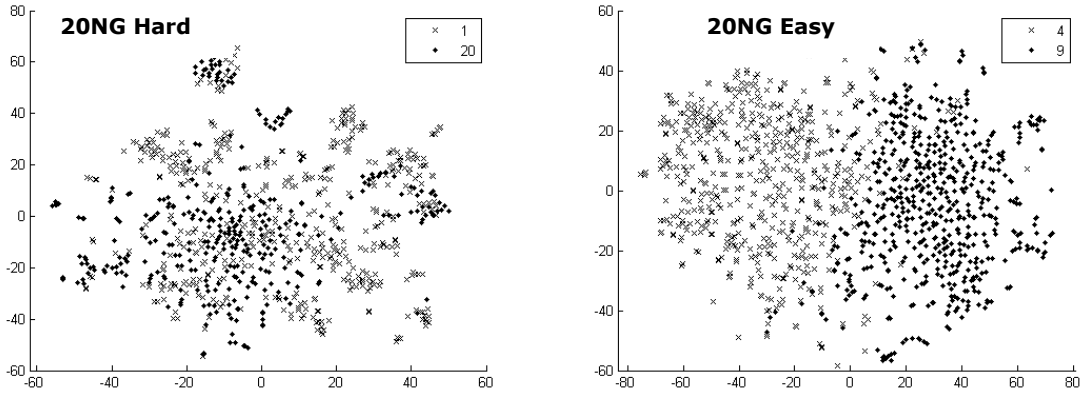


Figure 8. t-SNE embedding of the two 2-class subsets of 20Newsgroups dataset used as representative of separability extreme cases.

	Graphics	Windows	PC	Mac	Windows	For Sale	Autos	Motorcycles	Baseball	Hockey	Crypto	Electronics	Medicine	Space	Christian	Guns	Mideast	Politics	Re
Theism	0.94	0.94	0.96	0.95	0.93	0.97	0.93	0.95	0.93	0.96	0.94	0.92	0.86	0.97	0.87	0.93	0.89	0.92	0.77
Graphics		0.86	0.89	0.89	0.82	0.91	0.93	0.96	0.93	0.97	0.92	0.85	0.89	0.92	0.96	0.95	0.96	0.95	0.94
Windows			0.79	0.91	0.85	0.92	0.93	0.95	0.95	0.97	0.92	0.89	0.91	0.93	0.96	0.94	0.94	0.94	0.93
PC				0.86	0.90	0.92	0.95	0.98	0.96	0.98	0.94	0.83	0.93	0.96	0.97	0.96	0.96	0.96	0.96
Mac					0.91	0.91	0.95	0.97	0.95	0.97	0.94	0.87	0.91	0.92	0.97	0.96	0.96	0.96	0.95
Windows						0.92	0.94	0.95	0.94	0.97	0.92	0.88	0.91	0.93	0.96	0.96	0.97	0.96	0.96
For Sale							0.94	0.97	0.96	0.97	0.94	0.92	0.93	0.94	0.96	0.96	0.97	0.96	0.97
Autos								0.90	0.94	0.97	0.94	0.86	0.92	0.93	0.96	0.93	0.94	0.92	0.94
Motorcycles									0.96	0.98	0.98	0.94	0.94	0.96	0.99	0.94	0.97	0.96	0.95
Baseball										0.92	0.93	0.94	0.98	0.94	0.97	0.94	0.95	0.96	0.94
Hockey											0.95	0.95	0.96	0.97	0.98	0.95	0.97	0.98	0.96
Crypto												0.99	0.93	0.94	0.97	0.93	0.95	0.92	0.95
Electronics													0.87	1.00	0.94	0.96	0.94	0.94	0.94
Medicine														0.89	0.91	0.93	0.92	0.89	0.93
Space															0.96	0.93	0.95	0.92	0.92
Christian																0.96	0.96	0.96	0.89
Guns																	0.98	0.79	0.85
Mideast																		0.87	0.91
Politics																			0.89

Table 6. F1-measure for the linear SVM classifier trained on every pair of the 20 Newsgroups. We use this information as a measure of confusion between classes.

2. **Yahoo dataset** `www.public.asu.edu/~sji03/multilabel`: Documents are webpages from the top 11 Yahoo directories that are webpage categories such as *Art* or *Business*. Each of these top level categories has subcategories. For example, *Sport* category has a subcategory *Baseball*. Each webpage is labeled by one or more of these subcategories. Note the labels are not compatible across these 11 datasets and should be considered independently.
3. **New York Time Corpus** (NYT) available from `www.ldc.upenn.edu`: The New York Times abstracts of the articles from the year 2007, a total of 19,340 documents and 47444 unique words. Articles are manually labeled by the New York Time indexing service staff. Documents can have multiple labels and there are a total of 3031 labels.

Easiest Categories	# Docs	F1	Hardest Categories	# Docs	F1
Water	41	0.79	Air Pollution	77	0.00
Politics and Government	588	0.79	Hispanic-Americans	69	0.00
Ethics	899	0.72	Boards of Directors	66	0.00
Gifts	32	0.67	Zyprexa (Drug)	9	0.00
Elections	457	0.59	Influenza	26	0.00
US Armament and Defense	1277	0.55	Avian Influenza	18	0.00
US International Relations	1487	0.52	Sporting Goods	4	0.00
Taxation	250	0.45	Stress (Human)	13	0.00
Finances	917	0.44	Woodpeckers	1	0.00
Islam	171	0.43	Hotels and Motels	137	0.00

Table 7. Example labels from NYT dataset sorted by the F1-measure of one-vs-all linear SVM.

4. **Delicious:** `delicious.com` is a social bookmarking website where people store their bookmark and categorize them by tags. Tags are often single word category labels and depending on the user, they can refer to any aspect of the website. While there are many existing datasets for Delicious tags, we needed specific tags to replicate previous work and develop our algorithm and therefore we created a new dataset from the website. We started from the 20 tags used by Ramage et al. [2009] and crawled 4000 most common URLs for each of those tags. Then we obtained the webpages for the retrieved URLs. The final result is 62,757 documents, 129,045 unique words, and 34,655 labels. Most common tags: "reference" with 21,940, "design" with 13,059, and "education" with 13,001 web pages.

4.2.3 Concept Graph

We represent the relation among concepts as a graph. In the simplest form, this graph is a tree that organizes the concept in a taxonomy [Liu et al., 2005b; Chemudugunta et al., 2008b]. We use a weighted undirected graph $\mathcal{G} = \langle V = \{c_k\}, E = \{c_i, c_j\}\rangle$, where nodes are concepts, edges connect a pair of concepts, and edge weights are determined based on the concept pair that the edge is connecting $f: \{c_i, c_j\} \rightarrow w_{ij}$. We considered two functions in our experiments:

1. *Co-occurrence:* Based on the number of times the concepts co-occur within the same document in the training data. We use normalized point-wise mutual information and shift it from the interval $(-1, +1]$ and shift it to $(0, 1]$:

$$p(c_i, c_j) \propto \sum_d I(y_{di} = 1 \wedge y_{dj} = 1)$$

$$p(c_i) \propto \sum_d I(y_{di} = 1)$$

$$npmi(c_i; c_j) = \frac{\log \frac{p(c_i, c_j)}{p(c_i)p(c_j)}}{-\log p(c_i, c_j)} = \log \frac{p(c_i)p(c_j)}{p(c_i, c_j)} - 1$$

$$f_{co-occur}(c_i, c_j) = \frac{1}{2} \log \frac{p(c_i)p(c_j)}{p(c_i, c_j)}$$

2. *Classifier confusion:* Based on how well a binary classifier can separate the instances from the two concepts. We use the F1-measure from a *k*-fold cross validation only on instances labeled with at least one of c_i or c_j.

4.3 Identifying Task-Based Concepts

We now describe our approach for extracting the concepts from natural language text. While we describe the nuance between category labels and concepts in Section 4.2, we do not have existing labeled data to evaluate our algorithms. In Chapter 5, we will provide an approach to collect more authentic concept-annotated data, but for the purpose of developing our extraction algorithms and initial evaluations, we make the simplifying assumption that category labels are the same concepts. This assumption converts the problem of detecting concepts to an instance of text classification problem and we use the multi-label datasets introduced in Section 4.2.3. Since we later extend our algorithms for the settings that we do not have much prior knowledge about the concepts, we make minimum use of supervision and design algorithms that are robust in using small number of noisy labels and can discover novel classes. We also need to consider the special properties of the concepts space discussed in Section 4.2.1. The selected datasets share these special properties and we ensure that our methods perform well under those conditions.

We provided the background information regarding text classification in Section 2.1: we are interested in learning a function that outputs a subset of labels or a ranking of them for a given document. Additionally, our evaluations are performed in an active learning setting where we show the performance by changing the amount of labeled data.

We have developed our techniques for identifying a set of known concepts in a natural language corpus in two stages:

1. *Dimensionality reduction algorithms* (Section 2.3 for background): The objective of these algorithms is to find a mapping from the original space to a lower-dimensional space while retaining some properties of the original space.
2. *Probabilistic topic models* (Section 2.2 for background): The projection has is based on probabilistic graphical models. The lower-dimensional space dimensions are topics, which are usually multinomial word distributions. A projected document is a vector of probabilities for the document belonging to each of the topics.

DETECT-CONCEPTS-TM

Inputs: Corpus $\mathcal{D}$, Concept label set $\mathcal{C}$, Initial train set labels $\mathcal{L}_{train}$, Dev set labels $\mathcal{L}_{dev}$.

Outputs: Concept definitions $P(w|c)$, Concept assignments $P(c|d)$.

Initialize the topic model parameters $\mathcal{P} = \{\alpha, \beta, T\}$
Initialize topic labels $\mathcal{L}_{topics} \leftarrow \emptyset$
Group all the labels in one set $\mathcal{L} \leftarrow \{\mathcal{L}_{train}, \mathcal{L}_{topics}\}$
repeat
  Find a topics-words-documents assignments: $f_n(w, d, z) \leftarrow$ FIND-TOPICS($\mathcal{D}, \mathcal{P}, \mathcal{L}$)
  Find a mapping between topics and concepts: $f_l(z) \leftarrow$ LABEL-TOPICS($f_n, \mathcal{D}, \mathcal{L}$)
  Update the parameters: $\mathcal{P} \leftarrow$ UPDATE-PARAMS(f_n, f_l)
  Update the labels: $\mathcal{L} \leftarrow$ UPDATE-LABELS(f_n, f_l)
  Evaluate: EVAL-CONCEPTS(f_n, f_l, $\mathcal{L}_{dev}$)
until convergence
Output: $P(w|c) \propto \sum_d f_n(w, d, f_l(z))$, $P(c|d) \propto \sum_w f_n(w, d, f_l(z))$

Algorithm 1. Detecting the language concepts using probabilistic topic models.

4.3.1 Approach

We introduce the formal notations that we will in this chapter. Corpus $\mathcal{D}$ is a set documents $\{\boldsymbol{x}_i\}_{i=1}^{|\mathcal{D}|}$. Feature set $\mathcal{F}$ contains all features that can be extracted from documents. In many cases,

the features are unigram tokens, which are simply words. Document d has a bag-of-words vector representation $\boldsymbol{x}_i \in \mathbb{N}^{|\mathcal{F}|}$ and each vector component x_{ij} is the number of feature j in the document i. $\mathcal{C}$ is a set of concepts $\{c_k\}_{k=1}^{|\mathcal{C}|}$. Concept c_k has textual representation. The words used in this textual representation do not have to be in F; however, we usually use a subset of F. Each document in the corpus is assiged to each concept using a score vector $\boldsymbol{y_i} \in \mathbb{R}^{|\mathcal{C}|}$ and each vector component y_{ik} is the membership degree of concept c_k to document $\boldsymbol{x_i}$. We often consider a probabilistic assignment: $y_{ik} = P(c_k|\boldsymbol{x}_i)$ and therefore we have the constraints that $y_{ik} \in [0,1]$ and $\sum_k y_{ik} = 1$. We define language interpretation as learning a mapping function from the feature space to the concept space. In the special case that $|\mathcal{C}| \ll |\mathcal{F}|$, the problem is similar to the dimensionality reduction problem.

We use the topic model terminology to describe our algorithms; however, the same settings apply to dimensionality reduction methods. Algorithm 1 provides a high-level description of our approach to detecting the task-related concepts. The algorithm wraps a core topic extraction algorithm and iteratively optimizes its parameters to *guide* the topics toward concepts. We propose several alternative implementations for each step that intuitively have positive impact on the performance measures, and evaluate them empirically. We will now provide details on the functions used in this algorithm.

1. **Finding topics:** The algorithm in this step discovers the topics in a set of documents. The output is that for each word w in document d in our corpus $\mathcal{D}$, we have a topic assignment $z_{d,w}$, which is an integer between 1 and total number of topics T. We represent this output as function $f_n(d, w, z)$ that counts the total number of times the unique word w in document d has the topic z. Note that the same word in two places of a document can be assigned to different topics. We can use f_n to calculate the latent variables in the topic model, which are integrated out as part of the collapsed Gibbs Sampling:

   Word-topic distributions $$\phi_{ik} = P(z = k|w = i) = \frac{\sum_d f_n(d, w = i, z = k)}{\sum_{d,z} f_n(d, w = i, z)} \quad (1)$$

   Document-topic distributions $$\theta_{jk} = P(z = k|d = j) = \frac{\sum_w f_n(d = j, w, z = k)}{\sum_{w,z} f_n(d = j, w, z)} \quad (2)$$

   We experimented with the following choices of FIND-TOPICS:

a. Latent Dirichlet Allocation (LDA) [Blei et al., 2003]: Basic topic model algorithm which does not consider labels.
b. Labeled LDA (L-LDA) [Ramage et al., 2009]: A topic model which pre-allocates topics to labels.
c. Adaptive LDA (A-LDA): A new topic model that we will introduce in Section 4.3.1.2. The developed approach fulfils the requirements of Algorithm 1, particularly allowing for certain parameter updates of Step 3 and also quality constraints introduced in Section 4.3.1.1.

2. **Labeling topics:** After finding the topics, we need to find a mapping between topics and concepts. We define a function f_l that takes a topic as input and outputs a score for each of the concepts. This score is currently the probability of the concept given topic $P(c|z)$, approximated as probability of the label given topic $P(l|z)$. Note the use of probability as the score is optional. We follow two approaches to estimate $P(l|z)$, based on two types of labels that can be obtained from an oracle:
    a. *Document labels:* Oracle provides one or more labels for a given documents. Algorithms 2 and 3 are active learning algorithms for this approach.
    b. *Topic labels:* Oracle provides a label for a given topic. Algorithm 4 is for this approach.

    We experiment the following choices for SELECT-DOCS in Algorithms 2 and 3:
    a. *Random sampling:* Select documents uniformly at random.
    b. *Uncertainty sampling:* Train a classifier on the labeled documents and then use it to label unlabeled documents. Then select the documents that the classifier is most uncertain about the selected label.
    c. *Topic Entropy sampling:* Select documents in the order of the entropy of $P(z|d)$. The intuition is that the documents with higher entropy are more discriminative. This approach is related to density sampling scheme but for topic models. One problem with this measure is that it does not impose any diversity among the selected documents.
    d. *Document specificity score sampling:* Select documents in the order of the specificity score. This score is an improvement over the entropy measure and is explained in Section 4.3.1.1.

Algorithms 2 and 3 differ in how the function parameters are computed: Algorithm 2 trains one discriminative linear classifier per label on the topic proportion vectors of the training examples and the coefficients of the classifiers is used as the label score for each topic. Algorithm 3 uses the labeled documents to estimate word distribution for each label and then the topic label is the distribution distance of it to this label word distribution. The following are the choices for CALC-DISTANCE function which computes the distance between two probability distributions p and q:

a. Symmetric Kullback–Leibler [Kullback and Leibler, 1951]:

$$KL_{sym}(p,q) = KL(p||q) + KL(q||p)$$

$$KL(p||q) = \sum_i p(i) \ln \frac{p(i)}{q(i)}$$

b. Jensen-Shannon [Lin, 1991]:

$$JS(p,q) = \frac{1}{2} KL(p||\frac{q+p}{2}) + \frac{1}{2} KL(q||\frac{q+p}{2})$$

c. Metric distances:

$$L_1(p,q) = \sum_i |p(i) - q(i)|$$

$$L_2(p,q) = \left(\sum_i \big(p(i) - q(i)\big)^2 \right)^{\frac{1}{2}}$$

$$\text{Jaccard}(p,q) = \frac{\sum_i I(p(i) > 0 \ \wedge q(i) > 0)}{\sum_i I(p(i) > 0 \ \vee q(i) > 0)}$$

Where can obtain the topic label directly (Algorithm 4), oracle is provided with top N most probable words of a topic based on the $P(w|z)$, then outputs one or more of the following:

a. Whether the topic is “Useful” or “Not useful”.
b. Category label and its degree of relevance: Low, Medium, or High.
c. A subset of the top N words which are relevant to the label provided, or alternatively, the words that are considered not fitting the topic well.

We construct a classifier for each label by setting the classifier weight for each topic based on the input: Low = 0.25, Medium = 0.5, and High = 0.75. If the label is not provided

or voted “Not useful”, the classifier weight is 0. If the topic is voted “Useful” without the degree of relevance, “Medium” relevance is used. To assist the human oracle with the topic labeling task, we also show highly probable documents for the selected topic.

The following are the choices for SELECT-TOPICS:

a. Marginal probability of the topic $P(z)$
b. Entropy of the topic $E_{P(w|z)}[-\log_2 P(w|z)]$
c. Topic cohesion scores, which are introduced in Section 4.3.1.1.

DISC-LABEL-TOPICS

Inputs: Topic model output $f_n(w, d, z)$, Corpus $\mathcal{D}$, Train set document labels obtained from an oracle $\mathcal{L}_O$.

Output: Topic label scores.

$\widetilde{\mathcal{D}} \leftarrow \emptyset$

for each document d in $\mathcal{D}$ **do**

    $\widetilde{\mathcal{D}} \leftarrow \widetilde{\mathcal{D}} \cup \boldsymbol{\theta}_d$ compute the document topic proportion from equation 2.

batch ← SELECT-DOCS($\widetilde{\mathcal{D}}, \mathcal{L}_O$)

$\mathcal{L}_O \leftarrow \mathcal{L}_O \cup$ LABEL-DOCS(batch)

h = TRAIN-CLASSIFIER($\mathcal{L}_O$)

Output: weight parameters from the linear classifier h

Algorithm 2. Discriminative approach for LABEL-TOPICS.

GENR-LABEL-TOPICS

Inputs: Topic model output $f_n(w, d, z)$, Corpus $\mathcal{D}$, Train set document labels obtained from an oracle $\mathcal{L}_O$.

Outputs: Topic label scores.

batch ← SELECT-DOCS($\mathcal{D}, \mathcal{L}_O$)

$\mathcal{L}_O \leftarrow \mathcal{L}_O \cup$ LABEL-DOCS(batch)

Compute the concept-word distribution $P(w|c)$ for $\mathcal{L}_O$

for z = 1 to T **do**

    $w_z = \text{CALC-DISTANCE}(P(w|c), \boldsymbol{\phi}_d)$

Output: w

Algorithm 3. Generative approach for LABEL-TOPICS.

ORAC-LABEL-TOPICS

Inputs: Topic model output $f_n(w, d, z)$, Corpus $\mathcal{D}$, Train set document labels obtained from an oracle $\mathcal{L}_O$.

Outputs: Topic label scores.

batch ← SELECT-TOPICS($f_n(w, d, z), \mathcal{L}_O$)
Ask oracle to provide the topic labels for the batch (see descriptions for details)
Convert the oracle labels to the classifier weights w
Output: w

Algorithm 4. Direct topic labeling approach for LABEL-TOPICS.

3. **Updating parameters:** The information we obtain from projecting the documents to topics f_n and then topics to the concepts f_l is useful in optimizing the parameters of the model. The three parameters of the model are α, the prior for the document-topic distributions; β, the prior for word-topic distribution, and T, the number of topics. The optimization of the priors is sometimes performed as part of training of the topic models [Blei et al., 2003; Wallach et al., 2009]; however, we need to consider task performance and topic quality. Our approach generalize the related work where separate optimization problem is defined to set the priors, such as Dirichlet-multinomial regression (DMR) [Mimno and McCallum, 2008] and Yahoo news personalization model [Low et al., 2011]. The following are specific parameter update methods:
    a. *Updating α:* We can use the mapping from the topics to labels to adjust the document priors. In L-LDA, the document specific priors are obtained by keeping the α vector elements that corresponds to the label and setting other elements to zero. In other words, all documents share the same prior but it is truncated based on the document labels. We can set the document priors to be different than each other. We considered training a discriminative classifier based on the current labeled instances and then use the classifier scores for a subset of unlabeled data as an estimate for α. The estimates change in each iteration because the number of labeled and the topic-label assignments can change.
    b. *Updating β:* We consider the following approaches to update β:

i. *Word-label distributions* $P(w|l)$: We can estimate a conditional word-label probability distributions using the labeled documents. Smoothing can be used to improve the estimates, which are based on a small training set. We update the set of labels in Step 4 of the Algorithm 1 and therefore the value of β can change between iterations.

ii. *Topic Specificity Score* (TSS_2): We will introduce this measure in Section 4.3.1.1. It provides a score for each word that estimates its discriminative power based on the performance of the topic model. We can set the β for all topics to be this normalized score to force the model to use the words with potentially higher discriminative power more often. This approach does not make use of labels. We can use mutual information with the labels instead of or in combination with this measure. In each iteration, the $P(w|z)$ changes and therefore the TSS_2 scores and β will be different.

c. *Updating T:* Priors improve the prediction performance of the model by encoding additional useful information. The number of topics is also a critical parameter because it has substantial effect on the number of the parameters and the model performance. The common approaches for choosing the number of topics are the following:

i. *Choose a large number:* We will explain in our experiments that sometimes, simply choosing a "large enough" number suffices; however, this approach increases the computations unnecessarily and sometimes degrades the prediction performance.

ii. *Optimize on the held-out data:* We can maximize the log-likelihood of the model on a held out data by gradually increasing the number of topics. This approach is computationally intensive due to many evaluations needed and it also most likely will get stuck in local optima.

iii. *Use a non-parametric approach:* In non-parametric approaches, the number of parameters grow automatically with data. The non-parametric version of LDA is Hierarchical Dirichlet Process (HDP), which defines a Dirichlet process (DP) prior instead of the Dirichlet distribution on the topic mixtures [Teh et al., 2006]. One approach to generate a Dirichlet

process distribution is using the Chinese Restaurant Process (CRP) [Aldous, 1985]. Upon convergence, HDP can have any number of topics, often shown with an infinite sign on the plate diagram. In each re-sampling of z variable, either a new topic is created with a probability proportional to the *concentration parameter* of the DP or one of the existing topics is used with probability proportional to how many times they are used. In practice, CRP finishes with more topics than the optimal number of topics and it also requires significant amount of book keeping in implementation. More crucially, CRP only uses the number of times the existing topics are used and has no considerations for the relation among topics or their quality. It also does not initialize the new topic and rarely reduce the number of topics rather than increasing.

We propose a simple alternative approach to choose the number topics: we analyze topics at every iteration and decide how to apply the following operations to each of them.

1. **Split topic t:** We add a new topic t_{new}, and then randomly reassign a subset of word-topic assignments of $z = t$ to t_{new}.
2. **Merge t_1 to t_2:** We reassign all topic assignments of $z = t_1$ and $z = t_2$ to t_2, and then delete t_1.
3. **Leave as is.**

We generate a *plan* at each iteration which determines the sequence of merge and split operations. We choose a convention to execute the plan linearly in the order of the topic index. Figure 9 shows detailed execution steps of a plan. We describe two strategies for creating the plans in the experiments.

	Topics										
Steps	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Merge topic 2 to topic 1		**1**	S	3		S	S		8	S	
Split topic 2	.	**S**	**2**		S	S		7	S		
Merge topic 4 to topic 2		.	.	**2**		S	S		7	S	
Split topic 4		.		**S**	S		6	S			
Split topic 6				.	.	**S**		7	S		
Merge topic 9 to 8						.	.		**8**	S	
Split topic 10								.		**S**	
Done.										.	.

Legend: S: Split into two topics, *Number*: Merge to the shown topic number.

L Topic being processed by the current step

. Topics were affected by the previous step

Figure 9. Demonstration of how an example split/merge plan is executed. Total number of topics after 4 splits and 3 merges is 11. Note that the indices are adjusted as the plan is executed. As a convention, the larger topic index always merges to the smaller one that is the one stored in the plan.

4. **Updating labels:** If we have obtained any document labels from the oracle as part of the topic labeling step (Step 2), we can add them to the labeled document set and they will be used in the next iteration. We also have obtained topic labels either from oracle or using a discriminative or generative approach from document labels. The topic labels can be used for labeling the unlabeled documents. This approach is related to self-training and transductive learning, which were discussed in Section 2.4. We observed that the algorithm benefits from distinguishing the labels provided by human and those provided by the classifier, and therefore we use α and the label confidence mechanism of Adaptive LDA to include both sets of labeled documents separately instead of combining them.
5. **Evaluating:** This step measures the performance of the algorithm and stops the loop when the change in one of the convergence measures falls below a threshold. We consider two types of measures that signify the underlying objective function that we are optimizing with our algorithm.
    a. *Task-based:* A labeled development set which is not used in the learning of the model or optimization of the parameters can estimate the test set performance. We

can define this measure based on the text classification task or based on the final classification task for which the algorithm is extracting the concepts.

b. *Topic quality-based:* We can also use unsupervised convergence criteria based on the topic quality measure that are defined in Section 4.3.1.1. The specific metrics are defined in the experiment section.

4.3.1.1 Topic quality

Topic models are useful in capturing the thematic aspects of the corpus [Blei et al., 2003]; however, there are several drawbacks in using these models for our task of extracting concepts:

1. *Stability:* The model objective function is non-convex and therefore there are no guarantees that model will discover the same topics in every run. We do some experiments to illustrate and measure this issue and demonstrate its relation to the input data and the parameters.
2. *Interpretability:* The model uses information from word co-occurrences under the exchangeability assumption, which states the joint probability distribution is invariant to permutation of the entities. An example of this assumption is the commonly used bag-of-words assumption. One side-effect of this assumption is that the models may not be meaningful to the human. In practice, one can usually expect to obtain reasonably well defined topics and the usual trick is to run topic models with a large number of topics and manually identify *good* topics. Non-parametric methods [Teh et al., 2006] allow the data to choose the number of topics, but in practice they do not produce more interpretable topics [Boyd-Graber et al., 2009]. We explore how to measure and improve the quality of the topics. Human interpretability of the topics is independent than predictive performance of the model for the task. We are interested in both, especially when we need to ask human to evaluate the topics.

The properties of good topics are not well studied for the probabilistic topic models. Inspired by cluster quality in clustering algorithms, we quantify inter-topic cohesion and intra-topic separation. Our approach dynamically defines word constraints as part of the learning procedure and improves the cohesion and separation. Word constraints can be in the form of eliminating words or restricting them to certain topics. Topic models allow the same word to be assigned to multiple topics, which makes intuitive sense; however, when we seek to create better label–topic

mapping, assigning words to many topics is less desirable. We observe that if we can restrict words to topics, we may have a better representation of the topics. This approach results in concentration of the probability mass on fewer words. It decreases of the entropy of the topic, which is one of the topic coherence measures. Word constraints can be considered another approach for the entropy regularization in topic models [Newman et al., 2011].

SP-LDA

Input: Corpus $\mathcal{D}$

Output: Topic model $\{z, \phi, \theta\}$

1 Initialize the topic model parameters: $\mathcal{P} \leftarrow \{\alpha, \beta, z, \phi, \theta, T\}$
2 Initialize the word constraints $WC \leftarrow \emptyset$
3 **repeat**
4 $\mathcal{P} \leftarrow \text{AdaptiveLDA}(\mathcal{D}, \mathcal{P}, WC)$ (Section 4.3.1.2)
5 Update the word constraints $WC \leftarrow \text{FIND-WORD-CONST}(\mathcal{P})$
6 **until** convergence
7 **Output:** $\{z, \phi, \theta\}$

Algorithm 5. Extension of the Latent Dirichlet Allocation for generating sparse topics.

Algorithm 5 outlines our approach to increasing the topics quality. Adaptive LDA is the topic model that is capable of handling word constraints (Section 4.3.1.2). Algorithm 6 shows the modification to the Gibbs sampling procedure [Griffiths and Steyvers, 2004] for incorporating the word constraints that are either elimination of the words or constraining to a certain topic. The second part is FIND-WORD-CONST, which discovers the word constraints as detailed in Algorithm 7. Intuitively, the modified algorithm defines the constraints to achieve the clustering quality property discussed previously: intra-topic separation and inter-topic coherence. We define two measures in this algorithm:

1. **Topic specificity:** Identify words that are not discriminative across topics and mark them for elimination. The discrimination power is determined by what we call topic specificity score which is computed in GET-WORD-TSS. The following are the choices of this method we will experiment with:

a. *Term frequency:* It is one of the common heuristics to identify the stop words, but it can eliminate discriminative words.
b. *Entropy of $P(z|w)$:* Discriminative words have high mutual information with labels, that is, they co-occur more often with the labels they are discriminating, and less often with other labels. Hence a low entropy for $P(l|w)$ is a good indicator of word w being discriminative, but this distribution is unknown without knowing the labels. Since we assume correspondence between topics and labels, then we use $P(z|w)$ as an approximation.
c. *Weighted entropy:* We observed two problems with using the entropy. First, it does not account for the events that have not occurred in probability distributions. In our case, the events correspond to topics. For example, if a given words is assigned equal number of times to topic 1 and 2, the entropy value remains the same regardless of how many other topics exist in the model to which the word is not assigned. We are interested in a score that considers all topics. Intuitively, intra-topic separation changes when the total number of topics increase and a good model should be rewarded to continue using few topics even when many are available. Second problem has to do with the fact that the entropy operate on probability distribution, which computes the relative frequency of the events and their magnitude is ignore during the normalization. The absolute frequency of the events is an estimate of our confidence and we would like to include it in our score.

   As an example, we use the notation $[n_1, n_2, \dots]$ where n_z is the number of times a given word for which we are estimate the discriminative power is assigned to topic z and $H(.)$ is the entropy of this frequency counts after the normalization:

   - Unobserved counts do not change the entropy:
     $H([1,1,1]) = H([1,1,1,0]) = H([1,1,1,0,0]) = \cdots = 1.585$
   - The magnitude of the assignments do not matter:
     $H([1,1,1]) = H([2,2,2]) = H([3,3,3]) = \cdots = 1.585$

We define a new score that captures the topic specificity better. For a given word w, we have a $P(z|w)$ after running the topic model with T topics. Our topic specificity score is:

$$\text{Word count: } n(w) = \sum_{z=1}^{T} n(w, z)$$

$$TSS_1(w) = \left(\frac{\log_2 n(w)}{\log_2 \max_w n(w)}\right) \cdot \left(\frac{\log_2 T + 1}{E_{P(z|w)}[-\log_2 P(z|w)\,] + 1}\right).$$

The first part of the score is the normalized word count. The second part is the entropy of the uniform distribution divided by the entropy of the word-topic distribution. One way to intuitively understand this score is TFIDF score, where the IDF part of replaced by the entropy of the word-topic distribution. We also experimented with a bounded version of this score, which stays within [0,1]:

$$TSS_2(w) = \left(\frac{\log_2 n(w)}{\log_2 \max_w n(w)}\right) \cdot \left(1 - \frac{E_{P(z|w)}[-\log_2 P(z|w)]}{\log_2 T}\right).$$

To evaluate how well this score can identify the discriminative words, we use the class labels to calculate the mutual information of the words and the labels. Table 7 shows the words that have the highest mutual information with the two labels in 20NG Easy dataset: computer hardware and motorcycle.

Sorted by mutual information					Sorted by Topic Specificity Score (TSS_2)					
Words	**MI**	**Entropy**	**TSS_1**	**TSS_2**	**MI Rank**	**Words**	**MI**	**Entropy**	**TSS_1**	**TSS_2**
dod	0.33	1.00	1.47	0.36	5720	edu	0.00	0.00	3.27	0.58
bike	0.25	0.87	1.60	0.39	43	drive	0.04	0.00	3.12	0.55
card	0.14	0.84	1.53	0.37	13	scsi	0.08	0.00	3.04	0.54
ride	0.12	0.69	1.53	0.36	163	you	0.02	0.78	2.05	0.50
mb	0.11	0.79	1.56	0.38	613	hard	0.01	0.00	2.71	0.48
pc	0.10	0.99	1.34	0.33	21	disk	0.07	0.00	2.67	0.47
dos	0.10	0.36	1.95	0.42	14	drives	0.08	0.00	2.63	0.47
controller	0.10	0.74	1.56	0.37	457	bit	0.01	0.00	2.63	0.47
bus	0.09	0.99	1.36	0.34	15	dx	0.08	0.00	2.60	0.46
riding	0.09	0.04	2.30	0.42	155	help	0.02	0.00	2.56	0.45
motorcycle	0.09	1.25	1.06	0.27	716	were	0.01	0.00	2.53	0.45
bikes	0.08	0.89	1.24	0.31	19	windows	0.07	0.00	2.52	0.45
scsi	0.08	0.00	3.04	0.54	33	bios	0.05	0.00	2.51	0.45
drives	0.08	0.00	2.63	0.47	467	had	0.01	0.41	2.02	0.44
dx	0.08	0.00	2.60	0.46	730	really	0.01	0.10	2.33	0.44
ide	0.08	1.00	1.38	0.34	13149	get	0.00	0.63	1.87	0.44
bmw	0.07	0.90	1.30	0.32	3256	don	0.00	0.55	1.90	0.44
system	0.07	0.99	1.40	0.35	3389	did	0.00	0.04	2.39	0.44
windows	0.07	0.00	2.52	0.45	371	take	0.01	0.00	2.42	0.43
article	0.07	1.03	1.54	0.38	48	mhz	0.04	0.00	2.42	0.43

Table 8. Comparison of the sorting with mutual information and the topic specificity score.

Approach	ρ
Word Frequency	0.15
Log(Word Frequency)	0.39
Entropy of P(Topic|Word)	0.20
TSS_1	0.44
TSS_2	0.46

Table 9. Pearson correlation coefficient of the various topic score options with the mutual information.

2. **Topic coherence:** Identify words that are well represented by a topic and constrain them to the topic that represent them. The following are the choices for GET-TOPIC-COHR as the coherence measures:
    a. *Entropy of* $P(w|z)$: Lower entropy of $P(w|z)$ implies a "sharper" the distribution which puts more probability mass on smaller subset of words in compare with to

the uniform distribution which has maximum entropy. This may be a poor measure of coherence because the probability mass can be focused on the irrelevant words.

b. *Pairwise PMI*: Find the point-wise mutual information among all pairs of the top N words of topic based on $P(w|z)$ [Newman et al., 2010].

c. *Human score:* We evaluate based on the topic labels obtained from the human using the same scheme described in Algorithm 4 but a similar approach is used in related work as well [Boyd-Graber et al., 2009; Newman et al., 2010; Mimno et al., 2011].

LDA_{WC}

Inputs: Corpus $\mathcal{D}$, Word constraints WC,

Outputs: Topic model $\boldsymbol{z}$

---

1 Initialize $\boldsymbol{z}$
2 **repeat**
3 **for each** document d in $\mathcal{D}$ **do**
4 **for each** word w in document d **do**
5 **if** $w \in WC$ **then**
6 **if** $WC(w) > 0$ then
7 $z_{d,w} \leftarrow WC(w)$
8 **else**
9 Ignore the word.
10 **else**
11 Perform the usual $z_{d,w}$ update.
12 **until** convergence
13 **Output:** $\boldsymbol{z}$

Algorithm 6. Modifications to the Gibbs sampling to enforce the word constraints. Steps 5-9 check for the word constraints.

FIND-WORD-CONST

Inputs: Topic model $f_n(w, d, z)$, Vocabulary V, Topic Specificity Threshold θ_{TSS}, Coherence assignment threshold θ_{COHR}

Outputs: WC

for each word w in vocabulary V **do**
    Compute the topic specificity score TSS ← GET-WORD-TSS(w, f_n)
    Find the best topic for this word: $z^* \leftarrow argmax_z P(z|w)$
    Compute the coherence of the best topic for the word: $c \leftarrow$ GET-TOPIC-COHR(z^*, f_n)
    if $TSS < \theta_{TSS}$ then
        WC(w) ← 0
    else if $c > \theta_{COHR}$ then
        WC(w) ← z^*

Algorithm 7. Find the word constraints.

In our experiments, we compare the prediction performance of our approach with the approach of applying the constraints as the priors for both for the word-topic assignments and also the document-topic assignment. One advantage of *hard* thresholding as opposed to adjusting the prior is the gained performance from the reduction of the vocabulary size.

Our approach of defining dynamic word constraints improves on one of the most common feature selection preprocessing steps used in topic modeling and more generally in most machine leaning and information retrieval applications. Two groups of words are often eliminated:

1. *Stop words* such as the word "the" that happens too frequently in documents and therefore provide little discriminative information. The typical approach is to use predefined lists and ignore them as part of a preprocessing; however, there are two concerns with this approach:
    a. Some of the words, especially on longer versions of stop word lists, are useful for the formation of the topics and shorter stop word lists do not produce useful results.
    b. Some domain specific words act as stopwords and they also need to be eliminated. One heuristic is using the word counts, which eliminates many useful words as well.

2. *Rare words* that appear only a few times in the corpus are eliminated because they carry little information and cause large performance cost. The typical approach is picking an arbitrary threshold for minimum corpus term frequency and eliminating words occurring less than this threshold. The frequency cut-off approach can also be used to approximate stop words as mentioned.

High mutual information words which are **on** the stop word lists				**Low** mutual information words which are **not on** the stop word lists			
Words	**TF**	**MI**	**TSS_2**	**Words**	**TF**	**MI**	**TSS_2**
thanks	443	0.06	0.30	buy	207	0.00	0.31
using	290	0.05	0.33	time	522	0.00	0.21
use	568	0.04	0.20	work	435	0.00	0.27
her	201	0.03	0.30	local	203	0.00	0.42
does	548	0.03	0.20	make	327	0.00	0.30
uses	118	0.03	0.38	speed	274	0.01	0.19
she	116	0.03	0.38	good	486	0.01	0.28
in	4805	0.02	0.08	hard	431	0.01	0.48
com	1822	0.02	0.35	drive	1087	0.04	0.55
was	1329	0.02	0.43	writes	1284	0.06	0.35

Table 10. Topic specificity score provides a soft feature selection approach instead of the usual fixed stop words.

4.3.1.2 Adaptive LDA

The probability of corpus using the Latent Dirichlet Allocation model [Blei et al., 2003] is given by the following equation:

$$p(\mathcal{D}|\alpha,\beta) = \prod_{d=1}^{|\mathcal{D}|} p(\boldsymbol{w}_d|\alpha,\beta) = \prod_{d=1}^{|\mathcal{D}|} \int p(\theta_d|\alpha) \left(\prod_{n=1}^{n_d} \sum_{z_n=1}^{T} p(z_{dn}|\theta_d) p(w_{dn}|z_{dn},\phi_{z_{dn}}) \right) d\phi d\theta_d$$

The posterior distribution is intractable, therefore approximate inference algorithms is used. We use Gibbs sampling approach in our work [Griffiths and Steyvers, 2004]. Assuming the hyperparameters α and β are provided and fixed, there are three set of random variable that we need to sample: θ, ϕ, and z; however, this approach converges very slowly. Instead, it is possible to integrate out θ and ϕ and derive a *collapse* Gibbs sampling procedure in which only z are

sampled. In the equations below, subscript $-i$ indicates all variables except i. The posterior distribution of z according to the Bayes rule is the following:

$$p(z_i|\boldsymbol{z_{-i}}, \boldsymbol{w}_d) \propto p(w_i|z_i, \boldsymbol{w_{-i}})p(z_i|\boldsymbol{z_{-i}})$$

Using the Dirichlet-multinomial conjugacy, we can obtain the posterior based on the number of word assignments to topics for the document $\boldsymbol{w}_d$. $n^d_{-i,z}$ is the number words within the document d have the topic z except for the current word. $n^w_{-i,z}$ is the number times word w has topic z except for the current word. Any missing subscript or superscript indicates the summation over that index:

$$p(z_i = z|\boldsymbol{z_{-i}}, \boldsymbol{w}_d) \propto \frac{n^w_{-i,z} + \beta}{n_{-i,z} + W\beta} \cdot \frac{n^d_{-i,z} + \alpha}{n^d_{-i} + T\alpha}$$

Gibbs sampling procedure iterates over all words in all documents and computes the posterior distribution of z variables by plugging in the counts for $z_i = 1, 2, .., T$. Each time the counts are updates, a new topic for the current word is sample from the latest posterior.

Labeled LDA (L-LDA) [Ramage et al., 2009] modifies this process by projecting the globalα to a document specific α_d using a matrix defined by the document labels. $\boldsymbol{\Lambda}^d$ defines the topic-label assignment and α^dwill have only the components of the α vector for the topics for that the document has the corresponding label.

$$\boldsymbol{\Lambda^d} \sim Bernouli(.|\boldsymbol{\Phi})$$

$$\boldsymbol{\lambda^d} = \{k|\Lambda^d_k = 1\}$$

$$\boldsymbol{L^d}: |\boldsymbol{\lambda^d}| \times T$$

$$L^d_{ij} = \begin{cases} 1 \text{ if } \lambda^d_i = j \\ 0 \text{ otherwise.} \end{cases}$$

$$\boldsymbol{\alpha^d} = \boldsymbol{L^d} \times \alpha$$

In other words, the document priors are *constrained* to the topics corresponding to their labels. We now describe specific features of Adaptive LDA (A-LDA). A-LDA has L-LDA as its special case and provides several other essential capabilities for our iterative concept extraction algorithm.

1. *Extended priors:* The initial topic models based on LDA used symmetric Dirichlet prior, which are usually set experimentally [Griffiths and Steyvers, 2004] or chosen by empirical Bayes [Blei et al., 2003]. Recent work have shown the importance of using rich priors for topic models through approaches such as optimizing asymmetric

priors [Wallach et al., 2009] and defining structured prior to incorporate the domain knowledge [Andrzejewski et al., 2009]. We define both priors α and β at the lowest level, which is the same number of parameters as θ and ϕ; however, the parameters can be coupled to each other as needed to avoid overfitting and decrease the computational costs during optimization. In the graphical model notation, this change is equivalent to moving α in the documents plate, and moving β in the topics plate.

2. *Label confidence:* Extending the α parameter provides the flexibility we need for the document specific labels; however, we sometime need to consider the confidence for the label separately. The label confidence can also be represented using the α but the results are different than when it is considered as a separate observed random variable in the model, which is independent from the word and the topic mixture θ given the topic. We represent the confidence as the probability that the label is correct. We call the new random variable c, which take its value from the labels of the topic z_i:

$$p(z_i|\boldsymbol{z_{-i}}, \boldsymbol{w}_d) \propto p(w_i|z_i, \boldsymbol{w_{-i}})p(z_i|\boldsymbol{z_{-i}})p(c = label(z_i))$$

   This formulation is similar to supervised LDA [Blei and McAuliffe, 2007]; however, the random variable is the label confidence and not the document label.

3. *Word-level constraints:* Extending the β prior enabled us to initialize the topic with more information about the task. We experiment with two alternatives, which were discussed in Step 3 of the Algorithm 1: word-label distribution and topic specificity score. There are other constraints, however, that need special attentions. We described an approach for applying the word constraints in Algorithms 5, which involves a modification to the Gibbs sampling algorithm shown in Algorithm 6. This extension of the algorithm is general and can be used to impose other rich constraints to incorporate additional information in the model. For example, we can use examine a word context for the cues of negation and restrict or enforce allocation to a subset of topics. We leave exploring the options to use the word-level constraints to the future work.

4. *Flexible topic-label allocation:* L-LDA considers a single topic per label. We generalize their model to allow any assignment of the topics and labels. We experiment with equal number of topic-label assignment with the L-LDA title and

provide several heuristics using the concept graph that assigns different number of topics for each class. Details of the approaches are provided in the experiment section. The model provides the capability of creating topics shared among multiple labels, which is inspired by Error Correcting Output Codes (Section 2.1).

4.3.2 Experiments

We evaluate the proposed language interpretation framework from various aspects to answer the following questions:

1. How do various models rank in terms of their performance in predicting the label?
2. How does this performance change with respect to the special properties of concept space discussed in Section 4.2.1?

We begin by describing the default setting for all the experiments. The datasets are introduced in Section 4.2.2. The labels in these datasets are used as an approximation for the concepts. The evaluation approach is that if the concepts are reasonable representation for the documents, then a classifier should perform well on classifying the projected documents. The evaluation of predictive performance for the labels is using the same metrics introduced in Section 3.2.4: precision, recall, F-measure, and AUC which is the area under the ROC curve. For topic models, we also measure the perplexity, which is the normalized and transformed log likelihood of the held out data D_{test} [Blei et al., 2003; Griffiths and Steyvers, 2004]:

$$perplexity(D_{test}) = \exp\left\{-\frac{\sum_{d \epsilon D_{test}} \log p(\boldsymbol{w}_d)}{\sum_{d \epsilon D_{test}} N_d}\right\}$$

We introduced a new alternative to the common stopword removal; however, in several places we have used predefined lists or frequency based method, in which words occuring less frequent than 10 or more frequent than half of the documents in corpus are removed. 20NG and Yahoo datasets have pre-defined train and test splits, which we may use depending on the experiment; however unless otherwise noted, all experiments use 10-fold cross validation. For the infrequent labels, the *stratified* cross validation is used which maintains the ratio of the positive and negative examples in each fold because, otherwise, especially in the case of infrequent labels, some folds may not have any positive examples.

The cost parameter of the linear SVM $C = 10$ and the parameter of the L_1 and L_2 regularization $\lambda = 10$ for all experiments. We verified that our result is not sensitive to the choice of this parameter for the values in the neighbourhood of the selected values.

Statistical significance has been measure using the paired t-test and the permutation test, as described in Section 3.2.4. Additional experimental setup details has been provided within each of the next subsections.

4.3.2.1 Input Parameters Sensitivity

We analyze the sensitivity of the prediction performance of the topic models to their input parameters: number of topics T, document-topic symetric prior α, and word-topic prior β.

In the Gibbs sampling procedure we used for inference in topic model, there is no theoretical approach for verifying the convergence. The common approach is to run the procedure for a fixed number iterations [Griffiths and Steyvers, 2004]. Among many heuristics exist for verifying convergence, we used the one that stops when the change in the moving average of log-liklihood of the model falls below a threshold. We are, however, more interested to evaluate the impact of the convergence on predictive performance. We run LDA on 20NG Easy and Hard datasets with the following parameters: $T = 10$, $\alpha = 0.5$, $\beta = 0.01$, and changing the number of Gibbs sampling iterations N. The initialization random number seed is changed for each of the 5 runs. For each run and N, AUC of 10-fold cross validation using L1-regularized logistic regression on the projected document is plotted in Figure 10. In both case, the perfomance plateaued after around 20 iterations and there is no considerable change up to 3000 iterations. This behavior was the same for both datasets; however, the random seed has a more considerable effect on the performance of the resulting model in the hard dataset.

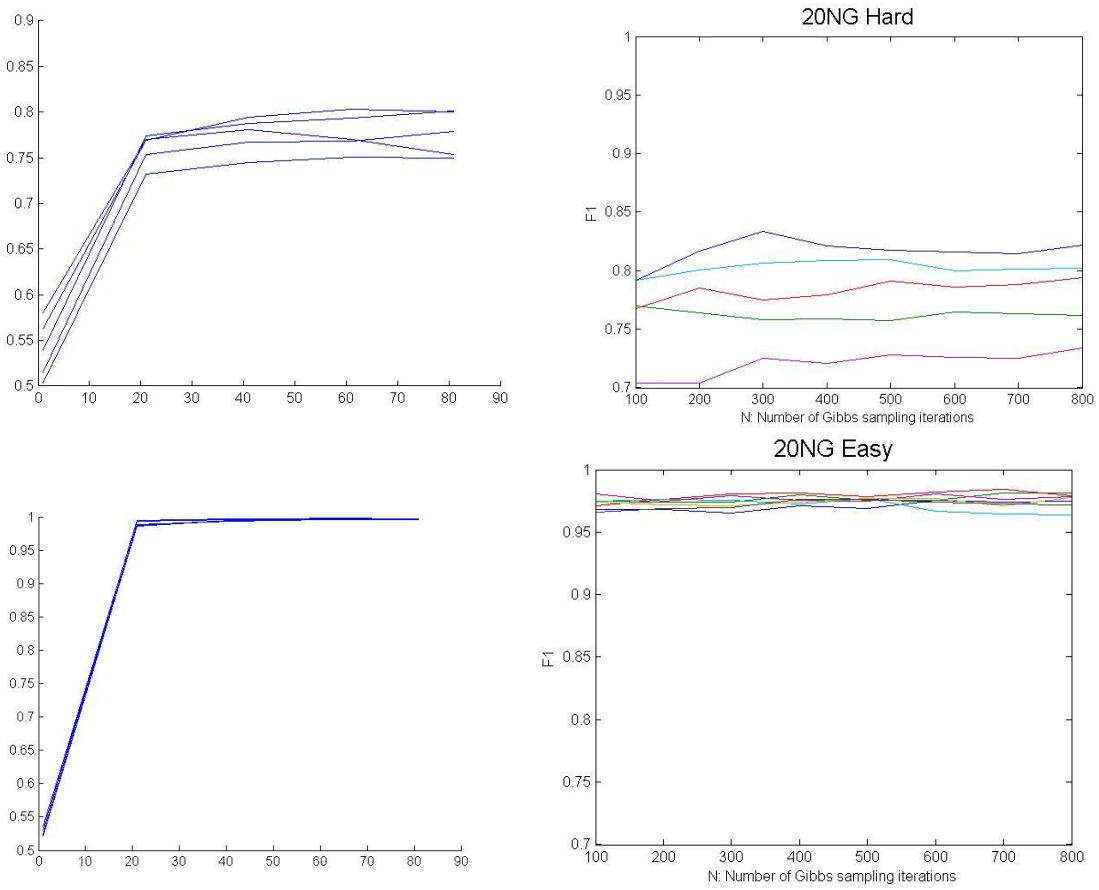


Figure 10. Effect of convergence of the topic model on the prediction performance. The left figures are the iterations below 100.

Next, we varied the values of the three parameters of LDA and the same convergence behavior was observed in all cases. We now report the impact of the three parameters on the predictive performance.

We fixed $\alpha = 0.5$, $\beta = 0.01$, and number of iterations $N = 300$, which ensured convergence based on log-liklihood approach. Results shown in Figure 11 suggest that the result is moderately sensitive to this parameter. The performance increases slower in the case of hard dataset in compare to the easy dataset, which is expected. The common practice in using topic models is to use large number of topics. This can usually be a good approach especially when the classifier is robust to redudant topics and it is better than risking with too few topics; however as it can be seen in the case of easy dataset, the result shows signs of degradation with too many

topics. There is also undesired time performance cost with too many topics. We will revisit this issue when optimizing the number of topics.

Lastly, we experimented with changing the priors α and β while keeping $T = 10$ and $N = 300$ and the result is shown in Figure 12. Intuitively, small values of Dirichlet parameter results in the distribution to generate *sparser* vectors. That is, small α for document-topics, results in fewer topics to be assigned to each document and small β for word-topics, results in fewer words to be assigned to each topic. This is an important effect because the topics are forced to specialize. The impact of the parameter change in the case of hard dataset is important but unfortunately looks random. The performance is more stable for the easy dataset and smaller value are better as expected. Effect of priors are also dependent on the number of topics T and also when they are defined as asymetric. Initial work on topic models did not consider the importance of prior. Some recent work have developed techniques for optimizing the prior [Wallach et al., 2009; Low et al., 2011]. We will revisit this subject when experimenting with the complete Algorithm 1.

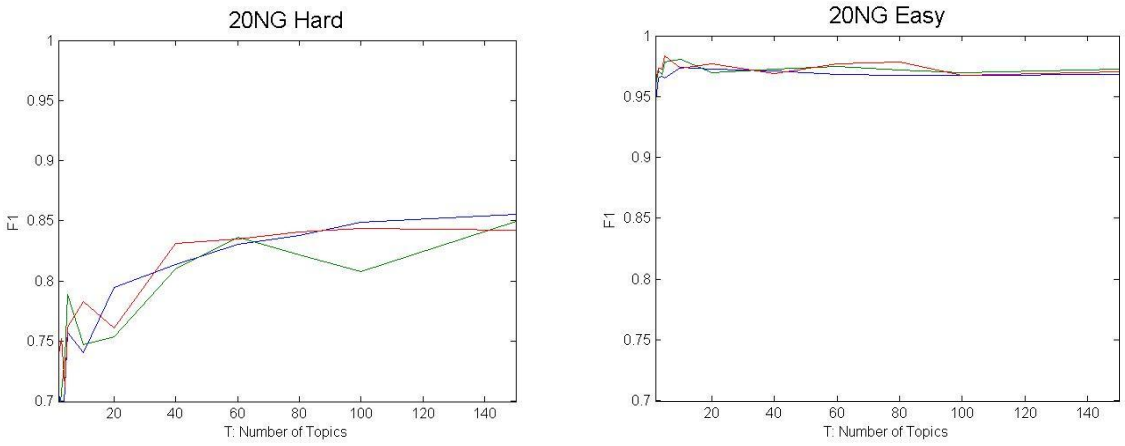


Figure 11. Effect of changing the number of topics ***T*** on prediction performance.

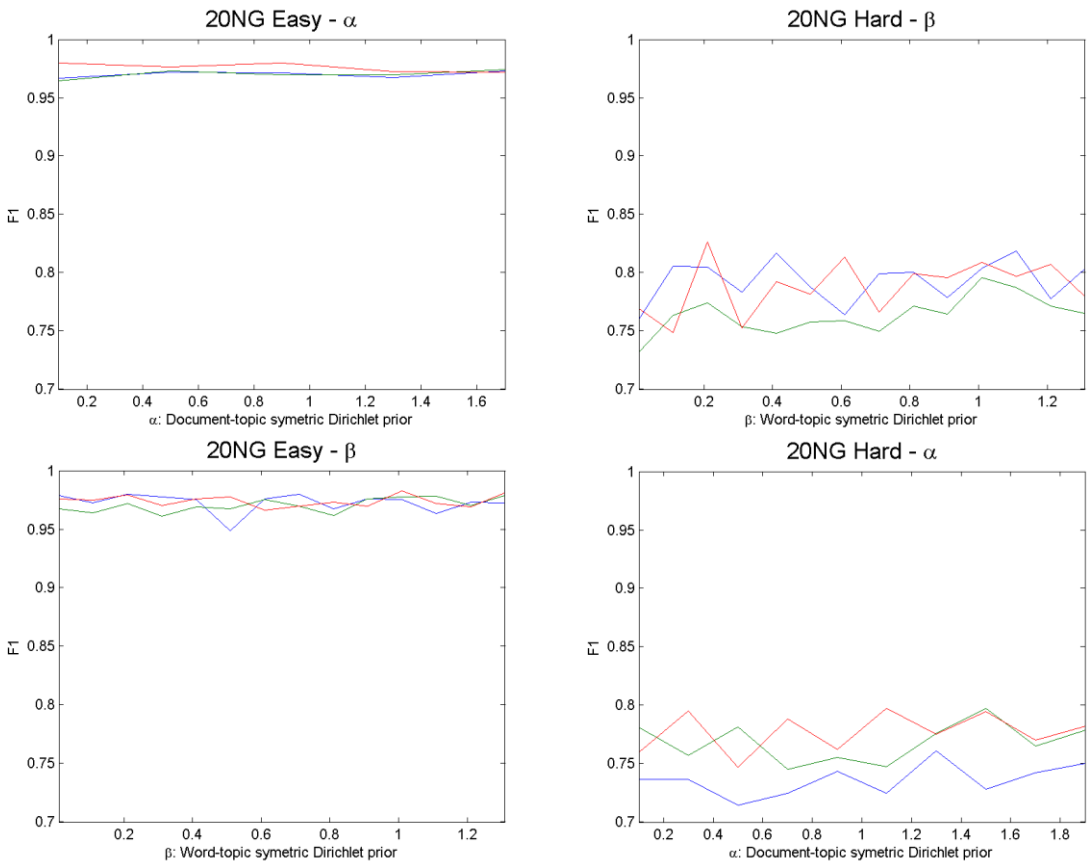


Figure 12. Effect of changing priors α and β on prediction performance**.**

4.3.2.2 Label Prediction Performance

We evaluate the performance of various approaches in predicting the labels in our datasets. Table 11 summarized the results of one-vs-all baseline approach on all datasets. The performance of both SVM and logistic regression are very similar and therefore going forward, we present the results from one of them. The performance on NYT is not surprisingly significantly lower due to the large number of labels. Table 12 shows more detailed about confusion that is caused for the classifier when we increase the number of the labels in two scenario: when we only limit to the documents containing the labels and when allow irrelevant document stay as the negative example. Figure 13 shows that number of instances available for a class has a strong correlation with the classifier performance. Inherent confusion between two classes, are less important in the large output space settings.

Dataset	# Labels	Classifier	Precision	Recall	Accuracy	F1	AUC
20NG Easy	2	SVM	0.968	0.985	0.977	0.977	0.997
		L2LR	0.961	0.981	0.971	0.971	0.996
20NG Hard	2	SVM	0.857	0.879	0.867	0.850	0.914
		L2LR	0.880	0.911	0.895	0.880	0.944
20NG All	20	L2LR	0.855	0.756	0.982	0.801	0.966
		SVM	0.812	0.760	0.980	0.784	0.918
Yahoo Art	26	L2LR	0.336	0.263	0.290	0.906	0.678
		SVM	0.316	0.277	0.292	0.905	0.635
Yahoo Business	30	L2LR	0.458	0.301	0.347	0.964	0.685
		SVM	0.334	0.311	0.306	0.962	0.665
Delicious	Top 20	L2LR	0.844	0.857	0.850	0.849	0.912
NYT	Top 675	L2LR	0.036	0.299	0.954	0.056	0.775

Table 11. Result of one-vs-all using linear SVM and L_2-regularized logistic regression on all dataset. For NYT, 675 labels had 10 or more documents, which is a requirement for 10-fold cross validation.

Experiment	# Docs	# Labels	Precision	Recall	Accuracy	F1	AUC
All NYT documents.	19,340	2	0.647	0.721	0.944	0.672	0.935
	19,340	10	0.418	0.650	0.932	0.502	0.900
	19,340	20	0.369	0.656	0.938	0.465	0.906
	19,340	50	0.261	0.612	0.942	0.356	0.898
Subsets of NYT documents containing at least one of the labels.	1,710	2	0.922	0.952	0.896	0.937	0.935
	7,881	10	0.816	0.764	0.950	0.788	0.900
	11,191	20	0.481	0.728	0.923	0.572	0.906
	15,257	50	0.328	0.640	0.944	0.425	0.898

Table 12. Effect of change in number of labels and the confusion of the classifier when documents from other labels are present. Results are using L_2-regularized Logistic Regression on the NYT dataset. Labels are selected from top-n most frequent ones.

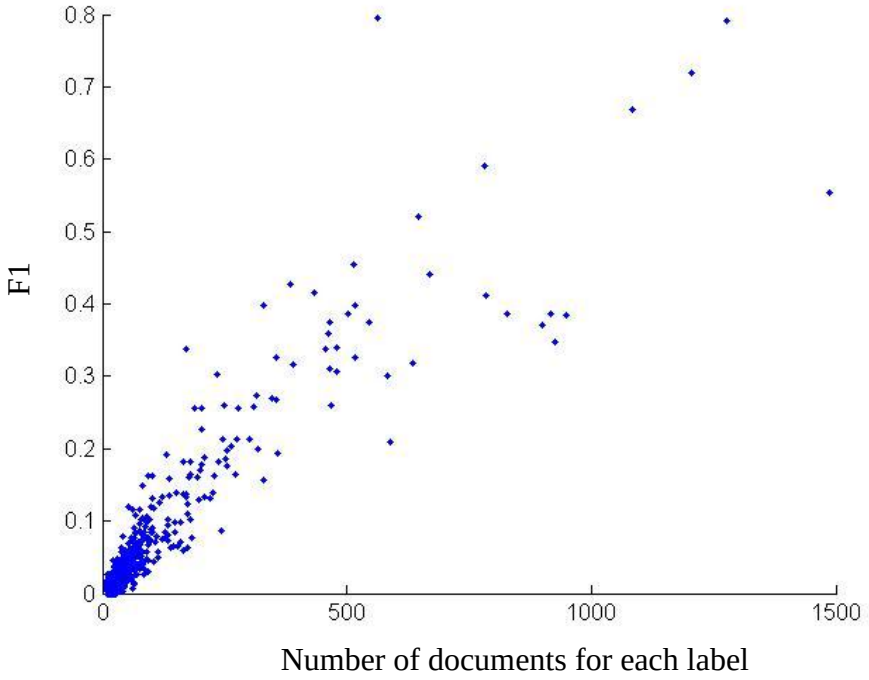


Figure 13. Correlation of F1-measure and the number of documents for each label. Each point is a the result of a one-vs-all L_2-regularized Logistic Regression on the NYT dataset. Pearson correlation coefficient: $\rho = 0.9188$, p-value<0.001.

Table 13 summarizes evaluation of LDA [Blei et al., 2003] and L-LDA [Ramage et al., 2009]. L-LDA restricts the topic assignments to documents based on their labels. Predictions using topic models can be done in two ways. The common approach, shown as LDA+SVM and L-LDA+SVM, is applying the classifier on the document topic proportions of the training and testing documents. Another approach for L-LDA, shown as "Direct", is using the fact that the topics and labels mapping is fixed ahead of time and therefore the test document-topic vector can be directly, thresholded for the prediction. We change the number of labels selecting from 10, 20 and 50 of the most frequent labels. The number of topics are chosen divisible by the number labels. First observation is that LDA with sufficient number of topics performs better than L-LDA and L-LDA performs better than LDA when topics per label is smaller. We present comparison of 10 labels case with 10-40 topics to compare the behaviour of LDA and L-LDA and when they are combine with the bag of word approach. We show the results in four cases in Figure 14 for 4 scenarios: when the 10 labels are the most frequent or infrequent, and when we limit the document to only those containing the selected labels or when all documents are

included. The performance of the LDA grows much faster than L-LDA when number of topics are increased. In all cases, the LDA addition to bag of words helps the performance though in some cases not significanrtly. Surprisingly, the addition of the L-LDA always hurts the performance even when the L-LDA performs individually better than LDA. This observation suggests that the topic constraints can create noisy features.

# Labels	# Topics	Topics per Label	LDA +SVM	L-LDA +SVM	L-LDA Direct
10	10	1	0.492	0.497	0.245
	20	2	0.660	0.511	0.177
	50	5	0.695	0.551	0.144
	100	10	0.694	0.554	0.114
20	20	1	0.333	0.389	0.151
	40	2	0.391	0.409	0.112
	100	5	0.432	0.386	0.087
50	50	1	0.249	0.273	0.096
	100	2	0.316	0.267	0.072

Table 13. Macro F1 of topic model methods for NYT dataset. Labels are top-n most frequent in the dataset. Average of 5 runs of 10-fold cross validation. Each run used a different random initial topic model seed but it was shared between models and different label-topic combinations to eliminate the effect of initialization condition.

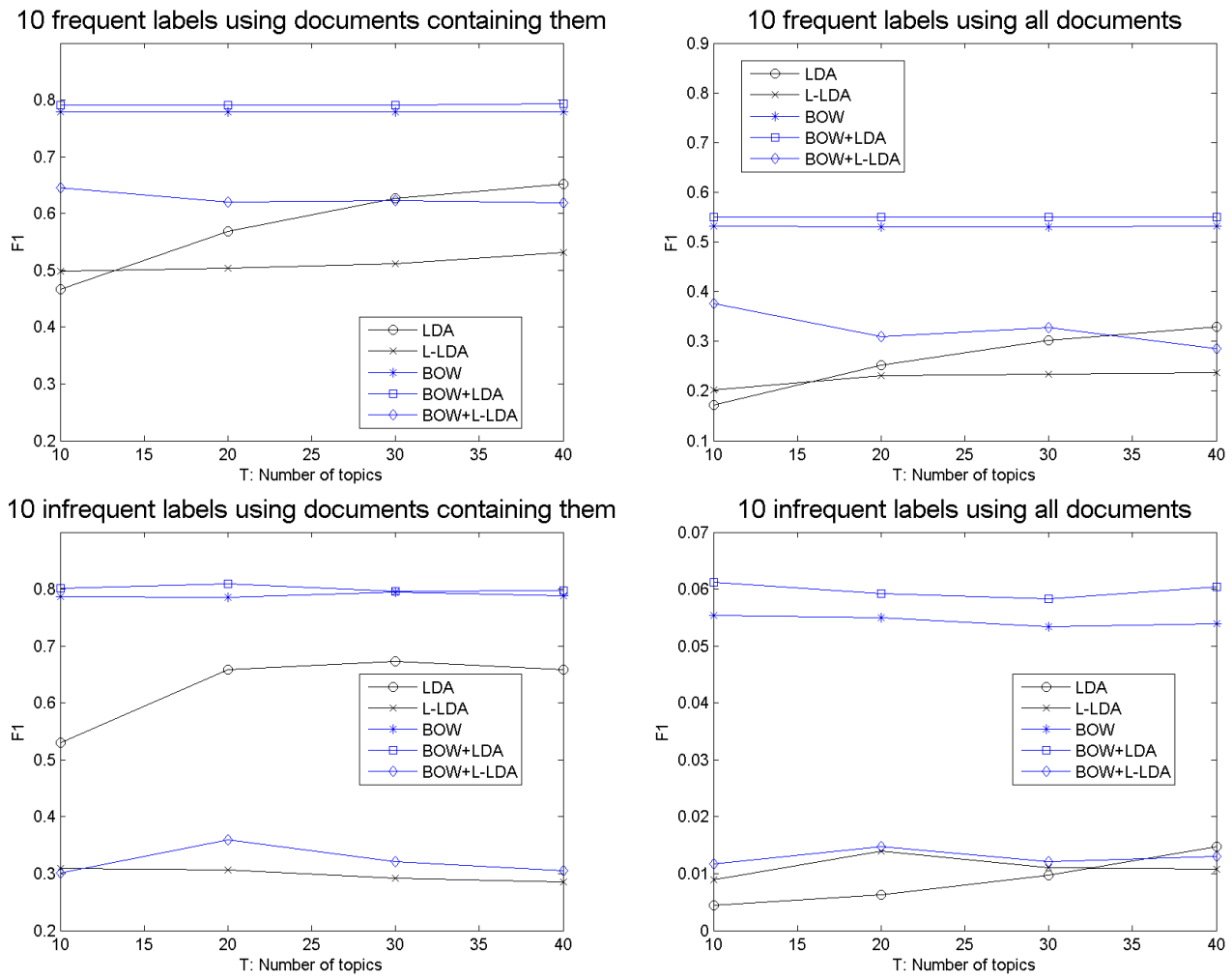


Figure 14. Comparison LDA and L-LDA and their combination with the Bag of words (BOW) model on 10 most frequent and 10 infrequent labels in around 30 documents.

Next, we evaluate the effect of number of labeled document on the performance of the two models. We run a 10-topic LDA and L-LDA for 5 rounds of 10-fold cross validation keeping the initialization seed constant. The data is documents containing top 10 most frequent labels of the NYT dataset. We fixed split in each round and kept one fold for testing, and added other folds one at a time and measured the performance. The result in Figure 15 suggests the addition of the labels provide a significant advantage over the LDA but then the performance degrades with additional labeled documents. One explanation of this behaviour is that additional labeled documents impose further constraints on the topics and increase the model's bias, therefore its flexibility to assign topics based on the document contents is reduced.

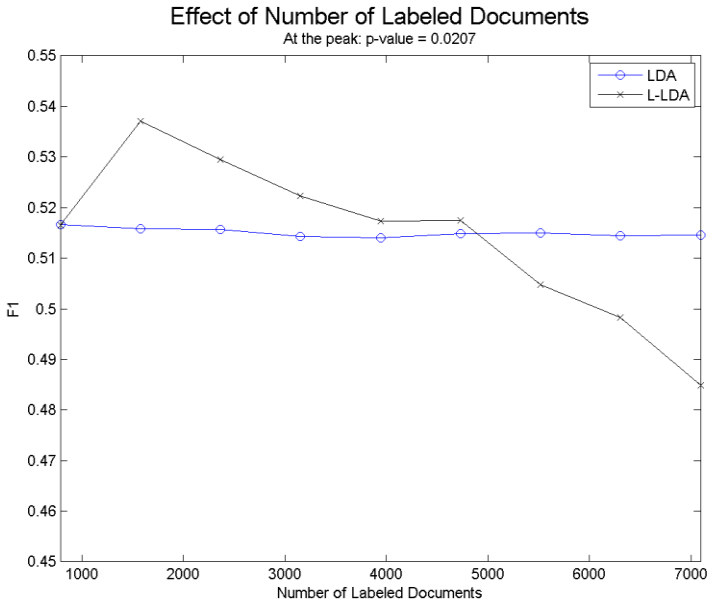


Figure 15. Effect of changing the number of labeled documents in the performance of LDA and L-LDA on the top 10 more frequent labels in NYT dataset.

4.3.2.3 Topic Specificity Score

We introduced topic specificity score (TSS) in Section 4.3.1.1 to measure the quality of the topics based on how well they are separated from each other. We also explained how this score can be used to dynamically select the discriminative words as an alternative to the hand-crafted stopword lists. We now evaluate the performance of this score. The stop word list that we used has 319 words, which are common English words such as pronouns, articles and conjunctions. We perform 10 iteration of incrementally applying the word constraints. The model is LDA and we used 20NG Easy and Hard datasets and word constraints based on the Algorithm 7, only using the TSS part and not the coherence part. The threshold θ_{TSS} is chosen to eliminate 30% of lowest scoring words that are not are not eliminated until a given iteration. Figure 16 shows the result in two scenarios: starting from all words and from all words that are not stopwords based on our list. Iterations have a more drastic effect on the hard dataset and generally improving, although not with a clear pattern. On the easy dataset, performance is consistently dropping but the range of variations is not statistically significant.

Iteration	# Words	w	P(w|z=1)	w	P(w|z=2)	w	P(w|z=3)	w	P(w|z=4)
1	15924	the	0.00125	the	0.00224	the	0.00219	the	0.00074
		and	0.00077	to	0.00106	to	0.00103	to	0.00048
		to	0.00073	and	0.00085	in	0.00073	of	0.00043
		it	0.00067	is	0.00084	and	0.00072	and	0.00031
		in	0.00059	it	0.00068	of	0.00071	in	0.00030
		that	0.00054	with	0.00057	you	0.00071	or	0.00020
		of	0.00048	that	0.00050	it	0.00065	for	0.00019
		for	0.00048	on	0.00049	on	0.00061	are	0.00017
		is	0.00047	of	0.00047	is	0.00051	this	0.00015
		you	0.00044	have	0.00047	that	0.00045	be	0.00015
4	6216	the	0.00298	the	0.00096	the	0.00208	and	0.00033
		to	0.00154	to	0.00063	to	0.00109	edu	0.00033
		and	0.00108	you	0.00052	and	0.00075	is	0.00030
		it	0.00094	and	0.00049	is	0.00073	com	0.00024
		that	0.00087	on	0.00041	with	0.00059	it	0.00023
		you	0.00065	bike	0.00037	it	0.00056	any	0.00020
		is	0.00058	com	0.00036	that	0.00051	on	0.00019
		have	0.00054	is	0.00035	drive	0.00048	me	0.00017
		my	0.00052	it	0.00033	have	0.00046	be	0.00017
		on	0.00048	edu	0.00031	scsi	0.00041	thanks	0.00016
7	2493	you	0.00083	com	0.00044	drive	0.00047	card	0.00020
		was	0.00040	writes	0.00030	scsi	0.00042	dx	0.00020
		my	0.00040	dod	0.00027	you	0.00030	my	0.00018
		they	0.00030	my	0.00026	ide	0.00025	if	0.00018
		when	0.00027	article	0.00024	mb	0.00024	windows	0.00017
		if	0.00027	bike	0.00022	controller	0.00023	com	0.00016
		up	0.00026	you	0.00020	disk	0.00022	mhz	0.00015
		writes	0.00024	apr	0.00018	my	0.00021	system	0.00015
		we	0.00024	was	0.00016	drives	0.00021	thanks	0.00015
		he	0.00024	they	0.00015	hard	0.00020	bus	0.00014
10	953	dod	0.00023	bike	0.00033	drive	0.00049	dx	0.00020
		he	0.00021	dod	0.00018	scsi	0.00042	mhz	0.00015
		we	0.00016	ride	0.00017	ide	0.00025	local	0.00014
		his	0.00015	bikes	0.00014	controller	0.00023	motherboard	0.00013
		dog	0.00015	down	0.00014	mb	0.00023	modem	0.00012
		go	0.00014	we	0.00013	disk	0.00022	board	0.00012
		ed	0.00013	helmet	0.00013	hard	0.00021	memory	0.00012
		rider	0.00012	road	0.00012	drives	0.00021	vlb	0.00012
		left	0.00011	riding	0.00011	data	0.00015	chip	0.00012
		bike	0.00011	front	0.00011	bios	0.00014	ram	0.00011

Table 14. Result of applying the word constraints iteratively on 20NG Easy. The words relevant to the two categories of are given higher probabilities within the topics after several iterations and less discriminative words are no longer appearing.

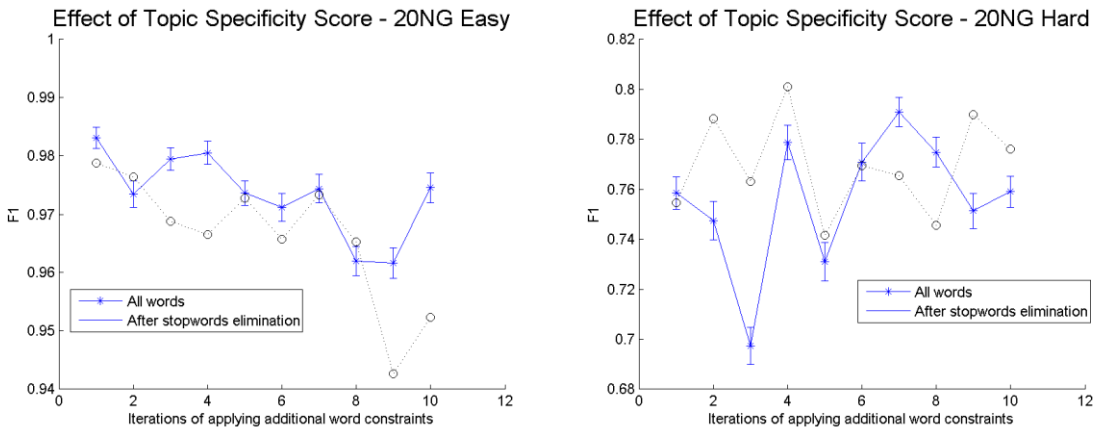


Figure 16. The effect of Topic Specificity Score used in iteration to narrow down the features for topic models.

4.3.2.4 Topic Labeling

We evaluate the approaches introduced in Section 4.3.1 for labeling the topics based on how well a given labeling performs in classification of the projected documents. We use 20NG Easy and Hard datasets, which represent the extremes in the separation situations. Since the number of topics has a major impact on the performance, we show the results by varying this parameter. We used the LDA as the topic detection and compare four discriminative approaches: random, which selects documents randomly; uncertainty, which selects documents that a classifier trained with all labeled data up to a given point has the least confidence; entropy, which selects document with the lowest entropy of $P(z|d)$; and TSS, which selects the documents with the highest topic specificity score (see Section 4.3.1.1).

Results are plotted in Figure 17. In all cases, the random selection approach performs reasonably well. For easy separation case, entropy-based approach performs better than the uncertainty and this pattern is reversed in the case of hard separations. The results show using fewer topics can help the model reach its peak performance with fewer labeled document, therefore using more topics than needed is not a good strategy when supervision is added, as was the case in the unsupervised case.

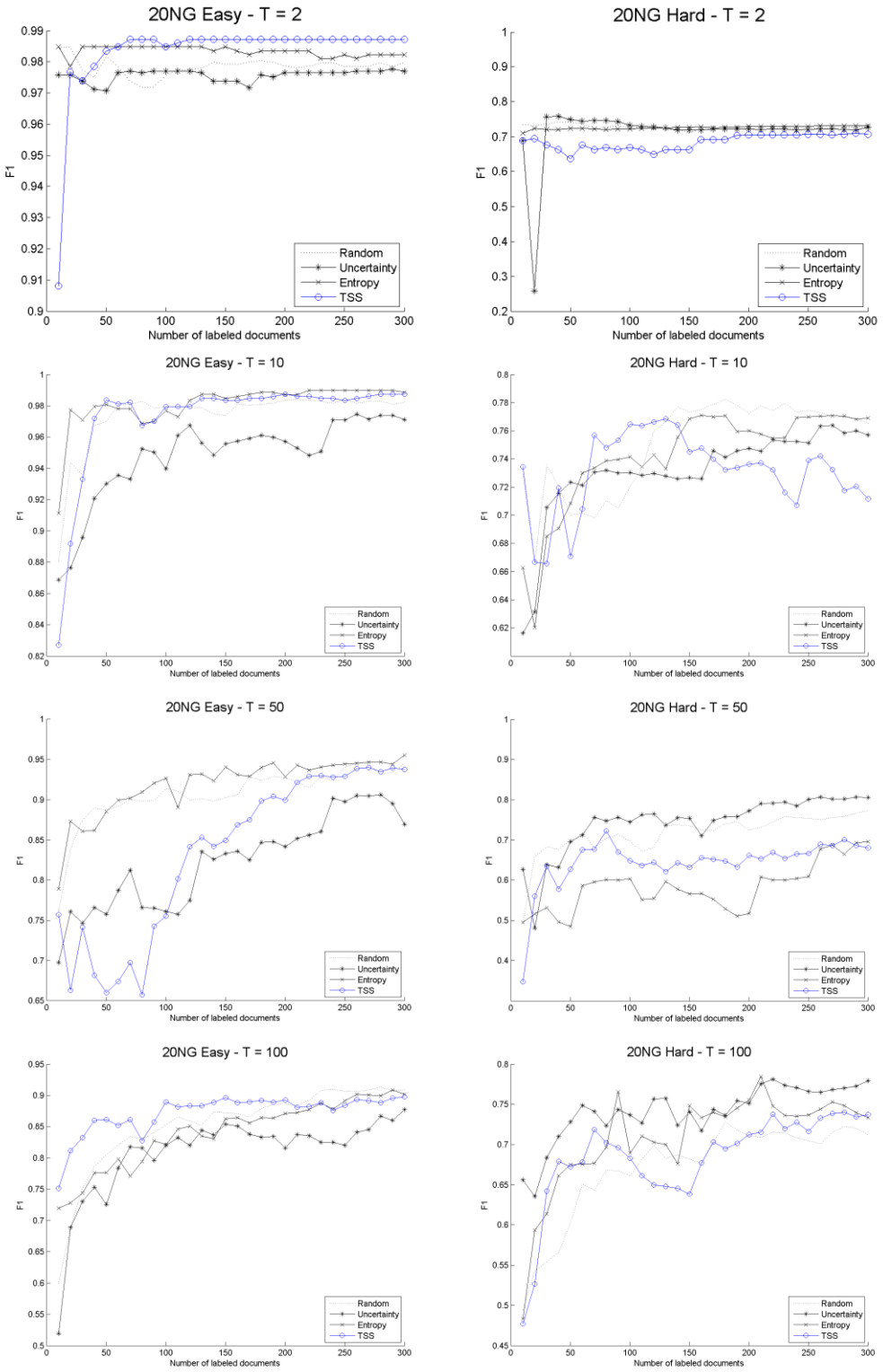


Figure 17. Performance of the discriminative approaches in topic labeling. The left column is for the 20NG Easy and right side for 20NG Hard. The number of topics are 2, 10, 50 and 100 from the top to bottom.

4.3.2.5 Dimensionality Reduction Approaches

We have evaluated several dimensionality reduction methods for their discrimination power in binary classification task. We focus on unsupervised methods and discuss their strengths and weaknesses. Additional unsupervised and supervised dimensionality reduction methods, especially those considering the large label spaces should be investigated in future work. We compared the following methods:

1. *Principal Component Analysis (PCA):* A common dimensionality reduction method based on projecting the documents to the space spanned by eigenvectors of the document-term matrix, which captures the variance in the data. PCA is often an effective method. Projection is performed using Eigenvector Decomposition or alternatively, Singular Value Decomposition (SVD). SVD is a deterministic approach with a unique solution. Its time complexity for a $|\mathcal{D}|$ by $|\mathcal{F}|$ matrix is $O(\min(|\mathcal{F}|^2.|\mathcal{D}|, |\mathcal{F}|.|\mathcal{D}|^2)$, which can be expensive for many practical applications. Another issue with PCA is its sensitivity to the outliers in the data. These shortcomings and other issues have been addressed in many extensions of this approach [Van Der Maaten et al., 2007; Xu et al., 2010].
2. *k*-means: A common clustering method based on iteratively assigning data points to the nearest *k* centroids with the objective function of minimizing the total of some distance metric such as Euclidean L_2 and cosine. The projected document vector is a real-valued vector of distances to each of the *k* centroids.
3. *Locality Sensitive Hashing (LSH)* [Indyk and Motwani, 1998; Charikar, 2002]: We introduced hashing method in Section 2.3. LSH is based on random projections theory, which has shown that for high dimensional spaces, random vectors are approximately mutually orthogonal and are suitable as the basis for projections that preserve the proximity in the original space. In one implementation of LSH that preserves the cosine distance [Charikar, 2002], *k* random vectors in the original space is generated from the standard Gaussian distribution, each representing a unique hyperplane to which they are orthogonal. The projected documents are the binary code vectors of length k and each bit is set 0 or 1 based on which side of the corresponding hyperplane the original document resides. Equivalently, we take the sign of the inner product of the random vectors with the document vectors. The procedure is computationally efficient; however, its

effectiveness is conditioned on the distribution of the data which is ignored in generation of the projection.

4. *Spectral Hashing (SH)* [Weiss et al., 2009]: an example of semantic hashing approaches, which we introduced in Section 2.3. It uses eigenvectors of the data, similar to PCA, and improves the projection as opposed to the data independent random projections used in LSH.
5. *Probabilistic Latent Semantic Analysis (PLSA)* [Hofmann, 1999]: a probabilistic version of LSA (Section 2.2.2) which assigns words in documents to k topics. The projected documents are the proportion of each topic within the document, an estimate to $P(\boldsymbol{z}|d)$, which is computed using *tempered* EM to avoid overfitting [Hofmann, 1999].
6. *Locally consistent Topic Model (LTM)* [Cai et al., 2009]: an example of approaches that improve PLSA. It adds a manifold regularization term to the objective function of the PLSA. Regularization penalizes high KL-divergence of the projected documents that are similar in the original space. The step of finding the pair-wise document similarities $O(|\mathcal{D}|^2)$ is the additional computation compared with PLSA.
7. *Latent Dirichlet Allocation (LDA)* [Blei et al., 2003]: another improvement to the PLSA by adding a Dirichlet prior to the document-topic mixtures (Section 2.2).

We performed a supervised evaluation of the clustering performance of the approaches. First, the best assignment of the labels to the k clusters is found using the Hungarian method [Lovász and Plummer, 1986], and then we compute two measures using n assigned labels $\hat{L}$ and gold labels L. I is the mutual information and H is the entropy.

1. *Accuracy (AC):* Accuracy, also known as purity, is the ratio of all correctly assigned labels.

$$AC(L, \hat{L}) = \frac{|L \cap \hat{L}|}{|L|}$$

2. *Normalized Mutual Information (NMI):* Accuracy does not consider the number of labels and clusters. NMI is an information theoretic measure of the dependence between two variables, which is symmetric and normalized to [0,1]. The difference between assignments is measured in terms of amount of information, that is, how much we can reduce the entropy of $\hat{L}$ after we learn about L.

$$NMI(L,\hat{L}) = \frac{I(\hat{L};L)}{\frac{1}{2}\left(H(L)+H(\hat{L})\right)}$$

Table 15 summarizes the performance of the mentioned algorithms on the 20NG Easy and Hard datasets. The best performance was obtained from LDA with number of topics equal to the number of classes. As we have shown in Section 4.3.2.1, the performance of LDA can be sensitive on its input parameters and in this case the performance with 50 topics is considerably worse due to the selection of the priors, which we did not optimize ($\alpha = 0.5, \beta = 0.01$). Note that we are evaluating the dimensionality reduction algorithms directly and not using the classifiers. Classifiers are usually robust to lower quality topics and we have used them for the evaluations in the remaining sections. We also had observed the performance advantage of the LDA in other tasks, such as disambiguation of noun phrases in the NELL knowledge-base [Mitchell et al., 2009], and we decided to develop our concept extraction algorithm based on topic models.

Approach	**# Dimensions**	**20NG Easy**		**20NG Hard**	
		AC	**NMI**	**AC**	**NMI**
None	-	0.830	0.422	0.566	0.002
PCA	2	0.709	0.183	0.580	0.010
	50	0.939	0.682	0.507	0.018
k-means	256	0.517	0.000	0.537	0.004
LSH	50	0.699	0.117	0.513	0.000
	256	0.808	0.296	0.540	0.002
SH	50	0.896	0.573	0.500	0.012
	256	0.932	0.665	0.529	0.000
PLSA	2	0.506	0.001	0.561	0.000
	50	0.738	0.188	0.505	0.000
LTM	2	0.919	0.648	0.634	0.048
	50	0.968	0.799	0.599	0.030
LDA	2	**0.983**	**0.877**	**0.732**	**0.163**
	50	0.568	0.050	0.521	0.025

Table 15. Comparison of the dimensionality reduction methods.

4.3.2.6 Adaptive LDA

We evaluate the Adaptive LDA model introduced in Section 4.3.1.2. We start by testing whether unequal number of topics per class can help the performance. We assign one topic per class and then assign the remaining topics to one class. Result is shown in Table 16. For certain classes

such as 4 and 9, the increase consistently reduces the performance; however, for classes 2 and 5, the performance increases initially. The drop in performance for 40 topics may be a result of the skewed allocation scheme of assigning too many topics to one class, which can prevent other classes to be modeled properly.

We evaluate three heuristic approaches to determine uneven allocation of topics to classes. First approach is to simply distribute the topics proportional to the number of document per labels. The other two approaches are based on the concept graph (Section 4.2.3), where the nodes correspond to the labels and we defined two methods for calculating the edge weights: classifier confusion and co-occurrence. We average the values of edges connected to every node, and then distribute topics proportional to node values. Table 17 shows the comparison for the three described approaches of assigning topics to classes. The performance of these three approaches and their inverse is compared in Table 18. We selected the best performing strategy, which was assigning more topics to the least confusing classes as determined by the classifier performance.

# Topics	**20**	**30**	**40**
L-LDA	0.492	0.509	0.537
Assign one topic to every label and the rest of topics to class shown in the first column.			
1	0.498	0.534	0.525
2	0.528	0.550	0.534
3	0.506	0.431	0.437
4	0.496	0.419	0.339
5	0.538	0.557	0.520
6	0.523	0.525	0.522
7	0.535	0.521	0.513
8	0.523	0.549	0.490
9	0.488	0.334	0.251
10	0.529	0.497	0.408

Table 16. Effect of changing the number of topics per class for AdaptiveLDA. The dataset is NYT and most frequent 10 labels are used. For each row, all but one topic is assigned to the shown class and then the model F1 is evaluated using 10-fold cross validation.

Class	F1	# Topics	Label Doc Freq	# Topics	Avg npmi	# Topics
1	0.81	3	1487	5	0.10	3
2	0.83	3	1277	4	0.10	3
3	0.95	4	1206	4	0.05	2
4	0.95	4	1083	3	0.10	3
5	0.69	3	949	3	0.10	3
6	0.57	2	927	3	0.11	3
7	0.67	2	917	2	0.10	3
8	0.70	3	899	2	0.10	3
9	0.76	3	829	2	0.09	3
10	0.74	3	784	2	0.10	4
Total	-	30	-	30	-	30

Table 17. Various strategies to distribute 30 topics among 10 topics using AdaptiveLDA.

Method	F1
F1	**0.535**
F1 Inv	0.502
Label Freq	0.529
Label Freq Inv	0.509
Avg npmi	0.511
Avg npmi Inv	0.509
Equal	0.509

Table 18. Comparison of various topic allocation strategies for Adaptive LDA with T=30. “Inv” refers to inversing the strategy. For example, F1 approach assigns more topics to classes with higher F1, and F1 Inv assigns more topics to classes with lower F1.

Finally, we compare the performance of Adaptive LDA (A-LDA) with other models in Table 19. When we have the same number of topics per class A-LDA is equivalent to L-LDA. Similar to the case when we compared L-LDA with LDA, the advantage of the A-LDA over L-LDA can be seen when the number of labels are large.

Models	# Labels # Topics	10 10	10 30	10 200	150 300
LDA		0.520	0.617	0.719	0.029
L-LDA		0.497	0.508	0.552	0.086
A-LDA		0.497	0.512	0.551	**0.095**

Table 19. Comparison of Adaptive LDA with other methods. Strategy used for A-LDA is F1. A-LDA performance in the last columns over L-LDA is significant at $p < 0.0029$.

4.3.2.7 Parameter Updates and Convergence

Convergence measures that are evaluated in the Step 5 to verify convergence are the following:

1. *Task-based:* In the 10-fold cross validation, we used 8 folds as train set, one fold as test set and last fold as development set on which we measure the task performance.
2. *Topic quality-based:* We consider the following specific measure in each group:
    a. Topic specificity:
        i. Sum of the topic specificity score for all words: $\sum_w TSS_2(w)$
        ii. Sum of the pairwise distances for all pairs of word-topic distributions: $\sum_{t_1} \sum_{t_2} \text{CALC-DISTANCE}(P(w|t_1), P(w|t_2))$
        iii. Label all documents by most likely topic $argmax_t P(t|d)$, then the measure is the 10-fold cross validation of F1 of linear SVM for these labels.
    b. Topic coherence: average of the shifted and normalize point-wise mutual information (Section 4.2.3) of top 10 words of the topic.

We perform parameter updates that affect the value of these convergence measures. We stop the loop when the calculated value falls below a threshold. We now provide an evaluation of the parameter optimization options of the Step 3 for each of the three parameters of the Adaptive LDA (Section 4.3.1).

1. *Updating* α: Setting the document-topic priors is equivalent to setting topic-label restrictions; however, since the initialization of the model still allows all topics to participate for all labels, it is possible that in the posterior some of the topics allocated to the document do not follow the restriction. As shown in Table 20, there is a substantial improvement in performance, when follow this approach. Further investigation is needed for understanding the cause of this performance improvement.

# Labels	# Topics	LDA	L-LDA	LDA+α
10	10	0.386	0.430	**0.599**
	20	0.544	0.452	**0.629**
	30	0.567	0.460	**0.643**
	100	**0.636**	0.515	0.628
50	100	0.265	0.283	**0.324**

Table 20. F1 scores on the NYT dataset for updating α.

2. *Updating* β: Table 21 shows the results for two approach of setting the β by matching them to the word label distributions and to the Topic Specificity Score (TSS). TSS improves the result from LDA in one case but hurts the performance of the L-LDA consistently. This result can possibly by explained by the effect of topic-label restrictions on the quality of the TSS score. The word-label distribution helps the performance of the L-LDA.

# Labels	# Topics	Approach	LDA	LDA+β	L-LDA	L-LDA+β
10	10	$P(w\|l)$	0.386	0.359	0.430	**0.528**
		TSS_2		0.464		0.418
	20	$P(w\|l)$	0.544	0.473	0.452	**0.465**
		TSS_2		0.445		0.437
	30	$P(w\|l)$	0.567	0.462	0.460	**0.484**
		TSS_2		0.381		0.445
	100	$P(w\|l)$	0.636	0.697	0.515	**0.530**
		TSS_2		0.332		0.432
50	100	$P(w\|l)$	0.346	0.224	0.294	**0.324**
		TSS_2		0.108		0.199

Table 21. F1 scores on the NYT dataset for strategies of updating β. All differences from LDA to LDA+β and from L-LDA to L-LDA+β are statistically significant at $p<0.0001$.

3. *Updating* T: We evaluated two approaches for updating number of topics and the results are shown in Figure 18. First approach (top of the Figure 18) is doing one split for the least coherent topic and one merge between the two topics with the least symmetric KL-divergence distance. The number of topics remains 10 for all 10 iterations and the performance improves. Note that the held-out perplexity of the model is not a direct measure of the prediction performance of the resulting model.

   The second strategy is to increase the number of topics by one in each iteration. We compare the approach of adding a randomly initialized topic to the approach of splitting

the least coherent topic. The results at the bottom of Figure 18 show that using coherence for splitting is almost always better than adding the topic to the model; however, as the number of topics increase the benefit of this approach diminishes.

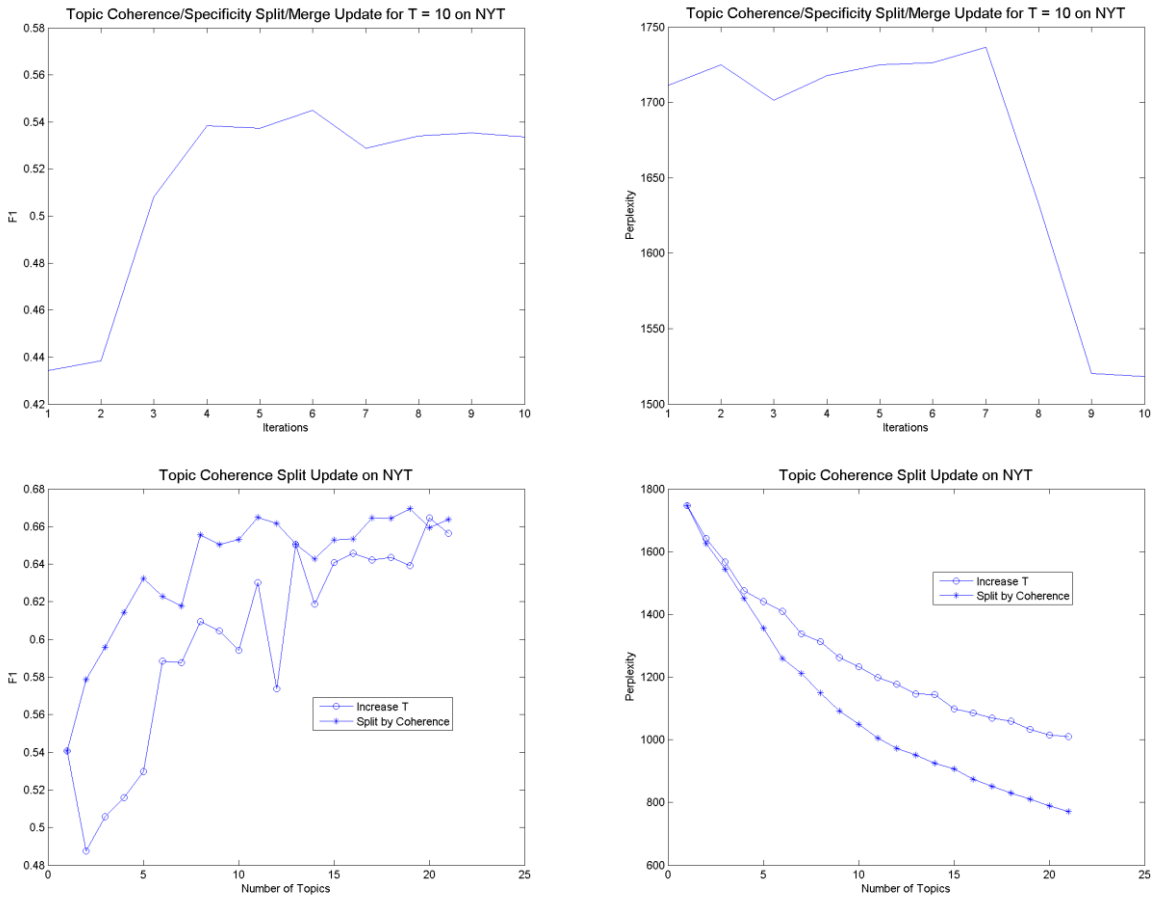


Figure 18. Evaluation of two approaches for updating the number of topics T. The top two figures are F1 and perplexity of the Coherence/Specificity strategy and the bottom two figures are for increasing the number of topics one-by-one and using the coherence split strategy.

4.3.3 Conclusion

We introduced a framework for the language interpretation task that iteratively applies topic models to the text data. We provided the experimental evaluation for different steps of the algorithm. There are many other possible choices for each steps of the main algorithm (Algorithm 1). The main advantage of the proposed framework is its flexibility in using labels and other available data for optimizing the parameters of the concept extraction approaches.

The proposed algorithms are general and can be applied to many tasks where the text data is available and when we expected that understanding the text data can be beneficial in improving the performance of a task. In some cases, it is difficult to formula the task and its performance improvement in this framework. One example is in the domain of product user reviews. The buyers are willing to optimize their satisfaction as the result of the purchase based on information provided by other people whose experiences and expectations can widely vary. The product buying decision is often multifaceted and subjective. We discuss in Section 6.2 how some of these issues can be addresses by additional guidance in collection concept annotations.

In Chapter 5, we apply this framework to the problem of detecting malicious websites and show an improvement in detection rate in compare with what was achieved in Chapter 3. We also discuss some of the practical challenges that we faced, such as collecting the initial ground truth data.

Lastly, we emphasize again that this form of shallow language understanding is appropriate only when limited knowledge about the concepts is sufficient for improving the task. It is not an appropriate choice when detailed information about entities and their interactions is necessary to make an inference about what the text data conveys.

Chapter 5

Improving Scam Detection Based on User Comments

In Chapter 4, we have introduced a general-purpose framework for language interpretation, which enables us to extract task-related concepts from natural-language text. We now revisit our main application area of detecting Internet scam. We have created additional datasets and have developed algorithms that utilize the user comments to address some of the discussed issues.

5.1 Motivation

Recall the motivating example from Chapter 1 (page 1), where the user is interested in purchasing a product from an obscure website. We are interested to use all possible signals to assist the user in making this trust decision. In Chapter 3, we stated by using the automatically collected reputation information and then apply machine learning to decide whether specific websites can be trusted. These reputation features are information that we collect from online sources such as blacklists that contain potentially malicious website usually reported by a community of user, and web metric companies that measure statistics such as monthly user traffic for websites. We showed that we can achieve high detection accuracy and that these features are relatively easy to collect but hard to manipulate by scammer; both desirable property for a scam detection system. We then discussed the limitation of this approach in Section 3.2.5 which is mainly sensitivity to the quality of the features and delays in updates for new or changing website. Achieving high accuracy in certain settings can be expensive, when the reliability and availability of the features are limited, which especially is unfortunately the case for obscure websites that they are needed the most.

We now consider another source of information provided by the web users in the form of either direct opinion about the website or comments describing the contents or behaviors of websites. This form of information has the same availability pattern as the reputation information: we have more comments for the more popular websites than for less popular websites; however, there are several differences between the reputation features and comments provided by the users.

1. When certain reputation features are unavailable or unreliable, it is unclear what can be done to alleviate the problem. The comments can be obtained more readily from the web users.
2. Textual comments are more information-rich than individual feature due to the power of natural language. They can represent various aspects of the malicious activities and broad range of scams. A fundamental problem with feature engineering for detecting malicious websites is that the attackers become more experienced at deceiving the machine learning algorithm and then we need to create new features, whereas natural language is more robust in representing the relevant information for the emerging attacks.
3. Some reputation features have a time lag for updates. During this time period the approach produces more false positives than expected [Shen et al., 2006]. The features are designed so that the reputation is built over time and they do not react fast enough. For example, a change in ownership of a website can stop or start its malicious behavior; however, search engines and blacklist do not immediately update the information. Textual comments are time-stamped and the analysis can take into consideration the recency.

We describe our approach to use the textual user comments along with the reputation features in this chapter.

5.2 Website User Comments

User comments have been abundant in the past decade and many web users frequently consult them to make better decisions. Prior work on automatic processing of user comments is related to movie reviews [Pang et al., 2002], product reviews [Popescu and Etzioni, 2005], service reviews such as restaurants [Sharifi, 2009], and hotels [Titov and McDonald, 2008a]. The target tasks have been sentiment analysis and detection of the product aspects. We are not aware of any prior work in using the user comments for the detection of malicious websites.

The user comments often contain detailed information about the contents and the behaviors of the websites, especially if the comments are provided by the security experts with specific objective of help other user browse the web safer. Use of such high quality comments is the basis of advanced safe-browsing tools such as SmartNotes [Sharifi et al., 2011b].

We have considered several sources for website comments. An ideal source is safe-browsing tools such as SmartNotes, because the users provide specific comments related to the malicious behavior of websites. Unfortunately, our implementation has been in use for only a short period of time and there is currently only a small number of users and comments. Many other safe-browsing tools, such as Google Safe-browsing, do not consider user comments. We observed that `twitter.com` contains real-time comments about websites. Other researchers have shown the use of Tweets for automatic detection of trends such as flu and other health issues [Paul and Dredze, 2011]. In our experience, most website comments on Twitter were not relevant to our task. They are mostly company news and information when the service is down and similar issues. Several web metric companies such as `alexa.com`, collect reviews about websites but those comments are also mostly unrelated to our safe-browsing goal. We have used website comments from the Web of Trust (WOT) community at `mywot.com`. WOT provides a platform for web users to share security related comments about websites. It contains over 16 million comments provided by millions of users world-wide since 2006. An example review is shown in Figure 19. Their safe-browsing solution, which is a browser add-on, also collects comments; however, they do not use the comments in their scoring system.

> **Positive:** "Amazing website, very informative, gives you lots of information. The site has lots of free software available for download." *Category label:* "Good site"
>
> **Negative:** "Website is fine, but they allow for popups that redirect to malware sites." *Category label:* "Annoying ads or popups"

Figure 19. Example of textual user comments about websites, collected from `mywot.com`. Each comment has a category label selected by its author.

We introduced our web scam dataset in Section 3.3.2, which contains 837 websites labeled as scam/non-scam, each with 42 features extracted from 11 online sources. We have collected 18,992 comments from WOT for the websites in our scam dataset. Popular websites have more comments and they usually are redundant. Including all comments would have biased our dataset toward more popular websites and therefore we limited the number of the comments per website to 200. Table 23 shows the breakdown of the comment counts. Nearly one-third of websites in

our scam dataset did not have any comments at the time of the data collection. Each comment has a category label specified by the user. We show the complete list of these labels and their number of comments in Table 22. Some of these labels are high level, such as "Ethical issues" and some are mixing different behaviors, such as pop ups and ads. Note that the user can only select one label per comment from this predefined list.

Category Label	# Comments	Group
Good site	4,475	Good
Useful, informative	4,030	Good
Phishing or other scams	1,712	Bad
Good customer experience	1,525	Good
Spam	1,247	Bad
Entertaining	881	Good
Adult content	853	Bad
Malicious content, viruses	821	Bad
Other	783	Neutral
Bad customer experience	735	Bad
Annoying ads or popups	437	Bad
Ethical issues	396	Bad
Spyware or adware	386	Bad
Useless	205	Neutral
Hateful or questionable content	201	Bad
Child friendly	198	Good
Browser exploit	107	Bad
Total	18,992	

Table 22. Category labels provided by the users for the comments on the WOT website. "Group" column shows the label used whenever we use the comment in the binary classification task, where *Bad* comments are positive, and *Good* and *Neutral* comments are considered negative.

Label	# Comments	# Unique websites
Scam	4164	347
Non-scam	14,828	206
Total	18,992	553

Table 23. Number of text comments in our dataset.

We have created another dataset for evaluating the real usage pattern of potential users of a safe-browsing tool created based on our method. We collected the unique URLs of the webpages visited by the users of the SmartNotes as of April 2012 (Section 3.3.2). Table 24 shows the

results after removing the duplicates and other issues. We have divided the dataset into two parts based on the website traffic rank feature from `alexa.com`. Then, we collected the features from the HostAnalyzer (Section 3.2.1) and the comments from WOT.

Group	# Websites	# Websites with comments	# Comments
Popular	6,658	3,802	45,751
Obscure	3,720	339	2,097
Total	10,378	4,141	47,848

Table 24. Dataset created based on the websites visited by the SmartNotes users.

5.3 Collecting Concept Annotations

After collecting the user comments about websites, we can detect the concepts using our language interpretation technique. These concepts are related to the observations of users regarding the contents and behaviors of websites such as existence of malware.

Automatic concept extraction can benefits from examples of text annotated with concepts. We also need ground truth data to evaluate the effectiveness of our approach. We previously used the category labels to approximate the concepts and we will do the same for the website comments; however, we prefer obtaining more realistic concept annotations.

We introduced our crowdsourcing system, SmartNotes, for collecting comments about websites; we created another system to collect concept annotations from the users using the Amazon Mechanical Turk (`mturk.com`) service. A related work has used crowdsourcing to obtain object attributes, such as color of a bird, by showing images to the users [Law et al., 2011]. The task is considerably harder for the annotators when it is applied natural language text instead of images. We tackled the problems in several iterations and will summarize our approach and findings.

1. *Using structured input:* Our initial approach, which we call *unstructured*, is shown in Figure 20. We provide several examples to the users and let them determine the concepts by providing short phrases. We observed low participation from the users and low quality of the responses. The other extreme in using the structure on the input is shown Figure 21. This approach was more successful because it is easier for the user; however, it limits the freedom of the user to express the concepts and it is not possible to show the large number of all possible concepts in this format. Our final

approach after several more design improvement iterations is shown in Figure 22, which is a hybrid approach. The information requested is split across multiple steps with the instructions and examples interspersed.

2. *Training the users:* It is common knowledge in the crowdsourcing community that the best approach to train the users is providing examples. Users usually work for a small pay and try to minimize their effort. They often will not read the instructions and try to understand the task from the examples. When the task is complex, it is hard to avoid the instructions and sometimes providing the examples limit the users' creativity in expressing the concepts. In Appendix A, we show a complete list of instructions and examples we provided.
3. *Quality control:* Perhaps the biggest challenge in all crowdsourcing systems is obtaining quality results. This subject has been researched extensively, including analysis of the types of the workers' personalities [Bernstein et al., 2010]. Our first step to improve the quality was changing from using the Mechanical Turk website to using their API because of the additional capability in validations and ability to record the history of the workers. The common approach of using gold standard data to evaluate the quality of the workers was not applicable in our case because of possible variations in concept expressions. We considered assigning the qualification to the workers but found the process of filtering the workers quite time consuming.

We have decided to switch to collecting expert annotations instead of the crowdsourced approach for two main reasons:

1. Task of annotating concepts is too complex for an average Mechanical Turk workers and it made process of training the user and quality control cumbersome.
2. There is a need for significant infrastructure code to support a crowdsourcing system for collecting the concept annotation, which was beyond the scope and the time limits of the project we defined.

We believe there is a great potential for the use of crowdsourcing for obtain concept annotations. A possible direction for future work is to embed the crowdsourcing as an active learning oracle in the concept detection system and have the user make the binary decision of accept or reject for a number of candidate concepts detected by the classifier and occasionally ask them to define new concepts.

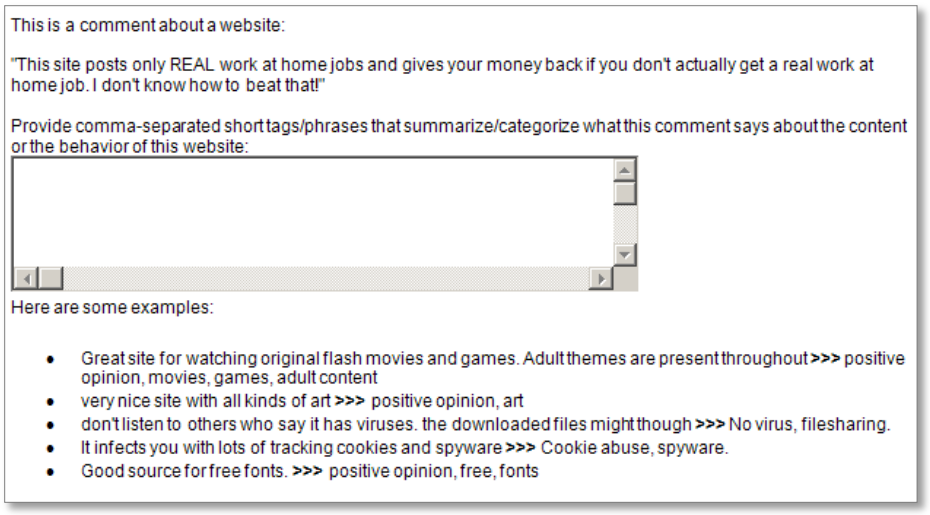


Figure 20. Unstructured approach for collecting concept annotations.

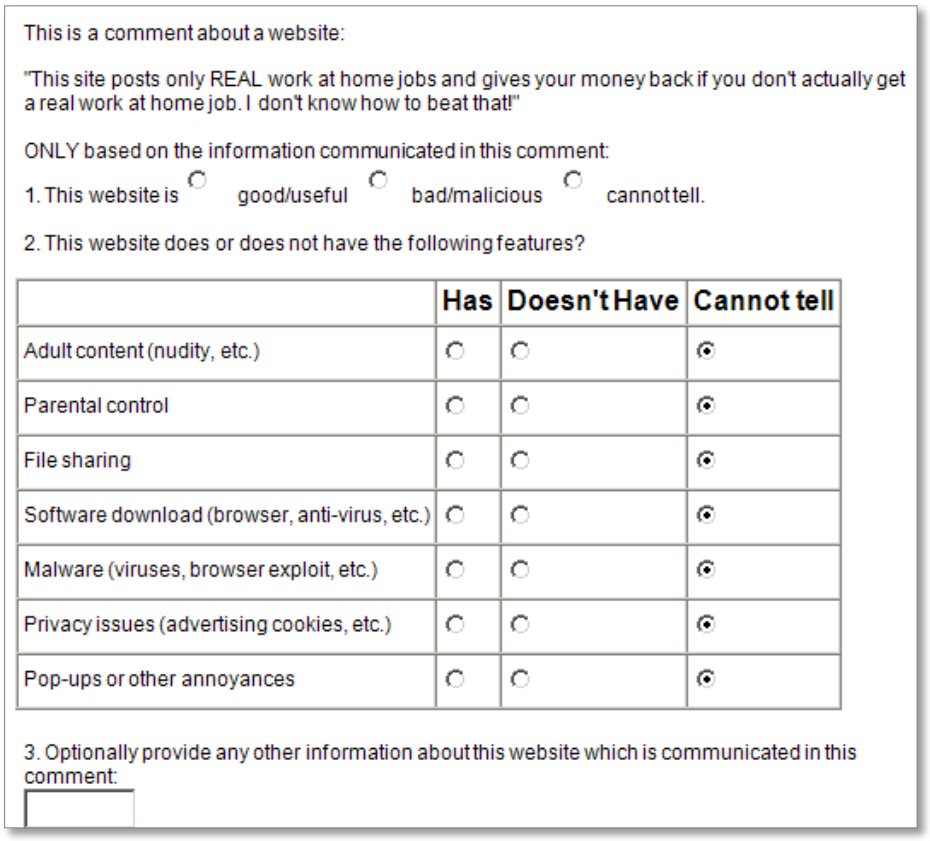


Figure 21. Structured approach for collecting concept annotations.

Figure 22. Final form for the collecting the concept annotations.

5.4 Approach

One approach to include user comments in the scam classifier is appending the text features to the reputation features vector that we used in Chapter 3. We also have category labels available for each comment and we provide results with and without using them.

The text features are different than the reputation features and adding all features to the same vector does not take advantage of structure between the text features. We developed another approach based on topic models in Chapter 4, which identifies the concepts mentioned in the documents. We also define another topic model that takes advantage of the reputation features in parallel with the text features. Previous work has shown potential advantage of generative models for the classification task over their discriminative counterparts [Lacoste-Julien et al., 2009; Ramage et al., 2009]. Our model is most similar to Supervised Topic Model (sLDA) [Blei and McAuliffe, 2007] with the several distinctions:

1. The label random variable is discrete not continuous. It is a Bernoulli random variable and its posterior indicates scam and non-scam prediction. We also used Gibbs sampling [Griffiths and Steyvers, 2004] instead of variation inference used in sLDA.
2. Reputation features and text features are modeled in parallel. This makes our model similar to Correspondence LDA [Blei and Jordan, 2003], which models images and their labels in parallel, and Statistical Entity Topic Model [Newman et al., 2006], which models entity and words in parallel to perform name entity extraction task.
3. We use an additional grouping of data for websites as demonstrated using *plate* notation in graphical models. LDA [Blei et al., 2003] has two nested plates in their graphical model: one that iterates over the documents in the corpus and a second place which is nested within that iterates over the words within the document. Our model has another plate over these two plates that iterates over the websites.

Figure 23 shows the graphical model, notation, and the generative process for the Scam Detection Topic Model. The top portion of the model is the similar to sLDA. Note the labels ls are inferred from the word topics zs. We can also infer the labels from the document topic mixture θ, but our experiments shows the separation of the topics between the labels and the resulting model has lower predictive performance, as previously observed in related work [Blei and Jordan, 2003].

Exact inference is intractable [Blei et al., 2003] and we need to use approximate inference. We derived a collapsed Gibbs sampling for the inference on the top portion of the model. We can only sample zs because θ and ϕ can be integrated out. $n_{-i,j}^{w_i}$ is the number of words w_i, excluding the current word, that is assigned to topic j. $n_{-i,j}$ is total number of words assigned to topic j. $n_{-i,j}^{d_i}$ is the total number of words in document d_i that is assigned to topic j. $n_{-i}^{d_i}$ is the length of document d_i.

$$p(z_i|z_{-i}, w, l) \propto p(w_i|z, l, w_{-i})p(z_i|z_{-i})p(l_s|z, w, l_{-s})$$

$$p(w_i|z, l, w_{-i}) = p(w_i|z, w_{-i}) = \int p(w_i|z_i, \phi)p(\phi|w_{-i}, z_{-i})d\phi = \frac{n_{-i,j}^{w_i} + \beta}{n_{-i,j} + V\beta}$$

Notation

Plates

S Number of websites: $s \in \{1,2,\dots,S\}$

D Number of comments for the current website: $d \in \{1,2,\dots,D\}$

N Number of words in the current comment: $i \in \{1,2,\dots,N\}$

K Number of topics: k $\in \{1,2,\dots,K\}$

Hyperparameters (observed)

α Symmetric Dirichlet distribution prior on comment topic mixture.

β Symmetric Dirichlet distribution prior on word-topic distributions.

γ Symmetric Beta distribution prior on the labels.

t_j Prior on the reputation feature j.

Random variables

θ_d Comment topic mixture.

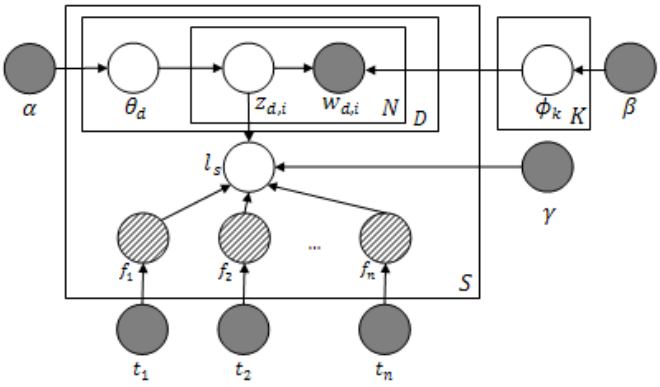
ϕ_k Word topic distribution.

$z_{d,i}$ Word topic.

$w_{d,i}$ Word (observed).

l_s Website label: scam or non-scam.

f_j Reputation feature j (partially observed).



Generative process

For each topic k $\in \{1,2,\dots,K\}$

Generate word topic distribution $\phi_k \sim Dirichlet(\beta)$

For each website $s \in \{1,2,\dots,S\}$

Collect the reputation metrics $\{f_1, f_2, \dots, f_n\}$

For each website comment $d \in \{1,2,\dots,D\}$

Generate a topic mixture $\theta_d \sim Dirichlet(\alpha)$

For each comment word $i \in \{1,2,\dots,N\}$

Generate the word topic $z_{d,i} \sim Multinomial(\theta_d)$

Generate the word $w_{d,i} \sim Multinomial(\phi_{z_{d,i}})$

Generate the website label $l_s \sim \psi(f_s, z_s)$

Figure 23. Graphical model representation of the approach for combining the reputation features, the bottom part of the model which is trained discriminatively as an L1-regularized logistic regression; and textual user comments (top part; trained using Gibbs sampling).

$$p(z_i|z_{-i}) = \int p(z_i|\theta)p(\theta|z_{-i})d\theta = \frac{n_{-i,j}^{d_i} + \alpha}{n_{-i}^{d_i} + K\alpha}$$

When the label l is not observed, then it is sampled as follows. $n_{-i,j}^{l_s}$ is the total number of words with topic j that are assigned to labeled l_s.

$$p(l_s|z, w, l_{-s}) = p(l_s|z, l_{-s}) = \frac{n_{-i,j}^{l_s} + \gamma}{n_{-i,j} + L\gamma}$$

We continue using the same logistic regression classifier for the bottom portion. The combination of the result is performed in the ψ function which can switch between the text and reputation features.

5.5 Experiments

We present experimental results for the task of predicting web scam with addition of the comment that we collected from WOT (Section 5.2) to the features in the scam dataset (Section 3.2.2). The final output of the system is a binary classification decision at the website level and for each website we may have a number of comments available. We evaluate two approaches in moving the comment level predictions to the website level. First, we can combine all the comments for each website into one larger comment. Second, we can process each comment separately and then combine the individual decisions. We use a simple voting method and leave more sophisticated methods, such as using a classifier confidence or weights from user reputation, for future work.

We can also label each comment in two ways: using the last columns of Table 22 to map the user provided category label or we can use the website scam label for all its comments. One observation for the first approach of using the category label mapping was that the dataset contains the websites such as `wikipedia.org`, which are unanimously commented as *good* websites by many users; however websites voted as *bad* websites have fewer comments, with around 20% of the total comments, and there is more disagreement among users.

We measure the statistical significance using the paired *t*-test and permutation test as described in Section 3.2.4.

5.5.1 Adding Text Features

All results are averaged over 10 runs of 10-fold cross validation. Table 25 summarizes the results for text features and reputation features individually and combined, which improves the overall prediction performance. We show three options for the classifier training and evaluations:

1. We can train the classifier by labeling each comment with the label of the website it belongs to and evaluate against the same labels.
2. We can train the same way as #1 and then combine votes from the classification result for each comment to predict the label for the website and evaluate on the website level.
3. We can combine the comment text for each website and then train and evaluate at the website level.

Approach	Precision	Recall	Accuracy	F1	AUC
Reputation features	0.911	0.975	0.953	0.940	0.988
Text features Predict for comments Evaluate	0.590	0.955	0.731	0.728	0.887
Predict for comments Combine by voting	0.876	0.997	0.933	0.920	N/A
Combine comments Predict for website	0.951	0.601	0.730	0.734	0.885
Reputation+Text features	0.956	0.958	0.967	**0.955**	0.982

Table 25. Prediction performance using the text features in addition to the reputation features. F1 improvement is significant p<0.00004, AUC degradation is not significant p<0.1214.

5.5.2 Adding Concepts

We evaluate the performance using the concepts extracted with Algorithm 1 from Chapter 4. We initially use LDA for the topic extraction step of the algorithm and report the results for number of topics $T = 10, 50, 100, 200, 1000$. Other parts of the algorithms are set to the defaults.

# Topics	Precision	Recall	Accuracy	F1	AUC
10	0.704	0.552	0.851	0.618	0.785
50	0.707	0.627	0.861	0.664	0.834
100	0.745	0.639	0.873	0.688	0.851
200	0.769	0.677	0.885	**0.720**	0.868
1000	0.746	0.693	0.881	0.718	0.872

Table 26. Using the topics to predict the website label. For differences in F1, p<0.001, except for the drop, which is p<0.0304.

# Topics	Precision	Recall	Accuracy	F1	AUC
10	0.806	0.775	0.910	0.790	0.919
50	0.813	0.771	0.911	0.791	0.918
100	0.806	0.769	0.909	0.786	0.918
200	0.795	0.764	0.905	0.779	0.911

Table 27. Using the topics and text features to predict the website labels. The increase of F1 in 50 topics is not significant and the decreases are significant at p<0.01.

5.5.3 Adding Category Labels

The comments have category labels that are listed in Table 14 along with a category group, which infers what decision the comment author has made for whether the website is malicious. Table 28 summarized the results for both the comment and website level prediction. Addition of category and category group to the result of Section 5.4.1 did not provide any improvements and therefore omitted for brevity. Obtaining category labels or the category groups from the user mimics the eliciting structured input from the user as opposed to unstructured text. This result is an evidence that obtaining unstructured data can be more beneficial than obtaining structured data.

Approach	Precision	Recall	Accuracy	F1	AUC
Comment Level					
Category	0.757	0.684	0.883	0.718	0.890
Category Group	0.558	0.810	0.818	0.661	0.834
Text	0.806	0.775	0.910	0.790	0.919
Text+Category	0.852	0.773	0.921	0.810	0.937
Text+Category Group	0.849	0.767	0.919	0.806	0.936
Website Level					
Category	0.847	0.983	0.910	0.895	
Category Group	0.893	0.939	0.916	0.897	
Text	0.876	0.997	0.933	0.920	
Text+Category or Group	0.853	0.997	0.919	0.906	

Table 28. Using the category labels from the WOT dataset. F1 and AUC of Text features approach compared to approaches shown above it is statistically significant but not from approaches below it for both comment and website levels.

5.5.4 Error Analysis

We performed error analysis to understand the reason why textual feature by themselves do not perform as well as the reputation features. More crucially, we are interested to understand the reason why addition of the concepts did not help the overall performance. The website traffic rank had the highest positive correlation with the classifier decision score and we also observed that most of the errors are separated based on this feature as shown in Figure 24. We show a breakdown of the results based on four buckets of the traffic rank feature of the website. The number of websites is almost equally distributed across buckets. We can observe that text features have lowest the performance in the case of the obscure websites possibly due to the

quality of the comments. Interestingly, concepts are able to make up some of this performance degradation.

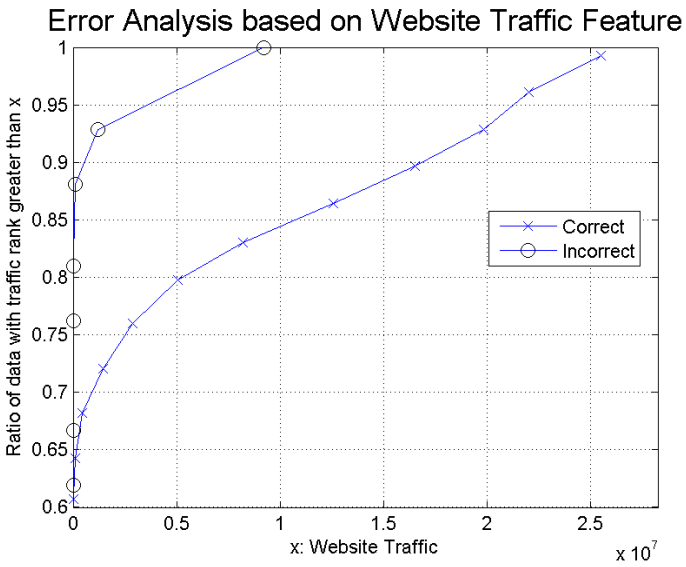


Figure 24. Relation between the classification error errors and the website traffic feature. Majority of the errors are from websites with low traffic.

Website Traffic Rank Buckets	Features				# Websites			
	Reputation	Text	Reputation Text	Reputation Text Concepts	All	Non-scam	Scam	Have Text
All	0.940	0.728	0.955	0.949	837	313	524	535
Rank < 10^3	0.964	0.959	0.800	0.954	191	180	11	130
10^3 < Rank < 10^6	0.910	0.765	0.969	0.900	187	99	88	128
10^6 < Rank	0.793	0.399	0.926	0.828	240	23	217	147
Unknown Rank	0.800	0.224	0.776	0.840	219	11	208	130

Table 29. F1-measure for each feature sets when websites are bucketed based on the global traffic rank of the website.

5.5.5 Scam Detection Topic Model

Scam Detection Topic Model is an extension of topic models introduced in Section 5.3. Parameters are set to $\alpha = 0.1, \beta = 0.01, \gamma = 1, T = 100$. We train the topic model part, the top half of the Figure 23, for 100 Gibbs sampling iterations and then train the logistic regression part, the top bottom half of the Figure 23, until convergences and iterate between the two parts until number of changes in label assignments for the websites fall below a threshold, which we set to

5% of the dataset. The model converges after about 10 iterations. Table 30 summarizes a 5-fold cross validation result, which is not an improvement on using the classifier. Table 31 shows several selected topics associated to scam and non-scam labels.

Approach	Precision	Recall	F1	AUC
Reputation features	0.911	0.975	0.940	0.988
Reputation+Text features	0.956	0.958	**0.955**	0.982
Scam Detection Topic Models	0.846	0.837	0.842	0.892

Table 30. Performance of the Scam Detection Topic Models.

	Topic 13		**Topic 18**		**Topic 19**		**Topic 23**		**Topic 66**	
Scam Topics	enom	0.09	Website	0.06	fake	0.05	cookies	0.07	malware	0.04
	spamvertised	0.05	Mail	0.06	download	0.04	tracking	0.06	bad	0.02
	blacklist	0.03	Scams	0.05	adware	0.04	privacy	0.05	threats	0.02
	domain	0.02	Warning	0.05	software	0.03	facebook	0.05	fake	0.02
	shows	0.02	hostname	0.05	stopzilla	0.03	advertising	0.04	malicious	0.02
	spammed	0.02	dangerous	0.03	antivirus	0.03	adware	0.02	trojans	0.02
	uribl	0.02	known	0.03	rogue	0.03	apps	0.02	downloads	0.02
	spamming	0.02	reason	0.03	virus	0.02	tools	0.02	bots	0.02
	Topic 10		**Topic 58**		**Topic 69**		**Topic 70**		**Topic 80**	
Non-Scam Topics	site	0.26	good	0.47	excellent	0.29	best	0.47	good	0.56
	useful	0.25	really	0.09	useful	0.12	world	0.12	simple	0.02
	informative	0.17	design	0.07	Website	0.03	weather	0.06	popular	0.01
	helpful	0.07	thanks	0.02	today	0.02	love	0.05	provides	0.01
	best	0.06	enjoy	0.02	alternative	0.02	available	0.04	place	0.01
	entertaining	0.02	personally	0.02	yang	0.02	biggest	0.03	convenient	0.00
	reliable	0.02	deals	0.02	health	0.02	organization	0.01	exist	0.00
	fast	0.01	nice	0.01	pro	0.02	management	0.01	reputable	0.00

Table 31. Selected topics from the scam detector topic model.

5.6 Discussion

We investigated several approaches for integrating the user comments to improve the prediction of malicious websites. We showed that the overall performance can be improved, especially if we focus on different classes of websites. We selected traffic rank in our error analysis and split the space of websites to show how the prediction performance can benefit from the addition of the text features and the task-based concepts extracted from the comments. Scam Detection Topic Model is an instance of a joint approach to integrate comments and the reputation features. While it was able to identify the scam related concept, further work is needed to improve its prediction performance.

Chapter 6

Conclusion

We have developed a machine learning approach for detection of malicious websites in Chapter 3, and briefly introduced SmartNotes as a crowdsourcing approach to collect additional information from website users. Then we introduced a general language interpretation framework for extracting concepts from natural language text in Chapter 4, which enables us to process the unstructured information provided by the website users. Finally in Chapter 5, we provided experiments to evaluate the performance of adding text features to the reputation features for the task of detecting malicious websites. We now describe further how the complete system works and outline directions for the future work.

6.1 Complete System

We introduced SmartNotes, our crowdsourcing platform for detecting malicious websites in Section 3.3. Once the user installs the add-on, the system starts analyzing the URLs that the user is visiting. The user can add notes to any website from the user interface we showed in Section

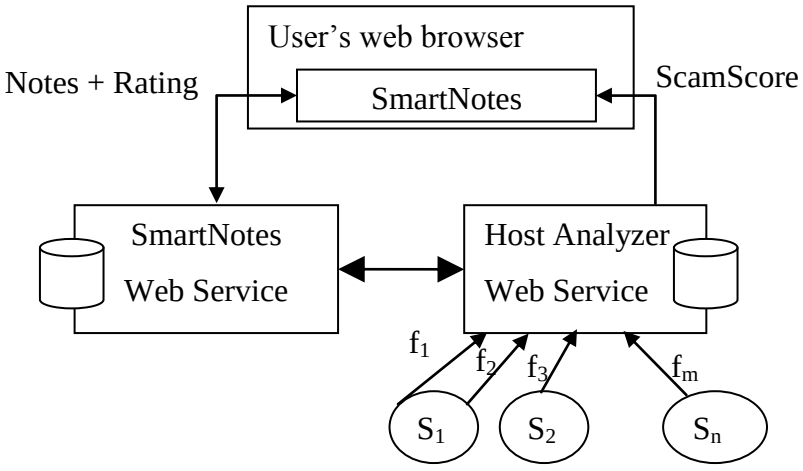


Figure 25. The SmartNotes architecture. The arrows show communications over HTTP. Host Analyzer web service calculates ScamScore based on *m* features from *n* sources. It also uses the notes from the SmartNotes web service.

3.3 and use the question-answering feature to communicate with the other users.

The overview of the architecture used in SmartNotes is shown in Figure 25. The browser add-on communicates with the backend webservices. The notes and rating provided by the user is saved in the database, which is then processed by Host Analyzer that contains an implementation of our language interpretation framework and the automatic data collection mechanism for the reputation features. The final output is the classifier decision score, which is the probability of a given website being a scam. This number is transferred back to the browser add-on. Currently, this feature can be invoked manually as shown in Figure 26; however as part of the future work we can investigate how this should be provided to the user without any actions, similar to the safe-browsing feature introduced in many recent browsers.

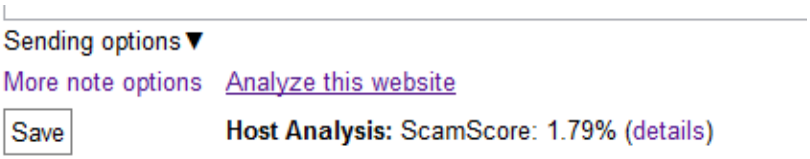


Figure 26. ScamScore shows the result of analysis of the website.

6.2 Future Work

We explore potential extensions of our work, particularly toward implementing this system as a practical safe-browsing tool.

6.2.1 Finding Experts

Malicious websites are designed to trick users and therefore the quality of the comments is very important. Our technique extracts the concepts instead of using the direct votes and therefore it is less prone to errors; however, specific concepts need to be mentioned in the text and expert users are better at identifying important aspects of a websites. Expert search is a research area with the goal of finding expert users within a network based on the contents and other features extracted from their interactions [Mccallum et al., 2005; Zhang et al., 2007a; Fang and Zhai, 2007; Zhang et al., 2008a; Karimzadehgan et al., 2009; Johri et al., 2010]. We believe some of these methods are complementary to the crowdsourcing solutions.

6.2.2 Trustworthy Comments

Once the system is widely adopted by the web users, the scammers are motivated to bias its accuracy by providing fake comments. Existing work on identifying fake reviews is relevant [Lim et al., 2010; Mukherjee et al., 2012] (www.cs.uic.edu/~liub/FBS/fake-reviews.html). These approaches rely on features extracted from the text or the network of the user to estimate the quality of the comments. We believe the most effective approach against such manipulations is through the software design of the websites. Here are several possible approaches:

1. Associating user reputation information with the user account such as number of votes the user received, total number of reviews, and number of friends.
2. Making process of registering accounts expensive in terms of human effort. For example, creating legitimate looking fake accounts in social networks is not trivial because the users have to spend a lot of time to provide contents and connect to other users.
3. Tying accounts to the real identity of the users. This approach can range from requiring emails from established domain names such as the university emails, to including personal information such as credit cards or drive license as part of the sign-up process.
4. Defining elaborate measures of user comment quality such as diversity. It is often difficult to generate a large number of fake accounts and provide high quality diverse content for them.

6.2.3 Real-time Language Interpretation

We believe there is a significant benefit if the result of the language interpretation is provided in real-time to the users, that is as they are typing their comments. This approach increases the chance of user interaction with the system and providing feedback on incorrect or incomplete interpretations. We have created an initial prototype for how the user interaction can be designed. Recent work in online algorithms for the topic model approximate inference [Hoffman et al., 2010; Wang et al., 2011] and parallelization of inference [Liu et al., 2011; Smola and Narayanamurthy, 2010; Wang et al., 2009; Newman et al., 2009] potentially enable us to design systems that perform faster than models based on traditional topic model inference approaches.

6.2.4 Collection of Concept Annotations

In Section 5.3, we discussed some of the limitation of collecting the concept annotation from users. Performing the interpretation in real-time as suggested in Section 6.2.3 can be one step

toward getting more engagement from the user, however, we believe the main barrier is the lack of a suitable and innovative user interface for this task which can be improved by human-computer interaction research. We obtain useful statistics about the textual data using the proposed models and we cannot communicate them effectively with the user to obtain their feedback. The proposed approaches are flexible for a specific task and accompanying text data but as a trade-off, they are not designed to represent the world's background knowledge and the goal has been to have human guidance as part of the process.

We believe the future work in this area should experiment with various visualization of the contextual information and scores provided by the probabilistic models, and also consider the degree of expertise of the users who interact with the system. The users can be motivated by observing how their contribution to the system improves the performance of the task for themselves and other users.

6.2.5 SmartNotes as a Proxy Server

SmartNotes can be implemented as a fast proxy that monitors the websites visited by an organization. An example of such implementations is SpyProxy [Moshchuk et al., 2007], which detects malicious activity by executing the page on a proxy server and monitor its behavior patterns. The main advantage of this approach is that the protection against malicious website is not dependent on individual users and it enables more efficient software designs where requested are processed in batches.

6.2.6 Extending Language Interpretation Framework

The developed framework for detecting task-based concepts is general and can be applied to domains other than the malicious website detections where textual information is available for improving a task. Examples include:

1. *User reviews:* The task is improving the user satisfaction after making a choice for a product or service and the concepts are the discriminative aspects of the products and services mentioned in the review, such as quality of the pictures for a digital camera or service quality in a restaurant.
2. *Robot instructions:* The task is for an agent to correctly perform an action based on an input text and the concepts are the possible actions that can be requested, such as picking

up an object. Our framework needs to be extended to identify finer distinctions between the concepts.

3. *Information retrieval:* The task is ranking of the documents based on the relevance to a query and the concepts are topics within the documents. Similar to the robot instructions, finer separation of the concepts is necessary. The background knowledge in the form of a knowledge base can be encoded in the concept graph and used in the parameter optimization step.

There are many specific choices within the steps of the main algorithm that requires further experimentation in future work. The core idea is that the task performance improves by iteratively improving the extracted concept quality. We presented an initial unifying view for the application of dimensionality reduction methods and topic models and we believe further work in this direction can be very beneficial.

Bibliography

[Aldous, 1985] D. Aldous. Exchangeability and related topics. *École d'Été de Probabilités de Saint-Flour XIII - 1983*, pages 1–198, 1985.

[Anderson *et al.*, 2007] D.S. Anderson, C. Fleizach, S. Savage, and G.M. Voelker. Spamscatter: Characterizing internet scam hosting infrastructure. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, page 10. USENIX Association, 2007.

[Andoni and Indyk, 2006] A. Andoni and P. Indyk. Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions. In *Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on*, pages 459–468. Ieee, 2006.

[Andrew Y. Ng, 2002] Michael I. Jordan Andrew Y. Ng. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. *Advances in neural information processing systems*, 14:841, 2002.

[Andrzejewski *et al.*, 2009] D. Andrzejewski, X. Zhu, and M. Craven. Incorporating domain knowledge into topic modeling via dirichlet forest priors. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 25–32. ACM, 2009.

[Bahl *et al.*, 1983] L.R. Bahl, F. Jelinek, and R.L. Mercer. A maximum likelihood approach to continuous speech recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (2):179–190, 1983.

[Banerjee and Pedersen, 2003] S. Banerjee and T. Pedersen. The design, implementation, and use of the ngram statistics package. *Computational Linguistics and Intelligent Text Processing*, pages 370–381, 2003.

[Barth *et al.*, 2008] Adam Barth, Collin Jackson, and John C. Mitchell. Robust defenses for cross-site request forgery. In *In To appear at the 15th ACM Conference on Computer and Communications Security (CCS*, 2008.

[Bayer *et al.*, 2009] U. Bayer, P.M. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda. Scalable, behavior-based malware clustering. In *Network and Distributed System Security Symposium (NDSS)*. Citeseer, 2009.

[Becchetti *et al.*, 2008] Luca Becchetti, Carlos Castillo, Debora Donato, S. Ricardo Baeza-YATE, and Stefano Leonardi. Link analysis for web spam detection. *ACM Trans. Web*, 2(1):1–42, feb 2008.

[Bernstein *et al.*, 2010] M.S. Bernstein, G. Little, R.C. Miller, B. Hartmann, M.S. Ackerman, D.R. Karger, D. Crowell, and K. Panovich. Soylent: a word processor with a crowd inside. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*, pages 313–322. ACM, 2010.

[Bernstein *et al.*, 2011] M.S. Bernstein, J. Brandt, R.C. Miller, and D.R. Karger. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 33–42. ACM, 2011.

[Bernstein *et al.*, 2012] M.S. Bernstein, D.R. Karger, R.C. Miller, and J. Brandt. Analytic methods for optimizing realtime crowdsourcing. *Arxiv preprint arXiv:1204.2995*, 2012.

[Blei and Jordan, 2003] D.M. Blei and M.I. Jordan. Modeling annotated data. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 127–134. ACM, 2003.

[Blei and Lafferty, 2006] David Blei and John D. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems*, 2006.

[Blei and Lafferty, 2009] D.M. Blei and J.D. Lafferty. Visualizing topics with multi-word expressions. *Arxiv preprint arXiv:0907.1013*, 2009.

[Blei and McAuliffe, 2007] D.M. Blei and J.D. McAuliffe. Supervised topic models. *Advances in Neural Information Processing Systems*, 20, 2007.

[Blei *et al.*, 2003] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, jan 2003.

[Bordes *et al.*, 2010] A. Bordes, N. Usunier, R. Collobert, and J. Weston. Towards understanding situated natural language. In *Proc. of the 13th Intern. Conf. on Artif. Intel. and Stat*, volume 9, pages 65–72, 2010.

[Boyd-Graber *et al.*, 2009] J. Boyd-Graber, J. Chang, S. Gerrish, C. Wang, and D. Blei. Reading tea leaves: How humans interpret topic models. In *Proceedings of the 23rd Annual Conference on Neural Information Processing Systems*, 2009.

[Branavan *et al.*, 2009] SRK Branavan, H. Chen, L.S. Zettlemoyer, and R. Barzilay. Reinforcement learning for mapping instructions to actions. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 82–90. Association for Computational Linguistics, 2009.

[Branavan *et al.*, 2010] S. R. K. Branavan, Luke S. Zettlemoyer, and Regina Barzilay. Reading between the lines: learning to map high-level instructions to commands. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 1268–1277, Morristown, NJ, USA, 2010.

[Breiman, 2001] Leo Breiman. *Random Forests*, volume 45, pages 5–32. 2001.

[Cai and Hofmann, 2004] Lijuan Cai and Thomas Hofmann. Hierarchical document categorization with support vector machines. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 78–87, New York, NY, USA, 2004.

[Cai *et al.*, 2008] D. Cai, Q. Mei, J. Han, and C. Zhai. Modeling hidden topics on document manifold. In *Proceeding of the 17th ACM conference on Information and knowledge management*, pages 911–920. ACM, 2008.

[Cai *et al.*, 2009] D. Cai, X. Wang, and X. He. Probabilistic dyadic data analysis with local and global consistency. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 105–112. ACM, 2009.

[Carlson *et al.*, 2010] A. Carlson, J. Betteridge, R.C. Wang, E.R. Hruschka Jr, and T.M. Mitchell. Coupled semi-supervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110. ACM, 2010.

[Castillo *et al.*, 2006] Carlos Castillo, Debora Donato, Luca Becchetti, Paolo Boldi, Stefano Leonardi, Massimo Santini, and Sebastiano Vigna. A reference collection for web spam. *SIGIR Forum*, 40(2):11–24, dec 2006.

[Castillo *et al.*, 2007] Carlos Castillo, Debora Donato, Aristides Gionis, Vanessa Murdock, and Fabrizio Silvestri. Know your neighbors: web spam detection using the web topology. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '07, pages 423–430, New York, NY, USA, 2007.

[Charikar, 2002] M.S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 380–388. ACM, 2002.

[Chemudugunta *et al.*, 2008a] C. Chemudugunta, A. Holloway, P. Smyth, and M. Steyvers. Modeling documents by combining semantic concepts with unsupervised statistical learning. *The Semantic Web-ISWC 2008*, pages 229–244, 2008.

[Chemudugunta *et al.*, 2008b] C. Chemudugunta, P. Smyth, and M. Steyvers. Text modeling using unsupervised topic models and concept hierarchies. *Arxiv preprint arXiv:0808.0973*, 2008.

[Chen and Goodman, 1996] Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, ACL '96, pages 310–318, Stroudsburg, PA, USA, 1996.

[Choi *et al.*, 2011] H. Choi, B.B. Zhu, and H. Lee. Detecting malicious web links and identifying their attack types. In *Proceedings of the 2nd USENIX conference on Web application development*, pages 11–11. USENIX Association, 2011.

[Chung *et al.*, 2010] Young-joo Chung, Masashi Toyoda, and Masaru Kitsuregawa. Identifying spam link generators for monitoring emerging web spam. In *Proceedings of the 4th workshop on Information credibility*, WICOW '10, pages 51–58, New York, NY, USA, 2010.

[Church, 1988] K.W. Church. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the second conference on Applied natural language processing*, pages 136–143. Association for Computational Linguistics, 1988.

[Cormack *et al.*, 2010] Gordon V. Cormack, Mark D. Smucker, and Charles L. A. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. apr 2010.

[Cormack, 2007a] Gordon V. Cormack. Content-based web spam detection. In *In Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb*, 2007.

[Cormack, 2007b] Gordon V. Cormack. Trec 2007 spam track overview. In Ellen M. Voorhees and Lori P. Buckland, editors, *TREC*, volume Special Publication 500-274. National Institute of Standards and Technology (NIST), 2007.

[Cova *et al.*, 2010] Marco Cova, Christopher Kruegel, and Giovanni Vigna. Detection and analysis of drive-by-download attacks and malicious javascript code. In *WWW '10:*

Proceedings of the 19th international conference on World wide web, pages 281–290, New York, NY, USA, 2010.

[Croft and Liu, 2005] WB Croft and X. Liu. Statistical language modeling for information retrieval, 2005.

[de Marneffe *et al.*, 2006] M.C. de Marneffe, B. MacCartney, T. Grenager, D. Cer, A. Rafferty, and C.D. Manning. Learning to distinguish valid textual entailments. In *Second Pascal RTE Challenge Workshop*, 2006.

[Deerwester *et al.*, 1990] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[Dietterich and Bakiri, 1995] T.G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Arxiv preprint cs/9501101*, 1995.

[Donmez and Carbonell, 2008] Pinar Donmez and Jaime G. Carbonell. Proactive learning: cost-sensitive active learning with multiple imperfect oracles. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, pages 619–628, New York, NY, USA, 2008.

[Druck *et al.*, 2008] G. Druck, G. Mann, and A. McCallum. Learning from labeled features using generalized expectation criteria. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 595–602. ACM, 2008.

[Fan and Lin, 2007] R.E. Fan and C.J. Lin. A study on threshold selection for multi-label classification. *Department of Computer Science, National Taiwan University*, 2007.

[Fang and Zhai, 2007] Hui Fang and Chengxiang Zhai. Probabilistic models for expert finding. In *In: ECIR*, volume 2007, pages 418–430, 2007.

[Freund *et al.*, 1997] Y. Freund, R.E. Schapire, Y. Singer, and M.K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 334–343. ACM, 1997.

[Garera *et al.*, 2007] Sujata Garera, Niels Provos, Monica Chew, and Aviel D. Rubin. A framework for detection and measurement of phishing attacks. In *Proceedings of the 2007 ACM workshop on Recurring malcode*, WORM '07, pages 1–8, New York, NY, USA, 2007.

[Ghani, 2000] Rayid Ghani. Using error-correcting codes for text classification. In Pat Langley, editor, *Proceedings of ICML-00, 17th International Conference on Machine Learning*, pages 303–310, Stanford, US, 2000.

[Griffiths and Steyvers, 2004] T.L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences of the United States of America*, 101(Suppl 1):5228, 2004.

[Gyöngyi and Garcia-Molina, 2005] Zoltán Gyöngyi and Hector Garcia-Molina. Link spam alliances. In *Proceedings of the 31st international conference on Very large data bases*, VLDB '05, pages 517–528. VLDB Endowment, 2005.

[Gyongyi and Molina, 2005] Z. Gyongyi and H. Garcia Molina. Web spam taxonomy, 2005.

[Hastie and Tibshirani, 1998] T. Hastie and R. Tibshirani. Classification by pairwise coupling. *The annals of statistics*, 26(2):451–471, 1998.

[Ho, 1998] T.K. Ho. The random subspace method for constructing decision forests. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 20(8):832–844, 1998.

[Hoffman *et al.*, 2010] M.D. Hoffman, D.M. Blei, and F. Bach. Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems*, 23:856–864, 2010.

[Hofmann, 1999] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of Uncertainty in Artificial Intelligence, UAIâ€™99*, pages 289–296. Citeseer, 1999.

[Hou *et al.*, 2010] Yung-Tsung Hou, Yimeng Chang, Tsuhan Chen, Chi-Sung Laih, and Chia-Mei Chen. Malicious web content detection by machine learning. *Expert Syst. Appl.*, 37(1):55–60, jan 2010.

[Huh and Fienberg, 2010] Seungil Huh and Stephen E. Fienberg. Discriminative topic modeling based on manifold learning. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '10, pages 653–662, New York, NY, USA, 2010.

[Indyk and Motwani, 1998] P. Indyk and R. Motwani. Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 604–613. ACM, 1998.

[Ji *et al.*, 2008] S. Ji, L. Tang, S. Yu, and J. Ye. Extracting shared subspace for multi-label classification. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 381–389. ACM, 2008.

[Joachims, 1999] T. Joachims. Transductive inference for text classification using support vector machines. In *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*, pages 200–209. MORGAN KAUFMANN PUBLISHERS, INC., 1999.

[Joachims, 2002] T. Joachims. *Learning to classify text using support vector machines: Methods, theory and algorithms*, volume 186. Kluwer Academic Publishers Norwell, MA, USA:, 2002.

[Johri *et al.*, 2010] Nikhil Johri, Dan Roth, and Yuancheng Tu. Experts' retrieval with multiword-enhanced author topic model. In *Proceedings of the NAACL HLT 2010 Workshop on Semantic Search*, SS '10, pages 10–18, Stroudsburg, PA, USA, 2010.

[Jolliffe and MyiLibrary, 2002] I.T. Jolliffe and MyiLibrary. *Principal component analysis*, volume 2. Wiley Online Library, 2002.

[Karimzadehgan *et al.*, 2009] M. Karimzadehgan, R. White, and M. Richardson. Enhancing expert finding using organizational hierarchies. *Advances in Information Retrieval*, pages 177–188, 2009.

[Kazawa *et al.*, 2005] H. Kazawa, T. Izumitani, H. Taira, and E. Maeda. Maximal margin labeling for multi-topic text categorization. *Advances in Neural Information Processing Systems*, 17:649–656, 2005.

[Kullback and Leibler, 1951] S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[Lacoste-Julien *et al.*, 2009] S. Lacoste-Julien, F. Sha, and M.I. Jordan. Disclda: Discriminative learning for dimensionality reduction and classification. *Advances in Neural Information Processing Systems*, 21:897–904, 2009.

[Lakkaraju *et al.*, 2011] H. Lakkaraju, C. Bhattacharyya, I. Bhattacharya, and S. Merugu. Exploiting coherence for the simultaneous discovery of latent facets and associated sentiments. In *SIAM International Conference on Data Mining. URL http://mllab. csa. iisc. ernet. in/html/pubs/FINAL. pdf*, 2011.

[Landauer *et al.*, 1998] T.K. Landauer, P.W. Foltz, and D. Laham. An introduction to latent semantic analysis. *Discourse processes*, 25(2-3):259–284, 1998.

[Laskov *et al.*, 2005] Pavel Laskov, Patrick Düssel, Christin Schäfer, and Konrad Rieck. Learning intrusion detection: supervised or unsupervised? In *IN: IMAGE ANALYSIS AND PROCESSING, PROC. OF 13TH ICIAP CONFERENCE. (2005) 50â€“57*, volume 3617, pages 50–57, 2005.

[Lau *et al.*, 2011] J.H. Lau, K. Grieser, D. Newman, and T. Baldwin. Automatic labelling of topic models. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1536–1545. Association for Computational Linguistics, 2011.

[Law *et al.*, 2011] E. Law, B. Settles, A. Snook, H. Surana, L. von Ahn, and T. Mitchell. Human computation for attribute and attribute value acquisition. In *Proceedings of the CVPR Workshop on Fine-Grained Visual Categorization*, 2011.

[Lay and Barbu, 2010] Nathan Lay and Adrian Barbu. Supervised aggregation of classifiers using artificial prediction markets. In Johannes Fürnkranz and Thorsten Joachims, editors, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 591–598, Haifa, Israel, jun 2010.

[Lee *et al.*, 2010] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: social honeypots + machine learning. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '10, pages 435–442, New York, NY, USA, 2010.

[Liang *et al.*, 2009] P. Liang, M.I. Jordan, and D. Klein. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*, pages 91–99. Association for Computational Linguistics, 2009.

[Lim *et al.*, 2010] E.P. Lim, V.A. Nguyen, N. Jindal, B. Liu, and H.W. Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM international conference on Information and knowledge management*, pages 939–948. ACM, 2010.

[Lin, 1991] J. Lin. Divergence measures based on the shannon entropy. *Information Theory, IEEE Transactions on*, 37(1):145–151, 1991.

[Liu *et al.*, 2003] Y. Liu, J. Carbonell, and R. Jin. A new pairwise ensemble approach for text classification. *Machine Learning: ECML 2003*, pages 277–288, 2003.

[Liu *et al.*, 2005a] Tie Y. Liu, Yiming Yang, Hao Wan, Hua J. Zeng, Zheng Chen, and Wei Y. Ma. Support vector machines classification with a very large-scale taxonomy. *SIGKDD Explor. Newsl.*, 7(1):36–43, 2005.

[Liu *et al.*, 2005b] T.Y. Liu, Y. Yang, H. Wan, H.J. Zeng, Z. Chen, and W.Y. Ma. Support vector machines classification with a very large-scale taxonomy. *ACM SIGKDD Explorations Newsletter*, 7(1):36–43, 2005.

[Liu *et al.*, 2011] Z. Liu, Y. Zhang, E.Y. Chang, and M. Sun. Plda+: Parallel latent dirichlet allocation with data placement and pipeline processing. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(3):26, 2011.

[Lovász and Plummer, 1986] L. Lovász and M.D. Plummer. *Matching theory*. Number 121. North Holland, 1986.

[Low *et al.*, 2011] Y. Low, D. Agarwal, and A.J. Smola. Multiple domain user personalization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 123–131. ACM, 2011.

[Ma *et al.*, 2009a] J. Ma, L.K. Saul, S. Savage, and G.M. Voelker. Beyond blacklists: learning to detect malicious web sites from suspicious urls. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1245–1254. ACM, 2009.

[Ma *et al.*, 2009b] J. Ma, L.K. Saul, S. Savage, and G.M. Voelker. Identifying suspicious urls: an application of large-scale online learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 681–688. ACM, 2009.

[Mccallum *et al.*, 2005] Andrew Mccallum, Andres Corrada-Emmanuel, and Xuerui Wang. Topic and role discovery in social networks. pages 786–791, 2005.

[McGrath and Gupta, 2008] D. Kevin McGrath and Minaxi Gupta. Behind phishing: an examination of phisher modi operandi. In *Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, LEET'08, pages 4:1–4:8, Berkeley, CA, USA, 2008.

[Mei *et al.*, 2007] Q. Mei, X. Shen, and C.X. Zhai. Automatic labeling of multinomial topic models. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 490–499. ACM, 2007.

[Mimno and McCallum, 2008] D. Mimno and A. McCallum. Topic models conditioned on arbitrary features with dirichlet-multinomial regression. In *Uncertainty in Artificial Intelligence*, pages 411–418, 2008.

[Mimno *et al.*, 2011] D. Mimno, H.M. Wallach, E.T.M. Leenders, and A. McCallum. Optimizing semantic coherence in topic models. EMNLP, 2011.

[Mitchell *et al.*, 2009] T. Mitchell, J. Betteridge, A. Carlson, E. Hruschka, and R. Wang. Populating the semantic web by macro-reading internet text. *The Semantic Web-ISWC 2009*, pages 998–1002, 2009.

[Mitchell, 2006] T.M. Mitchell. Chapter 1; generative and discriminative classifiers: Naive bayes and logistic regression. *Machine Learning, Sep*, 21:17, 2006.

[Moore *et al.*, 2009] Tyler Moore, Richard Clayton, and Henry Stern. Temporal correlations between spam and phishing websites. In *Proceedings of the 2nd USENIX conference on Large-scale exploits and emergent threats: botnets, spyware, worms, and more*, LEET'09, pages 5–5, Berkeley, CA, USA, 2009.

[Moshchuk *et al.*, 2007] Alexander Moshchuk, Tanya Bragin, Damien Deville, Steven D. Gribble, and Henry M. Levy. Spyproxy: execution-based detection of malicious web content. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, SS'07, pages 3:1–3:16, Berkeley, CA, USA, 2007.

[Mukherjee *et al.*, 2012] Arjun Mukherjee, Bing Liu, and Natalie Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, pages 191–200, New York, NY, USA, 2012.

[Nadeau and Sekine, 2007] David Nadeau and Satoshi Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, jan 2007.

[Newman *et al.*, 2006] D. Newman, C. Chemudugunta, and P. Smyth. Statistical entity-topic models. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 680–686. ACM, 2006.

[Newman *et al.*, 2009] D. Newman, A. Asuncion, P. Smyth, and M. Welling. Distributed algorithms for topic models. *The Journal of Machine Learning Research*, 10:1801–1828, 2009.

[Newman *et al.*, 2010] D. Newman, J.H. Lau, K. Grieser, and T. Baldwin. Automatic evaluation of topic coherence. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 100–108. Association for Computational Linguistics, 2010.

[Newman *et al.*, 2011] D. Newman, E. Bonilla, and W. Buntine. Improving topic coherence with regularized topic models. 2011.

[Nigam *et al.*, 2000] K. Nigam, A.K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2):103–134, 2000.

[Ntoulas *et al.*, 2006] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly. Detecting spam web pages through content analysis. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 83–92, New York, NY, USA, 2006.

[Pang *et al.*, 2002] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2002.

[Paul and Dredze, 2011] M.J. Paul and M. Dredze. You are what you tweet: Analyzing twitter for public health. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media (ICWSM)*, 2011.

[Popescu and Etzioni, 2005] Ana M. Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pages 339–346, 2005.

[Prakash *et al.*, 2010] Pawan Prakash, Manish Kumar, Ramana Rao Kompella, and Minaxi Gupta. Phishnet: predictive blacklisting to detect phishing attacks. In *Proceedings of the 29th*

conference on Information communications, INFOCOM'10, pages 346–350, Piscataway, NJ, USA, 2010.

[Raghavan *et al.*, 2006] H. Raghavan, O. Madani, and R. Jones. Active learning with feedback on features and instances. *The Journal of Machine Learning Research*, 7:1655–1686, 2006.

[Ramachandran and Feamster, 2006] Anirudh Ramachandran and Nick Feamster. Understanding the network-level behavior of spammers. In *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, SIGCOMM '06, pages 291–302, New York, NY, USA, 2006.

[Ramage *et al.*, 2009] Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora. *EMNLP '09: Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 248–256, 2009.

[Ramage *et al.*, 2011] D. Ramage, C.D. Manning, and S. Dumais. Partially labeled topic models for interpretable text mining. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 457–465. ACM, 2011.

[Ravichandran *et al.*, 2005] D. Ravichandran, P. Pantel, and E. Hovy. Randomized algorithms and nlp: Using locality sensitive hash functions for high speed noun clustering. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 43, page 622, 2005.

[Rennie and Rifkin, 2001] J.D.M. Rennie and R. Rifkin. Improving multiclass text classification with the support vector machine. 2001.

[Richard *et al.*, 2001] Bob C. Richard, Wordseye An, Automatic Text-to scene, and Conversion System. Wordseye: An automatic text-to-scene conversion system, 2001.

[Rieck *et al.*, 2008] Konrad Rieck, Thorsten Holz, Carsten Willems, Patrick DÃÂ¼ssel, and Pavel Laskov. Learning and classification of malware behavior. In Diego Zamboni, editor, *Detection of Intrusions and Malware, and Vulnerability Assessment*, volume 5137 of *Lecture Notes in Computer Science*, chapter 6, pages 108–125–125. Springer Berlin / Heidelberg, Berlin, Heidelberg, 2008.

[Rosen-Zvi *et al.*, 2004] Michal Rosen-Zvi, Thomas Griffiths, Mark Steyvers, and Padhraic Smyth. The author-topic model for authors and documents. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence*, pages 487–494, Arlington, VA, USA, 2004.

[Rubin *et al.*, 2011] T.N. Rubin, A. Chambers, P. Smyth, and M. Steyvers. Statistical topic models for multi-label document classification. *Machine Learning*, pages 1–52, 2011.

[Salakhutdinov and Hinton, 2009] Ruslan Salakhutdinov and Geoffrey Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, jul 2009.

[Sculley and Cormack, 2008] D. Sculley and G.V. Cormack. Filtering email spam in the presence of noisy user feedback. In *Proc of the fifth conf on email and anti-spam*. Citeseer, 2008.

[Settles, 2011] B. Settles. Closing the loop: Fast, interactive semi-supervised annotation with queries on features and instances. In *Proceedings of the Conference on Empirical Methods in*

Natural Language Processing, pages 1467–1478. Association for Computational Linguistics, 2011.

[Settles, 2012] Burr Settles. *Active Learning*. Morgan & Claypool, 2012.

[Sharifi *et al.*, 2010] Mehrbod Sharifi, Eugene Fink, and Jaime G. Carbonell. Learning of personalized security settings. pages 3428–3432, oct 2010.

[Sharifi *et al.*, 2011a] Mehrbod Sharifi, Eugene Fink, and Jaime G. Carbonell. Detection of internet scam using logistic regression. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, 2011.

[Sharifi *et al.*, 2011b] Mehrbod Sharifi, Eugene Fink, and Jaime G. Carbonell. Smartnotes: Application of crowdsourcing to the detection of web threats. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, 2011.

[Sharifi, 2009] Mehrbod Sharifi. Semi-supervised extraction of entity aspects using topic models. Master's thesis, Carnegie Mellon University, 2009.

[Shen *et al.*, 2006] Guoyang Shen, Bin Gao, Tie-Yan Liu, Guang Feng, Shiji Song, and Hang Li. Detecting link spam using temporal information. In *Proceedings of the Sixth International Conference on Data Mining*, ICDM '06, pages 1049–1053, Washington, DC, USA, 2006.

[Sheng *et al.*, 2010] Steve Sheng, Mandy Holbrook, Ponnurangam Kumaraguru, Lorrie F. Cranor, and Julie Downs. Who falls for phish?: a demographic analysis of phishing susceptibility and effectiveness of interventions. In *CHI '10: Proceedings of the 28th international conference on Human factors in computing systems*, pages 373–382, New York, NY, USA, 2010.

[Smola and Narayanamurthy, 2010] A. Smola and S. Narayanamurthy. An architecture for parallel topic models. *Proceedings of the VLDB Endowment*, 3(1-2):703–710, 2010.

[Smucker *et al.*, 2007] M.D. Smucker, J. Allan, and B. Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 623–632. ACM, 2007.

[Snow *et al.*, 2006] R. Snow, D. Jurafsky, and A.Y. Ng. Semantic taxonomy induction from heterogenous evidence. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 801–808. Association for Computational Linguistics, 2006.

[Sun *et al.*, 2008] J. Sun, Y. Xie, H. Zhang, and C. Faloutsos. Less is more: Sparse graph mining with compact matrix decomposition. *Statistical Analysis and Data Mining*, 1(1):6–22, 2008.

[Taskar *et al.*, 2005] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: A large margin approach. In *Proceedings of the 22nd international conference on Machine learning*, pages 896–903. ACM, 2005.

[Teh *et al.*, 2006] Y.W. Teh, M.I. Jordan, M.J. Beal, and D.M. Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[Ting *et al.*, 2011] K.M. Ting, J.R. Wells, S.C. Tan, S.W. Teng, and G.I. Webb. Feature-subspace aggregating: ensembles for stable and unstable learners. *Machine Learning*, 82(3):375–397, 2011.

[Titov and McDonald, 2008a] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models. In *Proceeding of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[Titov and Mcdonald, 2008b] Ivan Titov and Ryan Mcdonald. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL-08: HLT*, pages 308–316, Columbus, Ohio, jun 2008.

[Tong and Koller, 2002] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *The Journal of Machine Learning Research*, 2:45–66, 2002.

[Ueda and Saito, 2002] N. Ueda and K. Saito. Parametric mixture models for multi-labeled text. *Advances in neural information processing systems*, 15:721–728, 2002.

[Van Der Maaten *et al.*, 2007] LJP Van Der Maaten, EO Postma, and HJ Van Den Herik. Dimensionality reduction: A comparative review. *Published online*, 10(February):1–35, 2007.

[Von Ahn *et al.*, 2008] L. Von Ahn, B. Maurer, C. McMillen, D. Abraham, and M. Blum. recaptcha: Human-based character recognition via web security measures. *Science*, 321(5895):1465–1468, 2008.

[Wallach *et al.*, 2009] H. Wallach, D. Mimno, and A. McCallum. Rethinking lda: Why priors matter. *Advances in Neural Information Processing Systems*, 22:1973–1981, 2009.

[Wallach, 2006] H.M. Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM, 2006.

[Wang *et al.*, 2007] X. Wang, A. McCallum, and X. Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 697–702. Ieee, 2007.

[Wang *et al.*, 2009] Y. Wang, H. Bai, M. Stanton, W.Y. Chen, and E. Chang. Plda: Parallel latent dirichlet allocation for large-scale applications. *Algorithmic Aspects in Information and Management*, pages 301–314, 2009.

[Wang *et al.*, 2011] C. Wang, J. Paisley, and D.M. Blei. Online variational inference for the hierarchical dirichlet process. In *Proc. AISTATS*, 2011.

[Weiss *et al.*, 2009] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. *Advances in neural information processing systems*, 21:1753–1760, 2009.

[Weld *et al.*, 2008] D.S. Weld, F. Wu, E. Adar, S. Amershi, J. Fogarty, R. Hoffmann, K. Patel, and M. Skinner. Intelligence in wikipedia. In *Twenty-Third Conference on Artificial Intelligence (AAAI'08)*, 2008.

[Winograd, 1980] T. Winograd. What does it mean to understand language? *Cognitive science*, 4(3):209–241, 1980.

[Xiang and Hong, 2009] Guang Xiang and Jason I. Hong. A hybrid phish detection approach by identity discovery and keywords retrieval. In *Proceedings of the 18th international conference on World wide web*, WWW '09, pages 571–580, New York, NY, USA, 2009.

[Xu *et al.*, 2010] H. Xu, C. Caramanis, and S. Mannor. Robust pca for high-dimensional data. *Advances*, (June):1–8, 2010.

[Yang and Gopal, 2011] Y. Yang and S. Gopal. Multilabel classification with meta-level features in a learning-to-rank framework. *Machine Learning*, pages 1–22, 2011.

[Yang, 2001] Y. Yang. A study of thresholding strategies for text categorization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 137–145. ACM, 2001.

[Yarowsky, 1995] D. Yarowsky. Unsupervised word sense disambiguation rivaling supervised methods. In *Proceedings of the 33rd annual meeting on Association for Computational Linguistics*, pages 189–196. Association for Computational Linguistics, 1995.

[Zettlemoyer and Collins, 2005] Luke S. Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI*, pages 658–666. AUAI Press, 2005.

[Zhang and Zhou, 2005] Min-Ling Zhang and Zhi-Hua Zhou. A k-nearest neighbor based algorithm for multi-label classification. volume 2, pages 718–721 Vol. 2. The IEEE Computational Intelligence Society, 2005.

[Zhang *et al.*, 2007a] Jun Zhang, Mark S. Ackerman, and Lada Adamic. Expertise networks in online communities: structure and algorithms. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 221–230, New York, NY, USA, 2007.

[Zhang *et al.*, 2007b] Yue Zhang, Jason I. Hong, and Lorrie F. Cranor. Cantina: a content-based approach to detecting phishing web sites. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 639–648, New York, NY, USA, 2007.

[Zhang *et al.*, 2008a] J. Zhang, J. Tang, L. Liu, and J. Li. A mixture model for expert finding. *Advances in Knowledge Discovery and Data Mining*, pages 466–478, 2008.

[Zhang *et al.*, 2008b] Jian Zhang, Phillip Porras, and Johannes Ullrich. Highly predictive blacklisting. In *Proceedings of the 17th conference on Security symposium*, SS'08, pages 107–122, Berkeley, CA, USA, 2008.

[Zhu and Wu, 2004] X. Zhu and X. Wu. Class noise vs. attribute noise: A quantitative study. *Artificial Intelligence Review*, 22(3):177–210, 2004.

Appendix A: Concept Annotation Instructions

The following is the detailed user instructions provided to the Mechanical Turk users in the concept annotation task.

Instructions: Here are some tips for you to do better in this task:

- Try to cover the entire comment information with your tags. We prefer short informative tags.
  - Comment:"Avoid! Redirects to other websites."
  - Bad tag: "WARNING TO OTHERS FOR NOT OPENING"
  - Good tag: "redirects, avoid"
  - Comment:"Someone must have seen the word 'scams' in the URL and rated down strictly based on that. The site is obviously helpful and legit."
  - Bad tag: "scam"
  - Good tag: "miscategoried, word scam in URL, helpful, legit"
- Look for words that are more expressive/general/abstract than the words explicitly mentioned (e.g., "File-sharing" when music download is mentioned)
- Ignore information in the comments that are not about the website content or behaviour (e.g., "I have used this website for a long time")
- Provide concepts that are only inferred or based on some common background knowledge (e.g., "fake antivirus" is "scam")
- Do not use full sentences (e.g., don't copy-paste the input!). Also do not pick random words from the sentence. You submission will almost surely be rejected. Look for words and phrase with same semantic content.
- Do not create phrases that too long. Most concepts are 2-3 words but occasionally longer concepts are fine.
- Do not include information that is not directly mentioned in or inferred by the comments. We have provided optional textboxes for additional information and notes.

Improving Tags: When choosing tags you can think of phrases that complete these sentences:

- This website has/contains _____ or This website is _____ (the website content)
- Using this website you can _____ or This website does _____ (the website service offered)
- The users of this website should _____ or People running this website are _____ (advice for the other users)

More Examples:

- Great site for watching original flash movies and games. Adult themes are present throughout.
  Category: `Gaming`
  Positive aspects: `movies, flash games`
  Other aspects: `adult content`
- very nice site with all kinds of art
  Category: `Art`
  Positive aspects: `art`
- don't listen to others who say it has viruses. the downloaded files might though
  Category: `File download`
  Positive aspects: `no virus`
- It infects you with lots of tracking cookies and spyware
  Category: `Unknown`
  Negative aspects: `tracking cookies, spyware`
- Good source for free fonts.
  Category: `Font Download`
  Positive aspects: `free font`