Chapter 13

Learning to Rank Biomedical Documents with only Positive and Unlabeled Examples: A Case Study

Mingzhu Zhu

Information Systems Department, New Jersey Institute of Technology
University Heights, Newark, NJ 07102, USA
mz59@njit.edu

Yi-Fang Brook Wu

Information Systems Department, New Jersey Institute of Technology
University Heights, Newark, NJ 07102, USA
wu@njit.edu

Meghana Samir Vasavada

Bioinformatics Program, New Jersey Institute of Technology
University Heights, Newark, NJ 07102, USA
mv55@njit.edu

Jason T. L. Wang

Bioinformatics Program and Computer Science Department, New Jersey Institute of Technology
University Heights, Newark, NJ 07102, USA
wangj@njit.edu

In the text mining field, obtaining training data requires human experts' labeling efforts, which is often time consuming and expensive. Supervised learning with only a small number of positive examples and a large amount of unlabeled data, which is easy to get, has attracted booming interests in the field. A recently proposed relabeling method, which assumes unlabeled data as negative data for text classification, has been shown successful in identifying relevant biomedical documents on a specific topic. However, it's not known whether and how feature selection affects the performance of the method. In addition, no extensive research has been conducted to evaluate how the performance of the method changes when the proportion of positive

examples in the unlabeled dataset varies. Following the relabeling method, we train Support Vector Machines using positive and unlabeled data to rank incoming documents based on their probability values of being predicted as positive. Using an RNA-protein binding (RNAPB) dataset collected from PubMed, we conduct a series of experiments to evaluate the performance of the proposed text ranking algorithm. Our experimental results show that 1) feature selection is helpful in improving the performance of the algorithm; 2) the increase of the proportion of the positive examples in the unlabeled dataset decreases the performance of the algorithm, but when no reliable negative training data is available, the classifiers, built based on the unlabeled data that contains a small proportion of positive examples, may have comparable performance with the classifiers built based on pure negative data.

1. Introduction

Text mining tools aim to extract information of interest to a user effectively and efficiently. The literature, both on-line and off-line, is increasing at a considerable rate, which makes it almost impossible for a researcher to keep up-to-date with all the relevant literature manually, even on specialized topics [1]. To interpret the large-scale data sets that are being generated from diverse disciplines, it's necessary for researchers to expand their research fields beyond their core realm. Literature mining tools become essential for researchers to get access to the large amount of documents in multidisciplinary fields.

Take bioinformatics as an example. For the average user, the most frequently used literature mining or information retrieval (IR) tools are keyword-based search engines like PubMed. However, it is hard to use IR tools to get all the relevant literature on a specific topic, which is referred to as topic search here. For instance, a junior researcher or a student may hope to get all the published articles that are related to single-nucleotide polymorphism (SNP), which is a very important topic in genetics. It might seem that a keyword search using "single-nucleotide polymorphism" can meet the requirement of this task, but keyword-search is tedious and time consuming in that we need to try different keywords many times to identify the documents that we are interested in. For a layman in biology, sometimes it's not easy to find the appropriate

keywords for a specific topic. Moreover, the returned documents from IR tools may be irrelevant even the keywords are constructed by experts. As a result, it is important to develop novel tools for users to solve the information search problems.

Machine learning methods, such as supervised learning, have been widely used in information retrieval to help users locate the information they are interested in [1, 2]. In supervised learning, a large set of labeled training data is required to make sure the trained model has good performance. However, it is often the case that there is not enough reliable labeled data. With the development of the World Wide Web, the volume of data is increasing dramatically, which makes it even more difficult to create labeled data manually.

Recently, supervised learning with positive and unlabeled data (known as PU learning) [2] has attracted substantial interests from researchers in data mining and machine learning fields. It deals with text classification problems where only the positive examples and the unlabeled examples are available. Since it is easy to get unlabeled data, it is important to learn how to take advantage of these unlabeled data. In fact, many researches advocating PU learning have shown that it is beneficial to use unlabeled data with positive data rather than use the positive data only [2]. Since it is easy for a researcher to collect a small number of documents of interest, and get a large number of unlabeled documents, PU learning seems a good alternative to solve the problem of topic search. In this chapter, we explore how to take advantage of PU learning to get as many relevant documents as possible that are related to a specific topic, and rank these documents according to their relevance to the topic.

There are two machine learning paradigms about learning with unlabeled data [3][4][5][6][7]. The first one is called learning from labeled and unlabeled examples (LU learning). The second one is called learning with only positive and unlabeled data (PU learning). In both paradigms, unlabeled data has been shown useful for boosting learning accuracy. The main difference between LU learning and PU learning is that the latter has more restrictions. In LU learning, there is a small set of labeled data, which contain both positive and negative instances, and the unlabeled data are used to augment the available labeled training

examples. However, in PU learning, there is no negative training data. The positive training data are used to separate and identify positive and negative examples in the unlabeled data, and then a model is learned based on the identified positive and negative examples [6][8].

Many techniques about learning with unlabeled data have been proposed. Nigam et al. [3] use a small set of labeled instances and a large set of unlabeled instances to build a classifier. They show that the classifier built based on the labeled and unlabeled documents has better performance than that built based on a small set of labeled documents alone.

When both positive and unlabeled data are available, the positive training data can be used to estimate the positive class conditional probability, $p(x|+)$, and the unlabeled data can be used to estimate $p(x)$. With the prior $p(+)$, which is known or can be estimated using other sources, the negative class conditional probability can be obtained as follows:

$$p(x|-) = \frac{p(x) - p(+)\,p(x|+)}{1 - p(+)} \tag{1}$$

In [4], Denis et al. adopt the conditional probability $p(x|-)$ to perform text classification with Naïve Bayes method.

Another commonly used method in PU learning is called self-training [5]. The basic idea is that a classifier is first trained with a small set of labeled data. Then the classifier is used to classify the unlabeled data. The most confident unlabeled instances and their predicted labels will be used for iterative training.

In [6], Liu et al. adopt an EM algorithm and naïve Bayesian classification method to separate positive and negative examples. They first put some positive examples, called “spies”, in the unlabeled data set. After completing the EM algorithm, they use the probabilistic labels of the spies to determine the likelihood that a document is negative. The final classifier is built based on the reliable negative documents identified from the unlabeled data set.

In [9], Yu et al. propose a mapping-convergence algorithm for PU learning. There are two stages in their algorithm: mapping stage and

convergence stage. In the mapping stage, they perform initial approximation of highly negative examples. In the convergence stage, they iteratively run an internal classifier that maximizes margins to progressively achieve the true boundary of the positive class in the feature space.

In [10][11], Elkan et al. describe an iterative relabeling algorithm, which assumes that the unlabeled training examples are negative to learn a classifier in each of the iterations of the algorithm. The authors show that their algorithm works well in identifying biomedical documents related to proteins [10], but it is not known whether such a method is applicable to other biomedical document data sets.

Following the approach described by Elkan et al., we propose here a new method for ranking documents without negative training data. Unlike Elkan et al.'s approach, which focuses on binary classification of biomedical documents, the proposed method is mainly concerned with ranking these documents, and hence our method can be easily incorporated into a search engine. In addition, we want to investigate the role of feature selection in document ranking, which was not considered in Elkan et al.'s work. Since feature selection is of great importance in dealing with high dimensional data, we conduct experiments to see whether the performance of the proposed method changes when the number of features varies.

If we assume the unlabeled data (U) as negative examples for classifier training, then the positive documents in U, denoted as PU, become noises. The higher proportion of PU in U, the noisier the assumed negative examples are. So we hypothesized that with the increase of the proportion of PU in U, the performance of the trained classifier tends to degrade in terms of ranking testing documents. We conduct extensive experiments to test this hypothesis using the RNAPB data set collected from PubMed, which contains articles related to RNA-protein binding.

Our experimental results on the RNAPB data set show that when a small number of features are selected using the Chi-square statistic (χ^2-statistic) method [12], the performance of the proposed method can be as good as or even better than when no feature selection is used. Another finding from this research is that the performance of the proposed

method tends to degrade with more positive examples being included in the unlabeled training set U. However, the classifier that is built with positive examples and unlabeled data that contains a small proportion of positive examples, can achieve comparable performance with the classifiers that are built with positive and pure negative examples.

2. Background

Our work is closely related to two fields, namely information retrieval and supervised learning. We review some basic concepts in these two fields below.

2.1. *Information Retrieval*

Information retrieval is concerned with obtaining information resources meeting a user's need from a data collection. The user's need or request is usually represented as queries and the data collection may contain structured data or unstructured text documents. Automated information retrieval systems such as search engines are the most widely used tools for people to solve information overload problems [13].

The core of information retrieval is to model how people compare texts and design computer algorithms to accurately perform this comparison. With the development of the World Wide Web, information retrieval involves several tasks and applications, which include text search, multimedia search and other media search. A usual search scenario is that a user submits a query to a search engine, which will return a list of documents ranked based on some criterion. Often, the documents are ranked based on the extent to which they are relevant to the query. Thus, relevance is a fundamental concept in information retrieval. Simply speaking, a relevant document contains the information that the user who submits the query to a search engine is looking for. Because the same concept can be expressed in different words, simply comparing the text of a query with the text of documents to conduct exact match retrieval usually produces very poor results. To address this issue, many information retrieval models have be proposed and tested to see how well they work. An information retrieval model defines the

process of how to match a query with a document, which forms the basis of the ranking algorithms used by today's search engines for ranking search results.

State-of-the-art information retrieval models are usually based on the statistical properties of text rather than the linguistic structure of the text. For example, modern ranking algorithms are typically designed by considering the occurrence frequency of a word rather than whether the word is a noun or a verb. Although some models do incorporate linguistic features, they are proven less effective.

Performance evaluation of search engines is an important subject in information retrieval, as it is necessary to gauge the effectiveness of a search engine. Widely used performance measures include precision and recall. Precision is the proportion of retrieved documents that are relevant, and recall is the proportion of relevant documents that are retrieved. Since most of the information retrieval models produce a ranked output, to summarize the effectiveness of a ranking algorithm, precision and recall values are often calculated and combined into the Mean Average Precision (MAP) measure, whose definition will be given later in this chapter.

2.2. *Supervised Learning*

Supervised learning, also known as classification, is the task of automatically assigning labels to data, such as web pages, articles, or images. It finds many applications including spam detection, sentiment analysis and information retrieval, to name a few. It is analogous to human gaining new knowledge by learning from the past experiences [2]. For a machine, the "past experiences" are encoded in a set of data records.

A data record is also called an example, an instance, a case or a vector. It is described by a set of attributes or features $A = \{A_1, A_2, \ldots, A_n\}$. In addition, each data record has a special target attribute C, which is called the class attribute. The class attribute C has a set of discrete values, i.e. $C = \{C_1, C_2, \ldots, C_m\}$, where $m \geq 2$. A class value is also called a class label. For example, to classify food as "healthy" or "not healthy", there are two class values. When there are two class values or

labels (i.e., $m = 2$), we can refer to one of them as the positive label and the other as the negative label, and the classification is referred to as binary classification. Data with positive labels are called positive data while data with negative labels are called negative data.

In general, supervised learning is a two-step process. In the first step, a function is built from a data set D, which is called the labeled training data set, to relate values of attributes in A to class values in C. The function is also called a classification model, a predictive model or simply a classifier. It can be in any form, e.g., a decision tree, a set of rules, a Bayesian model or a hyperplane. In the second step, the function is used for classification. Here, a testing data set is used to assess the predictive accuracy of the function. The testing data set is randomly selected from a general data set, and is different from the training data set, which means the testing data set is not used to build the function or classifier.

The accuracy of a classifier on a testing data set is defined as the proportion of testing records that are correctly classified by the classifier. For each data record in the testing set, its predicted class value is compared with its true class label. If the accuracy of a classification model is considered acceptable, the model can be used to classify future data whose class values or labels are unknown.

3. Methods

In this chapter, we present a new method for ranking text documents without negative training data by following the iterative relabeling approach proposed by Elkan et al. [10][11]. Our focus here is not about iterative relabeling, but to see whether feature selection matters in each run of model training and prediction. Since feature selection is of great importance in dealing with high dimensional data, we conduct experiments to see whether the performance of our method changes when the number of features varies. We also explore how the number of positive examples in the unlabeled training data affects the performance of the proposed method.

Specifically, we adopt the relabeling process proposed by Liu and Yu [6][9] to learn a classification model to rank the documents in a given

testing set. An SVM model is trained based on some positive examples and unlabeled examples, where the unlabeled examples are assumed as negative data. This model is then run on separate testing data to predict the probability that a document in the testing set is positive. The documents in the testing set are ranked by the probability value returned by the SVM. The performance of our method is evaluated using the Mean Average Precision (MAP) measure, which is a popular measure in the information retrieval field.

We use the Libsvm toolkit [14] to conduct this research. Documents are transferred into vectors after carrying out stop words removal and stemming. The weight of each feature is calculated using the *tf-idf* method [13]. The χ^2-statistic (CHI) method [12][15] is adopted as the feature selection method to select the top *M* features that have the highest Chi-square scores. We evaluate the performance of our method with varying *M* values. We use the features that result in the best performance of our method to conduct experiments to study how the performance of the method changes with the increase of the proportion of the positive examples in the unlabeled training set.

3.1. *Feature Selection*

Feature selection is an important step in developing machine learning based systems. It refers to the process in which a subset of the features in the training set is selected and used for classification. In text mining and ranking, which is the main subject of this chapter, the terms occurring in documents are considered as features. Feature selection here serves two main purposes. First, it mitigates the problem of dimension curse by decreasing the size of the effective vocabulary. Second, feature selection solves overfitting problems by removing noisy features, which may result in the increase of the classification error on testing data. Feature selection is based on an algorithm in which a utility measure for each of the terms to a class is computed and the *M* terms that have the largest values of this measure will be selected. Other terms that have smaller values of the measure will not be used in text ranking.

In a comparative study of feature selection methods in statistical learning for text categorization, Yang and Pedersen [12] evaluated five

feature selection methods including document frequency (DF), information gain (IG), mutual information (MI), χ^2-statistic (CHI) and term strength (TS). The authors found that IG and CHI are the best methods. In this chapter, we use the CHI method for feature selection. We want to identify a subset of features through which a model can be learned to optimize the performance of the proposed text ranking algorithm.

Let A be the number of times a term t and a class c co-occur. Let B be the number of times t occurs without c. Let C be the number of times c occurs without t. Let D be the number of times neither c nor t occurs. N is the total number of documents. We define the term-goodness measure to be:

$$\chi 2(t,c) = \frac{N \times (AD - CB)^2}{(A+C)(B+D)(A+B)(C+D)} \tag{2}$$

The larger the term-goodness measure, the more relevant t to the class c is. In our case, c represents the positive class, and we will select the top M terms, t, that have the largest $\chi^2(t, c)$ values and use these terms for text ranking.

3.2. *Feature Weight Calculation*

In text mining and information retrieval fields, a document is usually represented as a vector, where the weight of each of the features (terms) in the vector is determined using the *tf-idf* method. Here, the term frequency $tf(t, d)$ is defined as the number of times the term t occurs in the document d. The higher value of $tf(t, d)$, the more important t is in d. On the other hand, the inverse document frequency $idf(t)$ indicates how important a term t is in distinguishing the documents in a collection of documents. The inverse document frequency is calculated using the following formula:

$$idf(t) = \log (N/df(t)) \tag{3}$$

where N is the total number of documents in the collection, and $df(t)$ is the document frequency of t, which is defined as the number of

documents containing *t*. The *tf-idf* value of the term *t* is defined as the product of its *tf* and *idf* values, i.e.

$$tf\text{-}idf(t,d) = tf(t,d) \times idf(t). \tag{4}$$

3.3. *Text Ranking Algorithm*

We use Support Vector Machines (SVMs) [14][16] to predict the likelihood that a document in the testing set is positive. The larger the likelihood is, the higher the testing document is ranked. A two-class, or binary, SVM classifier assigns labels to a testing document based on the sign of the decision function

$$f\left(\bar{x}\right) = \sum_i \alpha_i y_i K\left(\bar{x}_i, \bar{x}\right) + b \tag{5}$$

where $\bar{x}$ is the testing document to be classified or ranked, $\bar{x}_i$ are the training documents, $y_i \in \{-1,1\}$ are the class labels, positive or negative, for $\bar{x}_i$, α_i are weights assigned to the training documents during training, K is the kernel function, and b is a bias term. There are several kernel functions available. In this work, we use the radial basis function (RBF) kernel, which is defined as

$$K(\bar{u}, \bar{v}) = e^{-\gamma \|\bar{u} - \bar{v}\|^2} \tag{6}$$

where γ is a user-determined parameter.

In ranking the testing documents, we train a binary SVM using positive and unlabeled data where the unlabeled data are treated as negative data. The trained model is applied to the testing data set to predict the likelihood that a testing document is positive. The larger the likelihood value, the higher rank the testing document receives. We change the proportion of positive documents in the unlabeled training set to see how the proportion affects the performance of the proposed text ranking algorithm.

4. Experiments and Results

4.1. *Data Sets*

We conducted a series of experiments using the RNAPB data set collected from PubMed, which contains articles related to RNA-protein binding. There are 1160 positive documents and 3929 negative documents in the data set. The positive data set consists of biomedical articles that are about RNA-protein binding while the negative data set is comprised of biomedical articles that are irrelevant to RNA-protein binding. All these documents are collected manually through the PubMed search engine, and verified by our collaborators and domain experts [17][18][19][20][21]. Each document here only contains the title and abstract of a PubMed article. Other information of a published article, such as author and publication date, is not included. Table 1 shows the five most frequently occurring terms in the positive and negative data set, respectively.

Table 1. The five most frequently occurring terms in the data

Positive set	protein	binding	complex	interaction	RNA-protein
Negative set	gene	tree	sequence	phylogenetic	species

4.2. *Performance Measure*

Each document in the testing set is ranked based on its probability that it is predicted as being positive. A good ranking means all the relevant (i.e. positive) results are in the top ranked positions. We adopt the MAP measure [22][23] widely used in the information retrieval (IR) field to evaluate the performance of our approach. In ranking the results of a query, MAP represents the mean of the average precision scores of the results. Formally, let L be a ranked list of retrieved documents, and R be the set of relevant documents being retrieved. The MAP value of L is calculated using the following formula:

$$\mathrm{MAP}(L) = \frac{1}{|R|}\sum_{k=1}^{n}\left(p(k)\times rel(k)\right) \tag{7}$$

where $|R|$ is the number of retrieved relevant documents, k is the rank in the list L of retrieved documents, n is the total number of retrieved documents, $p(k)$ is the precision at cut-off k in the list, $rel(k)$ is an indicator function equaling 1 if the document at rank k is a relevant (i.e. positive) document, zero otherwise [24]. The precision at cut-off k, $p(k)$, equals the number of retrieved relevant documents in the top k ranked documents divided by the number of retrieved documents in the top k ranked documents (hence the denominator is k). The final MAP value is the mean of the MAP value of each query. In our experiments, the results ranked by a trained SVM classification model are equivalent to the search results returned by a query. Positive examples are considered as retrieved relevant documents while negative examples are considered as retrieved irrelevant documents. Experimental results on the RNAPB data set show that our approach combining support vector machines and feature selection performs well.

4.3. *Experimental Design*

Let P denote the set of positive examples in the training set, U denote the set of unlabeled examples in the training set, PU denote the set of positive examples in U, and NU denote the set of negative examples in U. Let |P|, |U|, |PU| and |NU| denote the size of P, U, PU and NU respectively. Thus, |U| = |PU| + |NU|.

In each run of our experiments, we randomly sample a subset of documents from the positive data set and negative data set respectively to form our testing data set. These testing documents are removed from the positive and negative data sets. From the remaining positive data set, we randomly sample a subset of |P| examples to form the positive training data, and |PU| examples for the unlabeled training data, with the constraint that there is no overlap between P and PU. We also randomly sample a subset of |NU| examples from the remaining negative data set for the unlabeled training data. Next, an SVM model is learned from the positive training dataset P and unlabeled training dataset U, where the unlabeled training data in U is treated as negative training data. The trained model is then applied to the testing data to predict the probability that a document in the testing set is positive. The documents in the

testing set are ranked based on the probability values. An MAP value is calculated for the document ranking. We carry out 10 runs of the experiments, and the final MAP value is calculated by averaging over the MAP values from the 10 runs.

In the experiments, each document is preprocessed and transferred into a vector. We use Stanford CoreNLP [15], available at http://nlp.stanford.edu/software/corenlp.shtml, for stop words removal, stemming and lemmatization. The CHI method (χ^2-statistic) [12] is adopted for feature selection. The top M features or terms that have the largest CHI scores are selected as features to create the document vectors. We then use the *tf-idf* method [13] to calculate the feature weights in each document vector.

Libsvm [14] is adopted as the supervised learner to perform document ranking. We employ the default parameters and the RBF kernel in Libsvm. Since Libsvm enables one to predict the probability that a document is positive, it is appropriate for our study, in which we rank each document in the testing set based on its probability of being predicted as positive. The larger the probability is, the higher the testing document is ranked.

4.4. *Experimental Results and Analysis*

Our first experiment is to evaluate the impact of the number of features, M, used in the text ranking algorithm on the performance of the algorithm. We fix |P|, |PU| and |NU| at 30. The testing set contains 30 positive examples and 30 negative examples.

Fig. 1 shows the experimental results. When M is less than 5, the more features are used, the better performance the algorithm achieves. The algorithm achieves the best performance when M is 5. When M is greater than 5, some noise features are included, and hence the performance of the algorithm degrades. When the number of features exceeds 500, the effect of increasing features and the effect of noise features counter each other. As a result, the performance of the algorithm remains stable, though the computational time increases with a larger M. In subsequent experiments, we fix M at 5, and use the top 5 features to

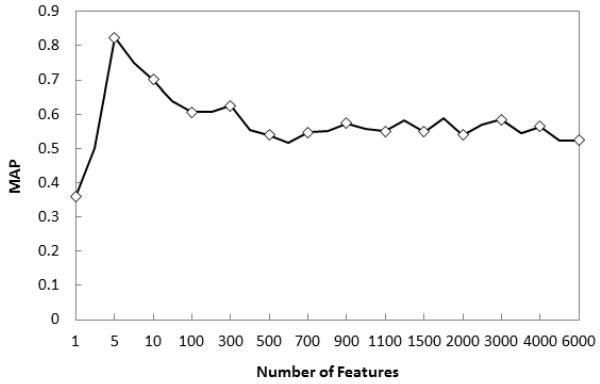

Fig. 1. Impact of the number of selected features on the performance of the text ranking algorithm.

evaluate how the proportion of the positive examples in the unlabeled training set affects the performance of our algorithm.

In our second experiment, we fix |P| and |U| values, and change |PU| and |NU| to make the proportion of PU in U increase from 0 to 100 percent. Fig. 2 summarizes the results, where four observations are made:

1. As expected, with a fixed |U|, the higher proportion of positive examples in U, the worse performance the algorithm achieves.

2. When the proportion of PU in U is less than 10%, the change in MAP values is less than 0.05, suggesting that when no reliable negative data is available, the unlabeled training data with a small proportion of positive examples can be used instead.

3. When the proportion of PU in U is less than 40%, the performance of the algorithm decreases about 15% with the increase of the proportion. On the other hand, when the proportion is between 40% and 80%, the performance of the algorithm decreases more than 20% with the increase of the proportion.

4. When the proportion of PU in U is less than 40%, the change of |U| has little effect on the performance of the algorithm. However, when the proportion is larger than 40%, the increase of |U| (e.g. from 100 to 170) seems helpful in improving the performance of the algorithm. This suggests that with a larger |U|, the impact of the proportion of PU in U is less significant.

It should be pointed out that when the proportion of PU in U is 0, U contains pure negative examples entirely. We observe that the performance of the algorithm in this situation is almost the same as when the proportion of PU in U is less than 10%. This result suggests that with a small proportion of positive examples being in the unlabeled training set U, our algorithm has comparable performance with the classifiers built based on pure negative data. It is also worth noting that the increase of |U| seems helpful in slowing down the speed of the decrease of the performance of our algorithm. For instance, when |U| is 100, the MAP value drops to 0.6 when there are about 50% positive examples included in the unlabeled training set. However, when |U| is 170, the MAP value doesn't drop to 0.6 until there are about 90% positive examples included in the unlabeled training set.

5. Conclusions and Outlook

In this chapter, we present a framework of adopting PU learning for ranking biomedical documents. The core of the framework is a text ranking algorithm that combines methods of feature selection, feature weighting and support vector machines. Using the RNAPB data set, we show experimentally that feature selection is helpful in improving the performance of the text ranking algorithm. A small number of features not only reduce the size of document vectors thus saving computational resources, but also lead to as good as or even better performance of the algorithm than when a larger number of features are used. Our experimental results on the set of biomedical documents also indicate that the performance of the text ranking algorithm tends to decrease when increasingly more positive examples are included in the unlabeled training set. However, when the proportion of the positive examples in the unlabeled training set is less than 10%, the increase of the proportion only changes the performance of the algorithm slightly, suggesting that unlabeled training data with a small proportion of positive examples can be used when no reliable negative examples are available. We have tested our approach on other data sets, and the qualitative conclusion obtained from those experiments remains the same.

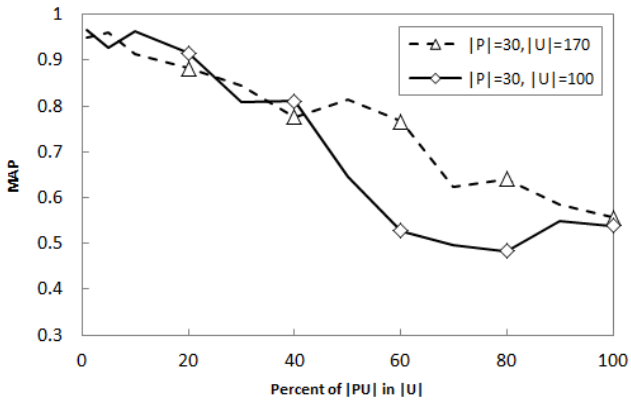

Fig. 2. Impact of |PU|/|U| on the performance of the text ranking algorithm.

In this research, we only adopt the CHI method (χ^2-statistic) to perform feature selection, without using domain-specific knowledge. It is known that the effectiveness of a feature selection process can be significantly enhanced when incorporating domain expertise into the process [17]. However, our goal here is to provide a general framework, which different researchers can use to rank different types of biomedical documents. Domain knowledge related to different types of biomedical documents would not be the same. For example, domain-specific features related to RNA-protein binding would be different from those related to single-nucleotide polymorphism (SNP). Incorporating domain-specific features into our framework would limit the application of this framework.

Our algorithm can be used to build personalized information gathering systems, which aim to locate and rank documents of interest to a particular person. The documents that a user is interested in can be regarded as positive examples, and the results returned by a search engine can be regarded as unlabeled examples. For example, a biologist working on RNA-protein binding collects biomedical documents related to this subject, which are positive examples. Results returned by the PubMed search engine are unlabeled examples. Our algorithm is trained by these positive and unlabeled examples, and the trained algorithm can then be used to rank and gather articles concerning RNA-protein binding

that are published in the future. Thus, the proposed approach lays a foundation for building personalized information gathering systems in the absence of reliable negative training examples.

Our approach takes unlabeled data, which is easy to obtain, and assumes the data to be negative training examples. When there are few positive examples in the unlabeled training set, our approach works well. In practice, however, it is not possible to know the exact proportion of positive examples in the unlabeled training set without carefully examining the data, which is a laborious task. One strategy is to let our algorithm rank the testing documents, and then let a domain expert or user examine some top-ranked documents as well as some bottom-ranked documents. These top-ranked documents are candidates for positive examples and the bottom-ranked documents are candidates for negative examples. Through human verification, we obtain more positive examples and negative examples, and can use these validated data to train the proposed text ranking algorithm again to obtain a better model. This iterative procedure will continue refining the labels of the data at hand, yielding a better ranking list of documents, and hence a better personalized information gathering system.

In general, in supervised learning, both positive and negative examples are needed, but it is often the case that no reliable negative training data is available. Learning a model from positive and unlabeled data is then useful in this case. Moreover, with PU learning, the unlabeled data can be used to create more labeled examples, as described above. We foresee the hybrid approach combining information retrieval, supervised learning, document ranking and human-computer interaction will be the key technology for building future personalized information gathering systems.

References

1. L. J. Jensen, J. Saric, and P. Bork, *Literature mining for the biologist: from information retrieval to biological discovery*, *Nature Reviews Genetics,* 7(2), 119 (2006).
2. B. Liu, *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data.* Springer (2006).

3. K. Nigam, A. McCallum, S. Thrun, and T. Mitchell, *Learning to classify text from labeled and unlabeled documents,* in *Proceedings of the 15th National Conference on Artificial Intelligence*, (1998), p. 792.
4. F. Denis, R. Gilleron, and M. Tommasi, *Text classification from positive and unlabeled examples,* in *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, (2002), p. 1927.
5. G. Haffari and A. Sarkar, *Analysis of semi-supervised learning with the Yarowsky algorithm,* in *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence*, (2007), p. 159.
6. B. Liu, W. S. Lee, P. S. Yu, and X. Li, *Partially supervised classification of text documents,* in *Proceedings of the 19th International Conference on Machine Learning*, (2002), p. 387.
7. W. S. Lee and B. Liu, *Learning with positive and unlabeled examples using weighted logistic regression,* in *Proceedings of the 20th International Conference on Machine Learning*, (2003), p. 448.
8. X. Li and B. Liu, *Learning to classify texts using positive and unlabeled data,* in *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, (2003), p. 587.
9. H. Yu, J. Han, and K. C. Chang, *PEBL: web page classification without negative examples, IEEE Transactions on Knowledge and Data Engineering*, 16(1), 70 (2004).
10. K. Noto, M. H. Saier Jr., and C. Elkan, *Learning to find relevant biological articles without negative training examples,* in *Proceedings of the 21st Australasian Joint Conference on Artificial Intelligence*, (2008), p. 202.
11. C. Elkan and K. Noto, *Learning classifiers from only positive and unlabeled data,* in *Proceedings of the 14th International Conference on Knowledge Discovery and Data Mining*, (2008), p. 213.
12. Y. Yang and J. O. Pedersen, *A comparative study of feature selection in text categorization,* in *Proceedings of the 14th International Conference on Machine Learning*, (1997), p. 412.
13. G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill, Inc. (1986).
14. C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines, ACM Transactions on Intelligent Systems and Technology*, 2(3), 1 (2011).
15. Stanford Core NLP Software. http://nlp.stanford.edu/software/corenlp.shtml (2012).
16. T. Joachims, *Text categorization with support vector machines: learning with many relevant features,* in *Proceedings of the 10th European Conference on Machine Learning*, (1998), p. 137.
17. C. Laing, D. Wen, J. T. L. Wang, and T. Schlick, *Predicting coaxial helical stacking in RNA junctions, Nucleic Acids Research*, 40(2), 487 (2012).

18. R. S. Bindra, J. T. L. Wang, and P. S. Bagga, *Bioinformatics methods for studying microRNA and ARE-mediated regulation of post-transcriptional gene expression, International Journal of Knowledge Discovery in Bioinformatics*, 1(3), 97 (2010).
19. M. Khaladkar, J. Liu, D. Wen, J. T. L. Wang, and B. Tian, *Mining small RNA structure elements in untranslated regions of human and mouse mRNAs using structure-based alignment*, *BMC Genomics*, 9, 189 (2008).
20. J. T. L. Wang, H. Shan, D. Shasha, and W. H. Piel, *Fast structural search in phylogenetic databases, Evolutionary Bioinformatics*, 1, 37 (2005).
21. J. T. L. Wang, T. G. Marr, D. Shasha, B. A. Shapiro, G.-W. Chirn, and T. Y. Lee, *Complementary classification approaches for protein sequences, Protein Engineering*, 9(5), 381 (1996).
22. E. Agichtein, E. Brill, and S. Dumais, *Improving web search ranking by incorporating user behavior information,* in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (2006), p. 19.
23. J. Xu and H. Li, *Adarank: a boosting algorithm for information retrieval,* in *Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (2007), p. 391.
24. A. Turpin and F. Scholer, *User performance versus precision measures for simple search tasks,* in *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, (2006), p. 11.