Thresholding Strategies for Text Classifiers: TREC-2005 Biomedical Triage Task Experiments

Luo Si[1*] **and Tapas Kanungo**[2]

[1]Language Technology Institute
School of Computer Science
Carnegie Mellon University
lsi@cs.cmu.edu

[2]IBM Almaden Research Center
650 Harry Road
kanungo@us.ibm.com

Abstract:

We participated in the triage task of biomedical documents in the TREC genomic track. In this paper we describe the methods we developed for the four triage subtasks. Logistic regression and support vector machine algorithms were first trained to generate ranked lists of test documents. Then a subset of the test documents was identified as positive instances by selecting the top-k documents of the ranked lists. Deciding on the ideal value for k requires a good thresholding strategy. In this paper we first describe two thresholding strategies based on i) logistic regression and ii) support vector machines. In addition to these methods, we describe a thresholding method that combines the outputs from logistic regression and support vector machine by applying a joint thresholding strategy.

1. Task Description

The goal of Mouse Genome Informatics project [2] is to provide structured, coded annotation of different topics from biological literature. Human curators spend a large amount of effort on documents of specific topics to generate annotated information. To reduce the amount of effort put in by human curators, the triage process can be utilized to identify relevant documents for specific topics and thus limit the number of documents sent to human curators for detailed analysis. Four triage subtasks were proposed for the 2005 TREC genomic track: find documents that contain information about i) alleles of mutant phenotypes, ii) embryologic gene expression, iii) gene ontology annotation and iv) tumor biology.

Papers from three journals were used as training and test data for the triage task [2]. In particular, 5,837 papers published in 2002 and their corresponding ground truth labels (binary variables that indicate whether specific documents are relevant and should be sent for detailed annotations) for the four topics were used for training data; 6,043 papers published in 2003 were used as test data. As the training data and test data were sampled from different publication years, the proportion of relevant documents within training and

[*]This work was done when Luo Si visited IBM Almaden Research Lab in summer, 2005.

Subtasks	**AP/Percentage on Training Data**	**AP/Percentage on Test Data**	**Assigned U_r**
Allele	338/5.8%	332/5.5%	17
Expression	81/1.4%	105/1.7%	64
Gene Ontology	462/7.9%	518/8.6%	11
Tumor	36/0.6%	20/0.3%	231

Table 1. The AP number and the percentage value of relevant documents as well as the assigned U_r values for four subtasks as Allele, Expression, Gene Ontology and Tumor. (The total number of training documents is 5,837 and the total number of test documents is 6,043)

test data were different.

The evaluation metric used for triage task was the *normalized utility measure*, which combines the utility/loss of retrieving a relevant document and retrieving a nonrelevant document, and is defined as [2]:

$$U_{norm} = \frac{(u_r * TP) + (u_{nr} * FP)}{(u_r * AP)} \qquad (1)$$

where u_{nr} is the relative utility/loss of retrieving a nonrelevant document and it is set to -1 for all subtasks; u_r is the relative utility of retrieving a relevant document; TP (true positive) denotes the number of retrieved relevant documents; FP (false positive) denotes the number of retrieved nonrelevant documents respectively; and AP (all positive) denotes the total number of relevant documents. The four subtasks have different AP for training and test data and have been associated with different u_r as shown in Table 1.

2. Algorithm Description

The triage task can be seen as a text categorization problem. Text categorization algorithms first extract useful features from text data. Then statistical models are built from training data and associated ground truth labels. Finally test documents are classified as relevant or not using the estimated models.

2.1 Feature Extraction

Feature extraction is the first step of building any text categorization system. The provided TREC data includes full text descriptions of each document. As the crosswalk files specified the PubMed ID for each document, we used the PubMed[2] search engine to acquire the MEDLINE records for all the documents. One piece of valuable information within MEDLINE records is the human annotated Medical Subject Headings (MeSH) categories. MeSH ontology is organized into a tree structure with 15 top level categories such as A (anatomy), B (organisms) etc, while each of them is in turn divided into many subcategories. The information within MeSH ontology has been shown to be very helpful for biomedical triage task in TREC 2004 [2]. In summary, we used the following features in this work:

[2] http://www.ncbi.nlm.nih.gov/entrez/query.fcgi

- Title Text: Text extracted between (<atl>) tags in the XML file
- Abstract Text: Text extracted between (<abs>) tags in the XML file
- Full Text: Text extracted between (<bdy>) tags in the XML file
- MeSH Text Word: Text keywords extracted from MeSH categories. In order to distinguish these words from regular text words, all MeSH keywords were associated with prefix "MH_" (e.g., MH_Diseases).
- MeSH Hierarchy Category ID: To associate MeSH category IDs at different levels, each annoated MeSH category, and all their ancestors, were treated as separate features and used to represent the document. The features were the IDs and not the MeSH category text words. (e.g., C04.928 for "Tumor Virus Infections" and its ancestor C04 for "Neoplasms").

The above features were extracted from the text data in the full text descriptions of the articles in XML format, and MEDLINE records in MEDLINE format. The XML and MEDLINE tags were deleted. Next, text preprocessing was done to remove stopwords, and stemming and case-folding was applied to reduce the number of terms. Finally, the utility BuildIndex within Lemur[3] information retrieval toolkit was used to build an index of terms using the extracted features.

It is more convenient to represent data as vectors of numeric feature values for building statistical learning models. As TF.IDF (terms frequency times inverse term frequency) has been demonstrated to be effective for other text categorization and information retrieval tasks, it was used to represent the features in this work. Specifically, the weight of each feature is represented as:

$$\text{TF.IDF} = (1+\log(\text{tf})) * \log \frac{(\text{N}+1)}{(\text{idf}+1)} \qquad (2)$$

where tf represents the feature frequency within the document, N, which is 5,837, is the number of documents within training set, and idf is the number of training documents that contain the feature in consideration. After the weights have been calculated for the features, they were normalized to make the vector has Euclidean norm as 1.0. This form of TF.IDF representation was also used in previous research [1].

2.2 Statistical Learning Methods

There has been considerable previous research on the application of statistical learning methods to text categorization tasks. In this work, we applied two state-of-the-art methods --- logistic regression and support vector machine --- to the TREC triage task.

Logistic regression method uses an exponential model to estimate the probability that a document belongs to a specific topic as follows [9]:

$$P(y_i = 1 | \vec{\beta}, \vec{d_i}) = \frac{\exp(\sum_j \beta_j f_{ij})}{1+\exp(\sum_j \beta_j f_{ij})} \qquad (3)$$

where $\{f_{ij}\}$ is the feature representation of ith document; $\{\beta_j\}$ is the set of model

[3] http://www.cs.cmu.edu/~lemur

parameters; and y_i=1 indicates that the ith document is relevant to the topic.

The model parameters are estimated by maximizing the log-likelihood of the posterior probability of the model parameters. Specifically, a Laplace distribution is used to model the prior distribution of model parameters and the training optimization problem is:

$$\beta^* = \underset{\beta}{\operatorname{argmax}} \left(\sum_i \log(P(y_i \mid \vec{\beta}, \vec{d_i})) - \frac{\|\vec{\beta}\|_1}{V} \right) \tag{4}$$

where V is the parameter to adjust the weight of the Laplace prior distribution, which is set by cross validation on the training data.

Support vector machine (SVM) is another statistical learning method for text categorization [3,8]. The basic idea is to recognize positive and negative data points accurately while maximizing the margin between the two sets of data points. An SVM with a linear kernel can be expressed as a solution to an optimization problem as follows:

$$\begin{aligned} &\{\vec{w^*}, b) = \underset{w}{\operatorname{argmax}} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \\ &\text{subject to: } y_i(\vec{d_i} \bullet \vec{w} + b) - 1 + \xi_i \geq 0 \quad \forall \xi_i \geq 0 \end{aligned} \tag{5}$$

where $\vec{w^*}, b$ are the SVM model parameters; C controls the trade off between classification accuracy and margin. The output for a test data point is calculated as:

$$y_t = \operatorname{sign}(\vec{d_t} \bullet \vec{w} + b) \tag{6}$$

Kernelized SVM uses transforms to map feature vectors to a vector space of higher dimension and classifies data points with hyperplanes in the higher dimensional space.

2.3 Thresholding Strategies

After estimating optimal model parameters of statistical learning models on training data, these models were used to generate output scores of test documents. Furthermore, a subset of the test documents was identified as relevant documents and the other documents were discarded. The last step should be conducted to maximize the utility measure defined in Equation 1, which requires a thresholding strategy that can select a subset of documents from the output results of statistical learning methods for optimal utility value.

Several thresholding strategies have been proposed and studied in text categorization literature. Researchers [7] have proposed score-based, rank-based and proportion-based thresholding strategies. However, it can be seen from Table 1 that the percentage of relevant documents in the training and test data is not very consistent. One possible explanation is that training data is not very representative of test data as these two sets of documents were published in different years. This observation indicates that ranked-

based and proportion-based thresholding strategies may not fit the triage task as the estimated rank threshold or proportion threshold on training data would not be good choices for the test data. Therefore, different types of score-based thresholding strategies have been utilized in the work.

Logistic regression model generates output scores that are probabilities of relevance for test documents. If we assume that estimated logistic model provides accurate probabilities of relevance, it can be shown [4] that the optimal score threshold is $1/(u_r+1)$. We call this threshold LR-D-Thre, which stands for the analytically derived threshold of logistic regression. However, the estimation of logistic model generally suffers from many problems such as limited amount of training data and inconsistency between training and test data. This indicates one of the disadvantages of using LR-D-Thre. Therefore, an alternative threshold as LR-CV-Thre was proposed that maximizes the utility value on the hold out set of training data. Different values of LR-CV-Thre were set through cross validation for the four subtasks.

In order to better reflect the higher utility value of retrieving one relevant document than discarding one nonrelevant document using SVM, researchers have proposed methods [5] to adjust the weights associated with training errors on positive data points and negative data points by using different values for C in Equation (4). However, this method still does not explicitly optimize for the goal of utility measures as shown in Equation (1). Therefore, we calculated score-based thresholds in our SVM-CV-Thre cross-validation method to explicitly optimize the utility measure.

Can one improve the accuracy of algorithms for triage task by combining the outputs from both logistic regression and support vector machine methods? One approach to address this question is to build a Meta classifier that combines the outputs by logistic regression and SVM methods. This approach was similar to the Stacking method [6] used in statistics community. However, our attempts at using a Meta classifier based on logistic regression did not yield satisfactory results. One possible reason is that Meta classifier approach could be causing more overfitting than the single-level classifiers. This is a serious problem as first level classifiers have already had many parameters associated with large number of features. We then experimented with a simple approach that is less sensitive to overfitting than the Meta classifier approach. In this method the outputs of the logistic regression method and the SVM method were used as sanity checks for each other. When a set of document were first selected through LR-CV-Thres method, these documents were further filtered by requiring their SVM scores to be higher than a threshold tuned on the outputs of SVM to generate better utility value. This method is called LR-SVM-CV-Thres. On the other side, another method first generated results from SVM-CV-Thres and then verified the results based outputs of logistic regression. The later method is called SVM-LR-Thres.

3. Experiment Results

The Bayesian Binary Regression[4] toolkit and SVM light toolkit [3] were used in this work to estimate logistic regression model and support vector machine model respectively. For logistic regression, the values for V in Equation (4) for the four tasks

[4] http://www.stat.rutgers.edu/~madigan/BBR/

Subtasks	LR-D	LR-CV	SVM-CV	LR-SVM-CV	LR-SVM-CV
Allele	0.055	0.03	-0.3	0.03/-0.45	-0.3/0.03
Expression	0.015	0.006	-0.75	0.006/-1	-0.75/0.005
Gene Ontology	0.083	0.04	-0.8	0.04/-1.1	-0.8/0.038
Tumor	0.004	0.004	0.035	0.004/-0.1	0.035/0.003

Table 2. The threshold values for different thresholding strategies. For LR-SVM-CV-Thres (SVM-LR-CV-Thres), the first threshold value is for logistic regression (SVM) and the second is for SVM (logistic regression).

Subtasks	LR-D	LR-CV	SVM-CV	LR-SVM-CV	SVM-LR-CV	Median Results	Best Results
Allele	0.849	0.859	0.833	**0.865**	0.845	0.779	0.871
Expression	0.849	0.828	0.816	**0.829**	0.825	0.655	0.871
Gene Ontology	0.547	**0.558**	0.544	0.559	0.548	0.458	0.587
Tumor	0.889	0.889	**0.941**	0.905	0.947	0.761	0.943

Table 3. The utility values by different statistical learning method with different thresholding strategies and also the median and best results of all submitted official TREC runs. The best results of our submitted official runs are shown in bold font.

were estimated by cross validation. The estimated values were: 60 (Allele), 40 (Expression), 25 (Gene Ontology) and 25 (Tumor). For support vector machine, the polynomial kernel with degree 3 was used. The C values in Equation (5) for the four tasks were set by across validation: 0.0055 (Allele), 0.0032 (Expression), 0.0125 (Gene Ontology) and 0.006 (Tumor). The threshold values of different thresholding strategies were derived or estimated by cross validation on training data and are shown in Table 2.

The results on test data were generated by logistic regression method and support vector machine method with different thresholding strategies. The details are shown in Table 3. It can be seen from the results that the performance of support vector machine (i.e., SVM-CV) is at the same level as that of logistic regression model (i.e., LR-D and LR-CV). This demonstrates that with careful estimation of model parameters and good thresholding strategy, SVM can provide results comparable to that of the logistic regression model. This is a new observation since previous research did not generate comparable results for SVM and logistic regression methods on the triage task of TREC 2004 [2].

For the two thresholding strategies used with the logistic regression model, the LR-CV-Thre method is better than the analytically derived thresholding method, LR-D-Thre, for two subtasks while LR-D-Thre is better than LR-CV-Thre for the one subtask. They used the same thresholding value for the Tumor subtask.

Moreover, the results in Table 3 provide some positive evidence for combining results from logistic regression method and SVM method. Although the improvement is small, both LR-SVM-CV and SVM-LR-CV methods consistently outperform the baseline methods LR-CV and SVM-CV respectively.

4. Conclusion

In this paper we describe the methods we have developed for the triage task of TREC 2005 genomic task. Statistical methods -- logistic regression and support vector machine -- were utilized to build classifiers. We studied different thresholding strategies for

generating optimal utility on test data. The empirical results show that with good estimation of model parameters and appropriate thresholding strategy, SVM method can achieve results comparable to the logistic regression method. Furthermore, the combination of results from the two methods generates consistent improvement over results from individual methods.

Acknowledgement

We thank Jamie Callan for helpful discussions regarding this work. The research presented in this paper is partially supported by an ARDA grant under Phase II of the AQUAINT program. Any opinions, findings, conclusions, or recommendations expressed in this paper are the authors', and do not necessarily reflect those of the sponsor.

Reference:

1. Dayanik. A., Fradkin. D., Genkin. A., Kantor. P., Lewis. D. D., Madigan. D. and Menkov, V. (2004). DIMACS at the TREC 2004 Genomics Track.
2. Hersh W, Bhupatiraju RT, Ross L, Johnson P, Cohen AM, Kraemer DF. (2004). TREC 2004 genomics track overview, The Thirteenth Text Retrieval Conference.
3. Joachims, T. (1999). Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, MIT Press.
4. Lewis D.D. Evaluating and optimizing autonomous text classification systems. (2005). In Proceedings of ACM SIGIR Conference.
5. Fujita S. (2004). Revisiting Again Document Length Hypotheses, TREC 2004 Genomics Track Experiments at Patolis, Thirteenth Text Retrieval Conference.
6. Wolpert D. H. (1992). Stacked generalization, Neural Networks, v.5 n.2, p.241-259.
7. Yang. Y. M. (2001). A Study on Thresholding Strategies for Text Categorization. In Proceedings of ACM SIGIR Conference.
8. Yang. Y. M. and Liu. X. (1999). A Re-Examination of Text Categorization Methods. In Proceedings of ACM SIGIR Conference.
9. Zhang. T. (2001). Text Categorization Based on Regularized Linear Classification Methods. Information Retrieval: 4(1): 5-31.