TREC 2005 Enterprise Track Results from Drexel

Weizhong Zhu[1], Min Song[2], and Robert B. Allen[1]

[1] College of Information Science and Technology
Drexel University
Philadelphia, PA 19104

[2] Department of Computer and Information Sciences
Temple University
Philadelphia, PA, 19122

1 Discussion Topic Search

The primary goal of Discussion Search is to identify a discussion about a topic. A secondary goal is to determine whether a given message expresses pro or con arguments with respect to the discussion. We employed a combination of POS-driven query expansion and a text-classification technique from [6]. The results of those previous experiments indicated that the technique best performed in extracting protein-protein interaction pairs from MEDLINE.

The original email corpus was extremely heterogeneous. We first applied the Tidy HTML parser to strip tags and to identify data such as the sender, thread history, and subject of the messages. We then linked messages into threads in two ways. The corpus provides thread index files for email communications. These thread indexes are composed of hieratically structured multiple discussion threads and single thread. For multiple discussion threads, we unified them into a thread document. We also combined single documents when they had the same subject.

1.1 Method

This is a supervised because the system was developed a small set of training data by taking a subset of discussion threads and manually tagging them. We developed a discussion thread classification module that is based on an SVM. Lemur [3] was used for the backend. Inspired by the traditional bag-of-words representation of text documents, we converted the retrieved documents into a bag-of-features through the feature extraction and selection process. Our approach to conversion of retrieved documents into a bag-of-features is that only important phrases and terms are selected by Part-Of-Speech (POS) tagging. With these important phrases, we constructed a vector for each example based on its bag-of-features: the entries/dimensions of the vectors correspond to all distinct features, and the value of each entry is the weight of its corresponding feature. We then used *tf*idf* weighting and all vectors were normalized to unit length. For example, the features are extracted from the following fields of discussion threads: main text, subject headings, and mail headers.

Information gain was used as a feature-selection criterion because it has been shown to work well for other text categorization tasks [4]. Although there are mixed results about the impact of feature selection on SVM performance in text categorization, we have found that aggressive feature selection is helpful to SVM. We believe this is because a large number of features generated by the above feature extraction method are irrelevant or redundant.

1.2 Runs

The Radial Basis function (RBF) is recommended as a good SVM model for text categorization [5]. First, the RBF kernel non-linearly maps samples into a higher dimensional

space, so it can handle the case when the relation between class labels and attributes is nonlinear. Second, the RBF has fewer hyper-parameters which influences the complexity of model selection. And, third, the RBF kernel has fewer computational difficulties.

We accepted the default values for all parameters of LIBSVM except C and γ. The parameter C determines the trade-off between training error and margin, while the parameter γ specifies the cost-factor by which training errors on positive examples outweigh errors on negative examples. Another parameter is the feature selection threshold. Our tactic for parameter tuning is similar to [1]. We trained SVM classifiers with different parameter settings and estimated their performance by leave-one-out cross-validation. LIBSVM can prune away cross-validation folds that do not need to be explicitly executed. The cross-validation procedure minimizes over-fitting.

We employed a "grid-search" on C and γ using cross-validation. Basically pairs of (C,γ) are tried and the one with the best cross-validation accuracy is picked. We found that trying exponentially growing sequences of C and J is a practical for identifying good parameters. Our experiments found that heuristically setting (C,γ) at $(2^3,2^{-5})$ yielded a cross-validation rate of 77.5%.

$$\mathcal{L} = \sum_{i=1}^{\ell} \alpha_i G(x_i, x_i) - \sum_{i=1}^{\ell} \sum_{j=1}^{\ell} \alpha_i \alpha_j G(x_i, x_j)$$

Where *G(xi,xi)* is the Kernel function to avoid computing inner product in high dimensions. Once the SVM models are built with the training data, the new data are classified in the following:

$$0 \leq \alpha_i \leq C \quad i = 1, \ldots, \ell \quad and \quad \sum_{i=1}^{\ell} \alpha_i = 1$$

When the parameters (α^*, b^*) are found by solving the required quadratic optimization on the training set of points, the SVM is ready to be used for classifying new points. Given a new point x, its class membership is sin[$f(\mathbf{x}, \alpha^*, b^*)$], where

$$f(\mathbf{x}, \alpha^*, b^*) = \mathbf{w}^T \mathbf{x} + b^* = \sum_{i=1}^{N} \alpha_i^* y_i \mathbf{x}_i^T \mathbf{x} + b^* = \sum_{i=1}^{N} \alpha_i^* y_i < \mathbf{x}_i, \mathbf{x} > + b^*$$
$$= \sum_{i \in SV} \alpha_i^* y_i \mathbf{x}_i^T \mathbf{x} + b^* = \sum_{i \in SV} \alpha_i^* y_i < \mathbf{x}_i, \mathbf{x} > + b^*$$

Our technique requires a set of parameters to generate a model that performs well on unseen data. These parameters and decisions are either data related or algorithm related. The decision on algorithm-specific variables includes the upper bound for Lagrange, multipliers, and tolerance. The decision on data-specific variables includes the number of features to be selected, the ratio of positive to negative documents in training data, and the sampling strategy for the negative class. The decision variables and the range of values explored are presented in Table 1.

Decision Variables	**Explored Values**

C (upper bound for Lagrange multipliers)	4, 12
Tolerance	0.001
Class Ratio	1:3, 1:10, use all training data
Sampling strategy	Random
Number of features f	f = 1,000, 10,000, 15,000, All

Table 1. Decision variables and values explored.

We submitted five runs with these decision variables: dsdrexel1, dsdrexel2, dsdrexel3, dsdrexel4, and dsdrexel5. For P@10, the best run is dsdrexel5. Table 2 describes the performance comparison of five runs in terms of various measures including average precision (Avg. Precision), precision at rank R (R-prec), reciprocal rank (Recip_rank), and P@10. For reciprocal rank, rank refers to the rank of the first correct answer returned by a system. Note that in our originally submitted files, there was a clerical error that we have corrected in the results reported here.

Run	***Avg. Precision***	***R-prec***	***Recip_rank***	***P@10***
Dsdrexel1	0.146	0.179	0.427	0.264
Dsdrexel2	0.132	0.172	0.439	0.266
Dsdrexel3	0.145	0.176	0.427	0.263
Dsdrexel4	0.168	0.197	0.479	0.298
Dsdrexel5	0.181	0.215	0.499	0.324

Table 2. Results for discussion topics.

2 Expert Search

The Expert Search task of the Enterprise track was designed to match people with W3C working groups based, primarily, on email communications. The topics are names of working groups and the experts are the members of those groups.

We represented each name extracted from corpus with a collection of documents (for instance, all the emails the person had sent) and then to use different information retrieval models to measure the relevance between the collections of documents and the topics. Our experiments applied Pat-tree-based n-gram extraction [9-11] and term re-weighting techniques to the Vector Space (VS) model [7] and Latent Semantic Indexing (LSI) model [8].

2.1 Method

2.1.1 Retrieval Models

The Vector Space Model is a way of representing documents through the words that they contain. Each document is originally broken down into a word frequency table and the table is called a vector. A vocabulary is built from all the words in all documents in the system and each document is represented as a vector based against the vocabulary. In our experiment, the vocabulary includes not only keywords but also 2-gram and 3-gram phrases obtained from Pat-tree n-gram extraction algorithms. For LSI, a technique, Singular Value Decomposition (SVD), is used to decompose a term-document matrix *A* into three separate matrices, a term by concept matrix *B*, a concept by concept matrix *C* and a concept by document matrix *D*.

2.1.1.1 Similarity computation for each person

Given a query *q* in Vector model, for each people *p* who is represented with a collection of emails *d*, a cosine similarity is computed as the score between *q* and *d*, where both *q* and *d* are represented as *tf*idf* weighted term vectors.

In LSI, a query is transformed as a pseudo-document q_d in matrix *D* and it can be represented by:

$$q_d = q^T B_k C_k^{-1} \ ,$$

Where *q* is simply the vector of words in the original user query and *k* is number of dimensions of matrix *C* [12] (*k*=300 applied in this study). Then for each person, who is represented with a set of emails, a *d* in *D*, the cosine similarity is computed between $q_d C_k$ and dC_k.

2.1.1.2 Query Processing and Term Re-weighting Strategies

Each query is expanded with 2-gram and 3-gram phrases in topics and these phrases are signed higher weight. For instance, a query "XML Schema", can be expanded as "XML", "Schema", and "XML Schema". The phrase "XML Schema" will be given a higher weight automatically. The weights for 1-gram keyword, 2-gram phrases and 3-gram phrases are set as 0.1, 1.0, and 1.5 respectively.

2.2 Results

The results of the two official runs submitted, which are listed in Table 3. They show that a traditional Vector model provide betters retrieval performance for this task. Further work could include text classification or developing more precise term re-weighting strategies.

Run	***Avg. Precision***	***R-prec***	***B-pref***	***Recip_rank***	***P@10***
DREXELEXP1 (VS)	0.1262	0.1743	0.3409	0.4365	0.2500
DREXELEXP2 (LSI)	0.0280	0.0511	0.1672	0.2541	0.0620

Table 3. Results for expert search task.

3 Acknowledgement

We thank Jason Proctor for assistance on these tasks and we thank Dr. Rosina Weber for providing support to Jason Proctor.

4 References

1. Lewis, D.D., Applying Support Vector Machines to the TREC-2001 Batch Filtering and Routing Tasks, *Proceedings of the 10th Text Retrieval Conference (TREC)*, NIST, Gaithersburg, MD 2001, 286-292.
2. Fan, R. E., Chen, P.H., and Lin, C.J., Working set selection using the second order information for training SVM. *Technical Report*, Department of Computer Science, National Taiwan University, 2005.
3. Lemur Project, *The Lemur Toolkit for Language Modeling and Information Retrieval*, http://www.lemurproject.org/
4. Joachims, T., Text Categorization with Support Vector Machines: Learning with Many

Relevant Features. *European Conference on Machine Learning (ECML)* 1998.
5. Shin, M. and Goel A., Empirical Data Modeling in Software Engineering Using Radial Basis Functions, *IEEE Transactions on Software Engineering*, 26(6): 567-576, 2000.
6. Song, M., *Robust Knowledge Extraction over Large Unstructured Text Collections.* Dissertation, College of Information Science and Technology, Drexel University, 2005.
7. Raghavan, V. V. and Wong, S. K. M, A critical analysis of vector space model for information retrieval, *Journal of the American Society for Information Science*, 1986, 37 (5), 279-287.
8. Deerwester, S., Dumais, S.T., Landauer, T.K, Furnas, G., and Harshman, R., Indexing by latent semantic indexing, *Journal of the American Society for Information Science*, 1990, 41(6), 391-407.
9. Lee-Feng C., Huang, T.I., and Chien, M.C., Pat-tree-based Keyword Extraction for Chinese Information Retrieval, *Proceedings of SIGIR* ,1997, 50-58.
10. Chien, L-F., Chen, C-L, Incremental Extraction of Domain-Specific Terms from Online Text Collections, *Recent Advances in Computational Terminology*, M.C. L'Homme, C. Jacquemin, Didier, and Bourigault, ed., John Benjamins Publishing Company, 2001, 89-109.
11. Wang, J-H., Teng, J-W., Cheng, P-J., Lu, W.H., and Chien, L-F., Translating Unknown Cross-Lingual Queries in Digital Libraries Using a Web-based Approach, *ACM/IEEE Joint Conference on Digital Libraries* (JCDL), 2004, pp. 108-116.
12. Berry, M.W., Dumais, S.T., and O'Brien, G.W., Using linear algebra for intelligent information retrieval. *SIAM Review*, 1995, 37(4): 573-595.