

Mapping Subsets of Scholarly Information

Paul Ginsparg*, Paul Houle, Thorsten Joachims, and Jae-Hoon Sul
(Cornell University, Ithaca, NY 14853, USA)

Abstract

We illustrate the use of machine learning techniques to analyze, structure, maintain, and evolve a large online corpus of academic literature. An emerging field of research can be identified as part of an existing corpus, permitting the implementation of a more coherent community structure for its practitioners.

1 Background

The arXiv¹ is an automated repository of over 250,000 full-text research articles² in physics and related disciplines, going back over a decade and growing at a rate of 40,000 new submissions per year. It serves over 10 million requests per month [Ginsparg, 2003], including tens of thousands of search queries per day, and over 20 million full text downloads during calendar year '02. It is a significant example of a Web-based service that has changed the practice of research in a major scientific discipline. It provides nearly comprehensive coverage of large areas of physics, and serves as an on-line seminar system for those areas. It also provides a significant resource for model building and algorithmic experiments in mapping scholarly domains. Together with the SLAC SPIRES-HEP database³, it provides a public resource of full-text articles and associated citation tree of many millions of links, with a focused disciplinary coverage, and rich usage data.

In what follows, we will use arXiv data to illustrate how machine learning methods can be used to analyze, structure, maintain, and evolve a large online corpus of academic literature. The specific application will be to train a support vector machine text classifier to extract an emerging research area from a larger-scale resource. The automated detection of such subunits can play an important role in disentangling other sub-networks and associated sub-communities from the global network. Our notion of “mapping” here is in the mathematical sense of associating attributes to objects, rather than in the sense of visualization tools. While the former underlies the latter, we expect the two will increasingly go hand-in-hand (for a recent review, see [Börner et al., 2003]).

*Presenter of invited talk at Arthur M. Sackler Colloquium on “Mapping Knowledge Domains”, 9–11 May 2003, Beckman Center of the National Academy of Sciences, Irvine, CA

¹See <http://arXiv.org/>. For general background, see [Ginsparg, 2001].

²as of mid-Oct 2003

³The Stanford Linear Accelerator Center SPIRES-HEP database has comprehensively catalogued the High Energy Particle Physics (HEP) literature online since 1974, and indexes more than 500,000 high-energy physics related articles including their full citation tree (see [O’Connell, 2002]).

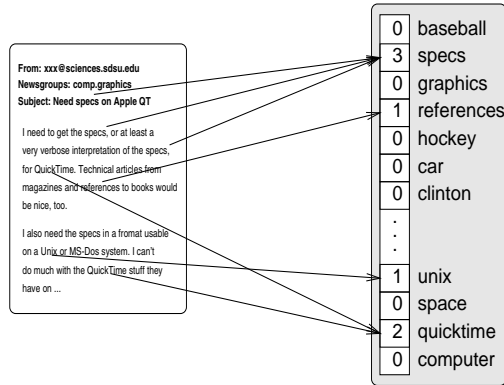


Figure 1: Representing text as a feature vector.

2 Text Classification

The goal of text classification is the automatic assignment of documents to a fixed number of semantic categories. In the “multi-label” setting, each document can be in zero or one or more categories. Efficient automated techniques are essential to avoid tedious and expensive manual category assignment for large document sets. A “knowledge engineering” approach, involving hand-crafting accurate text classification rules, is surprisingly difficult and time-consuming [Hayes and Weinstein, 1990]. We therefore take a machine learning approach to generating text classification rules automatically from examples.

The machine learning approach can be phrased as a supervised learning problem. The learning task is represented by the training sample S_n

$$(\vec{x}_1, y_1), (\vec{x}_2, y_2), \dots, (\vec{x}_n, y_n) \quad (1)$$

of size n documents, where \vec{x}_i represents the document content. In the multi-label setting, each category label is treated as a separate binary classification problem. For each such binary task, $y_i \in \{-1, +1\}$ indicates whether a document belongs to a particular class. The task of the learning algorithm \mathcal{L} is to find a decision rule $h_{\mathcal{L}} : \vec{x} \rightarrow \{-1, +1\}$ based on S_n that classifies new documents \vec{x} as accurately as possible.

Documents need to be transformed into a representation suitable for the learning algorithm and the classification task. Information Retrieval research suggests that words work well as representation units, and that for many tasks their ordering can be ignored without losing too much information. This type of representation is commonly called the “bag-of-words” model, an attribute-value representation of text. Each text document is represented by a vector in the lexicon space, i.e., by a “term frequency” feature vector $\text{TF}(w_i, x)$, with component values equal to the number of times each distinct word w_i in the corpus occurs in the document x . Fig. 1 shows an example feature vector for a particular document.

This basic representation is ordinarily refined in a few ways:

TF×IDF Weighting: Scaling the components of the feature vector with their *inverse document frequency* $\text{IDF}(w_i)$ [Salton and Buckley, 1988] often leads to improved performance. In general, $\text{IDF}(w_i)$ is some decreasing function of the word frequency $\text{DF}(w_i)$,

equal to the number of documents in the corpus which contain the word w_i . For example,

$$\text{IDF}(w_i) = \log \left(\frac{n}{\text{DF}(w_i)} \right) \quad (2)$$

where n is the total number of documents. Intuitively, the inverse document frequency assumes that rarer terms have more significance for classification purposes, and hence gives them greater weight. To compensate for the effect of different document lengths, each document feature vector \vec{x}_i is normalized to unit length: $\|\vec{x}_i\| = 1$.

Stemming: Instead of treating each occurrence form of a word as a different feature, stemming is used to project the different forms of a word onto a single feature, the word stem, by removing inflection information [Porter, 1980]. For example “computes”, “computing”, and “computer” are all mapped to the same stem “comput”. The terms “word” and “word stem” will be used synonymously in the following.

Stopword Removal: For many classification tasks, common words like “the”, “and”, or “he” do not help discriminate between document classes. Stopword removal describes the process of eliminating such words from the document by matching against a pre-defined list of stop-words. We use a standard stoplist of roughly 300 words.

3 Support Vector Machines

SVMs [Vapnik, 1998] were developed by V. Vapnik et al. based on the structural risk minimization principle from statistical learning theory. They have proven to be a highly effective method for learning text classification rules, achieving state-of-the-art performance on a broad range of tasks [Joachims, 1998, Dumais et al., 1998]. Two main advantages of using SVMs for text classification lie in their ability to handle the high dimensional feature spaces arising from the bag-of-words representation. From a statistical perspective, they are robust to overfitting and are well suited for the statistical properties of text. From a computational perspective, they can be trained efficiently despite the large number of features. A detailed overview of the SVM approach to text classification, with more details on the notation used below, is given in [Joachims, 2002].

In their basic form, SVMs learn linear decision rules

$$h(\vec{x}) = \text{sgn}\{\vec{w} \cdot \vec{x} + b\} , \quad (3)$$

described by a weight vector \vec{w} and a threshold b , from an input sample of n training examples $S_n = ((\vec{x}_1, y_1), \dots, (\vec{x}_n, y_n))$, $\vec{x}_i \in \mathbf{R}^N$, $y_i \in \{-1, +1\}$. For a linearly separable S_n , the SVM finds the hyperplane with maximum Euclidean distance to the closest training examples. This distance is called the margin δ , as depicted in Fig. 2. Geometrically, the hyperplane is defined by its normal vector, \vec{w} , and its distance from the origin, $-b$. For non-separable training sets, the amount of training error is measured using slack variables ξ_i .

Computing the position of the hyperplane is equivalent to solving the following convex quadratic optimization problem [Vapnik, 1998]:

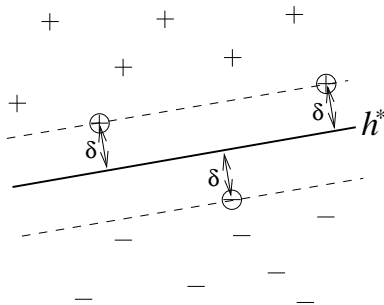


Figure 2: A binary classification problem (+ vs. -) in two dimensions. The hyperplane h^* separates positive and negative training examples with maximum margin δ . The examples closest to the hyperplane are called *support vectors* (marked with circles).

Optimization Problem 1 (SVM (primal))

$$\text{minimize: } V(\vec{w}, b, \vec{\xi}) = \frac{1}{2} \vec{w} \cdot \vec{w} + C \sum_{i=1}^n \xi_i \quad (4)$$

$$\text{subj. to: } \forall_{i=1}^n : y_i [\vec{w} \cdot \vec{x}_i + b] \geq 1 - \xi_i \quad (5)$$

$$\forall_{i=1}^n : \xi_i > 0 \quad (6)$$

The margin of the resulting hyperplane is $\delta = 1/||\vec{w}||$.

The constraints (5) require that all training examples are classified correctly up to some slack ξ_i . If a training example lies on the “wrong” side of the hyperplane, we have the corresponding $\xi_i \geq 1$, and thus $\sum_{i=1}^n \xi_i$ is an upper bound on the number of training errors. The factor C in (4) is a parameter that allows trading off training error vs. model complexity. The optimal value of this parameter depends on the particular classification task and must be chosen via cross-validation or by some other model selection strategy. For text classification, however, the default value of $C = 1/\max_i ||\vec{x}_i||^2 = 1$ has proven to be effective over a large range of tasks [Joachims, 2002].

OP1 has an equivalent dual formulation:

Optimization Problem 2 (SVM (dual))

$$\text{maximize: } W(\vec{\alpha}) = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n y_i y_j \alpha_i \alpha_j (\vec{x}_i \cdot \vec{x}_j) \quad (7)$$

$$\text{subj. to: } \sum_{i=1}^n y_i \alpha_i = 0 \quad (8)$$

$$\forall i \in [1..n] : 0 \leq \alpha_i \leq C \quad (9)$$

From the solution of the dual, the classification rule solution can be constructed as

$$\vec{w} = \sum_{i=1}^n \alpha_i y_i \vec{x}_i \quad \text{and} \quad b = y_{usv} - \vec{w} \cdot \vec{x}_{usv} \quad (10)$$

where (\vec{x}_{usv}, y_{usv}) is some training example with $0 < \alpha_{usv} < C$. For the experiments in this paper, SVM^{Light} [Joachims, 2002] is used for solving the dual optimization problem⁴. More detailed introductions to SVMs can be found in [Burges, 1998, Cristianini and Shawe-Taylor, 2000].

An alternative to the approach here would be to use Latent Semantic Analysis (LSA/LSI) [Landauer and Dumais, 1997] to generate the feature vectors. LSA can potentially give better recall by capturing partial synonymy in the word representation, i.e., by bundling related words into a single feature. The reduction in dimension can also be computationally efficient, facilitating capture of the full text content of papers, including their citation information. On the other hand, the SVM already scales well to a large feature space, and performs well at determining relevant content through suitable choices of weights. On a sufficiently large training set, the SVM might even benefit from access to fine distinctions between words potentially obscured by the LSA bundling. It is thus an open question, well worth further investigation, as to whether LSA applied to full text could improve performance of the SVM in this application, without loss of computational efficiency. Generative models for text classification provide yet another alternative approach, in which each document is viewed as a mixture of topics, as in the statistical inference algorithm for Latent Dirichlet Allocation used in [Griffiths and Steyvers, 2003]. This approach can in principle provide significant additional information about document content, but does not scale well to a large document corpus, so the real-time applications intended here would not yet be computationally feasible in that framework.

4 arXiv Benchmarks

Before using the machine learning framework to identify new subject area content, we first assessed its performance on the existing (author-provided) category classifications. Roughly 180,000 titles and abstracts were fed to model building software which constructed a lexicon of roughly 100,000 distinct words and produced training files containing the TD×IDF document vectors for SVM^{Light}. (While the SVM machinery could easily be used to analyze the full document content, previous experiments [Joachims, 2002] suggest that well-written titles and abstracts provide a highly focused characterization of content at least as effective for our document classification purposes.) The set of support vectors and weight parameters output by SVM^{Light} was converted into a form specific to the linear SVM, eq. (3): a weight vector \vec{w}_c and a threshold b_c , where c is an index over the categories.

As seen in Fig. 3, the success of the SVM in classifying documents improves as the size of a category increases. The SVM is remarkably successful at identifying documents in large ($> 10,000$ documents) categories and less successful on smaller subject areas (< 500 documents). A cutoff was imposed to exclude subject areas with fewer than 100 documents.⁵

In experiments with small categories ($N < 1000$), the SVM consistently chose thresholds that resulted in unacceptably low recall (i.e., missed documents relevant to the category). This is in part because the null hypothesis, that no documents are in the category, describes

⁴SVM^{Light} is available at <http://svmlight.joachims.org/>

⁵Some of the smaller subject areas are known to be less topically focused, so the difficulty in recall, based solely on title/abstract terminology, was expected.

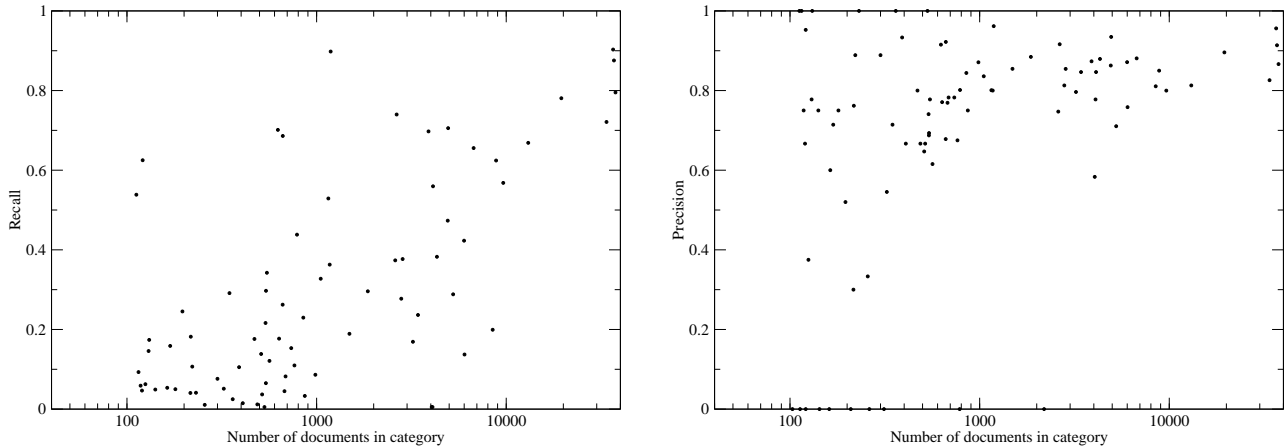


Figure 3: Recall and Precision as functions of category size for 78 arXiv major categories and minor subject classes. 2/3 of the sample was used as a training set and 1/3 as a test set. The four largest categories, each with over 30,000 documents are cond-mat, astro-ph, hep-th, and hep-ph.

the data well for a small category: e.g., if only 1000 of 100,000 documents are in a category, the null hypothesis has a 99% accuracy.⁶ To counteract the bias against small categories, 10% of the training data was used as a validation set to fit a sigmoid probability distribution $1/(1+\exp(Af+B))$ [Platt, 1999]. This converts the dot product $\vec{x}_i \cdot \vec{w}_c$ between document vector and weight vector into a probability $P(i \in c | x_i)$ that document i is a member of category c , given its feature vector x_i . Use of the probability in place of the uncalibrated signed distance from the hyperplane output by the SVM permitted greater levels of recall for small categories.

Other experiments showed that the use of TF×IDF weighting as in eq. (2) improved accuracy consistently over pure TF weighting, so TF×IDF weighting was used in the experiments to follow. We also used a document frequency threshold to exclude rare words from the lexicon, but found little difference in accuracy between using a document occurrence threshold of two and five.⁷ Increasing the weight of title words with respect to abstract words, on the other hand, consistently worsened accuracy, indicating that words in a well-written abstract contain as much or more content classification import as words in the title. Changes in the word tokenization and stemming algorithms did not have a significant impact on overall accuracy. The default value of $C = 1$ in eq. (9) was preferred.

⁶“Accuracy” is the percentage of documents correctly classified. We also employ other common terminology from information retrieval: “precision” is the fraction of those documents retrieved that relevant to a query, and “recall” is the fraction of all relevant documents in the collection that are retrieved.

⁷Words that appeared in fewer than two documents constituted roughly 50% of the lexicon, and those that appeared in fewer than five documents roughly 70%. Ignoring rare and consequently uninformative words hence reduces the computational needs.

5 q-bio Extraction

There has been recent anecdotal evidence of an intellectual trend among physicists towards work in biology, ranging from biomolecules, molecular pathways and networks, gene expression, cellular and multicellular systems to population dynamics and evolution. This work has appeared in separate parts of the archive, particularly under “Soft Condensed Matter”, “Statistical Mechanics”, “Disordered Systems and Neural Networks”, “Biophysics, and “Adaptive and Self-Organizing Systems” (abbreviated `cond-mat.soft`, `cond-mat.stat-mech`, `cond-mat.dis-nn`, `physics.bio-ph`, and `nlin.AO`). A more coherent forum for the exchange of these ideas was requested, under the nomenclature “Quantitative Biology” (abbreviated “q-bio”).

To identify first whether there was indeed a real trend to nurture and amplify, and to create a training set, volunteers were enlisted to identify the q-bio content from the above subject areas in which it was most highly focused. Of 5565 such articles received from Jan 2002 through Mar 2003, 466 (8.4%) were found to have one of the above biological topics as its primary focus. The total number of distinct words in these titles, abstracts, plus author names, was 23558, of which 7984 were above the $DF = 2$ document frequency threshold. (Author names were included in the analysis since they have potential “semantic” content in this context, i.e., are potentially useful for document classification. The SVM algorithm will automatically determine whether or not to use the information by choosing suitable weights.)

A data-cleaning procedure was employed, in which SVM^{Light} was first run with $C = 10$. We recall from eqs. (4) and (9) that larger C penalizes training errors and requires larger α 's to fit the data. Inspecting the “outlier” documents [Guyon et al., 1996] with the largest $|\alpha_i|$ then permitted manual cleaning of the training set. 10 were moved into q-bio, and 15 moved out, for a net movement to 461 q-bio (8.3%) of the 5565 total. Some of the others flagged involved word confusions, e.g., “genetic algorithms” typically involved programming rather than biological applications. Other “q-bio” words with frequent non-biological senses were “epidemic” (used for rumor propagation), “evolution” (used also for dynamics of sandpiles), “survival probabilities”, and extinction. “Scale-free networks” were sometimes used for biological applications, and sometimes not. To help resolve some of these ambiguities, the vocabulary was enlarged to include a list of most frequently used two-word phrases with semantic content different from their constituent words .

With a training set fully representative of the categories in question, it was then possible to run the classifier on the entirety of the same subject area content received from 1992 through Apr 2003, a total of 28,830 documents. The results are shown in Fig. 4. A total of 1649 q-bio documents was identified, and the trend towards an increasing percentage of q-bio activity among these arXiv users is evident: individual authors can be tracked as they migrate into the domain. Visibility of the new domain can be further enhanced by referring submitters in real time, via the automated classifier running behind the submission interface.

Some components of the weight vector generated by the SVM for this training set are shown in Fig. 5. Since the distance of a document to the classifying hyperplane is determined by taking the dot product with the normal vector, its component values can be interpreted as the classifying weight for the associated words. The approach here illustrates one of the

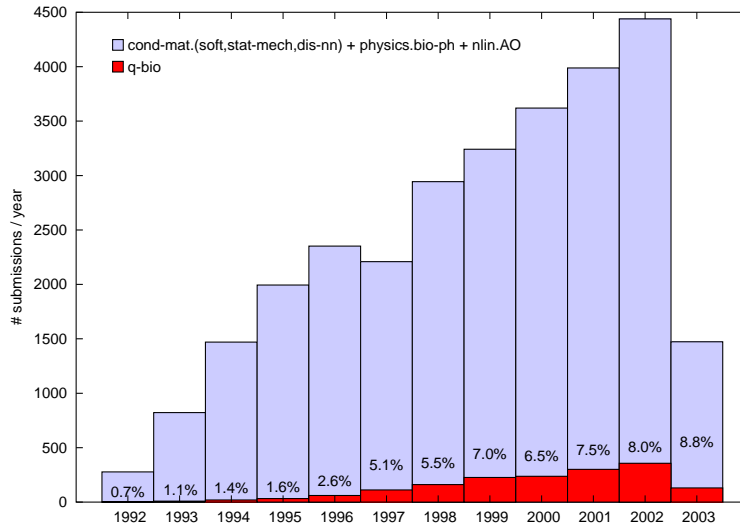


Figure 4: The number of submissions per year from 1992 through April 2003 in particular subsets of arXiv subject areas of cond-mat, physics, and nlin most likely to have “quantitative biology” content. The percentage of q-bio content in these areas grew from roughly 1% to nearly 10% during this timeframe, suggesting a change in intellectual activity among arXiv-using members of these communities.

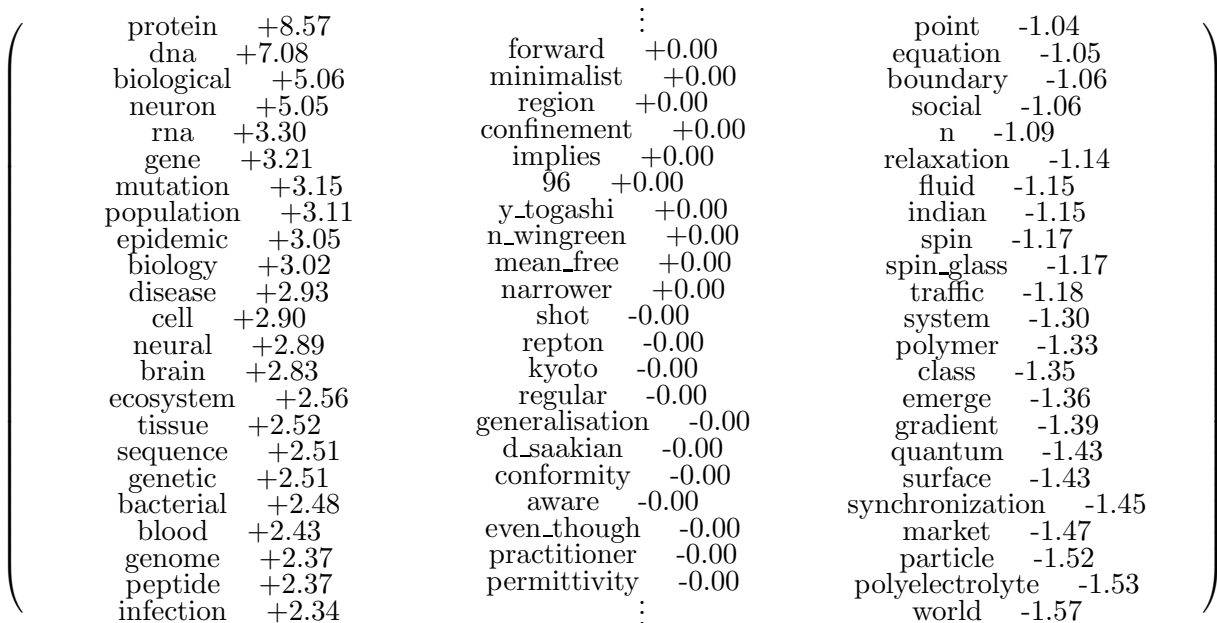


Figure 5: Shown above are the most positive, a few intermediate, and most negative components of the q-bio classifying weight vector.

major lessons of the past decade: the surprising power of simple algorithms operating on large datasets.

While implemented as a passive dissemination system, the arXiv has also played a social engineering role, with active research users developing an affinity to the system and adjusting their behavior accordingly. They scan new submissions on a daily basis, assume others in their field do so and are consequently aware of anything relevant that has appeared there (while anything that doesn't may as well not exist), and use it to stake intellectual priority claims in advance of journal publication. We see further that machine learning tools can characterize a subdomain and thereby help accelerate its growth, via the interaction of an information resource with the social system of its practitioners.⁸

Postscript: The q-bio extraction described in section 5 was not just a thought experiment, but a prelude to an engineering experiment. The new category went on-line in mid-September 2003,⁹ at which time past submitters flagged by the system were asked to confirm the q-bio content of their submissions, and to send future submissions to the new category. The activity levels at the outset corresponded precisely to the predictions of the SVM text classifier, and later began to show indications of growth catalyzed by the public existence of the new category.

References

- [Börner et al., 2003] Börner, K., Chen, C., and Boyack, K. (2003). Visualizing knowledge domains. *Annual Review of Information Science & Technology*, 37:179–255.
- [Burges, 1998] Burges, C. (1998). A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167.
- [Cristianini and Shawe-Taylor, 2000] Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press.
- [Dumais et al., 1998] Dumais, S., Platt, J., Heckerman, D., and Sahami, M. (1998). Inductive learning algorithms and representations for text categorization. In *Proceedings of ACM-CIKM98*.
- [Ginsparg, 2001] Ginsparg, P. (2001). Creating a global knowledge network. In *Electronic Publishing in Science II (proceedings of joint ICSU Press/UNESCO conference, Paris, France)*. ICSU Press. <http://users.ox.ac.uk/~icsuinfo/ginspargfin.htm> (copy at <http://arXiv.org/blurb/pg01unesco.html>).
- [Ginsparg, 2003] Ginsparg, P. (2003). Can peer review be better focused? *Science & Technology Libraries*, pages to appear, copy at <http://arXiv.org/blurb/pg02pr.html>.

⁸Some other implications of these methods, including potential modification of the peer review system, are considered elsewhere [Ginsparg, 2003].

⁹See <http://arXiv.org/new/q-bio.html>

- [Griffiths and Steyvers, 2003] Griffiths, T. and Steyvers, M. (2003). Finding scientific topics. In *Arthur M. Sackler Colloquium on "Mapping Knowledge Domains"*. Proceedings of the National Academy of Sciences, in press.
- [Guyon et al., 1996] Guyon, I., Matic, N., and Vapnik, V. (1996). Discovering informative patterns and data cleaning. In *Advances in Knowledge Discovery and Data Mining*, pages 181–203.
- [Hayes and Weinstein, 1990] Hayes, P. and Weinstein, S. (1990). CONSTRUE/TIS: a system for content-based indexing of a database of news stories. In *Annual Conference on Innovative Applications of AI*.
- [Joachims, 1998] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pages 137 – 142, Berlin. Springer.
- [Joachims, 2002] Joachims, T. (2002). *Learning to Classify Text Using Support Vector Machines – Methods, Theory, and Algorithms*. Kluwer.
- [Landauer and Dumais, 1997] Landauer, T. K. and Dumais, S. T. (1997). A solution to Plato’s problem: The latent semantic analysis theory of acquisition, induction and representation of knowledge. *Psychological Review*, 104:211–240.
- [O’Connell, 2002] O’Connell, H. B. (2002). Physicists thriving with paperless publishing. *HEP Lib. Web.*, 6:3, arXiv:physics/0007040.
- [Platt, 1999] Platt, J. (1999). Probabilities for SV machines. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press.
- [Porter, 1980] Porter, M. (1980). An algorithm for suffix stripping. *Program (Automated Library and Information Systems)*, 14(3):130–137.
- [Salton and Buckley, 1988] Salton, G. and Buckley, C. (1988). Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.
- [Vapnik, 1998] Vapnik, V. (1998). *Statistical Learning Theory*. Wiley, Chichester, GB.