Text Classification by Bootstrapping with Keywords, EM and Shrinkage

# Andrew McCallum' Kamal Nigam mcca11umHjustresearch.com knigamHcs.emu.edu

Just Research School of Computer Science

4616 Henry Street Carnegie Mellon University Pittsburgh, PA 15213 Pittsburgh, PA 15213

Abstract

When applying text classification to com- plex tasks, it is tedious and expensive to hand-label the large amounts of train- ing data necessary for good performance. This paper presents an alternative ap- proach to text classification that requires no labeled documents; instead, it uses a small set of keywords per class, a class hierarchy and a large quantity of easily- obtained unlabeled documents. The key- words are used to assign approximate la- bels to the unlabeled documents by term- matching. These preliminary labels be- come the starting point for a bootstrap- ping process that learns a naive Bayes clas- sifier using Expectation-Maximization and hierarchical shrinkage. When classifying a complex data set of computer science re- search papers into a 70-leaf topic hierar- chy, the keywords alone provide 45a accu- racy. The classifier learned by bootstrap- ping reaches 66% accuracy, a level close to human agreement.

# Introduction

When provided with enough labeled training exam- ples, a variety of text classification algorithms can learn reasonably accurate classifiers (Lewis, 1998; Joachims, 1998; Yang, 1999; Cohen and Singer, 1996). However, when applied to complex domains with many classes, these algorithms often require ex- tremely large training sets to provide useful classifi- cation accuracy. Creating these sets of labeled data is tedious and expensive, since typically they must be labeled by a person. This leads us to consider learning algorithms that do not require such **large** amounts of labeled data.

While labeled data is difficult to obtain, un- *labeled* data is readily available and plentiful. Castelli and Cover (1996) show in a theoretical framework that unlabeled data can indeed be used to improve classification, although it is exponentially less valuable than labeled data. Fortunately, unla- beled data can often be obtained by completely auto- mated methods. Consider the problem of classifying news articles: a short Perl script and a night of au- tomated Internet downloads can fill a hard disk with unlabeled examples of news articles. In contrast, it might take several days of human effort and tedium to label even one thousand of these.

In previous work (Nigam et a1., 1999) it has been shown that with just a small number of labeled docu- ments, text classification error can be reduced by up to 30% when the labeled documents are augmented with a large collection of unlabeled documents.

This paper considers the task of learning text clas- sifiers with no labeled documents at all. Knowledge about the classes of interest is provided in the form of a few keywords per class and a class hierarchy. Keywords are typically generated more quickly and easily than even a small number of labeled docu- ments. Many classification problems naturally come with hierarchically-organized classes. Our algorithm proceeds by using the keywords to generate prelim- inary labels for some documents by term-matching. Then these labels, the hierarchy, and all the unla- beled documents become the input to a bootstrap- ping algorithm that produces a naive Bayes classi- fier.

The bootstrapping algorithm used in this paper combines hierarchical shrinkage and Expectation- Maximization (EM) with unlabeled data. EM is an iterative algorithm for **maximum** likelihood estima- tion in parametric estimation problems with missing data. In our scenario, the class labels of the docu- ments are treated **as missing** data. Here, EM works by first **training a classifier** with only the documents



design jpgjg





**disuibuted sysiem sysmnu**

••• Harawsre & Archltccturt

circuits

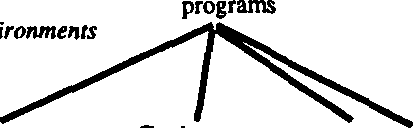
**design**

HCI



text

**6esed**



time

***environments***



dcnotational 8^<8° compiler natural

jqgq gt collection code processing



Machine Plunnlng Knowledge morning planning Rrprescntntlon

learning tempor‹ I knowledge

... Interface Cmpcrstise Multlmedis &lgn colfn6r›raii\*c multinudia



0 fMtWf0d *dWN*

”g'^^ r«nguoge

*teM*

**media**

Figure 1: A subset of Cora’s topic hierarchy. Each node contains its title, and the five most probable words, as calculated by naive Bayes and shrinkage with vertical word redistribution (Hofmann and Puzicha, 1998). Words among the initial keywords for that class are indicated in plain font; others are in italics.

preliminarily-labeled by the keywords, and then uses the classifier to re-assign probabilistically-weighted class labels to all the documents by calculating the expectation of the missing class labels. lt then trains a new classifier using all the documents and iterates. We further improve classification by incorporating *shrinkage, a* statistical technique for improving pa- rameter estimation in the face of sparse data. When classes are provided in a hierarchical relationship, shrinkage is used to estimate new parameters by us- ing a weighted average of the specific (but unreli- able) local class estimates and the more general (but also more reliable) ancestors of the class in the hier- archy. The optimal weights in the average are cal- culated by an EM process that runs simultaneously

with the EM that is re-estimating the class labels. Experimental evaluation of this bootstrapping ap-

proach is performed on a data set of thirty-thousand computer science research papers. A 70-leaf hier- archy of computer science and a few keywords for each class are provided as input. Keyword matching alone provides 45% accuracy. Our bootstrapping al- gorithm uses this as input and outputs a naive Bayes text classifier that achieves 66% accuracy. Inter- estingly, this accuracy approaches estimated human agreement levels of ?2%.

The experimental domain in this paper originates as part of the fto research project, an effort to build domain-specific search engines on the Web with ma- chine learning techniques. Our demonstration sys- tem, Cora, is a search engine over computer science research papers (McCallum et al., 1999). The boot- strapping classification algorithm described in this paper is used in Cora to place research papers into

a Yahoo-like hierarchy specific to computer science. The search engine, including this hierarchy, is pub- licly available at irtutu. *cora. justresearch.* corn.

# Generating Preliminary Labels with Keywords

The first step in the bootstrapping process is to use the keywords to generate preliminary labels for as many of the unlabeled documents as possible. Each class is given just a few keywords. Figure 1 shows examples of the number and type of keywords given in our experimental domain—the human-provided keywords are shown in the nodes in non-italic font. In this paper, we generate preliminary labels from the keywords by term-matching in a rule-list fashion: for each document, we step through the keywords and place the document in the category of the first keyword that matches. Finding enough keywords to obtain broad coverage while simultaneously finding sufficiently specific keywords to obtain high accuracy is very difficult; it requires intimate knowledge of the

**data** and a lot of trial and **error.**

As a result, classification by keyword matching is both an inaccurate and incomplete. Keywords tend to provide high-precision and low-recall; this brittle- ness will leave many documents unlabeled. Some documents will match keywords from the wrong class. In general we expect the low recall of the key- words to be the dominating factor in overall error. In our experimental domain, for example, 59Po of the unlabeled documents do not contain any keywords. Another method of priming bootstrapping with keywords would be to take each set of keywords as a

labeled mini-document containing just a few words. This could then be used as input to any standard learning algorithm. Testing this, and other keyword labeling approaches, is an area of ongoing work.

1. The Bootstrapping Algorithm

The goal of the bootstrapping step is to generate a naive Bayes classifier from the inputs: the (inac- curate and incomplete) preliminary labels, the un- labeled data and the class hierarchy. One straight- forward method would be to simply take the unla- beled documents with preliminary labels, and treat this as labeled data in a standard supervised set- ting. This approach provides only minimal benefit for three reasons: (1) the labels are rather noisy,

(2) the sample of preliminarily-labeled documents is skewed from the regular document distribution (i.e. it includes only documents containing key- words), and (3) data are sparse in comparison to the size of the feature space. Adding the remain- ing unlabeled data and running EM helps counter the first and second of these reasons. Adding hier- archical shrinkage to naive Bayes helps counter the first and third of these reasons. We begin a detailed description of our bootstrapping algorithm with a short overview of standard naive Bayes text classi- fication, then proceed by adding EM to incorporate the unlabeled data, and conclude by explaining hi- erarchical shrinkage. An outline of the entire algo- rithm is presented in Table 1.

* 1. The naive Bayes framework

We build on the framework of multinomial naive Bayes text classification (Lewis, 1998; McCallum and Nigam, 1998). It is useful to think of naive Bayes as estimating the parameters of a probabilis- tic generative model for text documents. In this model, first the class of the document is’ selected. The words of the document are then generated based on the parameters for the class-specific multinomial (i.e. unigram model). Thus, the classifier parame- ters consist of the class prior probabilities and the class-conditioned word probabilities. For formally, each class, cy , has a document frequency relative to

all other classes, written P(cy). For every word

* + - **Inputs:** A collection f of unlabeled documents, a class hierarchy, and a few keywords for each class.
    - Generate preliminary labels for as many of the unla- beled documents as possible by term-matching with the keywords in a rule-list fashion.
    - Initialize all the Aj’s to be uniform along each path from a leaf class to the root of the class hierarchy.
    - Iterate the EM algorithm:
      * (M-step) Build the maximum likelihood multinomial at each node in the hierarchy given the class probability estimates for each document (Equations 1 and 2). Normalize all the Aj’s along each path from a leaf class to the root of the class hierarchy so that they sum to 1.
      * **(E-step)** Calculate the expectation of the class labels of each document using the clas- sifier created in the M-step (Equation 3). In- crement the new Aj’s by attributing each word of held-out data probabilistically to the ances- tors of each class.
    - ‘ **Output:** A naive **Bayes classifier** that **takes** an un- labeled document and predicts a class label.

Table 1: An outline of the bootstrapping algorithm de- scribed in Sections 2 and 3.

this with Laplace smoothing that primes each esti- mate with a count of one to avoid probabilities of zero. Let *N[w , d;)* be the count of the number of times word iut occurs in document *d;,* and define

P(câ *|di)* e {0, 1}, as given by the document’s class label. Then, the estimate of the probability of word try in class cj is:





!\*l + E. -1 g;yp *N( „d,)P(c Id,)*

(1)

The class **prior** probability parameters are set in the same way, where |C| indicates the number of classes:

tub in the vocabulary Y, P(Int |cj) indicates the fre- quency that the classifier expects word try to occur

P(c;)

(2)

in documents in class cy.

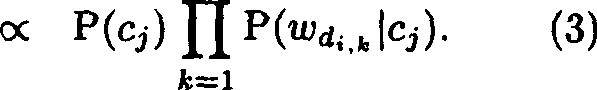


1

In the standard supervised setting, learning of the parameters is accomplished using a set of labeled training documents, P. To estimate the word prob- ability parameters, P(try |cj), we count the frequency with which word rat occurs among all word occur- rences for documents in class cy . We supplement

Given an unlabeled document and a classifier, we determine the probability that the document be- longs in class cj using Bayes’ rule and the naive Bayes assumption—that the words in a document occur independently of each other given the class. If we denote tud;„ to be the Cth word in document *d„* then classification becomes:

P(c;*|di)* o‹ P(c;)P(dj|c;)



Empirically, when given a large number of train- ing documents, naive Bayes does a good job of classifying text documents (Lewis, 1998). More complete presentations of naive Bayes for text classification are provided by Mitchell (1997) and McCallum and Nigam (1998).

* 1. Adding unlabeled data with EM

In the standard supervised setting, each document comes with a label. In our bootstrapping sce- nario, the preliminary keyword labels are both in- complete and inaccurate—the keyword matching leaves many many documents unlabeled, and la- bels some incorrectly. In order to use the entire data set in a naive Bayes classifier, we use the Expectation-Maximization (EM) algorithm to gen- erate probabilistically-weighted class labels for all the documents. This results in classifier parameters that are more likely given all the data.

EM is a class of iterative algorithms for maximum likelihood or maximum a posteriori parameter esti- mation in problems with incomplete data (Dempster et al., 1977). Given a model of data generation, and data with some missing values, EM iteratively uses the current model to estimate the missing values, and then uses the missing value estimates to im- prove the model. Using all the available data, EM will locally maximize the likelihood of the parame- ters and give estimates for the missing values. In our scenario, the class labels of the unlabeled data are the missing values.

In implementation, EM is an iterative two-step process. Initially, the parameter estimates are set in the standard naive Bayes way from just the preliminarily labeled documents. Then we iter- ate the E- and M-steps. The E-step calculates probabilistically-weighted class labels, P(cj|di), for every document using the classifier and Equation 3. The M-step estimates new classifier parameters us- ing all the documents, by Equations 1 and 2, where

P(c t *|di)* ie now continuous, as given by the E-step. We iterate the E- and M-steps until the classifier converges. The initialization step from the prelimi- nary labels identifies each mixture component with

a class and seeds EM so that the local maxima that it finds correspond well to class definitions.

In previous work (Nigam et at., 1999), we have shown this technique significantly increases text

classification accuracy when given limited amounts of labeled data and large amounts of unlabeled data. The expectation here is that EM will both correct and complete the labels for the entire data set.

* 1. Improving sparse data estimates with

**shrinkage**

Even when provided with a large pool of documents, naive Bayes parameter estimation during bootstrap- ping will suffer from sparse data because naive Bayes has so many parameters to estimate (|P| |G| + |U|). Using the provided class hierarchy, we can integrate the statistical technique *shrinkage* into the boot- strapping algorithm to help alleviate the sparse data problem.

Consider trying to estimate the probability of the word “intelligence” in the class NLP. This word should clearly have non-negligible probability there; however, with limited training data we may be un- lucky, and the observed frequency of “intelligence” in NLP may be very far from its true expected value. One level up the hierarchy, however, the Artificial In- telligence class contains many more documents (the union of all the children). There, the probability of the word “intelligence” can be more reliably esti- mated.

Shrinkage calculates new word probability esti- mates for each leaf class by a weighted average of the estimates on the path from the leaf to the root. The technique balances a trade-off between speci- ficity and reliability. Estimates in the leaf are most specific but unreliable; further up the hierarchy es- timates are more reliable but unspecific. We can calculate mixture weights for the averaging that are guaranteed to maximize the likelihood of held-out data with the EM algorithm.

One can think of hierarchical shrinkage as a gener- ative model that is slightly augmented from the one described in Section 3.1. As before, a class (leaf) is selected first. Then, for each word position in the document, an ancestor of the class (including itself) is selected according to the shrinkage weights. Then, the word itself is chosen based on the multinomial word distribution of that ancestor. If each word in the training data were labeled with which ancestor was responsible for generating it, then estimating the mixture weights would be a simple matter of maximum likelihood estimation from the ancestor emission counts. But these ancestor labels are not provided in the training data, and hence we use EM to fill in these missing values. We use the term *uer-* ticnf *EM* to refer to this process that calculates an- cestor mixture weights; we use the term *horizontal EM* to refer to the process of filling in the missing

class (leaf) labels on the unlabeled documents. Both vertical and horisontal EM run concurrently, with interleaved E- and M-steps.

More formally, let (P 1 (wt |cj), ... *, Pk (w c j) j* be word probability estimates, where P' (wt |cj) is the maximum likelihood estimate using training data just in the leaf, P 2( t i cy) is the maximum likeli- hood estimate in the parent using the training data from the union of the parent’s children, P\*\*' (wt |c;) is the estimate at the root using all the training data, and P’(wz *ct )* is the uniform estimate (P\*(w *ct )* =

# Related Work

Other research efforts in text learning have also used bootstrapping approaches. The co-training algo- rithm (Blum and Mitchell, 1998) for classification works in cases where the feature space is separable into naturally redundant and independent parts. For example, web pages can be thought of as the text on the web page, and the collection of text in hyperlink anchors to that page.

A recent paper by Riloff and Jones (1999) boot- straps a dictionary of locations from just a small set

1/ | V|). The interpolation weights among cy ’s “an-

cestors” (which we define to include *ct* itself) are written (A , A2 , ... , A }, where Z — \* y° = 1. The new word probability estimate based on shrinkage,

denoted P(wi |cy), is then

of known locations. Here, their mutual bootstrap algorithm works by iteratively identifying syntactic constructs indicative of known locations, and identi- fying new locations using these indicative constructs. The preliminary labeling by keyword matching used in this paper is similar to the seed collocations

P(w *c:i*

*)* - JtP!(ct |c;) + ... + *7\*Pk (tu/* |cj). (4)

used by Yarowsky (1995). There, in a word sense disambiguation task, a bootstrapping algorithm is

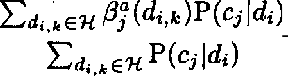
The Aj vectors are calculated by the iterations of EM. In the E-step we calculate for each class cj and each word of unlabeled held out data, 7-I, the probability that the word was generated by the tth ancestor. In the M-step, we normalize the 8um of these expectations to obtain new mixture weights AJ . Without the use of held out data, all the mix- ture weight would concentrate in the leaves.

Specifically, we begin by initializing the A mixture weights for each leaf to a uniform distribution. Let *Qq°(d;,*z) denote the probability that the nth ancestor of cy was used to generate word occurrence *d;, .* The E-step consists of estimating the Q’s:

*§%(d„y)* = (g)



In the M-step, we derive new and guaranteed im- proved weights, A, by summing and normalizing the Q’s:

 (6)

The E- and M-steps iterate until the A’s con- verge. These weights are then used to calculate new shrinkage-based word probability estimates, as in Equation 4. Classification of new test documents is performed just as before (Equation 3), where the Laplace estimates of the word probability estimates are replaced by shrinkage-based estimates.

A more complete description of hierarchical shrinkage for text classification is presented by McCallum et al. (1998).

seeded with some examples of common collocations with the particular sense of some word (e.p. the seed “life” for the biological sense of “plant”).

1. **Experimental Results**

In thla Section, we ptovlde embirlcal evidence that bootstrapping a text classifier from unlabeled data can produce a high-accuracy text classifier. As a test domain, we use computer science research papers. We have created a 70-leaf hierarchy of computer sci- ence topics, part of which is shown in Figure 1. Cre- ating the hierarchy took about 60 minutes, during which we examined conference proceedings, and ex- plored computer science sites on the Web. Select- ing a few keywords associated with each node took about 90 minutes. A test set was created by expert hand-labeling of a random sample of 625 research papers from the 30,682 papers in the Cora archive at the time we began these experiments. Of these, 225 (about one-third) did not fit into any category, and were discarded—resulting in a 400 document test set. Labeling these 400 documents took about six hours. Some of these papers were outside the area of computer science (e.q. astrophysics papers), but most of these were papers that with a more complete hierarchy would be considered computer science pa- **pers.** The class frequencies of the data are not too skewed; on the test set, the most populous class ac- counted for only 7% of the documents.

Each research paper is represented as the words of the title, author, institution, references, and ab- stract. A detailed description of how these seg- ments are automatically extracted is provided else- where (McCallum et at., 1999; Seymore et al., 1999).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Method | @ Lab | @ P-Lab | @ Unlab | Acc |
| i'ieyyoId | — | — | — | 45Po |
| NB | 10fi | — | — | 30Po |
| NB | 399 | — | — | 47Po |
| NB+EM+S | — | 12,657 | 18,025 | 66Po |
| NB | — | 12,657 | — | 47% |
| NB+S | — | 12,657 | — | 63Po |
| Human | — | — | — | 72Po |

Table 2: Classification results with different techniques: keyword **matching, human agreement, naive Bayes** (NB), and naive Bayes combined with hierarchical shrink- age (S), and EM. The classification accuracy (Acc), and the number of **labeled** (Lab), keyword-matched **preliminarily-labeled** (P-Lab), **and** unlabeled (Unlab) documents used by each method are shown.

Words occurring in fewer than five documents and words on a standard stoplist were discarded. No stemming was used. Bootstrapping was performed **using** the algorithm outlined in Table 1.

Table 2 shows classification results with different classification techniques used. The rule-list classifier based on the keywords alone provides 45%. (The 43% of documents in the test set containing no key- words cannot be assigned a class by the rule-list clas- sifier, and are counted as incorrect.) As an inter- esting time comparison, about 100 documents could have been labeled in the time it took to generate the keyword lists. Naive Bayes accuracy with 100 labeled documents is only 30%. With 399 labeled documents (using our test set in a leave-one-out- fashion), naive Bayes reaches 47%. When running the bootstrapping algorithm, 12,657 documents are given preliminary labels by keyword matching. EM and shrinkage incorporate the remaining 18,025 doc- uments, "fix” the preliminary labels and leverage the hierarchy; the resulting accuracy is 66%. As an in- teresting comparison, agreement on the test set be- tween two human experts was 72%.

A few further experiments reveal some of the inner-workings of bootstrapping. If we build a naive Bayes classifier in the standard supervised way from the 12,657 preliminarily labeled documents the clas- sifier gets 47% accuracy. This corresponds to the performance for the first iteration of bootstrapping. Note that this matches the accuracy of traditional naive Bayes with 399 labeled training documents, but that it requires less than a quarter the hu- man labeling effort. If we run bootstrapping with- out the 18,025 documents left unlabeled by keyword matching, accuracy reaches 63%. This indicates that shrinkage and EM on the preliminarily labeled doc- uments is providing substantially more benefit than the **remaining** unlabeled documents.

One explanation for the small impact of the 18,025 documents left unlabeled by keyword matching is that many of these do not fall naturally into the hierarchy. Remember that about one-third of the 30,000 documents fall outside the hierarchy. Most of these will not be given preliminary labels by key- word matching. The presence of these outlier docu- ments skews EM parameter estimation. A more in- clusive computer **science** hierarchy would allow the unlabeled documents to benefit classification more.

However, even without a complete hierarchy, we could use these documents if we could identify these outliers. Some techniques for robust estimation with EM are discussed by McLachlan and Bamford (1988). One specific technique for these text hierarchies is to add extra leaf nodes containing uniform word dis- tributions to each interior node of the hierarchy in order to capture documents not belonging in any of the predefined topic leaves. This should allow EM to perform well even when a large percentage of the documents do not fall into the given classification hierarchy. A similar approach is also planned for re- search in topic detection and tracking (TDT) (Baker et al., 1999). Experimentation with these techniques is an area of ongoing research.

# Conclusions and Future Work

This paper has considered building a text classifier without labeled training documents. In its place, our bootstrapping algorithm uses a large pool of un- labeled documents and class-specific knowledge in the form of a few keywords per class and a class hierarchy. The bootstrapping algorithm combines Expectation-Maximisation and hierarchical shrink- age to correct and complete preliminary labeling provided by keyword matching. Experimental re- sults show that accuracies close to human agreement can be obtained by the bootstrapping algorithm.

In future work we plan to refine our probabilis- tic model to allow for documents to be placed in interior hierarchy nodes, documents to have mul- tiple class assignments, and classes to be modeled with multiple mixture components. We are also in- vestigating principled methods of re-weighting the word features for "semi-supervised” clustering that will provide better discriminative training with un- labeled data.

Acknowledgements

Kamal Nigam was supported in part by the Darpa

**HPKB** program under contract F30602-97-1-02l5.

**References**

D. Baker, T. Hofmann, A. McCallum, and Y. Yang. 1999. A hierarchical probabilistic model for nov- elty detection in text. Technical report, Just Re- search. <http://www.cs.cmu.edu/> mccallum.

A. Blum and T. Mitchell. 1998. Combining labeled

E. Riloff and R. Jones. 1999. Learning dictionaries for information extraction using multi-level boot- strapping. In *AA AI-99.* To appear.

K. Seymore, A. McCallum, and R. Rosenfeld. 1999. Learning hidden Markov model structure for in- formation extraction. In *AA AI-99 Workshop on*



and unlabeled data with co-training. In *OOH T ’98.*

*Machine learning for*

appear.

*Extraction.* To

V. Castelli and T. M. Cover. 1996. The relative value of labeled and unlabeled samples in pat- tern recognition with an unknown mixing param- eter. ***IEEE*** *Transactions on Information Theory,* 42(6):2101—2117.

W. Cohen and Y. Singer. 1996. Context-sensitive learning methods for text categorization. In *SI- GTR ’96.*

A. P. Dempster, N. M. Laird, and D. B. Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B,* 39(1):1-38.

T. Hofmann and J. Puzicha. 1998. Statistical mod- els for co-occurrence data. Technical Report AI Memo 1625, AI Lab, MIT.

T. Joachims. 1998. Text categorization with Sup- port Vector Machines: Learning with many rele- vant features. In *EC!ML-98.*

D. D. Lewis. 1998. Naive (Bayes) at forty: The independence assumption in information retrieval. In *EC!MT-98.*

A. McCallum and K. Nigam. 1998. A comparison of event models for naive Bayes text classification. In *AA AI-98 Workshop* ***on £eorninq*** *for Test C!at- egorization.* Tech. rep. WS-98-05, AAAI Press. [http://www.cs.cmu.edu/-mcca11um.](http://www.cs.cmu.edu/-mcca11um)

A. McCallum, R. Rosenfeld, T. Mitchell, and A. Ng. 1998. Improving text clasification by shrinkage in a hierarchy of classes. In *UMT-98.*

Andrew McCallum, Kamal Nigam, Jason Rennie, and Kristie Seymore. 1999. Using machine learn- ing techniques to build domain-specific search en- gines. In *IJOAI-99.* To appear.

G.J. McLachlan and K.E. Bamford. 1988. 3fizture

*Models.* Marcel Dekker, New York.

T. M. Mitchell. 1997. *Machine* £eorninq. McGraw- Hill, New York.

K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. 1999. **Text** classification from labeled and unla- beled documents using EM. *Machine* £eorninq. To appear.

Y. Yang. 1999. An evaluation of statistical ap- proaches to text categorization. Journal o/ In- *formation Retrieval.* To appear.

D. Yarowsky. 1995. Unsupervised word sense disam- biguation rivaling supervised methods. In *AC!L- 95.*