Cross-training: Learning probabilistic mappings between topics

Sunita Sarawagi
IIT Bombay
sunita@it.iitb.ac.in

Soumen Chakrabarti*
IIT Bombay
soumen@cse.iitb.ac.in

Shantanu Godbole†
IIT Bombay
shantanu@it.iitb.ac.in

ABSTRACT

Classification is a well-established operation in text mining. Given a set of labels A and a set D_A of training documents tagged with these labels, a classifier learns to assign labels to unlabeled test documents. Suppose we also had available a different set of labels B, together with a set of documents D_B marked with labels from B. If A and B have some semantic overlap, can the availability of D_B help us build a better classifier for A, and vice versa? We answer this question in the affirmative by proposing *cross-training*: a new approach to semi-supervised learning in presence of multiple label sets. We give distributional and discriminative algorithms for cross-training and show, through extensive experiments, that cross-training can discover and exploit probabilistic relations between two taxonomies for more accurate classification.

Categories and subject descriptors: I.2.6 [**Artificial intelligence**]: Learning; I.5.2 [**Pattern Recognition**]: Design Methodology - classifier design and evaluation

Keywords: Semi-supervised multi-task learning, Document classification, EM, Support Vector Machines.

1. INTRODUCTION

Document classification is a well-established area of text mining. A document classifier is first *trained* using documents with preassigned *labels* or *classes* picked from a set of labels, which we call the *taxonomy* or *catalog*. Once the classifier is trained, it is offered *test* documents for which it must guess the best label/s. Depending on the application, the label may correspond to a broad topic (e.g., a topic in the Yahoo! directory), a product category, or a user's personal taste in books, CDs, or Web sites.

Support Vector Machines (SVMs) [8], nearest-neighbor classifiers [19], maximum entropy classifiers [14], and naive Bayes (NB) classifiers [13] are some of the commonly used document classifiers.

If all content creators and users agreed on a single catalog of universal labels, text classification could also help tag content with unambiguous semantic annotations. The Web, however, has evolved without central editorship. It is unclear if universal standards will emerge outside specific application segments, and even for those segments, there is a need to consolidate legacy data into content organized as per the agreed-upon standards. These standards, moreover, are far from static.

A few examples will illustrate the current scenario. An e-commerce site which consolidates catalogs of goods and services may want to organize them according to the (still evolving) codes being developed in cooperation between ECCMA and UNSPSC (see `http://www.eccma.org/unspsc/`). Meanwhile, vendors may have their own custom/legacy codes which generally evolve over time.

As another example, consider Yahoo! and Dmoz. Both cover the Web and have evolved to similar taxonomies, but show non-trivial differences. Among other artifacts, *taxonomy inversion* is rampant in the `Regional` categories; what is `Reference.Education.Colleges_and_Universities.Asia.India` in one may be `Regional.Asia.India.Education` in the other; in fact, they sometimes coexist in the same taxonomy! Other relationships are also common: `Dmoz.Recreation.Outdoors.Speleology` overlaps `Yahoo.Recreation.Outdoors.Caving`, but there are important non-overlapping sub-topics.

Given that documents are inherently a conglomeration of concepts, we believe that mappings between content-based taxonomies will be *complex*, *uncertain* and *noisy*. Therefore, text searching, ranking, and mining tools must exploit any available relationships, even probabilistic ones, between diverse meta-data standards.

Our contributions: We introduce a general semi-supervised learning framework called *cross-training* which can exploit knowledge about label assignments in one taxonomy B to make better inferences about label assignments in another taxonomy A. Cross-training generalizes several existing classification algorithms, while also comparing favorably with their accuracy on a host of related applications. Apart from increased classification accuracy, the benefits include a better understanding of probabilistic relationships between taxonomies, and more experience with encoding heterogeneous features for learning algorithms.

We propose two cross-training algorithms. One uses Expectation Maximization (EM) [5]. The other uses Support Vector Machines [16, 7, 8]. Through a detailed experimental study using real-life and semi-synthetic data from Yahoo! and Dmoz, we show that cross-training is decisively better than the best classifier we could induce on A or B alone. Neither of our approaches dominates the other across all data sets. Cross-training also compares favorably with one recent algorithm [1] for mapping taxonomies, as well as an earlier classification algorithm which could exploit a pool of unlabeled documents [15].

Cross-training is related to *multi-task learning* [18, 4], but quite different from co-training [3]. We will discuss these and other related work in §6. The connection between cross-

* Contact author.

† Supported by the Infosys Fellowship Award from Infosys Technologies Limited, India.

SIGKDD '03 August 24-27, 2003, Washington, DC, USA.

training and supervised learning with discrete/categorical attributes is discussed in §7.

Outline: We start with a formal definition of the cross-training problem in §2, and then present the two major classes of cross-training algorithms in §3 and §4. We conduct a detailed experimental evaluation of the accuracy of various algorithms in §5, review related work in §6, and make concluding remarks in §7.

2. PROBLEM SETTING AND TERMINOLOGY

Like most recent text classifiers, our system models each document as a *bag* (multi-set) of words. Term t occurs $n(d,t)$ times in document d. In contrast with prior work, there are *two* sets of class labels A and B. A document can be associated with a *pair* of labels (c_A, c_B) where $c_A \in A$ and $c_B \in B$. We will also denote these labels, for a specific document d, as $c_A(d)$ and $c_B(d)$.

Flat taxonomy assumption: In our setting, a taxonomy is a flat set of class labels. We apologize for continuing to use the term "taxonomy" which connotes hierarchical relations between concepts or topics. Generalizing our approach to that setting is left for future work.

Owing to the dichotomy of labels, training and testing processes must be defined more carefully than in standard document classification. There are two distinct learning scenarios, which we discuss separately.

2.1 Mapping half-labeled documents

The mapping problem can arise in a e-commerce setting where catalogs of goods and services of one site need to be integrated with those of another site. A could represent the target taxonomy, and B the taxonomy used at the data source.

A training document has exactly one of $c_A(d)$ and $c_B(d)$ known; we call such documents **half-labeled**. The system trains on half-labeled documents. During deployment, a new document comes with exactly one of the labels known, and the system has to estimate the missing label.

For benchmarking, we use a test set which is **fully-labeled**, and hide each label in turn, comparing the hidden label with the system's guess. The **accuracy** of the system is the fraction of documents that it assigns to the correct hidden label. Thus, mapping is a *symmetric* scenario.

D_A (respectively, D_B) is the set of documents with A-labels (respectively, B-labels), and $D_A - D_B$ and $D_B - D_A$ are the half-labeled instances available to an algorithm. $D_A \cap D_B$ is used for testing the algorithm.

Validation and tuning: As in earlier work [1], we will (sometimes) assume that some fraction of fully-labeled data can be sampled and made available to the system to help it tune its parameters and validate its models. This is called the **tuneset**. Typically, the available set of fully-labeled documents is partitioned into a tuneset and a test set, the tuneset used to fine-tune the system's models and parameters, and the test set used to evaluate the system. This split is done randomly, many times, and the average accuracy is reported. (The above discussion may hint that the tuneset ought to be a small portion of the fully-labeled data, but in earlier work [1] large tunesets have been used. We report experiments with both choices, to make a fair comparison.)

2.2 Classifying zero-label documents

Several personal bookmark managers [11, 9] need to train document classifiers on bookmarks organized into personal topic directories (say, B) with the intent of mapping subsequently visited pages to those topics. Because bookmarking and annotation takes effort, people bookmark far fewer pages than they visit.

Nigam *et al.* [15] showed that if training data is scarce, a pool of *unlabeled* documents can be used to induce more accurate classifiers. (We discuss their method in §3.1.2.) Unlabeled documents are plentiful and easy to collect. Extending Nigam *et al.*'s work, we note that plentiful *labeled* data is available as well, e.g., from Web topic taxonomies. The catch is that those taxonomies (say, A) may differ substantially from the target taxonomy—exactly the situation we are setting out to address. We wish to evaluate if the additional label data can be exploited to improve our accuracy further.

Evaluation: As with mapping, training documents are half-labeled but one taxonomy, say A, has significantly more documents than B and the goal is to improve the B classifier using A-labeled documents. In contrast, each test document d has only *one* label (say from B) and even that is hidden from the system. The system must guess $c_B(d)$. Accuracy is defined as before. We call this the "zero-label" setting.

3. DISTRIBUTIONAL CROSS-TRAINING

Distributional classifiers fit a *generative* model to the training data, and use this model to predict labels for test cases. E.g., a naive Bayes (NB) classifier posits that a document is generated by first fixing a label by invoking a (typically multinomial) prior distribution on labels, and then creating the document by invoking a term (feature) distribution conditioned on the label just chosen. In the next subsection, we discuss two settings with this generative framework: completely supervised and partially supervised learning of a single label. Then we propose our main algorithms for learning label-pairs.

3.1 Preliminaries

3.1.1 Naive Bayes (NB)

In NB classification for a single taxonomy with label set C,

$$\begin{aligned} \Pr(c|d) &= \frac{\Pr(c,d)}{\Pr(d)} = \frac{\Pr(c)\Pr(d|c)}{\Pr(d)} \\ &\propto \Pr(c)\Pr(d|c) \propto \pi_c \textstyle\prod_{t\in d} \theta_{c,t}^{n(d,t)}, \end{aligned} \tag{1}$$

where $c \in C$ is the label, d is the test document, t occurs $n(d,t)$ times in d, π_c is the fraction of documents tagged c (also called the *prior* probability of c), and $\theta_{c,t}$ are multinomial probability parameters [13, 1], estimated from training documents as

$$\theta_{c,t} = \frac{\lambda + \sum_{d\in D_c} n(d,t)}{\sum_{\tau\in T}\left(\lambda + \sum_{d\in D_c} n(d,\tau)\right)}, \tag{2}$$

where T is the vocabulary or feature set, D_c is the set of training documents marked with label c, and $0 < \lambda \leq 1$ is the *Lidstone's smoothing* parameter [17] ($\lambda = 1$ corresponds to the well-known Laplace's smoothing). Having estimated model parameters from training data, the goal is to find the best class $\arg\max_c \Pr(c)\Pr(d|c)$ for test documents.

3.1.2 Expectation maximization (EM1D)

The NB classifier needs each training document to be marked with one label. Can we make use of additional documents with no label information (such as the test documents themselves) or partial label information (e.g., that a document was generated from one among a restricted subset of labels)?

A classic approach to estimating distributions over missing values is Expectation Maximization (EM) [5]. Nigam *et al.* [15] use EM to induce a document classifier starting from a few labeled and many unlabeled documents (Figure 1). Because this algorithm is designed for only one label set, we will call it **EM1D**.

1: Use labeled documents to induce a naive Bayes classifier with parameters Θ
2: **while** model Θ has not stabilized to satisfaction **do**
3: set up new model parameters Θ'
4: collect contributions from labeled documents to Θ'
5: **for** each unlabeled document d **do**
6: **E-step:** calculate the class probabilities $\Pr(c|d,\Theta)$ based on current parameters
7: **M-step:** if term t occurs $n(d,t)$ times in d, let d "contribute" a fractional term count of $\Pr(c|d)\,n(d,t)$ to the next estimate $\theta'_{c,t}$
8: **end for**
9: Re-estimate new cluster model parameters Θ'
10: $\Theta \leftarrow \Theta'$
11: **end while**

Figure 1: Using standard EM ("EM1D") for semi-supervised learning of document labels.

3.2 EM2D: Cross-trained naive Bayes

We extend Nigam *et al.*'s EM algorithm to EM2D by creating a 2d grid of class labels taken from the product set $C = A \times B$. We assume a standard mixture model [5] for document generation. First the label pair (c_A, c_B) is picked with probability $\Pr(c_A, c_B)$, and then a conditional term distribution $\Pr(d|c_A, c_B)$ is sampled to generate the document. Thus,

$$\Pr(d) \quad = \quad \Pr(c_A, c_B)\Pr(d|c_A, c_B). \tag{3}$$

We assume the term distribution to be multinomial, extending parameters $\theta_{c,t}$ to $\theta_{c_A,c_B,t}$. Likewise, parameters π_{c_A,c_B} express the prior probability of a document being generated from label-pair (c_A, c_B).

Thus, each document belongs to exactly one cell of this grid[1]. However, in the mapping scenario (§2.1) each training document comes with exactly one label, which determines either the row or the column where the training document belongs, but not both. Thus, each document d identifies a subset $C_d \subset C$ to which it potentially belongs, and for $\gamma \notin C_d$, we are given that $\Pr(\gamma|d) = 0$. We force this constraint in the E-step shown in Figure 1, limiting the contributions from a training document to its correct row or column, and scaling the E-variables to add up to 1 over the row or column.

3.2.1 Initialization

The EM algorithm [5] guarantees only a locally optimum solution to the E and M variables. It is important to start the iterations from a reasonably good initial estimate of Θ.

[1]It is possible that a document belongs to more than one class in a single taxonomy; handling such cases is left to future work.

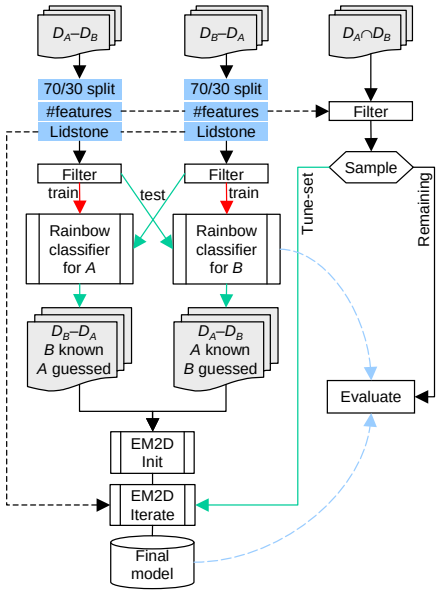

Figure 2: Design and evaluation of EM2D.

In EM2D, we have two resources at our disposal to achieve good initialization.

The first option is to train two naive Bayes classifiers on $D_A - D_B$ and $D_B - D_A$ (see §2.1 for their definition), and derive *guessed* B-labels for $D_A - D_B$ and guessed A-labels for $D_B - D_A$. The basic naive Bayes classifiers also help us (via random 70%/30% train/validate splits) to choose an initial number of features (in decreasing order of information gain) and an initial value of the Lidstone parameter λ in equation (2). These initial steps are shown near the top of Figure 2.

The second option is to use the tuneset of fully-labeled documents to seed the initial Θ distribution. However, the tuneset is generally rather small. Using *only* the tuneset would generally fail to populate all the cells of the label grid adequately. It is probably best to use both options in the rare case that fully-labeled data is available.

3.2.2 Update rules

Suppose a training document d has $\alpha = c_A(d)$ known but $c_B(d)$ unknown. Then $\sum_{c_B} \Pr(\alpha, c_B|d,\Theta) = 1$. Using the standard multinomial model in equation (1), we can write

$$\Pr(\alpha, c_B|d,\Theta) \quad = \quad \frac{\pi_{\alpha,c_B}\prod_{t\in d}\theta_{\alpha,c_B,t}^{n(d,t)}}{\sum_\beta \pi_{\alpha,\beta}\prod_{t\in d}\theta_{\alpha,\beta,t}^{n(d,t)}}. \tag{4}$$

This completes the specification of the E-step, although some care is required to preserve numerical precision. For the M-step, we set

$$\pi'_{\alpha,\beta} \quad = \quad \frac{1}{|D|}\left[\begin{array}{l}\sum_{d:c_A(d)=\alpha}\Pr(\alpha,\beta|d,\Theta) \\ +\quad \sum_{d:c_B(d)=\beta}\Pr(\alpha,\beta|d,\Theta)\end{array}\right], \tag{5}$$

which is simply the *expected fraction* of documents occupying label cell (α, β). Likewise, we set

$$\theta'_{\alpha,\beta,t} \quad = \quad \frac{\left[\begin{array}{l}\lambda + \sum_{d:c_A(d)=\alpha} n(d,t)\Pr(\alpha,\beta|d,\Theta) \\ +\ \sum_{d:c_B(d)=\beta} n(d,t)\Pr(\alpha,\beta|d,\Theta)\end{array}\right]}{\sum_\tau\left[\begin{array}{l}\lambda + \sum_{d:c_A(d)=\alpha} n(d,\tau)\Pr(\alpha,\beta|d,\Theta) \\ +\ \sum_{d:c_B(d)=\beta} n(d,\tau)\Pr(\alpha,\beta|d,\Theta)\end{array}\right]} \tag{6}$$

This expression closely resembles equation (2), except that again, contributions to term counts are weighted by the probability of each document occupying label cell (α, β), like in step 7 of Figure 1.

Damping: In the EM2D setup, different documents have different quality and extent of label information. Tuneset documents plug into exactly one known (α, β) slot and presumably have the most reliable label information. Training documents have one label pinned by human input, which is assumed to be reliable, but the other label is not as reliable. In the zero-label setting (§2.2), test documents have neither label known, but may still help the classifier gain accuracy by participating in the EM iterations "completely floating" over the label grid.

In the update equations above, we have given one vote to each document. However, it is common [15] to use a *damping factor* $L \leq 1$ to scale down the contribution of documents whose labels we consider less reliable. It is as though a fully-labeled document is worth one vote, but a singly labeled document is worth only $L = 0.5$, say. Thus, L can be thought of as an instance scaling mechanism like in boosting. It does not invalidate the theory of EM in any way. The best value of L can be set by cross-validation.

Early stopping: The NB generative model is a very crude approximation to reality. Therefore, maximizing data likelihood using EM may not improve classification accuracy in all cases. It is common to use a tuneset to stop EM iterations in case classification accuracy over (cross-) validation data is found to drop [15].

3.2.3 Deployment

The half-label setting is simple. Given a test document d with $c_A = \alpha$ known, we simply find $\Pr(\alpha, c_B|d)$ for all candidates c_B, and report the best. The zero-label setting gives us at least two distinct options: EM2D with guesses and EM2D with model aggregation.

EM2D with guesses (EM2D-G): To classify to target taxonomy B, we first apply an A-classifier to the test document. The guessed A-label now lets us deal with the zero-label test instance as if it were a mapping problem. Obviously, we should use the best possible A-classifier.

EM2D model aggregation (EM2D-D): After EM2D iterations are over, we use the final values of the E-variables to prepare a new classifier for target taxonomy B. More specifically, each training document $d \in D_A - D_B$ has associated E-values $\Pr(c_B|d, c_A)$. Just like in EM, we let d "contribute" its term counts in proportion to this probability to label c_B. Documents in D_B contribute fully to their respective labels in B. The resulting "aggregated" classifier for B is used to classify test instances.

3.3 Stratified EM1D

If EM2D improves upon the accuracy of single-taxonomy learners, that could be attributed to multiple reasons. Let B be the target taxonomy in this discussion. The mapping of A-labeled documents to B may improve simply because of the extra documents in $D_A - D_B$, not because these documents are A-labeled. Whether this is the case can be easily determined by calibrating EM2D against EM1D (run with B as target labels) with the documents in $D_A - D_B$ thrown in as unlabeled documents.

Between EM1D and EM2D there are options which let us use the A-labels, but in ways simpler than EM2D. E.g., we can set up an EM1D instance for each $\alpha \in A$. The B-labeled documents are shared across all such instances. A-labeled documents bearing the label α become unlabeled documents for the instance corresponding to α. The pseudo-code is shown in Figure 3. We call this **Stratified-EM**.

```
1: for each label α ∈ A do
2:     train a B classifier Θ^α using EM1D with D_B − D_A
       as the labeled set and {d ∈ D_A − D_B | c_A(d) = α} as
       the unlabeled set.
3: end for
4: for each test document labeled (c_A, ?) do
5:     use the EM1D model Θ^{c_A} to predict c_B
6: end for
```

Figure 3: Stratified EM to exploit A-labels while classifying for B.

If EM2D beats both EM1D and Stratified-EM, we can conclude that the mutual "corrections" of term distributions in EM2D are somehow vital to its higher accuracy.

4. DISCRIMINATIVE CROSS-TRAINING

The classifiers discussed thus far aim to fit a class-conditional generative distribution $\Pr(d|c)$ (or $\Pr(d|c_A, c_B)$), and use Bayes rule to estimate $\Pr(c|d)$ (or $\Pr(c_A|d, c_B)$ etc.). In contrast, discriminative classifiers seek to directly fit a regression function from the document to scores for label(s).

In this section we will discuss cross-training using two discriminative classifiers. The first (new) approach uses Support Vector Machines (SVMs), which have been reported to do well for text data [7]. The second (prior) approach [1] combines distributional and discriminative aspects.

4.1 SVM-based cross-training

Linear SVMs: Suppose we are given a vector representation of n documents. Each vector has a component for each feature (in our case, a term) which is proportional to the number of times the term occurs in the document[2]. Document vectors are usually scaled to unit L_2 norm. Each document vector is associated with one of two labels, $+1$ or -1. The training data is thus $\{(d_i, c_i), i = 1, \ldots, n\}, c \in \{-1, +1\}$.

A linear SVM finds a vector $\mathbf{w}$ and a scalar constant b such that for all i, $c_i(\mathbf{w} \cdot d_i + b) \geq 1$, and $\|\mathbf{w}\|$ is minimized. This optimization corresponds to fitting the thickest possible slab between the positive $(c = +1)$ and negative $(c = -1)$ documents. In case the training samples are not linearly separable, it is possible to trade off the slab width for the number of misclassified training instances.

If the data has more than two labels, it is common to create an *ensemble* of yes/no SVMs, one for each label. During training, a document marked c is a positive example for the SVM associated with c, and a negative example for all other SVMs. This is called the "one-vs-rest" ensemble approach. During testing, for a test document d, each SVM evaluates its regression function; the SVM corresponding to label c evaluates $\mathbf{w}_c \cdot d + b_c$. The label chosen is $\arg\max_c(\mathbf{w}_c \cdot d + b_c)$ (other policies can also be used).

Inducing a SVM classifier involves a complex, iterative numerical optimization. Several implementations of SVM

[2]SVMs have been commonly used on the standard TFIDF representation. We used both TF and TFIDF representations scaled to unit norm and found similar results.

are publicly available, including Sequential Minimum Optimization (SMO) [16, 7] and SVMLIGHT [8].

Cross-training SVMs: If A-labels are good predictors of B-labels, one way to enhance a purely text-based SVM learner for B is to allocate, over and above a column for each token in the training vocabulary, $|A|$ extra columns, one for each label in A. A document $d \in D_B - D_A$ is submitted to a text-based SVM ensemble for A, called $S(A, 0)$, which gives it a score $\mathbf{w}_{c_A} \cdot d + b_{c_A}$ for each class $c_A \in A$.

These scores can be inserted into the $|A|$ new columns, either as-is, or after some simple transformation, such as taking the sign of the score, or converting the largest score to $+1$ and the rest to 0 or -1 (we use the latter option in our experiments), and scaling ordinary term attributes by a factor of $f (0 \leq f \leq 1)$ and scaling these *label attributes* by a factor of $1 - f$. Document vectors are always scaled to unit L_2 norm.

The parameter f, which can be chosen through cross-validation on a tuneset, decides the relative importance of label and term attributes. We evaluated f from 0 to 1 in steps of 0.05 and set $f = 0.95$. These cross-trained SVMs are denoted by **SVM-CT**.

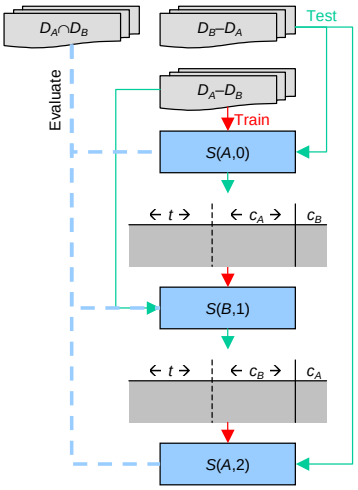

Figure 4: Cross-training SVMs.

Documents in $D_B - D_A$ thus get a new vector representation with $|T| + |A|$ columns where $|T|$ is the number of term features. They also have a supervised B label. These are now used to train a new SVM ensemble $S(B, 1)$. The document tables and how they are used to train and test SVMs are shown in Figure 4. We can obviously repeat the process iteratively in a ping-pong manner, each classifier providing synthetic columns for the other. The complete pseudo-code is shown in Figure 5.

Our experiments show that SVM-CT does outperform SVM, making effective use of the label attributes (although there is little improvement beyond the first ping-pong round). SVM-CT is also better than the distributional cross-training methods in about half the cases. SVM (which uses text alone), in turn, is much better than the baseline NB classifier. Moreover, inspecting the components of $\mathbf{w}$ along the label dimensions derived by SVM-CT gives us some interesting insights into various kinds of mappings between the label sets A and B. We will return to these observations in §5.

1: Represent each document as a vector d in term space and $\|d\| = 1$
2: Build one-vs-rest SVM classifiers $S(A, 0)$ and $S(B, 0)$ for $D_A - D_B$ and $D_B - D_A$ using text tokens only
3: **for** $i = 1, 2, \ldots$ **do**
4: **for** each document $d \in D_B - D_A$ **do**
5: Apply $S(A, i-1)$ to d, getting a vector $\gamma_A(d)$ of $|A|$ scores (see text)
6: Concatenate vectors d and $\gamma_A(d)$ into a single training vector with label $c_B(d)$, with relative term-label weight determined by f and maintaining $\|d\| = 1$
7: Add this vector into the training set for a one-vs-rest SVM classifier $S(B, i)$
8: **end for**
9: Similarly, use $S(B, i-1)$ to get $\gamma_B(d)$ and induce a new one-vs-rest SVM classifier $S(A, i)$ for all $d \in D_A - D_B$
10: **end for**

Figure 5: Cross-training SVMs.

4.2 The A&S mapping algorithm

Agrawal and Srikant (A&S) [1] proposed a hybrid distributional/discriminative classification algorithm by enhancing the prior estimation of NB (equation 1). Let the target label-set be C and the source label set be S (to be consistent with their notation). In the mapping setting, classifying document d entails finding $\arg\max_c \Pr(c|d, s)$, where the source label $s \in S$ is supplied and the target label c is sought.

Given d and s are fixed, $\arg\max_c \Pr(c|d, s)$ is equal to $\arg\max_c \Pr(c|s)\underline{\Pr(d|c, s)}$, which A&S approximate as $\Pr(c|s)$ $\underline{\Pr(d|c)}$, using the conditional independence assumption shown underlined (which is theoretically debatable, but seems to work in practice). All that remains is to propose parametric forms for $\Pr(c|s)$ and $\Pr(d|c)$. $\Pr(d|c)$ is modeled exactly as in equation (1), i.e., $\Pr(d|c) \propto \prod_{t \in d} \theta_{c,t}^{n(d,t)}$. All $\theta_{c,t}$ are pre-estimated by a C-trained classifier which has no knowledge of S-labels. (This is the distributional/generative part.)

The key innovation of A&S is to propose a parametric form for $\Pr(c|s)$ depending on inter-label relations. Let N_c be the number of C-labeled documents in the training set for C. As in EM2D, A&S use a C-trained classifier to guess classes of S-labeled documents; let $G(s, c)$ be the number of documents with source label s that this classifier assigns to target label c. The overall score uses a tuning parameter $R \geq 0$ and is given by

$$\begin{aligned} \Pr(c|d, s) &\propto N_c \; G(s, c)^R \prod_{t \in d} \theta_{c,t}^{n(d,t)}, \quad \text{or} \qquad (7) \\ \log \Pr(c|d, s) &= \text{constant} + \log N_c + R \log G(s, c) + \\ &\quad \textstyle\sum_{t \in d} n(d, t) \log \theta_{c,t}. \end{aligned}$$

Note that (once the $\theta_{c,t}$s are fixed) R is the only tunable parameter here. $R = 0$ coincides with standard NB on the master labels. Taking logs, we see that (like SVM) A&S is also a linear discriminant learner. A&S use a tuneset to set the best value of R, which can be chosen in two ways.

Random sampling: A fraction (varying between 10% and 90% in our experiments) of the fully-labeled documents is sampled to create a tuneset. The remaining documents are used as the test set. We evaluate a range of choices for $R \in \{0, 1, 3, 10, 30, 100, \ldots\}$ against the tuneset. The accuracy is averaged over dozens of such samples.

Active learning: The system repeatedly samples the fully-labeled documents. For each sample d, it varies R to see if R makes any difference to the estimated C-label. If it does, d is placed in the tuneset; otherwise it is put in the test/calibration set. A&S report that 5–10 actively chosen samples are adequate to pick a suitable R.

5. EXPERIMENTS

All our algorithms were coded in a few thousand lines of simple C++. A&S, EM2D, and variants were run on (over 1GHz) Pentium3 servers with 1–3GB of RAM. The models fit easily in tens of megabytes of RAM. We scanned the documents sequentially and did not need to hold document vectors in memory. The SVM implementation we used did load document vectors into memory. A&S, EM2D and its variants generally trained faster than SVM and SVM-CT.

5.1 Data collection and preparation

We collected example URLs from Dmoz and Yahoo!. Their intersection has 110926 documents, less than 10% of either's total size. This supports our claim that double-labeled documents are hard to find. Like A&S [1] we selected five data sets: Autos, Movies, Outdoors, Photo, and Software. However, their sub-topics and training examples were not available to us. Therefore, for each data set, in each of the two taxonomies, we picked immediate children as labels such that there were at least 10 URLs in common with a label of the other taxonomy. We then added in a few additional labels from each taxonomy. Finally we went back to the original Dmoz and Yahoo! sources to collect all URLs within the chosen label sets; some 70–80% of the fetches succeeded.

Data set	$\lvert D_A - D_B \rvert$	$\lvert A \rvert$	$\lvert D_B - D_A \rvert$	$\lvert B \rvert$	$\lvert D_A \cap D_B \rvert$
Autos	3589	31	3138	24	184
Movies	8003	33	11420	27	1222
Outdoors	8739	26	1540	39	181
Photo	2895	8	438	22	95
Software	9851	51	2383	25	264
Bookmarks	47247	154	365	7	1289

Figure 6: Sizes of various document and label sets in our collected data. The first five benchmarks Autos to Software are used primarily for studying the half-labeled mapping scenario, and Bookmarks is used for studying the zero-label scenario.

Figure 6 shows various properties of our main data sets. We felt uncomfortable about the small test sets, but A&S reported intersections of similar small sizes, and we also found human labeling (based on page text alone) to systematically reject pages with relatively unreliable text, exactly those cases where A&S and cross-training are likely to shine.

The Bookmarks data set was created mainly to study zero-labeled classification. We collected and inspected a dozen or so bookmark files published on the Web. We found it very common for bookmark authors to collect URLs into coherent topics. Usually, these topics had strong correspondence with one or few topics in Yahoo!/Dmoz. However, the number of URLs per topic was small (say 3–20), exactly the scenario we painted at the outset. We derived sample bookmark topics B from these bookmark collections, and populated them from Yahoo! (A) URLs, and removed them from D_A.

5.2 The naive Bayes baseline

We used naive Bayes (NB) classifier in the Rainbow package [12]. We created two Rainbow classifiers, one for A labels using $D_A - D_B$, the other for B labels using $D_B - D_A$.

Apart from providing a strawman, NB runs are used to set the Lidstone parameters and the feature sets for A and B. Consider the classifier for A. We first created a random 70%/30% train-test split of $D_A - D_B$. Rainbow ingested the 70% training subset and listed features in decreasing order of information gain (w.r.t. the labels). In an outer loop, we chose from λ_A between 0.1 and 1 in steps of 0.1. In an inner loop, we chose a prefix T_A of the feature list of size 10% through 90% in steps of 10% (similarly for B). We then used the 30% validation data to pick the best values for $\lambda_A, \lambda_B, T_A, T_B$. Finally, the NB baseline is obtained by subjecting the held-out $D_A \cap D_B$ to these optimized Rainbow classifiers. Figure 7 shows various accuracy statistics.

Data set		$\lvert T \rvert$	λ	70/30	$D_A \cap D_B$
Autos	A	10000	0.1	39.3	46.5
	B	10000	0.2	59.2	65.6
Movies	A	8385	0.5	44.6	43.0
	B	64434	0.2	50.9	41.0
Outdoors	A	2142	0.2	79.8	77.1
	B	813	0.5	68.0	78.0
Photo	A	27969	0.5	68.7	40.9
	B	325	1.0	49.6	35.5
Software	A	40000	0.1	40.0	47.8
	B	17000	0.1	58.4	54.3

Figure 7: Naive Bayes baseline accuracy with optimized choices of $\lvert T \rvert$, the number of features, and λ, the smoothing parameter. A is Dmoz and B is Yahoo!. Percent accuracy is shown for 70/30 cross validation and the unseen $D_A \cap D_B$ test set.

All other distributional cross-training algorithms used these optimized values of λ and T. In particular, EM2D used $T_A \cup T_B$ as the feature set, and the average of λ_A and λ_B.

Feature selection and the choice of λ matters a great deal for most data sets. Given high-dimensional data like text, feature selection would likely be helpful for any learning method, but the benefit from tuning λ is large mainly because the naive Bayes model results in terrible estimates of the joint distribution, and any "fix" to the innumerable θs is likely to help. Whereas the two classifiers can each optimize T_A, T_B, λ_A and λ_B in an unconstrained manner, EM2D is stuck with a single feature set and a single value of λ, which puts it at a disadvantage.

5.3 SVM and cross-training

We used SVMLIGHT [8] in one-vs-rest ensemble mode, with a linear kernel and default settings for all parameters. Documents were represented as unit vectors (§4.1). f was set to 0.95 as explained earlier in §4.1.

Figure 8 compares the accuracy of SVM and SVM-CT with the NB baseline. In most cases, SVM beats NB. This is consistent with folk wisdom that SVMs generally perform better than NB on text classification tasks. More interesting is the observation that SVM-CT sometimes has higher accuracy than SVM, which shows that it is possible

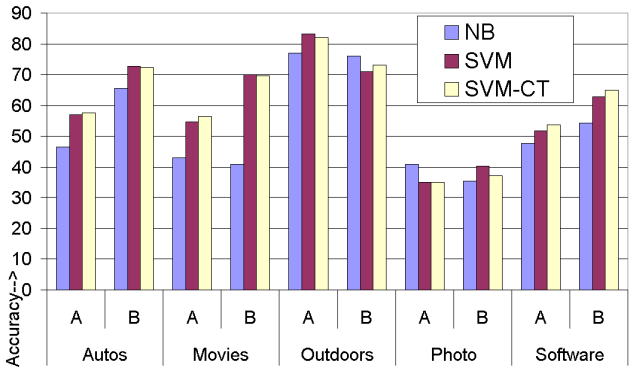

Figure 8: Comparative evaluation of NB, SVM and SVM-CT (cross-trained SVM).

for SVM-CT to exploit additional information from label-derived columns.

We made two additional studies of SVM-CT. First, we checked that the average magnitude of $\mathbf{w}$ for ordinary term features was always lower than the average magnitude of $\mathbf{w}$ for label-derived features. Recall that $|\mathbf{w}_t|$ is a measure of how strongly the feature t can influence the decision of the SVM, i.e., the sensitivity of the SVM to feature t.

Second, we tabulated the B (respectively, A) labels corresponding to the highest and lowest $\mathbf{w}$ values of various A (respectively, B) classifiers. We wanted to observe the mappings learned between the classes in the two taxonomies using cross-trained SVMs[3]. During cross-training, the label information was transformed into a vector of 1 and -1 values as mentioned in §4.1. In addition, a new dimension called none-of-the-above (NOTA) was introduced, whose value was set to 1 when all label scores obtained from B (respectively, A) were negative and all label dimensions were set to -1. The purpose of NOTA is explained shortly.

The results are shown in Figure 9. We show some Dmoz (respectively, Yahoo!) class labels along with the Yahoo! (respectively, Dmoz) class labels which had the greatest positive and negative influence in predicting the said Dmoz (respectively, Yahoo!) class. All positive couplings are very meaningful; some negative couplings are fairly intriguing too.

The `Outdoors` dataset for both taxonomies contains the class `ScubaDiving` which maps to its namesake in the other taxonomy with a large positive component along $\mathbf{w}$. Such one-to-one mappings are symmetric and expected. Even when there is no direct one-to-one correspondence between the labels (or there is a containment relationship) such as between `Yahoo.Movies.Genres.SciFi-Fantasy` and `Dmoz.Movies.Series.StarWars`, SVM-CT seems capable of extracting that information. On the other hand when the `Dmoz.Software.Accounting` class really has no relevant class in the Yahoo! taxonomy, the synthetic NOTA class indicates this by the high value of $|\mathbf{w}_{\text{NOTA}}|$.

The superclass `Dmoz.Photo.Techniques&Styles` duly maps to finer categories in `Yahoo.Photo`, such as Pinhole Photography, 3D Photography, and Panoramic Photography. The Dmoz to Yahoo! mapping gives high positive weights to most of these *child* classes as seen in Figure 9. Such instructive *parent-child*, or *one-to-many* mappings emerge in spite of our assumption of flat taxonomies.

[3]See `http://www.cse.iitb.ac.in/~soumen/doc/kdd2003/svmct.html` for examples

Dataset	`Dmoz.`	Maps to `Yahoo.`	Weight
Autos	News&Magazines	News&Media	0.147
		Volkswagen	−0.156
Movies	Genres/Western	Titles/Western	0.242
		Titles/Horror	−0.052
Outdoors	Scuba Diving	Scuba	5.878
		Snowmobiling	−0.647
Photo	Techs&Styles	Pinhole Ph'graphy	2.796
		3D	0.964
		Panoramic	0.921
		Organizations	−1.184
Software	Accounting	NOTA	0.156
		Screen Savers	0.103
		OS/Unix	−0.171
Dataset	`Yahoo.`	Maps to `Dmoz.`	Weight
Autos	Corvette	Chevrolet	0.981
		Parts&Accessories	−0.266
Movies	SciFi&Fantasy	Series/Star-Wars	1.123
		Reviews	−0.824
Outdoors	Scuba	Scuba Diving	4.822
		Wildlife	−0.437
Photo	Pinhole Ph'graphy	Techs&Styles	0.4842
		Photographers	−0.270
Software	OS/MSWindows	OS/MSWindows	0.018
		NOTA	−0.001
		OS/Unix	−0.008

Figure 9: Dmoz and Yahoo topic mappings learned with cross-trained SVMs

Another interesting mapping is from `Yahoo.Software.OS.MSWindows` to multiple high positive weights to classes in the Dmoz taxonomy. Here, the NOTA class can be interpreted as clearly separating all the Windows related classes above it from the Unix related classes below it within `Dmoz.Software`.

5.4 EM2D for mapping

Figure 10 shows the accuracy of EM2D in comparison with NB. EM2D is significantly better than NB with a maximum gap of 30% for the Movies dataset and average gap of 10%. This is reassuring, but in this section we wish to analyze carefully *why* this is the case.

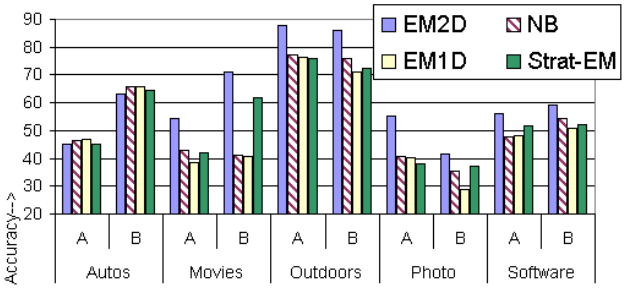

Figure 10: EM2D vis-a-vis Stratified-EM1D, EM1D, and NB. For EM1D we used its best damping parameter, $L = 0.01$.

There are two potential sources of information from $D_A - D_B$ which may improve the accuracy of classifying into B given d and c_A. The first is simply the addition of a bunch of documents, even if they are not labeled with B-labels and even if we ignore their A-labels. If this were the only source of extra information, EM1D should be able to match EM2D, which is clearly not the case. Therefore, knowledge of A-labels of specific documents is vital.

As we discussed in §3.3, A-labels can be used by Stratified-EM, which simply creates one instance of EM1D for each distinct label $\alpha \in A$. Figure 10 also shows that with only two minor exceptions, EM2D beats both EM1D and Stratified-EM. This despite the fact that each EM1D has denser data, lowering the variance of the parameter estimates compared to EM2D. These measurements help us establish that

- There is information available in cross-training which EM1D cannot exploit, and
- Stratified-EM, a relatively straight-forward extension to EM1D, does *not* work as well as EM2D.

5.5 Sensitivity to initial guesses of labels

EM2D, like EM1D, finds locally optimum values of the total data likelihood. Hence the final accuracy is sensitive to the initial assignment of half-labeled data in the 2D label grid.

Given the baseline classifiers trained on A (using documents in $D_A - D_B$) and B (using documents in $D_B - D_A$), it is natural to initialize EM2D by submitting documents in $D_A - D_B$ to the B-classifier and vice versa. We may use a "hard" assignment to the best guess, or a "soft" or fractional assignment based on the probabilities emitted by the baseline classifiers.

However, these are not the only options. How will EM2D behave if each document in $D_A - D_B$ is assigned *uniformly* over each label in B? In general, how sensitive is EM2D to perturbations and errors in the initial E-estimates?

To test EM2D's resilience, we randomly picked a fraction q of documents (with A-labels, say) and replaced their guessed scores for B-labels with a uniform distribution smeared over all B-labels. The remaining fraction $1 - q$ of documents are added to the EM2D system as before. Thus, $q = 1$ corresponds to full uniform assignment.

Obviously, the effect of smearing a fraction q depends on the accuracy of the guesses in the first place. Therefore we repeat the smearing experiments for varying level of starting guess accuracy. (We fake different guess accuracies by random flips in guesses. Note that these "flips" are distinct from the "smear.")

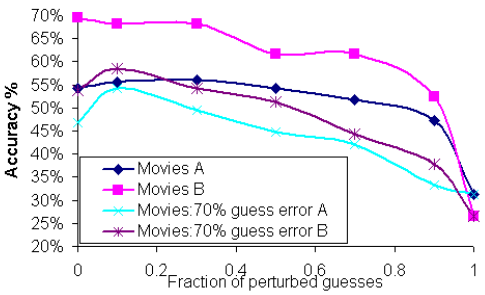

Figure 11: The effect on EM2D of smearing the initial guesses of a fraction of half-labeled training documents. The y-axis shows final EM2D accuracy.

In Figure 11 we show the change in accuracy with increasing fraction of smeared guesses on the Movie dataset, with two different settings of guess accuracy: the default as shown in Figure 7 and a second setting where 70% of the guesses have been pre-flipped to a random label (this results in guesses of very poor quality). These plots show that

- When the guesses are reasonably accurate, uniform assignment is worse than assignment based on guessed probabilities, which makes eminent sense.
- EM2D can handle limited ($q = 0.15$ to $q = 0.20$ for this data) smearing, beyond which accuracy starts to drop.
- When the accuracy of guesses is too poor, smearing a fraction of the guesses ($q = 0.10$ in this case) can improve accuracy. This was somewhat unexpected, and made sense only in hindsight.

5.6 Highly asymmetric scenarios

All the data sets we collected have relatively balanced sizes of $D_A - D_B$ and $D_B - D_A$. How well can EM2D do in highly unbalanced settings, especially w.r.t. the sparsely-populated taxonomy?

To answer this question, we (arbitrarily) picked B as the taxonomy to be decimated, and sampled $D_B - D_A$ down to 300 documents. (Actually, we decimated to 200, 300, and 5% of the original. Results were similar.) $D_A - D_B$ was left unchanged.

The small size of $D_B - D_A$ led to a poor baseline B-classifier. Therefore, the guessed B-labels for the documents in $D_A - D_B$ had a large error rate. Because information flow is bidirectional in EM2D, poor B guesses reduced overall accuracy. We propose three fixes for this problem:

- Taking our cue from Figure 11, we smear documents in $D_A - D_B$ over all B-labels. Documents in $D_B - D_A$ continue to use the A guesses.
- Like A&S, we use a tuneset sampled from $D_A \cap D_B$. Specifically, we sampled 5% of fully-labeled documents.
- We set the damping factor L (§3.2.2) so as to restore the relative weights of $D_A - D_B$ and $D_B - D_A$ to the same ratio as in the original dataset. This would mean $L \approx 0.05$ on documents in $D_A - D_B$.

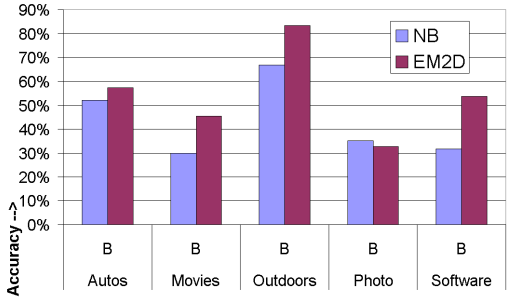

Figure 12: EM2D on a small sample of 300 documents from $D_B - D_A$.

Figure 12 shows that the accuracy gain of EM2D over NB in highly asymmetric settings can in fact be higher (average 11.4%) than in more balanced data (average 10% in Figure 10), provided EM2D is initialized properly. Nigam *et al.*'s experience [15] seems to corroborate that the gains from semi-supervised learning are larger when labeled data is limited.

5.7 Comparison with the A&S algorithm

For our A&S implementation, we fixed the feature set to $T_A \cup T_B$ as found by Rainbow, and also fixed the λ parameter

to one that gave the best accuracy for Rainbow for each of A and B prediction.

Making a fair comparison between A&S and EM2D involves exposing to EM2D at least the fully-labeled tuneset that A&S uses. In fact, it is very difficult to compare the active-learning version of A&S with EM2D in a principled way, because A&S inspects fully-labeled documents (not the labels, but the text) outside the tuneset as well. (EM2D is not designed for active learning.) Therefore, we focused on the randomly sampled tuneset paradigm only, because that could be used with both A&S and EM2D.

In addition, given the large skew between half-labeled and fully-labeled populations, we used damping to re-scale them to the same effective size (see §3.2.2 for more details).

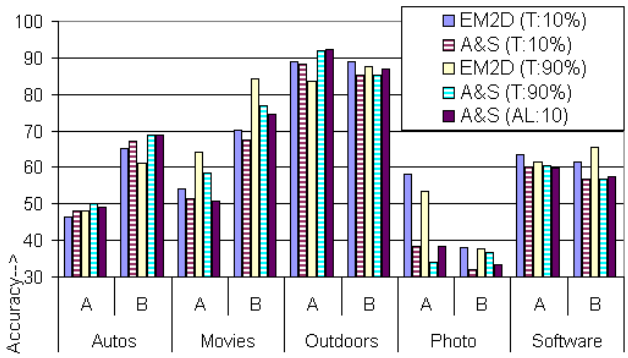

Figure 13: Accuracy of the A&S algorithm compared with EM2D for 10% and 90% tuneset (T) and A&S active learning (AL).

In Figure 13 we present the accuracy of A&S with 10% and 90% randomly sampled tuneset as well as a tuneset of size 10 picked by active learning (AL) from the entire test set of fully-labeled documents. Broadly, A&S and EM2D are comparable, but EM2D edges over A&S by a maximum of 20% and an average of 4% for the 10% tuneset and 2% for the 90% tuneset. When EM2D loses to A&S, the gap is very small.

5.8 EM2D for classifying zero-label documents

In this scenario, we are required to finally produce a classifier for B which does not depend on the test instances being labeled with A labels. In §3 we discussed two methods for deploying EM2D in this setting: EM2D-D, a model aggregation method, and EM2D-G which is essentially EM2D, except that the A-labels are supplied as guesses from an A-classifier.

Figure 14 shows the accuracy for EM2D-D, EM2D-G, EM1D and NB, for various sizes of labeled training sets, and two choices of the damping factor L discussed at the end of §3. These numbers are for the Bookmark data set. The accuracy values were averaged over three random choices of the training set for each choice of training set size.

As the fraction of training data is increased, the benefit of semi-supervised learning reduces, which is obvious. The damping factor does essential damage control when there are many labeled documents, but can hurt when the labeled set is very small. These observations corroborate with earlier EM1D results by Nigam *et al.* [15].

Unlike Nigam *et al.*, EM1D could improve beyond NB only for the smallest training sets in our case. One possible reason is that the unlabeled Yahoo! dataset, from which EM1D adds instances, is significantly different, and has many more irrelevant classes, compared to the initial labeled data in our Bookmark dataset. Nigam *et al.*'s experiments drew unlabeled and labeled documents from the same distribution.

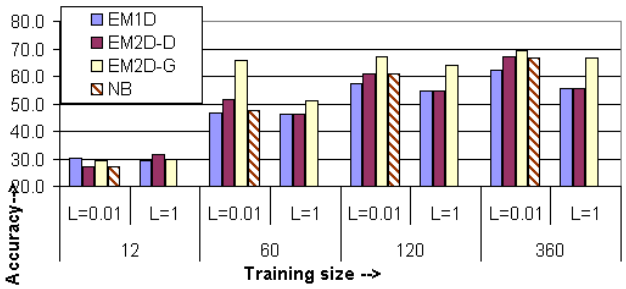

Figure 14: EM2D with guessing is the best methods for classifying zero-label documents. NB accuracy is shown only once for each size of the training set, because it does not change with L.

Finally, we were surprised to see that EM2D-G performed better than EM2D-D. Recall that EM2D-D is really a 1d classifier, which should reduce data sparsity and improve the reliability of its parameter estimates compared to EM2D. Despite this benefit, model aggregation appears to hurt. Even a noisy guess at the A-label, followed by a row-conditioned classification, outperforms the aggregated model.

6. RELATED WORK

In recent years, EM-like semi-supervised learning has been enhanced in several ways and applied to a number of settings.

Extending beyond EM1D, Liu *et al.* [10] and Yu *et al.* [20] consider the realistic situation where, apart from labeling only a few samples, the user is also unlikely to spend the effort to mark negative samples. Their EM-like algorithms can work on a set of positive examples P and a mixed pool of samples M which may contain both positive and negative instances.

Cross-training is related to *multi-task* learning or *life-long* learning, in which information (features, models, etc.) from one learning task is used for another. Thrun [18] discusses how to cluster learning tasks (not instances) and pick, for a given task, those other tasks that are likely to be related to the current one. Information from those tasks are then used to influence the distance function in a nearest-neighbor classifier. Caruana [4] discusses how to use multi-task learning in neural networks, and Baxter [2] provides a PAC analysis. Cross-training is a two-task setting with no instance submitted to more than one task. The similarity between tasks falls out naturally as we estimate $\pi_{\alpha,\beta}$.

A recent approach to semi-supervised learning (which might appear superficially similar to cross-training) is *co-training*, proposed by Blum and Mitchell [3]. In co-training, too, there are two learners, but, unlike cross-training, the learners have to use *disjoint* subsets of attributes, and assign labels from only *one* taxonomy. Each learner picks unlabeled training instances that it is most confident about classifying correctly, and makes it a labeled training instance for the other learner. Co-training and cross-training are quite

different things: two label sets are central to our formulation, and our approach depends on modeling a single term distribution conditioned on a pair of labels.

Doan *et al.* [6] study a related problem of identifying mapping between labels of two taxonomies (called ontologies in the paper). Their goal is to find for each label in one taxonomy, the label most similar to it in the other taxonomy. In contrast, our goal is to assist classification in one catalog without necessarily committing on a specific mapping relation with another catalog.

7. DISCUSSION AND FUTURE WORK

We have presented *cross-training*, a new technique for using sample documents from one taxonomy to improve classification tasks for another taxonomy. We have presented two algorithms for cross-training: a probabilistic algorithm based on EM and a discriminative algorithm based on SVMs.

Extensive experiments with real-life Web data show that our approach definitely beats baseline classifiers in each taxonomy, which is not very surprising. More reassuring are the observations that show our approach to compare favorably with the best existing approach, while providing a more sound foundation.

In principle, a sufficiently powerful supervised learner that can handle discrete categorical attributes can be used directly for cross-training. In practice, specifically for text data, the large number of dimensions and the heterogeneity across term and label attributes pose challenges. Luckily, in cross-training, the label attribute can take only a few values. Therefore (in decision tree terms) we can first stratify the data on the label attribute and then build distributions for each label value.

Our work suggests several natural research directions. Might it be possible to improve EM2D using a better generative model whose E-scores are not as close to 0/1 as NB? Might tempering or annealing let EM2D reach better local optima? Are we estimating the number of clusters in EM properly? The "correct" number of EM clusters may be as high as $|A|\,|B|$ (if the taxonomies are truly "orthogonal"), but is generally much smaller (perhaps even smaller than $|A|$ and $|B|$ for poorly separated labels). Can SVM-CT be improved further by designing better kernels? Is it an accident that neither of EM2D and SVM-CT dominates the other in accuracy? If not, can we predict which is likely to do better for a given problem? Can we design cross-trained classifiers which combine the strengths of the distributional and discriminative approaches? Finally, it would be useful to extend the algorithm for real taxonomies (as against flat sets of classes).

Acknowledgments: The research was supported in part by IBM Corporation, Tata Consultancy Services, and the Ministry of Information Technology. We thank Kinshuk Jerath for collecting part of the data and for writing a preliminary version of the A&S algorithm; Ganesh Ramakrishnan, Deepa Paranjpe, and Roger Menezes for helping us clean the data; and an anonymous reviewer for pointing out approaches for learning multiple tasks.

8. REFERENCES

[1] R. Agrawal and R. Srikant. On integrating catalogs. In *World Wide Web*, pages 603–612, 2001.

[2] J. Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research*, 12:149–198, 2000. http://www-2.cs.cmu.edu/afs/cs/project/jair/pub/volume12/baxter00a.pdf.

[3] A. Blum and T. M. Mitchell. Combining labeled and unlabeled data with co-training. In *Computational Learning Theory*, pages 92–100, 1998.

[4] R. Caruana. Multitask learning. *Machine Learning*, 28:41–75, 1997. http://www.cs.cmu.edu/afs/cs.cmu.edu/user/caruana/pub/papers/mlj97.ps.

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

[6] A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic Web. In *WWW*, pages 662–673, Honolulu, Hawaii, May 2002. http://www2002.org/CDROM/refereed/232/.

[7] S. Dumais, J. Platt, D. Heckerman, and M. Sahami. Inductive learning algorithms and representations for text categorization. In *7th Conference on Information and Knowledge Management*, 1998. Online at http://www.research.microsoft.com/~jplatt/cikm98.pdf.

[8] T. Joachims. Making large-scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods: Support Vector Learning*. MIT Press, 1999. See http://www-ai.cs.uni-dortmund.de/DOKUMENTE/joachims_99a.pdf.

[9] W.-S. Li, Q. Vu, D. Agrawal, Y. Hara, and H. Takano. PowerBookmarks: A system for personalizable Web information organization, sharing and management. *Computer Networks*, 31, May 1999. http://www8.org/w8-papers/3b-web-doc/power/power.pdf.

[10] B. Liu, W.-S. Lee, P. S. Yu, and X. Li. Partially supervised classification of text documents. In *ICML*, volume 19, pages 387–394, Sydney, Australia, July 2002.

[11] Y. S. Maarek and I. Z. Ben Shaul. Automatically organizing bookmarks per content. In *Fifth International World-Wide Web Conference*, Paris, May 1996.

[12] A. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. Software available from http://www.cs.cmu.edu/~mccallum/bow/, 1998.

[13] A. McCallum and K. Nigam. A comparison of event models for naive Bayes text classification. In *AAAI/ICML-98 Workshop on Learning for Text Categorization*, pages 41–48. AAAI Press, 1998. Online at http://www.cs.cmu.edu/~knigam/.

[14] K. Nigam, J. Lafferty, and A. McCallum. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999. See http://www.cs.cmu.edu/~knigam/ and http://www.cs.cmu.edu/~mccallum/papers/maxent-ijcaiws99.ps.gz.

[15] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using EM. *Machine Learning*, 39(2/3):103–134, 2000. See http://www-2.cs.cmu.edu/~mccallum/papers/emcat-mlj2000.ps.gz.

[16] J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, 1998. Online at http://www.research.microsoft.com/users/jplatt/smoTR.pdf.

[17] E. S. Ristad. A natural law of succession. Research report CS-TR-495-95, Princeton University, July 1995.

[18] S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: The TC algorithm. In L. Saitta, editor, *Proceedings of the 13th International Conference on Machine Learning ICML-96*, San Mateo, CA, 1996. Morgen Kaufmann.

[19] Y. Yang and X. Liu. A re-examination of text categorization methods. In *Annual International Conference on Research and Development in Information Retrieval*, pages 42–49. ACM, 1999. Available from http://www-2.cs.cmu.edu/~yiming/publications.html.

[20] H. Yu, J. Han, and K. C. Chang. PEBL: Positive example based learning for Web page classification using SVM. In *KDD*, volume 8, Edmonton, Canada, July 2002. http://www-faculty.cs.uiuc.edu/~kcchang/Papers/pebl-kdd02.pdf.