Transferring and Retraining Learned Information Filters

William W. Cohen
AT&T Labs–Research
600 Mountain Avenue
Murray Hill, NJ 07974
wcohen@research.att.com

Daniel Kudenko
Department of Computer Science
Rutgers University
Piscataway, NJ 08855
kudenko@cs.rutgers.edu

Abstract

Any system that learns how to filter documents will suffer poor performance during an initial training phase. One way of addressing this problem is to exploit filters learned by other users in a collaborative fashion. We investigate "direct transfer" of learned filters in this setting—a limiting case for any collaborative learning system. We evaluate the stability of several different learning methods under direct transfer, and conclude that symbolic learning methods that use negatively correlated features of the data perform poorly in transfer, even when they perform well in more conventional evaluation settings. This effect is robust: it holds for several learning methods, when a diverse set of users is used in training the classifier, and even when the learned classifiers can be adapted to the new user's distribution. Our experiments give rise to several concrete proposals for improving generalization performance in a collaborative setting, including a beneficial variation on a feature selection method that has been widely used in text categorization.

Introduction

There is an increasing amount of interest in intelligent personal assistants, and more specifically, in personalized information filters. One commonly proposed technique for creating such filters is via learning from examples (Maes 1994; Mitchell *et al.* 1994; Lang 1995; Pazzani *et al.* 1996; Cohen 1996a). An advantage of using learning is that it enables a naive user to easily construct a filter. A disadvantage is that performance may be poor during the initial training phase. While in principle the cost of this initial period of poor performance can be amortized over a long period of use, in practice it will mean that many users will simply become frustrated with the filtering system and stop using it.

Several researchers have proposed addressing this problem by adding a collaborative element to learned personal assistants (Maes 1994; Mitchell *et al.* 1994). Simply put, it is plausible that the training phase for a new user can be shortened, perhaps substantially, if another user has already trained a sufficiently similar information filter. Transfer among learned classifiers has been demonstrated in several other contexts (Pratt 1996; Caruana 1996; Thrun 1996). Here we will investigate transfer in the context of learning e-mail filters (Cohen 1996a). Although we consider alternative approaches, we focus our study on one very simple transfer technique: *direct transfer*, in which a learned classifier is simply copied from one user to another.

There are several reasons why we have elected, in this paper, to focus on direct transfer. First, direct transfer offers a plausible way that experience from other users can be of *immediate* benefit to a new user of a system. Intuition and experience suggests that many users will be unwilling to invest even a moderate amount of time in training a learning system without seeing any noticeable benefit; therefore the performance of collaborative learning systems given very small numbers of examples from a new user is important. Put another way, direct transfer is a limiting case for more complex transfer methods in which the number of examples labeled by the new user is zero—and this is an interesting special case, since the goal of collaborative learning is to provide some benefit as quickly as possible.

Second, the performance of a learned classifier under direct transfer is a special case of an important general property—the sensitivity of learned classifiers to distributional changes. There are many cases in which the distribution is likely to change markedly (*e.g.*, over time), or where access to the original distribution is problematic (*e.g.*, when only sampled data is available for training). Another instance of a case in which insensitivity to distributional change would be important would be in a networked environment in which learned classifiers are widely distributed. It is not implausible to imagine wishing to train a text classifier using locally-available data, and then distributing the learned classifier to many different sites to look for additional documents in the target class. Of course, the distributions of documents at different sites sites might be quite different–just as one user's distribution of e-mail differs from another's.

Finally, and perhaps most speculatively, the performance of a learned classifier under direct transfer is, we believe, closely related to its *comprehensibility*—a quality much harder to quantify, and therefore harder to study. To elaborate on this point, consider the following two rules of thumb for determining if an e-mail message is a talk announcement.

1. A message is a talk announcement if it contains the words "talk", "abstract" and "time".

2. A message is not a talk announcement if it was sent by Larry.

Most people would probably consider Rule 1 to be comprehensible, as the words that it uses as features are present in most talk announcements for obvious and intuitive reasons. Rule 2 seems harder to understand, even though it also be predictive for some users. (For instance, Larry might be a secretary, a system administrator, or a close colleague in a far-off location; in either of these cases he might send many messages, none of which were ever talk announcements.) The key difference seems to be that Rule 1 can be rationalized on the basis of commonly held, widely accepted knowledge, whereas Rule 2 can only be rationalized on the basis of more esoteric knowledge—namely, knowledge about the distributional properties of a particular user's mail.

To distinguish between rules like 1 and 2 it is clearly not sufficient to look only at measures like syntactic size and predictive power on the distribution from which they arose. However, they can perhaps be distinguished if one tests them on other, different distributions—as is done in direct transfer.[1]

To summarize the remainder of the paper, we will first describe the datasets that we have collected to investigate this problem, and the learning algorithms with which we will experiment. We then present a series of direct transfer experiments, and our interpretation of the results. In short, we hold that in these sorts of user-to-user transfers, it is better to use a learning algorithm that generalizes conservatively from the positive examples—that is, that only classifies a new instance as positive if it is quite similar to some positive example in the training set. One way of encouraging this behavior for symbolic learning algorithms seems to be to only use features that are positively correlated with the training data; this leads to definitions of categories in terms of words that they must contain (like "talk" and "abstract") rather than in terms of words they must not contain (like "Larry").

We will show that violating these principles almost uniformly leads to high error rates in transfer, even when it leads to lower error rates on the training distribution (as estimated by cross-validation.) Extreme violations of the principle (like Rule 2 above) can lead to catastrophic failure in direct transfer—*e.g.*, error rates substantially higher than the default error rate—even when cross-validated error rates on the training distribution are relatively low. More subtle violations of the principle show the same sort of effect, albeit less dramatically. The effect also persists when classifiers are trained on examples from many users instead of just one.

We also investigate a simple re-training approach, in which examples labeled by a new user are augmented with examples labeled by a second user. In this scenario also, the principle stated above seems to be important: while learning systems that adhere to it are generally improved by the addition of the second user's data, learning systems that violate it can be substantially degraded by additional data.

Encouragingly, it is often easy to ensure conservative generalization for symbolic learners such as C4.5. Furthermore, while it is often the case that better performance in direct transfer is obtained at the cost of higher cross-validated error rates on the original distribution; however, this is not uniformly true. A final contribution of this paper is a feature selection method for text that improves performance, both in direct transfer and in cross-validation studies.

Datasets and Learning Algorithms

The datasets

E-mail data. Ishmail is a recently developed customizable e-mail reader. Ishmail allows a user to define an ordered list of *mailboxes*, each of which has an associated binary classifier. Before it is read, an incoming e-mail message is placed in the first mailbox that classifies the message as positive. In addition to classifiers, separate archiving and alerting procedures can be assigned to each mailbox.

In the current implementation of Ishmail, classification rules are explicitly programmed by the user. A typical use of Ishmail is to route messages from high-volume mailing lists into separate mailboxes, which are given appropriate alerting and archiving policies. However, classification rules for many message categories are difficult to define manually, especially for naive users. In such cases, the ability to learn e-mail filters could be helpful. Training data can be generated by a user by manually routing incoming messages into the appropriate mailboxes. Once enough training data is collected, the routing task could be taken over by an learned filter.

In a previous survey of Ishmail users (Cohen 1996a), we found that three users had constructed mailboxes with vacuous or highly approximate classifiers, and manually filed messages into these mailboxes; this was typically done to exploit Ishmail's archiving features. Since these users were all computer scientists that had programmed some classifiers, it is reasonable to assume that these categories are both interesting and hard to program, and therefore suitable test problems for personal e-mail filters.

[1]Of course, there are clearly some aspects of comprehensibility that are not measured by direct transfer. For instance if one re-writes Rule 1 to read "contains the words 'talk', 'abstract', and one of the substrings ':00' or ':30"' then the rule is certainly harder to understand. However, it may perform no worse in direct transfer.

Table 1: The talk announcement problems

Name	Source	# examples	% talks
T1	Kudenko	145	50.34
T2	Ishmail 1	343	25.36
T3	Ishmail 2	3863	2.10
T4	msgs newsgroup	491	19.35
T5	grad newsgroup	1279	22.36
T6	Ishmail 3 (Cohen)	499	9.02

Table 2: The newsgroup problems

Name	Sources for negative class
U1	rec.games.backgammon, sci.classics
U2	rec.food.recipes, sci.classics
U3	rec.games.backgammon, soc.history.science
U4	rec.food.recipes, soc.history.science
U5	rec.games.backgammon, sci.logic
U6	rec.food.recipes, sci.logic

Eleven filters were collected from these users, along with datasets consisting of manually filed (and automatically archived) messages. At the same period of time one of us (Cohen) began using Ishmail in a similar way, creating an additional three filtering problems. Three of the fourteen categories were for some variation of the category of *talk announcements*, suggesting that there is indeed substantial overlap in the filtering needs of different users.

The talk announcement datasets collected from Ishmail users were supplemented by collecting three other sets of messages and manually classifying them as to whether or not they were talk announcements; the general characteristics of these messages are summarized in Table 1. The first set (labeled `msgs` in the table) were the messages posted to a center-wide general-interest electronic bulletin board spanning a period of several years. The second set was all e-mail messages received by one of the authors of this paper (Kudenko) over a four month period. The third set (`grad`) were the messages posted to a departmental graduate student newsgroup over a period of several years.

Newsgroup data. In addition to the talk announcement data, we generated artificial datasets from several Usenet newsgroups. Our goal was to simulate users with a single common filtering need but different distributions of e-mail messages. In addition to serving as an additional test of our conclusions, these datasets can also be made publicly available, whereas the talk announcement data contains many private messages.

Six different "users"[2] were generated using postings from six different newsgroups as shown in table 2. The target category (*i.e.*, the source for positive training data) for all six users is the `comp.os.ms-windows.misc` newsgroup. The newsgroup messages were equally divided, yielding datasets containing between 328-382 examples, with between 37-44% of these in the target class.

[2] We will call the synthetic datasets "users" even though they represent only simulated users.

The partial overlap in the negative categories simulates an overlap in interests. This overlap can be varied, and since many of our experiments would have rather predictable conclusions if there were no overlap for the negative class, we elected to introduce somewhat more overlap than was observed in the `talk` datasets. (However, note that the overlap in the datasets is with respect to subject matter only—the messages are all distinct.)

Learning algorithms

Two different learning algorithms are used in this paper. One is C4.5 (Quinlan 1994). The other is the rule learning algorithm RIPPER, which is described in detail elsewhere (Cohen 1995). Briefly, RIPPER builds a ruleset using a separate-and-conquer algorithm, coupled with combination of cross-validation and minimum-description length techniques to prevent overfitting. The basic separate-and-conquer strategy is based on that used by FOIL (Quinlan 1990; Quinlan and Cameron-Jones 1993), and the pruning algorithm is an extension of Fürnkranz and Widmer's *incremental reduced error pruning* method (Fürnkranz and Widmer 1994).

RIPPER also has one extension, which is important for text categorization problems. In RIPPER, every example is represented as a feature vector; however, the value of a feature can be a *set* of symbols, instead of a single symbolic value or a number (Cohen 1996b). This means that a structured document can be easily and naturally represented; for example, e-mail messages, in our experiments with RIPPER, were represented with four attributes, `from`, `to`, `subject`, and `body`. The value of each attribute is the set of all words that appear in the corresponding section of the mail message. The primitive tests on a set-valued attribute a (*i.e.*, the tests which are allowed in rules) are of the form "$w_i \in a$". As a user-controlled option, tests of the form "$w_i \notin field$" can also be used.

On a k-class learning problem, RIPPER first orders the classes. Next, a set of rules is constructed that separates the first class from the remaining classes, and all covered instances are removed. This process is repeated for each of the first $k-1$ classes, with the final class c_k becoming the default class (*i.e.*, an example is classified as class c_k only if no learned rule fires on it.) By default, classes are ordered from least frequent to most frequent, but the user can specify an alternative ordering.

Transfer of Learned Filters

Two differing classifiers

Most of our transfer experiments involved two users, the *source user* and the *destination user*. In these experiments, a classifier was learned from the examples

Table 3: Differing performance of learned classifiers under direct transfer

Source User	RIPPER, options 1 Destination User T1	T2	T4	T5	T6	RIPPER, options 2 Destination User T1	T2	T4	T5	T6
T1	**17.9**	18.1	7.7	8.2	5.4	**19.8**	72.6	79.6	58.8	79.8
T2	13.1	**5.6**	7.7	18.6	5.0	52.4	**23.3**	57.4	54.6	40.7
T4	21.4	15.5	**5.1**	9.7	4.0	50.4	25.4	**19.4**	22.4	9.0
T5	7.6	11.7	6.5	**2.4**	4.4	33.1	55.7	53.8	**9.3**	66.7
T6	8.3	10.5	4.9	6.7	**4.3**	50.4	25.4	19.4	22.4	**9.0**
Default	49.7	25.4	19.4	22.4	9.0	49.7	25.4	19.4	22.4	9.0
T∗	8.3	11.1	5.7	10.7	4.4	49.7	27.4	42.0	22.4	42.7

labeled by the source user, and then tested on the examples labeled by the destination user. Experiments were carried out both on the `talks` datasets, and on the `news` datasets.

The sometimes dramatic differences in performance under direct transfer are emphasized by the results shown in Table 3. The first table shows the error rate (in percent) of classifiers learned by RIPPER on all possible direct transfer experiments between five users on the talks datasets.[3] The diagonal entries (shown in bold) give the generalization error of the learned classifier for the corresponding user, as estimated by 10-fold cross-validation. A single user-controlled option of RIPPER was varied, resulting in two different classifiers (see details below). (The source labeled "T∗" will be described below.)

As measured by cross-validation, the second set of options seems to perform rather worse than the first: it is competitive only on problem `T1`, and only outperforms the default error rate[4] on one other problem (`T5`). However, when direct transfer is used, the classifiers learned with the second set of options are catastrophically worse than those learned with the first set of options. One error rate is ten times as large; further, most of the transfers with the second set of options result in an error rate above, or at best comparable to, the default accuracy, whereas all transfers with the first set of options result in an error rate below the accuracy of the default classifier.

In Table 3, the single option of RIPPER that was varied is the ordering of classes. In the first set of options, rules were learned for the positive class (talk announcements), and the negative class was the default (*i.e.*, a message was classified as negative if none of the learned talk announcement rules fired.) In the second set of options, rules were learned for the negative class of non-talk-announcement messages, and the positive class was the default. The interpretation of the differences in cross-validated error rate between these two learners is thus fairly obvious: presumably, the class of non-talk-announcements is highly disjunctive for most users, and hence harder to learn.

Our interpretation of the transfer results is that the distributions of positive examples from user to user are relatively similar, but the distribution of negative examples is relatively dissimilar. This situation is illustrated in Figure **??**, which shows graphically two similar distributions of positive examples (henceforth P_1^+ and P_2^+) and two dissimilar distributions of negative examples (henceforth P_1^- and P_2^-). In such a situation it would be desirable for a classifier to generalize conservatively from the positive examples, and classify all other instances as negative: since the positive distribution does not change dramatically, these conservative generalizations would tend to be accurate. On the other hand, a classifier which tends in the other direction and generalizes conservatively from the negative examples might be quite inaccurate when tested on examples drawn from a different distribution (*i.e.*, a different user).

The rule set learned by RIPPER under the first set of options is (modulo pruning) simply a set of conjunctions that together cover the positive examples. This tends to be a fairly conservative generalization of the positive examples—if an example is classified as positive, it must share a number of important keywords with a number of positive training examples. A set of rules that covers the negative examples is not conservative in the same sense—a new instance can be classified as positive even if it is quite dissimilar from all of the positive training examples.

Further results with direct transfer

Several additional experiments were conducted, which generally confirm the thesis that classifiers that are more conservative for the positive class transfer more reliably. These results are summarized in Table 4.

[3]Dataset T3 (Ishmail User 2's talk category) was different semantically from the other categories, also has a markedly different proportion of positive examples; this category was for announcements in a local talk series, rather than for all talk announcements. Because of these differences we did not use it in the direct transfer experiments.

[4]The default classifier always predicts the majority class.

Table 4: Further results with direct transfer

Learner	Topic	Transfers W-L-T	Aggregate W-L-T	10-CV W-L-T	Error Ratios 10-CV	Error Ratios Transfer	
RIPPER, negative class	talks	0-20-0	482-4357-6189	0-5-0	3.00	5.78	√
	news	0-30-0	1123-3292-5975	2-4-0	1.19	2.48	√ !
RIPPER, both classes	talks	6-8-6	93-454-10481	2-3-0	1.16	1.61	√ !
	news	22-8-0	192-94-10104	6-0-0	0.86	0.95	√
RIPPER, negative tests	talks	8-14-8	582-635-31881	3-2-0	0.87	1.32	√ !
	news	9-20-1	662-1478-8250	2-4-0	1.12	1.58	√ !
C4.5/100 + correlation	talks	11-5-4	734-214-10080	4-1-0	0.72	0.87	
	news	18-10-2	1563-596-8231	4-2-0	0.96	0.82	√
C4.5/100 − correlation	talks	0-16-4	544-3556-6928	0-4-1	2.46	4.49	√
	news	1-22-7	813-2454-7123	2-4-0	1.12	2.25	√ !
C4.5/10 + correlation	talks	15-2-3	646-99-10283	4-1-0	0.50	0.70	
	news	30-0-0	4192-975-5223	6-0-0	0.47	0.35	√
C4.5/10 − correlation	talks	0-8-12	105-1854-9069	0-2-3	1.60	2.66	√
	news	4-6-20	500-882-9008	1-1-4	1.07	1.17	√
RIPPER, negative class	talks∗	0-5-0	65-699-1993	0-5-0	3.00	5.52	√
	news∗	1-5-0	189-323-1566	2-4-0	1.19	1.81	√ !
C4.5/100 + correlation	talks∗	5-0-0	255-104-2398	4-1-0	0.72	0.56	√
	news∗	6-0-0	401-72-1605	4-2-0	0.96	0.42	√
C4.5/100 − correlation	talks∗	0-5-0	149-458-2150	0-4-1	2.46	1.47	
	news∗	2-3-1	235-287-1556	2-4-0	1.12	1.14	√

Because of space consideration we give only a few summary statistics, measuring the performance relative to an appropriate baseline learning system (Ripper with options 1 and C4.5 with default options, respectively). Specifically, for a learner L we give the total number of transfer experiments for which L's hypothesis "wins" by having a lower error rate than that of the base line; the number of transfers that L's hypothesis "loses", and the number of transfers that are tied. We also report the number of wins, losses, and ties on the individual predictions. (*E.g.*, the number of aggregate wins for L is the total number of individual messages, across all transfer experiments, for which L's hypothesis disagrees with the hypothesis of the baseline and is correct.) To summarize the performance of L as measured by cross-validation, we report a won-loss-tied summary of the performance of L versus the baseline, as well as the average ratio of L's cross-validated error rate to that of the baseline. In the penultimate column we report the average ratio of L's error rate to the baseline across all transfer experiments.

The final column will be checked (√) if the result generally confirms the hypothesis that conservative generalization from the positive examples yields a higher improvement of performance than expected, given the cross-validation results from table 3. We also give an exclamation point in this column if the result is "surprising"—*i.e.* if the performance under transfer is much worse than would be expected from the cross-validated error rate.

Rules for both negative and positive class. Some rule learning systems, such as C4.5rules (Quinlan 1994), do not order classes as RIPPER does; instead, rules are learned for every class. To classify an example, all the rules are tested. If no rule fires, then an appropriate default class is suggested; otherwise, if more than one class is suggested by the rules that do fire, some sort of weighted voting scheme is used to resolve the conflict. We implemented a simple version of this learning strategy for RIPPER. Rules were learned for both the positive and negative class, conflicts were broken in favor of the single rule that was most accurate on the training data[5], and the default class was chosen to be the most prevalent class among the training examples not covered by any rule.

Experiments we have conducted in more conventional settings suggest that this "unordered classes" variant of RIPPER is almost always competitive with choosing the default order, and occasionally substantially better. On the talk problems unordered RIPPER is better on 2 of the 5 problems with respect to cross-validated error, and uniformly better in the news data.

Unordered RIPPER is clearly less conservative than the baseline of learning rules for the positive class alone, and on the talks data, it is clearly worse than the baseline: although it usually agrees with the baseline, when it disagrees it is incorrect more than 80% of

[5] More precisely, the rule with lowest error as estimated by the Bayes-Laplace formula $(e+1)/(n+2)$.

the time. On the news data, it does yield an improvement over the baseline, but somewhat less than would be expected from the cross-validation rates.

Rules containing negated conditions. RIPPER can optionally learn rules containing conditions of the form $w \notin A$, where w is a word and A is a feature. (For example, "meeting$\notin$Subject".) This variant is again competitive in conventional learning settings, and in fact improves performance on average on a number of benchmarks (Cohen and Singer 1996). However, the negative conditions in the rule make it possible to construct more general rules by explicitly excluding some portion of the set of negative examples. Because of this, the test examples covered by these rules need not be as similar to the training examples covered by the rules, and hence this variant is less conservative than the baseline. Hence it performs worse, on average, in direct transfer—surprisingly so, given that it is at least competitive on both sets of problems with respect to cross-validated error rate.

In all of the symbolic learning methods with which we have experimented, the use of negatively correlated words as features has been instrumental in allowing the learning systems to generalize less conservatively; without these features, the hypotheses constructed must be more or less similar to those of the baseline RIPPER system. Thus a secondary conclusion of this paper is that learning methods that use negatively correlated features of the data perform poorly in transfer—at least for symbolic learning methods such as RIPPER and C4.5.

Restricted feature selection for C4.5. With a separate and conquer rule learner such as RIPPER, it is possible to choose which examples will be covered by rules; in contrast, it is not immediately obvious how a decision-tree learning system such as C4.5 can be modified so as to make it conservative in the same sense. However, it turns out that an appropriate feature selection strategy can be used to force (or at least, strongly encourage) a decision tree learner to output a more conservative hypothesis.

A frequently-used method for selecting features for a standard propositional learner is to take the k words that have the highest mutual information with the class (Lewis and Ringuette 1994; Apté *et al.* 1994). By selecting only words that are positively correlated with the target class, one can virtually force a decision tree learner to generate trees such that leaves label with the positive class can only be reached by a sequence of "true" conditions.[6] Such a tree can be converted to a ruleset of the form produced by the baseline RIPPER system above: all rules assert membership in the positive class, and all conditions check that a word is present in some field.

As our baseline, we used C4.5 (Quinlan 1994) using k binary attributes selected according to mutual information, for $k = 100$ and for $k = 10$. We compared this to selecting the best k positively-correlated attributes, and the best k negatively-correlated attributes. Using positively correlated attributes only leads to an improvement in transfer. Using negatively correlated attributes only leads to dramatic degradation in transfer performance; again, the degradation is far worse than would be expected from the cross-validated error rates. As might be expected, the effect of varying the feature selection method is stronger when k is smaller.

One unexpected result is that use of positively correlated features also improves the cross-validated error rate—often even more than it improves the error rate under transfer. This may be a result of the fact that we are not simply removing negatively correlated features but also adding additional positively correlated features to the set of k best features used by C4.5; in short the experiment may be confounding an improvement due to better feature selection with an improvement due to selecting more features. We note that in other experiments with text categorization using decision trees, larger feature sets generally reduce classification error (Lewis and Ringuette 1994; Cohen 1996b).

Transfer from multiple users. One might expect that transfer might be more reliable, regardless of the learning method, if the training distribution was larger and more diverse. To test this, we conducted a second variety of direct transfer experiment, in which a single user was chosen as the destination, and a combination of all the other user's datasets were used as the source. In other words, we combined the training examples for the first $k-1$ users, learned a hypothesis from this combined dataset, and then tested on the remaining user's examples.

The results are shown in Table 4 in the rows labeled with the topics `talk*` and `news*`. Interestingly, the differences among the learners under this type of transfer were equally consistent, and of nearly the same order of magnitude as when transferring from a single user.

To present a final result from this experiment, the row labeled "T*" in Table 3 indicates the actual error rates for two variants of RIPPER in performing this sort of transfer. Interestingly, combining examples from several users does not improve performance over transfer from a single user as much as one might hope; for instance, the multiple-user transfer is never as good as the best single-user transfer.

Retraining of learned filters. Another way of making use of previous user's learned classifiers is to take a small set of examples classified by a new user and augment it with a larger set of examples classified by other users. We call this procedure *re-training*, and will use it as a (very simple) case of an collaborative learning approach that actually adapts to the new user's distribution. In this section we will investigate how re-training performs—in particular if the same effects present in direct transfer also manifest themselves

[6] We say "virtually force" because there is some possibility that a condition that is positively correlated over the whole dataset will be negatively correlated within a subtree. This appears to happen only rarely.

Table 5: Retraining error rates on talks data

Source User	k = 40 Destination User T1	T2	T4	T5	T6	k = 60 Destination User T1	T2	T4	T5	T6
T1	**17.9**	15.4	6.8	9.0	6.2	**17.9**	13.1	7.8	7.7	5.4
T2	9.6	**5.6**	5.9	7.9	3.5	7.9	**5.6**	5.7	5.9	4.1
T4	13.0	12.4	**5.1**	10.9	4.9	7.8	9.9	**5.1**	8.3	4.3
T5	6.2	11.5	6.0	**2.4**	5.8	5.2	12.1	6.8	**2.4**	4.8
T6	7.7	10.9	5.0	6.7	**4.3**	6.6	10.9	5.0	6.7	**4.3**
Default	49.7	25.4	19.4	22.4	9.0	49.7	25.4	19.4	22.4	9.0

Table 6: Results with retraining

Learner	Topic	k	Won-loss-tied Transfers	Aggregate
RIPPER, positive class	talks	20	20-0-0	6181-2485-4574
		40	18-1-1	4564-2247-47103
		60	17-3-0	3039-1977-48401
	news	20	30-0-0	12993-2493-33499
		40	30-0-0	9309-2326-36099
		60	30-0-0	6811-2249-37479
RIPPER, negative class	talks	20	1-13-6	2109-4910-47388
		40	1-15-4	2665-4847-4640
		60	3-13-4	2107-3492-4781
	news	20	24-6-0	13076-10192-2571
		40	26-4-0	9965-7595-30174
		60	24-5-1	7570-6230-32739

in retraining.

In the experiments, k training examples were randomly moved from the destination dataset to the source dataset, and the augmented source dataset was then used to learn a classifier which was tested on the reduced destination dataset. We tested retraining using $k = 40$ and $k = 60$ destination examples from each class. Since the selection is random, we repeated each test five times and averaged.

Table 6 shows the result of pairing the classifiers learned with transfer and k retraining examples against a baseline in which only the k retraining examples were used. This comparison was made for both the "transfer friendly" version of RIPPER, which learns the positive class, and the "transfer unfriendly" version of RIPPER, which learns the negative class.

For "positive" RIPPER, performance generally improves substantially when the source data is used. The improvement is most marked when k is small, suggesting that simple re-training can be effective. For negative RIPPER, the addition of the source data frequently increases error rates. This effect occurs only occasionally on the news data (where there is often substantial overlap in the negative examples for source and destination datasets) but is pronounced on the talks data. This suggests that stability under transfer is also important in more adaptive collaborative settings.

Table 5 shows the actual error rates on the talks data; again the bold-faced numbers on the diagonal are cross-validated error rates for the destination set. Comparing $k = 40$ and $k = 60$ (and $k = 0$ from Table 3) it is easy to see that the results show a considerable improvement in accuracy as more re-training examples are used. In many cases only 40 retraining examples are enough to give a significantly classifier, confirming that simple re-training can be a practical collaborative training method—for appropriate learning methods.[7]

The observation that "transfer-unfriendly" learning methods can actually show an increased error rate given a greatly enlarged training set may seem quite surprising. However this result is prefigured by the results of Table 3, which show that training on a distribution different from the destination dataset can lead to performance much worse than even random guesssing.

Conclusions

Transfer of learned information filters can be beneficial if different users of a filtering system use similar categories. We evaluated two inductive learning systems, RIPPER and C4.5rules, on two sets of filtering problems. Our experiments concentrated on *direct transfer*, in which a learned classifier is trained on data labeled by one user (the source dataset) and then used to classify another user's data (the destination dataset.) We argued that stability under direct transfer is connected to comprehensibility, as well as being an important limiting case for collaborative learning approaches.

The principle conclusion of this paper is that stability under direct transfer is improved when classifiers generalize conservatively with respect to the positive examples. For symbolic learners, one way to encourage this is to avoid use of negatively correlated features; learners that use these features tend to perform worse in transfer, even when they perform well in more conventional evaluation settings. This effect can some-

[7] Results on the newsgroup data are similar, except that the differences are more consistent, presumably because of the greater uniformity of the datasets.

times lead to catastrophically bad transfer, even when cross-validated error rates are reasonable. The effect is also robust in that it persists when the source dataset is large and diverse, and when the source dataset is augmented by examples from the destination.

References

Chidanand Apté, Fred Damerau, and Sholom M. Weiss. Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems*, 12(3):233–251, 1994.

Rich Caruana. Algorithms and applications for multitask learning. In *Machine Learning: Proceedings of the Thirteenth International Conference*, Bari, Italy, 1996. Morgan Kaufmann.

William W. Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. In *Proceedings of the 19th Annual International ACM Conference on Research and Development in Information Retrieval*, pages 307–315, Zurich, Switzerland, 1996. ACM Press.

William W. Cohen. Fast effective rule induction. In *Machine Learning: Proceedings of the Twelfth International Conference*, Lake Tahoe, California, 1995. Morgan Kaufmann.

William W. Cohen. Learning rules that classify e-mail. In *Proceedings of the 1996 AAAI Spring Symposium on Machine Learning and Information Access*, Palo Alto, California, 1996.

William W. Cohen. Learning with set-valued features. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Portland, Oregon, 1996.

Johannes Fürnkranz and Gerhard Widmer. Incremental reduced error pruning. In *Machine Learning: Proceedings of the Eleventh Annual Conference*, New Brunswick, New Jersey, 1994. Morgan Kaufmann.

Jonathan Isaac Helfman and Charles Lee Isbell. Ishmail: Immediate identification of important information. Submitted to CHI-96, 1995.

Ken Lang. NewsWeeder: Learning to filter netnews. In *Machine Learning: Proceedings of the Twelfth International Conference*, Lake Taho, California, 1995. Morgan Kaufmann.

David Lewis and Mark Ringuette. A comparison of two learning algorithms for text categorization. In *Symposium on Document Analysis and Information Retrieval*, Las Vegas, Nevada, 1994.

Patty Maes. Agents that reduce work and information overload. *Communications of the ACM*, 37(7):31–40, July 1994.

Tom Mitchell, Rich Caruana, John McDermott, and David Zabowski. Experience with a learning personal assistant. *Communications of the ACM*, 37(7), July 1994.

Michael Pazzani, Jack Muramatsu, and Daniel Billsus. Syskill & webert: identifying interesting web sites. In *Proceedings of the Thirteenth National Conference on Artificial Intelligence*, Portland, Oregon, 1996.

Lorien Y. Pratt. Discriminability-based transfer between neural networks. In *Advances in Neural Information Processing Systems (NIPS) 5*, pages 204–211, Denver, Colorado, 1996. Morgan Kaufmann.

J. R. Quinlan and R. M. Cameron-Jones. FOIL: A midterm report. In Pavel B. Brazdil, editor, *Machine Learning: ECML-93*, Vienna, Austria, 1993. Springer-Verlag. Lecture notes in Computer Science # 667.

J. Ross Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3), 1990.

J. Ross Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann, 1994.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? In *Advances in Neural Information Processing Systems (NIPS) 8*, Denver, Colorado, 1996. Morgan Kaufmann.