Journal of Machine Learning Research 7 (2006) 1655–1686 Submitted 12/05; Revised 4/06; Published 8/06

Active Learning with Feedback on Both Features and Instances

Hema Raghavan[*] HEMA@CS.UMASS.EDU
140 Governor's Drive
University of Massachusetts
Amherst, MA 01003, USA

Omid Madani MADANI@YAHOO-INC.COM
Rosie Jones JONESR@YAHOO-INC.COM
Yahoo! Research
3333 Empire Ave, Burbank
CA 91504, USA

Editor: Isabelle Guyon

Abstract

We extend the traditional active learning framework to include feedback on features in addition to labeling instances, and we execute a careful study of the effects of feature selection and human feedback on features in the setting of text categorization. Our experiments on a variety of categorization tasks indicate that there is significant potential in improving classifier performance by feature re-weighting, beyond that achieved via membership queries alone (traditional active learning) if we have access to an *oracle* that can point to the important (most predictive) features. Our experiments on human subjects indicate that human feedback on feature relevance can identify a sufficient proportion of the most relevant features (over 50% in our experiments). We find that on average, labeling a feature takes much less time than labeling a document. We devise an algorithm that interleaves labeling features and documents which significantly accelerates standard active learning in our simulation experiments. Feature feedback can complement traditional active learning in applications such as news filtering, e-mail classification, and personalization, where the human teacher can have significant knowledge on the relevance of features.

Keywords: active learning, feature selection, relevance feedback, term feedback, text classification

1. Introduction

Automated text categorization has typically been tackled as a supervised machine learning problem (Sebastiani, 2002; Lewis, 1998). The training data should be fairly representative of the test data in order to learn a fairly accurate classifier. In document classification where categories can be as broad as *sports*, this means that a large amount of training data would be needed. The training data is often labeled by editors who are paid to do the job. Now consider a scenario where a user wants to organize documents on their desktop into categories of their choice. The user might be willing to engage in some amount of interaction to train the system, but may be less willing to label as much data as a paid editor. To build a generic text categorization system that could learn almost arbitrary categories based on an end user's changing needs and preferences, for example in applications such as news filtering and e-mail classification, the system should extract a large number of features. In

∗. This work was done in part when the author was at Yahoo! Research.

e-mail classification for example, any subset of the features extracted from the subject, the sender, and the text in the body of the message could be highly relevant. While algorithms such as Winnow (Littlestone, 1988) and Support Vector Machines (SVMs) (Joachims, 1998) are robust in the presence of large numbers of features, these algorithms still require a substantial amount of labeled data to achieve adequate performance.

Techniques such as active learning (Cohn et al., 1994), semi-supervised learning (Zhu, 2005), and transduction (Joachims, 1999) have been pursued with considerable success in reducing labeling requirements. In the standard active learning paradigm, learning proceeds sequentially, with the learning algorithm actively asking for the *labels* (categories) of some instances from a teacher (also referred to as membership queries). The objective is to ask the teacher to label the most informative instances in order to reduce labeling costs and accelerate the learning. Still, in text categorization applications in particular, active learning might be perceived to be too slow, especially since the teacher may have much prior knowledge on relevance of features for the task. Such knowledge may be more effectively communicated to the learner than mere labeling of whole documents. There has been very little work in supervised learning in which the teacher is queried on something other than whole instances.

One possibility is to ask the user questions about features. That users have useful prior knowledge which can be used to access information is evident in information retrieval tasks. In the information retrieval setting, the user issues a query, that is, states a few words (features) indicating her information need. Thereafter, feedback which may be either at a term or at a document level may be incorporated. In fact, even in traditional supervised learning, the editors may use keyword based search to locate the initial training instances [1]. However, traditional supervised learning tends to ignore this knowledge of features that the user has, once a set of training instances have been obtained. In experiments in this paper we study the benefits and costs of feature feedback via humans on active learning.

We try to find a marriage between approaches to incorporating user feedback from machine learning and information retrieval and show that active learning should be a twofold process – at the term-level and at the document-level. We find that people have a good intuition for important features in text classification tasks, since features are typically words, and the categories to learn may often be approximated by some disjunction or conjunction of a subset of the features. We show that human knowledge on features can indeed increase active learning efficiency and accelerate training significantly in the initial stages of learning. This has applications in e-mail classification and news filtering where the user has knowledge of the relevance of features and a willingness to label some (as few as possible) documents in order to build a system that suits her needs.

This paper extends our previous work in employing such a two-tiered approach to active learning (Raghavan et al., 2005). We state the active learning problems that we address and present our approach to use feedback on both features and instances to solve the problems in Section 2. We give the details of the implementations in Section 3. In Section 4 we describe the data and metrics we will use to evaluate the performance of active learning. We obtain a sense of the extent of the improvement possible via feature feedback by defining and using a feature oracle. The oracle and the experiments are described in Section 2, and the results are reported in Section 5. In section 6 we show that humans can indeed identify useful features. Furthermore, we find that labeling a feature

1. See `http://projects.ldc.upenn.edu/TDT4/Annotation/label_instructions.html`. The annotators at the LDC (Linguistic Data Consortium, home-page: `http://ldc.upenn.edu`) use a combination of techniques like nearest neighbors and creative search to annotate corpora for the Topic Detection and Tracking (Allan, 2002) task.

takes one fifth of the time of labeling a document. In Section 6.2 we show that the human-chosen features significantly accelerate learning in experiments that simulate human feedback in an active learning loop. We discuss related work in Section 7 and conclude in Section 8.

Standard Active Learning

Input: T (Total number of feedback iterations), $\mathcal{U}$ (Pool of unlabeled instances), init_size (number of random feedback iterations)
Output: $\mathcal{M}_T$ (Model)

$t = 1$; $\mathcal{U}_0 = \mathcal{U}$; $\mathcal{M}_0$ =NULL;
1. While $t \leq$ init_size
   a. $\langle X_t, \mathcal{U}_t \rangle$ = Instance_Selection($\mathcal{M}_0$, $\mathcal{U}_{t-1}$, *random*)
   b. Teacher assigns label Y_t to X_t
   d. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   c. $t++$
2. While $t \leq T$
   a. $\langle X_t, \mathcal{U}_t \rangle$ = Instance_Selection($\mathcal{M}_{t-1}$, $\mathcal{U}_{t-1}$, *uncertain*)
   b. Teacher assigns label Y_t to X_t
   c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   d. $t++$
Return $\mathcal{M}_T$

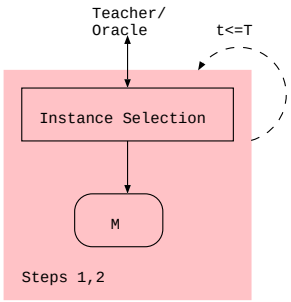


Figure 1: Algorithm and block diagram for traditional active learning where the system asks for feedback on instances only (**System 1**).

2. Active Learning

For background on the use of machine learning in automated text categorization as well as active learning, we refer the reader to the works of Sebastiani (2002) and Lewis and Catlett (1994). Active learning techniques are sequential learning methods that are designed to reduce manual training costs in achieving adequate learning performance. Active learning methods reduce costs by requesting training feedback selectively and intelligently from a *teacher*. The teacher is a human in the text categorization domain. The teacher may also be called the *user*, especially when the teacher training the model is the same as the person using it, for example a user who is training a personalized news filtering system. Traditionally in active learning the teacher is asked *membership queries* which are questions on the class labels or categories of selected instances (documents in our case).

The teacher is sometimes referred to as an oracle in the literature (Baum and Lang, 1992). We will also use the term oracle to refer to a source that gives feedback on instances and/or features, but in this paper we make a distinction between teacher and oracle. We will reserve the term teacher or user to refer to a real human, whose feedback may not be perfect, and we use the term oracle to refer to a source whose feedback is (close to) perfect for speeding active learning. See Section 2.1 for a longer discussion of the distinction between the two.

A typical algorithm for active learning and a block diagram are shown in Figure 1. An *instance* X (which is a document in our case) belongs to a *class* Y. X is represented as a vector $x_1 ... x_N$ of *features*, where N is the total number of features. The features we use for documents are words, bi-grams (adjacent pairs of words) and tri-grams (adjacent triples of words), since these have consistently been found to work well for topic classification. The value of x_j is the number of occurrences of term i in document X. We work on binary *one-versus-rest* classification. Therefore the value of Y for each learning problem of interest is either -1 or 1, signaling whether the instance belongs to the category of interest, or not. An instance in the document collection is *unlabeled* if the algorithm does not know its *label* (Y value). The active learner may have access to all or a subset of the unlabeled instances. This subset is called the *pool* (denoted by $\mathcal{U}$).

Active Learning Augmented with Feature Feedback

Input: T (Total number of feedback iterations), $\mathcal{U}$ (Pool of unlabeled instances), init_size (number of random feedback iterations)
Output: $\mathcal{M}_T$ (Model)

$t = 1$; $\mathcal{U}_0 = \mathcal{U}$; $\mathcal{M}_0$ =NULL;
1. While $t \leq$ init_size
   a.$\langle X_t, \mathcal{U}_t \rangle$ = Instance_Selection($\mathcal{M}_0$, $\mathcal{U}_{t-1}$, *random*)
   b. Teacher assigns label Y_t to X_t
   c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   d. $t++$
2. While $t \leq T$
   a. $\langle X_t, \mathcal{U}_t \rangle$=
      Instance_Selection($\mathcal{M}_{t-1}$, $\mathcal{U}_{t-1}$, *uncertain*)
   b. Teacher assigns label Y_t to X_t
   c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   d. i. $\{F1, ..., F_f\}$ = Feature_Selection($\mathcal{M}_t$, $\mathcal{U}_t$)
      ii. Teacher selects $\{F_1, .., F_k\} \subseteq \{F_1, ..., F_f\}$
   e. Incorporate_Feature_Feedback($\mathcal{M}_t$, $\{F_1, ..., F_k\}$)
   c. $t++$
Return $\mathcal{M}_T$.

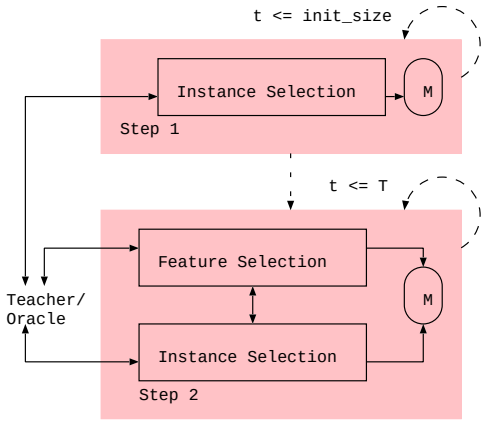


Figure 2: An active learning system where feedback on features is also requested (**System 2**).

The algorithm begins by training the classifier or model $\mathcal{M}$ on some initial set of labeled instances of size *init_size*. The subscript t on $\mathcal{M}$, $\mathcal{U}$, X and Y correspond to the value when t instances have been labeled. The initial set is picked by a random sampling procedure (step 1) from $\mathcal{U}$. The parameter *random* is passed to it. Sometimes one may use keyword based search or some other procedure in place of random sampling. Next, active learning begins. In each iteration of active learning the learner selects an instance from $\mathcal{U}$ using some criterion (*e.g.*, a measure of informativeness) and asks the teacher to label it (step 2.a). In a popular active learning method, called uncertainty sampling, the classifier selects the most *uncertain* instance (Lewis and Catlett, 1994), for a given model (M) and a pool of unlabeled instances ($\mathcal{U}$). The newly labeled instance is added to the set of labeled

instances and the classifier is retrained (step 2.c). The teacher is queried a total of T times. The *train_classifier* subroutine uses the labeled data as training, as well as the model ($\mathcal{M}$) learned in a previous iteration, allowing for the case of incremental training (Domeniconi and Gunopulos, 2001) or the case when the model may be initialized by prior knowledge (Wu and Srihari, 2004).

We will also consider the variant in which instances are picked uniformly at random in all iterations, which we call *random sampling* (it is equivalent to regular supervised learning on a random sample of data). In the pseudo-code in Figure 1, random sampling corresponds to the case when $init\_size > T$.

2.1 Our Proposal: Feature Feedback and Instance Feedback in Tandem

In this paper we propose to extend the traditional active learning framework to engage the teacher in providing feedback on features in addition to instances. A realization of this idea is system 2 shown in Figure 2, where the active learner not only queries the teacher on an informative document, but also presents a list of f features for the teacher to judge (step 2.d) at each iteration. The simplest implementation of such a system can consist of one where $f = |X|$ (the length of the document X), and where the user is simply asked to highlight relevant words or phrases (features) or passages while reading the document in order to label the document (step 2b), akin to the system in the paper by Croft and Das (1990). In our experiments, individual features are presented to the user for selection. Section 6.3 provides the details of our method.

In our proposed system the teacher is asked two types of questions: (1) membership queries and (2) questions about the relevance of features. A relevant feature is highly likely to help discriminate the positive class from the negative class. In this paper we aim to determine whether a human teacher can answer the latter type of question sufficiently effectively so that active learning is accelerated significantly. A human and a classifier probably use very different processes to categorize instances. A human may use her understanding of the sentences within the document, which probably involves some reasoning and use of knowledge, in order to make the categorization decision, while a (statistical) classifier, certainly of the kind that we use in this paper, simply uses patterns of occurrences of the features (phrases). Therefore, it is not clear whether a human teacher can considerably accelerate the training of a statistical classifier, beyond simple active learning, by providing feedback on features.

Before we address that issue, we determine whether feature feedback can accelerate active learning in an idealized setting. We seek to get a sense of the room for improvement. We will then examine how actual human teachers can approximate this ideal. Towards this goal we define a (feature) *oracle*. We use the oracle to obtain an upper bound on the performance of our proposed two-tiered approach. The oracle knows the correct answer needed by the learning algorithm. For example the word *ct* is a highly relevant feature for classifying Reuters news articles on the *earnings* category and our oracle would be able to determine that this feature is relevant when asked. However, a teacher (human) who did not understand that *ct* stood for *cents* may not be able to identify *ct* as relevant (we will see this exact example in Section 6.1). Therefore, the oracle and teacher may differ in their answers to questions about features, that is, questions of type (2) above. We assume that the oracle and the teacher always agree on the labels of documents that is, questions of type (1) above. After showing the usefulness of oracle feature selection, we will then show that humans can emulate the oracle for feature feedback to an extent that results in significant improvements over traditional active learning.

2.2 Extent of Speed Up Possible: Oracle Experiments

We perform two types of experiments with the oracle. In the first kind, the oracle, knowing the allotted time T, picks the best subset of features to improve, as much as possible, the performance of active learning. The procedure is shown in Figure 3. In Figure 3, the *Incorporate_Feature_Feedback* subroutine is called to initialize the model. When System 3 is used with a user instead of the oracle it is equivalent to a scenario where prior knowledge is used to initialize the classifier (Schapire et al., 2002; Wu and Srihari, 2004; Godbole et al., 2004; Jones, 2005). In Section 3.4 we describe how this oracle is *approximated* in our experiments.

Use of Feature Feedback Before Active Learning

Input: T (Total number of feedback iterations), $\mathcal{U}$ (Pool of unlabeled instances), *init_size* (number of random feedback iterations)
Output: $\mathcal{M}_T$ (Model)

$t = 1$; $\mathcal{U}_0 = \mathcal{U}$; $\mathcal{M}_0$ =NULL;
1.a. $\{F1,...,F_f\}$ = Feature_Selection($\mathcal{U}_0$)
  b. Oracle selects $\{F1,..,F_k\} \subseteq \{F_1,...,F_f\}$
2.Incorporate_Feature_Feedback($\mathcal{M}_0$, $\{F_1,...,F_k\}$)
3. While $t \leq$ init_size
  a. $\langle X_t, \mathcal{U}_t \rangle$=Instance_Selection($\mathcal{M}_{t-1}$, $\mathcal{U}_{t-1}, random$)
  b. Oracle assigns label Y_t to X_t
  c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
  d. $t++$
4. While $t \leq T$
  a. $\langle X_t, \mathcal{U}_t \rangle$=Instance_Selection($\mathcal{M}_{t-1}$, $\mathcal{U}_{t-1}, uncertain$)
  b. Oracle assigns label Y_t to X_t
  c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i \rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
  d. $t++$
Return $\mathcal{M}_T$

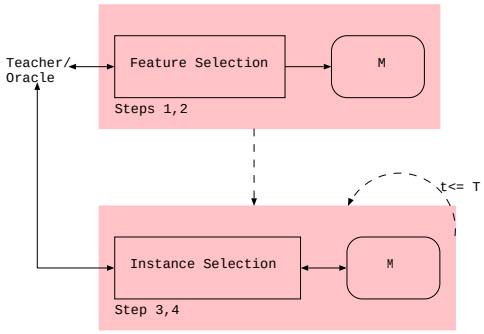


Figure 3: An active learning system where feature selection is done before instance selection (**System 3**). This is one of the two set-ups used in our oracle experiments described in Section 2.2. The second set-up is shown in Figure 4.

The second type of experiment is a slight variation designed to isolate the effect of oracle feature selection on example selection versus model selection during active learning. In these experiments, active learning proceeds normally with all the features available, but after all the instances are picked (after T iterations), the best set of k features that improve the resulting trained classifier the most are picked and the resulting performance is reported. This is shown schematically and with pseudo-code in Figure 4. We note that even when starting with the same initial set of labeled instances, the classifiers learned during active learning, hyperplanes in our case, in these two systems may be different as they are learned in different spaces (using different feature subset sizes). Besides, the set of labeled instances is small, so the learning algorithm may not be able to find the best "unique" hyperplane. In turn, the instances picked subsequently during active learning may differ

substantially as both the spaces the instances reside in and the learned classifiers may be different. The classifier learned in the feature reduced space may have better accuracy or lead to better choice of instances to label during active learning, though this is not guaranteed or the benefits may be negligible. In short, the trajectory of the active learning process, that is, the instances labeled and classifiers learned, can be different in the two regimes, which may lead to substantially different active learning performance. In Section 5 we provide the details of these experiments.

Systems 3 and 4 can also be used with a teacher (a human) instead of an oracle. For an actual use in practice, we prefer an approach that combines feature selection and instance selection (*e.g.*, as proposed in Section 2.1) because it also allows the system to benefit from the increase in the knowledge of the teacher or the process may help remind the teacher about the usefulness of features as she reads the documents. For example, the teacher who did not know that *ct* stood for *cents* may realize that the word is indeed relevant upon reading documents containing the term. We will discuss these related approaches in Section 7.

Use of Feature Feedback After Active Learning

Input: T (Total number of feedback iterations), $\mathcal{U}$ (Pool of unlabeled instances, init_size (number of random feedback iterations)
Output: $\mathcal{M}_T$ (Model)

$t = 1$; $\mathcal{U}_0 = \mathcal{U}$; $\mathcal{M}_0$ =NULL;
1. While $t \leq$ init_size
   a. X_t = Instance_Selection($\mathcal{M}_0$, $\mathcal{U}_{t-1}$, *random*)
   b. Oracle assigns label Y_t to X_t
   c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i\rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   c. $t++$
2. While $t \leq T$
   a. $\langle X_t, \mathcal{U}_t\rangle$ = Instance_Selection($\mathcal{M}_{t-1}$, $\mathcal{U}_{t-1}$, *instance*)
   b. Oracle assigns label Y_t to X_t
   c. $\mathcal{M}_t$ = train_classifier($\{\langle X_i, Y_i\rangle | i = 1...t\}$, $\mathcal{M}_{t-1}$)
   d. $t++$
3. a. $\{F1, ..., F_f\}$ = Feature_Selection($\mathcal{M}_T$, $\mathcal{U}_T$)
   b. Oracle selects $\{F1, .., F_k\} \subseteq \{F_1, ..., F_f\}$
4. Incorporate_Feature_Feedback($\mathcal{M}_T$, $\{F_1, ..., F_k\}$)

Return $\mathcal{M}_T$

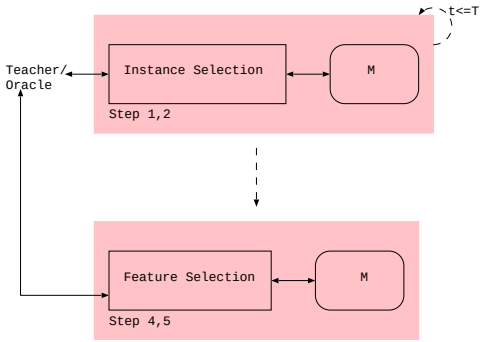


Figure 4: An active learning system where feature selection is done after instance selection (**System 4**). This is one of the two set-ups used in our oracle experiments described in Section 2.2. The first set-up is shown in Figure 3.

3. Implementation

In this section we give implementation details for our experiments. While our approach is applicable to a variety of machine learning algorithms and feature selection approaches, we give the details of our implementation. We use Support Vector Machines (SVMs) as the machine learned classifier, uncertainty sampling as our approach to active learning and information gain as the feature selection technique. We also give details on how we construct the approximate feature oracle.

3.1 Classifier: Support Vector Machines

We use support vector machines (SVMs) in our experiments (the model $\mathcal{M}$ is a Support Vector Machine (SVM)) (Joachims, 1998). An SVM learning algorithm tries to find a hyperplane of maximum margin that separates the data into one of two classes ($Y \in \{-1,+1\}$). A linear SVM is a binary classifier given as

$$f(X) = sign(w \bullet X + b), \tag{1}$$

where w is the vector of weights and b is a threshold, both learned by the SVM learning algorithm.

SVMs are considered to be state-of-the-art classifiers in the domains that we described in Section 4.1 and have been found to be fairly robust even in the presence of many redundant and irrelevant features (Brank et al., 2002; Rose et al., 2002.). Our SVM implementation uses the LibSVM toolkit (Chang and Lin).

3.2 Active Learning: Uncertainty Sampling

Uncertainty sampling (Lewis and Catlett, 1994) is a type of active learning in which the instance that the teacher is queried on is the unlabeled instance that the classifier is most uncertain about. In the case of a naive Bayes classifier, this is the instance which is almost equally likely to be in either of the two classes in a binary classification setting. When the classifier is an SVM, unlabeled instances closest to the margin are chosen as queries (Schohn and Cohn, 2000; Tong and Koller, 2002). This results in the version space being split approximately in half each time an instance is queried. We use a pool size of 500 in our experiments, such that for each instance selection, we look at a new random sample of 500 instances from the unlabeled data. All our methods start out with two randomly picked instances, one in the positive class and one in the negative class. Each subsequent instance is picked through uncertainty sampling.

3.3 Feature Selection: Information Gain

We could have chosen any one of several methods for the ordering of features (Sebastiani, 2002; Brank et al., 2002). Information gain is a common measure for ranking features and has been found to be quite effective (Sebastiani, 2002; Brank et al., 2002), and is easy and quick to compute.

Information gain is given as

$$IG = \sum_{c \in \{-1,+1\}} \sum_{\tau \in \{0,1\}} P(c,\tau) \log \frac{P(c,\tau)}{P(c)P(\tau)} \tag{2}$$

where c denotes the class label (+1 or -1) from section 3.1, and τ is 0 or 1 indicating the presence or absence of a feature respectively. We used information gain wherever we needed to do feature selection.

3.4 Construction of the Approximate Feature Oracle

The (feature) oracle in our experiments has access to the labels of all documents in the data-set (hence the name oracle) and uses this information to return a ranked list of features sorted in decreasing order of importance. We use information gain for feature ranking since it is easy to compute, especially with a large number of training instances. Other feature selection methods (*e.g.*, forward selection) may somewhat increase our upper bound estimates of usefulness of oracle feature feedback. Such improvements will further motivate the idea of using feature feedback, but we don't expect the improvements to be very high. In our oracle experiments, we cut off the ranked list (therefore obtaining a feature subset) at the point that yields the highest average active learning performance. The next section describes our experiments and performance measures.

4. Experimental Set Up

We will now describe our data sets and our data collection methodology for experiments which use teacher feedback on features.[2] We then describe our evaluation framework.

4.1 Data Sets

Our test bed for this paper comes from three domains. The first data set consists of the 10 most frequent classes from the Reuters-21578 corpus (Rose et al., 2002.). The 12,902 documents are Reuters news articles categorized based on topics such as *earnings* and *acquisitions*. The Reuters corpus is a standard benchmark for text categorization. The second corpus is the 20-Newsgroups data set collected by Lang (1995). It has 20,000 documents which are postings on 20 Usenet newsgroups. This is a slightly harder problem because it has a large vocabulary compared to the Reuters corpus (news articles tend to be more formal and terse) and it has many documents in each category which are tangentially related to the topic. The topics reside in a hierarchy with broader topics like *sports* and *computers* at the top level which are further divided into narrower subdivisions. For example, *sports* encompasses more focused groups like *baseball* and *hockey*. There are 20 categories at the lowest level of the hierarchy.

The third corpus is the TDT3 corpus (Allan, 2002) . We used 10 topics from the TDT3 corpus which has 67,111 documents in three languages from both broadcast and news-wire sources. The Linguistic Data Consortium (LDC) provides the output of an automatic speech recognizer (ASR) for the broadcast news sources. Similarly they provide the machine translations of all documents that are not originally in English. We use the ASR and machine translated documents in our experiments. The noise in the ASR and machine translation output makes the TDT corpus particularly difficult to work with. The topics in the TDT corpus are based on news events. Thus, hurricane Mitch and hurricane George would be two different topics and developing a classifier to separate the two classes is seemingly a more difficult problem. The two classes would have a lot of common words especially with regard to lives lost, rescue operations etc. For example, the words *storm* and *damage* each respectively occur in 50% and 27% of the documents on hurricane Mitch and in 75% and 54% of the documents on hurricane George. These common words are probably useful to detect a generic topic like *hurricane* but are not that useful in discriminating hurricane Mitch from hurricane George. However, we think it would be fairly trivial for a human to point out *Mitch* and *George* as two keywords of importance which could then accelerate learning. The word Mitch occurs in

2. The data sets have been made available at `http://ciir.cs.umass.edu/~hema/data/jmlr2006/`.

42% documents on hurricane Mitch and in 0 documents on hurricane George. Similarly, the word George appears in 0.05% documents on the topic of hurricane Mitch and in 88% of the documents on hurricane George.

For all three corpora we consider each topic as a one-versus-rest classification problem, giving us a total of 40 such problems listed in Appendix A. We also pick two pairs of easily confusable classes from the 20-Newsgroups domain to obtain two binary classification problems viz., *baseball vs hockey* and *automobiles vs motorcycles*. In all we have 42 classification problems. As features we use words, bi-grams and trigrams obtained after stopping and stemming with the Porter stemmer (Porter, 1980) in the Rainbow toolkit (McCallum, 1996).

4.2 Data for Whether Humans Can Emulate the Oracle

We picked five classification problems which we thought were perceptible to a non-expert and also represented the broad spectrum of problems from our set of 42 classification problems. We took the two binary classification problems and from the remaining 40 one-versus-rest problems we chose three (*earnings*, *hurricane Mitch* and *talk.politics.mideast*). For a given classification problem we took the top 20 features as ranked by information gain on the entire labeled set. We randomly mixed these with features which are much lower in the ranked list. We showed each user one feature at a time and gave them two options – *relevant* and *not-relevant/don't know*. A feature is relevant if it helps discriminate the positive or the negative class. We measured the time it took the user to label each feature. We did not show the user all the features as a list, though this may be easier, as lists provide some context and serve as a summary. Hence we expect that our method provides an upper bound on the time it takes a user to judge a feature. The instructions given to the annotator are given in Appendix B.

Similarly, we obtain judgments on fifteen documents in each of five categories (see Appendix C). In this case we gave the user three choices – Class 1, Class 2, Don't know. We randomly sampled documents such that at least five documents belonged to each class. We have complete judgments on all the documents for all three data sets. The main purpose of obtaining document judgments was to determine how much time it would take a person to judge documents. We compare the time it takes a user to judge a feature with the time it takes a user to judge a document. We measure the precision and recall of the user's ability to label features. We ask the user to first label the features and then documents, so that the feature labeling process receives no benefit due to the fact that the user has viewed relevant documents. In the learning process we have proposed, though, the user would be labeling documents and features simultaneously, so the user would indeed be influenced by the documents she reads. Hence, we expect that the feature labels we obtained by our experimental method are worse in terms of precision and recall than the real setting. We could in practice ask users to highlight terms as they read documents. Experiments in this direction have been conducted in information retrieval (Croft and Das, 1990).

Our users (participants) were six graduate students and two employees of an Information Technology company, none of whom were authors of this paper. Of the graduate students, five were in computer science and one from public health. All our users were familiar with the use of computers. Five users understood the problem of document classification but none had worked with these corpora. One of our users was not a native speaker of English. The topics were distributed randomly, and without considering user expertise, so that each user got an average of two to three topics. There were overlapping topics between users such that each topic was labeled by two to

three users on average. A feedback form asking the users some questions about the difficulty of the task was handed out at the end (see Appendix D).

4.3 Evaluation

The *deficiency* measure was proposed by Baram et al. (2003) as a measure of the speed of an active learning algorithm, useful for comparing different active learning algorithms. Baram et al. defined deficiency in terms of accuracy. Accuracy is a reasonable measure of performance when the positive class is a sizable portion of the total. Since this is not the case for all the classification problems we have chosen, we modify the definition of deficiency, and define it in terms of the $F1$ score (harmonic mean of precision and recall). For deficiency a lower value is better. As we also report on the $F1$ scores, for which higher values are better, for consistency and easier interpretation of our charts and tables we define *efficiency* $= 1 -$ *deficiency*. Efficiency has a range from 0 to 1, and a larger value indicates a faster rate of learning. Thus, in all our reports higher values are better.

Let $F1_t(\text{RAND})$ be the average $F1$ achieved by an algorithm when it is trained on t randomly picked instances and $F1_t(\text{ACT})$ be the average $F1$ obtained using t actively picked instances.

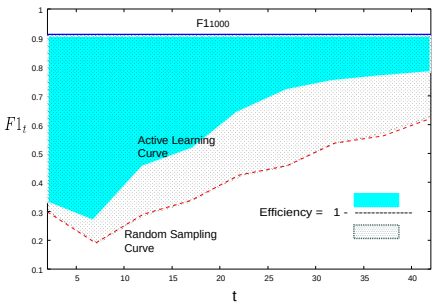
Efficiency, E_T is defined as

$$E_T = 1 - \frac{\sum_{t=2}^{T}(F1_M(\text{RAND}) - F1_t(\text{ACT}))}{\sum_{t=2}^{T}(F1_M(\text{RAND}) - F1_t(\text{RAND}))}. \quad (3)$$

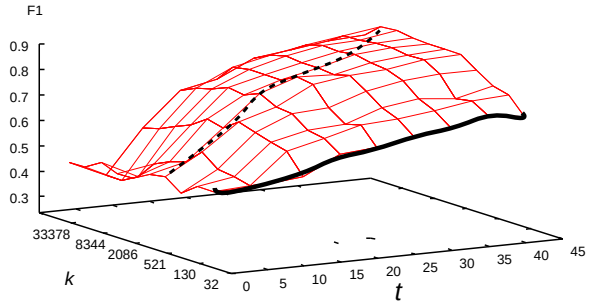
$F1_M(\text{RAND})$ is the $F1$ obtained with a large number (M) of randomly picked instances. The value $F1_M(\text{RAND})$ represents the performance of a classifier with a large amount of training data, and can be considered the optimal performance under supervised learning. With large amounts of training data, we expect the performance of a classifier trained using active learning to be about the same as a classifier trained using random sampling. However, we would like active learning to approach this level as *quickly* as possible. The metric therefore takes into consideration how far the performance is from the optimal performance by computing the difference $F1_M(\text{RAND}) - F1_t(\text{ACT})$ and $F1_M(\text{RAND}) - F1_t(\text{RAND})$. The metric compares this difference when t documents have been actively picked to the difference when t documents have been randomly picked for increasing number of training documents t.

Since we are concerned with the beginning of the learning curve, we stop after $T = 42$ number of documents have been sampled. For expedience, we did not measure performance at every point from 2 to 42 labeled documents, but compute the summation at discrete intervals, measuring $F1$ after each additional five documents have been labeled: $t = 2, 7, 12, 17 \ldots 42$. For this paper we take $M = 1000$, that is, we consider the optimal random-learning performance to be attained after the classifier has seen 1000 labeled instances. In our experiments $F1_t(\bullet)$ is the average $F1$ computed over 10 trials. In addition to efficiency we report $F1_t$ for some values of t.

To understand the intuition behind efficiency, we can draw the active learning curve by plotting $F1_t(\text{ACT})$ for increasing values of t, as shown in Figure 5(a). Similarly we can draw the random learning curve by measuring $F1_t(\text{RAND})$ for increasing values of t. $F1_M$ is a straight line representing the best achievable performance. Then efficiency is one minus the ratio of the solid colored area to the spotted area. The higher the efficiency, the better the active learning algorithm. We aim to maximize both efficiency and $F1$. In some of our experiments we obtain efficiencies exceeding 1. This is due to using a finite M: it is possible that a classifier produced by active learning on 42 or fewer instances may do better than a classifier trained on a random sample of a 1000 instances.



(a) Efficiency



(b) The best feature set size increases as the training data increases.

Figure 5: The figure on the left (a) illustrates *efficiency*, the performance metric which captures rate of learning. The figure on the right illustrates the *learning surface*. The plot is a measure of $F1$ as a function of the number of features and training documents. The dotted line traces the region of maximum $F1$. With few training documents, aggressive feature selection (few features) are needed to maintain high accuracy. The thick dark band illustrates traditional active learning.

5. Results: Experiments with an Oracle

In this section we seek the answer to the following questions:

- Can feature feedback significantly boost active learning performance?
- Should we use feature feedback during the entire active learning process (both instance selection, and model selection) or only for model selection?

To measure how much gain we can get from feature feedback we can measure the impact of the oracle (which has knowledge of the best set of features) on active learning. This gives us an upper bound on how useful feature feedback is for active learning. Then in the next section we go on to measure the extent to which humans can emulate the oracle.

We will use systems 3 and 4 (described in Section 2.2) to help understand the answers to the above questions.

5.1 Improvements to Active Learning with Feature Selection

Following the algorithm for system 3 (see Section 2.2, Figure 3), let $f = N$ (the total number of features) and let us assume that the oracle selects the k most important features (by information gain) in step 1.b, which is used to initialize the model in step 2. Random sampling (step 3.a), in this particular implementation, does not use any of the feature information or the initial model. Then in step 3.c, we prune the data set by retaining only the chosen k features for each instance. We now perform active learning on the instances in this reduced feature space (step 4). We evaluate these experiments at many points in the two-dimensional space of number of features k versus number of labeled documents t by measuring the F1 score: $F1_t(\mathrm{ACT}, k)$. We can similarly measure

Data Set ↓	$E_{42}(k)$ $k=N$	$k=n$	n	$F1_7(\text{ACT},k)$ $k=N$	$k=m$	m	$F1_{22}(\text{ACT},k)$ $k=N$	$k=p$	p	$F1_{1000}$
Reuters	0.59	**0.68**	11179.3	0.36	**0.48**	8481.1	0.580	**0.66**	11851.6	0.73
20 NG	0.40	**0.66**	41.5	0.07	**0.22**	48.3	0.21	**0.29**	487.1	0.45
TDT	0.26	**0.34**	1275.7	0.19	**0.29**	11288	0.28	**0.41**	10416.1	0.75
Bas vs Hock	0.29	**0.55**	25	0.59	**0.70**	25	0.78	**0.83**	200	0.96
Auto vs Mot.	0.68	**0.32**	125	0.43	**0.72**	62	0.76	**0.86**	31	0.90

Table 1: Improvements in efficiency, $F1_7$ and $F1_{22}$ using an oracle to select the most important features (Figure 3). We show results for each metric at N (total number of features for a particular data set) and at feature set sizes for which the scores are maximized (n, m and p for E_{42}, F_7, and F_{22} respectively). For each of the three metrics, figures in bold are statistically significant improvements over uncertainty sampling using all features (the corresponding columns with feature set size of N). We see that with only seven documents labeled ($F1_7$) the optimal number of features is smaller (8481.1 on average), while with more documents labeled, (22 documents labeled for $F1_{22}$) the optimal number of features is larger (11851.6 on average). When 1000 documents are labeled ($F1_{1000}$) using the entire feature set leads to better scores with the $F1$ measure. This suggests that our best active-learning algorithm would adjust the feature set size according to the number of training documents available.

performance in the reduced feature space when instances are picked randomly. Thus we can compute efficiency in the reduced feature space as $E_T(k)$. When $f = k = N$ the algorithm reduces to traditional active learning (Figure 1).

Figure 5(b) shows a plot of $F1_t(\text{ACT},k)$ for different values of the number of features k and number of labeled training instances t, for the *earnings* category in Reuters. The dotted curve traces the maximum F_t for each value of t. The x, y and z axes denote k, t and $F1_t(\text{ACT},k)$ respectively. The number of labeled training instances t ranges from 2 to 42 in increments of 5. The number of features used for classification k has values from $33,378$ (all features), $33378/2$, $33378/4$ to 32. The dark band represents the case when all features are used. This method of learning in one dimension is representative of traditional active learning. Clearly when the number of documents is few, performance is better when there is a smaller number of features. As the number of documents increases the number of features needed to maintain high accuracy increases. From the figure it is obvious that we can get a big boost in accuracy by starting with fewer features and then increasing the complexity of the model as the number of labeled documents increase.

Table 1 captures the behavior of all the problems in the Reuters corpus when there is an oracle to do the feature selection. The second column ($k = N$) in Table 1 shows the efficiency obtained using uncertainty sampling and all (N) features. The third column ($k = n$) indicates the average efficiency obtained using uncertainty sampling and a reduced subset of features. The feature set size n at which this efficiency is attained is shown in column four. For each classification problem, we identify the feature set size which optimizes the efficiency, that is, optimizes the rate at which classification performance under active learning approaches learning with all of the data. This optimal feature set size for active learning n is given by

$$n = \text{argmax}_k E_{42}(k).$$

Figure 6 shows the efficiencies at $E_{42}(N)$ and $E_{42}(n)$ for the individual problems in the three corpora. In many cases, $E_{42}(N)$ is much less than $E_{42}(n)$.

Column 5 ($k = N$) in Table 1 shows the value of $F1_7(\text{ACT}, N)$: the F1 score with seven instances selected using active learning, when all features are used. Column 6 shows the average $F1_7(\text{ACT}, m)$ using a reduced feature subset. As for efficiency the best feature subset size (m) for each classification problem is obtained as the feature subset size at which $F1_7(\text{ACT}, k)$ is maximum. For example in Figure 5(b) at seven instances the best $F1$ is obtained with 512 features. Figure 7 shows the values of $F1_7$ computed using all (N) features and using a reduced subset of (m) features for individual problems.

Columns 7, 8, and 9 in Table 1 show similar results for $F1_{22}(\text{ACT}, k)$ with the best feature subset size at $t = 22$ being denoted by p. The values for individual problems is illustrated in Figure 8. The last column shows $F1_{1000}(\text{RAND})$.

All 42 of our classification problems exhibit behavior as in Figure 5(b). For all classification problems, n, m and p are less than the maximum number of features. Also, for 31 of 42 cases $m \leq p$ (that is, the number of features optimal for seven labeled instances, m is less than the number of features optimal for 22 labeled instances, p) meaning that as the number of labeled instances (t) increases, the complexity of the classifier also needs to increase. For 20-Newsgroups, for all classes we observe that efficiency, $F1_7$ and $F1_{22}$ are best at very small feature subset sizes. For Reuters and TDT there are classes for which a large number of features become important very early (for example: *trade, Bin Laden indictment, NBA labor disputes*).

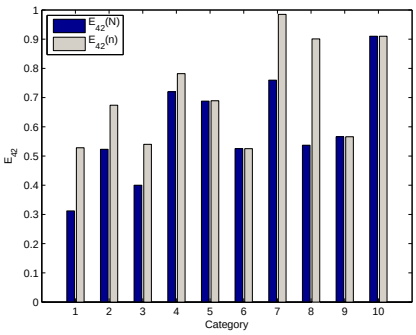
5.2 Feature Selection for Instance Selection or Model Selection

As mentioned in Section 2.2 the difference between systems 3 and 4 is in that feature selection precedes active learning in the former, and the best feature subset is picked in a retrospective manner, while it follows active learning in the latter. The two systems when used with oracle feature selection will help us understand the extent to which oracle feedback aids different aspects of the active learning process. Figure 9 compares the results of using system 4 and system 3 on the Reuters corpus.

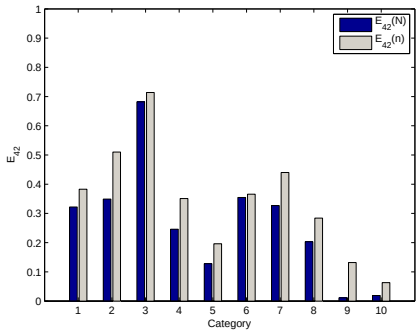
There is hardly any difference between systems 3 and 4, especially on $F1_7$. All other data sets exhibit the same behavior. The $F1_{22}$ and E_{42} values are slightly better for the method that does feature selection before active learning (system 3) but it is not significantly different (determined using a t-test at the 0.05 level of confidence) from the method where feature pruning is done after instance selection (system 4). Thus, our experimental results suggest there is some benefit for instance selection but most of the benefit from oracle feature selection comes from improving the model learned (model selection).

5.3 Discussion: Why Does Feature Selection Help?

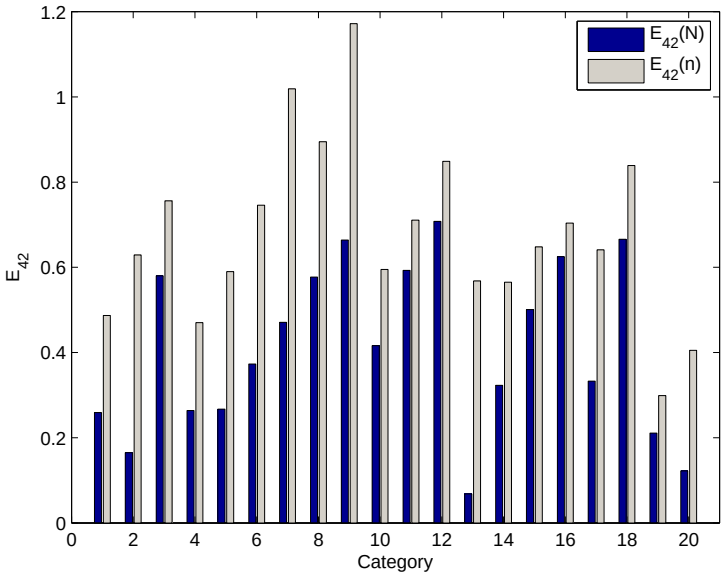
Intuitively, with limited labeled data, there is little evidence to prefer one feature over another, so the learner has to spread the feature weights more or less evenly on many features. In other words, the learner has to remain conservative. Feature/dimension reduction by the oracle allows the learner to "focus" on dimensions that matter, rather than being overwhelmed with numerous dimensions



(a) Reuters

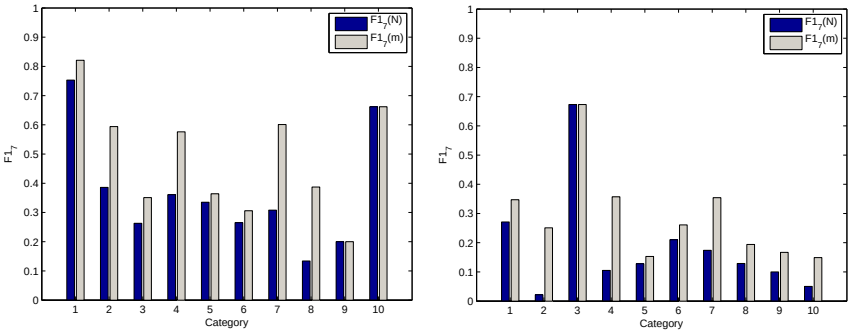


(b) TDT

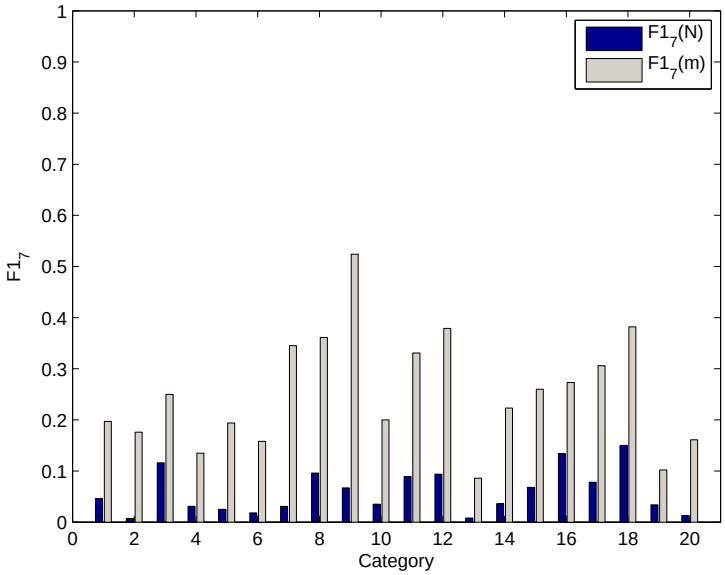


(c) 20 Newsgroups

Figure 6: Improvements in efficiency using an oracle to select the most important features. For each problem we show efficiency at N (total number of features for a particular data set) on the right and efficiency at the feature set sizes for which the efficiency is maximized (n) on the left. The class keys are given in Appendix A.

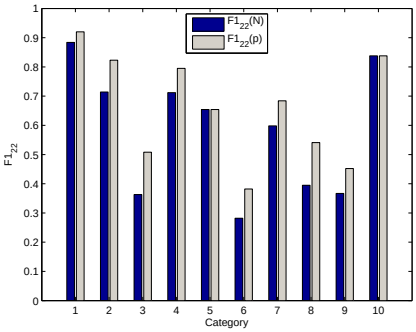


(a) Reuters (b) TDT

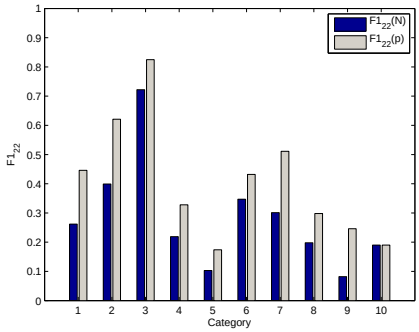


(c) 20 Newsgroups

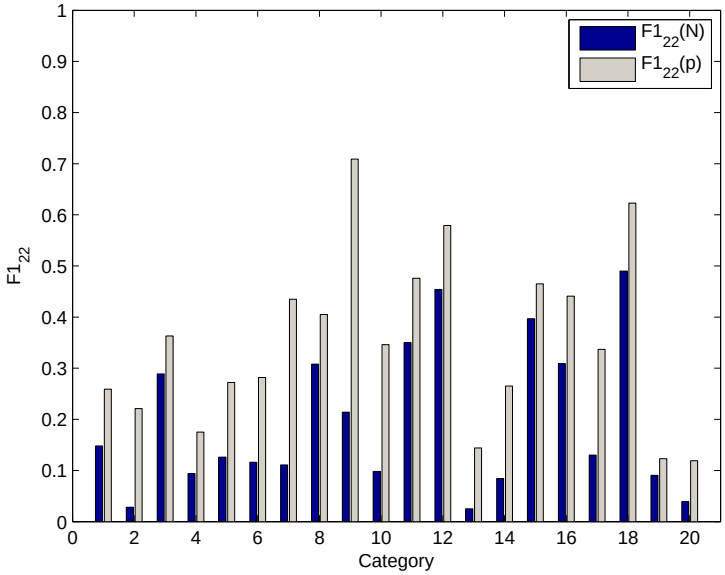
Figure 7: Improvements in $F1_7$ using an oracle to select the most important features. For each problem we show $F1_7$ at N (total number of features for a particular data set) on the left and $F1_7$ at the feature set sizes for which the $F1_7$ is maximized (m) on the right. Remember, the objective is to maximize $F1_7$. The class keys are given in Appendix A.



(a) Reuters

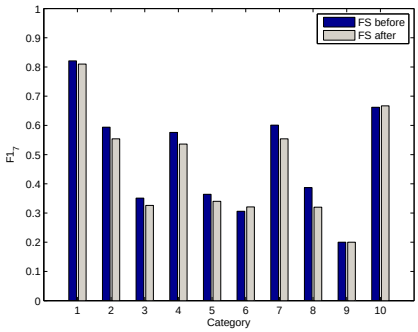


(b) TDT

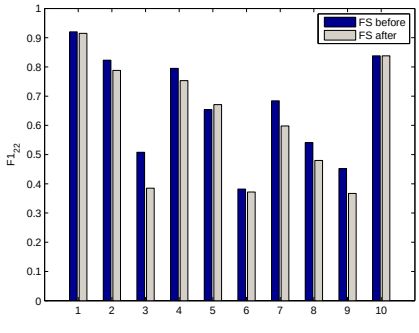


(c) 20 Newsgroups

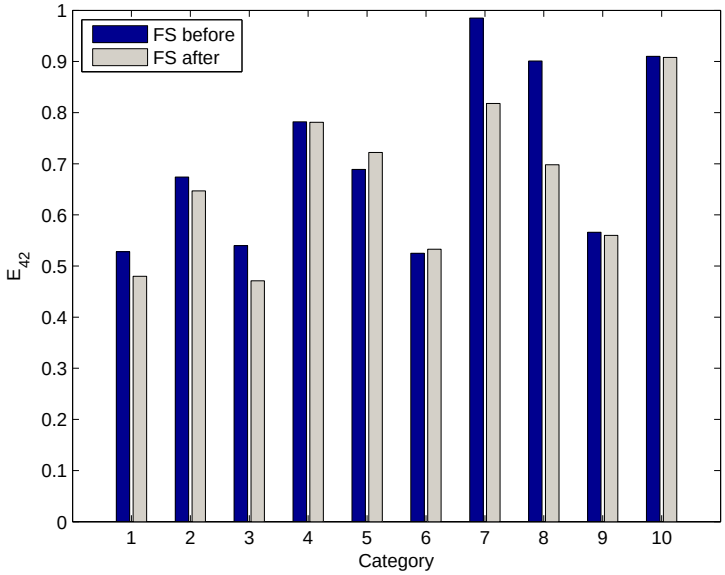
Figure 8: Improvements in $F1_{22}$ using an oracle to select the most important features. For each problem we show $F1_{22}$ at N (total number of features for a particular data set) on the right and $F1_{22}$ at the feature set sizes for which the $F1_{22}$ is maximized (p). Remember that the objective is to maximize $F1_{22}$. The class keys are given in Appendix A.



(a) $F1_7$



(b) $F1_{22}$



(c) E_{42}

Figure 9: $F1_7$, $F1_{22}$ and efficiency E_{42} for the Reuters corpus when feature selection is done before active learning (system 3) and when feature selection is done after active learning (system 4).

right at the outset of learning. Oracle feature reduction allows the learner to assign higher weights to fewer features. This tends to improve accuracy, since the oracle selected features are the actual most predictive features. Oracle feature reduction may also improve instance selection as the learner obtains instances to query that are important for finding better weights on the features that matter. As the number of labeled instances increases, feature selection becomes less important, as the learning algorithm becomes better capable of finding the discriminating hyperplane (feature weights) on its own. We experimented with filter based methods for feature selection, which did not work very well (we got tiny or no improvements). This is expected given such limited training set sizes, and is consistent with most previous findings (Sebastiani, 2002). Next we determine if humans can identify these *important features*.

6. Results: Experiments with a Human (Teacher)

Consider our introductory example of the editor who was looking for training instances for the topic *hurricane Mitch*. From a human perspective the words *hurricane, Mitch* etc may be important features in documents discussing this topic. Given a large number of documents labeled as on-topic and off-topic, and given a classifier trained on these documents, the classifier may also find these features to be most relevant. With little labeled data (say two labeled instances) the classifier may not be able to determine the discriminating features. While in general in machine learning the source of labels is not important to us, in active learning scenarios in which we expect the labels to come from humans we have valid questions to pose:

1. Can humans label features as well as documents? In other words are features that are important to the classifier perceptible to a human?

2. If the feature labels people provide are imperfect, is the feedback still beneficial to active learning?

We address the first question in the following section. Our concern in this paper is asking people to give feedback on features, or word n-grams, as well as entire documents. We may expect this to be more efficient, since documents are often long and may contain redundant or irrelevant content, and results from our oracle experiments indicate great potential in doing feature selection. We then move on to discuss a real system which employs a two-tiered approach of document feedback and feature feedback like the system in Figure 2 which we evaluate using a simulation: we obtain feedback on features and documents apriori, and use the judgments so obtained to measure the effectiveness of our approach. We employed this approach rather than one where an actual user labels features and documents in tandem because our approach allows us to run many repeated trials of our experiments, enabling us to do significance testing. Given that we have demonstrated the effectiveness of our algorithm, we reserve a more realistic evaluation with a true human in the loop for future work.

6.1 Can Humans Emulate the Oracle?

We evaluated user feature labeling by calculating their average precision and recall at identifying the top 20 features as ranked by an oracle using information gain on the entire labeled set. Table 2 shows these results. For comparison we have also provided the precision and recall (against the

Class	Precision		Recall		Avg. Time (secs)		kappa
Problem	Hum.	@50	Hum.	@50	Feat.	Docs	
baseball vs hockey	0.42	0.30	0.70	0.30	2.83	12.60	0.503
auto vs motorcycle	0.54	0.25	0.81	0.25	3.56	19.84	0.741
earnings	0.53	0.20	0.66	0.25	2.97	13.00	0.495
talk.politics.mideast	0.68	0.35	0.55	0.35	2.38	12.93	0.801
hurricane Mitch	0.72	0.65	0.56	0.65	2.38	13.19	0.857
Average	0.580	0.35	0.65	0.38	2.82	14.31	0.68

Table 2: Ability of users to identify important features. Precision and Recall against an oracle, of users (Hum.) and an active learner which has seen 50 documents (@50). Note that precision and recall denote the ability of the user to recognize the oracle features and are not measures of classification accuracy. Average labeling times for features and documents are also shown. All numbers are averaged over users.

same oracle ranking of top 20 features) obtained using 50 labeled instances (picked using uncertainty sampling) denoted by @50. Precision and recall of our participants is high, supporting our hypothesis that features that a classifier finds to be relevant after seeing a large number of labeled instances are obvious to a human after seeing little or no labeled data (the latter case being true of our experiments). Additionally the precision and recall @50 is significantly lower than that of humans, indicating that a classifier like an SVM needs to see much more data before it can find discriminatory features.

Table 2 also shows the times taken for labeling features and documents. On average humans take five times longer to label one document than to label one feature. Note that features may be even easier to label if they are shown in context – as lists, with relevant passages etc. We measured whether document length influences document labeling time. We found the two to be correlated by $r = 0.289$ which indicates a small increase in time for a large increase in length. The standard deviations for precision and recall are 0.14 and 0.15 respectively. Different users vary significantly in precision, recall and the total number of features labeled relevant. From the post-labeling survey we are inclined to believe that this is due to individual caution exercised during the labeling process.

We also measure the extent to which our users tend to agree with each other about the importance of features. For this we use the kappa statistic (Cohen, 1960) which is a measure that quantifies the agreement between annotators that independently classify a set of entities (in our case the features) into classes (relevant versus non-relevant/don't know). Kappa is given by:

$$\text{kappa} = (p_o - p_c)/(1 - p_c) \tag{4}$$

Where p_o is the observed proportion of agreement and p_c is the agreement due to chance (Cohen, 1960; Landis and Koch, 1977). Landis and Koch (1977) provide a table giving guidelines about how to interpret kappa values. We find a value of 0.68 to be the average kappa across the five categories in our user study. According to Landis and Koch (1977) this indicates substantial agreement.

We obtained judgments on a handful of documents for each user. We used those judgments to measure time. Some of our users had difficulty judging documents. For example, for the earnings category, one of our users had very low agreement with the true Reuters categories. This person did

not have a finance background and could not distinguish well between earnings and acquisitions, often confusing the two. But this user did quite a good job of identifying useful features. She missed only six of 20 of the relevant features and had only five false alarms. The features that she marked relevant, when used in the human-in-the-loop algorithm resulted in an efficiency of 0.29. This is still an improvement over traditional uncertainty sampling which has a efficiency of 0.10. These results can be explained by looking at the question posed to the annotator. When it came to features, the question was on the discriminative power of the feature. Hence a user did not have to determine whether the words *shares* was pertinent to *earnings* or not but rather she only needed to indicate whether the word was likely to be discriminatory. Additionally, one of our users suggested that terms shown in context would have carried more meaning. The user said that she did not realize the term *ct* stood for *cents* until she read the documents. But since she was made to judge terms before documents this user's judgment had marked the term *ct* as non-relevant/don't know.

Some of the highlights of the post-labeling survey are as follows. On average users found the ease of labeling features to be 3.8 (where 0 is most difficult and 5 is very easy) and documents 4.2. In general users with poor prior knowledge found the feature labeling process very hard. The average expertise (5=expert) was 2.4, indicating that most users felt they had little domain knowledge for the tasks they were assigned. We now proceed to see how to use features labeled as relevant by our naive users in active learning.

6.2 Using Human Feature Feedback simultaneously with Document Feedback in Active Learning

We saw in Section 5 that feature selection coupled with uncertainty sampling gives us big gains in performance when there are few labeled instances. In Section 6.1 we saw that humans can discern discriminative features with reasonable accuracy. We now describe our approach of applying term and document level feedback simultaneously in active learning. In Section 2.2 we discussed the possible cognitive advantages of an interleaved approach of feature selection and instance selection. Additionally, we found that feature selection does not hurt uncertainty sampling and may aid it. In the following section we describe an implementation for system 2.

6.3 Implementation

Following Figure 2, the features to be displayed to the user (in step 2.d.i) are the top f features obtained by ordering the features by information gain. More specifically, we trained the SVM classifier on these t labeled instances. Then to compute information gain, we used the five top ranked (farthest from the margin on the positive side) documents from the unlabeled set in addition to the t labeled documents. Using the unlabeled data for term level feedback is very common in information retrieval and is called pseudo-relevance feedback (Salton, 1968). The user labels $k \geq 0$ of the f features as relevant or discriminative (step 2.d.ii). If a user has labeled a feature in a previous iteration, we don't show the user that feature again (the top f are picked from the unlabeled features). We set f to 10 in our experiments.

We incorporate feature feedback (step 2.e) as follows. Let $\vec{s} = s_1 \ldots s_N$ be a vector containing weights of relevant features. If a feature number i that is presented to the user is labeled as relevant then we set $s_i = a$, otherwise $s_i = b$, where a and b are parameters of the system. For each X in the

labeled and unlabeled sets we multiply x_i by s_i to get x'_i. In other words, we scale all the features that the user indicated as relevant by a and the rest of the features by b. We set $a = 10$ and $b = 1$. [3]

By scaling the important features by a we are forcing the classifier to assign higher weights to these features. We demonstrate the intuition with the following example. Consider a linear SVM, $N = 2$ and two data points $X_1 = (1,2)$ and $X_2 = (2,1)$ with labels $+1$ and -1 respectively. An SVM trained on this input learns a classifier with $w = (-0.599, +0.599)$. Thus, both features are deemed equally discriminative by the learned classifier. If feature 1 is indicated to be more discriminative by our user, then by our method $X'_1 = (10,2)$ and $X'_2 = (20,1)$ and $w' = (0.043, -0.0043)$, thus f_1 is assigned a much higher weight in the learned classifier. Now, this is a "soft" version of the feature selection mechanism of section 5. But in that case the oracle knew the ideal set of features. Those experiments may be viewed as a special case where $b = 0$. We expect that human feedback is imperfect and we do not want to zero-out potentially relevant features.

6.4 Simulating User Feedback

We use the relevance judgments on features obtained as described in Section 6.1 to simulate the user in each iteration. At each iteration of the algorithm, if a feature that is presented had been marked by the user as relevant, in the relevance judgment experiments of the previous section, we mark the value of that feature as 1 in the vector $\vec{s}$. The vector $\vec{s}$ is noisier (less complete) than the case where we would have obtained relevance judgments on features during the actual execution of the algorithm. This is because in addition to mistakes made by the user, we lose out on those features that the user might have considered relevant, had she been presented that feature when we were collecting relevance judgments for a relatively small subset of features. In a real life scenario this might correspond to the lazy user who labels few features as relevant and leaves some features unlabeled in addition to making mistakes.

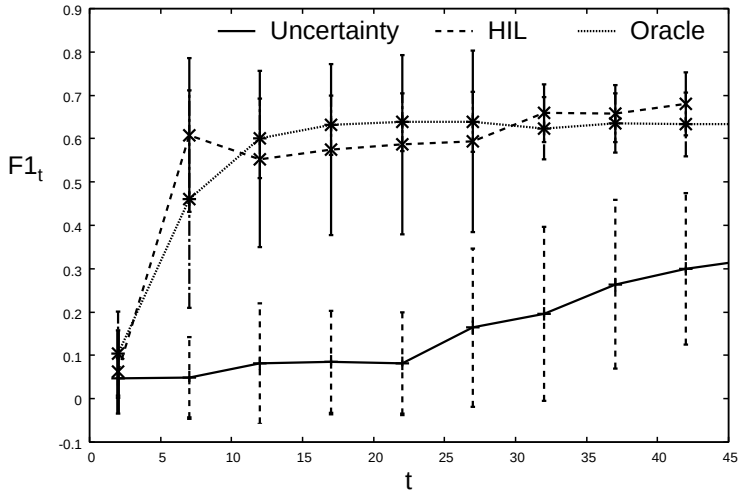
To make our experiments repeatable (to compute average performance and for convenience) we simulate user interaction as follows. For each classification problem we maintain a list of features that a user might have considered relevant had she been presented that feature. For these lists we used the judgments obtained in Section 4.2. Thus for each of the five classification problems we had two or three such lists, one per user who judged that topic. For the 10 TDT topics we have topic descriptions as provided by the LDC. These topic descriptions contain names of people, places and organizations that are key players in this topic in addition to other keywords. We used the words in these topic descriptions to be equal to the list of relevant features. Now, given these lists we can perform the simulated HIL (*human in the Loop*) experiments for 15 classification problems. Figure 10 shows the performance of the HIL experiments. Like before we report efficiency (E_{42}), the $F1$ score with 7 labeled documents ($F1_7$), and the $F1$ score with 22 labeled documents ($F1_{22}$) for each of uncertainty sampling (Unc), oracle feature selection with uncertainty sampling (Ora) and the Human in the Loop (HIL) algorithm. As a baseline we also report results for the case when the top 20 features as obtained by the information gain oracle are input to the simulated HIL experiments (this represents what a user with 100% precision and recall would obtain by our method). The oracle is (as expected) much better than plain uncertainty sampling, on all three measures, validating the effectiveness of our proposed system Section 2.1. The performance of the HIL experiments is almost as good as the oracle, indicating that user input (although imperfect)

3. We picked our algorithm's parameters based on a quick test on three topics (baseball, earnings, and acquisitions) using the oracle features of Section 5.

can help improve performance significantly. The plot on the right is of $F1_t(\text{HIL})$ for *hurricane Mitch*. As a comparison $F1_t(\text{ACT})$ is shown. The HIL values are much higher than for uncertainty sampling.

Dataset	E_{42}			$F1_7$			$F1_{22}$		
	Unc	Ora	HIL	Unc	Ora	HIL	Unc	Ora	HIL
Baseball	0.29	0.59	0.54	0.49	0.63	0.60	0.63	0.79	0.70
Earnings	0.10	0.36	0.36	0.61	0.79	0.73	0.80	0.85	0.86
Auto vs Motor	0.18	0.66	0.40	0.35	0.62	0.60	0.71	0.83	0.73
Hurr. Mitch	0.11	0.62	0.62	0.04	0.46	0.60	0.08	0.63	0.58
mideast	0.51	0.72	0.72	0.14	0.28	0.29	0.32	0.49	0.49
TDT (avg)	0.14	0.23	0.11	0.09	0.21	0.24	0.18	0.32	0.22

(a)



(b) The graph shows the learning curves for *Hurricane Mitch* (6th row of the above table) with the x-axis being the number of labeled documents and y-axis $F1(\text{HIL})$.

Figure 10: Improvements due to human feature selection. The $F1_7$ and $F1_{22}$ scores in the table show the points on the curves where 7 and 22 documents have been labeled. The difference between no feature feedback (Unc) and human-labeled features (HIL) is greatest with few documents labeled, but persists up to 42 documents labeled.

When to stop asking for labels on both features and documents and switch entirely to documents remains an area for future work. We provide some initial results in this regard. Consider that we ask for both document and feature feedback up to j iterations and after that we only ask for document feedback. Figure 11 shows the active learning curves for different values of j for the *hurricane Mitch* problem in the TDT corpus. The case when $j = 0$ represents traditional uncertainty sampling. When $j = 5$ there is improvement over the case when $j = 0$, and when $j = 10$ there is even more improvement. Beyond $j = 10$ there is little gain in obtaining feature feedback. It seems that relevant features are usually spotted in very early iterations. We see similar behavior for other problems in our domains. For the *auto vs motorcycles* problem, the user has been asked to label 75% of the

oracle features (averaged over multiple iterations and multiple users) at some point or the other. The most informative words (as determined by the oracle) – *car* and *bike* are asked of the user in very early iterations. The label for *car* is always (100% of the times) asked, and 70% of the time the label for this word is asked to the user in the first iteration itself. This is closely followed by the word *bike* which the user is queried about within the first five iterations 80% of the time. Most relevant features are asked within 10 iterations which makes us believe that we can often stop feature level feedback in around 10 iterations.

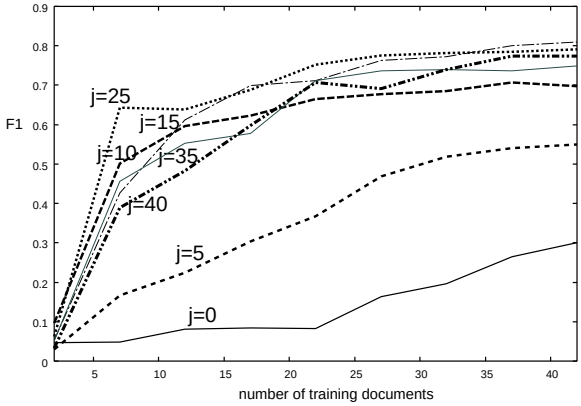


Figure 11: Human Feature Selection for *Hurricane Mitch* for different amounts of feature feedback. The legend indicates the number of iterations (j) for which there was both feature and document feedback, after which only document feedback was asked for. The line at the bottom, labeled $j = 0$ corresponds to regular uncertainty sampling or the case when feature feedback was asked for 0 iterations. The line corresponding to $j = 5$ iterations is significantly better than when $j = 0$. All other cases, $j = 10$ ... $j = 40$ are clumped at the top.

7. Related Work

Our work is related to a number of areas including query learning, active learning, use of (prior) knowledge and feature selection in machine learning, term-relevance feedback in information retrieval, and human-computer interaction.

Term level feedback has been studied in information retrieval (Anick, 2003; Croft and Das, 1990; Belkin et al., 2001). Many participants in the TREC HARD track (Voorhees and Buckland, 2005) generate clarification forms for users to refine or disambiguate their query. Many of the effective forms are composed of lists of terms and the user is asked to mark terms as relevant or not, and some have found that term level feedback is more effective than document level feedback (Diaz and Allan, 2005). The TREC interactive task has focused on issues regarding the kinds of questions that can be asked of the user. They find that users are happy to use interfaces which ask the user to reformulate their queries through a list of suggested terms. They also find that users are willing to mark both positive and negative terms (Belkin et al., 2001).

Our proposed method is an instance of query-based learning and an extension of standard ("pool-based") active learning which focuses on selective sampling of instances from a pool of unlabeled data alone (Cohn et al., 1994). Although query-based learning can be very powerful in theory (Angluin, 1992), arbitrary queries may be difficult to answer in practice (Baum and Lang, 1992). Hence the popularity of pool-based methods, and the motivation for studying the effectiveness and ease of predictive feature identification by humans in our application area. To best of our knowledge, all prior work on query learning and active learning focused on variants of membership queries, that is, requesting the label of a possibly synthesized instance. Our work is unique in the field of active learning as we extend the query model to include feature as well as document level feedback.

Feature feedback may be viewed as the teacher providing evidence or an explanation for the learner on the reasoning behind the labeling. The field of explanation-based learning, however, concerns itself on a deductive rather than an inductive learning task, using one instance and a given domain theory to generalize (Mitchell et al., 1986; DeJong and Mooney, 1986).

Feature selection can lead to improvements in the performance (accuracy) or in the space or time efficiency of the classifier. When there are sufficient labeled instances, most state of the art learning algorithms are able to distinguish the relevant features from the irrelevant ones (Brank et al., 2002). Hence there is little improvement in performance with an additional feature selection component. When there are few labeled instances, working with a small set of relevant features tends to be more useful. This phenomenon has been referred to in statistics as the Hughes phenomenon (Hughes, 1968). Weight regularization may be viewed as a soft version of feature selection: for best performance, in general the smaller the training set, the smaller the total weight that is allowed to be spread over the features. Unfortunately, to do automatic feature selection well, we need sufficient training data, leading to a chicken-and-egg problem. Fortunately, in document classification users have the intuition to point out a small subset of useful features which would be beneficial when there are few labeled instances.

Budgeted learning also works on identifying the predictive features during an active learning setting, but in this case the feature values are unknown and there is a cost to finding each feature's value for each instance of interest (such as the outcome of blood test on an individual) (Lizotte et al., 2003). That human prior knowledge can accelerate learning has been investigated by Pazzani and Kibler (1992), but our work differs in techniques (they use prior knowledge to generate horn-clause rules) and application domains. Beineke et al. (2004) use human prior knowledge of co-occurrence of words, at feature generation time, to improve classification of product reviews. None of this work, however, considers the use of prior knowledge in the active (sequential) learning setting.

Our study of the human factors (such as quality of feedback and costs) is also a major differentiating theme between our work from previous work in incorporating prior knowledge for training. Past work has not addressed this issue, or might have assumed experts in machine learning taking a role in training the system (Schapire et al., 2002; Wu and Srihari, 2004; Godbole et al., 2004; Jones, 2005). We only assume knowledge about the topic of interest. Our algorithmic techniques and the studied modes of interaction also differ somewhat and are worth further comparison. Jones (2005) also used single feature-set labeling in the context of active learning: the user was queried on a feature rather than the whole instance. The labeled feature was taken as a proxy for the label of any instance containing that feature, so a single feature labeling potentially labeled many documents (similar to the *soft* labeling technique discussed next). This was found to be more economical than whole-instance labeling for some tasks. The instances in this work consisted of only two features (a

noun-phrase and a context), so labeling one feature is equivalent to labeling half an instance. Our work differs in that our instances (documents) contain many features (words) and we combine both feature labeling and document labeling. Our work also differs in that we use the labeled features for feature selection and feature re-weighting, rather than as proxies for document labels.

Both Wu and Srihari (2004) and Schapire et al. (2002) assume that prior knowledge is given at the outset which leads to a "soft" labeling of the unlabeled data. This extra labeling is incorporated into training via modified boosting or SVM training. By soft labeling, we mean the extra labels, generated via prior knowledge, are not certain and a method that uses such information may for example assign low confidences to such labellings or lower the misclassification costs compared to misclassification costs for instances labeled directly by a human. However, in our scheme the user is labeling documents and features in an interactive and interleaved fashion. We expect that our proposed interactive mode has an advantage over requesting prior knowledge from the outset, as it may be easier for the user to identify or recall relevant features while labeling documents in the collection and being presented with candidate features. Our method of scaling the dimensions and training (without using the unlabeled data) has an advantage over soft labeling in situations where one may not have access to much unlabeled data, for example in online tasks such as filtering news streams and categorizing personal emails. Furthermore, we simplify the user's task in that our technique does not require the user to specify whether the feature is positively or negatively correlated with the category, just whether the user thinks the feature is relevant or predictive. On the other hand, in the presence of ample unlabeled data, soft labeling methods might more effectively incorporate the information available in the unlabeled data. Both approaches require extra parameters specifying how much to scale the dimensions or the confidence or misclassification costs to assign to the generated labellings, though some fixed parameter settings may work for most cases, or automated methods could be designed.

The work of Godbole et al. (2004) emphasizes system issues and focuses on multi-class training rather than a careful analysis of effects of feature selection and human efficacy. Their proposed method is attractive in that it treats features as single term documents that can be labeled by humans, but they also study labeling features before documents (and only in an "oracle" setting, without using actual human annotators). They do not observe much improvements using their particular method over standard active learning in the single domain (Reuters) they test on. Finally, we mention another method of incorporating prior knowledge that has much similarity to our method of differential scaling of dimensions: differential weightings of features in feature weight initializations when using online methods such as Winnow. A better understanding of effective ways of incorporating (prior) knowledge in various learning scenarios is a promising research direction.

8. Conclusions and Future Work

We have demonstrated experimentally that for learning with few labeled examples good (oracle-based) feature selection is extremely useful. As the number of examples increases, the "vocabulary" of the system, in other words, the effective feature set size for best performance, also needs to increase. A teacher, who may not necessarily be knowledgeable in machine learning, but has prior knowledge on the relevance of the features, can help accelerate training the system by pointing out the potentially important features for the system to focus on. We conducted a user study to see how well naive users performed as compared to a feature oracle in the domain of text categorization. Our technique weighted the features marked relevant by the users more than the other features. We

used our users' outputs in realistically simulated *human in the loop* experiments and observed a significant increase in learning performance with our techniques over plain active learning.

In summary, our contributions are:

1. We demonstrated that access to a feature importance oracle can improve performance (the $F1$ score) significantly over uncertainty sampling, even with as few as 7 examples labeled.

2. We found that even naive users can provide effective feedback on the most relevant features (about 60% accuracy of the oracle in our experiments).

3. We measured the manual costs of relevance feedback on features versus labeling documents: we found that feature feedback takes about one fifth of the time taken by document labeling on average.

4. We devised a method of simultaneously soliciting class labels and feature feedback that improves classifier performance significantly over soliciting class labels alone.

Consider a user who is interested in training a personalized news filter that delivers news stories on topics of their interest as and when they appear in the news. The user is probably willing to engage in some form of interaction in order to train the system to better suit their need. Similarly a user wanting to organize their e-mail into folders may be willing to train the e-mail filter as long as training is not too time consuming. Both the news filter and the e-mail filter are document classification systems. The idea of using as few documents as possible for training classifiers has been studied in semi-supervised learning and active learning. In this paper we extended the traditional active learning setting which concerns the issue of minimal feedback and proposed an approach where the user provides feedback on features as well as documents. We showed that such an approach has good potential in significantly decreasing the overall amount of interaction required for training the system.

This paper points to three promising inter-related questions for further exploration. The first question concerns what to ask from the user. In general, the active learner has to make decisions at various time points during active learning regarding the choice of feedback. For example, whether to ask for feedback on a document or on a feature, or even whether to stop asking questions all together (ask nothing), appropriate for a scenario where no additional feedback is likely to improve performance significantly. This involves some implicit or explicit assessment of the expected benefits and costs of different kinds of feedback. Furthermore, there are alternate kinds of feedback that are potentially useful – feedback on clusters of features for example. The second question involves human computer interaction issues and seeks to explore how to translate what the learner needs to know, into a question, or a user interface, that the human teacher can easily understand. In our case, the learner asked the teacher labels on word features and documents, both of which required little effort on the part of the teacher to understand what was being asked of him. Our subjects did indeed find labeling words without context a little hard, and suggested that context might have helped. An attractive alternative or complementary method of soliciting feature feedback is asking users to highlight some relevant or predictive terms as they read a document. Experiments in this direction have been conducted in information retrieval (Croft and Das, 1990). The third question is about the choice of learning algorithms for effectively incorporating these alternate forms of feedback. We explored one method in this paper and discussed alternatives in Section 7. Related to the above is better understanding and quantifying the potential of active learning enhanced with feature feedback

as a function of various aspects of the learning problem, such as measures of the difficulty of the category that one seeks to learn.

Acknowledgments

This work was supported in part by the Center for Intelligent Information Retrieval and in part by SPAWARSYSCEN-SD grant number N66001-02-1-8903. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor. We would also like to thank our users who willingly labeled data. We are also grateful to the action editor and the anonymous reviewers who helped enhance this paper with their useful comments.

Appendix A. Class Key

The class key for the Reuters corpus is given below:

1. earnings	2. acquisitions	3. money-fx	4. crude	5. trade	6. interest	7. wheat
8. corn	9. money supply	10. gold				

The class key for the 20 Newsgroups corpus is given below:

1. alt.atheism	2. comp.graphics	3. comp.os.wind.misc	4. comp.sys.ibm.pc.hw
5. comp.sys.mac.hw	6. comp.windows.x	7. misc.forsale	8. rec.autos
9. rec.motorcycles	10. rec.sport.baseball	11. rec.sport.hockey	12. sci.crypt
13. sci.electronics	14. sci.med	15. sci.space	16. soc.rel.christian
17. talk.politics.guns	18. talk.politics.mideast	19. talk.politics.misc	20. talk.religion.misc

Similarly the class key for the TDT corpus is:

1. Cambodian government coalition	2. Hurricane Mitch	3. Pinochet Trial
4. Chukwu Octuplets	5. Bin Laden Indictment	6. NBA Labor Disputes
7. Congolese Rebels	8. APEC Summit Meeting	9. Anti-Doping Proposals
10. Car Bomb in Jerusalem		

Appendix B. Instructions for Annotating Features

Class 1: Documents from the Usenet newsgroups that discuss baseball
Class 2: Documents from the Usenet newsgroups that discuss hockey

Instructions: You will be shown a list of features one at a time. For each feature you will be asked to determine whether it is relevant or not for the given classification problem. If it is relevant to Class 1 or to Class 2, mark the radio button which says “Relevant”. If it is not relevant or you don’t know whether the feature is relevant mark DONT KNOW correspondingly

A feature is relevant if it helps discriminate between documents in Class 1 versus documents in Class 2. Features are words, pairs of words (bi grams) and so on. Think of a bi gram as a pair of words that may occur in close proximity to each other For every feature ask yourself the following question: “Is this more likely to occur in a document in Class 1 as opposed to Class 2?”. If that is the

case mark the feature as relevant. If the reverse is true then again mark the feature as relevant. If the feature is not really relevant, for example "banana" may make no sense in trying to find documents in either class mark the "Not relevant/Don't know" option. DO NOT use any resources(the web, encyclopedias etc) to determine your answer. If you are not sure simply click the "Don't Know" option

The time between which you are shown a feature and you hit the submit button is timed. So do not do anything else in this time. After you submit, A THANK YOU page is displayed. You may take a break here before you proceed to the next feature.

To modify the last annotation use the browsers BACK button.

To begin annotating click here.

Appendix C. Instructions for Annotating Documents

Class 1: Documents from the Usenet newsgroups that discuss baseball
Class 2: Documents from the Usenet newsgroups that discuss hockey

Instructions: You will be shown a list of documents one at a time. For each documents you will be asked to determine whether it belongs to class 1 or class 2. You also have the option to mark a document as DONT KNOW. Read as much of the document as is needed to make an informed judgment. The time between which you are shown a document and you hit the submit button is timed. So do not do anything else in this time. After you submit, A THANK YOU page is displayed. You may take a break here before you proceed to the next document.

To modify the last annotation use the browsers BACK button

To begin annotating click here

Appendix D. End of Labeling Survey

Please take 2 minutes to fill out the following:

1. How easy was it to mark features?
   (a) On an integer scale of 1-5 (1=very difficult, 5=very easy) (b) Remarks:

2. How easy was it to mark documents?
   (a) On an integer scale of 1-5 (1=very difficult, 5=very easy) (b) Remarks:

3. For each of the following tasks please state your domain knowledge (only if you did relevance assessments for them) on a scale of 1-5 (1=very little, 5=expert):
   (a) Baseball versus Hockey. (b) Earnings versus All.
   (c) Automobiles versus Motorcycles. (d) Hurricane Mitch versus all.
   (e) Middle eastern crisis versus all.

4. Your Internet connection
   (a) DSL/Cable (b) T1 LAN (c) Dial-up

References

J. Allan. *Topic detection and tracking*. Kluwer Academic Publishers, 2002.

D. Angluin. Computational learning theory: survey and selected bibliography. In *Proceedings of the 24th Annual ACM Symposium on the Theory Computation*, pages 351–369, 1992.

P. Anick. Using terminological feedback for web search refinement: a log-based study. In *Proceedings of SIGIR '03: The 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 88–95, 2003.

Y. Baram, R. El-Yaniv, and K. Luz. Online choice of active learning algorithms. In *Proceedings of ICML 03: The 20th International Conference on Machine Learning*, pages 19–26, 2003.

E. B. Baum and K. Lang. Query learning can work poorly when human oracle is used. In *International Joint Conference in Neural Netwroks*, 1992.

P. Beineke, T. Hastie, and S. Vaithyanathan. The sentimental factor: Improving review classification via human-provided information. In *Proceedings of ACL 04: The 42nd Meeting of the Association for Computational Linguistics, Main Volume*, pages 263–270, 2004.

N. J. Belkin, C. Cool, D. Kelly, S. J. Lin, S. Y. Park, J. Perez-Carballo, and C. Sikora. Iterative exploration, design and evaluation of support for query reformulation in interactive information retrieval. *Information Processing and Management*, 37(3):403–434, 2001.

J. Brank, M. Grobelnik, N. Milic-Frayling, and D. Mladenic. Feature selection using linear support vector machines. Technical report, Microsoft Research, 2002.

C. C. Chang and C. J. Lin. Libsvm: a library for support vector machines. Available electronically at `http://www.csie.ntu.edu.tw/cjlin/libsvm`.

J. Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:27–46, 1960.

D. A. Cohn, L. Atlas, and R. E. Ladner. Improving generalization with active learning. *Machine Learning*, 15(2):201–221, 1994.

W. B. Croft and R. Das. Experiments with query acquisition and use in document retrieval systems. In *Proceedings of SIGIR '90: The 13th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 349–368, 1990.

G. DeJong and R. Mooney. Explanation-based generalization: an alternative view. *Machine Learning*, 1(2):145–176, 1986.

F. Diaz and J. Allan. When less is more: Relevance feedback falls short and term expansion succeeds at HARD 2005. In *Text REtrieval Conference (TREC 2005) Notebook*. Dept. of Commerce, NIST, 2005.

C. Domeniconi and D. Gunopulos. Incremental support vector machine construction. In *Proceedings of ICDM 01:2001 IEEE International Conference on Data Mining*, pages 589–592, 2001.

S. Godbole, A. Harpale, S. Sarawagi, and S. Chakrabarti. Document classification through interactive supervision of document and term labels. In *Proceedings of PKDD 04: The 8th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 185–196, 2004.

G. F. Hughes. On the mean accuracy of statistical pattern recognizers. *IEEE Transactions on Information Theory*, 14:55–63, 1968.

T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *ECML 98: The 10th European Conference on Machine Learning*, pages 137–142, 1998.

T. Joachims. Transductive inference for text classification using support vector machines. In *ICML '99: Proceedings of the Sixteenth International Conference on Machine Learning*, pages 200–209, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc. ISBN 1-55860-612-2.

R. Jones. *Learning to extract entities from labeled and unlabeled text*. PhD thesis, Carnegie Mellon University, Pittsburgh, USA, 2005.

G. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.

K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of ICML 95: The 12th International Conference on Machine Learning*, pages 331–339, 1995.

D. D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In *Proceedings of ECML 98: 10th European Conference on Machine Learning*, pages 4–15, 1998.

D. D. Lewis and J. Catlett. Heterogeneous uncertainty sampling for supervised learning. In *Proceedings of ICML 94: The 11th International Conference on Machine Learning*, pages 148–156, 1994.

N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2(4):285–318, 1988.

D. J. Lizotte, O. Madani, and R. Greiner. Budgeted learning of naive-bayes classifiers. In *Proceedings of UIA 03: The 19th Conference on Uncertainty in AI (UAI)*, 2003.

A. K. McCallum. Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering. Available electronically at `http://www.cs.cmu.edu/~mccallum/bow`, 1996.

T. Mitchell, R. Keller, and S. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1:47–80, 1986.

M. J. Pazzani and D. Kibler. The role of prior knowledge in inductive learning. *Machine Learning, 9 , 54-97.*, 9, 1992.

M. Porter. An algorithm for suffix stripping. *Automated Library and Information Systems*, 14(3): 130–137, 1980.

H. Raghavan, O. Madani, and R. Jones. Interactive feature selection. In *Proceedings of IJCAI 05: The 19th International Joint Conference on Artificial Intelligence*, pages 841–846, 2005.

T. G. Rose, M. Stevenson, and M. Whitehead. The Reuters Corpus Vol. 1 - from yesterday's news to tomorrow's language resources. In *Proceedings of International Conference on Language Resources and Evaluation*, 2002.

G. Salton. *Automatic information organization and retrieval*. McGraw Hill, 1968.

R. Schapire, M. Rochery, M. Rahim, and N. Gupta. Incorporating prior knowledge into boosting. In *Proceedings of ICML 02: The 19th International Conference on Machine Learning*, 2002.

G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *Proceedings of ICML 00: The 17th International Conference on Machine Learning*, pages 839–846, 2000.

F. Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1): 1–47, 2002.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *Journal of Machine Learning Research*, 2:45–66, 2002. ISSN 1533-7928.

E. M. Voorhees and L. P. Buckland, editors. *Text REtrieval Conference (TREC 2005) Notebook*, 2005. Dept of Commerce, NIST.

X. Wu and R. Srihari. Incorporating prior knowledge with weighted margin support vector machines. In *Proceedings of KDD 04: Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 326–333, 2004.

X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005.