Active Learning for e-Rulemaking: Public Comment Categorization

Stephen Purpura
Information Science Program
Cornell University
Ithaca, NY USA
sp559@cs.cornell.edu

Claire Cardie
Faculty of Computing and Information Science
Cornell University
Ithaca, NY USA
cardie@cs.cornell.edu

Jesse Simons
Dept. of Computer Science
Cornell University
Ithaca, NY USA
jmsimons@cs.cornell.edu

ABSTRACT

We address the e-rulemaking problem of reducing the manual labor required to analyze public comment sets. In current and previous work, for example, text categorization techniques have been used to speed up the comment analysis phase of e-rulemaking — by classifying sentences automatically, according to the rule-specific issues [2] or general topics that they address[7, 8]. Manually annotated data, however, is still required to train the supervised inductive learning algorithms that perform the categorization. This paper, therefore, investigates the application of *active learning* methods for public comment categorization: we develop two new, general-purpose, active learning techniques to selectively sample from the available training data for human labeling when building the sentence-level classifiers employed in public comment categorization. Using an e-rulemaking corpus developed for our purposes [2], we compare our methods to the well-known query by committee (QBC) active learning algorithm [5] and to a baseline that randomly selects instances for labeling in each round of active learning. We show that our methods statistically significantly exceed the performance of the random selection active learner and the query by committee (QBC) variation, requiring many fewer training examples to reach the same levels of accuracy on a held-out test set. This provides promising evidence that automated text categorization methods might be used effectively to support public comment analysis.

Keywords

active learning, e-rulemaking, machine learning, text categorization, public comment

1. INTRODUCTION

Each year federal regulatory agencies issue more than 4,000 new rules [6]. By law, many of these must be created through a complex and expensive process in which the agency drafts a proposed rule and then exposes the proposal, any underlying data, and its legal and policy rationale to public comment. This process, *notice and comment (N&C) rulemaking*, is the mechanism through which most agencies make major regulatory policy.

In N&C rulemaking, the agency may receive anywhere from dozens, to hundreds of thousands, of comments, depending on the subject and complexity of the rule. The agency's fundamental legal obligation is to review all the comments received and, if it chooses to adopt the proposed rule, to issue a statement that not only (i) demonstrates why its choice is within its statutory authority and sound as a matter of regulatory policy, but also (ii) responds to significant criticisms made in the comments and explains why it rejected alternative approaches suggested there [19][1]. The stakes for the agency are high. Failure to adequately address critical comments and discuss alternatives in the statement accompanying the final rule can lead a court to invalidate the rule — thereby requiring still more agency time and effort to perform additional review and explanation [3, 19][2].

Electronic rulemaking (e-rulemaking) includes a wide range of ways that information technology might be used in rulemaking. It includes, but is not limited to: converting the agency's docket (the filing system showing all its activities, including rulemaking) to electronic form and making it available via the Internet; allowing submission of comments via e-mail and the Internet in addition to (and perhaps eventually in place of) conventional mail and fax; and using search engines, hyperlinked text, and other IT capacities to allow both the public and agency rulewriters to find, sort, and link the massive amount of material relevant in a rulemaking more easily and cheaply than could possibly be done with hardcopies.

This paper focuses on the application of information technology to the comment analysis phase of e-rulemaking. To expedite this time-consuming, but critical, stage of the e-rulemaking process[3], researchers have begun to investigate methods from natural language processing, information retrieval, and machine learning. Yang & Callan [21, 22], for example, extend duplicate detection methods from information retrieval to handle "e-postcard campaigns", i.e. e-mail

[1] At 524-50.
[2] Strauss et al. at 524-50 and 1016-26.
[3] It is important to note that review of public comments by government representatives is usually a statutory requirement [3].

campaigns organized by special interest groups that supply constituents with electronic form letters for submission during the comment period. When comments were submitted on paper, modifying the form letters was difficult — the letter would need to be re-typed to add or remove text. As a result, most form letters were exact duplicates of one another. These are fairly easy to identify and need be analyzed for content only once. In electronic form, however, form letters are very easy to change, and it is exactly the changed snippets that agency rulewriters want to locate to determine if the modification introduces substantive information not present in the original. The Yang & Callan [21, 22] work develops automatic methods to identify these *near-duplicate* submissions and to delineate the modified portions from the original letter.

In addition, Kwon et al. [8] and Kwon & Hovy [7] investigate the use of natural language processing methods to identify the main claims of a comment and then categorize them according to whether they support the proposed rule, oppose the proposed rule, or are proposing a new idea.

Finally, and more relevant for our work, Kwon et al. [8] and Kwon & Hovy [7] also apply text categorization methods from machine learning to automate the comment sorting process. In particular, they develop a set of eight general topic codes — ECONOMIC, ENVIRONMENT, GOVERNMENT RESPONSIBILITY, HEALTH, LEGAL, POLICY, POLLUTION, and TECHNOLOGY and train a machine learning algorithm (they use a support vector machine (SVM) [20]) to classify individual sentences according to the topics they address. As is the case for all supervised inductive machine learning algorithms, SVM-based text categorization methods require an initial "training phase" in which the learning algorithm is provided with many examples of the task to be learned. For comment categorization, the algorithms need training data in the form of sentences from public comments that have been manually annotated with their correct subtopic codes. After training, the SVM can then decide which, if any, of the subtopic codes applies to sentences that it has not seen before. As in the near-duplicate detection research, the goal of subtopic categorization is to speed up the (required) manual review of public comments, in this case by grouping similar comment snippets so that rulewriters can read and respond to them as a whole.

In recent work, we show that text categorization methods can be used to categorize public comments not only with respect to a small set of general subtopic codes, but according to the much larger, and often hierarchical, set of *rule-specific issues* employed during comment analysis [2]. Our agency collaborators — rulewriters from the U.S. Department of Transportation (DOT) and the U.S. Department of Commerce — indicate that this finer-grained categorization corresponds more closely to what is actually done as a first step in the comment analysis process: analysts map each portion of a comment to the substantive rule-specific issue(s) that it addresses, if any.

This paper extends previous e-rulemaking research in comment categorization in a different direction: we investigate the use of **active learning** algorithms as a means of significantly reducing the amount of human time required to provide annotated training data for the learning-based text categorization methods. Since the practical purpose of employing machine learning methods is, in the first place, to reduce the strain on agency rulewriters, reducing the total amount of manual issue annotation necessary for accurate results is a means to those ends; and the goal of active learning is exactly that.

We present two new approaches to active learning. Each extends the well known "query by committee" (QBC) algorithm [5]. The first extension exploits the hierarchical nature of issue categorization to select the most uncertain instance w.r.t the category to be labeled next. We call this the Hierarchical Query by Committee (HQBC) method. The second extension, Hierarchical Query by Committee by Clustering (HQBCBC), builds on HQBC and relies on clustering methods to select the instance that maximizes both uncertainty and influence for the learning algorithm. More specifically, HQBCBC selects instances (a) that are maximally confusing for the committee of classifiers and (b) whose correct classification is likely to (positively) affect the largest number of other instances in the data set.

Both approaches are evaluated on the same e-rulemaking data sets — the CeRI[4] FTA Grant Circulars Corpus [2]. Our results show that both HQBC and HQBCBC obtain set precision levels faster than randomly selecting training instances for labeling, usually more than twice as fast. Furthermore, HQBC and HQBCBC exceed the performance of standard QBC on five, and all, of the six FTA data sets, respectively. Although additional research is required, we conclude that our results provide encouraging evidence that an active learning approach to issue categorization of public comments for e-rulemaking is feasible.

2. ADDITIONAL RELATED WORK

Related work in e-rulemaking was discussed above. This section describes related work on active learning for text categorization from the fields of machine learning and information retrieval.

The goal of active learning algorithms is to process (unlabeled) training examples in the order in which they are most useful or informative to the learning algorithm [4]. In each iteration of active learning, one or more such instances are selected, labeled with respect to their correct classification or category (usually by a person), and added to the set of examples used to train the classifier; the classifier is then retrained with this (slightly larger) training set. Once retrained, the classifier is applied to the remaining unlabeled instances in the data pool and the process repeats — either for a fixed number of iterations or until performance of the resulting classifier is good enough.

In the active learning setting, "usefulness" is commonly quantified as the learner's uncertainty about the class of an instance [9] — the more uncertain the learning algorithm is w.r.t. an instance, the more useful that instance will be to the learner. One of the standard methods for choosing uncertain instances for labeling during active learning is the query by committee (QBC) approach [5]. In this method,

[4]Cornell e-Rulemaking Initiative, URL: ceri.law.cornell.edu.

three or more different classifiers are trained using the available labeled training data, and in each round of active learning, an instance for which there is the most classification disagreement within the ensemble is selected for labeling (see, for example, Seung et al. [17] and Muslea et al. [11]).

The active learning algorithms we develop here are, at their core, QBC active learners. In general, a QBC classifier ensemble can be created by training different learning models (using the same feature set) for each classifier (e.g. an SVM, a decision tree, and a k-nearest neighbor algorithm) or by using a single learning model for each classifier, but encoding the training instances for each using a different feature set (e.g. Muslea et al. [11]).

Another common "selective sampling" method for active learning relies on the confidence associated with classifications made by a single classifier [9]. And there are many variations of active learning that aim to increase diversity among the selected sample (see, for example, Brinker [1] and Melville & Mooney [10]). Still others have focused on the development of active learning variants for specific learning algorithms (e.g. Brinker [1] and Scholkopf & Smola [16] do so for SVMs).

In this paper, we develop two new active learning algorithms, neither of which is specific to the e-rulemaking domain: both are general-purpose extensions of the QBC approach to text categorization. In addition, both approaches handle hierarchical categories while previous approaches assume a flat topic list.

In the remainder of the paper, we describe the e-rulemaking corpus from which our data is derived (Section 3); the new active learning algorithms (Sections 4 and 5); our experimental methodology (Section 6); and, finally, the results of our experiments (Section 7).

3. THE FTA GRANT CIRCULARS CORPUS

Working with analysts from the Federal Transit Authority (FTA) in the Department of Transportation, we identified FTA "guidance circulars" that the agency proposed to issue. Such circulars are a type of document on which the FTA frequently seeks public comments. Here, the proposed advice involved grants under three federal statutes that fund local transportation services for the elderly, disabled persons, and low income persons commuting to work.[5]

The 267 comments submitted electronically during the circular's 45-day comment period comprise the CeRI FTA Circular Grants Corpus. Next, we constructed a list of 38 issues likely to be raised in the comments. This list was derived by consulting both the actual issue summaries prepared by the FTA analyst when she reviewed the comments, and the Federal Register notice seeking comments, which explained the proposal in detail and highlighted various aspects. The issues are organized into a shallow categorization hierarchy in which the 38 issues are leaf nodes. Seventeen issues form the first level of the hierarchy, five of which expand into two or more sub-issues at level two. The issue hierarchy is shown with abbreviated category names in Figure 1. NONE is a special category (shown as the 39th "issue") that is automatically assigned to sentences deemed by the annotator to address none of the rule-specific issues.

A small team of law school students annotated the comments at the sentence-level w.r.t. the fine-grained issue set. They are free to assign more than one issue to a single span of text, if warranted, but rarely do so (4% of the sentences in the corpus with a maximum of three issues per sentence).

In all, there are 11,094 sentences in the corpus. On average, there are 41.55 sentences per comment. The shortest comment has one sentence; the largest has 1420 sentences. Each comment has an average of 41.55 sentences with a minimum of one sentence per comment and a maximum of 1420 sentences. Our experiments report results for three of the annotators randomly chosen from the set of six.

146 of the 267 comments were used for the interannotator agreement study, with an average of 2.66 annotators per comment. Because there can be multiple issues per sentence and the annotators covered different numbers and subsets of the documents, we currently measure interannotator agreement using a basic agreement (AGR) measure (rather than Fleiss' kappa): for all pairs of annotators across all comments that were annotated by both annotators, we calculate the percentage of sentences for which the annotators assign overlapping issue labels. In most cases, this amounts to checking for an exact issue match (since 96% of the sentences are assigned a single issue).

For the fine-grained 39 issue set, the average interannotator agreement score is 46.4%; for the 17 top-level issues, agreement goes up to 64.7%. Although somewhat low, the scores are comparable to those in the subtopic categorization studies of Kwon et al. [8]. Additional details regarding the CeRI FTA Grant Circulars Corpus and its creation can be found in Cardie et al. [2].

4. HQBC: THE HIERARCHICAL QUERY BY COMMITTEE ALGORITHM

As in standard Query by Committee (QBC) active learning algorithms, our Hierarchical Query by Committee (HQBC) algorithm employs three classifiers in its ensemble, each a different model type. In contrast to QBC, however, HQBC aims to exploit the category hierarchy when assigning a "disagreement score" to each instance during active learning iterations. The more that the classifiers disagree w.r.t. the detailed and top-level issues to associate with an instance, the better that instance is for active learning. For each model type in the ensemble, HQBC trains two classifiers:

COARSE classifies an instance according to the 17 top-level issues, and

DETAILED classifies an instance according to the 39 more detailed issues.

Then, for any particular instance in the data pool, HQBC makes six decisions — one by the COARSE classifier and one

[5]Docket No. FTA-2006-24037: Elderly Individuals and Individuals With Disabilities, Job Access and Reverse Commute, and New Freedom Programs: Coordinated Public Planning Guidance for FY 2007 and Proposed Circulars.

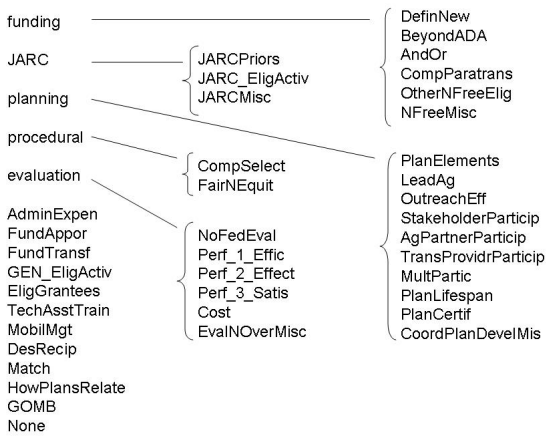

Figure 1: Issue Hierarchy

by the DETAILED classifier for each of the three model types. The disagreement score for the instance can then be calculated as follows (where a higher number indicates more disagreement and, hence, more uncertainty w.r.t. the class value)[6]:

score=7	when	none of the COARSE predictions agrees
score=6	when	2 of 3 COARSE predictions agree
	and	none of the DETAILED predictions agrees
score=5	when	2 of 3 COARSE predictions agree
	and	2 of 3 DETAILED predictions agree
score=4	when	all 3 COARSE predictions agree
	and	none of the DETAILED predictions agrees
score=3	when	2 of 3 COARSE predictions agree
	and	all 3 DETAILED predictions agree
score=2	when	all 3 COARSE predictions agree
	and	2 of 3 DETAILED predictions agree
score=1	when	all 3 DETAILED predictions agree
	and	all 3 COARSE predictions agree

Once the score is calculated for all instances in the unlabeled data pool, they are sorted in decreasing numerical order by score and the top n records are selected for manual annotation and removed from the unlabeled data pool. After an annotator supplies the correct category for the selected instances, they are added to the training data, the ensemble classifiers are re-trained, and the process repeats.

[6]The algorithm used in our experiments does not verify that the DETAILED prediction is from the correct subclass of COURSE for scoring purposes. Modifying the HQBC algorithm to account for this situation slightly improves performance for the data set used in these experiments.

5. HQBCBC: THE HIERARCHICAL QUERY BY COMMITTEE BY CLUSTERING ALGORITHM

The Hierarchical Query by Committee by Clustering (HQBCBC) algorithm builds on the HQBC algorithm of Section 4. Its goal is to identify useful instances in the unlabeled data pool when the HQBC algorithm is not particularly discriminating, i.e. when HQBC assigns the same maximum disagreement scores to many instances. In particular, HQBCBC aims to select instances (a) that are maximally confusing for the committee of classifiers (as in HQBC) and (b) whose correct classification is likely to (positively) affect the classification of most additional instances in the data set. It accomplishes this by using the HQBC disagreement scores as an estimate of the true uncertainty score to assign each unlabeled instance and by relying on clustering to determine an instance's influence.

More specifically, each round of HQBCBC active learning operates as follows. We will refer to those instances with the maximal HQBC disagreement score for the round as "seed" instances. For each seed instance, *seed*,

1. Place *seed* in its own cluster, $cluster_{seed}$.
2. Calculating distance using a standard cosine similarity measure, find all instances in the unlabeled data pool within a distance of d from *seed* and add them to $cluster_{seed}$.
3. Calculate the influence score of *seed* as the size of its cluster:
   $influence(seed) = |cluster_{seed}|$

Note that instances from the data pool can be part of more than one cluster. In addition, the distance threshold d is a parameter that should be set empirically based on the training data.

With a disagreement score, *disagree*, and an influence score, *influence*, in hand for each seed instance, HQBCBC then

sorts the seeds according to decreasing *disagree* * *influence*. Finally, for each of the top n seeds, *seed*, HBQBCBC returns one randomly selected instance from $cluster_{seed}$ for manual annotation and removes *seed* from the unlabeled data pool. After manual annotation, the n newly labeled instances are added to the training data, the ensemble classifiers are retrained, and the process repeats.

Intuitively, HQBCBC prefers selectively sampling a "confusing" sentence for annotation if that sentence has more near duplicates in the corpus. Consider an extreme case where 10% of the corpus is a duplicate of a single sentence, all of the duplicates of the sentence are unannotated, and the classifiers found the sentence to be confusing. In this circumstance, the algorithm would prefer the annotation of this sentence over another sentence that was equally confusing but lacked the same number of duplicates. The worst case for this algorithm occurs when multiple sentences are nearly identical in their use of words yet have interpretations that result in different classifications. For our data sets, this is apparently not a problem.

6. METHODOLOGY

Section 3 briefly described the creation of the CeRI FTA Grant Circulars Corpus used in our experiments. There are a few options for creating a gold standard from this corpus. We might create an "aggregate" gold standard that covers the entire comment set, comprised of comments whose annotations have been reconciled by a pair of annotators. A second option — one that more closely approximates what we understand, from our agency partners, to be real-world agency practice — involves creating one gold standard per annotator. In particular, when more than one analyst reviews a comment set to find, extract, and organize the issue references for subsequent analysis and preparation of the accompanying final statement, these analysts typically divide the issues among themselves: each reads all the comments, taking responsibility for collecting material as to his or her allotted issues. As a result, there typically is not more than one "annotator" per issue in the real-world. The gold standard under this annotation scheme would then be the union of the issue-specific annotations of each analyst.

We have adopted yet a third strategy for creating a gold standard for the purposes of this paper. In particular, we are interested in investigating the ability of the text categorization algorithms to learn to duplicate the annotations produced by an arbitrary agency analyst. As a result, we treat the annotations of each annotator as a separate gold standard, producing six separate corpora. We report results for three of the six annotators. (There is nothing substantially different across the set of six vs. the set of three.)

We evaluate our algorithms w.r.t. both the COARSE (17 categories, 16 issues plus NONE) and the DETAILED (39 categories, 38 issues plus NONE) issue sets. Table 1 specifies the instance counts, i.e. the number of sentences, associated with each data set.

We measure the performance of our classifiers using precision. (Here, precision is equivalent to accuracy because one categorization decision is made for each sentence in the corpus.) For the active learning curves, we report precision across the entire data set. Classifiers that produce higher levels of precision with fewer training instances, i.e. fewer rounds of active learning, are better. To compare classifiers we use paired two sample for means t-tests.

Data Set Name	Instance Count
ad	1,318
hlc	1,119
lck	1,601

Table 1: Number of Instances per Data Set

6.1 Text Pre-processing

Input to text categorization systems is usually pre-processed to create word/term vectors for each training and test instance [15]. In addition, the word-based feature vectors are associated with a corresponding weight vector that ascribes a different weight to each word. Before creating word vectors, we remove non-word tokens, map text to lower case, and then apply the Porter Stemming Algorithm described in Porter [13].

Weighting strategies such as *tf-idf* (i.e. term frequency multiplied by inverse document frequency) have been shown to be generally effective, but specialized weighting schemes often provide improvements [12]. After empirical testing of various weighting schemes on the training data, this work adopts a term weighting strategy related to mutual information, which is the ratio of sentence-based word frequency and the overall frequency of the word across the corpus. Equation 1 for the feature value w_i is shown:

$$w_i = log\left(\frac{p(w|s)}{p(w)}\right) \quad (1)$$

In equation 1, the top term, $p(w|s)$, is the probability of a word in a particular sentence (the number of occurrences in each sentence, divided by the number of total words in the sentence). The denominator term $p(w)$ is the probability of a word across all sentences (the number of occurrences of this word in all sentences, divided by the total number of words in all sentences).

Finally, only words with $w_i > 0$ are included in the sentence-based term vectors.[7]

6.2 Classifiers and Parameters for the Active Learners

For the ensembles in each of the QBC, HQBC, and HQBCBC algorithms, we employ a Support Vector Machine (SVM), a Maximum Entropy classifier, and a Naive Bayes classifier. In particular, we selected Mallet 0.4[8] for its Naive Bayes and Maximum Entropy implementations, and SVMlight[9] for its single-class SVM implementation. For the experiments here, we did not learn the optimal parameter settings for each

[7]The run-svm-text.pl script from Purpura and Hillard [14] performs the pre-processing steps described above and is available for download from www.stephenpurpura.com.
[8]Available at http://mallet.cs.umass.edu.
[9]Available at http://svmlight.joachims.org.

classifier based on a validation set. Rather, we ran each algorithm under a number of parameter settings and selected the settings that provided the best performance on a portion of the CeRI FTA Circulars Corpus when the classifier was used in isolation, i.e. not in an ensemble.

To support multi-class classification with SVMlight, we used the run-svm-text.pl script from footnote 7 that implements pairwise voting instead of the common one vs. the rest voting schemes.

In each round of active learning, we select n=10 instances for labeling. In practice, the selection of n will require an appropriate balance of computational cost vs. system accuracy.

7. RESULTS

Figure 2 shows the performance of the best base classifier (i.e. the best classifier among the active learning ensemble, when used in isolation) — the Maximum Entropy classifier — for each of the 17- and 39-category versions of the three annotator-specific gold standards. Not surprisingly, performance is better on the 17-category data than the 39-category data. Although not directly comparable, the results for both versions of the data approach the interannotator agreement results of 46.4% for the 39 issues and 64.7% for the 17 top-level issues.

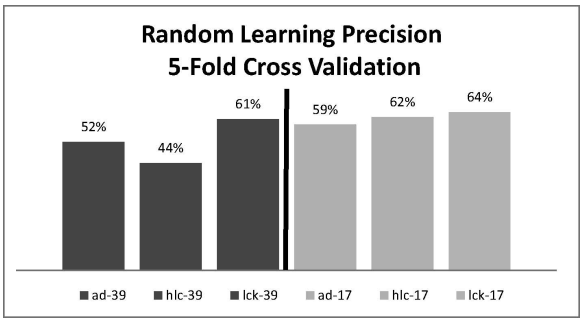

Figure 2: 5-fold cross Validation Accuracy for the MaxEnt Classifiers. Results for the 17-issue data sets are shown in black on the left; results for the 39-issue version of the data sets are shown in gray on the right.

Learning curves for the three base classifiers (labeled *svm*, *nb*, and *me*) when trained via random instance selection are shown in Figure 3 and Figure 4. The straight line (labeled *baseline*) in each Baseline graph corresponds to the performance of a classifier that selects the majority class (i.e. the issue that occurs most often) for the data set. (This is usually NONE.) In interpreting the results, the performance of the base classifiers is important for the active learning experiments because weakness of the base learners can significantly impact the overall performance of the active learning system.[10]

[10]For example, using an ensemble with weak learners, such as a classifier trained with binary presence features using unstemmed words, significantly decreases the performance of all of our active learning ensembles.

Learning curves for each of the active learning algorithms applied to each of the data sets are also shown in Figure 3 and Figure 4. In particular, we compare the HQBC and HQBCBC algorithms with the standard QBC algorithm (using the same base classifiers), which should be considered the baseline with which to compare the NQBC and HQBCBC algorithms.

In general, the graphs show that the active learners achieve higher levels of precision faster than the non-active learners. This is especially the case for the HQBC and HQBCBC algorithms. Using single tailed t-tests, we validated that the sample mean of the precision curve produced by HQBCBC was always expected to be higher than the sample mean of the curve produced by QBC. We found similar results for HQBC, with the lone exception being the 39-category experiment using the data from the 'ad' annotator. For the ad-39 experiment, the difference between the sample mean of precision produced by either the QBC or HQBC curves was expected to be 0.

Understanding hcl-17's Performance. At first glance, the results of HQBC on hcl-17 seem highly suspect because they achieve extremely strong results when compared to the associated inter-annotator agreement scores. But examining the success of the algorithm on this data set is illuminating. Uncertainty, as represented by the mean HQBC SCORE, remains higher for the hcl annotator for a larger proportion of the data set than for other two annotators. When the mean HQBC score is higher, there is greater differentiation between uncertainty for the instances, and when this translates into picking better instances for labeling, the algorithm's performance increases faster.

Figure 5 shows the mean HQBC SCORE for each data set at each point on the learning curve. In comparison, QBC's uncertainty measure (the average number of dissenting votes from the majority of the ensemble) always approaches zero faster than mean HQBC score. When the QBC ensemble votes begin to stabilize, performance reverts to levels that approach random selection.

In the early rounds of annotation, our active learning algorithms try to avoid labeling a lot of instances from highly confident and accurate instance groups. Instead, we pick instances where the learners lack confidence in the predictions on the instances. When learning the correct estimate for an instance affects the predictions for large blocks of the data, performance improves dramatically. When learning the correct estimate for a seemingly incongruent (and potentially noisy) instance, *the algorithm actually selects better in subsequent rounds.* As the Figure shows, the noise in the training data causes an increase in a subset of HQBC SCOREs. The spread associated with differentiation of HQBC SCOREs is what allows the algorithm to selectively subsample better. So noisy data helps improve the process of selective subsampling, either because the system will keep selecting from the instances that compose the confused class to improve its confidence or because the system learns to overcome the noise. When the former happens, the process of removing the confused instances from the test set takes longer. When the latter occurs, the process can be very quick.

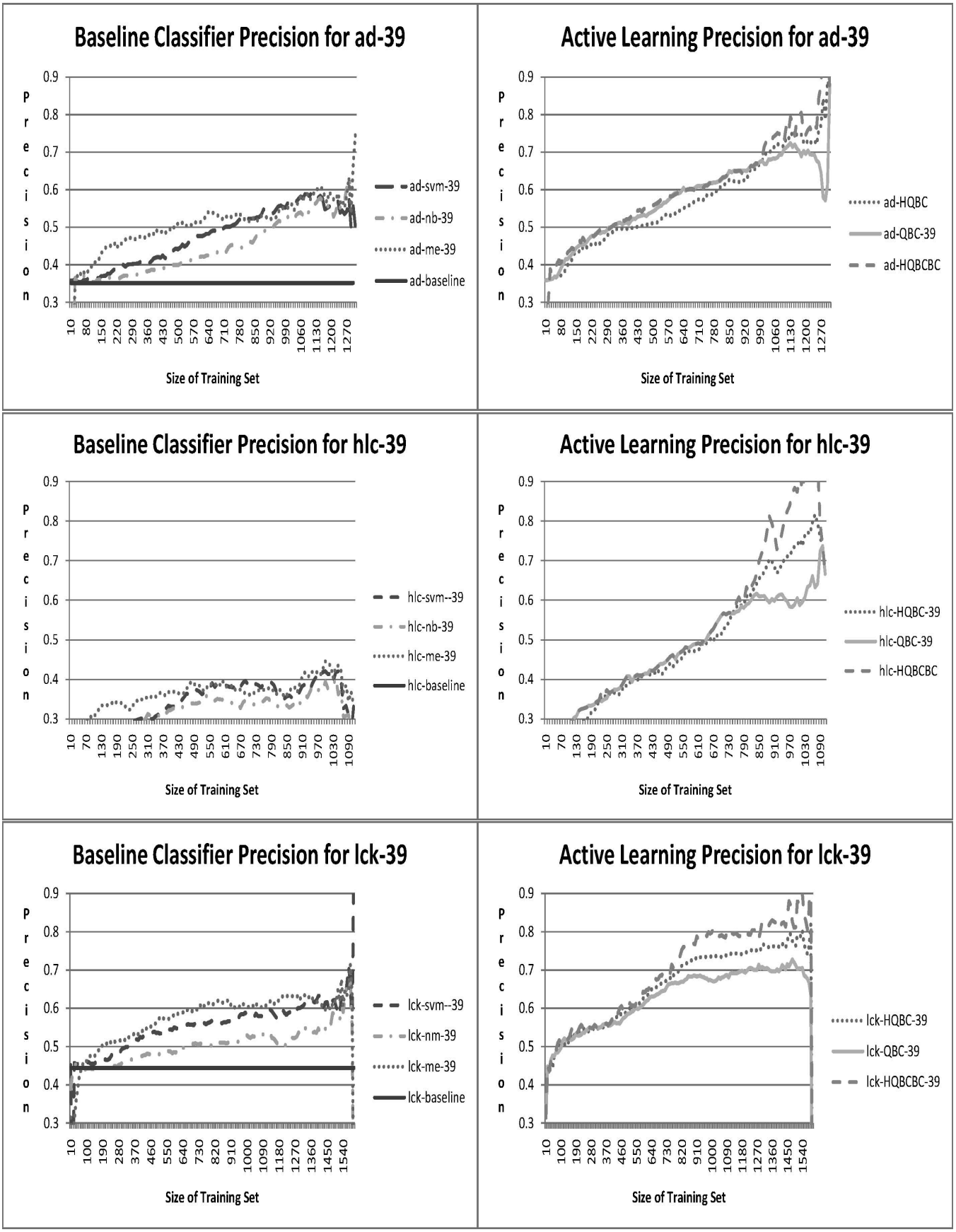

Figure 3: Baseline Classifier versus Active Learning Precision for 39 Categories

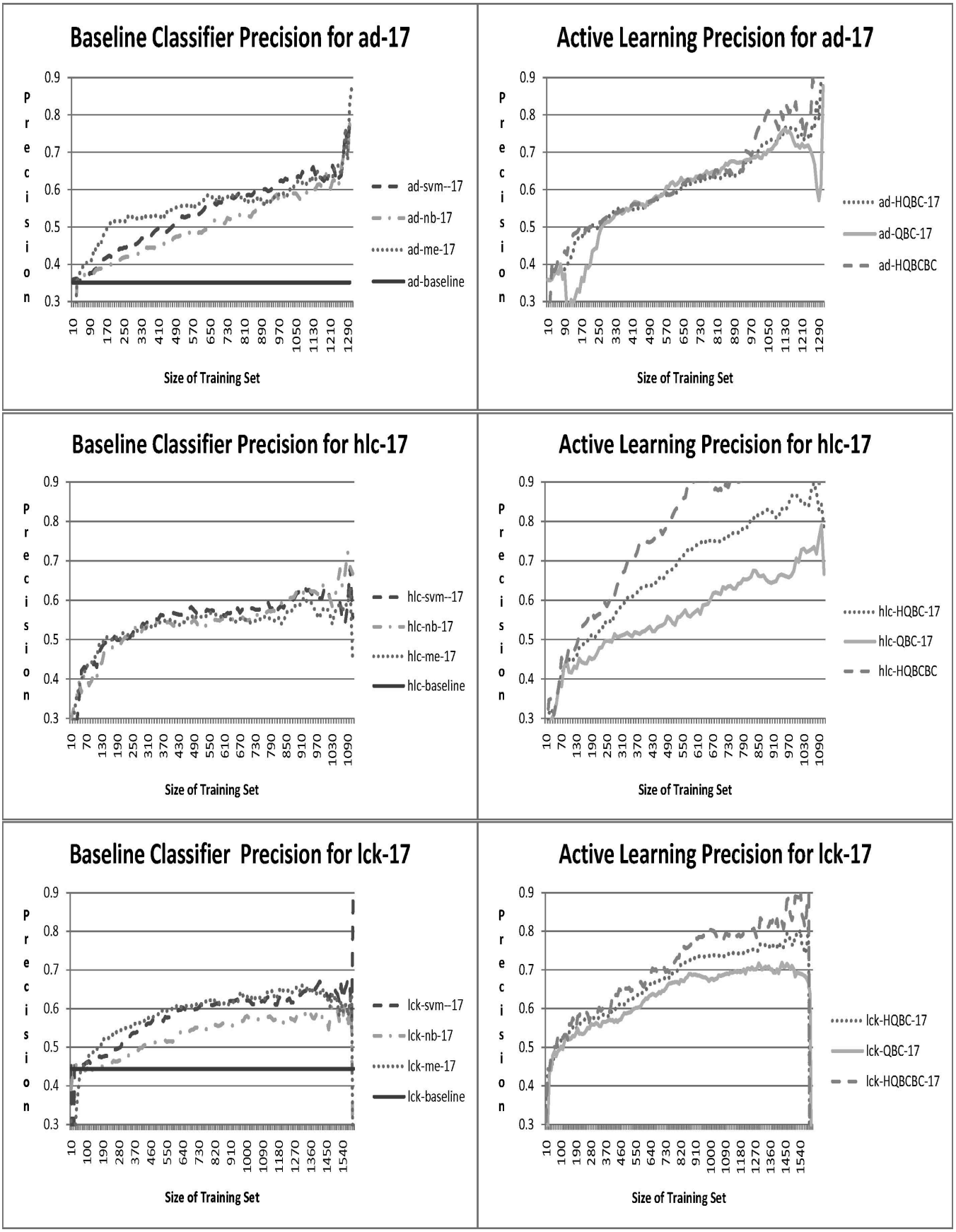

Figure 4: Baseline Classifier versus Active Learning Precision for 17 Categories

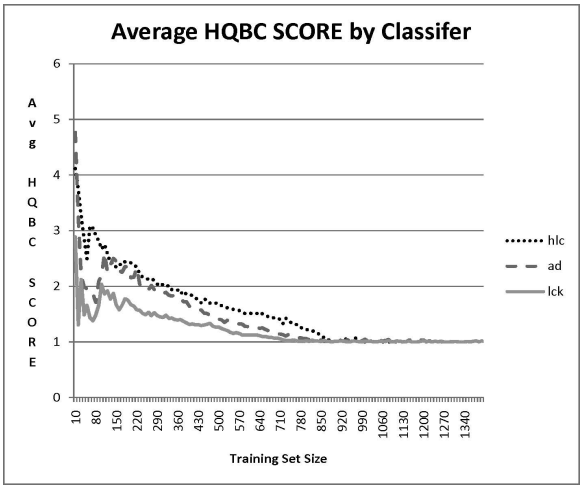

Figure 5: Average HQBC Scores for Each Annotator-Specific Data Set

Intuitively, we have created a sort algorithm that sorts a data set such that the instances at the tail end of the data set are most likely to be either near-duplicates in content of instances at the beginning of the data set or near-duplicates in a dimension determined to be important by the machine learning algorithms. Since the tail end of the data set is the 'test set', our performance against it appears solid. But this also directly translates into assistance to the user, who must only annotate records that are truly distinguishing.

7.1 Maximum Set Precision Results

Another way to compare the performance of the active learners is to examine the maximum set precision achieved by each classifier on each of the data sets (i.e. when 200 - 209 instances remain in the test set). This is shown in Table 2, in which the three active learning algorithms are compared to each other and to an algorithm that randomly selects the same number of instances in each round of active learning.

For each annotator, we see (not surprisingly) that the active learners achieve higher results on the 17-issue data set. In addition, the HQBC and HQBCBC algorithms outperform both QBC and the Random Selection active learner. Although QBC improves upon Random Selection, the 17- and 39-issue data sets perform comparably: one would expect the 17-issue data set to be consistently easier. In the results shown below, however, we will see that QBC does not do a good job at distinguishing the important instances for labeling.

In addition, the HQBCBC algorithm is able to attain essentially perfect replication of the annotator's behavior for data set hcl-17 because the patterns in the performance of the annotator are found early in the process and adding subsequent instances to the training set resolves both noise and edge cases (low instance count data categories).

8. OTHER DATA SETS

One of the unresolved issues from our work thus far is determining whether HQBC and HQBCBC will be useful on other data sets. Luckily, we can quickly gain some insight into potential performance improvements using two data sets: the Congressional Bills data set used in [14] and the Wolves data set used in [18].

The Congressional Bills data set is already annotated with respect to a hierarchical classification scheme. The classification scheme has a 21-class course-grained level and a 226-class detailed (fine-grained) level. The data set from 1948–1998 has 375,517 instances and of these, 120,927 are near-duplicates. This interesting attribute of the data was caused by a perverse incentive in the United States Congress during the early years of the period — the maximum number of bill cosponsors was limited, so Congressional Members would reintroduce the same bill multiple times but with different cosponsors. Using this full data set, an estimate for how quickly HQBCBC can reach 70% accuracy when using the 226-class detailed data sets can be achieved by computing the cosine similarity matrix for the data set and ranking by HQBCBC score (influence score multiplied by the uncertainty score). The resulting estimate for the number of instances to be labeled to achieve 70% accuracy is just less than 3,500.

Analysis of the Wolves data set[11] is more complicated. Although it consists of regulatory rule comments that are conceptually similar to the rule comments in the data set analyzed in this work, they have not been classified using a hierarchical scheme. As the first step in analysis of the scope of the problem of classifying the Wolves data set by issue, we have computed some statistics. First, it contains 254,378 comments and an estimated 8 million sentences. Of these 8 million sentences, a conservative analysis suggests that about half of the data set could be correctly classified by labeling about 800 sentences using HQBCBC. This estimate stems from the cosine similarity matrix: 4 million sentences are near-enough duplicates to form 800 clusters.

9. CONCLUSIONS

We have presented the first active learning results to date on rule-specific issue categorization in e-rulemaking. We furthermore present two new active learning algorithms that are able to take advantage of the rule-specific issue hierarchy to improve performance. Our results suggest that active learning is a promising avenue for further research for categorization of public comments according to rule-specific issues and other hierarchical classification problems. In future work, we plan to incorporate these and other active learning strategies into existing comment annotation tools and to evaluate them with rulewriters during the comment analysis phase for an active proposed rule.

10. ACKNOWLEDGMENTS

This work was supported by NSF Grant IIS-0535099.

Thank you to Jamie Callan, Stuart Shulman, Ed Hovy, and Nancy Steadle for access to the Gray Wolves data set and to John Wilkerson for the Congressional Bills data set.

11. REFERENCES

[11] Available at http://erulemaking.cs.cmu.edu/data.php.

Data Set	Random Selection	Standard QBC	HQBC	HQBCBC
ad-39	60%	71%	74%	75%
hlc-39	44%	61%	74%	81%
lck-39	66%	71%	75%	88%
ad-17	65%	71%	75%	80%
hlc-17	59%	70%	87%	99%
lck-17	67%	71%	75%	84%

Table 2: Maximum Set Precision of the Active Learning Algorithms

[1] K. Brinker. Incorporating diversity in active learning with support vector machines. In *Proceedings of ICML-03, 20th International Conference on Machine Learning*. Morgan Kaufmann Publishers, San Francisco, US, 2003.

[2] Claire Cardie, Cynthia Farina, Matt Rawding, Adil Aijaz, and Stephen Purpura. A Study in Rule-Specific Issue Categorization for e-Rulemaking. In *Proceedings of the 9th Annual International Conference on Digital Government Research*, 2008.

[3] C. Coglianese. Weak democracy, strong information: The role of information technology in the rulemaking process. In V. Mayer-Schoenberger and D. Lazer, editors, *Electronic Government to Information Government: Governing in the 21ST Century*, 2007.

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

[5] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Machine Learning*, 28:133–168, 1997.

[6] C. Kerwin. The state of rulemaking in the federal government. Technical report, Transcript Panel 1, 2005.

[7] N. Kwon and E. Hovy. Information acquisition using multiple classifications. In *Proceedings of the Fourth International Conference on Knowledge Capture (K-CAP 2007)*, 2007.

[8] N. Kwon, E. Hovy, and S. Shulman. Multidimensional text analysis for erulemaking. In *Proceedings of the 7th Annual International Conference on Digital Government Research*, 2006.

[9] D. D. Lewis and J. Catlett. Heterogeneous Uncertainty Sampling for Supervised Learning. In *Proceedings of the Eleventh International Conference on Machine Learning*, pages 148–156, Rutgers University, New Brunswick, NJ, 1994. Morgan Kaufmann.

[10] P. Melville and R. Mooney. Diverse ensembles for active learning. In *Proceedings of ICML-04, 21st International Conference on Machine Learning*. Morgan Kaufmann Publishers, San Francisco, US, 2004.

[11] I. Muslea, S. Minton, and C. Knoblock. Selective sampling with redundant views. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence*, pages 621–626, 2000.

[12] K. Papineni. Why inverse document frequency? In *Proceedings of the North American Association for Computational Linguistics, NAACL*, pages 25–32, 2001.

[13] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130 – 137, 1980.

[14] S. Purpura and D. Hillard. Automated Classification of Congressional Legislation. In *Proceedings of the 7th Annual International Conference on Digital Government Research*, 2006.

[15] G. Salton and M. McGill. *Introduction to Modern Information Retrieval*. McGraw-Hill, New York, 1983.

[16] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond (Adaptive Computation and Machine Learning)*. MIT Press, Cambridge, MA, 2002.

[17] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Computational Learning Theory*, pages 287–294, 1992.

[18] S. Shulman. Perverse incentives: The case against mass e-mail campaigns. In *Proceedings of the Annual Meeting of the American Political Science Association*, 2008.

[19] P. Strauss, T. Rakoff, and C. Farina. *Administrative Law*. 10th edition, 2003.

[20] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 1995.

[21] H. Yang and J. Callan. Near-duplicate detection for erulemaking. In *Proceedings of the Fifth National Conference on Digital Government Research*, 2005.

[22] H. Yang and J. Callan. Near-duplicate detection by instance-level constrained clustering. In *Proceedings of the Twenty-Ninth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006.