Towards Minimizing the Annotation Cost of Certified Text Classification

Mossaab Bagdouri
William Webber
CS, iSchool
University of Maryland
College Park, MD 20742, USA
{mossaab,wew}@umd.edu

David D. Lewis*
David D. Lewis Consulting
Chicago, IL 60614, USA
cikm2013pap@
DavidDLewis.com

Douglas W. Oard
iSchool/UMIACS
University of Maryland
College Park, MD 20742, USA
oard@umd.edu

ABSTRACT

The common practice of testing a sequence of text classifiers learned on a growing training set, and stopping when a target value of estimated effectiveness is first met, introduces a sequential testing bias. In settings where the effectiveness of a text classifier must be certified (perhaps to a court of law), this bias may be unacceptable. The choice of when to stop training is made even more complex when, as is common, the annotation of training and test data must be paid for from a common budget: each new labeled training example is a lost test example. Drawing on ideas from statistical power analysis, we present a framework for joint minimization of training and test annotation that maintains the statistical validity of effectiveness estimates, and yields a natural definition of an optimal allocation of annotations to training and test data. We identify the development of allocation policies that can approximate this optimum as a central question for research. We then develop simulation-based power analysis methods for van Rijsbergen's F-measure, and incorporate them in four baseline allocation policies which we study empirically. In support of our studies, we develop a new analytic approximation of confidence intervals for the F-measure that is of independent interest.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and software—*performance evaluation*.

Keywords

E-discovery; supervised learning; text categorization; evaluation

*This work was done during the author's affiliation with the University of Maryland.

1. INTRODUCTION

As text classifiers trained by machine learning have become common in practical settings, their effectiveness has become of interest not just to experimentalists, but to practitioners, decision makers, and, increasingly, the courts. This process has been accelerated in *e-discovery* (electronic discovery, i.e. finding digital evidence in legal matters) [20]. The growth of content that organizations must review, to select documents for delivery to opposing parties in civil lawsuits, or to government regulators, has increased interest in information retrieval in general, and supervised learning in particular [12, 20, 21]. The need to provide statistically valid estimates of classifier effectiveness is not, however, unique to e-discovery, but can be found in other areas where classifier results are subject to controversy, such as text classification in evidence-based medicine [4] and microarray classification in cancer diagnosis [24].

A classifier can be tested on data that is randomly sampled from the population of interest, labeled, and not used in training (a "held out" certification test set). Combined with an appropriate estimator, this approach produces effectiveness estimates that are valid and unbiased.[1] When classifiers are produced by supervised learning, a common approach is to create a succession of classifiers from training sets of increasing size, stopping this process when the effectiveness estimate on the test set reaches some desired level.[2] Unfortunately, using the certification test set to decide when to stop growing the training set introduces a sequential testing bias [30]. The reason is that effectiveness estimated on the test set fluctuates above and below true effectiveness over the sequence of classifiers, and the stopping rule is more likely to terminate training at an overestimate than at an underestimate.

We begin in Section 2 by reviewing past work on classifier evaluation and test set minimization. The contributions of the paper then are 1) a framework that avoids sequential bias in certifying classifiers, while allowing the practitioner to use arbitrary learning methods and termination policies (Section 3); 2) a method, based on ideas from statistical power analysis, for estimating the probability that a classi-

[1]Throughout this paper, we use *bias* in the statistical sense, to mean that the expected value of an estimator is not equal to the population value being estimated.
[2]Several products for e-discovery and data mining make this approach (too) easy by displaying a graph showing how effectiveness estimated on the test set is changing as additional training data is labeled.

fier will pass a certification test on an unseen test set of a specified size (Section 4); 3) experimental results on three baseline termination policies, exhibiting both the potential annotation savings possible and the challenges in developing good policies (Sections 5 and 6); and 4) a new analytic approximation for confidence intervals on van Rijsbergen's F-measure that was used in our experiments and is of independent interest (Appendix A). Section 7 summarizes our contributions and points to future work.

2. BACKGROUND

In this section we review techniques for estimating classifier effectiveness and choosing test set sizes. We also discuss methods for minimizing the use of labeled data in evaluating classifiers.

2.1 Estimating and Testing Effectiveness

In typical text classification evaluations, a classifier is built and applied to a test set. Its predictions on the test set are compared to known labels, and some measure of classifier effectiveness is computed. If the test set is a random sample from the population of documents to be classified, effectiveness on the test set is a *point estimate* of effectiveness on the population [18, Ch. VII].

Point estimates have uncertainty resulting from the finite random test set. We can make this uncertainty explicit by producing an estimate in the form of an interval. A *confidence interval* $[\theta_1, \theta_2]$ on a parameter x consists of lower and upper values for the parameter [18, Ch. VIII]. Any method for producing confidence intervals from a random sample has an associated *confidence level*, traditionally written $1 - \alpha$. The confidence level is the probability that, if many random samples of the same size were drawn in the same way from the population, and a confidence interval calculated from each, we would expect at least $1 - \alpha$ of those intervals to contain the true value for the parameter.

A practitioner often must do more than produce an estimate of effectiveness. They must decide whether there is sufficient evidence that the actual effectiveness, x, exceeds some minimum acceptable value, τ. Such evidence can be provided by a *one-tailed hypothesis test* [18, Ch. IX]. We pose the *null hypothesis* that $x = \tau$ (our classifier is poor) and the *alternative hypothesis* $x > \tau$ (our classifier is good enough).[3] The practitioner hopes to see evidence supporting the rejection of the null hypothesis.

The goals of both estimating effectiveness and becoming confident it exceeds a minimum value can be combined by estimating a *lower one-sided confidence interval* (LOSCI), i.e. a confidence interval whose upper limit is the logical maximum of the quantity being estimated (typically 1.0 for effectiveness measures) [15, 3]. Finding a $1-\alpha$ LOSCI $[\theta, 1.0]$ implies that for all $\tau < \theta$ a one-tailed hypothesis test would reject the null hypothesis $x = \tau$ at the α significance level.

2.2 Evaluating Learned Classifiers

Many text classifiers are themselves produced from labeled examples by applying supervised learning. Reusing the training data for testing such a classifier would produce an optimistic estimate of effectiveness, but using a *held out* (separate) sample makes unbiased estimates possible, given an appropriate estimator [27]. For a fixed labeling budget, one must then choose how to allocate labeled examples between training and testing.

A common policy is to specify a *fixed test set proportion*, with typical splits ranging from 50% training/50% test, to 80% training/20% test. An alternate policy used in e-discovery is *fixed test set size*, which allocates a specified number of labeled examples for testing, and whatever is left for training. In a third policy, *fixed training set size*, the classifier is first built, then any remaining documents are dedicated to testing. All these strategies are valid when used to choose the size of both training and test sets prior to using the test set. However, if examples are incrementally allocated to the training or test set based on test set effectiveness estimates, all three policies produce biased estimates of effectiveness [30].

The cost of held out test sets has led to many alternative approaches to evaluation. *Training set bounds* are based on analyzing the range of classifiers allowed by a particular learning algorithm, and how the algorithm searches that space [14]. Except when training sets are very small, however, even a small held out test set provides tighter confidence intervals than do training set bounds based on all labeled data [13, 14].

Cross-validation is a resampling method that partitions the labeled data into subsets, using each in turn as the test set for a classifier trained on the union of the remaining subsets. Intervals produced by cross-validation reflect a biased variance estimate and are not valid frequentist confidence intervals [1]. Other resampling methods, such as bootstrapping, jackknifing, repeated random sampling, and repeated independent design and test, similarly produce biased estimates of variance [6, 19, 31]. Further, these methods do not evaluate the actual classifier deployed (typically one trained on all labeled data). In addition, resampling methods make it difficult to compare classifiers produced by machine learning with those produced in other fashions (e.g. manual rule writing).

2.3 Power Analysis

A single application of a fixed size held out test set, while avoiding the biases discussed above, poses two challenges in our setting. First, the practitioner needs to be confident, without using the held out test set, that the classifier has the desired effectiveness. For this purpose, the bias of cross-validation and related methods (Section 2.2), when small, can be tolerated.

The second and deeper problem is that, to avoid sequential bias, the size of the certification test set must be fixed before it is used. Similar problems arise in designing surveys, clinical trials, and so on. How to determine test set size in these situations is the subject of *power analysis* in statistics [5]. The *power*, $1 - \beta$, of a statistical experiment is the probability that the null hypothesis will be rejected when it is in fact false. All things being equal, the larger the test set, the greater the power.

A critical notion in power analysis is the *effect size* [5]. In our context, the effect size is the amount by which the classifier's effectiveness exceeds the target effectiveness. If true effectiveness greatly exceeds the target, this can be demonstrated with a relatively small certification test set.

[3]Intuitively, the null hypothesis is $x \leq \tau$, but for test purposes the null hypothesis is the single point value most difficult to reject.

Conversely, if true effectiveness is barely above the target, a large certification test set will be needed to show this.[4]

Formulae for determining sample sizes given the effect size, desired significance level, and desired power are available for many standard statistics [5]. Unfortunately, van Rijsbergen's F-measure is not one of them, so in this paper we use computationally intensive simulation methods to find the sample size with a given power.

3. ANNOTATION MINIMIZATION

The research reviewed in the previous section presents a dilemma. Many strategies have been proposed to allocate training and test data in a way that reduces the total number of annotations. However, the strategies that produce valid, unbiased effectiveness estimates (training set bounds and their variants) do not save many annotations. Conversely, the strategies that substantially reduce annotations (cross-validation, sequential allocation) lead to biased estimates.

However, there is more room for maneuver here than it first appears. In Section 3.1 we present a new framework for supervised learning which allows for aggressive minimization of annotations while preserving unbiased classifier evaluation. We then introduce in Section 3.2 the notion of *policies* for the allocation of annotations in this framework, and how to evaluate such policies.

3.1 An Annotation Minimization Framework

We propose that classifiers requiring certification be developed as follows:

Step 1 Examples are selected, annotated, and used in training until the practitioner decides training is completed and a final classifier has been produced.

Step 2 The practitioner chooses the number of examples to be annotated for certification testing. A random sample of that size is drawn from the unannotated examples and labeled.[5]

Step 3 The classifier is applied to the certification test set, and a hypothesis test is made on the effectiveness of the classifier. The classifier either passes or fails this certification test.

The goal in our framework is to produce a classifier that has a specified probability of passing the certification test, while minimizing the total cost of annotating training and test data.

This framework is subtly but importantly different from common practice in e-discovery and related applications of supervised learning. First, the test set is used only once (Step 3), after the practitioner has committed to a final classifier. This frees the practitioner in Step 1 to use any techniques they like (manual classifier creation, online learning, active learning, resampling methods, sequential stopping, etc.) for training, for (non-official) effectiveness estimation, and for deciding when to end training. Any biases in these techniques may affect how often the classifier passes certification (Section 6), and may affect the annotation budget used, but cannot affect the validity of the certification test. From the standpoint of certification, classifier creation occurs within a black box, and actions taken "within the box" are irrelevant to certification.

Second, our framework postpones the choice of certification test set size until immediately before certification. This reflects a key insight of power analysis, which is that the optimal test set size depends on effect size. In the supervised learning setting, this translates to the observation that we cannot optimally choose test set size without having some estimate (even if biased) of the effectiveness of the classifier to be evaluated.

Third, our framework focuses on minimizing the joint cost of training and test data annotation. This perspective is unusual in e-discovery, and in most applications of machine learning. Yet it appears the right one for e-discovery at least, where both types of annotation may require an attorney billing at US $400-$800/hour or more.

3.2 Policies for Annotation Minimization

Our framework allows wide latitude for minimizing annotation, but does not itself tell a practitioner how to allocate annotations to training vs. testing. Further, in exchange for ensuring the certification test is valid, our framework exposes the practitioner to the risk that certification will fail, possibly requiring expensive remediation. We therefore propose that the central objects of study within our framework are *allocation policies* that provide guidance to the practitioner on allocating annotations. The notion of a policy is taken from reinforcement learning [11] where it refers to a rule that maps states to actions in a sequential decision setting while attempting to achieve or maximize some goal.

In our framework, the state of the system is the training data observed so far (and the classifiers trained from it). The actions available to the policy are to continue training or, alternatively, stop and choose a certification test set size. The two goals against which we evaluate a policy are:

1. *Achieving a power level specified by the practitioner.* Over a large number of trials, does the classifier pass certification the specified fraction of the time?

2. *Minimizing the cost of annotation.* How does the average annotation cost under this policy compare with that of other policies? If a budget is specified, what proportion of the time does the policy stay within budget while meeting other goals?

3.3 Design Principles for Policies

Two observations provide guidance for policy design. The first is the observation from power analysis that larger effect sizes allow smaller test set sizes. In our framework this means that the greater the amount by which the effectiveness of a classifier exceeds the certification target, the

[4]A common consideration in experiment design is what constitutes a *meaningful* effect size. As a practical matter, one may not care whether drug A is shown to be better than drug B unless drug A is quite a bit better. E-discovery is different, in that missing an agreed effectiveness target by any amount may require expensive remediation. In such a setting, effect size is of interest only indirectly, through its impact on sample size.

[5]If the population is logically infinite, then there is no distinction between drawing from the original population and drawing from the unannotated examples. Effectiveness estimates are conventional estimates of generalization to that infinite population. We assume a logically infinite population in this paper, but return to the case of a finite population in our discussion of future work.

smaller the certification test set that will be necessary to demonstrate that fact.

The second is the observation that learning curves (the plot of classifier effectiveness vs. amount of training data) climb steeply at first and then level off [22, 30]. Thus each new training example provides less of an increase in effectiveness than the previous one.

This leads to a natural criterion for terminating training. After the jth allocation of a group of training examples, the practitioner should estimate the effectiveness of a classifier trained on the r_j training examples seen so far. They should then do a power analysis to compute the size, s_j, of the test set that will, with desired power (probability of success) $1-\beta$, show that the classifier has met the target effectiveness. The sum $r_j + s_j = c_j$ would be the total budget for annotation if training were stopped at that time.

The leveling off of the learning curve means that while this total budget decreases at first, it inevitably reaches a minimum from which it starts to increase. If the policy is operating under a fixed budget, b, it might choose to stop training the first time that c_j falls under that budget, or might continue training in hopes of further declines in total budget. There are three possible outcomes that the policy may experience:

- The classifier passes the certification step using the certification test set of size s_j.
- The classifier fails the certification step using the certification test set of size s_j.
- The policy never attempts certification, because it never finds an allocation $r_j + s_j \leq b$ for which it estimates it has $1-\beta$ confidence of passing the certification step.

Whether policies should actually be allowed to terminate in a fixed budget setting without attempting certification is a design choice. In a practical setting one probably wants the policy to take its best shot.

Figure 1 shows an idealized view of how the total annotation budget necessary to meet an effectiveness target τ (with significance level $1-\alpha$ and power $1-\beta$) varies with the number of training examples labeled. Early in training the true classifier effectiveness is below the target value, so no amount of test data would let the classifier pass certification at the desired significance level.

Later (assuming the classification task is tractable given the selected threshold), the true classifier effectiveness hits the threshold effectiveness τ, but barely, so that a large test set, and thus a large combined annotation budget would be necessary. At this point each additional training example reduces the necessary test set size by more than one example. The minimum budget occurs after more training, when the classifier's actual effectiveness is well over the target value, and new training examples stop paying for themselves in test set size reduction. Eventually, a limit on classifier effectiveness may be reached, after which additional training data gives no benefit, and the cost c_i increases linearly with r_i.

3.4 Allocation Policies in Practice

Deciding when to stop training is more challenging than the idealized Figure 1 suggests. First, there is no standard formula to go from an effect size for F_1 to a test set size that achieves a given power. We therefore in Section 4.2 develop a simulation method for this purpose. Second, a policy does not get to observe the true effectiveness of any classifier produced during training. We propose in Section 4.4 the use of cross-validation to estimate that effectiveness.

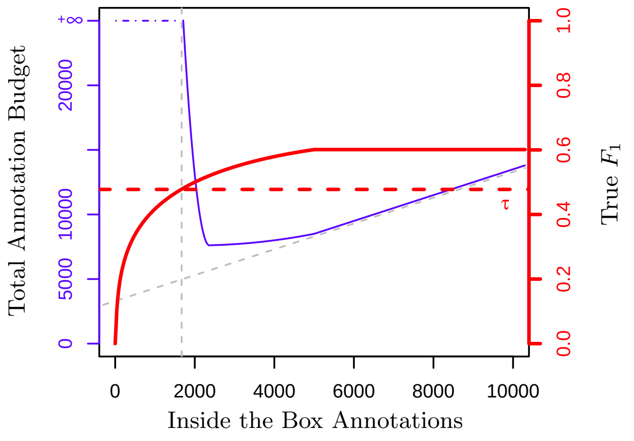

Figure 1: An idealized depiction of annotation budget minimization. The increasing curve plots the unknown true value of F_1 (right y-axis) against number of training examples (x-axis). The certification test can first be passed with high confidence when $F_1 > \tau$. Additional training data improves F_1 and allows smaller test set sizes, decreasing total annotation budget (left y-axis). Eventually effectiveness reaches a maximum and total annotation cost increases linearly with training set size.

Third, a policy does not have access to the entire curve of annotation budgets for different training set sizes. Instead, it observes only the portion of the curve for training set sizes produced so far, and must make a decision online [2]. Further, it does not observe a smooth budget curve based on monotonically increasing true effectiveness (Figure 1). Instead, it sees a scattering of budgets derived from noisy and possibly biased estimates of a true effectiveness that may itself not increase monotonically. After labeling each increment of training data, the policy must decide, based on that noisy history, whether it should take the budget it can achieve at this point, or press on in hopes of doing better. Finally, the very process of making such a decision is itself susceptible to a sequential testing bias.

Figure 2 illustrates the challenges a policy faces. We show a single run made on one RCV1-v2 topic (Section 5.1), increasing training sets by 20 at each iteration. The method described in Section 4 is used to estimate the required certification test set size after each iteration. Early on, estimated effectiveness is below the target effectiveness, so no certification test set size allows meeting the significance level and power requirements. Later, the estimated effectiveness does exceed the target, enabling the estimation of a test set size and thus a combined budget. To create Figure 2 we draw a single certification test set of the specified size, and check whether the classifier would have passed (green plus sign) or failed (red dot) the certification test with that test set.

In Section 6 we study some baseline policies that have partial success at choosing a stopping point. Note, however, that inadequacy in policies affects only the probability of passing certification and the annotation cost to do so. Our

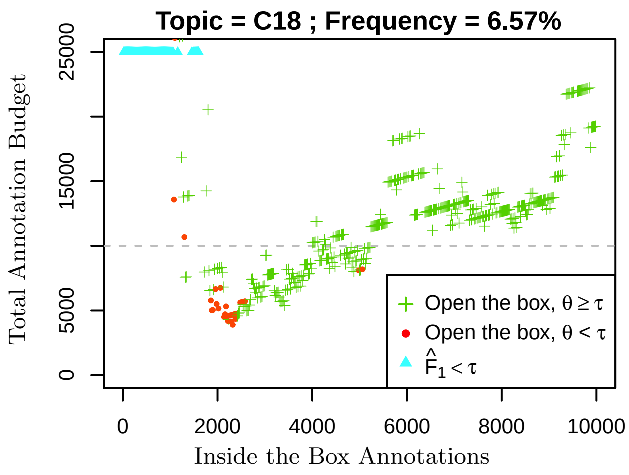

Figure 2: Annotations are added to the box with increments of 20 documents. A green plus sign (red dot) indicates a success (failure) in predicting that the threshold will be reached. Cyan triangles are situations in which the threshold is unachievable no matter what the certification budget is, since the point estimate on F_1 is itself below the threshold. (The striations in the data are the result of the computational shortcut of using nested sets in cross-validation.)

framework ensures that the *validity* of certification is unaffected.

4. INFERRING CERTIFICATION SET SIZE

Our framework requires a policy to make two decisions: when to stop training, and how big a certification test set to label. In this section we address the second of these decisions.

We assume throughout that we have two classes, relevant and nonrelevant, and a binary classifier with two outputs, predicted relevant (which we call "retrieved") or predicted nonrelevant ("unretrieved"). The inputs (Section 2) to the algorithm are the current training set, the effectiveness target τ, the confidence level $1-\alpha$, and the desired test power $1-\beta$. Our algorithm works as follows:

Step 1 For a randomly-sampled training set of r annotations, run n-fold cross-validation, yielding n classifiers and n corresponding confusion matrices of size r/n. Sum the confusion matrices to produce an overall confusion matrix $\mathcal{M}$ of size r (Section 4.4).

Step 2 Treat $\mathcal{M}$ as if it were a sample from an infinite population classified by a classifier trained on the entire training set. Derive a posterior distribution $P(\mathcal{P}|\mathcal{M})$ over infinite population confusion matrices for the classifier (Section 4.3).

Step 3 Call Algorithm 2 (Section 4.2) with inputs $P(\mathcal{P}|\mathcal{M})$, τ, $1-\alpha$, and $1-\beta$. Algorithm 2 does a binary search for the smallest certification test set size such that a classifier with behavior characterized by $P(\mathcal{P}|\mathcal{M})$ has a probability of at least $1-\beta$ of rejecting the null hypothesis $x=\tau$ in a one-tailed test with significance level α and alternative hypothesis $x>\tau$. Each proposed test set size s is checked as follows:

Algorithm 1 *EstimateThetaStar*

```
input: A distribution $P(\mathcal{P})$ over infinite population confusion matrices, test-set size $s$,
        confidence level $1-\alpha$, power $1-\beta$
output: $\theta^*$
1: initialize $N \leftarrow 1000$, $\Theta \leftarrow \{\}$
2: for $i \in [1, N]$ do
3:     Draw an infinite population confusion matrix $\mathcal{P}$ from distribution $P(\mathcal{P})$
4:     Simulate a random sample $\mathcal{S}$ of size $s$ from $\mathcal{P}$.
5:     $\Theta \leftarrow \Theta \cup \theta_\alpha(\mathcal{S})$
6: return $\theta^* \leftarrow Quantile(\Theta, \beta)$
```

Step 3.1 Call Algorithm 1 (Section 4.1) with inputs $P(\mathcal{P}|\mathcal{M})$, s, $1-\alpha$, and $1-\beta$. Algorithm 1 estimates θ^*, the minimum value of effectiveness which would fall inside a fraction $1-\beta$ of $1-\alpha$ LOSCIs $[\theta, 1]$ produced from samples of size s.

Step 3.2 If and only if $\theta^* \geq \tau$ then the test set size is at or above the minimum needed (Section 4.2).

Section 6 tests some baseline stopping policies which make use of this algorithm.

4.1 Inferring Highest Achievable Target from Classifier Behavior on a Population

Our first algorithm (Algorithm 1) takes as inputs $P(\mathcal{P})$ a distribution over infinite population confusion matrices, along with the definition of an effectiveness measure (F_1 in this study), a required significance level $1-\alpha$ and power $1-\beta$, and a test set size s. The algorithm estimates the minimum effectiveness, θ^*, that would fall inside at least a fraction $1-\beta$ of the $1-\alpha$ LOSCIs $[\theta, 1]$ produced from samples of size s from populations drawn from $P(\mathcal{P})$. This θ^* is equal to the maximum target effectiveness τ for which a one-tailed hypothesis test with null hypothesis $x=\tau$ (and alternative hypothesis $x>\tau$) would have power $1-\beta$ on a sample of size s.

We find θ^* by Monte Carlo simulation. Algorithm 1 simulates simple random samples from the infinite population confusion matrix $\mathcal{P}$ and calculates a $1-\alpha$ LOSCI $[\theta, 1]$ for each. It uses the β quantile of the θ values as θ^*.

4.2 Inferring Certification Test Set Size from Classifier Behavior on the Population

Algorithm 1 finds the maximum target effectiveness for which we can achieve a specified power, given a known certification test set size. What we actually wish to know, however, is the minimum certification test set size which yields a specified power, given a known target. Using Algorithm 1 as a subroutine, our Algorithm 2 solves the latter problem by a binary search of possible test set sizes.

Algorithm 2 takes the same inputs as Algorithm 1 except that a known test set size s is replaced by a known target effectiveness τ. Algorithm 2 first checks that $\hat{F}_1$, the point estimate on F_1, is greater than the target effectiveness τ. If not, no certification test set size can give the required confidence level and power.

If we do have $\hat{F}_1 > \tau$, we first find an upper bound by doubling candidate sizes until the first size u is found where

Algorithm 2 Estimate a Certification Test-Set Size

```
input: confusion matrix $\mathcal{M}$, threshold $\tau$,
       confidence level $1-\alpha$, power $1-\beta$
output: certification test-set size
1: if $F_1(\mathcal{M}) < \tau$ then
2:     return $+\infty$
3: else
4:     initialize $\epsilon \leftarrow 0.01 \times \tau$, Upper bound $u \leftarrow 1$
5:     while $EstimateThetaStar(\mathcal{M}, u, \alpha, \beta) < \tau$ do
6:         $u \leftarrow 2 \times u$
7:     for Size $s \in [u/2, u]$ selected with binary search do
8:         if $\tau \leq EstimThetaStar(\mathcal{M}, s, \alpha, \beta) \leq \tau + \epsilon$ then
9:             return $s$
```

Algorithm 1 returns θ^* such that $\theta^* > \tau$. This corresponds to a test set size that would give power greater than $1-\beta$.

We then do a binary search of test set sizes in the interval $[u/2, u]$. Algorithm 1 is invoked on each candidate test set size. The search stops when a certification test set size is found with θ^* such that $\tau + \epsilon \geq \theta^* \geq \tau$. The certification test set size at that point is returned.

There is a potential sequential sampling bias in this binary search. However, since Algorithm 1 does not directly use labeled data, this bias can be made arbitrarily small. We can simply increase the number of Monte Carlo simulations N in Algorithm 1, particularly for candidate test set sizes giving θ^* close to the target τ.

4.3 Inferring Certification Test Set Size from Classifier Behavior on a Held Out Sample

Algorithm 2 determines the certification test set size given a known distribution $P(\mathcal{P})$ over classifier behaviors on an infinite population of interest. In practice, what we typically have is a finite confusion matrix for the classifier on some sample from the population. To use Algorithm 2 we must use this sample to derive a distribution on infinite population confusion matrices.

We do this by deriving posterior beta distributions from Jeffreys priors upon the retrieved and unretrieved segments of the population as determined by the classifier [29]. The posterior distributions capture how likely various classifier behaviors are, given the behavior observed on the sample.

Consider the retrieved segment. Let the number of true positives in the sample be n_{11}, and the number of false positives be n_{10}. Then the beta posterior $R_\beta \sim \text{Beta}(0.5 + n_{11}, 0.5 + n_{10})$ is taken as a posterior upon the population proportion relevant in the retrieved segment. A similar beta posterior is derived on the unretrieved segment.

Finally, we assume that the proportion of the population retrieved by the classifier is known exactly.[6] The two beta posteriors, along with the proportion of the population retrieved, completely specify a posterior distribution $P(\mathcal{P})$ on infinite population confusion matrices.

We validate this method by randomly sampling $n = 100$ contingency tables empirically observed on our experimental data (Section 5). Treating these as infinite populations, we draw $m = 100$ samples from each, and estimate the certification set size required for thresholds of τ of $f = \{0.5, 0.6, 0.7, 0.8, 0.9\}$ of actual F_1 score (for $1-\alpha = 0.95$ and $1-\beta = 0.93$). Then, we observe the proportion of times that θ exceeds τ on a sample of this size drawn from the notional true population. We expect this proportion to be $1-\beta$ (subject to random variability, and preferring to be slightly above than slightly below). Table 1 shows the results of our validation; our inference method is accurate at predicting the sample size required to pass certification.

[6]In our experiments we assume this value is equal to the proportion retrieved in the sample. If a more precise value is needed, the classifier can be applied to all unlabeled data.

τ	0.5 F_1	0.6 F_1	0.7 F_1	0.8 F_1	0.9 F_1
Mean	93.53	93.25	93.92	94.26	94.29
SE	3.09	2.80	2.68	2.38	2.86

Table 1: Validation of Algorithm 2 with $1-\beta = 0.93$.

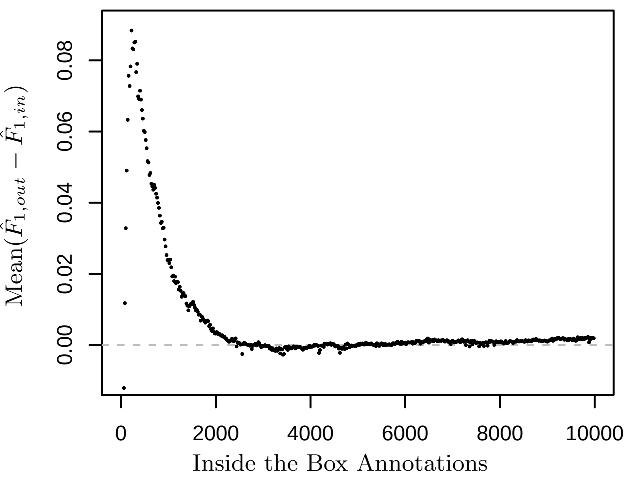

Figure 3: Performance predicted by cross-validation versus actual performance of classifier on a subset of the (remaining) population of a size estimated by Algorithm 2.

4.4 Certification Size from Cross-Validation

Section 4.3 provides a method for deriving a posterior distribution over classifier behaviors (infinite population confusion matrices) from a sample. A held-out sample would be ideal, but expensive. An attractive alternative is to employ n-fold cross-validation (Section 2.2), since this thriftily allows all annotations to be used both for training and for estimation.

Cross-validation produces a summary confusion matrix from across the n folds. Our approach is to use this as the sample required in Section 4.3. Since the cross-validated classifiers are trained on $(n-1)/n$ as much training data as the classifier whose effectiveness we wish to certify, cross-validation will generally understate true classifier effectiveness. Figure 3 plots the difference between a point estimate of classifier effectiveness as estimated by cross-validation, and true performance of the classifier as measured on a very large labeled set. For small training sets, the gap is substantial, with the maximum understatement of performance being around 0.08 for F_1. However, as the training set size increases, the understatement of performance declines quickly, until by around 3,000 training documents, cross-validation gives an unbiased average estimate of performance (although individual estimates may still be high or low).

5. EXPERIMENTAL METHODS

Our experiments use an SVM classifier trained and tested on subsets of the RCV1-v2 test collection, for which we wish to certify a desired level of effectiveness.

5.1 Test Collection

Our experimental data is the RCV1-v2 test collection of Reuters newswire stories [17]. The collection contains 804,414 documents, and their assignments to 103 topic categories. To aid replicability, we used the RCV1-v2 token files distributed as On-Line Appendix 12 [17]. The token files are converted to SVM^{perf} format using a modified version of the `prep_rcv1` program developed by Bottou.[7] The feature value of each word in a document is a TF x IDF weight [23]. We set the TF (term frequency) component to zero if a word does not appear in a document, and to one plus the natural log of the number of occurrences of the word in the document if it does. We set the IDF (inverse document frequency) component to the natural log of 804,414 divided by the number of documents the word occurs in. A total of 288,062 distinct stemmed words occur in the token files, so feature vectors nominally are of length 288,062, but with only an average of ≈ 77.5 nonzero values per vector.

Binary topic assignments are taken from the Appendix 8 transaction files [17]. These specify 2,606,875 assignments of topics to the 804,414 documents (a mean of 3.24 categories per document). We use only the 29 topics with 25,000 or more positive examples among the 804,414. This allows exploring a range of annotation budgets, while ensuring a reasonable number of positive examples will occur in random samples. The range of prevalences we use, from 3.1% to 47.4%, captures typical values encountered in e-discovery.

5.2 SVM Classifier

Since our focus is evaluation, we wished to use an existing, widely accepted classification approach. Support vector machines (SVMs) with linear kernels give good effectiveness across a range of text classification tasks [7, 32], so we take that approach to training a classifier. However, most SVM implementations attempt to produce a classifier minimizing error rate, rather than maximizing F_1. We therefore use the SVM^{perf} (Version 3.00) package [9, 8, 10], which can optimize for F_1 (among other effectiveness measures). For training, we run `svm_perf_learn` with flags "`-c 1000 -l 1 -w 3`", and for applying the classifier to a certification test set we use `svm_perf_classify` without flags.

5.3 Measures

We use F_1, the balanced harmonic mean of recall and precision [28], as our effectiveness measure for binary classifiers. The F_1 measure is defined as $(2 \times tp)/(2 \times tp + fp + fn)$, where tp is the number of true positives on the test set, fp is the number of false positives, and fn is the number of false negatives.

We use a confidence level of $1 - \alpha = 0.95$ for one-sided confidence intervals and so equivalently a significance level of $\alpha = 0.05$ for the corresponding one-tailed hypothesis test. We assume the user desires a test power of $1 - \beta = 0.93$ when choosing the certification test size.[8]

[7] `leon.bottou.org/_media/projects/sgd-2.0.tar.gz`

[8] A power level of 0.80 or lower is common in experimental science, but seems too low in an applied setting. A power of 0.95 seems natural, but we use 0.93 here to avoid the confusion of too many 0.95's in our exposition.

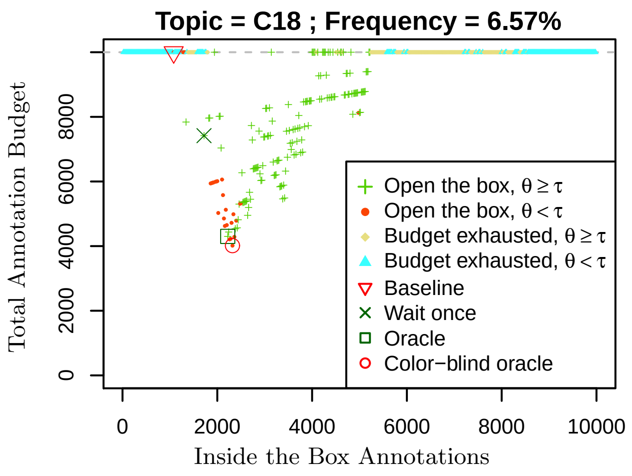

Figure 4: Policies that seek to minimize the total annotation budget. The baseline stops when the cost falls within budget. The oracles stop at the lowest possible annotation budget, with the "color-blind" oracle being subject to a possible failure (i.e. $\theta < \tau$). The "wait w" policy waits until observing w stopping points that are strictly below the total budget ($b = 10,000$ annotations).

In practice τ, the threshold on effectiveness, would be chosen based on the needs of a particular application. For experimental purposes we set τ for each topic to be 0.9 times the maximum value of F_1 observed on a single training run for that topic on training set sizes up to 10,000. This provides a target that is usually, but not always, reachable for arbitrary training sets of size 10,000.

We use the certification test set to produce an estimate of the classifier's F_1 in the form of a $1 - \alpha = 95\%$ lower one-sided confidence interval $[\theta, 1.0]$. To do this, we first produce a point estimate, $\hat{F}_1$, which is simply the value of F_1 on the certification test set. The value of θ is then found by assuming a normal distribution on $\hat{F}_1$, and analytically deriving $\text{Var}(\hat{F}_1)$ using the method of propagation of error (see Appendix A).

We define the requirement for passing the certification test on a particular run to be rejecting the null hypothesis that $F_1 = \tau$ (and thus implicitly rejecting $F_1 < \tau$ as well). This corresponds to τ for the topic falling within the interval $[\theta, 1.0]$ generated on that run.

More accurate, but computationally expensive, methods are known for producing confidence intervals on F_1 [30, 29]. Our goal here, however, is to study policies for passing certification tests. Certification tests in practice will often be computationally simple (perhaps using even less accurate estimates than we use here). In any case, we believe policy design will be relatively insensitive to test design, though this is an area for further study.

6. RESULTS

We study the behavior of three simple policies to illustrate the issues arising in designing an efficient and reliable stopping policy under our framework.

6.1 Stop When Cost Falls Within Budget

In the first policy, the developer has a fixed total annotation budget b. The policy stops and proceeds to certification when the total cost first falls within this budget; that is, when the training cost r plus the predicted required certification test set size s first sums to no more than the budget b. For this experiment, we set $b = 10{,}000$. In 29.48% of cases, this budget is never achieved; that is, there is no point for which $r + s \leq b$. (That is not necessarily to say that there is no way of splitting the budget between training and certification that would pass certification; but if there is one, our policy has failed to find it.)

Among the 70.52% of cases where the required certification test set size does fall within budget (and recall that we stop and proceed to certification as soon as it does), the success rate is 79.46%, versus our specified power of 93%. Why does the power not reach our specified level? The reason is sequential testing bias [30], now pushed into process management. Our cross-validation estimates of classifier effectiveness are scattered above and below the true effectiveness, due both to sampling variance and the fact that inside the policy we use cross-validation rather than directly evaluating a classifier trained from the entire training set. The first estimate that falls within budget is more likely to be an overestimate of classifier effectiveness than an underestimate.

6.2 Oracle Policies

The second policy is to stop at the lowest total cost projected by the inference method. In terms of Figure 4, this means picking the lowest dot in the figure, whether green (success) or red (failure). (We refer to this as a "blind" oracle because it ignores whether the certification itself is a failure or a success.) This (pseudo-)policy is not implementable in practice, as the classifier developer is not able to go backwards in time (retrospectively shrink the training allocation to the point that gives the lowest total budget). Instead this pseudo-policy sets a form of lower bound on achievable cost.

Compared to opening the box the first time the total cost falls below 10,000 annotations, and ignoring runs for which our algorithm fails to find a split of annotations less than this maximum budget (29.48% out of the 580 runs), the blind oracle policy saves on average 43.21% of the total annotation budget (training and certification included). However, the blind oracle fails the certification step 27.14% of the time, against the nominal failure rate of $\beta = 7\%$—a higher failure rate even than the 20.54% failure rate of stopping at the first within-budget estimate. Again, we are suffering from a sequential sampling bias, where by picking a minimum value, we are much more likely to pick an estimate that is randomly optimistic rather than randomly pessimistic.

If we instead choose the (even less realistic) sighted oracle policy, which always picks the lowest green dot (lowest total cost that passes the certification), then the saving over the stop-when-cost-falls-within-budget policy is on average 38.08% (ignoring the failed runs). In principle, the maximum achievable average saving falls somewhere in between the blind and sighted oracles, since our choice of β allows for a 7% failure rate. Naturally, the saving achievable for particular topics varies, depending on rate at which classifier effectiveness improves (and, of course, the nominal fixed budget against which we are comparing).

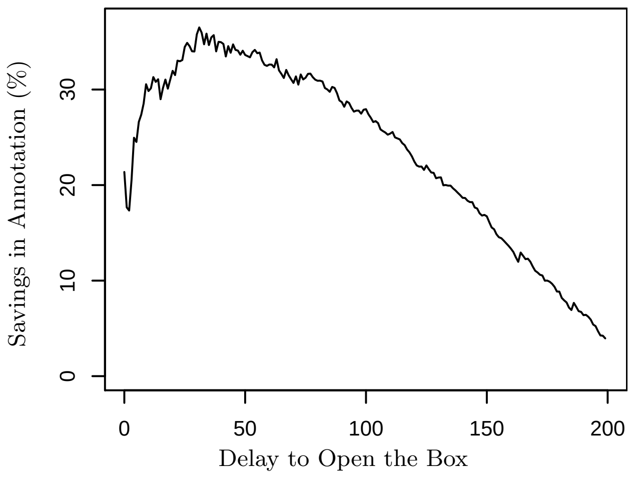

Figure 5: Savings in annotations if we defer opening the box w times, as opposed to opening at the first time the cost falls within budget.

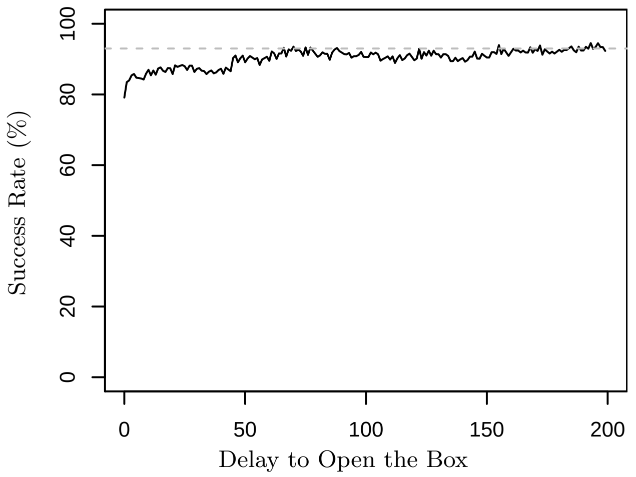

Figure 6: Success rate over 580 runs when the wait-a-while family of policies decide to stop.

6.3 Wait-a-While Policies

The third simple policy, or rather family of policies, builds upon the policy of stopping when a total budget is first achievable (Section 6.1). Instead of stopping immediately, however, this policy waits until a certain number w of further stopping points that are below total budget are observed, where w is a parameter chosen by the user. Note that this policy involves an additional risk—it may be that there are not w additional stopping points below budget, in which case the policy fails to proceed to certification at all.

Figure 5 shows the average saving in annotations from adopting the wait-a-while policy, compared to stopping the first time the total cost is at or under the budget of 10,000.[9] On average, it is optimal to wait for 25 to 50 iteration cycles (of 20 documents added to training for each iteration), though again there will be considerable variability between

[9]We limit our training sets to 10,000 for computational reasons; we therefore do not treat cases that exceed this as failures.

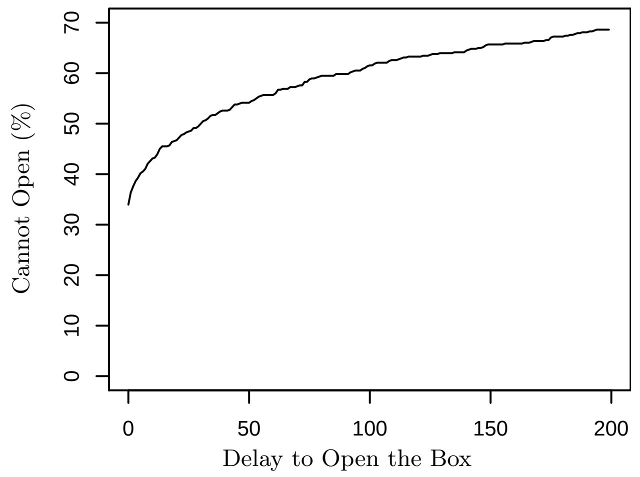

Figure 7: Failure to open the box over 580 runs because the threshold τ is not reachable, or because the policy has missed the last training set for which the box could have been opened.

different topics. Figure 6 shows the success rate of the wait-a-while policy. As we move further from the first time we hit the budget, the sequential sampling bias fades, until eventually we are more or less hitting nominal success rate of 93%. However, these achievements come at a cost, as Figure 7 shows: the longer we wait, the more likely we are to have missed the chance of achieving our desired budget altogether. (In fact, the increased savings with additional waiting observed in Figure 5 are likely exaggerated by the exclusion of the more difficult runs and topics by this censoring effect.)

7. CONCLUSION AND FUTURE WORK

Failure to achieve required classifier effectiveness is usually handled by training the learner on additional annotations and then re-testing the resulting model. This approach unfortunately introduces statistical bias. In this paper we proposed an algorithm that, for a confidence level, indicates if and when a classifier built on a growing training set can achieve a threshold, and how many testing documents will be required to certify the trained model. This method can be integrated as a subroutine for policies that aim to solve a more complex problem—minimizing the total annotation budget spread over training and certification sets. We provided a framework in which such policies can be studied and evaluated. Our experiments suggest that there is a room for saving about 40% in annotation budget without compromising the success rate of reaching the targeted effectiveness.

Our study examined only simple random sampling for producing both training and test sets. Simple random sampling is widely used in e-discovery (where its simplicity is reassuring in an adversarial context), and in other applications of supervised learning. However, other approaches, in particular active learning for training [16] and stratified random sampling [26] for testing, are of interest for their potential to further reduce the annotation budget.

We have also confined ourselves here to the traditional generalization framework for studying classifier effectiveness. In e-discovery and some other applied settings, we are actually dealing with a finite population of examples, so that each example annotated is an example to which the classifier need not be applied. This finite population annotation framework provides additional opportunities for minimizing annotations.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. 1065250. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

8. REFERENCES

[1] Y. Bengio and Y. Grandvalet. No unbiased estimator of the variance of k-fold cross-validation. *Journal of Machine Learning Research*, 5:1089–1105, September 2004.

[2] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambirdge University Press, 2005.

[3] T. T. Cai. One-sided confidence intervals in discrete distributions. *Journal of Statistical Planning and Inference*, 131:63–88, 2005.

[4] A. M. Cohen, W. R. Hersh, K. Peterson, and P.-Y. Yen. Reducing workload in systematic review preparation using automated citation classification. *Journal of the American Medical Informatics Association*, 13:206–219, 2006.

[5] J. Cohen. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates, 2nd edition, 1988.

[6] A. Isaksson, M. Wallman, H. Göransson, and M. G. Gustafsson. Cross-validation and bootstrapping are unreliable in small sample classification. *Pattern Recognition Letters*, 29:1960–1965, 2008.

[7] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *ECML*, pages 137–142, 1998.

[8] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, pages 377–384, 2005.

[9] T. Joachims. Training linear svms in linear time. In *KDD*, pages 217–226, 2006.

[10] T. Joachims and C.-N. J. Yu. Sparse kernel svms via cutting-plane training. In *ECML PKDD: Part I*, 2009.

[11] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

[12] A. Kershaw and J. Howie. eDiscovery institute survey on predictive coding. Technical report, Electronic Discovery Institute (http://www.ediscoveryinstitute.org/pubs/ PredictiveCodingSurvey.pdf), October 2010.

[13] J. Langford. Combining train set and test set bounds. In *ICML*, pages 331–338, 2002.

[14] J. Langford. Tutorial on practical prediction theory for classification. *Journal of Machine Learning Research*, 6(1):273–306, 2005.

[15] E. L. Lehmann and J. P. Romano. *Testing Statistical Hypotheses*. Springer, 3rd edition, 2005.

[16] D. D. Lewis and W. A. Gale. A sequential algorithm for training text classifiers. In *SIGIR*, pages 3–12, 1994.

[17] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, December 2004.

[18] A. M. Mood, F. A. Graybill, and D. C. Boes. *Introduction to the Theory of Statistics*. McGraw-Hill, 3rd edition, 1974.

[19] C. Nadeau and Y. Bengio. Inference for the generalization error. *Machine Learning*, 52:239–281, 2003.

[20] D. W. Oard, J. R. Baron, B. Hedin, D. D. Lewis, and S. Tomlinson. Evaluation of information retrieval for e-discovery. *Artificial Intelligence and Law*, 18(4):347–386, 2010.

[21] N. M. Pace and L. Zakaras. Where the money goes: Understanding litigant expenditures for producing electronic discovery. Technical report, RAND Institute for Civil Justice, Santa Monica, CA, 2012.

[22] F. Provost, D. Jensen, and T. Oates. Efficient progressive sampling. In *KDD*, pages 23–32, 1999.

[23] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523, 1988.

[24] R. Simon, M. D. Radmacher, K. Dobbin, and L. M. McShane. Pitfalls in the use of DNA microarray data for diagnostic and prognostic classification. *Journal of the National Cancer Institute*, 95(1):14–18, January 2003.

[25] J. R. Taylor. *Introduction to error analysis.* University Science Books, 2nd edition, 1997.

[26] S. K. Thompson. *Sampling.* Wiley, 2nd edition, 2002.

[27] G. T. Toussaint. Bibliography on estimation of misclassification. *IEEE Transactions on Information Theory*, IT-20(4):472–479, July 1974.

[28] C. J. van Rijsbergen. *Information Retrieval.* Butterworth, 1979.

[29] W. Webber. Approximate recall confidence intervals. *ACM Transactions on Information Systems*, 31(1):2:1–33, 2013.

[30] W. Webber, M. Bagdouri, D. D. Lewis, and D. W. Oard. Sequential testing in classifier evaluation yields biased estimates of effectiveness. In *SIGIR*, pages 933–936, July 2013.

[31] U. Wickenberg-Bolin, H. Göransson, M. Fryknäs, M. G. Gustafsson, and A. Isaksson. Improved variance estimation of classification performance via reduction of bias caused by small sample size. *BMC Bioinformatics*, 2006.

[32] Y. Yang and X. Liu. A re-examination of text categorization methods. In *SIGIR*, pages 42–49, 1999.

APPENDIX

A. A NORMAL-APPROXIMATION CONFIDENCE INTERVAL ON F_1

The contingency table in Figure 8 describes the population values of the classifier. Then F_1 is:

$$F_1 = \frac{2 \cdot R_1}{R_1 + R_0 + N_1}$$

We draw a simple random sample of size n_1 from the segment of classified relevant documents, and observe r_1 of them to be relevant. Our estimate of R_1 is therefore:

$$\hat{R}_1 = N_1 \cdot \frac{r_1}{n_1}$$

and similarly for $\hat{R}_0$. Hence, our estimate for F_1 is:

$$\hat{F}_1 = \frac{2\hat{R}_1}{\hat{R}_1 + \hat{R}_0 + N_1}$$

What we now require is an expression for $\widehat{\mathrm{Var}}(\hat{F}_1)$. If:

$$X = f(A, B)$$

then the theory of propagation of error states [25]:

$$\mathrm{Var}(X) = \left(\frac{\partial f}{\partial A}\sigma_A\right)^2 + \left(\frac{\partial f}{\partial B}\sigma_B\right)^2 + 2\frac{\partial f}{\partial A}\frac{\partial f}{\partial B}\mathrm{Cov}_{AB} \ . \quad (1)$$

Let $X = \hat{F}_1$, $A = \widehat{R}_1$, and $B = \widehat{R}_0$. Because the classified-relevant and classified-irrelevant strata are independently sampled, $\mathrm{Cov}_{AB} = 0$. It can be shown that:

$$\frac{\partial f}{\partial A} = \frac{2}{A + B + N} - \frac{2A}{(A + B + N)^2} \ , \quad (2)$$

and that:

$$\frac{\partial f}{\partial B} = -\frac{2A}{(A + B + N)^2} \ . \quad (3)$$

Plugging Equation (2) and Equation (3) into Equation (1), we find:

$$\widehat{\mathrm{Var}}(\hat{F}_1) = \frac{4}{\left(\widehat{R}_1 + \widehat{R}_0 + N_1\right)^4} \cdot \left[\left(\widehat{R}_0 + N_1\right)^2 \widehat{\mathrm{Var}}(\widehat{R}_1) + \widehat{R}_1^2 \,\widehat{\mathrm{Var}}(\widehat{R}_0)\right] \ , \quad (4)$$

where:

$$\widehat{\mathrm{Var}}(\widehat{R}_1) = N_1^2 \cdot \frac{r_1}{n_1^2} \cdot \left(1 - \frac{r_1}{n_1}\right)$$

(ignoring the finite-population adjustment of $(1 - n_1/N_1)$), and similarly for $\widehat{\mathrm{Var}}(\widehat{R}_0)$. Equation 4 gives our expression for the sampling variance of F_1. If we assume that sample F_1 score is approximately normally distributed with equal variance, then a $1 - \alpha$ two-sided confidence interval on F_1 is:

$$\mathrm{CI}_{\mathrm{norm}}(F_1) = \widehat{F_1} \pm z_{1-(1-\alpha)/2}\sqrt{\widehat{\mathrm{Var}}(\hat{F}_1)/(n-1)} \quad (5)$$

where z_a is the a'th quantile of the cumulative normal distribution (for instance, $z_0.975 \approx 1.96$ for a 95% confidence interval), and n is the sample size.

		R	
		1	0
C	1	R_1	$N_1 - R_1$
	0	R_0	$N_0 - R_0$

Figure 8: Contingency table of true and false positives and negatives, defined by intersection of the sets classified as relevant (C) and actually relevant (R).