A Large-Scale Active Learning System for Topical Categorization on the Web

Suju Rajan
Yahoo! Labs
Sunnyvale, CA, USA
suju@yahoo-inc.com

Dragomir Yankov
Yahoo! Labs
Sunnyvale, CA, USA
yankov@yahoo-inc.com

Scott J. Gaffney
Yahoo! Labs
Sunnyvale, CA, USA
gaffney@yahoo-inc.com

Adwait Ratnaparkhi
Yahoo! Labs
Sunnyvale, CA, USA
adwaitr@yahoo-inc.com

ABSTRACT

Many web applications such as ad matching systems, vertical search engines, and page categorization systems require the identification of a particular type or class of pages on the Web. The sheer number and diversity of the pages on the Web, however, makes the problem of obtaining a good sample of the class of interest hard. In this paper, we describe a successfully deployed end-to-end system that starts from a biased training sample and makes use of several state-of-the-art machine learning algorithms working in tandem, including a powerful active learning component, in order to achieve a good classification system. The system is evaluated on traffic from a real-world ad-matching platform and is shown to achieve high categorization effectiveness with a significant reduction in editorial effort and labeling time.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology

General Terms

Design

Keywords

Classification with Imbalanced datasets, SVMs, Active Learning, Web Scale Performance Evaluation

1. INTRODUCTION

In recent years, the problem of categorizing web pages has become particularly relevant. The wide-spread prevalence of ad matching systems, the rise of vertical specific search engines and information extraction systems require the automated detection of pages of a particular class or with specific semantics. For instance, ad matching systems require that ads not be shown on specific classes of pages such as adult pages or pages with mature content. Vertical search engines, such as a blog search engines, operate only on a particular type of pages such as blog, or review, or forum pages. Similarly, information extraction systems use different extraction mechanisms based on the type of a page. Thus, a restaurant home page will have extraction rules that are different from that of a University's home page. It is important to note that for any topic or semantic concept, such as the ones mentioned above, the number of Web documents that belong to this concept is usually very small if compared to all available documents in a population[1].

The rarity of pages from a particular topical class on the Web poses a number of challenges in each stage of learning a classifier for that topic: starting from training data generation, through model training, to evaluating the expected performance on the population. For example, the standard technique of generating training data by obtaining a random sample of the population does not work in this case as the data thus generated will have very few or no examples at all from the class of interest. Training a model, on the other hand, usually requires that both the positive and the negative class be well represented. Compiling a representative negative set of examples, however, is infeasible as a negative example can be any page on the Web on any topic different from the topic of interest.

The problem of learning from imbalanced datasets has applicability in many domains, such as fraud detection [6], spam detection [26], or object recognition in images [28, 29] (cf. Section 2). Most of the solutions proposed in these areas involve the stratification of data based on some domain-specific criterion, resampling the rare class data in each stratification, followed by training a classifier on each stratification. While the above mentioned approach has been shown to work with varying degrees of success in an offline setting, the technique is not easily adaptable to the Web because of its scale. Firstly, it is not clear what the stratification should be and secondly, even if some stratification was obtained the chances of finding pages from the class of interest in each stratification continues to be small.

In this paper, we describe an end-to-end topical categorization system that has been successfully deployed in a real time ad matching platform on the Web. The system is designed to detect pages on a number of "sensitive" topics, such as pages with adult, gambling, hard news (i.e. news about death and suffering), or drug related content (cf. Section 5).

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

[1]By "population" we refer, for example, to all pages on which online ads might be displayed; or to all pages that a search engine indexes.

Reputable advertisers do not want their ads to be associated with pages of such nature. To avoid lawsuits and protect the reputation of the advertisers we have designed a methodology to automatically detect sensitive pages among the advertising traffic. The primary requirement of our system is for it to have very high recall, i.e. retrieve as many of the sensitive pages as possible, with *reasonable* precision. While we do not quantize what reasonable means, we point out that having too many false positives, i.e. normal pages being labeled as sensitive, decreases the advertising inventory and has a direct impact on the profits. These two competing objectives are especially hard to achieve because some categories may have very subtle nuances in their definition. As an anecdotal example, certain advertisers allow their ads to be shown on a page discussing over-the-counter drugs, but not on a page discussing illegal drugs.

Our proposed system, consists of a number of binary categorizers, one for each category of interest. Through the rest of the text, we describe the methodology devised for building these categorizers. We first elaborate on the problem of imbalanced class learning and some solutions discussed in the related literature (Section 2). We then propose a scalable procedure for obtaining concept rich training data for topical categorization on the Web (Section 4.1). We describe our model training with an emphasis on active learning mechanisms that quickly overcome the bias introduced during the training data generation phase (Section 4.2 and 4.3). We finish by presenting the steps followed in evaluating the performance of our binary categorizers on the entire population and the subsequent setting of a classification threshold for deployment (Section 4.5).

2. RELATED WORK

The importance of robust classification in the case of imbalanced datasets has been stressed by researchers across diverse fields [6, 7, 17, 23, 26, 28, 29]. For example, Yan et al. [29] demonstrate that in scene classification, discriminative classes of objects are usually very rare. Chan et al. [6] and Tang et al. [26] point out respectively that fraudulent credit card transactions and non-spam mail sending IPs constitute a very small number of the entire set of transactions or mail populations. Chawla et al. and Woods et al. [7, 28] show that the very few cancer indicating pixels on mammograms can be easily overlooked when classifying medical pathologies. Rare class detection has also been studied in the context of document categorization-the focus of our work. Joshi et al. [15], for instance, analyze the effect of different base learners while using boosting for the topical categorization of documents from a medical corpus, while Mladenic et al. [21] discuss how feature selection and different scoring methods might impact Web document categorization in the imbalanced setting.

Provost [23] summarizes two fundamental questions that should be addressed when dealing with highly imbalanced classes: 1) What is the ultimate cost function to be optimized? 2) How can differences in training and testing distributions be accommodated? The first question stresses an essential artifact emerging in the context of imbalanced class learning. Optimizing the usual 0-1 classification loss and computing the accuracy of the classifier may be rather misleading in such settings. Indeed, the trivial classifier that assigns the majority label will often have near perfect accuracy yet its precision for most problems like cancer detection will be unacceptable. A majority of the literature on unbalanced class learning focuses on aspects and solutions of this problem. The general approaches suggested by researchers in the area are either to re-define the cost function to be more sensitive to errors in misclassifying the rare class [6, 10, 16], or to artificially change the distribution of the data by upsampling the minority or downsampling the majority class and thus simulate a learning problem with a more balanced class distribution [7, 17]. Both directions focus ultimately around the same idea, which is how to force the learner to recognize that the rare class has higher importance than what is suggested by the training distribution.

As noted by Provost [23], cost-sensitive corrections and sampling based methods have been demonstrated to achieve promising results when applied in certain domains but fail to perform consistently when applied across a variety of imbalanced problems. Furthermore, they fail to address the second question stated above, namely, how shall we proceed when the true distribution is not only skewed but also infeasible to sample from, i.e. when our train and test distributions differ? This turned out to be especially critical in our system for Web document categorization because of the following reasons. On one hand, the number of pages from different topics is dynamic - a devastating phenomena or a death of a celebrity may increase the number of "hard news" pages dramatically over a day and change the "hard news" distribution significantly. On the other hand, even for a particular snapshot in time there is no clear understanding of the actual distribution of topics on the Web - how many drug related pages are there on the Web? A very negligible fraction probably compared to all pages. To answer such questions through standard sampling techniques, we may need to obtain over a million documents and have them labeled by experts which is apparently infeasible. Later on (Section 4.1) we introduce our approach for generating a rich and representative initial "training seed". The method is close in spirit to *importance sampling* and uses search queries as a proxy to capture the distribution of the positive class. While this allows the generation of a training set that reflects the importance of the rare class, the procedure also introduces a lot of bias in the trained classifiers, which again brings us to the the second question raised above.

There are very few guarantees that can be given about the performance of a classifier if nothing is known about the true test distribution in advance. *Covariate shift* [25], for example, shows a method for correcting training bias given that there is information about the conditional distribution of the test set labels y given data x. In particular, the method requires that $p(y|x)$ remains unchanged on the train and test sets. Model selection under covariate shift has been demonstrated on imbalanced learning problems too, such as spam filtering [2], where spam patterns may change over time, or cancer detection in gene expressions [3] where test and train data might come from different study groups. Considering covariate shift, however, we found that the assumption for $p(y|x)$ to remain constant is too strong for document categorization on the Web. For certain x, e.g. pages discussing a drug like "propofol", we may not even have a training estimate of $p(y|x)$ because at the time of training set collection there were no high ranked documents in the population discussing the drug. Even if we have a training estimate for $p(y|x)$, its value may change significantly in test time. To illustrate this, consider a situation where the training data

contains several slightly different pages x discussing hurricanes, some of them related to a hurricane fiction movie and others from a weather forecasting service. If, however, a deadly hurricane phenomena occurs at some point in time, then $p(y = \text{“}hard\_news\text{”}|x)$ in our test environment will increase with pages from all popular news sites suddenly discussing the devastating effects of the phenomena.

Instead of trying to infer the covariate shift in the test data, in our system we decided to adopt a more data driven approach, one in which we try to make our classifier less susceptible to sudden changes in distribution. We achieve this by tuning the classifiers to the entire population. For this purpose, we resolve to a hitherto unprecedented large scale active learning procedure. In active learning the classifiers iteratively augment their training set with a limited number of examples from the entire population. The method has been studied broadly in the academic literature, including in the context of document categorization and imbalanced datasets [11, 20, 27], but has remained unpopular in the data mining industry mostly due to its heavy requirement for expert editorial resources. We defer our discussion on active learning to Section 4.3 where we outline a number of previously overlooked aspects that emerge when trying to scale the method to a production system with inherently noisy and hardly separable data.

3. AD MATCHING PLATFORM

Contextual advertising is a form of advertising on the Internet, wherein an ad matching system acts as an intermediary between an advertiser and the publisher of Web pages. Whenever a user views a page that is being served by the ad matching platform, the system at real-time selects ads from a corpus to display on a publisher's page. Typically, an ad is matched to a page either based on the content of the page, or the profile of the user viewing the page. There has been a lot of recent work in making this ad matching process effective and efficient [1].

As mentioned in Section 1, advertisers have a very strong notion about the type of pages they would not want their ads to appear on. For instance, most advertisers would not like their ads placed on pages that contain mature adult content, or on pages that deal with death and suffering, as they do not want to be associated with content that is distasteful to their customer profile. The problem of identifying the set of pages that are of a certain nature has very strong precision and recall considerations. The precision metric ensures that pages that are being identified as being of a certain type should truly be so, otherwise the ad matching system loses out on an advertising opportunity and this directly impacts profits. The recall metric, on the other hand, guarantees that pages that belong to a black-listed category never go undetected thereby making sure that the advertiser guidelines are consistently met.

A large-scale ad matching platform processes a few tens of millions of pages at any point of time thereby making manual filtering of pages in black-listed categories impossible. An obvious approach is to use a dictionary based string matching method, wherein a human editor generates a list of indicative terms, the presence of which on a page signals the possibility of the page being in the undesirable category. Such methods while having a high precision suffer from low recall. Hence, there is a need to categorize the pages that are served by the ad matching platform automatically with a more intelligent method than string matching. Given the nature of the ad serving platform, a candidate categorization system should be scalable, be able to classify a page within a hundred milliseconds and be relatively stable across changes in the training and test distributions. In the following section, we address the design of such a categorization system.

4. DESIGN OF THE CATEGORIZATION SYSTEM

As with any active learned categorizer, our categorization system implements the following steps:

- Generation of training data.
- Initial model training.
- Validation of trained model.
- Model adjustment based on the results from the validation step. Iteration with previous step until convergence.
- Performance evaluation on real traffic.

In the following sections, we describe the design methodology and the intuition behind the choices for each of the above steps. While model adjustment typically implies parameter selection, we also include the active learning phase in it as the learned model has to adjust to the differences between the training and test distributions.

4.1 Generating Training Data

Once a category of interest is identified, the first step in training a classifier is to generate labeled training data for it. Obtaining a uniform sample of pages from the Web and having it labeled by human editors is not feasible as such a sample, even with a few 100K examples, might not contain any pages from a rare class of interest. Thus, a very large sized sample has to be collected in order to have a reasonable number of examples from the rare class for classifier training. Such a naive approach is a waste of valuable human editorial time. Optionally, one can rely on an editor's knowledge of the Web to obtain good positive and negative training examples for the classifier. This approach introduces some bias in the training data as there will be a number of less popular domains the editors maybe unaware of which are important to learn from for the sake of generalization. In order to obtain a snapshot of the pages in the category of interest on the Web, we used the search engine as a front-end processor. Human editors first draw up a list of search terms that can potentially retrieve the pages of interest. For instance, given a reasonably sized index of the Web, searching for the term *online gambling casinos* will surface a variety of pages that offer online gambling; a category that is offensive to some advertisers. Once the list of query terms has been prepared, the search engine is queried with the same. A smaller sample of the URLs retrieved by the search engine is obtained and added to the editorial queue. The editors then judge the actual pages associated with the set of URLs and mark them up as positive or negative, where positive indicates that the page is about the rare class of interest and negative indicates that the page retrieved by the search engine did not qualify for the category as per pre-defined guidelines. This methodology provides us with a biased training data in which the

negative examples are borderline positive, given that they were retrieved by the search engine along with the positive examples, and are potentially harder to learn from.

In order to obtain better representation of the space of negative examples, human editors also made use of web directories that are manually maintained. Examples of such directories are the Yahoo! Directory [2] and the DMOZ Open Directory Project [3]. Both these directories contain a large number of nodes that are arranged as a taxonomy. Each node represents a set of curated topical pages with the corresponding URLs. For example, the *Arts/Photography* node in the Yahoo! directory contains URLs of pages that are related to the art of photography. For our purposes, nodes in the Yahoo! directory whose semantic names are widely different from the semantics of the category of interest are first identified. URLs in such negative nodes are then sampled and added to the negative set obtained by search engine querying. Note that this wholesale addition of negative examples typically works when the category of interest is topical, such as the *Adult* or *Gambling* categories. If the class of interest cuts across topical pages, as with a generic *Blog* classifier, the identification of representative negative data is less straight-forward. In the latter case, a classifier is first trained on the initial editorial set. The classifier is then used to score about 1M pages sampled at random from the search index. Pages that are labeled as strongly negative by the classifier are then added as negative examples to the training set. In either case, approximately 100K such *background* negative pages are added to the training set.

4.2 Training an Initial Model

The training data collected via the process detailed in Section 4.1 was processed to obtain a bag-of-words representation over the unigrams extracted from the page [14]. Since our training data contains only Web pages with HTML markup, information about the HTML tags was also preserved in the feature construction process. For example, *adult_t* (the word *adult* appearing in the title) is treated as a feature different from the feature *adult_b* (the word *adult* appearing in the body of the document). Preserving HTML tag information proved to be a useful signal in our initial experimentation. For example, in the case of the *adult* classifier a page that contains strongly adult words in the title is more likely to be a page with adult content than a page that contains such terms only in the body. Experimentation with bigram features resulted in almost no gains for the large increase in the size of the feature space. We also made use of a stopwords list (with words like 'and', 'for', etc.) to remove the high frequency non-descriptive words. A 'global' dictionary was constructed as explained later in Section 4.3 and only those features of the document that appear in the dictionary were retained. The resulting bag-of-words was further processed such that each page was represented as $x_i = (\frac{x_{i0}}{\|x_i\|}, \frac{x_{i1}}{\|x_i\|}, \ldots, \frac{x_{in}}{\|x_i\|})$, where x_{ij} are the frequencies with which the j-th word in the dictionary appears in the i-th example. Experimental evidence showed that the above representation of examples as directions on the unit sphere yielded significantly better performance as opposed to no normalization and a similar performance to feature-wise standardization. Feature-wise standardization techniques, however, produce a non-sparse representation which is a significant concern as the feature space that we work with contains hundreds of thousands of dimensions. Standardization is also prone to instability due to a possible bias in the estimate of the two modes (mean and deviation) only on the training set.

[2]http://dir.yahoo.com
[3]www.dmoz.org

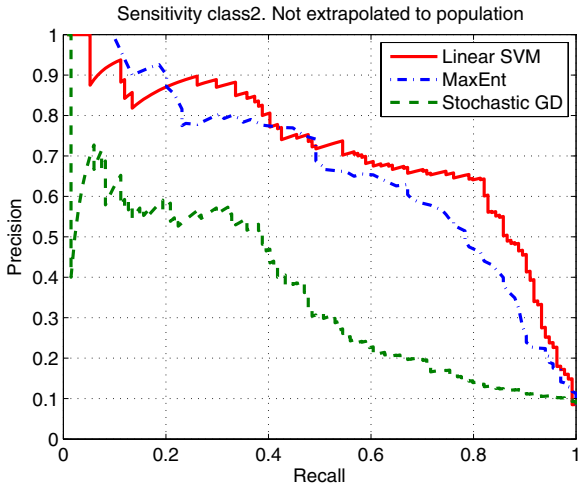


Figure 1: Comparison of linear SVM, MaxEnt and stochastic gradient descent on a five fold split of the *Sensitivity Class 2* training set (cf. Section 5).

A number of learning methods, including linear and RBF kernel SVMs [5], MaxEnt models [22], decision trees (C5.0) [24], Naive Bayes [19], and stochastic gradient descent [18] were evaluated on a five fold cross-validation split of the training data. The decision trees and the RBF kernel SVMs could not handle the large training data and feature space. The Naive Bayes classifier was significantly outperformed by the linear SVMs as is generally observed [14]. While recent linear SVM implementations allow the optimization method to scale easily to millions of documents represented over millions of features [12, 13], this classifier also performed better than MaxEnt and stochastic gradient descent as can be seen for one of the categories in Figure 1.

The optimization problem that SVM solves has the form $\min_w \frac{1}{2}\|w\| + C(c_1 \sum_{i=1}^{p} \xi_i^1 + c_0 \sum_{i=p+1}^{l} \xi_i^0)$, where $\frac{w}{\|w\|}$ is the normal of the separation hyperplane defining the classification function; ξ_i^1 and ξ_i^0 is the L_2 loss incurred in misclassifying a positive or negative example respectively; $0 \leq c_1 \leq 1$ is the weight associated with the penalty paid by the positive examples; $c_0 = 1 - c_1$ is the weight associated with the penalty paid by the negative examples; and C is the trade off between the complexity of the classifier and its performance on the training set. The best parameters for a SVM classifier were obtained by subjecting the training data to a five fold cross-validation search on the grid of values $C = \{2^{-5}, 2^{-4}, \ldots, 2^8\}$ and $c_1 = \{0.2, 0.3, \ldots, 0.8\}$, to identify the parameters (C^*, c_1^*) that optimize the precision at recall level 90%. This best set of parameters, as identified by the grid search, was then used with the entire training data to build a classifier.

4.3 Active Learning

An initial evaluation of the classifier, built as detailed in

Section 4.2, using the results of the 5-fold cross-validation showed a good precision/recall curve. However, when the classifier was evaluated on a hold-out test set that was drawn from the traffic, the classifiers did not generalize well as can be seen in Figures 3(a) and 3(c). Despite the use of a search-engine as a proxy to the Web and the addition of a large number of negative examples, the training data continued to have some bias that caused it to not generalize well to the test distribution. One source of bias came from the Web domain bias, that is, editors while labeling became attuned to the fact that pages from a particular domain typically contained adult content and would consistently label pages from the domain in question to the positive dataset. What was missing in this picture was that the adult content was just one of the sub-domains and there were other parts of the site which had non-adult content. Unless and until we had negative data from such non-adult sub-domains in our training set the classifier would invariably assign a very high weight to the domain name itself. At the time of ad matching, given a non-adult page from this site, even though the page contains non-adult terms it could barely offset the very high positive weight given to other features such as the name of the domain or other site-specific features. This is just one example of the bias that can be introduced in the training data. To compensate for the change in the feature distributions between the training set and the real traffic, we relied on active learning [27]. The idea behind active learning is to iteratively select a small set of unlabeled examples that can maximally improve the classifier's performance had it been labeled and added to the training set. The trick is to identify this set prior to it being labeled.

Since SVMs with linear kernels were used as the underlying classifier, we employed with modifications the active learning procedure from [27] which was introduced specifically for fast convergence for a large margin classifier. The objective of the procedure is to select examples that would maximally shrink the space of admissible hypothesis which also contains the true hypothesis that would be learned if all data was labeled in advance. It has been shown that the examples which truncate the space most are those which are closest to the separation hyperplane. We refer the reader to [27] for details and discuss instead certain practical issues encountered while training the sensitivity models and which have not received much attention in the literature.

There were three practical problems that we were confronted with:

- Should we allow the feature space to change from one iteration to the next?
- Should the expensive process of parameter selection be performed after each iteration?
- How do we improve the diversity of the active learned set?

Feature Space: As new pages are being labeled and added to the training set, it is quite possible that new features that were hitherto unseen are also being made available to the classifier. For example, consider a classifier for *Alcohol* related pages and suppose that the training data did not contain the feature "*Budweiser*" and we restrict ourselves to the original feature space. Now even if we select "*Budweiser*" related pages from the traffic for labeling we risk missing a very important feature for this classifier by the feature space restriction. On the other hand, allowing the feature space to change between active learning iterations essentially changes the hypothesis space itself and all guarantees of the active learning algorithm converging to a good solution are off as each classifier essentially undergoes only one round of active learning. Whether this method of changing hypothesis spaces gives us a similar performing classifier as that obtained with a fixed feature space is a question worth exploring. In our training process, we tried to minimize the effect of losing out on important features by computing a global dictionary using all pages from the traffic prior to the process of training data generation. Constructing a global dictionary requires no manual effort and is easily parallelizable. Thus all the pages from the traffic to the ad-matching platform over a period of a month, in the order of a few tens of millions, were used to compute the feature dictionary. While this approach still has a problem of being unable to exploit a new feature introduced sometime in the future, it is better than restricting ourselves to the feature space of the training data. Note also, that the only feature pruning method that was performed was based on the document frequency of the feature in question. Words that appeared in less than 10 pages from all traffic were removed from the dictionary.

Parameter Space: As mentioned in Section 4.2 each SVM has the following tunable parameters (C, c_1). As with the feature space, the question here is does one allow the parameters to change between active learning iterations? Allowing parameters to change at each iteration causes the hypothesis space to change thereby affecting convergence possibilities. Experimental results showed that while this method ultimately lead to better classifiers, it has very high variance in its performance in the initial iterations. This makes it necessary to have better methods at detecting the convergence of the active learned classifier and performing additional active learning iterations until convergence is detected. The effect of the global (one setting used across all iterations) versus local (different settings for each iteration) parameter selection can be seen in Figure 2 which is a simulation of the active learning environment. Given the erratic convergence of the local parameter selection technique, and the limited editorial support which restricted the number of active learning iterations that can be performed, we resorted to performing active learning with a single global setting. The best set of parameters (C^*, c_1^*) was picked on the initial training data using 5-fold cross validation and subsequent SVMs learned on the different active learning iterations used the same set of parameters. Thus, at any given active learning iterations, we are guaranteed to have a classifier that performs better than the previous one.

Diversity of the Active Learning Set: Most active learning approaches advocate the idea of adding one newly labeled data point during each iteration to the initial training set [4]. This setting makes sense if the turnaround time between obtaining a newly labeled data point from the editor to generating the next data point is small enough such that the editorial bandwidth is fully occupied at all times. However, the turnaround time in our system was approximately 15 minutes and this when data was distributed across multiple nodes and a parallel process used to score the preprocessed few tens of millions of pages. Thus, the maximum number of newly labeled samples that can be added to the training data in a day is ≈ 40. Instead, to maximize editorial

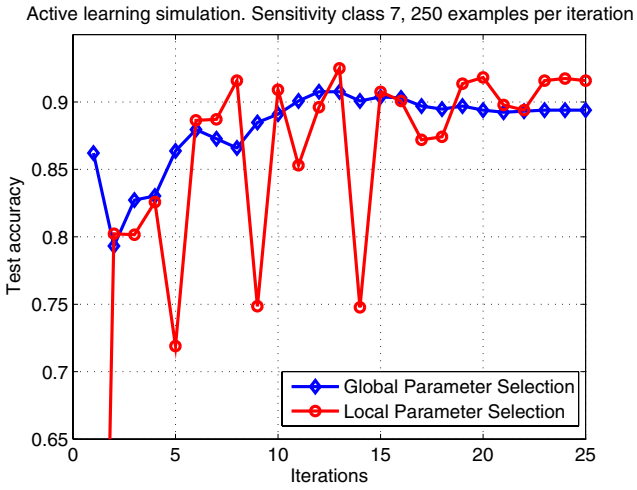


Figure 2: Effects of Global versus Local parameter selection in active learning.

bandwidth we resorted to batch active learning techniques in which 50 data points were presented to the editor for each active learning iteration. The active learning sets were selected from the traffic to the ad matching platform over a few months and the traffic contains multiple copies of the same page. Rather than waste editorial resources by presenting copies of a page for labeling we attempted to improve the diversity of the active learning sample by using the technique proposed in [4]. The approach proposed in [4] was modified to be a two step-process. In the first step, the top 200 data-points that were closest to the SVM decision boundary were first chosen. Within this set, we then maximized the angle diversity as proposed in [4]. The resulting set thereby not only contained no duplicates but we also improved the convergence time of the active learning algorithm. As we were interested not only in the accuracy of the classifier but in its performance across the entire score interval, at each iteration we also drew a small set of examples randomly selected from different scores. Section 4.5 demonstrates the effect of the active set generation methods discussed here on the final classifier.

4.4 Calibration of SVM output

SVMs output a raw score between $-\infty$ to ∞, the distance of the data point to the margin. A requirement for our system, however, is to have a score with a certain probabilistic meaning, so that subsequently an appropriate threshold could be computed. We therefore, calibrate the raw scores of the classifiers. By definition [9], a classifier is well calibrated if $p(x = 1) = s(x), \forall x$, where $s(x)$ is the score of the example. E.g., if there are 100 examples in the population to which the classifier assigns score 0.2, we should expect 20 of them to be positive and the rest negative. Suppose that the business requirement for the system is to have precision better than say 70%. We can then set the threshold on the calibrated scores to 0.7, i.e. treat examples with score ≥ 0.7 as positive and the rest as negative. If the classifier is well calibrated, among all examples with score 0.7 70% will be positive, among the ones with score 0.8 80% will be positive etc. Therefore, the precision for all examples with score ≥ 0.7 will be not less than 0.7. A provably optimal, in a maximum likelihood sense, and very efficient algorithm to compute calibrated from raw scores is the Pairwise Adjacent Violators [30]. We also use the method to calibrate the scores of the categorization system. As suggested in [30] the calibration function is estimated using the final training dataset that was obtained after the active learning iterations. One should keep in mind though that calibration is merely a way to give probabilistic meaning to the raw scores. Just running a calibration method, such as PAV, does not produce a "well" calibrated, as per the above definition, classifier. For that the initial classifier should already be accurate and its raw scores should rank correctly all examples in the population.

4.5 Performance Evaluation on the Traffic

We now explain how the classifier performance is evaluated on real-world traffic.

4.5.1 Performance Metrics:

Let X represent the set of data points that is to be classified and let x be a specific data instance. Let Y be a random variable that can be assigned one of a finite set of labels. Thus, in the case of binary classifiers, $Y \in \{0, 1\}$. Now, (x, y) represents a specific labeled instance. Let $S \in R$ be a random variable that represents the score assigned to X by the classifier. Thus, for each data point we have a (s, y) pair that we use to define the evaluation metrics. The score S being real-valued, we expect to set a threshold (θ) such that the predicted label pr, is 1 when the corresponding score $s \geq \theta$ else $pr = 0$. Thus, $PR \in \{0, 1\}$ is another random variable that represents the predicted label. Let $P(.)$ represent the probability of the random variable in question.

We use Precision and Recall for evaluation. As shown in [8], Precision-Recall curves are better suited for estimating the performance of classifiers on imbalanced data sets than ROC curves.

Precision is defined as the ratio of the cardinality of the true positive set to that of the positive set.

$$\begin{aligned} Precision &= P(Y = 1|PR = 1) \\ &= \frac{P(Y = 1, PR = 1)}{P(PR = 1)} \end{aligned} \quad (1)$$

Recall (or the True Positive Rate) is defined as the ratio of the cardinality of the true positive set to that of the true set.

$$\begin{aligned} Recall &= P(PR = 1|Y = 1) \\ &= \frac{P(Y = 1, PR = 1)}{P(Y = 1)} \end{aligned} \quad (2)$$

We see from the above equations that we need to estimate only $P(Y, PR)$ to obtain the evaluation metrics we are interested in. The remaining probabilities $P(Y = 1)$ can be estimated by summing $P(Y, PR)$ over PR, while $P(PR = 1)$ can be obtained directly by counting.

As explained earlier, obtaining a simple random sample from the traffic (or the population we are interested in) to evaluate the classifier trained on imbalanced datasets is not a feasible approach especially when the cardinality of the population is in the millions. An evaluation set constructed by simple random sampling would have to include few thousands of negative examples before obtaining even a single positive example. In such cases a stratified sampling ap-

proach is more feasible as the size of evaluation set with such a sampling scheme can be restricted to be manageable.

4.5.2 Stratified Sampling Scheme

The random variable S representing the score is first converted into a discrete variable B using a simple binning scheme. The PAV calibration function produces a step function (cf. [30]) that maps a range of raw SVM scores to a set of discrete scores. We make use of the discrete PAV scores as our binning boundaries. Thus, a data point x with a score s is directly assigned a binned score b when calibrated. With this mapping function in place, we then select a stratified random sample from the traffic for our evaluation set.

We first obtain the scores b for each x in the target population set X. Let X_b represent the set of data points whose binned score is b and $|X_b|$ be the cardinality of this set. Then for each b we pick a simple random sample from the corresponding X_b set and add it to our evaluation set. Typically, the number of binned scores are in the order of a few 10s and we typically sample approximately 100 data points from each bin. The evaluation set thus obtained is then labeled by the editors.

4.5.3 Scaling up to the Traffic

Ideally, the performance of a classifier should be studied on the traffic or at least a simple random sample of it. However, using the stratified sampling scheme violates this requirement. Hence, we propose a methodology for weighting up or down the samples in the evaluation set dependent on the score bin that they belong to. The final set of weighted data points in the evaluation set is equivalent to a simple random sample from the population.

Given that we want to estimate precision and recall, the predicted label is obtained by setting a threshold, say θ, as before. All the data points with a binned score b greater than or equal to the threshold will have the corresponding predicted label $pr = 1$. As mentioned in 4.5.1, all we need to estimate is $P(Y, PR)$ on the population. We use the subscripts Po or Ev to indicate whether the probability estimates are obtained on the sample or on the population.

$$\begin{aligned} P_{Po}(Y, PR = 1) &= P_{Po}(Y|PR = 1)P_{Po}(PR = 1) \\ &= \sum_{b \geq \theta} P_{Po}(Y|B = b)P_{Po}(B = b) \\ &\approx \sum_{b \geq \theta} P_{Ev}(Y|B = b)P_{Po}(B = b) \\ &= \sum_{b \geq \theta} P_{Ev}P(Y, B = b)\frac{P_{Po}(B = b)}{P_{Ev}(B = b)} \quad (3) \end{aligned}$$

Similarly, we define $P_{Po}(Y, PR = 0)$ as follows:

$$P_{Po}(Y, PR = 0) \approx \sum_{b < \theta} P_{Ev}P(Y, B = b)\frac{P_{Po}(B = b)}{P_{Ev}(B = b)} \quad (4)$$

While the $P_{Po}(B = b)$ and $P_{Ev}(B = b)$ are directly estimated from the population and the evaluation set, we make use of the editorial labels on the evaluation set to estimate $P_{Ev}P(Y, B = b)$.

As can be seen in Eqns 3 and 4, the effect of the reweighting primarily comes from $P_{Po}(B = b)/P_{Ev}(B = b)$. The distribution of the binned scores in the population is expected to be very different from that of the evaluation set. In particular, the evaluation set might have oversampled or undersampled data in certain bins when comparing it to the population distribution. We respectively weight down or weight up such effects thereby making sure that the final distribution of binned scores in the evaluation set is similar to that on the real-world traffic. Once the precision-recall curve is estimated on the traffic, the appropriate operating threshold can be picked depending on the levels of recall and precision that are needed.

5. EVALUATION OF THE SYSTEM

In this section, we provide specific details about the binary classification problems that we evaluated our system on. We also show the performance improvements in the precision-recall of the system after the use of active learning.

5.1 Categorization Problems

We evaluate the system on a number of sensitive categories which the ad matching platform needs to handle differently from the rest of the traffic. In particular, we build models for detecting *adult* content pages; pages promoting *alcohol*, *drugs*, *tobacco* or *gambling*; *hard news* describing death and suffering; news on *controversial* topics, such as abortion issues or news about *scandals*, e.g. celebrities misbehavior; and lastly, pages discussing or promoting *firearms* or other type of *weapons* used in martial arts, hunting, fishing etc. We focus once again the attention of the reader to the subtleties in the nuances in some of the required categories - pages about over-the-counter medicines are treated differently from pages for illegal drugs, and so are pages with scientific discussions on the process of distilling alcohol or growing tobacco from pages promoting these products.

Two of the most important categories are adult and hard news, and we build several models for these categories for several high traffic markets. The reason for the importance of the adult category lies in the fact that adult websites are constantly generated through automated means and therefore the number of unique adult pages is significant in the population. Breaking hard news, on the other hand, e.g a death of a celebrity or a burst of an armed conflict, attract vast numbers of unique users. This effect has escalated during the last years with users on social networks quickly propagating certain news links across the networks.

Category	Training Data	% Positives
Sensitivity Class 1	13196	78%
Sensitivity Class 2	24721	5%
Sensitivity Class 3	30574	51%
Sensitivity Class 4	35738	51%
Sensitivity Class 5	36198	38%
Sensitivity Class 6	23172	10%
Sensitivity Class 7	10285	58%
Sensitivity Class 8	12776	6%
Sensitivity Class 9	18954	37%
Sensitivity Class 10	16098	30%
Sensitivity Class 11	8132	55%
Sensitivity Class 12	5707	43%

Table 1: Characteristics of the Categorization Problems.

Table 1 provides details on the size of initial training data and the proportion of positive examples in it for each of the categorizers that were trained as a part of the active learning system. For confidentiality reasons, we henceforth refer to the specific categorizers only as 'Sensitivity Class [1-12]'. The size of the training dataset varies from a few thousands to tens of thousands. The percentage of positive data in the training set also varies from 5% to 78%. Note that these percentages are hardly reflective of the actual percentages of such pages in real-world traffic. For example, the percentage of pages about Alcohol in the traffic to the ad matching platform is believed to be orders of magnitude smaller than the 51% seen in the initial training data. As mentioned in Section 4.1, a sizable number of negative pages is also added as negative examples to each of the training sets. This typically results in an addition of approximately 100K to 200K negative data points to each categorization problem. Due to this modification the problems at hand become severely imbalanced with less than 1% sensitive examples for some of the models. The hope, however, is that the negative class is rich enough to capture many non-sensitive aspects and thus reduce the false positive rates significantly.

5.2 Performance Results

Features were extracted from the training data using the bag-of-words approach appended with HTML tag information. Liblinear SVMs were used as the base classifier and the parameters (C^*, c_1^*) described in Section 4.1 were set using 5-fold cross-validation on the training data. Pages from the ad platform traffic spanning over more than two months were used for active learning iterations. The output scores on the training data of the model from the final active learning iteration were calibrated as detailed in Section 4. Note that editorial resources varied over the course of the deployment and thus we have different numbers of active learning iterations for different classifiers. As mentioned in Section 4.5.1, the measure of interest is precision at a pre-specified level of recall. Since it is impossible to evaluate the classifier on all traffic and the rareness of the categories of interest makes random sampling useless, we instead resort to the stratified sampling technique as detailed in Section 4.5.3 and obtain an evaluation set with approximately 1500 test examples for each model. This is done for both the initial model and the model after active learning. This set of 1500 is then labeled by experts and the precision recall estimates from it are then extrapolated to the traffic as detailed earlier. Table 2 shows the number of active learning iterations and the percentage lift in the precision estimates at a specified recall level and in the area under the precision-recall curve of the active learned classifier over the initial classifier.

We also show the entire precision recall graphs before and after active learning (Figure 3) for two of the sensitivity categories. Similar trends were observed across the rest of the models too. The curves from the plots reveal the insights behind the striking improvements summarized in Table 2. We can see that if we treat all 1500 test examples equally (the solid red curves on the plots) then, although not perfect, the initial models perform relatively well (Figures a) and c)). This performance, however, is not well correlated with the expected performance on the traffic (the dashed blue curves on the same plots). We now explain this behavior.

In general, due to the rareness of the listed categories, we intuitively expect that for any of them the positive examples in the population will not exceed 5%. Let us assume that our population contains 10.5M examples, which means that not more than 500K examples (≈5%) are positive. Suppose, that we have a score bucket [0,0.01]. We would expect (see explanation below), that if the classifier is well calibrated, most of the population, say 10M examples, would fall into this score bucket. We now sample 100 examples from the bucket. If the editors label 5 of these examples as positive, then extrapolating the examples to the population results in 500K expected positives among the 10M examples falling into this score bucket. But this is what we expect to be the maximum number of all positive (rare class) examples in the entire population. What this implies is that if we want to recall even only 20-30% of all positive examples we need to set our threshold to the lower bound of the interval [0,0.01], i.e. 0. Naturally, the precision for such threshold is extremely low. To avoid scenarios like the one described, a good model should be almost perfect on the high volume low score buckets, i.e. for the 100 examples sampled from the above score bucket the editors should return on average no more than 1 positive label. Failing to achieve that suggests a model that will have unacceptably low precision when deployed in production. This is exactly the behavior of the initial models as depicted by the extrapolated dashed graphs on Figure 3(a) and 3(c).

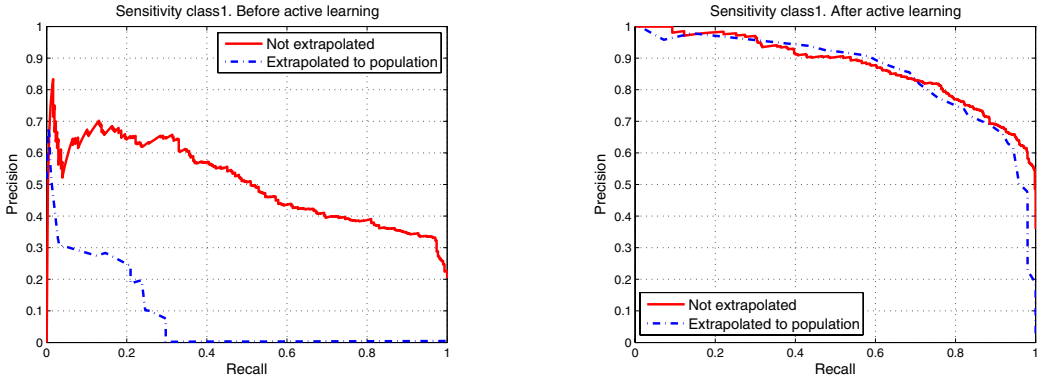
We stated above that if the classifier is well calibrated we should expect the majority of the 10.5M examples to fall in a very low score bucket, such as [0,0.01]. To see this, suppose that we have a score bucket [0.1,0.2] and that 10M examples are assigned a score within this bucket by a classifier. According to the definition of a well calibrated classifier [9] we should expect 10-20% of the examples, i.e. 1-2M examples, in the bucket to be positive. But this is far more than the expected number of all positive examples, which simply implies that the classifier is not well calibrated and assigns scores incorrectly. Therefore, especially when our understanding is that the category is very rare, say less than 0.1% of the population, a well calibrated classifier will assign a very low score (close to 0) to the majority of the examples.

Figure 3(b) and 3(d) shows improvement in both the unweighted and the extrapolated to population curves after active learning. It is interesting to note, however, that the unweighted curves (solid graphs) are very close to the weighted ones (dashed graphs). It means that the classifiers are now well calibrated and whatever results we see on the test set they remain valid after extrapolation to the population too. To the best of our knowledge, application of active learning to produce classifiers that are not only accurate at a pre-specified threshold but also better calibrated across all score ranges, has not been discussed previously in the literature. We believe we observe the effect due to the **diversified active set heuristics** described earlier.

We finish this section, by pointing out that when the category of interest is not very rich in concepts and the initial training set captures most of its nuances, the improvement introduced by active learning is not that substantial, see e.g. the results for Sensitivity Class 12.

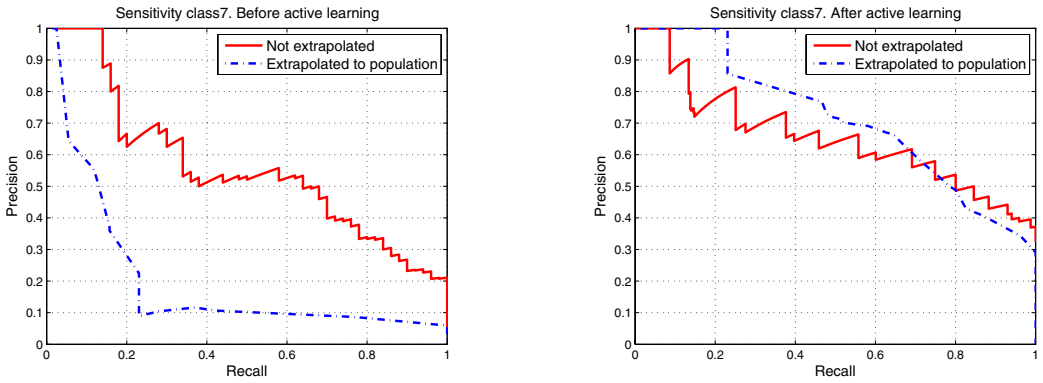
6. CONCLUSION

In this paper, we described the challenges and practical issues related to the deployment of a large scale active learning system for the categorization of rare classes on the Web. The proposed solution employing active learning provided



(a) Sensitivity Class 1. Before AL.

(b) Sensitivity Class 1. After AL.



(c) Sensitivity Class 7. Before AL.

(d) Sensitivity Class 7. After AL.

Figure 3: Precision Recall Curves before and after Active Learning with and without extrapolation to the Traffic for a couple of Sensitivity Categories.

us with a very efficient and effective way to quickly design a classifier and evaluate its performance on real-world traffic. Our active learning system, not only provides us valuable savings in editorial time but enables us to maximize the use of the labeling process, converging to classifiers that significantly outperform the initial models. In the future, we plan to expand this technique to work with multi-label and multi-class categorization problems, and attempt to perform knowledge transfer wherein a model learned for one market can be brought up to speed via active learning.

7. ACKNOWLEDGMENTS

The authors would like to thank Dun Liu, Wanlin Pang and Sanjay Kshetramade for their invaluable help in deploying the prototype.

8. REFERENCES

[1] H. Becker, A. Broder, E. Gabrilovich, V. Josifovski, and B. Pang. Context transfer in search advertising. In *SIGIR '09: Proc. of 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 656–657, 2009.

[2] S. Bickel and T. Scheffer. Dirichlet-enhanced spam filtering based on biased samples. In *Advances in Neural Information Processing Systems 19*, pages 161–168. MIT Press, 2007.

[3] K. M. Borgwardt, A. Gretton, M. J. Rasch, H.-P. Kriegel, B. Schölkopf, and A. J. Smola. Integrating structured biological data by kernel maximum mean discrepancy. In *ISMB (Supplement of Bioinformatics)*, pages 49–57, 2006.

[4] K. Brinker. Incorporating diversity in active learning with support vector machines. In *ICML '03: Proc. of the 20th International Conference on Machine learning*, pages 408–415, 2003.

[5] C. J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[6] P. Chan and S. J. Stolfo. Toward scalable learning with non-uniform distributions: Effects and a multi-classifier approach. In *KDD '99: Proc. of the 4th International Conference on Knowledge Discovery and Data Mining*, pages 164–168. AAAI Press, 1999.

[7] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357, 2002.

[8] J. Davis and M. Goadrich. The relationship between precision-recall and roc curves. In *In ICML Š06: Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

[9] M. DeGroot and S. Fienberg. The comparison and evaluation of forecasters. *The Statistician*, 32:12–22, 1983.

[10] P. Domingos. Metacost: A general method for making classifiers cost-sensitive. In *Proc. of the 5th*

Model	% Lift in Prec@Rec=0.3	% Lift in Prec@Rec=0.8	% Lift in Area Under PRC	AL Iterations
Sensitivity Class 1	1113%	7400%	963%	10
Sensitivity Class 2	82%	36%	50%	46
Sensitivity Class 3	265%	97%	186%	40
Sensitivity Class 4	80%	21%	49%	42
Sensitivity Class 5	46%	36%	32%	35
Sensitivity Class 6	27%	13%	22%	41
Sensitivity Class 7	655%	512%	255%	10
Sensitivity Class 8	71%	42%	38%	20
Sensitivity Class 9	144%	24%	129%	30
Sensitivity Class 10	29%	24%	18%	23
Sensitivity Class 11	940%	633%	860%	17
Sensitivity Class 12	-1%	6%	5%	24

Table 2: Performance of the deployed models. Percentage improvement in Precision at recall 30% and 80% and Area under the Precision-Recall curve of the active learned model over the model prior to active learning.

International Conference on Knowledge Discovery and Data Mining, pages 155–164, 1999.

[11] S. Ertekin, J. Huang, L. Bottou, and L. Giles. Learning on the border: active learning in imbalanced data classification. In *CIKM '07: Proc. of the 16th ACM Conference on Information and Knowledge Management*, pages 127–136, New York, NY, USA, 2007. ACM.

[12] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. Liblinear: A library for large linear classification. *J. Mach. Learn. Res.*, 9:1871–1874, 2008.

[13] C.-J. Hsieh, K.-W. Chang, C.-J. Lin, S. S. Keerthi, and S. Sundararajan. A dual coordinate descent method for large-scale linear svm. In *ICML '08: Proc. of the 25th International Conference on Machine Learning*, pages 408–415, 2008.

[14] T. Joachims. Text categorization with support vector machines: learning with many relevant features. In *Proc. of 10th European Conference on Machine Learning*, pages 137–142, 1998.

[15] M. V. Joshi, R. C. Agarwal, and V. Kumar. Predicting rare classes: can boosting make any weak learner strong? In *KDD '02: Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 297–306, 2002.

[16] M. V. Joshi, V. Kumar, and R. C. Agarwal. Evaluating boosting algorithms to classify rare classes: Comparison and improvements. In *ICDM '01: Proc. of 1st IEEE International Conference on Data Mining*, 2001.

[17] M. Kubat, R. C. Holte, and S. Matwin. Machine learning for the detection of oil spills in satellite radar images. *Mach. Learn.*, 30(2-3):195–215, 1998.

[18] J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. *J. Mach. Learn. Res.*, 10:777–801, 2009.

[19] A. McCallum and K. Nigam. A comparison of event models for naive bayes text classification, 1998.

[20] A. Mccallum and K. Nigam. Employing em in pool-based active learning for text classification. In *Proc. of the 15th International Conference on Machine Learning*, pages 350–358, 1998.

[21] D. Mladenic and M. Grobelnik. Feature selection for unbalanced class distribution and naive bayes. In *Proc. of the 16th International Conference on Machine Learning (ICML)*, pages 258–267, 1999.

[22] K. Nigam. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

[23] F. Provost. Machine learning from imbalanced data sets 101 (extended abstract). In *Proc. of AAAI Workshop on Imbalanced Data Sets*, 2000.

[24] J. R. Quinlan. *C4.5: programs for machine learning*. Morgan Kaufmann Publishers Inc., 1993.

[25] M. Sugiyama and K. robert Müller. Model selection under covariate shift. In *Proc. of the International Conference on Artificial Neural Networks*. Springer, 2005.

[26] Y. Tang, S. Rrasser, P. Judge, and Y.-Q. Zhang. Fast and effective spam sender detection with granular svm on highly imbalanced mail server behavior data. *International Conference on Collaborative Computing: Networking, Applications and Worksharing*, 0:27, 2006.

[27] S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *J. Mach. Learn. Res.*, 2:45–66, 2002.

[28] K. Woods, J. Solka, C. Priebe, C. Doss, K. Bowyer, and L. Clarke. Comparative evaluation of pattern recognition techniques for detection of microcalcifications. *J. of Intelligent Automation*, 1993.

[29] R. Yan, Y. Liu, R. Jin, and A. Hauptmann. On predicting rare classes with svm ensembles in scene classification. In *In ICASSP*, pages 21–24, 2003.

[30] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proc. of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699, 2002.