Doubly Aggressive Selective Sampling Algorithms for Classification

Koby Crammer
Department of Electrical Engneering, Technion - Israel Institute of Technology, Haifa 32000, Israel

Abstract

Online selective sampling algorithms learn to perform binary classification, and additionally they decided whether to ask, or query, for a label of any given example. We introduce two stochastic linear algorithms and analyze them in the worst-case mistake-bound framework. Even though stochastic, for some inputs, our algorithms query with probability 1 *and* make an update even if there is no mistake, yet the margin is small, hence they are doubly aggressive. We prove bounds in the worst-case settings, which may be lower than previous bounds in some settings. Experiments with 33 document classification datasets, some with $100K$s examples, show the superiority of doubly-aggressive algorithms both in performance and number of queries.

1 Introduction

In many real world problems, input examples are cheap and easy to collect, while labeling them is a long and an expensive process. Today, one can easily access an enormous amount of text, audio, images and video, on the web. Nevertheless, much effort is needed to provide correct output for each input. We study online selective sampling in which algorithms are fed with a sequence of inputs and ask or query labels only for a subset of the inputs. Unlike the well studied area of active-learning, an algorithm has access to one input at a time, and it can query its label only at that time, with no ability to store examples which possibly will be queried in the future.

Another motivation for a selective sampling technique is to reduce the size of kernel-based models built online, such as the Perceptron algorithm combined with kernels. This is because the size of the model is bounded by the number of queried labels, which may be small for selective sampling.

Many algorithms for selective sampling, that build on linear models, query for a label of an input $\mathbf{x}$, if the absolute value of the margin is small, $|\boldsymbol{w} \cdot \mathbf{x}|$ (e.g. [11, 24]). However, this quantity may be small in one of two cases: (1) when only few labels are observed (i.e. uncertainty due to a small sample, and there is a real need to query); (2) if there is labeling noise for $\mathbf{x}$ (uncertainty due to noise, and there is no real need to query). Previous algorithms are not distinguishing between these cases, and thus are not focusing their queries in the first case.

We revisit the design of selective sampling with linear models based on this observation. As a by product, our algorithms query more aggressively in early rounds where the uncertainty is large. We introduce *doubly aggressive algorithms* for selective sampling (DAGGER), which query for a label based both on margin and a notion of confidence (intervals) (e.g. [10]).

We present two algorithms that integrate both quantities to decide whether to query. One algorithm builds on the work of Cesa-Bianchi et al [11], and the other on a recently introduced algorithm for online regression of Moroshko and Crammer [22]. We analyze the theoretical properties of both algorithms, showing bounds that may be lower than bounds of other algorithms in some settings. Extensive evaluation on 33 datasets, with hundred of thousand examples, demonstrate the superiority of our algorithms, which achieve lower test error with less number of queries. Finally, all figures in the paper are best viewed in color.

2 Problem Setting

Linear selective-sampling online algorithms maintain a weight vector $\boldsymbol{w}_i$ and work in rounds. On the ith round an algorithm receives an input $\mathbf{x}_i \in \mathbb{R}^d$ and outputs a binary label $\hat{y}_i = \text{sign}(\mathbf{x}_i^\top \boldsymbol{w}_{i-1}) \in \{\pm 1\}$. The algorithm then makes (a possibly) stochastic decision if to ask (or query) for the true label $y_i \in \{\pm 1\}$. If indeed the label is asked for, the algorithm may use y_i to update its prediction model to $\boldsymbol{w}_i$. Otherwise, no change is performed, $\boldsymbol{w}_i = \boldsymbol{w}_{i-1}$, and it proceeds to the next round. Algorithms have two complementary goals, to make few prediction mistakes and to ask, or query, small amount of labels.

The decision of the algorithm is represented by a binary

Appearing in Proceedings of the 17^{th} International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33. Copyright 2014 by the authors.

variable Q_i, where $Q_i = 1$ iff the label y_i was queried on the ith round. Note, the algorithm is evaluated on all examples, even those not queered. The two goals above are to have both $\sum_i Q_i$ (queries) and $\sum_i [\![y_i \neq \hat{y}_i]\!]$ (mistakes) small. Algorithms will be analyzed in the worst-case setting, bounding the expected number of mistakes using the cumulative hinge loss, inter alia, defined by,

$$\sum_i \ell\left(y_i \mathbf{v}^\top \mathbf{x}_i\right) \text{ where } \ell\left(y \mathbf{v}^\top \mathbf{x}\right) = \max\left\{0, 1 - y\mathbf{v}^\top \mathbf{x}\right\} .$$

The bounds we describe below are of expected number of mistakes. Expectation of a quantity defined at round i is with respect to all events at prior (and present) of the query, and are independent of future times.

3 Algorithms

We focus in second-order algorithms that are adapted to classification from regression, and specifically, the Aggregating Algorithm for regression (AAR) of Vovk [25], which was also described by Forster [18], and is related to the work of Azoury and Warmuth [3]. On each iteration, these algorithms solve (recursively) a time-dependent batch regression problem. The resulting prediction function depends on two quantities, a vector $\mathbf{b}_i = \sum_j \mathbf{x}_j y_j$ and a matrix $A_i = \mathbf{I} + \sum_j \mathbf{x}_j \mathbf{x}_j^\top$. Cesa-Bianchi et al [9] proposed to adapt the online ridge-regression (squared loss and Euclidean regularization) to online binary classification. Given a new example, the second-order perceptron predicts the sign of the margin,

$$\hat{p}_i = \mathbf{x}_i^\top \left(A_i^{-1} \mathbf{b}_i\right) . \tag{1}$$

A selective-sampling algorithm is required to make two choices, the first choice is if to query a label or not. Consider the case that the absolute margin is large $|\hat{p}_i| \gg 0$. This may be interpreted that the example lies far from the decision boundary, and thus it will not be useful to query the label, as even if the current model is not the optimal one, but only close to it, still it makes the same prediction as the optimal model. The hidden assumption is that the current model is indeed not far from the optimal one, but this clearly is not the case in early rounds. Thus, there is a need to employ an additional quantity that is used to differentiate between these two cases of large absolute margin $|\hat{p}_i|$: (1) computed with a model after observing already examples close (or similar) to the input $\mathbf{x}_i$; (b) computed with a model after seeing only few, if any, examples close (or similar) to the input $\mathbf{x}_i$. In the former case, in fact, there will not be a need to query, while in the later, the quality of the estimator $\hat{p}_i$ is not good, and there is a need to query.

A useful quantity which is used below is the projection of an input $\mathbf{x}_i$ on the inverse of the matrix A_{i-1} (conceptually, it is a covariance-like matrix),

$$r_i = \mathbf{x}_i^\top A_{i-1}^{-1} \mathbf{x}_i , \tag{2}$$

which can be thought of the confidence in the prediction. Indeed, confidence-weighted (CW) linear prediction algorithms [13] employ a similar formal model, and in CW, r_i is exactly the variance in prediction, under a Gaussian distribution. Some algorithms, including AAR, use a related quantity that incorporates also the input $\mathbf{x}_i$ itself. From Woodbury equality we have, $\mathbf{x}_i^\top A_i^{-1} \mathbf{x}_i = \frac{\mathbf{x}_i^\top A_{i-1}^{-1} \mathbf{x}_i}{1 + \mathbf{x}_i^\top A_{i-1}^{-1} \mathbf{x}_i} = \frac{r_i}{1+r_i}$. We will use this quantity exactly for the purpose of differentiating between these two cases. Low values of r_i represent low uncertainty (as many examples close to $\mathbf{x}_i$ were already observed), while large values of r_i reflect high-uncertainty, as only few, if any, examples similar to $\mathbf{x}_i$ were observed. In the extreme case, when $\mathbf{x}_i$ is orthogonal to all previous examples, we have $r_i = 1$.

The second choice an algorithm should make is how to update the model given the input $\mathbf{x}_i$ and the queried label y_i. Previous linear algorithms for selective-sampling [11, 10] are based on the Perceptron update which is passive (or conservative), i.e. it updates only when mistakes occur. Yet, it is known that aggressive updates yield algorithms that perform better (e.g. the second order preceptron [9] vs AROW [14], the latter is aggressive and performs better). The algorithms we describe below couples these two decisions. When they query aggressively (ie with probability 1), they also perform an aggressive update when the signed margin is small, according to a rule of the form $y_i \hat{p}_i \leq f(r_i)$, where $f(r_i)$ is a decreasing function. Hence, they are doubly aggressive, as they query- and update aggressively simultaneously. (Most aggressive algorithms, i.e. PA [12], employ a constant function $f(x) = 1$.)

Intuitively, a stochastic choice to query, means that the algorithm is confident about its own prediction, and thus there is no need for a large or aggressive update. Similarly, a query with probability 1 (i.e. deterministic, which we call below an aggressive query), means that the algorithm is not confident about its own prediction, and the model is not converged yet. Thus, an aggressive update is needed.

We design two selective-sampling algorithms that make a stochastic decision based both on $\hat{p}_i$ and r_i. Both algorithms below flip a coin with bias,

$$2c/\left(2c + \max\{0, F(|\hat{p}_i|, r_i)\}\right) , \tag{3}$$

where $F(\hat{p}_i, r_i)$ is some function increasing in $\hat{p}_i$ and $c > 0$ is a positive parameter. If $F(\hat{p}_i, r_i)$ is negative, a query will be issued aggressively, and otherwise, it will be issued if the margin $\hat{p}_i$ is close to zero, or r_i is large, namely, there were only few past inputs similar to the current input $\mathbf{x}_i$.

To summarize, on each iteration, after receiving $\mathbf{x}_i$, both algorithms compute $\hat{p}_i$ and r_i. They then have one of two alternatives. First, draw a coin with bias defined in (3), each with a specific value of bias, and use the outcome of the draw to decide if to query. If indeed the label is queried, a conservative update is performed, i.e. an update is per-

Algorithm 1 DAGGER-algorithms

```
Input: parameter $c > 0$ ($b > 1$ wemm) (Tab. 1 line 1)
Initialize $\mathbf{b}_0 = \mathbf{0}$ , $A_0 = \mathbf{I}$ (ridge) or $A_0 = b\mathbf{I}$ (wemm)
(Tab. 1 line 2)
for $i = 1, \ldots, m$ do
  Get instance $\mathbf{x}_i \in \mathbb{R}^d$,$\|\mathbf{x}_i\| \leq 1$
  Compute $\hat{p}_i$ (Tab. 1 line 3)
  Predict $y_i = \mathrm{sign}(\hat{p}_i)$
  Compute $r_i = \mathbf{x}_i^\top A_{i-1}^{-1} \mathbf{x}_i$ (2)
  if $F(|\hat{p}_i|, r_i) \geq 0$ (Tab. 1 line 4) then
    Draw a Bernoulli random variable $Q_i \in \{0, 1\}$ with
    probability $\frac{2c}{2c+F(|\hat{p}_i|,r_i)}$
    if $Q_i = 1$ then
      Receive label $y_i \in \{\pm 1\}$
      Set $Z_i = 1$ if $y_i \neq \hat{y}_i$ (and $Z_i = 0$ otherwise)
    end if
  else
    Set $Q_i = 1$ (i.e. $Q_i = 1$ with probability 1).
    Receive label $y_i \in \{\pm 1\}$
    Set $Z_i = 1$.
  end if
   {If $Q_i = 0$ the rhs of both equations below is zero.}
  Set $a_i$ (Tab. 1 line 5)
  $\mathbf{b}_i = \mathbf{b}_{i-1} + a_i Z_i Q_i \mathbf{x}_i y_i$
  $A_i = A_{i-1} + a_i Z_i Q_i \mathbf{x}_i \mathbf{x}_i^\top$
end for
```

formed only if $y_i \hat{p}_i \leq 0$. Second, when an aggressive query is performed, the update is also aggressive, it is performed when $y_i \hat{p}_i \leq f(r_i)$ for some non-decreasing non-negative function $f(r_i)$.

We conclude et al [11, Alg. 3] is a special case for which $F(|\hat{p}_i|, r_i) = \hat{p}_i$ (always stochastic) and $f(r_i) = 0$. We refer to this algorithm by the initials of the authors, namely CBGZ-ridge, where the suffix indicates the underlying regression algorithm. An illustration of this query and update schema is shown in Fig. 1(a) for $c = 0.1$. The bias of querying is fixed with respect to r_i and thus the colored lines are parallel to the y-axis. Clearly, the probability drops fast when $|\hat{p}_i|$ gets larger, it is 0.25 when the margin is about 0.3. Also, the left half-plane is shaded indicating an update if the margin is negative, ignoring the uncertainty r_i.

3.1 DAGGER: Doubly AGGREssive algorithms

Our first algorithm, builds on the work of Cesa-Bianchi et al [11] and employs the following function in the denominator of (3). It is motivated from the analysis in Thm. 1.

$$\Theta(|\hat{p}_i|, r_i) = (1 + r_i)\hat{p}_i^2 + 2|\hat{p}_i| - \frac{r_i}{1 + r_i} \ . \qquad (4)$$

To better understand the query rule, we compute under what conditions a query will be issued aggressively. The function $\Theta(|\hat{p}_i|, r_i)$ is quadratic in $|\hat{p}_i|$, and attains negative values in a closed interval. Solving for $|\hat{p}_i|$ we get,

Table 1: Settings for DAGGER-ridge and DAGGER-wemm

Algorithm	DAGGER-ridge	DAGGER-wemm
1 Parameters	$c > 0$	$c > 0, b > 1$
2 A_0	$\mathbf{I}$	$b\mathbf{I}$
3 $\hat{p}_i$	$\mathbf{x}_i^\top \left(A_{i-1} + \mathbf{x}_i \mathbf{x}_i^\top\right)^{-1} \mathbf{b}_{i-1}$	$\mathbf{x}_i^\top A_{i-1}^{-1} \mathbf{b}_{i-1}$
4 $F(\|\hat{p}_i\|, r_i)$	$\Theta(\|\hat{p}_i\|, r_i)$ (4)	$\Gamma(\|\hat{p}_i\|, r_i)$ (7)
5 a_i	1	$\frac{1}{1-r_i}$ (6)

$$\Theta(|\hat{p}_i|, r_i) \leq 0 \Leftrightarrow |\hat{p}_i| \leq \theta(r_i) = \frac{-1 + \sqrt{1 + r_i}}{1 + r_i} . \qquad (5)$$

That is, there is a first order phase transition point of the bias that depends on r_i. For margin values $|\hat{p}_i|$ less than $\theta(r_i)$, a query will always be issued, while for margin values above this threshold, it will be issued only with some probability strictly less than 1. This upper bound is decreasing with the uncertainty r_i. If the uncertainty is maximal $r_i = 1$, then a query will be issued if the margin is less than $(\sqrt{2} - 1)/2 \approx 0.2$. While if the algorithm observe the same example (or linear combination of it) many times, that is, $r_i \approx 0$, than only if the margin is zero, $\hat{p}_i = 0$, a query will be issued aggressively (with probability 1).

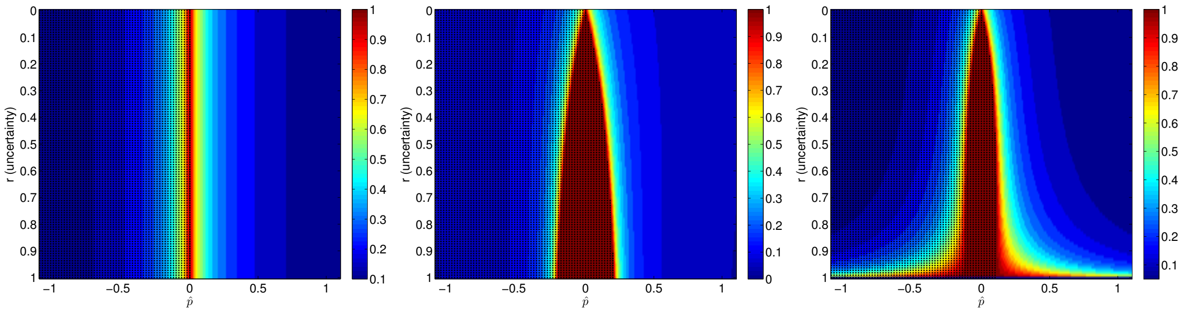
If a stochastic query is made (i.e. with probability less than one, when $\Theta(|\hat{p}_i|, r_i) \geq 0$), then a conservative update is made, that is only when a mistake is performed. Otherwise, if a deterministic query is made (i.e. with probability 1, called aggressive-query), then also an aggressive update is performed $y_i \hat{p}_i \leq \theta(r_i)$ (see (5)), when the margin is negative or small.

We call this algorithm DAGGER-ridge, for a Doubly AGGRessive based on (online) ridge-regression, as an algorithm that (sometimes) is aggressive simultaneously in querying (deterministic, probability 1) and in its update. Pseudocode of DAGGER-ridge is given in Alg. 1 and the left column of Tab. 1.

An illustration of this query and update schema are shown in Fig. 1(b) for $c = 0.1$. For large uncertainty r_i, the bias is 1 (deterministic, or aggressive query) even if the absolute margin is far from zero, $|\hat{p}_i| \gg 0$. As the uncertainty r_i increases, the interval of absolute values with deterministic bias shrinks, until it is a point ($|\hat{p}_i| = 0$, as in CBGZ-ridge). Also, the left half-plane *and the red (middle) area* are shaded, indicating that an update is performed if the margin is negative or if an aggressive query was performed.

3.2 DAGGER-wemm

Our next algorithm builds on a recent online regression algorithm of Crammer and Moroshko [22] called WEMM. The algorithm is a last-step min-max online regression algorithm, as the one of Forster [18], yet it weights the loss



(a) CBGZ-ridge (b) DAGGER-ridge (c) DAGGER-wemm

Figure 1: Bias vs. $\hat{p}$ and r for 3 algorithms. Updates are performed for pairs $(\hat{p}_i, r)$ which are shaded with black dots.

term of the ith example with,

$$a_i = \frac{1}{1-\mathbf{x}_i^\top A_{i-1}^{-1}\mathbf{x}_i} = \frac{1}{1-r_i}\ . \tag{6}$$

On each round, the algorithm finds the minimizer of $b\mathbf{I} + \sum_j^{i-1} a_j \left(y_j - \boldsymbol{w}^\top \mathbf{x}_j\right)^2$, for some parameter $b > 1$. It was shown to have sub-logarithmic regret, and has an aggressive update (for regression). The sequence of weights a_i is effectively giving higher weights to new examples, or directions in space, making it an ideal basis for a new aggressive selective sampling algorithm. For example, if the algorithm receives the same input $\mathbf{x}$ again and again, the weights a_i will decrease from a value larger of a 1, to 1, in the limit. The bias function we use for this algorithm is,

$$\Gamma(|\hat{p}_i|, r_i) = \left(-\frac{r_i}{1+r_i} + 2|\hat{p}_i| + \hat{p}_i^2\right)/a_i\ , \tag{7}$$

which, as before, its specific form is motivated from an analysis, summarized in Thm. 2. The main difference is the multiplicative factor of $1/a_i = (1 - r_i)$. Thus, even if there is no aggressive update ($\Gamma(|\hat{p}_i|, r_i) \geq 0$), the bias (probability of update) is close to 1 as r_i is close to 1, i.e. where there were only few examples similar to $\mathbf{x}_i$.

As before we compute the point of phase transition,

$$\Gamma(|\hat{p}_i|, r_i) \leq 0 \Leftrightarrow |\hat{p}_i| \leq \gamma(r_i) = -1 + \sqrt{1 + \frac{r_i}{1+r_i}}\ , \tag{8}$$

and update conservatively if $\Gamma(|\hat{p}_i|, r_i) \geq 0$ and aggressively ($y_i\hat{p}_i \leq \gamma(r_i)$) otherwise.

We call this algorithm DAGGER-wemm, since it is based on the WEMM algorithm. A pseudocode appears in Alg. 1 and the right column of Tab. 1. Finally, an illustration of this query and update schema is shown in Fig. 1(c) for $c = 0.1$. It has similar properties as DAGGER-ridge, with two differences: for large uncertainty, the probability of query is very close to unit, effectively ignoring the margin $\hat{p}_i$, and the area for which $\Gamma(|\hat{p}_i|, r_i)$ is negative is slightly smaller. To conclude both DAGGER algorithms have high probability rate where there is uncertainty, which naturally occur more on early rounds of the run. Thus, we expect these algorithms to query more on early stages. Additionally, DAGGER algorithms update aggressively (on selected cases) as opposed to CBGZ-ridge, which is conservative.

4 Analysis

We now analyze both algorithms in the worst-case mistake bound model, starting with DAGGER-ridge. We denote for convenience the matrix,

$$A_{\mathbf{v}} = \mathbf{I} + \sum_{i=1}^{m} Z_i Q_i \mathbf{x}_i \mathbf{x}_i^\top\ . \tag{9}$$

The bound partitions the inputs into two disjoint sets of indices, $\mathcal{S} = \{\ i\ : \Theta(|\hat{p}_i|, r_i) \geq 0\ \}$, $\mathcal{A}\{\ i\ : \Theta(|\hat{p}_i|, r_i) < 0\ \}$. The set $\mathcal{S}$ is of examples for which a stochastic query was made, and the other set $\mathcal{A}$ is when there is an aggressive query. Additionally, denote by $\mathcal{M}$ the set of example indices for which the algorithm makes a mistake (that is, where $\hat{p}_i \leq 0$) and let $M = |\mathcal{M}|$. Similarly, denote by $\mathcal{U}$ the set of example indices for which there is an update but not a mistake ($0 < \hat{p}_i \leq \theta(r_i)$) and let $U = |\mathcal{U}|$. The remaining examples, for which the algorithm had a large margin ($\theta(r_i) < \hat{p}_i$), do not affect the behavior of the algorithm and can be ignored.

Theorem 1. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ be an arbitrary input sequence, where $\|\mathbf{x}_i\| \leq 1$. Assume DAGGER-ridge (Alg. 1 and left column of Tab.1) is run on the sequence with parameter $c > 0$. Then, for all $\mathbf{v} \in \mathbb{R}^d$,*

$$\begin{aligned}\mathbb{E}\left[M\right] \leq & \frac{1}{2} c \mathbf{v}^\top \mathbb{E}\left[A_{\mathbf{v}}\right] \mathbf{v} + \mathbb{E}\left[\sum_i Z_i Q_i \ell\left(y_i \mathbf{v}^\top \mathbf{x}_i\right)\right] \\ & + \frac{1}{c}\mathbb{E}\left[\sum_{i\in\mathcal{A}} \frac{r_i}{1+r_i}\right] - \mathbb{E}\left[U\right]\ .\end{aligned} \tag{10}$$

Additionally, the expected number of queries is upper bounded by $\mathbb{E}\left[|\mathcal{A}| + \sum_{i\in\mathcal{S}} \frac{2c}{2c+\Theta(|\hat{p}_i|, r_i\}}\right]$.

Remark 1. *The last term of the bound, $\mathbb{E}\left[\sum_{i\in\mathcal{A}} \frac{r_i}{1+r_i}\right] - \mathbb{E}\left[U\right]$ is strictly lower than the corresponding term of Cesa-Bianchi et al [11, Thm. 3] for two reasons. First, it contains a sum over less examples, $\sum_{i\in\mathcal{A}} \frac{r_i}{1+r_i} \leq \sum_i \frac{r_i}{1+r_i} \leq$*

$\sum_k \log(1+\lambda_k)$, where λ_k are the eigenvalues of $A_{\mathbf{v}}$. Second, the expected number of times the algorithm makes an update when no mistakes occur ($\mathbb{E}[U]$) is subtracted from it. Nevertheless, the bounds are not comparable as both contain quantities that depend on a specific run of each.

Remark 2. *The exact value of the bound depends explicitly on the choice of parameter c. The optimal value of c can be compute easily, $c = \sqrt{\frac{\mathbb{E}\left[\sum_{i\in\mathcal{A}} \frac{r_i}{1+r_i}\right]}{\mathbf{v}^\top \mathbb{E}[A_{\mathbf{v}}]\mathbf{v}}}$, yielding a bound similar to Cesa-Bianchi et al [11, Thm. 3], which may be lower due to Remark 1. Note that there is no real "optimal" parameter as we trade-off between accuracy and number of queries. Additionally, from the experiments below, we see that both DAGGER algorithms perform reasonably well for a large range of values for c, and in fact they outperform other algorithms we evaluated for all values of c used.*

The proof appears in App. A.2 (supplementary material). We conclude by noting that the set $\mathcal{A}$ is not too large. Consider the simple case in which examples are restricted to some basis, e.g. $\mathbf{x}_i \in \{\boldsymbol{e}_1 \ldots \boldsymbol{e}_d\}$. Since examples are orthogonal they do no affect each other. When $\boldsymbol{e}_1$ is observed for the first time we have $\hat{p}_i = 0$ and $r_i = 1$ and thus it will be queried aggressively and be used for an update. Now, if it is seen again, we have $\hat{p}_i = 0.5$ and $r_i = 0.5$, and thus $\Theta(0.5, 0.5) > 0$, i.e., it will not belong to $\mathcal{A}$ again.

We now turn to analyze DAGGER-wemm. We emphasize that weights of WEMM a_i defined in (6) are random variables depending on the specific outcomes of the algorithm's randomization. We define $\mathcal{S} = \{i : \Gamma(|\hat{p}_i|, r_i) \geq 0\}$ and $\mathcal{A}\{i : \Gamma(|\hat{p}_i|, r_i) < 0\}$, analogously to (9), we denote,

$$A^a_{\mathbf{v}} = b\mathbf{I} + \sum_{i=1}^{m} a_i Z_i Q_i \mathbf{x}_i \mathbf{x}_i^\top \ . \tag{11}$$

Theorem 2. *Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m)$ be arbitrary input sequence, where $\|\mathbf{x}_i\| \leq 1$. Assume DAGGER-wemm (Alg. 1 and right column of Tab. 1) is run on the sequence with parameter $b > 1$ and $c > 0$. Then for all $\mathbf{v} \in \mathbb{R}^d$,*

$$\mathbb{E}[M] \leq \frac{1}{2} c \mathbf{v}^\top \mathbb{E}[A^a_{\mathbf{v}}]\mathbf{v} + \frac{b}{b-1}\mathbb{E}\left[\sum_i Z_i Q_i \ell\left(y_i \mathbf{v}^\top \mathbf{x}_i\right)\right] + \frac{1}{c}\left(\left(\frac{b}{b-1}\right)^2 + 1\right)\mathbb{E}\left[\sum_{i\in\mathcal{M}\cup\mathcal{U}} \frac{r_i}{1+r_i}\right] - \mathbb{E}[U] \ .$$

Additionally, the expected number of queries is upper bounded by $\mathbb{E}\left[|\mathcal{A}| + \sum_{i\in\mathcal{S}} \frac{2c}{2c+\Gamma(|\hat{p}_i|, r_i\}}\right]$.

The proof appears in App. A.3 (supplementary material).

Remark 3. *Both Remarks 1 and 2 also apply for this theorem. Its last term, is lower than the last term of Cesa-Bianchi et al [11, Thm. 3], and one can compute the optimal parameter c. Nevertheless, the three terms may be higher than the previous bound, as $A_{\mathbf{v}} \preceq A^a_{\mathbf{v}}$, and $b > 1$. Still, the bounds are not comparable, as they depend on the exact updates of the user, which may yield this bound to be lower than the previous one, as DAGGER-wemm may update more aggressively during the beginning of its exception, but much less later.*

Table 2: Properties of datasets.

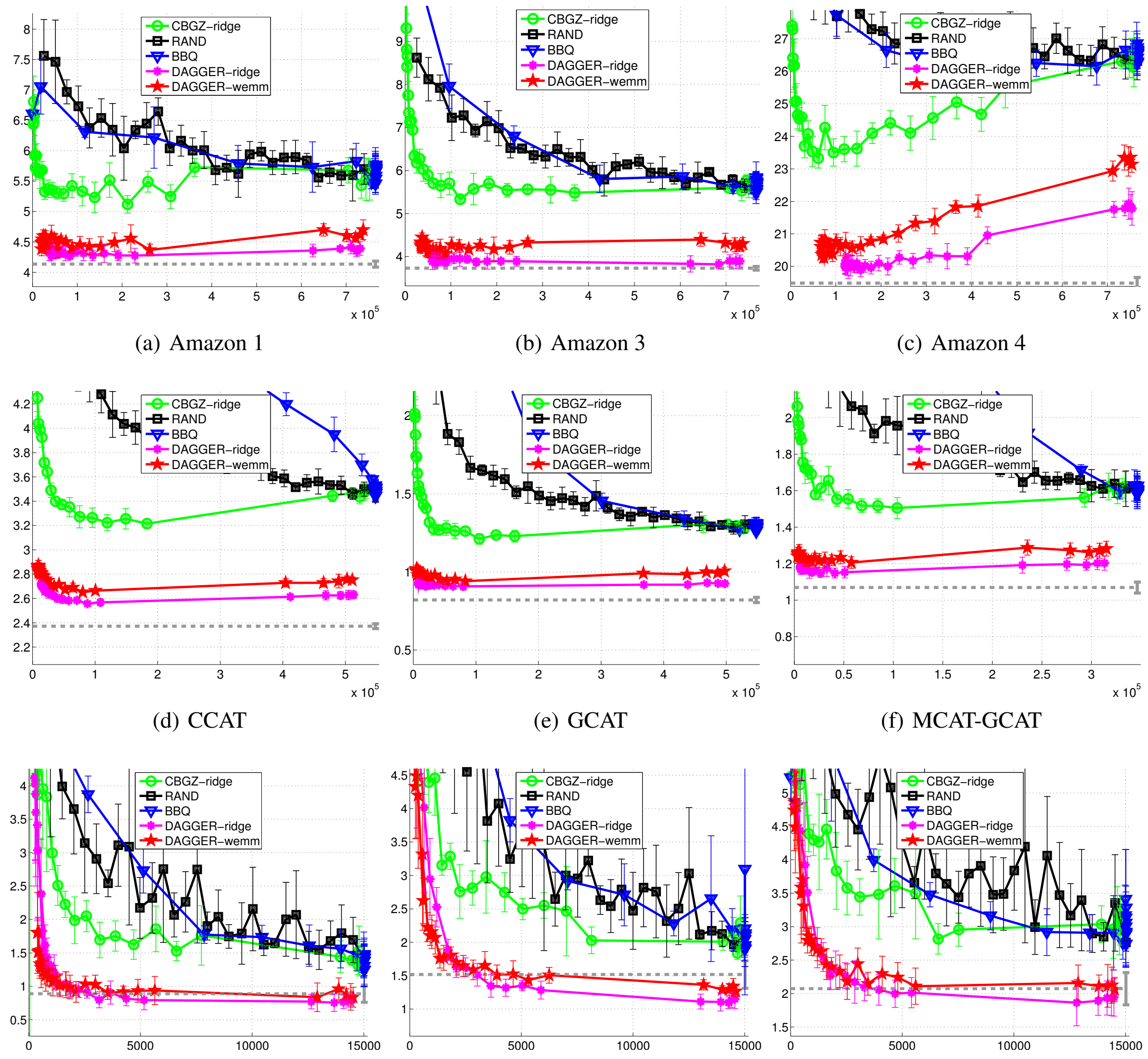
Data	Generated	Examples	Features	Fraction of smaller class
20NG	20 1vs-rest	18K	61K	3.3-5.3%
RCV1	4 1vs-rest	685K	268K	8-43%
RCV1	6 1vs1	273-537K	112K-253K	20-46%
Amazon	3 Rank	769K (4.5K)	16M	6-33%

5 Empirical Evaluation

We evaluate our algorithms on a range of binary document classification problems. We selected three tasks and generated 33 binary classification problems, by either predicting one class against the rest (1vs-rest), or taking pairs of classes (1vs1). We used five-fold cross validation.

20 Newsgroups corpus contains messages partitioned across 20 different newsgroups [1]. We generated 20 1vs-rest binary problems, yielding datasets with about $18K$ examples and $61K$ features. These datasets are highly unbalanced with about 5% positive examples. **The Reuters Corpus** (RCV1-v2/LYRL2004) contains manually categorized newswire stories [21]. Each article contains one or more labels describing its general topic, industry and region. We used the four most general labels (E-, G-, C-, M-CAT) and generated 10 datasets: four 1vs-rest (with about $685K$ examples and $268K$ features) and six 1vs1 sets. Details on document preparation and feature extraction are given by Lewis et al [21]. **Amazon** reviews were used to predict sentiment. We used a larger version of the multi-domain data set of Blitzer et al [6] used by Dredze et al [17, 2]. This data consists of product reviews. The goal in each domain is to classify a product review as either positive or negative. Feature extraction creates unigram and bigram features using counts following Blitzer et al [6]. We induced three binary datasets with $769K$ training examples, $4.5K$ test examples, and $16M$ features. The task in the first dataset was to predict if the number of stars associated with a review is one or more. For the second dataset, it's to predict if the score is two or less, vs more than two. Finally, for the third one, is it four or less, or five. We followed previous preprocessing and omitted all reviews with three stars, as they are very noisy. Properties of all datasets are summarized in Tab. 2.

We evaluated five algorithms: DAGGER-ridge , DAGGER-wemm (with a fixed value of $b = 1.1$) (both algorithms are in Alg. 1 and Tab. 1), CBGZ-ridge of Cesa-Bianchi et al [11, Alg. 3] (performed best in their evaluation), BBQ of Orabona et al [10] and RANDom sampling. Since the data is of high-dimension (up to



(a) Amazon 1 (b) Amazon 3 (c) Amazon 4

(d) CCAT (e) GCAT (f) MCAT-GCAT

(g) rec sport hockey (h) sci space (i) comp windows x

Figure 2: Error vs no of queries labels for 9 datasets: Amazon (first row), 1vs-rest and 1vs1 RCV (second row), and 20NG 1vs-rest (third row). Figure best viewed in color.

$16M$) we diagonalized A_i after processing each example. (Synthetic simulations with full matrices showed similar results as shown below.) Each algorithm was executed with 20 values of the constant c having one pass over the training data, recording the number of queried labels and the test error. We report averages over all folds.

Results for 9 representative datasets appear in Fig. 2 (more results are in Fig. 4 and Fig. 5 in the supplementary material). Each panel summarizes the results for a single dataset, where the first row shows the results for all Amazon datasets, second for two (out of four) 1vs-rest RCV and one (out of six) 1vs1 RCV, and third for three (out of twenty) 1vs-rest 20NG. Each panel plots five lines connecting markers, one line per algorithm. Each marker shows the average (test) error vs average number of updates for a run with a specific choice of the parameter c. A line shows the tradeoff between number of queries (x-axis) and error (y-axis) for some algorithm. Error bars indicate 95% confidence intervals. The horizontal dashed gray line indicates the performance of AROW [15], a state-of-the-art online algorithm, a possible skyline that always makes a query.

In most cases CBGZ-ridge (green-circles) shows a large range of the tradeoff, with the ability to query very little (points at left) to very much (points at right). In some cases RANDom is worse than BBQ , and in other cases, the other way around. Note that BBQ has strong assumptions about the label generating process, and furthermore it ignores the labels when deciding to query.

DAGGER algorithms perform best than the other algorithms, where in general, DAGGER-ridge outperforms DAGGER-wemm. We were not able to drive them to query

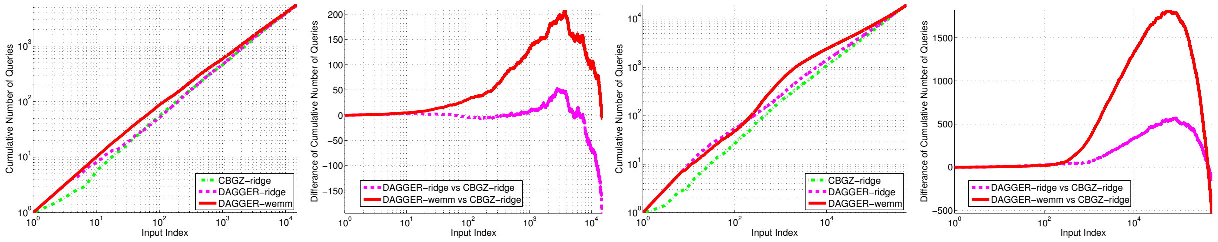
Table 3: Average number of queries and test error Results for six algorithms for 1-vs-rest binary classification problems based on the 20NG dataset, where the maximal number of allowed queries is $1,500$ (1st block) and $4,500$.

		RAND		BBQ		CBGZ-ridge		DAGGER-ridge		DAGGER-wemm		AROW	
		queries	error	queries	error	queries	error	queries	error	queries	error	queries	error
queries< 1500	rec.sport.hockey	969	5.43 (2.50)	560	5.25 (0.13)	1,296	2.51 (0.34)	1,136	1.21 (0.30)	1,191	**1.14** (0.20)	15,019	0.89 (0.13)
	rec.sport.baseball	1,020	6.02 (1.19)	565	5.99 (0.78)	1,248	4.50 (0.70)	1,071	2.26 (0.39)	1,233	**1.85** (0.19)	15,019	1.47 (0.25)
	sci.electronics	1,017	8.70 (3.76)	404	13.33 (10.26)	1,363	5.34 (0.37)	997	5.16 (0.26)	1,346	**4.51** (0.21)	15,019	3.14 (0.18)
	sci.crypt	989	4.49 (0.54)	593	5.89 (0.43)	1,378	3.55 (0.47)	1,335	2.04 (0.51)	1,262	**1.86** (0.41)	15,019	1.48 (0.31)
	sci.space	1,012	5.07 (0.44)	516	5.65 (0.19)	1,450	3.15 (0.17)	1,208	2.52 (0.34)	1,353	**1.76** (0.16)	15,019	1.52 (0.20)
	sci.med	994	7.24 (2.09)	553	6.25 (1.03)	1,399	4.04 (0.38)	815	4.60 (0.73)	1,289	**2.78** (0.72)	15,019	1.68 (0.32)
	talk.politics.guns	1,010	5.31 (0.69)	713	5.11 (0.32)	1,306	4.41 (0.74)	1,296	3.42 (0.93)	1,292	**2.94** (0.35)	15,019	2.09 (0.14)
	soc.religion.christian	994	6.68 (1.85)	785	6.02 (0.86)	1,180	5.40 (1.02)	1,175	**3.04** (0.41)	1,007	3.32 (0.44)	15,019	2.10 (0.17)
	talk.politics.misc	1,004	4.39 (0.39)	698	4.48 (0.56)	1,236	4.24 (0.73)	977	3.70 (0.35)	1,048	**3.35** (0.36)	15,019	2.40 (0.47)
	talk.politics.mideast	996	6.66 (3.63)	690	6.28 (1.38)	1,327	3.56 (1.78)	984	2.34 (1.20)	1,237	**1.44** (0.12)	15,019	1.16 (0.16)
	talk.religion.misc	1,005	5.81 (1.56)	786	3.99 (0.66)	1,470	3.67 (0.50)	1,463	3.23 (0.31)	1,388	**3.22** (0.37)	15,019	2.75 (0.39)
	alt.atheism	986	5.71 (1.04)	704	4.45 (0.40)	1,239	4.12 (0.44)	1,117	**3.43** (0.50)	1,019	3.65 (0.45)	15,019	2.17 (0.39)
	comp.os.ms-windows.misc	1,013	6.14 (0.80)	456	6.25 (0.94)	1,198	4.49 (0.39)	1,162	3.61 (0.61)	1,113	**3.56** (0.21)	15,019	2.74 (0.28)
	comp.graphics	1,010	6.14 (0.67)	416	6.10 (0.32)	1,273	7.08 (3.64)	1,097	3.90 (0.85)	1,138	**3.87** (0.37)	15,019	3.01 (0.34)
	comp.sys.mac.hardware	999	6.25 (0.84)	500	8.85 (4.96)	1,359	4.37 (0.48)	981	4.29 (0.89)	1,232	**3.26** (0.32)	15,019	2.51 (0.23)
	comp.sys.ibm.pc.hardware	993	9.29 (3.02)	441	7.33 (2.49)	1,254	6.02 (1.37)	950	4.70 (0.26)	1,223	**4.34** (0.34)	15,019	3.50 (0.27)
	misc.forsale	983	5.09 (1.22)	349	6.33 (4.76)	1,414	3.00 (0.34)	1,382	**1.84** (0.24)	1,303	1.98 (0.22)	15,019	1.82 (0.25)
	comp.windows.x	1,012	5.78 (0.64)	375	5.76 (0.29)	1,271	4.27 (0.82)	1,199	2.69 (0.34)	1,203	**2.66** (0.19)	15,019	2.07 (0.24)
	rec.motorcycles	997	10.00 (10.38)	432	6.38 (1.15)	1,349	3.92 (1.67)	1,206	2.32 (1.39)	1,300	**1.50** (0.20)	15,019	1.27 (0.13)
	rec.autos	1,017	10.20 (3.14)	519	6.51 (0.46)	1,409	5.55 (1.85)	1,104	3.51 (0.69)	1,303	**2.61** (0.29)	15,019	2.12 (0.34)
queries< 6000	rec.sport.hockey	5,526	2.32 (0.60)	5,125	2.73 (0.39)	5,689	1.86 (0.39)	5,164	**0.79** (0.16)	5,645	0.94 (0.24)	15,019	0.89 (0.13)
	rec.sport.baseball	5,552	3.87 (2.14)	4,332	3.69 (0.28)	5,745	2.24 (0.04)	5,532	**1.38** (0.16)	5,036	1.41 (0.21)	15,019	1.47 (0.25)
	sci.electronics	5,499	4.96 (0.54)	5,702	5.16 (0.34)	5,976	4.44 (0.69)	5,299	**3.04** (0.31)	5,331	3.08 (0.18)	15,019	3.14 (0.18)
	sci.crypt	5,516	3.60 (1.57)	4,825	3.85 (0.30)	5,904	2.79 (0.86)	5,786	**1.44** (0.23)	5,166	1.52 (0.22)	15,019	1.48 (0.31)
	sci.space	5,030	3.68 (0.97)	4,495	3.82 (0.37)	5,064	2.50 (0.18)	5,043	**1.35** (0.09)	4,634	1.52 (0.23)	15,019	1.52 (0.20)
	sci.med	5,015	3.47 (0.55)	4,168	5.56 (1.19)	5,117	2.62 (0.16)	4,447	**1.54** (0.23)	4,737	1.60 (0.21)	15,019	1.68 (0.32)
	talk.politics.guns	5,482	4.11 (1.32)	5,157	4.50 (0.36)	5,794	3.13 (0.38)	4,944	2.15 (0.41)	5,344	**2.12** (0.31)	15,019	2.09 (0.14)
	soc.religion.christian	5,526	5.79 (3.69)	5,406	4.80 (0.50)	5,633	3.25 (0.52)	5,574	**2.00** (0.16)	5,100	2.25 (0.31)	15,019	2.10 (0.17)
	talk.politics.misc	5,505	3.64 (0.58)	5,460	3.88 (0.49)	5,871	3.46 (0.85)	5,792	**2.18** (0.33)	5,426	2.35 (0.42)	15,019	2.40 (0.47)
	talk.politics.mideast	5,524	2.43 (0.47)	5,575	3.98 (0.52)	5,834	1.78 (0.27)	5,457	**0.95** (0.11)	5,280	1.04 (0.10)	15,019	1.16 (0.16)
	talk.religion.misc	5,557	4.61 (1.76)	5,346	5.40 (1.92)	5,695	3.29 (0.54)	5,213	2.76 (0.29)	5,160	**2.68** (0.22)	15,019	2.75 (0.39)
	alt.atheism	5,491	3.81 (0.87)	5,121	4.50 (1.69)	5,816	3.77 (0.81)	5,654	**1.90** (0.24)	5,201	2.26 (0.28)	15,019	2.17 (0.39)
	comp.os.ms-windows.misc	5,053	5.10 (0.83)	3,802	6.87 (3.49)	5,514	5.17 (1.25)	5,420	**2.75** (0.18)	4,725	3.12 (0.16)	15,019	2.74 (0.28)
	comp.graphics	5,447	6.09 (1.75)	3,666	4.91 (0.62)	5,628	4.98 (1.89)	4,831	**2.96** (0.33)	5,028	3.17 (0.38)	15,019	3.01 (0.34)
	comp.sys.mac.hardware	5,511	4.97 (0.93)	3,894	4.49 (0.67)	5,715	4.78 (1.34)	4,898	**2.46** (0.31)	5,095	2.64 (0.39)	15,019	2.51 (0.23)
	comp.sys.ibm.pc.hardware	5,494	5.54 (0.84)	3,603	5.98 (1.36)	5,547	4.46 (0.53)	5,507	**3.53** (0.21)	4,948	3.58 (0.37)	15,019	3.50 (0.27)
	misc.forsale	4,969	3.28 (0.28)	4,250	3.08 (0.31)	5,117	2.43 (0.23)	4,318	**1.75** (0.29)	4,949	2.03 (0.21)	15,019	1.82 (0.25)
	comp.windows.x	5,545	3.66 (0.28)	3,725	4.00 (0.19)	5,604	3.50 (0.78)	5,410	**2.01** (0.26)	5,598	2.11 (0.31)	15,019	2.07 (0.24)
	rec.motorcycles	5,461	4.55 (2.21)	4,453	3.76 (1.13)	5,846	2.09 (0.32)	5,654	**1.20** (0.10)	5,207	1.37 (0.11)	15,019	1.27 (0.13)
	rec.autos	5,506	3.66 (0.68)	4,197	4.40 (0.39)	5,991	3.06 (0.31)	5,202	**2.03** (0.32)	5,339	2.05 (0.35)	15,019	2.12 (0.34)

only a very few labels, despite low values of c used. This is due to the phase transition of the bias which is set to 1 for some values of the margin. This phenomena is more evident in the Amazon (top row) and less in the RCV and 20NG datasets. We hypothesize that it is correlated with the dimension, as it is higher, more examples needed to be observed before $\Theta(\cdot,\cdot)$ or $\Gamma(\cdot,\cdot)$ will not be negative. Also, both DAGGER algorithms perform better than AROW on the 20NG datasets, and close to it on the sentiment datasets, even though they query at most third than AROW.

Interestingly, in some datasets (e.g. Amazon 1 in Fig. 2(a)), the test error is increasing when more labels are observed. For less than $100K$ examples, it is reduced from about 6% to 5%, then when more labels are observed (up to $700K$), the test error goes up to about 5.6%. We now focus in the twenty 20NG datasets using Tab. 3 (more results are in Tab. 4 and Tab. 5 in the supplementary material) where we specify the number of *queries* and the average test *error* for all five algorithms (and AROW) for selected runs. For the first block we set the value of the parameter c such that the number of queries for CBGZ-ridge does not exceed $1,500$, and the number of queries for all other algorithms does not exceed the number of queries of CBGZ-ridge per dataset. In other words, per dataset, CBGZ-ridge uses the maximum number of queries compared to the other selective sampling algorithms. The error of the best performing selective sampling algorithm is emphasized in bold fonts. DAGGER-wemm is the best algorithm for 17 datasets, and DAGGER-ridge for the remaining 3 datasets. Also, these algorithms have a lower confidence interval than CBGZ-ridge. The second block shows the results where the number of queries for CBGZ-ridge is limited to $6,000$ (out of about $15K$ examples). Now DAGGER-ridge is the best for 16 datasets, and as before both DAGGER algorithms are better than the other algorithms, and both have a performance closer to the performance of AROW than of the performance of CBGZ-ridge.

We illustrate the difference of CBGZ-ridge and DAGGER in Fig. 3. We used the comp.windows.x dataset (line 18 in the second block of Tab. 3; dataset name and number of updates are underlined). Fig. 3(a) shows the cumulative number of queries for all three algorithms, CBGZ-ridge made $5,604$ queries, DAGGER-ridge $5,410$ and DAGGER-wemm $5,598$. Fig. 3(b) shows the difference of cumulative number of queries of both DAGGER algorithms with respect to the CBGZ-ridge algorithm. Note



(a) comp.windows.x-cumsum (b) comp.windows.x - diff (c) ECAT vs rest - cumsum (d) ECAT vs rest - diff

Figure 3: Cumulative number of queries (a,c) and difference of the DAGGER algorithms and CBGZ-ridge (b,d), for comp.windows.x dataset (a,b), and ECAT-vs-rest (c,d). x-axis is plotted is log-scale, and y-axis of (a,c) also in log-scale.

the log scale used. observe the run, and in a higher rate than of CBGZ-ridge; and DAGGER-ridge queries more than CBGZ-ridge, in very early stages. Fig. 3(b) shows that until about $3,000$ examples the DAGGER algorithms query more and more aggressively than the CBGZ-ridge algorithm. After that point, both DAGGER algorithms query less aggressively, until they even make less queries than CBGZ-ridge. Figs. 3(c),3(b) show similar trends with ECAT-vs-rest dataset (row 8 of Tab. 4). To conclude, both DAGGER algorithms query more aggressively in earlier rounds, where the wemm based version is more aggressive.

6 Related Work

Selective sampling and active learning, such as query-by-committee [19] and "apple tasting" [20], were studied in the past decades. Clearly, we can not cover all that work. Recent papers [4, 5] discuss stochastic query decisions for active learning. Their methods are designed for general PAC-learnable classes. We focus in linear methods, and use natural quantities such as margin $\hat{p}_i$ and confidence r_i.

Cesa-Bianchi et al [10] and Orabona et al [23] developed algorithms for selective sampling in the stochastic setting. The decisions of their algorithms are based on a quantity r_i (aka upper confidence interval). If r_i is small there is no need to query, and if it is large, there is. Intuitively, if $\mathbf{x}_i$ lies in the span of the preceding examples (computed via A_i) the algorithm has enough "knowledge" from previous examples about the direction of $\mathbf{x}_i$, and there is no need to query. These algorithms are in fact ignoring the margin $|\hat{p}_i|$, and have a domestic query rule, although the authors mention that stochastic queries are possible.

Much of recent work on linear functions is based on a quantity called the margin. It is used in active learning [24], perceptron active learning [16], and SVMs for selective sampling is also based on the margin [7, 8] (also use a finite period of non-stop querying and later sporadic querying). The work of Cesa-Bianchi et al [11] is the closest to our work, and has a stochastic rule with the same form as (3) with $F(|\hat{p}_i|, r_i) = \hat{p}_i$ and $f(r_i) = 0$ (conservative).

Each of these quantities lacks information (separately). Consider for example the case of an input $\mathbf{x}_i$ for which $\hat{p}_i$ is small and r_i is large. Methods that focus only in $\hat{p}_i$ will decide to query (as the margin is close to zero), even though it could be the case that enough samples like $\mathbf{x}_i$ were already observed, yet there is large label noise at this region. Here, $\hat{p}_i$ would be small, so there is no need to query for this input. This information could be inferred from the fact that r_i is not large, which means that indeed $\mathbf{x}_i$ was observed many times. On the other hand, a medium value of r_i means that $\mathbf{x}_i$ was observed a few times already, and thus methods that consider only this quantity would decide not to query, even though it might be the case that this example lies in a region with inputs hard to predict (as $\hat{p}_i$ is small), and thus it is valuable to query. We take the best of these two approaches by considering both quantities.

Finally, recently, few works were published about second order algorithms for classification, e.g. the second order Preceptron [9], confidence weighted classifiers [13], and adaptive regulariztion of weights (AROW) [15].

7 Conclusion

We presented two new online learning selective sampling algorithms, that may query a label of each input example. Our algorithms use both margin and a notion of uncertainty (or variance), to stochastically decide if to query. The algorithms are doubly aggressive, both for querying and updating. We analyzed the algorithms in the worst-case mistake bound setting, and showed that under proper choice of the learning parameter c, they may work as well as algorithms that observe all labels. Experiments with 33 datasets coming from three document classification tasks show that our algorithms perform best compared to the other algorithms, and occasionally, outperform algorithms that observes *all* the input labels. DAGGER-wemm seems outperform DAGGER-ridge when only few inputs are queried, either by being able to query a few input (see Tab. 4,Tab. 5) or having low errors (see Tab. 3). We plan to extend the algorithms for multi-class problems and other settings.

Acknowledgments

This research was funded in part by the Intel Collaborative Research Institute for Computational Intelligence (ICRI-CI), and in part by the Israeli Science Foundation grant ISF-1567/10.

References

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/.

[2] http://www.cs.jhu.edu/~mdredze/datasets/sentiment/.

[3] Katy S. Azoury and Manfred K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

[4] Alina Beygelzimer, Sanjoy Dasgupta, and John Langford. Importance weighted active learning. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pages 49–56, 2009.

[5] Alina Beygelzimer, Daniel Hsu, John Langford, and Zhang Tong. Agnostic active learning without constraints. In J. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R.S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 199–207. 2010.

[6] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *Association for Computational Linguistics (ACL)*, 2007.

[7] Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Learning noisy linear classifiers via adaptive and selective sampling. *Machine Learning*, 83(1):71–102, 2011.

[8] Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. Learning probabilistic linear-threshold classifiers via selective sampling. In *COLT*, pages 373–387, 2003.

[9] Nicolo Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. *Siam Journal of Commutation*, 34(3):640–668, 2005.

[10] Nicolò Cesa-Bianchi, Claudio Gentile, and Francesco Orabona. Robust bounds for classification via selective sampling. In *ICML*, page 16, 2009.

[11] Nicolò Cesa-Bianchi, Claudio Gentile, and Luca Zaniboni. Worst-case analysis of selective sampling for linear classification. *Journal of Machine Learning Research*, 7:1205–1230, 2006.

[12] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *JMLR*, 7:551–585, 2006.

[13] Koby Crammer, Mark Dredze, and Fernando Pereira. Confidence-weighted linear classification for text categorization. *J. Mach. Learn. Res.*, 98888:1891–1926, June 2012.

[14] Koby Crammer and Claudio Gentile. Multiclass classification with bandit feedback using adaptive regularization. *Machine Learning*, 90(3):347–383, 2013.

[15] Koby Crammer, Alex Kulesza, and Mark Dredze. Adaptive regularization of weight vectors. *Machine Learning*, 91(2):155–187, 2013.

[16] Sanjoy Dasgupta, Adam Tauman Kalai, and Claire Monteleoni. Analysis of perceptron-based active learning. In *COLT*, pages 249–263, 2005.

[17] Mark Dredze, Alex Kulesza, and Koby Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1-2):123–149, 2010.

[18] Jurgen Forster. On relative loss bounds in generalized linear regression. In *FCT*, 1999.

[19] Y. Freund, H.S. Seung, E. Shamir, and N. Tishby. Selective sampling using the Query By Committee algorirhm. *Machine Learning*, 28:133–168, 1997.

[20] David P. Helmbold, Nick Littlestone, and Philip M. Long. Apple tasting. *Inf. Comput.*, 161(2):85–139, 2000.

[21] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research (JMLR)*, 5:361–397, 2004.

[22] Edward Moroshko and Koby Crammer. Weighted last-step min-max algorithm with improved sub-logarithmic regret. In *ALT*, pages 245–259, 2012.

[23] Francesco Orabona and Nicolò Cesa-Bianchi. Better algorithms for selective sampling. In *ICML*, pages 433–440, 2011.

[24] Simon Tong and Daphne Koller. Support vector machine active learning with application sto text classification. In *ICML*, pages 999–1006, 2000.

[25] Volodya Vovk. Competitive on-line statistics. *International Statistical Review*, 69, 2001.