Active Learning as Non-Convex Optimization

Andrew Guillory
Computer Science and Engineering
University of Washington
guillory@cs.washington.edu

Erick Chastain
Neurobiology and Behavior
University of Washington
erickc@u.washington.edu

Jeff Bilmes
Electrical Engineering
University of Washington
bilmes@ee.washington.edu

Abstract

We propose a new view of active learning algorithms as optimization. We show that many online active learning algorithms can be viewed as stochastic gradient descent on non-convex objective functions. Variations of some of these algorithms and objective functions have been previously proposed without noting this connection. We also point out a connection between the standard min-margin offline active learning algorithm and non-convex losses. Finally, we discuss and show empirically how viewing active learning as non-convex loss minimization helps explain two previously observed phenomena: certain active learning algorithms achieve better generalization error than passive learning algorithms on certain data sets (Schohn and Cohn, 2000; Bordes et al., 2005) and on other data sets many active learning algorithms are prone to local minima (Schütze et al., 2006).

1 Background

We address the active learning problem in this paper. We assume data points $X \in \mathcal{R}^d$ and labels $Y \in \{-1, 1\}$ are drawn from some fixed unknown distribution $p(x, y)$. We wish to choose the classifier $f \in F$ that minimizes the expected loss $\mathbb{E}_{X,Y}[l(Y, f, X)]$ where l is a loss function. In general, the size of F may be uncountable. In standard *passive* supervised learning we approximately minimize this expected loss by minimizing instead the loss on a set or stream of i.i.d. labeled training examples $(x_i, y_i) \sim p(x, y)$. In active learning, we also have access to i.i.d. x_i, but we selectively query the labels y_i for the examples. In

this way we avoid querying uninformative labels and potentially reduce the number of labels we need. Labeling data is often much more expensive and/or time consuming than collecting input data, so this problem is of great practical importance.

We also wish to make the distinction between *offline* learning algorithms and *online* algorithms. In offline active learning, we have a set of i.i.d. unlabeled examples simultaneously available, and an initially empty set of labeled examples. At each step of learning, we query the label of one or more unlabeled examples based on some criteria, add it to the set of labeled examples, and retrain the classifier on the entire set of labeled examples. This process continues until a stopping criteria is met.

Online algorithms, by contrast, usually operate on a stream of examples. In online active learning, we process a stream of i.i.d. unlabeled examples one by one, deciding at each step whether or not to query for a label and if so update the current classifier. Labeled examples are not saved. In fact, online algorithms have space complexity independent of the number of examples processed and may have time complexity dependent only on the required error (Shalev-Shwartz et al., 2007). This is desirable since it allows online learning algorithms to be run on arbitrarily large data sets. Online learning algorithms are also potentially useful for applications with evolving data distributions where learning never stops. When data is scarce, however, offline algorithms may achieve lower error rates since they can consider all examples at once.

Previous work in active learning has shown that active learning can give provably better label complexity than passive learning under certain distribution assumptions (Balcan et al., 2006; Dasgupta et al., 2007, 2005). Of these algorithms, most that work with noisy data are currently impractical since they minimize zero-one loss. Hanneke (2008) shows that with convex losses, exponentially better label complexity is typically not possible. This is because outlier points far away from the decision boundary can have a large effect on the classifier, so more labels need to be queried. Recently, Beygelzimer et al. (2008)

presented the first practical, consistent active learning algorithm that gives label complexity improvements for convex losses although these improvements are not as good as in the zero-one loss case and require the loss to satisfy certain properties. Moreover, certain active learning algorithms sometimes achieve generalization error below that of passive learning (Schohn and Cohn, 2000; Bordes et al., 2005) on the fully labeled data set. Bordes et al. (2005) speculate that, among other possible causes, these active learning algorithms could be ignoring noise far away from the decision boundary or the sparser solutions found by active learning could have better generalization guarantees. Har-Peled et al. (2007) also draw a connection between active and noise tolerant learning; the authors give label bounds for a variant of min-margin learning for noise-free data and extended this to reject outliers. Some active learning algorithms are also susceptible to local minima (Schütze et al., 2006).

Heuristic *offline* active learning algorithms—for example many algorithms for Support Vector Machines (SVMs) that greedily label the point with the smallest absolute margin—are practical and useful in a variety of settings (Schohn and Cohn, 2000; Tong and Koller, 2001). These algorithms typically do not have guarantees aside from if the algorithm labels the entire data set it converges to the passive solution. Previous work with *online* active learning algorithms includes two perceptron style algorithms (Dasgupta et al., 2005; Cesa-Bianchi et al., 2006). The algorithm of Dasgupta et al. (2005) has provably better error rates under certain noise free distribution assumptions. The algorithm of Cesa-Bianchi et al. (2006) guarantees that it is no worse than passive learning. These two algorithms are practical (Monteleoni and Kaariainen, 2007).

In this work, we view active learning algorithms as optimization. Specifically, we show that many online active learning algorithms can be viewed as stochastic gradient descent on non-convex objective functions. Variations of some of these algorithms and objective functions have been previously proposed (Dasgupta et al., 2005; Cesa-Bianchi et al., 2006; Monteleoni and Kaariainen, 2007) without noting this connection. While the online learning algorithms we present do not have label complexity results, our algorithms differ from previous heuristic active learning algorithms in that they optimize an objective function—when used with noisy data and early stopping heuristics it's not clear what solution, for example, the greedy SVM algorithm finds. We also point out a connection between the standard min-margin offline active learning algorithm and non-convex losses. Finally, we discuss and show empirically how this connection helps explain the aforementioned phenomena: certain active learning algorithms achieve better generalization error than passive learning algorithms on certain data sets (Schohn and Cohn, 2000; Bordes et al., 2005) and on other data sets many active learning algo-
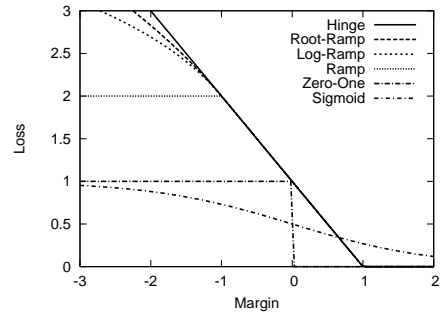

Figure 1: Losses as a function of $yw^T x$. For ramp loss variants we use $t = 1$, for root-ramp loss we use $s = .5$.

rithms are prone to local minima (Schütze et al., 2006).

2 Online Active Learning as Stochastic Gradient Descent

We can think of any online learning algorithm as a stochastic optimization method in which updates to our estimate f correspond to steps which in expectation move along the objective function gradient. If the expected update for a learning algorithm is equal to the gradient of a particular objective function, then the learning algorithm is performing stochastic gradient descent on that objective function. More formally, assuming the stream of data points are sampled from some fixed distribution we require

$$\forall f, \quad -\nabla O(f) \approx \mathbb{E}_{X,Y}[\textbf{Update}(f, X, Y)] \qquad (1)$$

where $\textbf{Update}(f, x, y)$ is a function specifying the update to apply to f for a point x with label y. In other words, when x and y are sampled we set f to $f + \textbf{Update}(f, x, y)$. More generally $\textbf{Update}$ can be a random function in which case we also take the expectation over this randomization. If the objective function is non-differentiable and a *subgradient* of the objective function is equal to the average update then the algorithm performs stochastic sub-gradient descent. To make the analysis concrete, we consider algorithms for linear classifiers $f(x) = w^T x$ and margin based losses $l(x, f, y) = l(yw^T x)$. The objective function is then $O(w) = \mathbb{E}_{X,Y}[l(Yw^T X)]$.

As an example, consider stochastic gradient descent on hinge loss $(l(z) = \max(0, 1 - z))$. The standard hinge loss update function is

$$\textbf{Update}(w, x, y) = \begin{cases} \eta y x & \text{if } yw^T x < 1 \\ 0 & \text{if } yw^T x >= 1 \end{cases}$$

where η is a learning rate. The hinge loss objective is

$$O(w) = \mathbb{E}_{X,Y}[\max(0, 1 - Yw^T X)]$$

Ramp Loss Active Learning

```
1: loop
2:    Sample x
3:    if |wᵀx| < t then
4:       Query y
5:       if ywᵀx < 1 then
6:          w ⇐ w + ηyx
7:       end if
8:    end if
9: end loop
```

Log-Ramp Loss Active Learning

```
1: loop
2:    Sample x
3:    if |wᵀx| < t or Rand(0, 1) < 1/(|wᵀx|−t+1) then
4:       Query y
5:       if ywᵀx < 1 then
6:          w ⇐ w + ηyx
7:       end if
8:    end if
9: end loop
```

Figure 2: Example online active learning algorithms

A valid subgradient of the objective is

$$-\nabla O(w) = \mathbb{E}_{X,Y}[-\nabla \max(0, 1 - Yw^T X)]$$
$$= \mathbb{E}_{X,Y}[-\nabla \max(0, 1 - Yw^T X)I(Yw^T X < 1)]$$
$$= \mathbb{E}_{X,Y}[YXI(Yw^T X < 1)]$$
$$\propto \mathbb{E}_{X,Y}[\mathbf{Update}(w, X, Y)]$$

where I is the indicator function. Here we used $-yx$ as a subgradient for points with $yw^T x < 1$, and the zero vector as a subgradient for points with $yw^T x >= 1$. This update rule satisfies Equation 1 for the hinge loss objective.

This same connection applies for online active learning algorithms. For active learning algorithms, points for which the learning algorithm does not query a label or perform an update intuitively must correspond to points at which the gradient is zero or relatively small. For an algorithm that queries for the label of a point x only if $\mathbf{Query}(f, x) = 1$ and uses $\mathbf{Update}(f, x, y)$ for these points, we require

$$\forall f, \quad -\nabla O(f) \approx \mathbb{E}_{X,Y}[\mathbf{Update}(f, X, Y)\mathbf{Query}(f, X)] \tag{2}$$

$\mathbf{Query}(f, x)$ may also be random, in which case we take the expectation over this randomness as well.

We find with this interpretation many online active learning algorithms optimize non-convex losses. For non-differentiable non-convex losses we use the notion of subgradient used by Kiwiel (1985); the subgradient of f at x is defined to be convex hull of all limits of $\{\nabla f(x_i) : x_i \to x\}$ where f is differentiable at all x_i (i.e. limits of the gradients approaching the point of non differentiability). With non-smooth, non-convex losses we no longer have the standard convergence guarantees for subgradient descent although we find in practice these algorithms still perform well. It is also possible to replace the non-smooth losses with smooth variations (Chapelle, 2007). We first give two examples.

2.1 Ramp Loss Active Learning

The left of Figure 2 shows a very simple threshold based active learning algorithm which has a positive threshold $t >= 1$. The algorithm queries labels for points where $|w^T x| < t$ and then for points with $yw^T x < 1$ performs

a perceptron style update. The algorithm is inspired by the algorithm of Dasgupta et al. (2005) and the offline active learning algorithm of Balcan et al. (2007) (in these papers the threshold changes, the threshold can be less than one, and the updates are different).

The update rule is the same as in the hinge loss case

$$\mathbf{Update}(w, x, y) = \begin{cases} \eta yx & \text{if } yw^T x < 1 \\ 0 & \text{if } yw^T x >= 1 \end{cases}$$

but with an additional sampling rule

$$\mathbf{Query}(w, x) = \begin{cases} 1 & \text{if } |w^T x| < t \\ 0 & \text{if } |w^T x| >= t \end{cases}$$

One objective function which satisfies Equation 2 is a modified version of the hinge loss objective with

$$l(z) = \min(\max(1 - z, 0), t + 1)$$

We call this loss ramp loss. Figure 2 plots it along with other losses used in this paper. With this objective function we have a subgradient satisfying Equation 2

$$-\nabla O(w) = \mathbb{E}_{X,Y}[-\nabla \min(\max(0, 1 - Yw^T X), t + 1)]$$
$$= \mathbb{E}_{X,Y}[YXI(Yw^T X < 1 \text{ and } |w^T X| < t)]$$
$$\propto \mathbb{E}_{X,Y}[\mathbf{Update}(w, X, Y)\mathbf{Query}(w, X)]$$

Here we use $-yx$ as a subgradient for points with $-t < yw^T x < 1$ and the zero vector for other points.

Except for the restriction that $t >= 1$, this loss is identical to the ramp loss used by for example Collobert et al. (2006). Collobert et al. (2006) argue ramp loss, although non-convex, has better scaling properties and can be optimized faster than hinge loss for kernel classifiers because it produces more sparse solutions. They do not however note this connection with active learning. We see this connection with active learning as another possible advantage over hinge loss: with a suitable optimization method, ramp loss optimization may give label savings. Decreasing the threshold while training can be interpreted as optimizing tighter and tighter non-convex bounds on zero-one loss.

2.2 Log-Ramp Loss Active Learning

The right of Figure 2 gives an active learning algorithm inspired by the randomized rule used by Cesa-Bianchi et al. (2006). The probability of querying the label for a point decreases with the inverse of the absolute value of the margin. Cesa-Bianchi et al. (2006) use a slightly different sampling rule, and their algorithm performs updates for any point with $yw^T x < 0$. Here t is again a positive threshold with $t >= 1$. The expected update rule is still

$$\textbf{Update}(w, x, y) = \begin{cases} \eta y x & \text{if } yw^T x < 1 \\ 0 & \text{if } yw^T x >= 1 \end{cases}$$

while the sampling rule is a random function

$$\textbf{Query}(w, x) = \begin{cases} 1 & \text{if } |w^T x| < t \\ 1 & \text{w/ prob. } \frac{1}{|w^T x| - t + 1} \text{ if } |w^T x| >= t \end{cases}$$

A valid loss is similar to ramp loss but with a logarithmic section below $-t$.

$$l(z) = \begin{cases} t + 1 + log(1 - t - z) & \text{if } z < -t, \\ 1 - z & \text{if } -t <= z <= 1, \\ 0 & \text{if } z > 1. \end{cases}$$

We call this log-ramp loss. This algorithm queries labels less selectively than the ramp loss algorithm but is also more smooth and may be easier to optimize.

2.3 Deriving Algorithms for Other Losses

This analysis method is general in that for any active learning algorithm for which we can write down and integrate the update rule, we can derive a corresponding objective function satisfying Equation 2. It's also interesting to consider the converse question: for what loss functions can we derive an active learning algorithm using this method? Considering this question also helps to show why online active learning seems to lead to non-convex losses.

The left of figure 3 gives a general online algorithm for minimizing an arbitrary differentiable margin based loss $l \in \{\Re \to \Re^+\}$ with an arbitrary margin based selective sampling strategy specified by $q \in \{\Re^+ \to [0, 1]\}$. The function q provides the probability of querying for the label of a point given that point's absolute margin. Although q is typically related to l, q may be arbitrary with the restriction that $q(|w^T x|) = 0$ only where $l'(+w^T x) = l'(-w^T x) = 0$. For any such l and q, this algorithm satisfies Equation 2 for $O(w) = \mathbb{E}_{X,Y} l(Yw^T X)$. The weights $\frac{l'(yw^T x)}{q(|w^T x|)}$ used in the update to ensure Equation 2 is satisfied are similar to the importance resampling weights used in Monte Carlo methods and by Beygelzimer et al. (2008). These ensure the expected update for a particular data point is exactly $\eta y x l'(yw^T x)$. For non-differentiable loss functions, a sub-gradient of the loss gives stochastic subgradient descent.

However, for arbitrary l and q, although the update function is proportional to the gradient in expectation, the update may have large variance. Intuitively, if l and q are mismatched, $\frac{l'(yw^T x)}{q(|w^T x|)}$ could be arbitrarily large and the algorithm could make very large steps. This could slow convergence and negate any potential label savings. We therefore suggest using a tight upper bound on the absolute value of the loss derivative:

$$q(|w^T x|) = \min(\max(|l'(+w^T x)|, |l'(-w^T x)|), 1)$$

This q is the smallest valid q such that the weight $\frac{l'(yw^T x)}{q(|w^T x|)}$ is never larger than $\max(1, l'(yw^T x))$. The example algorithms we've presented all correspond to this choice of q.

With this choice of q, the algorithm is naturally suited for non-convex losses for which $\lim_{x \to \infty} q(x) = 0$. For convex losses the algorithm makes less sense since label queries would frequently occur for points with very large margins. For example, with hinge loss, the algorithm would query for every label. In addition to the ramp and log-ramp loss functions, we implemented active learning algorithms for two other losses using this general approach: 1) a ramp variant we call root-ramp loss

$$l(z) = \begin{cases} 0 & \text{if } z > 1, \\ 1 - z & \text{if } -t <= z <= 1, \\ t + 1 - 1/s + 1/s(1 - t - z)^s & \text{if } z < -t. \end{cases}$$

with parameter $s \in (0, 1)$ and 2) a sigmoid loss

$$l(z) = \frac{1}{1 + e^z}$$

We find root-ramp loss interesting because as s approaches 1 the loss approaches hinge loss. A different sigmoid loss was used by Perez-Cruz et al. (2003).

2.4 Greedy Online Active Learning

A popular and effective alternative to margin based selective sampling rules we've discussed is to at each step sample a small set of points and then greedily query for the label of the point with the smallest margin. This labeled point can then be used to update the classifier using any standard rule. This is the strategy used in an offline setting by Schohn and Cohn (2000) and with dual optimization by Bordes et al. (2005).

This kind of algorithm is not as easy to analyze because the probability of querying for an example's label is dependent not only on that example's margin but also on the margins of the other examples. However, assuming we are sampling from a fixed pool of examples, a sufficient statistic for calculating the probability of querying for the label of an example is that example's rank in the list of examples

Margin Based Online Active Learning

1: **loop**
2:    Sample x
3:    **if Rand**$(0, 1) < q(|w^T x|)$ **then**
4:       Query y
5:       $w \Leftarrow w + \eta \frac{l'(y w^T x)}{q(|w^T x|)} y x$
6:    **end if**
7: **end loop**

Margin Based Offline Active Learning

1: $U \Leftarrow \{x_1, x_2, ...x_n\}$
2: $L \Leftarrow \emptyset$
3: **repeat**
4:    $x_i \Leftarrow \underset{x_i \in U}{\operatorname{argmin}} |w^T x_i|$
5:    $U \Leftarrow U \setminus \{x_i\}$
6:    Query for y_i
7:    $L \Leftarrow L \cup \{(x_i, y_i)\}$
8:    Train f to minimize $1/n \sum_{(x_i, y_i) \in L} l(y_i w^T x_i)$
9: **until** Stopping criteria met

Figure 3: Left: Online active learning algorithm for margin based loss l and selective sampling rule q. Right: Offline active learning algorithm for margin based loss function l.

sorted by $|w^t x|$. If an example has the ith smallest $|w^t x|$ and we sample m of n points at each step then

$$\Pr(\text{Querying for the label of } x_i) = \frac{\binom{n-i}{m-1}}{\binom{n}{m}}$$

With these probabilities it's possible to give a piecewise continuous objective function (continuous for regions for which the sorted order is constant) which satisfies Equation 2. However, such an objective function only describes the behavior of the algorithm within continuous regions. Online greedy algorithms can also be thought of as adaptive threshold based algorithms.

3 Offline Active Learning and Non-Convex Losses

Standard heuristic offline algorithms (Schohn and Cohn, 2000; Bordes et al., 2005; Tong and Koller, 2001) minimize a convex loss over the labeled subset of the data set and use an early stopping heuristic to avoid labeling the entire data set. These methods converge to the convex loss solution when they label the entire data set, but otherwise, it's not clear what solutions these methods find.

The right of Figure 3 gives the standard greedy offline algorithm using the min-margin heuristic (Schohn and Cohn, 2000; Bordes et al., 2005; Tong and Koller, 2001). At each step the algorithm minimizes a loss l over the labeled set. In the standard SVM approach, this loss is hinge loss. The unlabeled point selected is the point with minimum margin.

This greedy min margin heuristic was originally motivated in terms of reducing the version space of the classifier (Tong and Koller, 2001). However, when l is a non-convex loss, the heuristic has an alternative interpretation. For non-convex losses, it's often the case that if $x_i = \underset{x_i \in U}{\operatorname{argmin}} |w^T x_i|$ then

$$\max(|l'(+w^T x_i)|, |l'(-w^T x_i)|) =$$
$$\max_{x_j \in U} \max(|l'(+w^T x_j)|, |l'(-w^T x_j)|) \quad (3)$$

In other words, the x_i which minimizes the margin also maximizes $\max(|l'(+w^T x_i)|, |l'(-w^T x_i)|)$. This is true for all the non-convex losses we discuss. Then, for these losses, the greedy min-margin heuristic can be seen as selecting the unlabeled point which for worst case choice of label has the largest loss function derivative. Viewed in this way, the heuristic is very similar to working set selection heuristics used in passive algorithms (Tsang et al., 2005). These heuristics are designed to quickly minimize loss.

Equation 3 also holds for hinge loss but only trivially as $\max(|l'(+w^T x_i)|, |l'(-w^T x_i)|) = 1$ for every point (i.e. every point maximizes $\max(|l'(+w^T x_i)|, |l'(-w^T x_i)|)$). For other standard convex losses (for example squared hinge loss $l(z) = (\max(0, 1 - z))^2$), the connection doesn't hold. For convex losses, typically $\max(|l'(+w^T x_i)|, |l'(-w^T x_i)|)$ increases as $|w^T x_i|$ increases. From the perspective of minimizing loss on the full training set, the min-margin selection heuristic doesn't make as much sense for convex losses. This raises interesting questions concerning the demonstrated practical effectiveness of greedy active learning with convex losses—if the min margin heuristic is not trying to quickly minimize loss on the training set, what objective is it using?

4 Experiments

We illustrate the different losses on a synthetic data set (Figure 4). Figure 5 shows loss surfaces for each loss. The non-convex losses achieve lower zero-one loss as seen in Figure 4, with the exception of log-ramp and root-ramp functions which also have smoother error surfaces more similar to hinge loss. In fact the smoother losses do no better than guessing.

We also tested the online algorithms presented here on real world data. We use the Covtype data set (Shalev-Shwartz et al., 2007), the version of the USPS data set used by Tsang et al. (2005), and the MNIST data set and the binary version of RCV1 available from http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/. For the MNIST data set we classify the digit 8 against the other classes and the digit 0 against 1. Table 1 lists these datasets.

Loss Function	Error
Hinge	.51
Ramp	.33
Log-Ramp	.51
Root-Ramp	.51
Sigmoid	.33
Zero-One	.33

Figure 4: Left: a synthetic data set. Right: (zero-one) error rates for optimal classifiers for different loss functions.
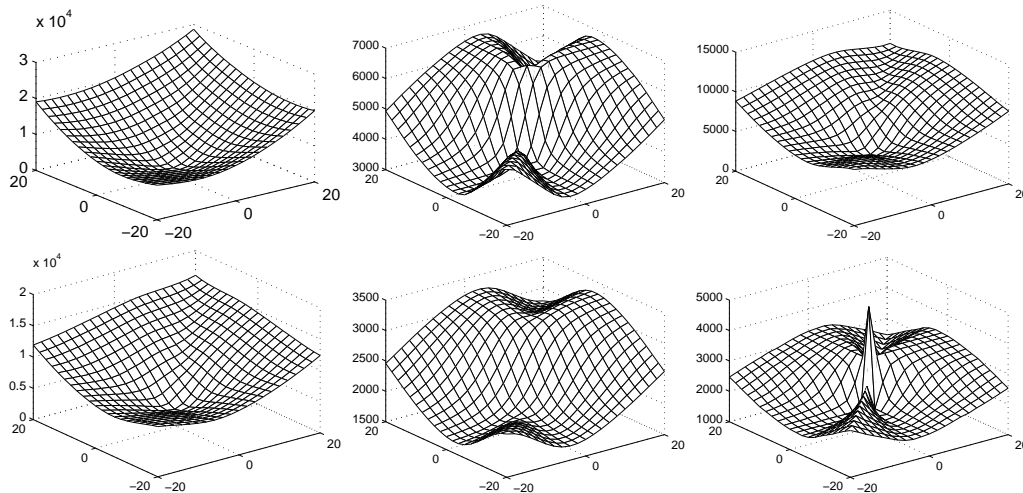

Figure 5: Loss surfaces for (clockwise from top-left) hinge loss, ramp loss, log-ramp loss, root-ramp loss, sigmoid loss, and zero-one loss on the synthetic data set as a function of w.

As an active learning baseline we perform experiments with a greedy online active learning algorithm that at each step queries for the label of the point with the smallest absolute margin out of m random samples (we use $m = 5$) and uses the hinge loss update rule. As a passive baseline we use stochastic subgradient descent on hinge loss; Shalev-Shwartz et al. (2007) show a variation of stochastic gradient descent with a different learning rate and a projection step is state of the art. For simplicity we use constant learning rates η when training each classifier and use no regularization. We leave the parameters of the loss functions untuned (we use $t = 1$ for the loss functions with thresholds and $s = .5$ for root-ramp loss). In all of the experiments, if an active learning algorithm does not query a label for 1000 steps, we terminate the algorithm early (this can be seen as indicative of a small gradient norm at that point). We initialize w to zero except where noted.

To evaluate the speed with which the active learning algorithms reduce error, we use an experiment set up similar to the one used by Monteleoni and Kaariainen (2007) who compute the average number of labels it takes an algorithm to reach a target error rate. We choose the target error rates for each data set to be the average error given by stochastic subgradient descent on hinge loss after 10000 training steps on a held out validation set (using a 66/33 percent train/validation split). We average over 50 trials and use

the fixed step size η which minimizes validation error out of all powers of 10 between 10^{-6} and 10^2. Table 1 shows these target error rates.

Then, having set the target error rates, for each data set and algorithm we chose the η values which on average reaches the target validation error using the fewest number of labels. We evaluate validation error every 100 labels during parameter fitting and again average over 50 trials and chose η between 10^{-6} and 10^2. We limit the algorithms to 10000 labels and use 10000 as a label count if an algorithm fails to reach the target on a trial. The ramp and sigmoid loss algorithms failed to reach the target for any learning rate on the MNIST-8 data set so for those data set / algorithms we arbitrarily chose the learning rates selected for hinge loss. This tuning procedure (a variation of which is used by Monteleoni and Kaariainen (2007)) is unrealistic in that it uses more labels than the final testing procedure. However, it allows for a fair comparison of the different methods—any particular heuristic could favor particular loss functions. A practical method for tuning remains an open problem.

We finally use these chosen parameters to train on the full training set (including the validation set). We compute the average number of labels it takes each algorithm to reach the target error rate on the test set. For evaluation, we average over 100 trials for each algorithm and data set and compute test error every 10 labels. We again limit each

Data Set	Training Size	Test Size	Dimensionality	Target Error Rate
Cov	522911	58101	54	.235
MNIST-01	12665	2115	780	.001
MNIST-8	60000	10000	780	.063
RCV1	20242	677399	47236	.040
USPS	266079	75383	675	.109

Table 1: Real world data sets used in our experiments

Data set	Passive Hinge	Ramp	Log	Root	Sigmoid	Greedy	R+H	S+H
Cov	1588	577	1074	1269	**460**	550	1269	1231
MNIST-01	163	140	134	126	4609	**57**	305	176
MNIST-8	3030	10000	1888	2380	10000	**1567**	1503	10000
RCV1	10000	**5324**	10000	10000	7189	9095	5243	7665
USPS	6236	6120	**1883**	3310	8409	2599	1565	2428

Table 2: The average number of labels required to reach the target error rate. R+H and S+H are ramp and sigmoid loss initialized with 1000 steps of hinge loss. We bold the best of the algorithms initializing w to zero.

algorithm to 10000 labels. Table 2 shows these results.

On all of the datasets an active learning algorithm achieves the fewest number of average labels. On the MNIST-8 data set the less smooth ramp and sigmoid loss functions fail to reach the target error rate and in fact do no better than the predicting the mode class. Log-ramp also sometimes failed to reach the target on this data set but not frequently enough to greatly increase the average label counts. This data set is imbalanced (only about 10 percent is class 8). The ramp loss and sigmoid loss algorithms also sometimes failed to reach the target error rate on the USPS data set. We think this is indicative of susceptibility to local minima. Figure 7 shows a histogram of the final error rates achieved by the ramp loss algorithm. The bimodal distribution suggests local minima. Schütze et al. (2006) observe other active learning algorithms have problems with local minima .

To further test this, we tried running the ramp loss and sigmoid loss algorithms with different initializations; we tried initializing w with a zero mean Gaussian with unit variance and initializing w with 100 and 1000 steps of stochastic gradient descent on hinge loss. We lack space for full results and discussion. However, we give results for initializing with 1000 steps of hinge loss training in Table 2 and note this greatly improved performance on MNIST-8 and USPS but hurt performance on Cov and MNIST-01.

On the RCV1 dataset, only the ramp loss, sigmoid and greedy algortihms ever reach the target error rate and only then inconsistently; on this data set the train test split is chronological, so it is not surprising the validation error rates do not match the test error rates. These two loss functions possibly reach lower error rates because they optimize tighter bounds on zero-one loss. However, the algorithms were tuned to minimize label use not asymptotic error. We have also performed some experiments tuning to minimize error and found the non-convex losses did sometimes reach lower error rates, although the differences are not often dramatic, consistent with observations of Collobert et al.
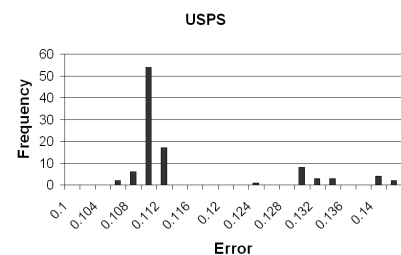

Figure 7: Error histogram for ramp loss on USPS.

(2006). Figure 6 shows error vs label plots for each data set (MNIST-01 is excluded, because convergence is very fast on that dataset). In these charts we plot error rates up to where the first run of an algorithm terminated. We also exclude plots for which the algorithm sometimes converged early with a high error rate (there are few points to plot).

We also ran experiments with the offline algorithm on the right of Figure 3 with sigmoid loss and smooth variations of hinge and ramp loss. We didn't find a significant difference between resulting algorithms, except on MNIST-8 and the USPS data set, where the non-convex loss algorithms often converged to higher error rates than the convex loss algorithm—this is consistent with the problems exhibited with the online algorithms. As in the online case, it may also be possible to avoid these problems with smarter initialization. We think the connection between the min margin heuristic and non-convex losses is still interesting.

5 Future Work

There are some remaining practical issues with applying these algorithms to real world data. As noted by Monteleoni and Kaariainen (2007), online active learning algorithms are not competitive with offline active learning algorithms, limiting their usefulness to those applications that
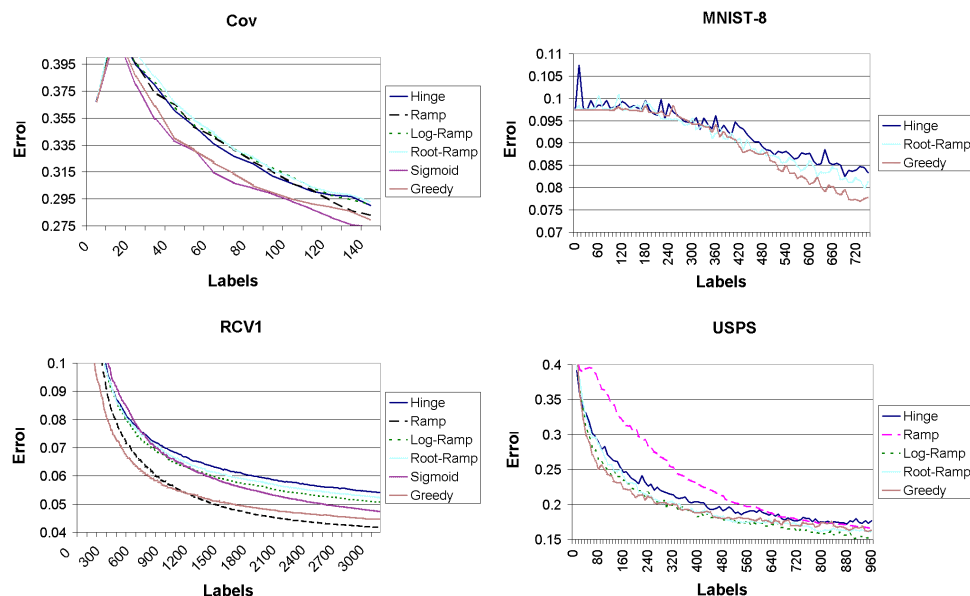
Figure 6: Test error rate vs label count for the real world data sets. We exclude plots for which the algorithm sometimes converged early at a high error rate (there are too few points to plot).

require online learning. Also, practical applications would require methods for setting parameters and choosing losses with limited label use. Finally, we would like to see if this intuition can be used to derive new theoretical results.

References

M.-F. Balcan, A. Beygelzimer, and J. Langford. Agnostic active learning. In *ICML*, 2006.

M.-F. Balcan, A. Z. Broder, and T. Zhang. Margin based active learning. In *COLT*, 2007.

A. Beygelzimer, S. Dasgupta, and J. Langford. Importance weighted active learning, 2008. URL http://www.citebase.org/abstract?id=oai:arXiv.org:0812.4952.

A. Bordes, S. Ertekin, J. Weston, and L. Bottou. Fast kernel classifiers with online and active learning. *JMLR*, 6, 2005.

N. Cesa-Bianchi, C. Gentile, and L. Zaniboni. Worst-case analysis of selective sampling for linear classification. *JMLR*, 7, 2006.

O. Chapelle. Training a support vector machine in the primal. *Neural Computation*, 19(5), 2007.

R. Collobert, F. Sinz, J. Weston, and L. Bottou. Trading convexity for scalability. In *ICML*, 2006.

S. Dasgupta, A. T. Kalai, and C. Monteleoni. Analysis of perceptron-based active learning. In *COLT*, 2005.

S. Dasgupta, D. Hsu, and C. Monteleoni. A general agnostic active learning algorithm. In *NIPS*, 2007.

S. Hanneke. Thesis proposal. 2008. Unpublished.

S. Har-Peled, D. Roth, and D. Zimak. Maximum margin coresets for active and noise tolerant learning. In *IJCAI*, 2007.

K. C. Kiwiel. A linearization algorithm for nonsmooth minimization. *Mathematics of Operations Research*, 10(2), 1985.

C. Monteleoni and M. Kaariainen. Practical online active learning for classification. In *CVPR*, 2007.

F. Perez-Cruz, A. Navia-Vazquez, A. R. Figueiras-Vidal, and A. Artes-Rodriguez. Empirical risk minimization for support vector classifiers. *IEEE Transactions on Neural Networks*, 14 (2), 2003.

G. Schohn and D. Cohn. Less is more: Active learning with support vector machines. In *ICML*, 2000.

H. Schütze, E. Velipasaoglu, and J. O. Pedersen. Performance thresholding in practical text classification. In *CIKM*, 2006.

S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal Estimated sub-GrAdient SOlver for SVM. In *ICML*, 2007.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *JMLR*, 2, 2001.

I. W. Tsang, J. T. Kwok, and P.-M. Cheung. Core vector machines: Fast svm training on very large data sets. *JMLR*, 6, 2005.