Maximum Margin Coresets for Active and Noise Tolerant Learning

Sariel Har-Peled[*] and **Dan Roth**[†]
Department of Computer Science
University of Illinois at Urbana-Champaign
{**sariel, danr**}**@uiuc.edu**

Dav Zimak[†‡]
Yahoo! Inc.
Santa Clara, CA
davzimak@yahoo-inc.com

Abstract

We study the problem of learning large margin half-spaces in various settings using coresets and show that coresets are a widely applicable tool for large margin learning. A large margin coreset is a subset of the input data sufficient for approximating the true maximum margin solution. In this work, we provide a direct algorithm and analysis for constructing large margin coresets[1]. We show various applications including a novel coreset based analysis of large margin active learning and a polynomial time (in the number of input data and the amount of noise) algorithm for agnostic learning in the presence of outlier noise. We also highlight a simple extension to multi-class classification problems and structured output learning.

1 Introduction

Large margin techniques are the basis for both practical algorithms and theoretic analysis in machine learning. Algorithmically, the most notable example is the support vector machine (SVM) [Vap95] that finds a maximum-margin separation of a given data set. The SVM has proven very successful in practice and theoretically, a large margin separation implies good generalization performance [KS94].

The SVM has a simple representation and a straightforward implementation — find the set of *support vectors* that uniquely define the maximum margin separation. Amazingly, this approach simultaneously allows the classifier to be represented with a (potentially small) subset of the input data and, through the use of kernel functions, to utilize an arbitrarily powerful hypothesis space. If a small support set can be found, then one can guarantee high performance on unseen data when using a hypothesis class with unbounded complexity. Unfortunately, there is no guarantee that the size of the support set will be small. Additionally, the running time of the SVM algorithm to find an exact solution to the large margin problem is $O(m^3)$ time and $O(m^2)$ space using m examples and is infeasible for large datasets.

Most practical algorithms, such as chunking [Vap82], decomposition [OFG97], and sequential minimal optimization (SMO) [Pla98], reduce the problem to manageable sub-tasks are heuristic solutions which may converge slowly. Furthermore, the running time remains crucially dependent on the number of support vectors in the solution. Alternatively, recent work has focused on on-line approaches to approximate the maximum-margin algorithms [Kow00; Gen01; LL02]. On-line algorithms are iterative solutions that add examples to the large margin solution based on various conditions – all related to the relative margin of the example under consideration. As a result, they can bound the number of examples necessary to guarantee a large margin separation.

In this paper we relax the requirement that we find the unique maximum margin separation. Specifically, we find an approximate maximum margin separation – a hyperplane that separates *all* of the input data with margin larger than $(1 - \epsilon)\rho^*$, where ρ^* is best achievable. We use the *coreset* method first described in [BC03] and extended to the maximum margin setting in [TKC05]. A *coreset* for a maximum margin separating hyperplane is a subset, $C \in D$ of examples such that the maximum margin hyperplane on C is an approximate maximum margin separating hyperplane on D. In some sense, it captures all of the necessary information for the approximation just as the set of support vectors does for the true maximum margin separating hyperplane.

This paper studies the running time of a simple coreset algorithm for binary and structured-output classification and the use of the coreset as an analysis tool for active learning and noise-tolerant learning in the agnostic setting. In previous work, the coreset was constructed as a reduction to finding a coreset for a different problem – the minimum enclosing ball [BC03]. In Section 3, we show a direct algorithm for finding a coreset of size at most $|C| = O((R/\rho_*)^2/\epsilon)$ in time $O(nd|C| + |C|T(|C|))$ where R and ρ_* measure the size of the example set and the quality of the maximum-margin classifier and $T(|C|)$ is the time to run an SVM black-box on $|C|$ examples. This improves previous bounds by a factor $1/\epsilon$ and

[1]Throughout the paper, we often present only proof sketches due to lack of space. The full proofs appear in Technical Report UIUCDCS-R-2006-2784 available from the University of Illinois Computer Science Department.

provides an explicit running time to the algorithm.

In Section 4 we analyze one of the most effective active learning algorithms based on the maximum-margin principal [TK02] and give a running time and a $(1 - \epsilon)$-approximation guarantee. We show that in time $O(\frac{d|C|}{\epsilon_e}\left(\ln|C| + \ln\frac{1}{\delta}\right) + |C|T(|C|))$ one can compute a coreset C of size *at most* $|C| = O((R/\rho_*)^2/\epsilon^2)$ such that with high confidence, $1 - \delta$, the classifier produced will have large margin and small, ϵ_e, error. In Section 5, we analyze learning with outlier noise. Roughly speaking, a set of outliers is a small subset of the input data such that, if removed would yield the correct maximum-margin classifier. Thus if we assume there are k outliers, the best maximum margin classifier is well-defined. We show a polynomial time algorithm for learning the maximum margin separation in this setting. Finally, in Section 6 we highlight an important connection between the coreset algorithm and recent work for learning SVM for structured output [THJA04]. Indeed, we can view SVM$^{\text{struct}}$ as a coreset algorithm.

2 Preliminaries

We are given $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_M, y_M)\}$, a labeled training set of cardinality M drawn from some distribution $\mathcal{D}_{\mathcal{X}, \mathcal{Y}}$, where $\mathbf{x}_m \in \mathcal{X}$ are the examples in a inner-product space and $y_m \in \mathcal{Y}$ are labels. For most of the paper, we assume a real valued input, $\mathcal{X} = \mathbb{R}^d$, and binary output, $\mathcal{Y} = \{-1, 1\}$. However, in Section 6 we note an important extension to the structured output domain.

In this paper, we use the maximum margin principle to discover a hypothesis $h \in \mathcal{H}$ represented by a halfspace $h(\mathbf{x}) = \operatorname{argmax}_{y \in \{-1,1\}} y(\mathbf{w} \cdot \mathbf{x})$, where $\mathbf{w} \in \mathbb{R}^d$. The *binary margin* (or *geometric margin*), $\rho(\mathbf{w}, \mathbf{x}, y) = y(\mathbf{w} \cdot \mathbf{x})$, for an example, $\mathbf{x}$, is defined as the distance from the example to the discriminating hyperplane $\mathbf{w} \cdot \mathbf{x} = 0$. Notice that a negative margin is indicative of a misclassified example. The margin of hypothesis h is $\rho(h, D) = \min_{(\mathbf{x}, y) \in D} \rho(\mathbf{w}, \mathbf{x}, y)$.

Therefore, given a sample, D, the maximum margin hypothesis (hyperplane) is

$$\mathcal{L}(D) = \operatorname*{argmax}_{\mathbf{w} \in \mathbb{R}^D, ||\mathbf{w}||=1} \min_{(\mathbf{x}, y) \in D} y(\mathbf{w} \cdot \mathbf{x})$$

is the uniform length hyperplane with maximum margin over the data.

Definition 2.1 ($(1 - \epsilon)$-Approximation) A hypothesis h is a $(1 - \epsilon)$-approximation to the optimal hypothesis h^* if $\rho(h, D) \geq (1 - \epsilon)\rho(h^*, D)$.

Definition 2.2 (Maximum margin coreset) A maximum margin coreset is a set of examples $C = C(\epsilon, \rho) \subset D$ such that $h = \mathcal{L}(C)$ is a $(1 - \epsilon)$-approximation to $h^* = \mathcal{L}(D)$.

3 Coreset Learning Algorithm

Large margin coresets were first introduced in [TKC05] to form the Core Vector Machine (CVM). In that work, they reduced finding a maximum margin hyperplane to finding the minimum enclosing ball of a set of points around the origin.

Algorithm CORESET SVM
    INPUT:
        $S = ((\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_m, y_m))$,
        Approximation parameter $\epsilon \in (0, 1)$
            where $S \in \left\{\mathbb{R}^d \times \{-1, 1\}\right\}^m$
    OUTPUT: A classifier $h \in \mathcal{H}$
begin
    Set $C = ((\mathbf{x}_1, y_1))$
    For $i = 1 \ldots T$
        Set $h_i = \text{SVM}(C)$
        Set $\rho_i = \rho(h_i, C)$
        $(\mathbf{x}_{\min}, y_{\min}) = \operatorname{argmin}_{(\mathbf{x}, y) \in S \setminus C} \rho(h_i, \mathbf{x}, y)$
        if $\rho(h_i, \mathbf{x}_{\min}, y_{\min}) < (1 - \epsilon)\rho_i$
            $C = C \bigcup (\mathbf{x}_{\min}, y_{\min})$
        else
            return h_i
    Return $\text{SVM}(C)$
end

Figure 1: Maximum-margin learning via coresets.

For the latter task, there exists a coreset algorithm that runs in time linear in the number of points [BC03]. In this section, we provide very similar results with a slight (factor $1/\epsilon$) improvement by providing a direct algorithm and analysis.

In Figure 1, the simplified coreset algorithm is presented for learning binary labeled data in a noise-free setting. The coreset C is built iteratively. At each step, we construct the true maximum margin classifier, $h_i = \text{SVM}(C)$ and use it to find the example with the smallest (or negative) margin. This example is then added to the coreset and the process repeats. It is possible to tell that h_i is a $(1 - \epsilon)$-approximation by observing the ratio between the margin on the coreset, $\rho(h_i, C)$, and the margin on the entire data set, $\rho(h_i, D)$. If this ratio is small enough then the margin of h_i on the entire data set is sufficiently large and the algorithm halts.

Lemma 3.1 *Let $\rho^* = \rho(\mathcal{L}(D), D)$ be the optimal margin for data set $D \in \{\mathbb{R}^d \times \{-1, 1\}\}^M$ of size M. Given a parameter, $\epsilon \in (0, 1)$, one can compute a coreset C of size $|C| = O((R/\rho^*)^2/\epsilon)$ in time $O(nd|C| + |C|T(|C|))$, where $R = \max_{(\mathbf{x}, y) \in D} ||\mathbf{x}||$.*

Proof sketch: We show using a simple geometric argument that each time an example is added to the coreset, the margin on the next integration decreases by at least a constant factor. That is,

$$\rho_{i+1} \leq \left(1 - \frac{\alpha_i^2}{8R^2}\right)\rho_i, \tag{1}$$

where $\alpha_i = \rho_i - \rho(h_i, \mathbf{x}_i)$ measures how much the added example violates the current margin guess using the current hypothesis h_i. Then, it can be shown that the decreasing sequence of margins will be smaller than $\rho_i \leq (1 + \epsilon)\rho$ after at most $\frac{64R^2}{\rho^2\epsilon}$ steps. Once the margin is small enough, we show that it quickly decreases and outputs a $(1 - \epsilon)$-approximation to the maximum margin classifier. ∎

At each step, the algorithm in Figure 1 adds the example with the smallest margin to the coreset. However, the algorithm is easily modified such that any example with margin small enough can suffice. Specifically, if we know that each example $(\mathbf{x}, y)$ added to the coreset is such that $\rho(h_i, \mathbf{x}, y) < (1-\epsilon)\rho_i$, but not necessarily the example with minimum margin, the algorithm still converges, but with a larger coreset.

Corollary 3.2 *Let $\rho_* = \rho(\mathcal{L}(D), D)$ be the optimal margin for data set $D \in \{\mathbb{R}^d \times \{-1, 1\}\}^M$ of size M. Given a parameter, $\epsilon \in (0, 1)$, one can compute a coreset C of size $|C| = O((R/\rho_*)^2/\epsilon^2)$ in time $O(nd|C| + |C|T(|C|))$, where $R = \max_{(\mathbf{x},y) \in D} ||\mathbf{x}||$.*

Proof sketch: If, rather than add the minimum margin example at each step, we add any example with small enough margin, $\rho(h_i, \mathbf{x}, y) < \rho_i\epsilon$, then we can simplify the simple geometric proof sketch for Lemma 3.1 and show that

$$\rho_{i+1} \leq \left(1 - \rho_i \frac{\epsilon^2}{8R^2}\right) \rho_i.$$

Then, similarly to Lemma 3.1, we use this decreasing sequence to prove the result. ∎

3.1 Related Work

Central to the coreset algorithms presented here, is the idea of iteratively building a *working set* of examples by carefully selecting examples to add at each step. In the online learning literature, this idea has appeared in work related to the coreset approach.

Indeed, even the perceptron algorithm can be viewed as building a working set. At each iteration, an example, $(\mathbf{x}_i, y_i)$, is added to the working set if $y_i(\mathbf{w}_i \cdot \mathbf{x}_i) < 0$. The hypothesis is a linear sum of elements in the set.

Various approximate algorithms for online learning have also been proposed. In [Kow00], Kowalczyk proposed a perceptron-like update rule, with various criteria for choosing which example to update. One of them is exactly the minimum margin approach used in coresets. After only $O(\frac{R^2}{\epsilon^2} \log \frac{R}{2})$ updates, the algorithm would converge to a $(1 - \epsilon)$-approximate classifier — a result very similar to ours, modulo $1/\epsilon$ and $\log R$ terms. At roughly the same as Kowalczyk's algorithm, two additional algorithms were proposed: the Relaxed Online Maximum Margin Algorithm (ROMMA) and the Approximate Large Margin Algorithm (ALMA$_p$). ROMMA is an online algorithm that learns a $(1 - \epsilon)$-approximate maximum margin separation. Both have similar selection criteria, and perform similarly in practice. ALMA$_p$ also provides a mistake bound of $O(\frac{R^2}{\epsilon^2\rho^2})$.

Recently a new algorithm, SVM-Perf, was introduced and implemented [Joa06] with similar $O(\frac{1}{\epsilon^2})$ bounds. SVM-Perf is presented as a cutting-plane algorithm, where at each iteration a cutting-plane is found that represents a fixed subset of the training examples. The coreset methods are a special case of the cutting-plane algorithm where each cutting plane is described by a single example. Indeed, this very clever algorithm converges very fast. Many experimental results are presented that show the fast convergence time in practice, and

we think this work helps support our claim that coreset-based algorithms can be practical.

4 Maximum Margin Active Learning

In active learning, the learner is presented with a set of unlabeled data, $U = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ and an oracle, ORACLE : $\mathcal{X} \to \{-1, 1\}$ that provides a label to any example $\mathbf{x} \in U$ consistent with a large margin hypothesis. The goal is to learn exactly this maximum margin separation using a limited number of oracle queries.

Recently, iterative algorithms for active learning SVM have been proposed [TK02; CCS00]. After presenting a slight modification of these algorithms using coresets and introducing an explicit stopping criteria, we show that the algorithm converges quickly to a $(1 - \epsilon)$-approximation of the true maximum margin hypothesis that exists given all labels (with high probability).

4.1 Coreset Active Learning Algorithm

In Figure 2, the active learning algorithm from [TK02] is adapted by adding a verification stage. The algorithm runs in iterations, where at each step, the unlabeled example that is closest to the decision boundary is added to the coreset. However, if there are no examples near the decision boundary (i.e. they are further than the current large-margin guess), we may think that all labels are classified correctly and thus the algorithm can halt. Of course, since the labels are unknown, it is possible that there are still a large number of misclassified examples. At this point the algorithm enters a verification phase, VERIFY(), where examples are sampled uniformly at random from the entire data set according to UNIFORMRANDOM() and labeled using ORACLE().

It is important to note that the algorithm presented in Figure 2 repeatedly cycles over the unlabeled dataset to find the single example closest to the decision boundary. This is easily modified to two important cases when the number of examples is very large (i.e. $m >> |C|$) or when there is an infinite stream of examples (i.e. $m = \infty$). In these cases, any example, $\mathbf{x}$ where $|\rho(h_i, \mathbf{x}, y)| < (1 - \epsilon)\rho_i$ can be added to the coreset and the algorithm can halt after enough examples are seen without making a mistake. Indeed Lemma 4.2 below applies to these more general cases.

4.2 Analysis

Algorithm 2 seeks the true maximum margin hypothesis of the data that would be found if all of the labels were known. Here, we show that with high probability $(1 - \delta)$, it finds a $(1-\epsilon_a)$-approximation to this hypothesis. Unfortunately, it is impossible to guarantee error free learning (see Section 4.3), so we must accept a small, ϵ_e, prediction error.

The analysis follows from two facts. First, there exists a coreset that can be constructed by adding examples that lie close to the decision boundary. Any example with very small margin $(< (1 - \epsilon_a)\rho_i)$ will improve the approximation irrespective of the actual label. Second, the verification stage will halt either because it has found an example with very small (i.e. negative) margin that will improve the coreset or because enough examples have been seen with no mistakes.

```
Algorithm ACTIVE CORESET SVM
    INPUT: Data U = (x₁,...,xₘ) ∈ {ℝᵈ}ᵐ
    OUTPUT: A classifier h′ : ℝᵈ → {−1, 1}
begin
    Set C = ∅
    Repeat until Halt
        Set hᵢ = SVM(C)
        Set ρᵢ = ρ(hᵢ, C)
        xₘᵢₙ = argminₓ∈U\C ρ(hᵢ, x, y)
        if minᵧ∈{−1,1} |ρ(hᵢ, xₘᵢₙ, y)| < (1 − ε)ρᵢ
            yₘᵢₙ = ORACLE(xₘᵢₙ)
            C = C ⋃(xₘᵢₙ, yₘᵢₙ)
        else
            (xᵥ, yᵥ) = VERIFY(U \ C, hᵢ)
            if xᵥ = NULL
                return hᵢ and Halt
            else
                C = C ⋃(xᵥ, yᵥ)
        else
            Return SVM(C) and Halt
end
```

(a) ACTIVE CORESET SVM

```
Algorithm VERIFY
    INPUT:
        Data U = (x₁,...,xₘ) ∈ {ℝᵈ}ᵐ
        A classifier h : ℝᵈ → {−1, 1}
    OUTPUT:
        (x, y) ∈ ℝᵈ × {−1, 1} or NULL
begin
    for i = 1 ... T do
        x = UNIFORMRANDOM(U)
        y = ORACLE(x)
        if ρ(h, x, y) < 0
            Return (x, y)
    Return NULL
end
```

(b) VERIFY

Figure 2: (a) Active learning using coresets. Abusing nota-
tion, $U \setminus C = (\mathbf{x} \in U | (\mathbf{x}, \text{ORACLE}(\mathbf{x})) \notin C)$. (b) Verify
procedure. UNIFORMRANDOM(U) returns a random exam-
ple from the unlabeled set. ORACLE($\mathbf{x}$) returns the correct
label for $\mathbf{x}$ and UNIFORMRANDOM(U) selects an example
from U uniformly at random.

In the latter case, we can apply the following lemma, adapted
from [KLPV87; Ang87] that shows learning from member-
ship queries can be used to give PAC-bounds.

Lemma 4.1 *If L is a conservative on-line algorithm with
mistake bound M and access to an example oracle
ORACLE() drawing examples i.i.d. from distribution $\mathcal{D}$, then
after at most $\frac{M}{\epsilon} \left(\ln M + \ln \frac{1}{\delta} \right)$ calls to ORACLE(), with con-
fidence $1 − \delta$, L produces a hypothesis with expected error
less than ϵ on examples drawn from $\mathcal{D}$.*

Using Lemma 4.1 and Corollary 3.2, we bound the running
time of this algorithm to converge to an approximate solution.

Lemma 4.2 *Let $D = \text{ORACLE}(U)$, be the entire labeled
data set and $\rho_* = \rho(\mathcal{L}(D), D)$ be the optimal margin for
data set D of size M. Given parameters, $\delta, \epsilon_e, \epsilon_a \in [0, 1]$,
one can compute a coreset C of size at most $O((R/\rho_*)^2/\epsilon_a^2)$
in time $O(\frac{d|C|}{\epsilon_e} \left(\ln |C| + \ln \frac{1}{\delta} \right) + |C|T(|C|))$, where $R =
\max_{(\mathbf{x}, y) \in D} ||\mathbf{x}||$. The total number of calls to ORACLE()
is less than $O(\frac{|C|}{\epsilon_e} \left(\ln |C| + \ln \frac{1}{\delta} \right))$, and T(m) is the running
time of the SVM for m examples.*

Proof: The coreset will be at most $|C| = O((R/\rho_*)^2/\epsilon_a^2)$
as a result of Corollary 3.2 by noticing that each time an ex-
ample, $(\mathbf{x}, y)$ is added to the coreset, $\rho(h_i, \mathbf{x}, y) < (1 − \epsilon)\rho_i$
— either because an unlabeled example is added in Line (1)
in Algorithm 2 (a) or because a labeled example is added
in Line (2) in Algorithm 2 (b). In the former we know that
$\rho_i − |\rho(h_i, \mathbf{x}, y)| > \epsilon\rho_i$, and in the latter we know that
$\rho(h_i, \mathbf{x}, y) < 0$.
Since we know that at most $|C|$ examples will be
added to the coreset, and at each iteration, the margin
of the current working hypothesis decreases we can use
Lemma 4.1 to bound the total number of Oracle queries in
$\frac{1}{\epsilon_e} \left(\ln |C| + \ln \frac{1}{\delta} \right)$ to ensure $1 − \delta$ confidence that the classifier
has at most ϵ_e mistakes. Therefore, the total number of calls
to ORACLE() is at most $O(|C| + |C|\frac{1}{\epsilon_e} \left(\ln |C| + \ln \frac{1}{\delta} \right)) =
O(|C|\frac{1}{\epsilon_e} \left(\ln |C| + \ln \frac{1}{\delta} \right))$. The running time follows since
SVM must be run each time an example is added to the core-
set. ∎

4.3 Related Work

Previously, the efficacy of maximum margin active learning
algorithms were explained because by choosing the example
closest to the decision boundary, the version space will be ap-
proximately halved [TK02][2]. More precisely, it was argued
that because at each iteration the version space is an intersec-
tion of half-spaces in the kernelized feature space. If we as-
sume that each example is of constant size (i.e. $||\mathbf{x}|| = 1$) then
the hypothesis with maximum margin separation is a point in
the version space at the center of the largest enclosed ball in
this polytope. Therefore, by choosing an example with small
margin, it is hoped that it comes close to bisecting the en-
closed ball and also the version space.

If one could guarantee that the version space was in-
deed halved at each iteration, then the algorithm would con-
verge quickly to the true maximum margin hypothesis [TK02;
FS97] Unfortunately, no such guarantee can be made, either
in practice or in theory, thus the "halving" argument falls
short to adequately explain the practical success of choosing
the minimum absolute margin example at each iteration.

In addition, as first presented in [Das05] a zero-error active
learning algorithm is impossible without requesting the label

---

[2]In [TK02] it is assumed that $||\mathbf{x}|| = 1$ for all examples. While
relaxing this assumption does cause a different view of the versions
space and changes the motivation, it does not affect their results. In-
deed, here, we propose an alternate justification that does not depend
on the version space.
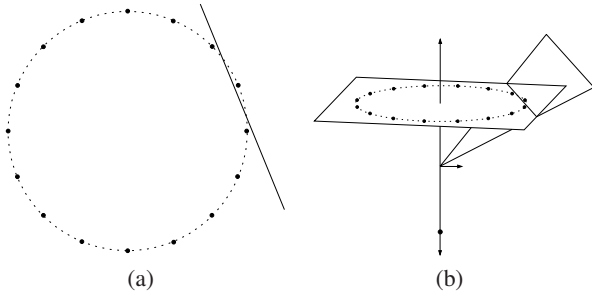
(a)   (b)

Figure 3: Impossibility of Zero-Error Approximation: (a) Active learning nightmare – all examples are positive, with a single negative. (b) Nightmare in 3D – 2 negative points, one below origin, a second on the plane of the circle.

of *all* examples. To see this, we consider a sample of examples spread at a constant interval on the surface of a circle in $\mathbb{R}^2$. See Figure 3 for an illustration. The concept represented by Figure 3(a) is one where a single example is negative and the rest are positive. The maximum margin separation thus separates a single example from the rest. Consider any algorithm that computes the maximum margin separation of any labeled subset of this data. Unless the single negative example is included in the labeled subset, then there is no hope of achieving an approximate large-margin separation that correctly classifies all examples in the data set. Therefore, if an adversary controls the oracle, by simply answering "+" to every query until the final query, the learner is forced to ask the label of every example.

5 Agnostic Learning with Outlier Noise

Learning in the presence of noise is of great interest. While there are many definitions of noise, such as attribute, label, and malicious noise, we consider a very general model, *outlier noise*. One can think of outlier noise in the following way: without the "noisy" examples, a "clean" function could be learned. Thus if the noisy examples could be identified a priori, we could learn the true maximum margin classifier. In some sense, many types of noise can be viewed as outlier noise, so the analysis presented in this section can be widely applied. We show a polynomial time algorithm for learning an approximate maximum margin hyperplane in the presence of outliers.

Definition 5.1 (Outlier Set) Consider a data set $D = \{(\mathbf{x}_i, y_i)\}_{i=m}^M$ of binary ($y \in \{-1, 1\}$) examples. For any set of outliers, $V \subseteq D$, we can consider the maximum margin hyperplane, $h^{D'} = \mathcal{L}(D')$, on the examples $D' = D \setminus V$. Then an *outlier set* of size k is a subset, V_k, of size k that achieves maximum margin on the remaining data

$$V_k = \underset{V \in D, |V| = k}{\operatorname{argmax}} \rho(\mathcal{L}(D \setminus V), D \setminus V).$$

Because a coreset exists for the "clean" data and we know that there are at most k (or a fixed fraction) outliers, we avoid exponential running time by subsampling based on the

---

```
Algorithm OUTLIER SVM
    INPUT:
        Data D = ((x_m, y_m))_1^M ∈ {R^d × {-1,1}}^M
    OUTPUT:
        A classifier h : R^d → {-1, 1}
begin
    Set R = max_{(x,y)∈D} ||x||
    For i = 0, 1, 2, ...
        ρ = R/2^i
        Set c = ⌈32 R²/(ερ²)⌉
        For each subset D_s ∈ (D choose c)
            h_s = SVM(C)
        h_max = argmax_s ρ_k(h_s, D)
        if ρ_k(h_max, D) > ρ
            Return h_max and Halt.
end
```

(a) OUTLIER SVM

```
Algorithm SIMPLE OUTLIER SVM
    INPUT:
        Data D = ((x_m, y_m))_1^M
    OUTPUT:
        A classifier h : R^d → {-1, 1}
begin
    Set R = max_{(x,y)∈D} ||x||
    For i = 0, 1, 2, ...
        Set c = 2^i
        For each subset D_s ∈ (D choose c)
            h_s = SVM(D_s)
        h_max = argmax_s ρ_k(h_s, D)
        if ρ_k(h_max, D) > √(32 R²/(cε))
            Return h_max and Halt.
end
```

(b) SIMPLE OUTLIER SVM

Figure 4: (a) Approximate maximum-margin learning with outlier noise. $\rho_k(h, D)$ returns the margin of the example in D that is smaller than the margin of all but k examples (i.e. the $k + 1$-th smallest margin). (b) Simple algorithm. An alternate description of the algorithm that simply doubles the size of the sub-sample sets at each iteration.

expected coreset size. This provides an extremely simple polynomial-time algorithm for learning with noise.

Lemma 5.2 *Let* $\rho_k = \rho(\mathcal{L}(D \setminus V_k), D \setminus V_k)$ *be the optimal margin for data set* D *of size* M *with* k *outliers. Given a parameter,* ϵ, *one can compute a separating hyperplane with margin* $(1 - \epsilon)\rho_k$ *on* $D \setminus V_k$ *in polynomial time* $O(dT(c)M^{c+1} \log M)$, *where* $c = O((R/\rho_k)^2/\epsilon)$. *and* $R = \max_{(\mathbf{x},y) \in D} ||\mathbf{x}||$.

Proof sketch: Because the clean data, $D \setminus V_k)$, contains a coreset of size $c = O((R/\rho_k)^2/\epsilon)$ from Lemma 3.1, it suffices to find this set and observe that there are at most k outliers. It

is possible to examine *all* $\binom{M}{c}$ subsets of the input data. For each one, create the maximum margin hypothesis using an SVM black-box and measure the margin obtained by removing the k examples with minimum margin (or negative margin). Then, one of these hypotheses will have the maximum margin and have at most k outliers. Finally, since the margin is unknown a-priori, we must repeat the above procedure for exponentially decreasing guesses and stop once we see only k outliers with margin as large as the guess. ∎

6 Structured Output Learning

Structured output learning is one of the most important new areas in machine learning. Structured output can be sequences, trees, rankings and general structures and are ubiquitous in important applications in areas from NLP to web search to biology. Machine learning have begun to address these problems. Here, we show a common modeling approach for structured output learning and highlight the connection to standard maximum margin learning.

Structured classifiers produce complex, structured output $\mathcal{Y} = \mathcal{Y}^1 \times \ldots \times \mathcal{Y}^L$, where $\mathcal{Y}^l \in \{1, \ldots, K\}$. It is common to write the decision rule as

$$h(\mathbf{x}) = \operatorname*{argmax}_{\mathbf{y}' \in \mathcal{Y}} \mathbf{w} \cdot \Phi(\mathbf{x}, \mathbf{y}'),$$

where $\Phi(\mathbf{x}, \mathbf{y})$ represents features of each (example, label) pair. Therefore, the maximum margin hypothesis (hyperplane) is

$$\mathcal{L}^S(D) = \operatorname*{argmax}_{\mathbf{w} \in \mathbb{R}^D, ||\mathbf{w}||=1} \min_{(\mathbf{x},\mathbf{y}) \in D} \rho(\mathbf{w}, \mathbf{x}, \mathbf{y}),$$

where

$$\rho(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \min_{\mathbf{y}' \neq \mathbf{y}} \mathbf{w} \cdot (\Phi(\mathbf{x}, \mathbf{y}) - \Phi(\mathbf{x}, \mathbf{y}'))$$

As a result, the algorithms and analysis presented in this paper extend to the structured output setting.

6.1 Related Work

Indeed, the SVM was recently extended to the structured output domain. SVM$^{\text{struct}}$ [THJA04] is an algorithm that runs in iterations, each time adding $(\mathbf{x}, \mathbf{y}, \mathbf{y}')$-triples to the working set. Indeed, they use exactly the same $\Phi(\mathbf{x}, \mathbf{y}, \mathbf{y}')$ feature vector used here. Then, the working hypothesis is updated (in the dual) by optimizing over the Lagrange multipliers, similar to the Sequential Minimal Optimization (SMO) procedure introduced by Platt [Pla98]. They also show a bound of $O(\frac{R^2}{\epsilon^2 \rho^2})$ on the size of the working set.

7 Conclusions

In this paper, we give a simple coreset algorithm with an improved bound on the coreset size. We are mainly concerned with running time analysis of various algorithms. Using coresets, we give bounds for maximum margin active learning and structured output learning. We also formulate a novel and polynomial time algorithm for learning in the agnostic (noisy) setting. Coresets are a very general tool in approximation algorithms and we have shown that they have important uses in maximum margin learning and analysis. We think that this is the tip of the iceberg, and envision that coresets will find many more applications in machine learning.

References

D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1987.

M. Badoiu and K.L. Clarkson. Smaller core-sets for balls. In *SODA*, pages 801–802, 2003.

C. Campbell, N. Cristianini, and A.J. Smola. Query learning with large margin classifiers. In *Proceedings of the Seventeeth ICML*, pages 111–118, 2000.

S. Dasgupta. Analysis of a greedy active learning strategy. In *NIPS*, volume 17, pages 337–344. MIT Press, 2005.

Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139, 1997.

C. Gentile. A new approximate maximal margin classification algorithm. *JMLR*, 2:213–242, 2001.

T. Joachims. Training linear svms in linear time. In *Proceedings of the Twelfth ACM SIGKDD*, New York, 2006.

M. Kearns, M. Li, L. Pitt, and L. Valiant. Recent results on boolean concept learning. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 337–352, 1987.

A. Kowalczyk. Maximal margin perceptron. In *Advances in Large Margin Classifiers*. MIT Press, Cambridge MA, 2000.

M. Kearns and R. Schapire. Efficient distribution-free learning of probabilistic concepts. In *Computational Learning Theory and Natural Learning Systems, Constraints and Prospect*, volume 1. The MIT Press, 1994.

Y. Li and P.M. Long. The relaxed online maximum margin algorithm. *Machine Learning*, 46(1-3):361–387, 2002.

E. Osuna, R. Freund, and F. Girosi. Improved training algorithm for support vector machines. *NNSP*, 1997.

J. Platt. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report 98-14, Microsoft Research, April 1998.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the twenty-first ICML*, page 104. ACM Press, 2004.

S. Tong and D. Koller. Support vector machine active learning with applications to text classification. *JMLR*, 2:45–66, 2002.

I.W. Tsang, J.T. Kwok, and P. Cheung. Core vector machines: Fast svm training on very large data sets. *JMLR*, 6:363–392, 2005.

V.N. Vapnik. *Estimation of Dependences Based on Empirical Data*. Springer Series in Statistics. Springer, New York, 1982.

V.N. Vapnik. *The nature of statistical learning theory*. Springer Verlag, Heidelberg, DE, 1995.