Learning Using Query by Committee, Linear Separation and Random Walks

Shai Fine Ran Gilad-Bachrach Eli Shamir *

Institute of Computer Science
The Hebrew University, Jerusalem, Israel
{fshai,ranb,shamir}@cs.huji.ac.il

Abstract. In the *Active Learning* paradigm [CAL90] the learner tries to minimize the number of labeled instances it uses in the process of learning. The reasoning comes from many real life problems where the teacher's activity is an expensive resource (e.g. text categorization, part of speech tagging).
The *Query By Committee* (QBC) [SOS92] is an Active Learning algorithm acting in the *Bayesian* model of concept learning [HKS94] i.e. it assumes that the concept to be learned is chosen according to some fixed and known distribution. When trying to apply the QBC algorithm to learn the class of linear separators one is facing the problem of implementing the mechanism of sampling hypotheses (the *Gibbs* oracle). The major problem is time-complexity, where the straightforward *Monte Carlo* method lasts exponential time.
In this paper we address the problems involved with the implementation of such mechanism. We show how to convert them to questions about sampling from convex bodies or approximating the volume of such bodies. Similar problems have been recently solved in the field of computational geometry based on random walks. These techniques enable us to device efficient implementations of the QBC algorithm.
Furthermore, we were able to suggest few improvements and corrections to the QBC algorithm. The most important is the withdrawal from the Bayes assumption when the concept classes posses a sort of symmetry property (such as linear separators). As a byproduct of our analysis we developed a geometric lemma which bounds the maximal radius of a ball contained in a convex body. We believe that this lemma has its own importance.
This paper forms a connection between random walks and certain machine learning notions such as ϵ-net and support vector machines. Working out this connection is left for future work.

* Partially supported by project I403-001.06/95 of the German-Israeli Foundation for Scientific Research [GIF].

1 introduction

The task of learning an unknown target concept out of a class of concepts by means of query the target labels at random sample of instances, has generated many studies and experimental works in recent years. In this work we re-examine the *Query By Committee* (QBC) algorithm, formulated and analyzed at [SOS92, FSST97].

QBC is an active learning algorithm [CAL90] which incorporate a relevance test for a potential label. Having access to a stream of unlabeled instances, the relevance test filters out the instances for which it assigns a low value, trying to minimize the number of labels used while learning. The motivation comes from many real life problems where the teacher's activity is an expensive resource. For example, if one would like to design a program that classifies articles into two categories ("interesting" and "non-interesting") then the program may automatically scan as many articles as possible (e.g. through the Internet). However, articles which the program needs the teacher's comment (label) - the teacher must actually read. That is a costly task. The same problem arise in many other fields such as medical diagnostics and natural language processing.

Several experimental implementations of QBC were suggested for document classification [LT97] and for part of speech tagging [DE95]. These studies give experimental evidence for the economy gained by QBC which is more substantial as the precision requirement increases. In this context the complexity of the algorithm is measured by the number of label requests directed to the teacher during the learning process.

The relevance test of QBC involves volume reduction of the version space[1]: It measures the reduction in the uncertainty about the learned target. In the specific case of learning the family of linear separators in $\Re^n$, the version space is represented by an Euclidean convex body, and we are able to use efficient volume approximating algorithms (based on suitable random walk - rather then straightforward Monte-Carlo methods). For a quantitative estimate of version space size, one is naturally led to work in a *Bayesian* framework [HKS94]: The target concept is picked from a known family according to some fixed prior distribution. Posteriors are obtained by restricting the prior to sub-families (the current version space).

Here is an outline of the QBC algorithm. It uses three oracles: The *Sample* oracle returns a random instance x, the *Label* oracle returns the label for an instance, and the *Gibbs* oracle returns a random hypothesis from the version space. The algorithm gets two parameters - accuracy (ϵ) and reliability (δ) - and works as follows:

1. Call *Sample* to get a random instance x.
2. Call *Gibbs* twice to obtain two hypotheses and generate two predictions for the label of x.

[1] The version space is the subset of all hypotheses consistent with the labels seen so far.

3. **If** the predictions are not equal
   Then call *Label* to get the correct label for x.
4. **If** *Label* was not used for the last t_k consecutive instances[2] , where k is the current number of labeled instances,
   Then call *Gibbs* once and output this last hypothesis
   Else return to the beginning of the loop (step 1).

A natural mean for tracing the progress of the learning process is the rate at which the size of the version space decreases. We adopt the notion of information gain as the measure of choice for the analysis of the learning process:

Definition 1 (Haussler et. al. [HKS94]). *Let $\mathcal{V}$ be the current version space and let $\mathcal{V}_x = \{h \in \mathcal{V} | h(x) = c(x)\}$ be the version space after the instance x had been labeled*

- *The instantaneous information gain is $\mathcal{I}(x, c(x)) = -\log \Pr_{h \in \mathcal{V}}[h \in \mathcal{V}_x]$*
- *The expected information gain of an instance x is*

$$\begin{aligned} \mathcal{G}(x|\mathcal{V}) &= \Pr_{h \in \mathcal{V}}[h(x) = 1] \cdot \mathcal{I}(x, 1) + \Pr_{h \in \mathcal{V}}[h(x) = -1] \cdot \mathcal{I}(x, -1) \\ &= \mathcal{H}(\Pr_{h \in \mathcal{V}}[h(x) = 1]) \end{aligned} \tag{1}$$

where H is the binary entropy, i.e. $\mathcal{H}(p) = -p \log p - (1-p) \log(1-p)$.

We proceed by quoting the main theoretical result about QBC, give some explanations and briefly outline the main contribution of the present paper, which show how to implement efficiently the relevance test.

Theorem 1 (Freund et. al. [FSST97]). *If a concept class C has VC-dimension $0 < d < \infty$ and the expected information gain of the queries to* Label *oracle made by QBC are uniformly lower bounded by $g > 0$ bits, then the following holds with probability greater than $1-\delta$ over the selection of the target concept, the sequence of instances(the sample), and the choices made by QBC:*

1. *The number of calls to* Sample *is $m_0 = (\frac{d}{\epsilon \delta g})^{O(1)}$.*
2. *The number of calls to* Label *is less then*[3] *$k_0 = \frac{10(d+1)}{g} \ln \frac{4m_0}{\delta}$.*
3. *The algorithm generates an ϵ-close hypothesis, i.e.*

$$\Pr_{c,h,QBC}[\Pr_x[h(x) \neq c(x)] \geq \epsilon] \leq \delta \tag{2}$$

The main theme governing the proof of this theorem is the capability to bound the number of queries made by QBC in terms of g, the lower bound for the expected information gain: If the algorithm asks to label all m instances then $\Pr_X \left[\sum_i^m \mathcal{G}(x_i|\mathcal{V}) \geq (d+1)(\log \frac{em}{d})\right] < \frac{d}{em}$, meaning the accumulated information gain grows logarithmically with m. Obviously, when filtering out instances

[2] $t_k = \frac{2\pi^2(k+1)^2}{3\epsilon\delta} \ln \frac{2\pi^2(k+1)^2}{3\delta}$ is a correction for the expression given at ([FSST97]).

[3] $k_0 = O(\log \frac{1}{\epsilon\delta})$ results in an exponential gap between the number of queries made to *Label*, comparing to "regular" algorithms.

the accumulated information gain cannot be larger. On the other hand, kg is a lower bound on the accumulated expected information gain from k labeled instances. These two observations suggest that $kg \leq (d+1)(\log \frac{em}{d})$, which results in a bound on k and implies that the gap between consecutive queried instances is expected to grow until the stop-condition will be satisfied.

Theorem 1 can be augmented to handle general class of filtering algorithms: Let L be an algorithm that filters out instances based on an internal probability assignment and previous query results. Using a stop-condition identical to the one used by the QBC algorithm and following the basic steps at the proof of Theorem 1, one may conclude similar bounds on the number of calls L makes to *Sample* and *Label* oracles.

By stating a lower bound on the expected information gain, Freund et. al. were able to identify several classes of concepts as learnable by the QBC algorithm. Among them is class of perceptions (linear separators) defined by a vector w such that for any instance x:

$$c_w(x) = \begin{cases} +1 \text{ if } x \cdot w > 0 \\ -1 \text{ otherwise} \end{cases} \tag{3}$$

This class is learnable using QBC under the restriction that the version space distribution is known to the learner and both sample space and version space distributions are almost uniform. A question which was left open is how to efficiently implement the *Gibbs* oracle and thus reduce QBC to a standard learning model (using only *Sample* and *Label* oracles). It turns out that this question falls naturally into a class of approximating problems which got much attention in recent years: How to get an efficient approximation of volume or perform a random sampling of rather complex defined and dynamically changing spaces. Moreover, unraveling the meaning of random walks employed by these approximate counting methods seems to have interesting implications in learning theory.

Let us focus on the problem of randomly selecting hypotheses from the version space and limit our discussion to the class of linear separators: There are several known algorithms for finding a linear separator (e.g. the perceptron algorithm or linear programming), but none of them suffice since we need to *randomly* select a separator in the version space. A possible straightforward solution is the use of the *Monte Carlo* mechanism: Assuming (as later we do) that the linear separators are uniformly distributed, we randomly select a point in the unit sphere[4], identifying this point as a linear separator, and check whether it is in the version space. If not, proceed with the sampling until a consistent separator will be selected. This process yields several problems, the most important of which is efficiency:

Recall that the QBC algorithm assumes a lower bound $g > 0$ for the expected information gain of queried instances. Let $p \leq 1/2$ be such that $\mathcal{H}(p) = g$. Having k labeled instances, the probability to select a consistent separator is smaller then $(1-p)^k$. This implies that the expected number of iterations the

[4] pick n normally distributed variables $\zeta_1 \ldots \zeta_n$ and normalize them by the square root of the sum of their squares.

Monte Carlo algorithm makes until it finds a desired separator is greater then $(1-p)^{-k}$. If the size of the sample the algorithm uses (number of instances) is m, and k is the number of labeled instances, then the computational complexity is $\Omega(m(1-p)^{-k})$. Plugging in the expected value for k in the QBC algorithm, i.e. $\frac{10(d+1)}{g} \ln \frac{4m}{\delta}$, the Monte Carlo implementation results in a computational complexity exponential in g, d (the VC-dimension, i.e. n in our case) and a dependence on m^2. Furthermore, since g is a lower bound on the expected information gain, the size of the version space might decrease even faster, making the task of picking an hypothesis from the version space even more time consuming. The algorithms suggested in this paper work in time polynomial in n, g and depend on $mk^{O(1)}$, i.e. they are exponentially better in terms of the VC-dimension and g and have better polynomial factors in terms of m. We also avoid the problem of rapid decrease in the size of the version space by employing a detecting condition for that situation.

1.1 Summary of Results

The problem of generating a *Gibbs* oracle is converted to the geometric problem of sampling or approximating the volumes of convex bodies (section 1.3). This observation enables us to present two efficient implementations of the QBC algorithm with linear separators. The first, which we term QBC' (section 2.1), uses volume approximation technique (section 1.4) to implement the *Gibbs* oracle. The second implementation, QBC" (section 2.2), uses sampling technique to solve the same problem. In both cases we were able to prove the efficiency of the algorithms and their ability to query for only a small amount of labels (section 4).

Even though QBC, QBC' and QBC" assumes a uniform prior on the hypotheses space we were able to show, (section 5) that they all perform well in the worst case, hence the *Bayes* assumption is redundant when learning linear separators. Furthermore, the assumption of uniform distribution on the sample space made in the proofs of QBC' and QBC", can be relaxed to include general distribution as long as these distributions are "smooth". At the end of this section we augment these results to handle general classes which poses symmetry property.

During the process of investigating the performances of QBC' and QBC", we also developed a geometric lemma (Lemma 1) which we believe has its own importance: The lemma suggest a lower bound for the maximal radius of a ball contained in a convex body in terms of the volume of that body.

1.2 Mathematical Notation

The sample space Ω, is assumed to be a subset of $\Re^n$ and therefore an instance is a vector in $\Re^n$. A linear separator is an hyperplane, defined by a vector $v \in \Re^n$ orthogonal to the hyperplane [5].

[5] Commonly, a linear separator is defined by a tuple v, b where v is a vector and b is an offset. The label a concept assigns to an instance x is defined by $\text{sign}(\langle v, x \rangle + b)$.

The concept to be learned is assumed to be chosen according to a fixed distribution $\mathcal{D}$ over $\Re^n$. We denote by x^i a queried instance and t^i the corresponding label, where $t^i \in \{-1, 1\}$. The version space $\mathcal{V}$ is defined to be $\mathcal{V} = \{v | \forall i (\langle x^i, v\rangle \cdot t^i) > 0\}$. Let $Y^i = t^i \cdot x^i$. Then a vector v is in the version space if $\forall i \ \langle Y^i, v\rangle > 0$. Using matrix notation we may further simplify notation by setting Y^i to be the i'th row of matrix A and writing $\mathcal{V} = \{v | Av > 0\}$.

1.3 Preliminary Observations

Upon receiving a new instance x, the algorithm needs to decide whether to query for a label. The probability for labeling x with $+1$ is: $P^+ = Pr_{\mathcal{D}}[v \in V^+]$ where $\mathcal{D}$ is the distribution induced on the version space $\mathcal{V}$ and $V^+ = \{v | Av > 0, \langle x, v\rangle \geq 0\}$. Similarly we define V^- and P^- which correspond to labeling x with -1. The QBC algorithm decides to query for a label only when the two hypotheses disagree on x's label and this happens with probability $2P^+P^-$. Thus P^+ and P^- are all we need in order to substitute *Gibbs* oracle and make this decision. Normalizing $||v|| \leq 1$, the version space of linear separators becomes subset of n dimensional unit ball B_n. Under the uniform distribution on B_n, the value of P^+ (and P^-) can be obtained by calculating n dimensional volume: $P^+ = \frac{\text{Vol}(V^+)}{\text{Vol}(\mathcal{V})}$. Now $\mathcal{V}$, V^+ and V^- are convex *simplexes* [6] in the n dimensional unit ball. Having P^+ and P^-, we can substitute the *Gibbs* oracle: Given a set of labeled instances and a new instance x, query for the label of x with probability $2P^+P^-$.

1.4 Few Results about Convex Bodies

In order to simulate the *Gibbs* oracle we seek efficient methods for calculating the volume of a convex body and uniformly sampling from it. Similar questions relating convex bodies have been addressed by Dyer et. al. [DFK91], Lovasz and Simonovits [LS93] and others.

Theorem 2 (The Sampling Algorithm [LS93]). *Let K be a convex body such that K contains at least $2/3$ of the volume of the unit ball (B) and at least $2/3$ of the volume of K is contained in a ball of radius m such that $1 \leq m \leq n^{3/2}$. For arbitrary $\epsilon > 0$ there exist a sampling algorithm that uses $O(n^4 m^2 \log^2(1/\epsilon)(n \log n + \log(1/\epsilon))$ operations on numbers of size $O(\log n)$ bits and returns a vector v such that for every Lebesgue measurable set L in K:* $|\Pr(v \in L) - \frac{Vol(L)}{Vol(K)}| < \epsilon$

The algorithm uses random walks in the convex body K as its method of traversal and it reaches every point of K with almost equal probability. To

In our case we assumed that $b = 0$. However, by forcing x_0 to be 1 and fixing $v_0 = b$ the two definitions converge.

[6] the term *simplex* is used to describe a conical convex set of the type $K = \{v \in \Re^n | Av > 0, ||v|| \leq 1\}$. Note that it is a nonstandard use of this term.

complete this result, Grotchel et. al. [GLS88] used the ellipsoid method to find an affine transformation T_a such that given a convex body K, $B \subseteq T_a(K) \subseteq n^{3/2}B$. Assume that the original body K is bounded in a ball of radius R and contains a ball of radius r then the algorithm finds T_a in $O(n^4(|\log r| + |\log R|))$ operations on numbers of size $O(n^2(|\log r| + |\log R|))$ bits. Before we proceed, let us elaborate on the meaning of these results for our needs: When applying the sampling algorithm to the convex body $\mathcal{V}$, we will get $v \in \mathcal{V}$. Since V^+ and V^- are both simplexes, then they are Lebesgue measurable, hence $|\Pr(v \in V^+) - P^+| < \epsilon$. Note that we are only interested in the proportions of V^+ and V^- and the use of the affine transformation T_a preserve these proportions. The sampling algorithm enables Lovasz and Simonovits to come out with an algorithm for approximating the volume of a convex body:

Theorem 3 (Volume Approximation algorithm [LS93]). *Let K be a convex body[7]. There exists a volume approximating algorithm for K: Upon receiving error parameters $\epsilon, \delta \in (0,1)$ and numbers R and r such that K contains a ball of radius r and is contained in a ball of radius R centered at the origin, the algorithm outputs a number ζ such that with probability at least $1-\delta$*

$$(1-\epsilon)\,Vol(K) \leq \zeta \leq (1+\epsilon)\,Vol(K) \tag{4}$$

The algorithm works in time polynomial in $|\log R| + |\log r|$,n,$1/\epsilon$ and $\log(1/\delta)$.

For our purposes, this algorithm can estimate the expected information gained from the next instance. It also approximates the value of P^+ (and P^-) and thus we may simulate the *Gibbs* oracle by choosing to query for a label with probability $2P^+(1-P^+)$. Both the sampling algorithm and the volume approximation algorithm require the values of R and r. Since in our case all convex bodies are contained in the unit ball B, then fixing $R = 1$ will suffice and we are left with the problem of finding r. However, it will suffice to find r such that $\frac{r^*}{4} \leq r \leq r^*$ where r^* is the maximal radius of a ball contained in K. Moreover, we will have to show that r is not too small. The main part of this proof is to show that if the volume of V^+ is not too small, then r is not too small (Lemma 1). Since we learn by reducing the volume of the version space, this lemma states that the radius decreases in a rate proportional to the learning rate.

2 Implementations of the Query By Committee Algorithms

In this section we present two implementations of the QBC algorithm: the first uses volume approximation of convex bodies, while the second uses the technique for sampling from convex bodies. Both algorithms are efficient and maintain the

[7] K is assumed to be given using a separation oracle. I.e. via an algorithm such that for a given point x it returns "true" if $x \in K$ or a separating hyperplane if $x \notin K$.

exponential gap between labeled and unlabeled instances. QBC" is especially interesting from computational complexity perspective, while the mechanism in the basis of QBC' enables the approximation of the maximal a-posteriori (MAP) hypothesis in $Poly(\log m)$ time as well as a direct access to the expected information gain from a query.

2.1 Using Volume Approximation in the QBC Algorithm (QBC')

Every instance x induces a partition of the version space $\mathcal{V}$ into two subsets V^+ and V^-. Since $\mathcal{V} = V^+ \cup V^-$ and they are disjoint, then $\text{Vol}(\mathcal{V}) = \text{Vol}(V^+) + \text{Vol}(V^-)$ and $P^+ = \frac{\text{Vol}(V^+)}{\text{Vol}(V^+)+\text{Vol}(V^-)}$. Hence, approximating these volumes results in approximation of P^+ that we can use instead of the original value. In order to use the volume approximation algorithm as a procedure, we need to bound the volumes of V^+ and V^-, i.e. find balls of radii r^+ and r^- contained in (V^+) and (V^-) respectively. If both volumes are not too small then the corresponding radii are big enough and may be calculated efficiently using convex programming. If one of the volumes is too small then we are guaranteed that the other one is not too small since $\text{Vol}(\mathcal{V})$ is not too small (lemma 7). It turns out that if one of the radii is very small then assuming that the corresponding part is empty (i.e. the complementary part is the full version space) is enough for simulating the *Gibbs* oracle (lemma 3). The QBC' algorithm follows:

Algorithm QBC'**:**

Given $\epsilon, \delta > 0$ and let k be the current number of labeled instances,

define $\delta_k = \frac{3\delta}{4\pi^2(k+1)^2}$, $\epsilon_k = \frac{\epsilon\delta_k}{96}$, $\mu_k = \frac{2\epsilon_k}{n}$, $t_k = \frac{32 \log \frac{2}{\delta_k}}{\epsilon^2\delta_k^2(1-2\delta_k)}$

1. Call *Sample* to get a random instance x.
2. Use convex programming to simultaneously calculate the values of r^+ and r^-, the radii of balls contained in V^+ and V^-.
3. **If** $min(r^+, r^-) < \mu_k(max(r^+, r^-))^n$
   Then assume that the corresponding body is empty and **goto** 6.
4. Call the volume approximation algorithm with r^+ (and r^-) to get ζ^+ (and ζ^-) such that

$$(1-\epsilon_k)\text{Vol}(V^+) \leq \zeta^+ \leq (1+\epsilon_k)\text{Vol}(V^+)$$

   with probability greater then $1-\delta_k$
5. Let $\widehat{P^+} = \frac{\zeta^+}{\zeta^+ + \zeta^-}$, with probability $2\widehat{P^+}(1-\widehat{P^+})$ call *Label* to get the correct label of x.
6. **If** *Label* was not used for the last t_k consecutive instances,
   Then call the sampling algorithm with accuracy $\epsilon = \frac{\epsilon}{2}$ to give an hypothesis and stop. **Else** return to the beginning of the loop (step 1).

With probability greater then $1-\delta$, the time complexity of QBC' is m times $Poly(n, k, \frac{1}{\epsilon}, \frac{1}{\delta})$ (for each iteration). The number of instances that the algorithm uses is polynomial in the number of instances that the QBC algorithm uses. Furthermore, the exponential gap between the number of instances and number of labeled instances still holds.

2.2 Using Sampling in the QBC Algorithm (QBC")

Another variant of QBC is the QBC" algorithm which simulate the *Gibbs* oracle by sampling, almost uniformly, two hypothesis from the version space:

Algorithm QBC''**:**

Given $\epsilon, \delta > 0$ and let k be the current number of labeled instances,
define $\delta_k = \frac{3\delta}{\pi^2(k+1)^2}$, $t_k = \frac{8 \log 4/\delta_k}{\epsilon \delta_k}$, $\epsilon_k = \frac{\epsilon \delta_k}{32}$, $\mu_k = \frac{2}{n}\epsilon_k$

1. Call *Sample* to get a random instance x.
2. Call the sampling algorithm with ϵ_k and r to get two hypotheses from the version space.
3. **If** the two hypotheses disagree on the label of x
   Then use convex programming to simultaneously calculate the values of r^+ and r^-, the radii of the balls contained in V^+ and V^-.
4. **If** $min(r^+, r^-) \geq \mu_k (max(r^+, r^-))^n$
   Then call *Label* to get the correct label and choose r^+ (or r^-), the radius of a ball contained in the new version space, to be used in step 2.
5. **If** *Label* was not used for the last t_k consecutive instances,
   Then call the sampling algorithm with accuracy $\epsilon = \frac{\delta}{2}$ to give an hypothesis and stop.
   Else return to the beginning of the loop (step 1).

QBC" is similar to QBC' but with one major difference: calculating new radius is conducted almost only when the version space changes and this happens only when querying for a label. Hence each iteration takes $O(n^7 \log^2(1/\epsilon)(n \log n + \log(1/\epsilon))$ operations and an extra $Poly(n, 1/\epsilon, 1/\delta, k)$ is needed when a label is asked. Due to the exponential gap between the number of instances and number of labeled instances, QBC" is more attractive from practical point of view.

3 Bounds on the Volume and Radius of a Simplex

We start the analysis by presenting a geometric lemma which seems to be new, giving a lower bound for the radius r of a ball contained in a simplex as a function of its volume [8] . The algorithms for estimating the volume of a convex body,

[8] Eggleston [Egg58] (also quoted at M. Berger's book *Géométrie*, 1990) gives a rather difficult proof of the lower bound $r \geq \frac{\min \text{width}(K)}{2\sqrt{n}}$ for n odd (for even n the bound is somewhat modified). The two lower bounds seem unrelated. Moreover, width(K) seems hard to approximate by sampling.

or sampling from it, work in time polynomial in $\log r$ ($R = 1$ in our case). Our lemma gives a lower bound on the size of r, hence will be useful for analyzing the time-complexity of our algorithms. For the sake of completeness, in lemma 11 we show that the bound we get is tight up to a factor of approximately $\frac{1}{\sqrt{n}}$.

Lemma 1. *Let K be a convex body contained in the n-dimensional unit ball B (assume $n > 1$). Let $v = Vol(K)$ then there exists a ball of radius r contained in K such that*

$$r \geq \frac{Vol(K)}{Vol(\partial K)} \geq \frac{Vol(K)}{Vol(\partial B)} = \frac{v\Gamma(\frac{n+2}{2})}{n\pi^{n/2}} \tag{5}$$

Proof. We shall assume that K is given by a countable set of linear inequalities and the constraint that all the points in K are within the unit ball. Let r^* be the supermum of the radii of balls contained in K, and let $r \geq r^*$. We denote by ∂K the boundary of K. Then ∂K has a derivative at all points except a set of zero measure.

We construct a set S by taking a segment of length r for each point y of ∂K such that ∂K has a derivative at y. We take the segment to be in a direction orthogonal to the derivative at y and pointing towards the inner part of K (see figure 1).

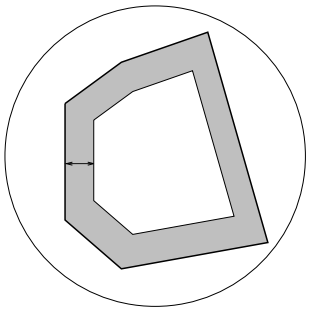

Fig. 1. An example of the process of generating the set S (the gray area) by taking orthogonal segments to ∂K (the bold line). In this case, the length r (the arrow) is too small.

The assumption that $r \geq r^*$ implies $K \subset S$ up to a set of zero measure. To show this we look at the following: Let $x \in K$, then there is $\hat{r}$ which is the maximal radius of a ball contained in K centered at x. If we look at the ball of radius $\hat{r}$ than it intersects ∂K at least at one point. At this point, denote y, the derivative of ∂K is the derivative of the boundary of the ball (the derivative exists). Hence the orthogonal segment to y of length $\hat{r}$ reaches x. Since $\hat{r} \leq r^* \leq r$ then the orthogonal segment of length r from y reaches x. The only points that might not be in S are the points in ∂K where there is no derivative, but this is a set of measure zero. Therefore

$$\mathrm{Vol}(S) \geq \mathrm{Vol}(K) = v \tag{6}$$

But, by the definition of S we can see that

$$\mathrm{Vol}(S) \leq \left| \int_{\partial K} r \right| = r\mathrm{Vol}(\partial K) \tag{7}$$

Note that $\mathrm{Vol}(\partial K)$ is $n-1$ dimensional volume. Hence, we conclude that $v \leq r\mathrm{Vol}(\partial K)$ or $r \geq \frac{v}{\mathrm{Vol}(\partial K)}$. This is true for every $r \geq r^*$ therefore

$$r^* \geq \frac{\mathrm{Vol}(K)}{\mathrm{Vol}(\partial K)} \tag{8}$$

It remains to bound the size of $\mathrm{Vol}(\partial K)$. In App. C we show that the convex body with maximal surface is the unit ball itself (this result can be obtained from the isoperimetric inequalities as well). Since the volume of n-dimensional ball is $\frac{r^n \pi^{n/2}}{\Gamma(\frac{n+2}{2})}$ the $n-1$ dimension volume of its surface is $\frac{nr^{n-1}\pi^{n/2}}{\Gamma(\frac{n+2}{2})}$. Substituting $\mathrm{Vol}(\partial K)$ with the volume of the surface of the unit ball in equation (8) we conclude that $r^* \geq \frac{v\Gamma(\frac{n+2}{2})}{n\pi^{n/2}}$. □

Corollary 1. *Let K be a simplex in the unit ball such that the maximal radius of the ball it contains is r. then*

$$\frac{r^n \pi^{n/2}}{\Gamma(\frac{n+2}{2})} \leq \mathit{Vol}(K) \leq \frac{rn\pi^{n/2}}{\Gamma(\frac{n+2}{2})} \tag{9}$$

Proof. The lower bound is obtained by taking the volume of the ball with radius r contained in K, and the upper bound is a direct consequence of Lemma 1. □

Convex programming provides the tool for efficiently estimating the radius of the ball contained in a simplex as shown in the next lemma:

Lemma 2. *Let K be a simplex contained in the unit ball defined by k inequalities. Let r^* be the maximal radius of a ball contained in K. Using the ellipsoid algorithm, it is possible to calculate a value r, such that $r^* \geq r \geq r^*/4$, in time which is $\mathit{Poly}(n, |\log r^*|, k)$.*

Proof. K is an intersection of a simplex and the unit ball. Let A be the matrix representation of the k inequalities defining K. Given a point $x \in K$, we would like to find r^*, the maximal radius of a ball centered at x and contained in K. This problem can be formulated as follows:

$$r^* = \arg\max_r \{r | \exists x \text{ s.t. } ||x|| + r \leq 1, Ax \geq r\} \tag{10}$$

This problem is a *convex programming* problem. Moreover, given an invalid pair (x, r) such that the ball of radius r centered at x is not contained in K, it is easy to find an hyperplane separating this pair from the the valid pairs. If there exists i such that $\langle A_i, x\rangle < r$ then the hyperplane $\{y | \langle A_i, y\rangle = \langle A_i, x\rangle\}$ is a separating hyperplane. Otherwise, if $||x|| + r > 1$ then $\{y | \langle y - x, x\rangle = 0\}$ is a separating hyperplane.

To conclude we need to show that the *ellipsoid algorithm* can do the job efficiently. First notice that r^* is always bounded by 1. At Lemma 7 we were able to show that r^* is not worst then exponentially small. For our purposes, finding an r such that $r \geq r^*/4$ will suffice. Note that if $r \geq r^*/4$ then the

volume of a ball with the radius r centered at x and contained in K is at least $(\frac{r^*}{4})^n \mathrm{Vol}(B)$ where $\mathrm{Vol}(B)$ is the volume of the n dimensional unit ball. Hence, it is not worst than exponentially small in $\log(r^*)$ and n. Since r^* is not too small, efficiency of the *ellipsoid algorithm* is guaranteed. □

4 Deriving the Main Results

We are now ready to present the main theorem for each variant of the QBC algorithm. However, since the difference in the analysis of the two algorithms provides no further insight we chose to describe in this sequel only the proof of QBC" which seems more adequate for practical use and defer the detailed proof of QBC' to App. B.

Theorem 4. *Let $\epsilon,\delta > 0$, let $n > 1$ and let c be a linear separator chosen uniformly from the space of linear separators. Then with probability $1-\delta$ (over the choice of c, the random sample oracle and the internal randomness of the algorithm), algorithm QBC"(as described in section 2.2)*

- *will use $m_0 = Poly(n, 1/\epsilon, 1/\delta)$ instances.*
- *will use $k_0 = O(n \log \frac{m_0}{\delta})$ labels.*
- *will work in time $m_0 \times Poly(n, k, 1/\epsilon, 1/\delta)$.*
- *will generate an ϵ-close hypothesis h, i.e.*

$$Pr_{c,h}[Pr_x[c(x) \neq h(x)] > \epsilon] < 1 - \delta \tag{11}$$

Proof. Our proof consists of three parts: the correctness of the algorithm, it's sample complexity and computational complexity. We base the proof on three lemmas which are specified at App. A.

In Lemma 4 we show that if q is the probability that QBC will ask to label an instance x and $\hat{q}$ is the probability that QBC" will ask to label the same instance then $|q - \hat{q}| \leq 4\epsilon_k$. Using this result, we show in Lemma 6 that if the algorithm didn't ask to label more then t_k consecutive instances and $\mathcal{V}$ is the version space then

$$\mathrm{Pr}_{c,h\in\mathcal{V}}[\mathrm{Pr}_x[c(x) \neq h(x)] > \epsilon] \leq \delta/2 \tag{12}$$

therefore if we stop at this point and pick an hypothesis approximately uniformly from $\mathcal{V}$ with an accuracy of $\delta/2$, the error of the algorithm is no more then δ and this completes the proof of the correctness of the algorithm.

We now turn to discuss the sample complexity of the algorithm. Following Freund et. al. [FSST97] we need to show that at any stage in the learning process there is a uniform lower bound on the expected information gain from the next query. Freund et. al. showed that for the class of linear separators such a lower bound exists, when applying the original QBC algorithm under certain restriction on the distribution of the linear separators and the sample space. In corollary 2 we show that if g is such a lower bound for the original QBC algorithm then there exists a lower bound $\hat{g}$ for the QBC" algorithm such that

$\hat{g} \geq g - 4\epsilon_k$. Since $\epsilon_k \leq \epsilon\delta$ we get a uniform lower bound $\hat{g}$ such that $\hat{g} \geq g - 4\epsilon\delta$, so for $\epsilon\delta$ sufficiently small $\hat{g} \geq g/2$. I.e. the expected information QBC" gains from each query it makes, is "almost" the same as the one gained by QBC.

Having this lower-bound on the expected information gain and using the augmented Theorem 1 , we conclude that the number of iterations QBC" will make is bounded by $\text{Poly}(n, \frac{1}{\epsilon}, \frac{1}{\delta}, \frac{1}{g})$ while the number of queries for labels it will make is less then $\text{Poly}(n, \log\frac{1}{\epsilon}, \log\frac{1}{\delta}, \frac{1}{g})$.

Finally we would like to discuss the computational complexity of the algorithm. There are two main computational tasks in each iteration QBC" makes: First, the algorithm decides whether to query on the given instance. Second, it has to compute the radii of balls contained in V^+ and V^-.

The first task is conducted by activating twice the algorithm which approximate uniform sampling from convex bodies in order to obtain two hypotheses. The complexity of this algorithm is polynomial in $|\log\epsilon_k|, n, |\log r|$ where ϵ_k is the accuracy we need, n is the dimension and r is a radius of a ball contained in the body. In Lemma 7 we bound the size of r by showing that if the algorithm uses m instances then with probability greater then $1-\delta$, in every step $r \geq \frac{\delta\left(\frac{n}{em}\right)^n}{n}$. Since we are interested in $\log r$ this bound suffice.

The second task is to calculate r^+ and r^-, which is done using convex programming as shown in Lemma 2. It will suffice to show that at least one of the radii is "big enough" since we proceed in looking for the other value for no more then another $\log\mu_k$ iterations (a polynomial value): From Lemma 7 we know that $\text{Vol}(\mathcal{V}) \geq \frac{\delta\left(\frac{n}{em}\right)^n \pi^{n/2}}{\Gamma\left(\frac{n+2}{2}\right)}$ at every step. At least one of V^+ or V^- has volume of at least $\frac{\text{Vol}(\mathcal{V})}{2}$. Using the lower bound on the volume of $\mathcal{V}$ and Lemma 1 we conclude that for the maximal radius $r \geq \frac{\delta\left(\frac{n}{em}\right)^n}{2n}$.

We conclude that with probability greater then $1-\delta$ the time-complexity of each iteration the QBC" algorithm is $\text{Poly}(n, k, \log\frac{1}{\delta}, \log\frac{1}{\epsilon})$. In the final iteration, the algorithm returns an hypothesis from the version space. Using the sampling algorithm this can be done in polynomial time. Combined with the fact that there are at most m_0 iterations the statement of the theorem follows. □

5 The Bayes and the Uniform Sample Space Distribution Assumptions - Revisited

The analysis suggested at Freund et. al. [FSST97] assumed a known distribution over the hypothesis space, i.e. learning in a Bayes setting. This assumption have also been used in this sequel for motivating and proving the results regarding QBC and it's variants. Furthermore, a uniform distribution was assumed over the sample space as well. We now turn to argue that these assumptions are in fact unnecessary: For the case of linear separators, all three algorithms (QBC, QBC' and QBC") retain their performance when the target concept is chosen by an adversary and when the sample distribution is "smooth" enough.

Let A be any of the three learning algorithms - QBC, QBC' and QBC". The input to the learning algorithm is an infinite sample vector X, a vector of random bits r, the learning parameters ϵ, δ and the label oracle c. Hence, A can be viewed as deterministic algorithm where all the random choices it makes are given by the input vector r. The algorithm A learns linear separators while assuming (falsely) a uniform distribution over the hypothesis space.

The main idea governing the revisited analysis is the symmetry embedded in the class of linear separators. For any target concept and any sequence of random choices forcing the learning algorithm to fail, we can always convert (by rotating the instances) these choices such that algorithm will fail in learning a different concept. I.e., there are no concepts which are harder to learn then the others in the class of linear separators. In the end of this section we augment this statement to handle other classes of concepts which share a sort of symmetry property.

Theorem 5 states that QBC, QBC' and QBC" learns also in the worst case, i.e. when the target concept is chosen by an adversary.

Theorem 5. *Let $A(X, r, \epsilon, \delta, c)$ be any of the QBC, QBC', QBC" algorithms. Let c be any linear separator and $\epsilon, \delta > 0$ be the learning parameters. Assuming uniform distribution over the sample space, algorithm A will generate an hypothesis h which is ϵ-close to the target concept c with probability $1 - \delta$ over the internal randomization of the algorithm.*

Proof. In the Bayes setting, $A(X, r, \epsilon, \delta, c)$ learns on average when the target concept c is chosen uniformly:

$$\Pr_{X,r,c}[A(X, r, \epsilon, \delta, c) \text{ is not } \epsilon \text{ close to } c] < \delta \tag{13}$$

It follows from (13) that there exists a target concept $\hat{c}$ such that A learns

$$\Pr_{X,r}[A(X, r, \epsilon, \delta, \hat{c}) \text{ is not } \epsilon \text{ close to } \hat{c}] < \delta \tag{14}$$

Let c be any target concept. There exists a unitary transformation T such that $\hat{c} = c(T)$. Note that A (which stand for QBC, QBC' or QBC") is invariant to unitary transformation in the sense that if A generates the hypothesis h when applied to $\langle X, r, \epsilon, \delta, c\rangle$ it will generate the hypothesis $h(T)$ when applied to $\langle TX, r, \epsilon, \delta, c(T)\rangle$ (TX stands for T applied to each instance in the sample X).

A fails to learn c on the input $\langle X, r, \epsilon, \delta, c\rangle$ iff it fails to learn $\langle TX, r, \epsilon, \delta, c(T)\rangle$. Due to the choice of $\hat{c}$ it follows that the probability for an input such that A will fail to learn $\hat{c}$ is less then δ. But the uniform distribution is invariant under unitary transformations, hence the probability that A will fail to learn c is equal to the probability that it will fail to learn $\hat{c}$ and hence bounded by δ. □

The next theorem states that not only that there is no need for the Bayes assumption, the demand for uniform distribution over the sample space can be relaxed to handle "smooth" distributions. Let us first define this class of distributions.

Definition 2. *Let U be the uniform distribution. A probability measure μ is γ* far from uniform *if for every measurable set S: $\mu(S) \leq U(S)\gamma$.*

Theorem 6. *Let $A(X, r, \epsilon, \delta, c)$ be any of the QBC, QBC', QBC" algorithms. Let μ be a γ far from uniform probability measure over the unit ball. For every target concept c and any $\epsilon, \delta > 0$, upon receiving X and r, A will generate an $(\epsilon\gamma)$-close hypothesis to c with probability $1 - \delta\gamma$.*

Proof. Fix a target concept c. In Theorem 5 it was shown that if A is applied when X is chosen using a uniform distribution and the learning parameters are ϵ, δ, then A will generate an hypothesis which is ϵ-close to c. Let h be ϵ-close to c under the uniform distribution and let S be the set of instances on which h and c disagree. Then $U(S) \leq \epsilon$. Since μ is γ far from uniform, it follows that $\mu(S) \leq \epsilon\gamma$. Therefore we may conclude that if h is ϵ-close under the uniform distribution then it is $(\epsilon\gamma)$-close under μ.

Let W be the set of all inputs to A for which it fails to generate an ϵ-close hypothesis under the uniform distribution:

$$W = \{(X, r) : A\text{'s hypothesis on the input } X, r \text{ is not } \epsilon\text{-close}\} \tag{15}$$

Since A fails with probability less then δ, then $U(W) \leq \delta$. Recall that if A does not fails to produce an ϵ-close hypothesis under the uniform distribution, it will not fail to produce $(\epsilon\gamma)$-close hypothesis under μ when applied with the same input. Therefore, the probability that A will fail to generate an $(\epsilon\gamma)$-close hypothesis under μ is bounded by $\mu(W)$. Since $U(W) \leq \delta$ and μ is γ far from uniform, it follows that $\mu(W) \leq \delta\gamma$ which proves the statement. □

Note that these results can be augmented to handle not only the case of linear separators and not only QBC and its clones. The only demand we had is some sort of symmetry property of the class of hypothesis, the learning algorithm and the probability measure. This property will be formulated using the next three definitions:

Definition 3. *An hypothesis class $\mathcal{H}$ is W* symmetric *if for each $h_1, h_2 \in \mathcal{H}$ there exists $T \in W$ such that $h_1 = h_2(T)$.*

Definition 4. *A learning algorithm A is W-*invariant *if for every $T \in W$ applying T to the training set of A is the same as applying T to the hypothesis A generate, i.e. $T\left(A\left(\langle X, c\rangle\right)\right) = A\left(T\left(\langle X, c\rangle\right)\right)$.*

Definition 5. *The* distortion *of a class of transformations W on a measure μ is bounded by γ if for any measurable set S and any $T \in W$: $\gamma\mu(S) \geq \mu(T(S))$.*

From Theorem 6 and Theorem 5 it follows that if A and μ has these properties, then A learns also in the worst case and not only on average.

6 Conclusions and Further Research

In this paper we presented a feasible way to simulate the *Gibbs* oracle picking almost uniformly distributed linear separators and thus reduce *Query by Committee* to a standard learning model. To this end, we used convex programming and formed a linkage to a class of approximation methods related to convex body sampling and volume approximation. These methods use random walks over convex bodies, on a grid which depends on the parameters ϵ and δ, in a similar fashion to PAC algorithms. It seems that such random walks could be described using ϵ-net terminology or alike. We thus suggest that this connection have further implication in the theory of learning.

We were also able to extend the cases to which QBC and its clones are applicable. For the class of linear separators, these algorithms work even in the worst case and hence the Bayes assumption can be withdrawn. Moreover, we were able to show that these algorithms may handle an extended class of "smooth" distributions over the sample space. We augmented these results for classes which poses a sort of symmetry property

Freund et. al. [FSST97] assumed the existence of the *Gibbs* oracle and essentially used the information gain only for proving convergence of the QBC algorithm (Theorem 1). The use of the volume approximation technique (i.e. QBC') provides a direct access to the instantaneous information gain. This enable us to suggest another class of algorithms, namely *Information Gain Machines*, which make use of this extra information. Combining our results with the ones presented by Shawe-Taylor and Williamson [STW97] and McAllester [McA98] may allow to obtain generalization error estimates for such information gain machines and make use of QBC' ability to produce maximum a-posteriori estimate.

The two algorithms presented have to estimate a radius of a ball contained in a convex body, which in our case is the version space $\mathcal{V}$. Finding the center of a large ball contained in the version space is also an essential task in the theory of *Support Vector Machines* (SVM) [VGS96, Vap98, Bur98]. In this case the radius is the margin of the separator. It is also clear that QBC is a filtering algorithm which seeks instances that are going to be in the support set of the current version space. This similarity implies a connection between the two paradigms. However the mapping to a high dimensional space performed in the SVM paradigm results in a sparse sample space distribution. Hence, the *Gibbs* oracle implementation suggested here will probably not work. Exploring the similarity and suggesting alternative volume approximation or sampling techniques is left for a further research.

7 Acknowledgment

We would like to thank also R. El-Yaniv, N. Linial and M. Kearns for their good advises.

References

[Bur98] C. Burges. A tutorial on support vector machines for pattern recognition. 1998. Available at `http://svm.research.bell-labs.com/SVMdoc.html`.

[CAL90] D. Cohn, L. Atlas, and R. Ladner. Training connectionist networks with queries and selective sampling. *Advanced in Neural Information Processing Systems 2*, 1990.

[DE95] I. Dagan and S. Engelson. Committee-based sampling for training probabilistic classifiers. *Proceedings of the 12th International Conference on Machine Learning*, 1995.

[DFK91] M. Dyer, A. Frieze, and R. Kannan. A random polynomial time algorithm for approximating the volume of convex bodies. *Journal of the Association for Computing Machinery*, 38, Number 1:1–17, 1991.

[Egg58] H. G. Eggleston. *Convexity*. Cambridge Univ. Press, 1958.

[FSST97] Y. Freund, H. Seung, E. Shamir, and N. Tishby. Selective sampling using the query by committee algorithm. *Macine Learning*, 28:133–168, 1997.

[GLS88] L. Grotchel, L. Lovasz, and A. Schrijver. *Geometric algorithms and Combinatorial Optimization*. Springer-Verlag, Berlin, 1988.

[HKS94] D. Haussler, M. Kearns, and R. E. Schapie. Bounds on the sample complexity of bayesian learning using information theory and the vc dimension. *Machine Learning*, 14:83–113, 1994.

[LS93] L. Lovasz and M. Simonovits. Random walks in a convex body and an improved volume algorithm. *Random Structures and Algorithms*, 4, Number 4:359–412, 1993.

[LT97] Ray Liere and Prasad Tadepalli. Active learning with committees for text categorization. In *AAAI-97,*, 1997.

[McA98] D. A. McAllester. Some pac-bayesian theorems. *Proc. of the Eleventh Annual Conference on Computational Learning Theory*, pages 230–234, 1998.

[SOS92] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committe. *Proc. of the Fith Workshop on Computational Learning Theory*, pages 287–294, 1992.

[STW97] J. Shawe-Taylor and R. C. Williamson. A pac analysis of a bayesian estimator. *Proc. of the Tenth Annual Conference on Computational Learning Theory.*, 1997.

[Vap98] V. Vapnik. *Statistical Learning Theory*. Wiley, 1998.

[VGS96] V. Vapnik, S. Golowich, and A Smola. Support vector method for function approximation, regression estimation, and signal processing. *Advances in Neural Information Processing Systems*, 1996.

A lemmas for QBC"

Lemma 3. *Let K_1, K_2 be two simplexes and let r_1, r_2 be the maximal radii of corresponding contained balls. For $i = 1, 2$ let $v_i = Vol(K_i)$. Define $p = \frac{v_1}{v_1+v_2}$, if $r_1 \leq \frac{\mu r_2^n}{n}$ then $p \leq \mu$.*

Proof. From Corollary 1 it follows that $v_1 \leq \frac{r_1 n \pi^{n/2}}{\Gamma(n+2/2)}$ and $v_2 \geq \frac{r_2^n \pi^{n/2}}{\Gamma(n+2/2)}$. Therefore

$$\frac{v_1}{v_2} \leq \frac{n r_1}{r_2^n} \tag{16}$$

Let $\mu \leq 1$. Substituting $r_1 \leq \frac{\mu r_2^n}{n}$ we conclude that

$$p = \frac{v_1}{v_1 + v_2} \leq \frac{v_1}{v_2} \leq \mu \tag{17}$$

□

Lemma 4. *For any instance x, let q be the probability that the QBC algorithm will query for a label and let $\hat{q}$ be the similar probability for QBC". Then*

$$|q - \hat{q}| \leq 4\epsilon_k \tag{18}$$

Proof. We start by analyzing the case that the radii size condition of QBC" is satisfied, i.e. $min(r^+, r^-) \geq \mu_k (max(r^+, r^-))^n$. Let $p = \Pr_{u \in \mathcal{V}}[u(x) = 1]$, then $q = 2p(1-p)$.

The QBC" algorithm samples two hypotheses, h_1 and h_2, from the version-space $\mathcal{V}$. The sampling algorithm guarantees that

$$|\Pr_{h_1 \in \mathcal{V}}[h_1 \in V^+] - \Pr[V^+]| \leq \epsilon_k \tag{19}$$

and the same holds for V^- and h_2. Denote $a = \Pr_{h_1 \in \mathcal{V}}[h_1 \in V^+]$ and $b = \Pr_{h_2 \in \mathcal{V}}[h_2 \in V^-]$. Since h_1 and h_2 are independent random variables then $\hat{q} = 2ab$. Therefore, in order to bound $|q - \hat{q}|$ we need to maximize $|2p(1-p) - 2ab|$ subject to the restrictions $|p - a| \leq \epsilon_k$ and $|(1-p) - b| \leq \epsilon_k$. It is easy to check that the maximum is achieved when $|a - p| = \epsilon_k$ and $|b - (1-p)| = \epsilon_k$ and therefore

$$|2p(1-p) - 2ab| \leq 2\epsilon_k + 2\epsilon_k^2 \leq 4\epsilon_k \tag{20}$$

We now consider the case where the radii size condition fails. Without loss of generality, assume $r^+ < \mu_k (r^-)^n$. From Lemma 3 it follows that $p < n\mu_k$ and by the definition of μ_k we get $p < 2\text{min}(\epsilon_k, 1 - \epsilon_k)$ which means that $q < 4\text{min}(\epsilon_k, 1 - \epsilon_k)(1 - 2\text{min}(\epsilon_k, 1 - \epsilon_k)) < 4\epsilon_k$. Therefore, by defining $\hat{q} = 0$ we maintain the difference $|q - \hat{q}| \leq 4\epsilon_k$. □

Lemma 5. *Let L be any instances filtering algorithm. For any instance x, let q be the probability that the QBC algorithm will query for a label and let $\hat{q}$ be the corresponding probability for L and assume $|q - \hat{q}| \leq \gamma$. Let g be a lower bound on the expected information gain for QBC. Then there exists a lower bound on the expected information gain for L, denoted by $\hat{g}$, such that*

$$\hat{g} \geq g - \gamma \tag{21}$$

Proof. Let $r(x)$ be the density function over the sample space X. Let $p(x)$ be the probability that x is labeled 1, i.e. $p(x) = \Pr_{u \in \mathcal{V}}[u(x) = 1]$, then $q(x) = 2p(x)(1 - p(x))$. Since g is a lower bound on the expected information gain for the QBC algorithm then

$$g \leq \int_x r(x) q(x) \mathcal{H}(p(x)) dx \tag{22}$$

since $|q - \hat{q}| \leq \gamma$, the expected information gain for L is bounded by

$$\int_x r(x)\hat{q}(x)\mathcal{H}(p(x))dx \geq \int_x r(x)q(x)\mathcal{H}(p(x))dx - \gamma \geq g - \gamma \tag{23}$$

taking a close enough lower bound $\hat{g}$ for the leftmost term, the statement of the lemma follows. □

Corollary 2. *Let g be a lower bound on the expected information gain of the QBC algorithm. From Lemmas 4 and 5 it follows that there exists $\hat{g} \geq g - 4\epsilon_k$ such that $\hat{g}$ is a lower bound on the expected information gain of QBC".*

Lemma 6. *Assume that after getting k labeled instances, algorithm QBC" does not query for a label in the next t_k consecutive instances. If c is a concept chosen uniformly from the version space and h is the hypothesis returned by QBC", then*

$$Pr_{c,h}[Pr_x[c(x) \neq h(x)] > \epsilon] \leq \delta/2 \tag{24}$$

Proof. We define a *bad pair* to be a pair of hypotheses from the version space that differ on more then proportion ϵ of the instances. We would like the algorithm to stop only when the queries it made form an $(\epsilon, \frac{\delta_k}{4})$-net, i.e. if two hypotheses are chosen independently from the version space, the probability that they form a bad pair is less then $\frac{\delta_k}{4}$. We will show that if the algorithm did not make a query for t_k consecutive instances, then the probability that the queries made so far do not form an $(\epsilon, \frac{\delta_k}{4})$-net, is bounded by $\delta_k/4$.

Let $W = \{(h_1, h_2) \,|\Pr_x[h_1(x) \neq h_2(x)] \geq \epsilon\}$. If $\Pr[W] \leq \delta_k/4$ then the probability that (c, h) is a bad pair is bounded by $\delta_k/4$ (when picked uniformly).We would like to bound the probability that $\Pr[W] > \delta_k/4$ when QBC" didn't query for a label for t_k at the last consecutive instances:

If $\Pr[W] > \delta_k/4$, then the probability that the QBC algorithm will query for a label is greater then $\epsilon\delta_k/4$. From Lemma 4 we conclude that the probability that the QBC" algorithm will query for a label is greater then $\epsilon\delta_k/4 - 4\epsilon_k$. Plugging in $\epsilon_k = \frac{\epsilon\delta_k}{32}$ we conclude that the probability that QBC" will query for a label is greater then $\epsilon\delta_k/8$. Therefore, the probability that it won't query for a label t_k consecutive instances is bounded by $(1 - \epsilon\delta_k/8)^{t_k}$. Using the well known relation $(1 - \varepsilon)^n \leq e^{-n\varepsilon}$ and plugging in $t_k = \frac{8\log 4/\delta_k}{\epsilon\delta_k}$ it follows that

$$(1 - \frac{\epsilon\delta_k}{8})^{t_k} \leq e^{-t_k\frac{\epsilon\delta_k}{8}} = e^{-\frac{8\epsilon\delta_k \log 4/\delta_k}{8\epsilon\delta_k}} = e^{\log(\frac{\delta_k}{4})} = \frac{\delta_k}{4} \tag{25}$$

Hence, $\Pr_{x_1,x_2,\ldots}[\Pr[W] > \delta_k/4] \leq \delta_k/4$. Thus if the algorithm stops after t_k consecutive unlabeled instances, the probability of choosing an hypothesis which forms a bad pair with the target concept is lower then $\delta_k/2$, since the probability of W being larger then $\delta_k/4$ is less then $\delta_k/4$ and if it is smaller than $\delta_k/4$ then the probability for a mistake is bounded by $\delta_k/4$. Since $\delta_k = \frac{3\delta}{\pi^2(k+1)^2}$ it follows that $\sum_k \delta_k = \delta/2$ and we get the stated result. □

Lemma 7. *Let $a > 0$ and let m be the number of calls to the* Sample *oracle that QBC" (or QBC') makes (assume $m \geq n$), then the following holds with probability greater then $1 - 2^{-a}$:*

Each intermediate version space the algorithm generates has a volume greater than $\frac{2^{-a}\left(\frac{d}{em}\right)^d \pi^{n/2}}{\Gamma\left(\frac{n+2}{2}\right)}$, and there is a ball of radius greater than $\frac{2^{-a}\left(\frac{d}{em}\right)^d}{n}$ contained in it.

Proof. The final version space, the one that is being used when the algorithm stops, is the smallest of all intermediate version spaces. Moreover, if the algorithm was not filtering out instances, but querying labels for all instances, then the final version space would have been smaller. Since we are interested in the worst case, we will assume that the algorithm did not filter out any instance.

Fix X a sample, while X^m is its first m instances. Let c be a concept chosen from Ω. X^m divides Ω, the set of all concepts, to equivalence sets, two concepts are in the same set if they give the same label to all m instances in X^m. If Ω has a VC-dimension d, we know from *Sauer's lemma* that the number of equivalence sets is bounded by $\left(\frac{em}{d}\right)^d$. Using the distribution $\Pr_c$ over Ω it follows that

$$\Pr_c\left[-\log \Pr[C] \geq a + d\log\frac{em}{d}\right] \leq 2^{-a} \tag{26}$$

where C is the equivalence set of c. We now turn to discuss the special case of linear separators and the uniform distribution over the unit ball, i.e. Ω. Note that if $\mathcal{V}$ is the version space after getting the labels for X^m, then $\mathcal{V} = C$. Therefore $\Pr[C] = \frac{\mathrm{Vol}(\mathcal{V})}{\mathrm{Vol}(B)}$ where $\mathrm{Vol}(B)$ is the volume of the n-dimensional unit ball. Using (26) we get

$$\Pr_c\left[-\log \mathrm{Vol}(\mathcal{V}) \geq a + d\log\frac{em}{d} - \log \mathrm{Vol}(B)\right] \leq 2^{-a} \tag{27}$$

Assume that $-\log \mathrm{Vol}(\mathcal{V}) \leq a + d\log\frac{em}{d} - \log \mathrm{Vol}(B)$ (this is true with probability greater then $1 - 2^{-a}$), from Lemma 1 we know that there is a ball in $\mathcal{V}$ with radius r such that $r \geq \frac{\mathrm{Vol}(\mathcal{V})\Gamma\left(\frac{n+2}{2}\right)}{n\pi^{n/2}}$. We conclude that there is a ball in $\mathcal{V}$ with radius r such that

$$r \geq \frac{2^{-a}\left(\frac{d}{em}\right)^d \mathrm{Vol}(B)\Gamma\left(\frac{n+2}{2}\right)}{n\pi^{n/2}} \tag{28}$$

□

B Theorem and Lemmas for QBC'

We begin by presenting a series of lemmas which provide the means for proving Theorem 7:

Lemma 8. *Let K_1, K_2 be two convex simplexes and let $v_i = Vol(K_i)$ be the corresponding volumes. We denote $\hat{v}_i$ the approximation of v_i and $\hat{p} = \frac{\hat{v}_1}{\hat{v}_1 + \hat{v}_2}$. If $\mu \leq 2\epsilon$ and $\epsilon \leq 3$ then*

$$|\hat{p} - p| \leq 2\epsilon \tag{29}$$

with probability greater then $1-2\delta$.

Proof. We shall term a result of the approximation algorithm a "good" result if it's value remains within the ϵ bound. We know that the probability for a bad result is $\leq \delta$. Since we activate the algorithm twice at most, then the probability for two good results is greater then $1-2\delta$. Therefore, we may assume that we got two good results.

Following Lemma 3, if we halt the computation of $\hat{v}_i$ when r_i become too small, we get a difference of μ at most (in this case $\hat{p}$ will be zero), since $\mu_k = \frac{2\epsilon_k}{n}$ the bound holds in this case. We shall proceed with the general case:

$$p - \hat{p} \leq \frac{v_1}{v_1+v_2} - \frac{v_1(1-\epsilon)}{v_1(1-\epsilon)+v_2(1+\epsilon)} = \frac{2v_1v_2\epsilon}{(v_1+v_2)^2+\epsilon(v_2^2-v_1^2)} \tag{30}$$

Fixing v_2 we need to find v_1 that will maximize the right hand of (30). From Lemma 3 and the above discussion we may assume $v_2 > 0$. If $v_1 = 0$ or $v_1 = \infty$ we get a zero gap in (30). Hence we can use the derivative to find the maximal value which turned up to be

$$v_1 = \sqrt[2]{\frac{1+\epsilon}{1+3\epsilon}}v_2 \tag{31}$$

it then follows

$$p - \hat{p} \leq \frac{2\epsilon\sqrt[2]{1+\epsilon}}{2+4\epsilon+2\epsilon^2+2\sqrt[2]{1+\epsilon}(1+3\epsilon)} \leq \epsilon\sqrt[2]{1+\epsilon} \tag{32}$$

fixing $\epsilon \leq 3$ results in $p - \hat{p} \leq 2\epsilon$.

Let us define $q = \frac{v_2}{v_2+v_1}$ and $\hat{q}$ respectively. The previous discussion implies that $q - \hat{q} \leq 2\epsilon$. Since $q = 1-p$ and $\hat{q} = 1-\hat{p}$, then $\hat{p} - p \leq 2\epsilon$ which results in $|p - \hat{p}| \leq 2\epsilon$. □

Corollary 3. *With probability greater then* $1-2\delta$

$$|p(1-p) - \hat{p}(1-\hat{p})| \leq 6\epsilon \tag{33}$$

Proof.

$$|p(1-p) - \hat{p}(1-\hat{p})| = |(p-\hat{p})(1+(p+\hat{p})| = |p-\hat{p}||1+p+\hat{p}| \tag{34}$$

since $p, \hat{p} \leq 1$ and $|p - \hat{p}| \leq 2\epsilon$ we obtain the stated result. □

Lemma 9. *Let c be the concept to be learned. For any $\epsilon, \delta > 0$, if h is chosen uniformly from the version-space then:*

$$Pr_{QBC_{I,c}}[Pr_x[c(x) \neq h(x)] > \epsilon] < \frac{\delta}{2} \tag{35}$$

Proof. Define the set of "bad" hypothesis pairs:

$$W = \{(c,h) \in \mathcal{V}^2 | Pr_x[c(x) \neq h(x)] > \epsilon\} \tag{36}$$

Denote $Pr[W] = \gamma$. If $\gamma \leq \delta_k/2$ then choosing at random $h \in \mathcal{V}$ will generate a bad hypothesis with probability γ. Assume the algorithm didn't query for label for t_k consecutive instances, after observing the k'th labeled instance. We would like to bound the probability that $\gamma \geq \delta_k/2$.

$$Pr_{c,h\in\mathcal{V}}[Pr_x[h(x) \neq c(x)] > \epsilon] = \gamma \tag{37}$$

therefore

$$Pr_{c,h,x}[h(x) \neq c(x)] > \epsilon\gamma \tag{38}$$

since $(h(x) \neq c(x))$ is a binary event, we may rewrite (38 as

$$\mathtt{E}_x[\mathtt{E}_{c,h}[h(x) \neq c(x)]] > \epsilon\gamma \tag{39}$$

Define $p = Pr_x[\mathtt{E}_{c,h}[c(x) \neq h(x)]] > \epsilon\gamma/2$. Since $0 \leq \mathtt{E}_{c,h}[c(x) \neq h(x)] \leq 1$ then

$$p + (1-p)\epsilon\gamma/2 > \epsilon\gamma \tag{40}$$

$$p > \frac{\epsilon\gamma}{2-\epsilon\gamma} \tag{41}$$

i.e.

$$Pr_x[\mathtt{E}_{c,h}[c(x) \neq h(x)] > \frac{\epsilon\gamma}{2}] > \frac{\epsilon\gamma}{2-\epsilon\gamma} > \frac{\epsilon\gamma}{2} \tag{42}$$

Given an instance $x \in X$ such that $Pr_{c,h}[h(x) \neq c(x)] \geq \epsilon\gamma/2$, the original algorithm will make a query on this instance with probability greater then $\frac{\epsilon\gamma}{2}$. In Corollary 3 we were able to show that $|p(1-p) - \hat{p}(1-\hat{p})| \leq 6\epsilon_k$ with probability greater then $1-2\delta_k$. Therefore, $QBC\prime$ will ask to label such an instance with probability greater then $(1-2\delta_k)(\epsilon\gamma/2 - 12\epsilon_k)$. Since the Probability for such $x \in X$ is greater then $\epsilon\gamma/2$, then the probability that $QBC\prime$ will ask to label the next instance is greater then

$$\frac{\epsilon\gamma}{2}(1-2\delta_k)(\epsilon\gamma/2 - 12\epsilon_k) \tag{43}$$

The probability that the algorithm would not label t_k consecutive instances is bounded by

$$(1 - \frac{\epsilon\gamma}{2}(1-2\delta_k)(\epsilon\gamma/2) - 12\epsilon_k))^{t_k} \leq \exp(-t_k\frac{\epsilon\gamma}{2}(1-2\delta_k)(\epsilon\gamma/2 - 12\epsilon_k)) \tag{44}$$

Since we assume that $\gamma \geq \delta_k/2$ and the choice we made for ϵ_k donate $12\epsilon_k = \frac{\epsilon\delta_k}{8}$, then the exponent at the left side of (44) can be upper bounded by

$$\exp -t_k\frac{\epsilon\delta_k}{4}(1-2\delta_k)\frac{\epsilon\delta_k}{8} = \exp -\log\frac{2}{\delta_k} \tag{45}$$

$$= \frac{\delta_k}{2} \tag{46}$$

By summing δ_k over k from zero to infinity we get the stated result. □

We are now ready to present the main theorem for algorithm QBC':

Theorem 7. *Let ϵ,$\delta > 0$, let $n > 1$ let c be a linear separator chosen uniformly from the space of linear separators. With probability $1 - \delta$ algorithm QBC' (as described in section 2.1) returns an hypothesis h such that*

$$Pr_{c,h}[Pr_x[c(x) \neq h(x)] > \epsilon] < 1 - \delta \tag{47}$$

using $m_0 = (n, 1/\epsilon, 1/\delta)^{O(1)}$ instances, $k_0 = O(n \log \frac{m_0}{\delta})$ labeled instances and the time complexity is m (the number of iterations) times Poly$(n, k, \frac{1}{\epsilon}, \frac{1}{\delta})$ (the complexity of each iteration).

Proof. The proof is almost the same as the one presented for the QBC" algorithm and consists of three parts: the correctness of the algorithm, it's sample complexity and computational complexity. We base the proof on the lemmas previously specified.

In Lemma 8 we show that if q is the probability that QBC will ask to label an instance x and $\hat{q}$ is the probability that QBC' will ask to label the same instance then $|q - \hat{q}| \leq 12\epsilon_k$ with probability greater then $1 - 2\delta_k$. Using this bound, we show in Lemma 9 that if the algorithm didn't ask to label more then t_k consecutive instances and $\mathcal{V}$ is the version space then

$$\Pr_{c,h \in \mathcal{V}}[\Pr_x[c(x) \neq h(x)] > \epsilon] < \delta/2 \tag{48}$$

therefore if we stop at this point and pick an hypothesis approximately uniformly from $\mathcal{V}$ with accuracy of $\delta/2$, the error of the algorithm is no more then δ and this completes the proof of the correctness of the algorithm.

Following Freund et. al. [FSST97] (cf. Theorem 1) it will suffice to show that QBC' has a lower bound on the expected information gain of the queries it makes. By Lemma 8 we know that $|q - \hat{q}| \leq 12\epsilon_k$ with probability greater then $1 - 2\delta$. Since the probability is over the choices of QBC' (and not over x), we conclude that $|q - \hat{q}| \leq 12\epsilon_k + 2\delta_k$. From Corollary 2 we conclude that if g is a lower bound on the expected information gain of the QBC algorithm, then there exists $\hat{g}$, a lower bound for the QBC' algorithm, such that $\hat{g} \geq g - 12\epsilon_k - 2\delta_k$. For the class of uniformly distributed linear separator, under certain restriction on the sample space, such a lower bound do exists when applying the original QBC algorithm (cf. [FSST97]). Since $\epsilon_k, \delta_k < \frac{\delta}{(k+1)^2}$ then for δ sufficiently small or k sufficiently large, $\hat{g} > g/2$

Having this lower-bound on the expected information gain and using the augmented Theorem 1, we conclude that the number of iterations QBC' will make is bounded by Poly$(n, k, \frac{1}{\epsilon}, \frac{1}{\delta})$ while the number of queries it will make is less then Poly$(n, k, \log \frac{1}{\epsilon}, \log \frac{1}{\delta})$.

Finally we shall briefly discuss the computational complexity, following the same discussion at Theorem 4. The algorithm computes the two radii r^+ and r^-. It follows from Lemma 7 that with high probability at least one of the radii is not too small and hence can be found in polynomial time (in the learning parameters) using convex programming (cf. Lemma 2). As for the other radius, if it takes more then polynomial time to find its value then this value is exponentially

smaller then the former radius, as measured by the parameter μ_k, and hence could be assumed to equal zero.

After computing the radii, the volume approximation algorithm is used. This algorithm takes polynomial time in the parameters $1/\epsilon_k, 1/\delta_k, |\log r^+|, |\log r^-|$. It follows that each iteration of QBC' takes polynomial time and since the number of iterations is m_0 we get the stated bound.

□

C Complementary Results on Convex Bodies

Lemma 10. *The simplex with the maximal surface contained in the unit ball is the unit ball itself.*

Proof. Assume K is a simplex other then the unit ball. We will show that there exists a simplex $\widehat{K}$ such that all it's vertices are on the unit sphere and it's surface is at least as large as the surface of K. Thus, by adding another vertex and taking the convex-hull we increase $\widehat{K}$'s surface. Adding a countable number of vertices we may approach the unit sphere as close as desired. This implies that K's surface is smaller then the surface of the unit ball, which is the unit sphere.

We start by constructing $\widehat{K}$. Let u be a vertex in the inner part of the ball. We would like to move u to the sphere while maintaining or increasing the surface size. In order to simplify the argument we shall assume working in $\Re^2$, the general case is exactly the same. We would like to drag u out of K. Denoting $\widehat{u}$ the repositioned vertex, two constraints should be considered:

1. For any vertex w adjacent to u, the size of the edge (w, u) should not decrease. This limits $\widehat{u}$ to be outside of a ball centered at w with a radius $||w - u||$.
2. Dragging u to $\widehat{u}$ and taking the convex-hull will eliminate one or more vertices of the original K. This bounds $\widehat{u}$ to be not "too far" from u (figure 2).

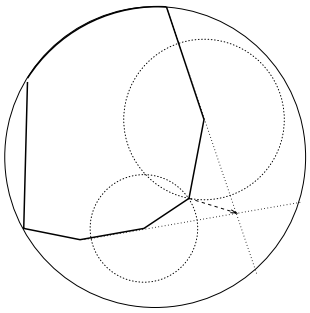

Fig. 2. The simplex is in bold face and the two types of constrains on $u\prime$ are in broken circles and lines.

Dragging u until we reach the bounds we've described (figure 3) will not decrease the size of the surface, due to the triangle inequality. Hence we may

eliminate some of the original vertices and apply the same procedure all over again. Note that each time a vertex is repositioned we either reach the unit sphere (figure 4) or eliminate at least one vertex. Since K has finite number of vertices, then after a finite number of steps we end up with $\widehat{u}$ on the unit sphere or with no vertex, i.e. the ball itself. Assuming that all vertices are on

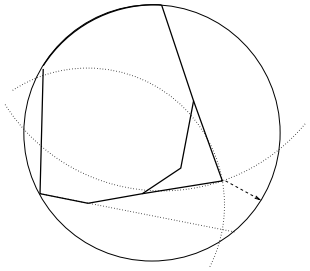

Fig. 3. The new generated by dragging a vertex until in reaches the boundary of the constrains. In broken circles and lines are the new constrains after deleting some of the vertexes.

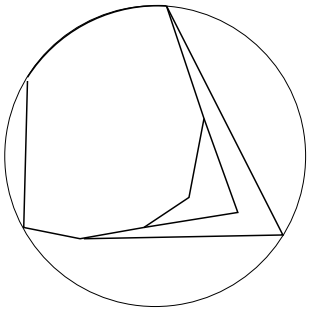

Fig. 4. The simplex, after dragging the vertex until it reaches the sphere. One can see that the size of the surface only increases in the process.

the unit sphere we further add more vertices to the sphere and take the convex-hull. Obviously the size of the surface wont decrease by these actions (figure 5), meaning that adding a countable number of vertices we transform K to the unit ball without decreasing the size of it's surface. We thus conclude that any simplex K within the unit ball has a surface which is not larger then the surface of the unit ball, i.e. the unit sphere. □

Lemma 11. *For any $\gamma > 0$, there exists a convex body $K \subseteq B_n$ such that the radius r of the maximal ball contained in it is*

$$r \leq \frac{Vol(K)\Gamma(\frac{n+1}{2})}{\pi^{\frac{n-1}{2}}}(1+\gamma) \tag{49}$$

Proof. We know from the proof of lemma 1 that the maximal radius of a ball contained in a convex body K is related to the ratio between the volume of the body and the volume of its edge. Therefore we would like to present a body with low volume and large edge. The body we will use is a disc.

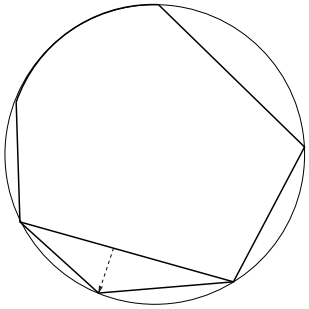

Fig. 5. Adding a vertex on the sphere after all the other vertexes are already on the sphere. Again, the size of the surface increases.

Fix $1 > r > 0$ and define K to be $K = \{x \in B_n | x_0 \in [-r/2, r/2]\}$. K is a convex body such that the maximal radius of a ball contained in it is r. The volume of K is easily seen to be bounded by

$$r\mathrm{Vol}(B_{n-1}) \geq \mathrm{Vol}(K) \geq r\mathrm{Vol}(B_{n-1})(1 - \frac{r}{2})^{\frac{n}{2}} \tag{50}$$

For small enough r, $(1 - \frac{r}{2})^{\frac{n}{2}} \geq \frac{1}{1+\gamma}$ and therefor

$$r \geq \frac{\mathrm{Vol}K}{\mathrm{Vol}(B_{n-1})}(1 + \gamma) \tag{51}$$

and by plugging the volume of the $n-1$ dimensional ball the proof is completed. □

It follows from Lemma 1 and Lemma 11 that the bounds presented are almost tight:

Corollary 4. *Let c be the maximal constant such that any convex body K in the unit ball has a ball of radius $cVol(K)$ contained in it then*

$$\frac{\Gamma\left(\frac{n+1}{2}\right)}{\pi^{\frac{n-1}{2}}} \geq c \geq \frac{\Gamma\left(\frac{n+2}{2}\right)}{n\pi^{\frac{n}{2}}} \tag{52}$$