アルゴリズム 67－8
（1999．3．15）

PAC 学習モデルを用いた 3 次元空間における半空間の共通領域の学習

岡田 清孝†，　今井 桂子††
†中央大学 大学院 理工学研究科 情報工学専攻
††中央大学 理工学部 情報工学科

概要

本稿ではラベル付けされた例から，3 次元で k 個の半空間の共通領域を学習する問題について考えていく．学習モデルには 1984 年に Valiant が学習領域上に仮定した任意の確率分布に基づいて提案した PAC 学習モデルを適用する．この問題は一般次元として扱っている場合が多く，最近では Blum と Kannan[1] が多項式時間で学習を可能とするアルゴリズムを，Vempala[7] がそのアルゴリズムをより早く改良したアルゴリズムを提案している．しかし，これらのアルゴリズムを実装する場合，学習に必要な例の個数が理論的に多項式でおさえられているものの，実際は低次元の場合でさえも非常に大きな数となってしまう．そこで，実装の面から扱う問題を 3 次元空間に限定し，計算機で扱うことができるアルゴリズムを提案することにする．その上で一般の次元を扱うことができるかどうかの評価を行っていく．

Learning an Intersection of three dimensional half-spaces in PAC model

Kiyotaka Okada†，　Keiko Imai††
†Information and System Engineering Course,Chuo University
††Department of Information and System Engineering, Chuo University

Abstract

In this paper, we consider the problem of learning an intersection of three dimensional halfspaces, over the uniform distribution on a unit ball in PAC learning model. Blum and Kannan [1] gave a polynomial algorithm for the problem in n dimensions, and Vempala[7] presented a randomaized algorithm for the same problem. However,those, and it is very hard to impliment their algorithms. Therefor, we propose a simple algorithm for learning an intersection of three dimensional halfspaces, over the uniform distribution on a unit ball in PAC learning model ,and experimental results are also shown.

1　はじめに

本稿ではラベル付けされた例から 3 次元で k 個の半空間の共通領域を学習する問題について考えていく．この問題は $k = 1$ のとき，単純な半空間 (たいてい，perceptorn と呼ばれている) を学習する問題に一致していて，機械学習の最も古い問題の 1 つである．単純な perceptron を学習することは，線形計画問題に等価であるので多項式時間で解くことができる．この問題に対する別の解法である perceptron algorithm は既に研究されているが，より複雑な概念クラスを実際のある現象モデルに必要とするケースがしばしばある．そこで，perceptron を一般化して k 個の半空間の共通領域を学習するような研究が必要となってきた．

この問題は特別な分布に対して，いくつかの成果が報告されている．1990 年に，Baum[2] は原点対称な任意の分布 $\mathcal{D}$，すなわち，任意の $x \in \mathbf{R}^n$ の点に対して，$\mathcal{D}(x) = \mathcal{D}(-x)$ (任意の点 x と原点を通るその反転した点が等しい) となるような分布上で，2 つの斉次な半空間 (homogenous halfspace) を学習するアルゴリズムを提案している．1993 年には，Blum と Kannan [1] が n 次元上の単位球上に存在する一様分布に対して，任意の定数個の半空間を学習する問題に対する多項式時間アルゴリズムを提案している．これらアルゴリズムは半空間を正確に見つけてくるわけではないが，定数個の半

空間に対して多項式時間でその近似概念を見つけてくる．そして，このアルゴリズムが見つけてきた近似概念は，限りなく学習概念に近い．1997 年には Vempala[7] が，この問題に対してランドマイズドアルゴリズムを提案し，前出の Blum と Kannan のアルゴリズムを改善している．

Blum-Kannan のアルゴリズムは多項式時間で半空間の共通領域を学習させることができるという点で注目されるべき点はあるが，実装の面から見ると必要となる膨大な例や低次元問題の再帰呼出の回数などの面で問題が残る．Vempala のアルゴリズムではランダムサンプリングを用いて前出の Blum-Kannan のアルゴリズムよりも小さいサンプル数で学習させることに成功しているが，やはり実装の面から見るとその例の個数は大きすぎる．この 2 つのアルゴリズムはいずれも高次元の問題を低次元におとして学習概念を抽出しているにもかかわらず，例の個数が大きくなってしまっている．

そこで，実装を目標にして，一般次元での半空間の共通領域の学習という問題を 3 次元に限って考察してみる．ただし，半空間を制限する超平面は原点を通るものとする．

こうして実装の面から，この問題にアプローチして一般次元への応用が可能かどうかを考察することにする．

2 半空間の共通領域の学習

本稿では PAC(probably approximately correct) 学習モデルを用いて，3 次元単位球内に一様分布をもとにした点データから，k 個の半空間の共通領域を探索する問題を扱う．学習アルゴリズムに与えられる情報は学習の対象となる概念クラスとその学習概念 (target concept) に含まれるか含まれないかを表したラベル付き点データのみで，そこから推測した仮説 (hypothesis) である半空間の共通領域をその半空間を制限している超平面 $w \cdot x \leq 0$(wは法線ベクトル) の集合として出力する．

学習概念 P は n 次元空間の l 個の半空間の共通領域とし，この半空間を制限している超平面の法線ベクトル (超平面からみて，法線ベクトルの方向は正領域 P の方向に向いている) は，k 次元の空間を作る．また，B_n を半径 1 の n 次元球とし，γ_n を B_n の体積とする．例は，この単位球内で一様分布をもとに点データとして発生させ，P に含まれる点を正例 (positive example)，含まれない点を負例 (negative example) と呼ぶ．それゆえ，半空間の共通領域である学習概念 P とその仮説 H は単位球 B_n の部分空間ということになる．この学習概念 P と仮説 H の誤差 ϵ は，領域 R の体積を $\mathrm{vol}(R)$ とすれば，$\frac{\mathrm{vol}((H \triangle P) \cap B_n)}{\mathrm{vol}(B_n)}$ で表すことができる．

ここで，学習アルゴリズムは誤差 ϵ と信頼度 δ を与えられた場合，$1-\delta$ 以上の確率で，ϵ 以下の仮説を $\frac{1}{\epsilon}, \frac{1}{\delta}$ の多項式時間で生成することができる．

3 Blum-Kannan のアルゴリズム

ここでは，Blum と Kannan が提案したアルゴリズムを示す．その準備として，アルゴリズムで用いられている “relevant subspace” と “$n-1$ 次元問題” の概念に触れておく．また，このアルゴリズムは non-homogenous な学習概念を仮定している．

3.1 relevant subspace

領域 R を $R = \{x \in B_n : w_i \cdot x \leq a_i \quad i = 1, 2, \cdots, l\}$，半空間を制限する超平面を $w_i \cdot x \leq a_i$ と仮定する．このとき超平面の法線ベクトル $w_1, \cdots, w_l$ で張られる空間を R の relevant subspace と呼び，$\mathbf{V}_{\mathrm{rel}}(R)$ で表すことにする．このとき，一次独立性を考えれば，$w_1, \cdots, w_l$ で張られている空間は $w_1, \cdots, w_k (k \leq l)$ で張ることができる．また，この $\mathbf{V}_{\mathrm{rel}}(R)$ の直交補空間を R の irrelevant subspace と呼び，$\mathbf{V}_{\mathrm{irrel}}(R)$ と表す．そして，$\mathbf{V}_{\mathrm{rel}}(R)$ のベクトルを relevant vector，$\mathbf{V}_{\mathrm{irrel}}(R)$ のベクトルを irrelevant vector と呼び，$\mathbf{V}_{\mathrm{irrel}}(R)$ の基底ベクトルを irrelevant direction とする．

3.2 $n-1$ 次元問題

ここでは n 次元の点データを $n-1$ 次元におとす手法について述べる．これにより，n 次元上の問題が $n-1$ 次元の問題に帰着される．

まず，ベクトル u に直交するスライスを次のように定義する．

$$\mathbf{slice}(m) = \{x \in B_n : m\epsilon_1 \leq u \cdot x \leq (m+1)\epsilon_1\}$$

m は整数で，ϵ_1 は各スライスの薄さを定義する変数とする．ただし，$N(k-1)$ を $k-1$ 次元での学習に必要な例の個数とすると，ϵ_1 は $\frac{312 ln^2 \epsilon_1}{\epsilon} \leq \frac{\epsilon}{6}$ をみたす．全スライスは $\frac{2}{\epsilon_1}$ 個となり，スライスの体積が $\frac{1}{12}\epsilon_1\epsilon\gamma_n$ 以上のスライスを “big スライス” と呼び，$\frac{1}{12}\epsilon_1\epsilon\gamma_n$ 以下のスライスを “small スライス” と呼ぶことにする．そうすると，この big スライスを誤差が $\frac{\epsilon}{6}$ 以下，誤り確率が $\frac{\delta\epsilon}{48n}$ 以下で再帰的に PAC 学習させることができる．small スライスはすべて負 (negative) に分類する．未知の正領域を

$$P = \{x \in B_n : a_i \cdot x \leq a_{io} \ \ i = 1, 2, \cdots, l\}$$

とする．このとき，$a_i = \frac{w_i}{|w_i|}$ (ここでの a_i は R を定義した a_i とは異なる) で，このベクトルは k 次元の relevant space $\mathbf{V}_{\mathrm{rel}}(P)$ をつくる．つまり，$\mathbf{V}_{\mathrm{rel}}(P)$ は k 本のベクトル a_i で張られる空間となっている．そして，u_1 は relevant space に u を射影したベクトルで，a_i はある実数ベクトルとす

る. a_i が λ_i に対して $\lambda_i u_1 + b_i$ と分解できるとき, このベクトル b_i は $\mathbf{V}_{\mathrm{rel}}(P)$ の要素になっていて, ベクトル u_1 に直交する.

また, 前述の λ_i と b_i を用いてもとの次元で仮説の法線ベクトルを作る. そして $c_i = \lambda_i u + b_i$ とし, 仮説を以下のように定義する.

$$P_1 = \{x \in B_n : c_i \cdot x \leq a_{io}\ , i = 0, 1, \cdots, l\}$$

ここで, 各整数 m に対して, スライス内に存在する正例の点の集合 **slice**(m) の切口の点集合を $P_1(m)$ で表し, $m \geq 0$ に対して $P_1(m) = \{x : u \cdot x = (m+1)\epsilon_1\}$, $m < -1$ に対して $P_1(m) = \{x : u \cdot x = m\epsilon_1\}$ と定義する.

これからわかるように, $P_1(m)$ は 次数 $k-1$ の relevant space を持つ $n-1$ 次元の凸集合であることがわかる. というのは, $b_1, \cdots, b_l$ が u_1 に直交する $\mathbf{V}_{\mathrm{rel}}(P)$ のベクトルの $k-1$ 次元空間を張っているからである. ここで, 次のような仮定をする. **slice**(m) 内にあるすべての例は $P_1(m)$ に対応してラベル付けがされている. この $P_1(m)$ 上の点をスライス内の点 x に対して $f(x)$ とする. そうすれば 1 つ次元の低い relevant space を再帰呼出しすることができる.

3.3 アルゴリズム

ここで, アルゴリズムに必要な例の個数を $E(k, n, \epsilon, \delta)$ とし, $E\left(k-1, n-1, \frac{\epsilon}{6}, \frac{\epsilon\delta}{48n}\right)$ を $N(k-1)$ で表すことにする.

このとき, 以下のアルゴリズムをえることができる.

Blum-Kannan のアルゴリズム

1. 少なくとも $(1 - \frac{\delta}{4})$ の確率で, 高々 ϵ_1 の誤差で, 学習概念の近似重心ベクトル u を見つける.
2. small スライスを 全て負に分類する.
3. B_n から $\frac{24}{\epsilon_1 \epsilon}\left(N(k-1) + 4\log\left(\frac{8}{\delta\epsilon_1}\right)\right)$ 個のラベルづけされた例を取ってくる.
4. 任意の big スライス中に例が $N(k-1)$ 個以下しかなければアルゴリズムを終了し, 学習を失敗とする.
5. 4. が成立しない場合, 各 big スライスに含まれている $N(k-1)$ 個の例を取り出し, $f(x)$ に x と同じラベルを貼る. そして, 誤差は高々 $\frac{\epsilon}{6}$, 誤り確率は高々 $\frac{\delta\epsilon}{48n}$ で, $k-1$ 個の relevant direction を持つ $n-1$ 次元問題を再帰的に学習させる.

3.4 必要なサンプル数

ϵ_1 以下の誤差の近似重心ベクトル u を最初に見つけるのに必要な例の個数は下の量だけ必要である.

$$O\left(\epsilon_1^{-4} \cdot poly\left(n, l, \frac{1}{\epsilon}, \log\frac{1}{\delta}\right)\right)$$

この量は次で表される.

$$O\left(N(k-1)^4 \cdot poly\left(n, l, \frac{1}{\epsilon}, \log\frac{1}{\delta}\right)\right)$$

アルゴリズムのステップ 2 で必要な例の個数は以下の通りである.

$$O\left(N(k-1)^2 \cdot poly\left(n, l, \frac{1}{\epsilon}, \log\frac{1}{\delta}\right)\right)$$

よって, k が定数なので, 上の式がたとえ k の二重指数だとしても, これは各パラメータの多項式になる.

4 Vempala のアルゴリズム

ここでは, Vempala が提案したアルゴリズムを示す. このアルゴリズムは Blum-Kannan のアルゴリズムとは違い, homogenous な学習概念を仮定している.

4.1 準備

前節までの準備及び概念の他に必要なことを示しておく. アルゴリズムに与えられる正例の集合を S として, これら正例の集合 S によって形成される錐を C とする. この錐 C に双対な空間を D_C とする. すなわち, 錐 C の超平面の法線ベクトルの集合のことである.

定義 1 R^k の凸錐 (convex cone)K は $K \subseteq K'$ で, どのような点 $x \in K' \cap B_n$ に対しても, ベクトル x とベクトル y のなす角度が, 高々 ϵ であるような点 $y \in K \cap B_n$ が存在するとき, K は K' に ϵ-*enclose* されているという.

4.2 Vempala のアルゴリズム

$\alpha = \frac{\epsilon_2}{\sqrt{k \log n}}$ とし, 各 $w \in D_P$ と $\hat{w} \in \hat{D_C}$ が ϵ_2(非常に小さな値) 以上の射影角を持つことはないと仮定する. そして, $\epsilon_3 = \frac{\epsilon}{2k} - \epsilon_2$ を満たすとき, 以下のアルゴリズムを定義する.

Vempala のアルゴリズム

1. 正例の集合から仮説の法線ベクトルの集合で原点を通る錘 D_C を作る.
2. 自分自身への $D_C \cap B_n$ の射影距離が最小になるようなベクトル $\{x_1, x_2, \ldots, x_{n-k}\}$ を見つけて, 部分空間を近似する.
3. $\{x_1, x_2, \ldots, x_{n-k}\}$ で作られる直交部分空間に D_C を射影し, $\hat{D_C}$ とする.
4. $\hat{D_C}$ を ϵ_3-*enclose* する格子点の箱を構成する.
5. 新たに取り出した $\Omega(nl)$ 個の例の集合を S_1 とする. このとき, S_1 からなるべく多くの負例を切り取る超平面の法線ベクトルを $\{x_1, x_2, \ldots, x_{n-k}\}$ の中から m 本選び出力する.
6. 5 で出力した法線ベクトルをもとの n 次元に戻して, 仮説を構成する超平面の集合として出力する.

4.3 必要なサンプル数

定理 1 パラメータ ϵ と δ を与えたとき, これまで説明してきた PAC 学習アルゴリズムが, 必要とする例の個数とその実行時間の上界は

$$O\left(poly(n)lk^k \left(\frac{p}{\epsilon}\right)^k \log\frac{1}{\delta}\right)$$

である. ただし, 学習概念は l 個の半空間の共通領域で表され, この法線は k 次元部分空間を作り出す.

5 アルゴリズム

ここでは, 問題空間を 3 次元, 学習概念は homogenous として Blum-Kannan 及び Vempala のアルゴリズムよりも単純なアルゴリズムを提案する.

入力は, 誤差 $\epsilon(0 < \epsilon < 1)$, 信頼度 $\delta(0 < \delta < 1)$, 学習に必要なサンプル数と学習概念を作る超平面の式 $(w \cdot x \leq 0)$ の集合とし, 出力は仮説を作る超平面の式 $(w \cdot x \leq 0)$ の集合とする. このとき, 以下のアルゴリズムを提案する.

アルゴリズム

1. 学習概念 P の重心ベクトルを近似するベクトル u を正例から計算する.
2. u の方向に原点から距離 1 で, u を法線ベクトルとするような平面に正例を射影する.
3. 射影した平面で凸包を構成する.
4. 凸包上の点から幾何概念を学習し, その結果を幾何概念の頂点の集合として出力する.
5. 4 での出力結果をもとの単位球内に戻す.
6. 学習概念と仮説の誤差を計算し, 誤差範囲内におさまらなければ学習を失敗とし, そうでなければ成功とする.

近似重心ベクトルの計算 ここで重心ベクトルを計算するのは, 学習空間を低次元空間におとすときの方向を決めるために必要とするベクトルであるからである. 低次元におとした問題を学習させることを考えると, 低次元の問題がもとの学習概念の性質を正確に保持していることが必要となる. これを満たすベクトルは学習概念の中に存在するベクトルが最適である. そこで, このベクトルを重心ベクトルとするのがもっともよいと思われる. よって, 最初に重心ベクトルを近似するベクトルを計算する. 計算方法は単純に, 正例の点の平均を求めればよい. 正例の平均を取れば必ず学習概念の中に, その平均ベクトルが存在することは自明である.

学習空間を低次元におとす ここでは 3 次元での問題を 2 次元の問題に変換する. 各正例の点を原点から u の方向に距離 1 で u を法線ベクトルとするような平面に射影する. こうすることで学習概念の性質を変えずに問題を低次元におとすことができる.

学習に必要な例の抽出 2 次元での学習に必要な点データは凸包上の点さえあればよいので, ここでは凸包を構成することにする. 凸包を構成するアルゴリズムは非常に有名で $O(n \log n)$ のアルゴリズム等が提案されている. そこで, どのアルゴリズムを採用するかはデータ構造等により変わってくると思われる. 本論では "Graham Scan" のアルゴリズムを用いることにしている. 理由は, 今回実装したプログラムが, データのやり取りを殆ど全てファイルを媒介に行っているためである. この場合, 各関数間での大きなデータの受け渡しがファイルであるため "Quick Hull" のような再帰呼び出しのあるアルゴリズムは実装上の不都合が生じる. よって, 再帰呼び出しのないアルゴリズム採用したのである. また, データの受け渡しがメモリ内で行われているならば, "Quick Hull" のような再帰呼び出しのあるアルゴリズムでもかまわない.

幾何概念の学習　ここでは, 3 次元の学習問題が 2 次元の学習問題になっている. 2 次元において任意の幾何図形を学習させることは難しい. 例えば軸に平行な正方形や長方形, 各内角が全て等しい幾何図形のような特徴的な図形は学習可能で, その方法も単純である [6]. しかし, これら以外のいびつな図形を学習させることは難しい.

そこで, このような学習にむかない図形のうち少しでも多くの概念を学習可能とするために以下のアルゴリズムを提案する. ここでは, 凸包を構成する点集合を辺を表すベクトルの列とみなし (図 1), 現在, 指しているベクトルを pt とする. また, このアルゴリズムに与えられる点集合は凸包上の点で凸包上を順番に並んでいるものとする.

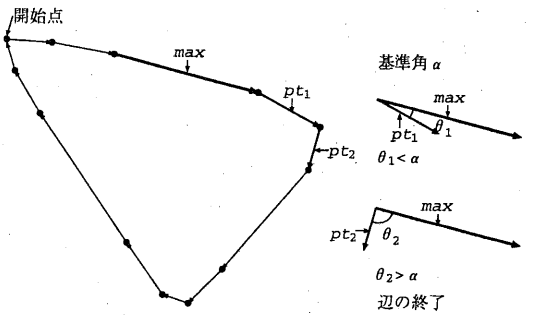

図 1. 仮説の辺の端点の探索

仮説を抽出するアルゴリズム

1. 凸包上の点集合の中で x 座標がもっとも小さい点を開始点にする.
2. 開始点からスタートして最も長いベクトルを max とする. この max は変更があるたびに逐次, 更新する. この max と pt のなす角がθ 以下のあいだ 3 を繰り返す. もし, θ 以上になったら 4 を行う.
3. pt の指すベクトルの長さを計算して, max の指すベクトルの長さと比較する. もし, pt が指すベクトルの方が長ければ, max が指すベクトルを pt が指すベクトルに更新する. そうでないならば, max はそのままにして, pt が指すベクトルを 1 つ進める.
4. 一つの辺が終了したとみなし, 仮説の辺を構成するリストに max が指すベクトルを追加する. pt が指すベクトルを max として, 新しい探索を再び開始する.

上記のアルゴリズムで取り出した辺が多すぎる場合は, 別途, 削除を行なう.

6　計算時間

アルゴリズム中の凸包を構成する部分は $O(n\log n)$ であるが, それ以外は $O(n)$ か定数時間でおさえることが出来る. 仮説を抽出する部分でも, 抽出・削除ともに $O(n)$ でおさえられるので, 全体としては $O(n\log n)$ のアルゴリズムである.

7　必要なサンプル数

学習概念が k 個の半空間で構成されているとき, この主細分関数 [4] $\pi_S(m)$ は以下のようになる.

$$\pi_S(m) = 1 + \binom{m+1}{k}$$

そこで, この主細分関数 $\pi_S(m)$ を定理 2 を用いて上から抑える [3, 4]. このときの VC 次元 [3, 5, 8] は $2k+1$ となるので,

$$\pi_S(m) \le \Phi_d(m) = \sum_{i=0}^{d}\binom{m}{i} = \sum_{i=0}^{2k+1}\binom{m}{i}$$

となる. これを定理 3 にあてはめればサンプル数の理論値を求めることができる [3, 4]. また, VC 次元を用いなくとも, $\pi_S(m) \le (am)^d$(a, d は正の定数) で抑えることができれば, それでも良い.

定理 2　領域空間 $S = (X, \mathcal{R})$ の VC 次元を d とする. このとき, 任意の $m \ge 0$ に対し,

$$\pi_S(m) \le \Phi_d(m)$$

が成り立つ. ただし,

$$\Phi_d(m) = \begin{cases} \sum_{i=0}^{d}\binom{m}{i} & (m > d) \\ 2^m & (m \le d) \end{cases}$$

である.

定理 3　定義域 X 上の概念クラス $\mathcal{C}$ が $\pi_{\mathcal{C}}(m) \le (am)^d$ (a, d は正の定数) を満たすとき, 任意の概念 $C_T \in \mathcal{C}$ と任意の正の実数 $\epsilon, \delta (0 < \epsilon, \delta < 1)$ に対して,

$$m \ge \max\left\{\frac{4}{\epsilon}\log\frac{2}{\delta}, \frac{6d}{\epsilon}\log\frac{16ad}{\epsilon}\right\}$$

ならば, m 点以上のサンプルを入力したとき, 無矛盾なアルゴリズムが出力する仮説 C_H は, $(1-\delta)$ 以上の確率で,

$$P(C_H \triangle C_T) < \epsilon$$

を満たす.

8 計算機実験

ここでは, 前節のアルゴリズムを実装して行った計算機実験について述べている.

8.1 実験方法

各学習概念に対して, ϵ, δ のパラメーターを 0.05 ～ 0.2 まで 0.05 の幅で動かして計算機実験を行なった. 実験の一例として, 表 1 を載せる. この表は学習概念, その概念を学習するのに必要なサンプル数を理論的に与える式, 計算機実験より得られた学習に必要なサンプル数を与える式を表している.

表 1. 学習結果の一例

学習概念	$\begin{cases} x+y-z \leq 0 \\ -x+y-z \leq 0 \\ \frac{1}{2}x-\frac{3}{2}y-z \leq 0 \end{cases}$
サンプル数の理論値	$\max\left\{\frac{4}{\epsilon}\ln\frac{2}{\delta}, \frac{18}{\epsilon}\ln\frac{16}{\epsilon}\right\}$
実験から導いたサンプル数	$\frac{168}{\epsilon}\ln\frac{0.52}{\delta}$

8.2 実験結果

表 1 の他にも半空間の数を増やした場合の学習概念についても実験を行なった. その結果, 学習概念の 2 枚角が学習アルゴリズムで定めている基準角よりも小さい角度を含んでいる場合, 学習は失敗に終わる結果となった. しかし, 学習概念が持つ 2 枚角全てが学習角の範囲内におさまるときは, 学習は成功する. また, 学習角を小さく取りすぎると, 学習に時間がかかるばかりでなく, その仮説にも変化があり, 良い学習をさせるには概念にあった学習角を設定する必要があった.

9 結論

このアルゴリズムは 3 次元上の学習概念を 2 次元上に移しているため, 必要な例の個数を小さくおさえることができる. しかし, 任意の半空間の共通領域の学習に適用することはできない.

さらに, このアルゴリズムをより一般的な問題に適用するには以下の問題点がある.

- このアルゴリズムを n 次元で扱うことはできない. というのは, 問題を低次元おとす場合, homogenous の条件が失われてしまうからである.
- このアルゴリズムを non-homogenous なケースに適用することはできない. というのは 3 次元から 2 次元に問題をおとすときに, 学習概念の性質が失われてしまうからである.

謝辞

この研究の一部は文部省科学研究費の援助を受けている.

参考文献

[1] A.Blum and R.Kannan: "Learning an Intersection of k Halfspaces over a Uniform Distribution," *Proceedings of the 34th Symposium on the Foundations of Compyuter Science*, 1993, pp.312-320.

[2] E.B.Baum: "Polynomial time algorithms for learning neural nets," *Proceedings of the Third Annual Workshop on Computational Learning Theory*, 1990, pp.258-272.

[3] 今井浩: "計算幾何学," 数理科学, No.328, 1990, pp.18-23.

[4] 今井浩, 今井桂子: "計算幾何学," 情報数学講座, 共立出版, 1994, pp.135-170.

[5] M.Anthony and N.Biggs: "Computational Learning Theory," *Cambridge University Press*, 1992.

[6] 岡田 清孝, 今井 桂子: "幾何概念の学習に対する計算機実験による検証," 離散・計算幾何学 *Workshop*, 1997, pp.75-79.

[7] S.Vempala: "A Random Sampling based Algorithm for Learning the Intersection of Halfspaces," *Proceedings of the 38th Symposium on the Foundations of Compyuter Science*, 1997, pp.508-513.

[8]http://theory.lcs.mit.edu/~mona/lectures.html