

# COLT Models

Nader H. Bshouty

## Abstract

Here we give the models of learning that we will study.

## 1 Complexity

Let  $C$  be a class of functions  $f : X \rightarrow \{0, 1\}$ . The domain  $X$  can be finite, countable infinite, or  $\mathcal{R}^n$  for some  $n \geq 1$ . In learning, a *teacher* has a *target function*  $f \in C$  and a *probability distribution*  $D$  on  $X$ . The *learner* knows  $C$  but does not know the probability distribution  $D$  nor the function  $f$ .

The *problem size*  $I_f$  that we will use in here depends on  $X$ ,  $C$  and  $f$  and it can be different in different settings. The term “polynomial” means polynomial in the problem size  $I_f$ . For example, for Boolean functions with  $X = \{0, 1\}^n$ ,  $C$  is a set of formulas (e.g. DNF, Decision tree, etc.). The problem size is  $I_f = n + size_C(f)$  where  $size_C(f)$  is the minimal size of a formula in  $C$  that is equivalent to  $f$ . Then “polynomial” means  $poly(I_f) = poly(n, size_C(f))$ . For infinite domains  $X$  the parameter  $n$  is usually replaced by the VC-dimension of the class  $V_C$  and  $I_f = V_C + size_C(f)$ . Then “polynomial” in this case is  $poly(V_C, size_C(f))$ .

## 2 Queries

The (randomized) learner can ask the teacher *queries* about the target. The teacher can be regarded as an adversary with unlimited computational power that must answer honestly but also wants to fail the learner from learning quickly. The teacher does not know the coin tosses of the learner. The queries we consider are:

**Ex<sub>D</sub>** **Example Query according to  $D$**  [V85] For the example query the teacher chooses  $x \in X$  according to the probability distribution  $D$  and returns  $(x, f(x))$  to the learner.

**EQ** **Equivalence Query** [A88] In the equivalence query the learner asks EQ( $h$ ) for some polynomial size circuit  $h$ . The teacher chooses  $y \in X_{f\Delta h} \triangleq \{x \in X \mid f(x) \neq h(x)\}$  and returns  $y$ . If  $X_{f\Delta h}$  is empty, the teacher answers “YES”, indicating that  $h$  is equivalent to  $f$ .

### 3 \*Other queries

**EQ<sub>f</sub>** **Equivalence Queries according to the function  $f$**  The teacher answer equivalence queries according to a function  $f$  unknown to the learner. That is, the teacher does not change the target during the learning process but it still can choose the counterexample it wants.

**EQ<sub>D</sub>** **Equivalence Query according to  $D$**  [B97] For the equivalence query according to distribution  $D$  the learner asks EQ<sub>D</sub>( $h$ ) for some polynomial size circuit<sup>1</sup>  $h$ . The teacher chooses  $y \in X_{f\Delta h}$  according to the induced distribution of  $D$  on  $X_{f\Delta h}$  and returns  $(y, f(y))$ . If  $\Pr_D[X_{f\Delta h}] = 0$ , the teacher answers “YES”.

The EQ<sub>D</sub> can be extended to also handle random hypothesis [BG02]. A random hypothesis  $h_r : X \times R \rightarrow \{0, 1\}$  is a polynomial size circuit where for an input  $x_0 \in X$  it randomly uniformly chooses  $r_0 \in R$  and returns  $h_{r_0}(x_0)$ . In that case, EQ<sub>D</sub>( $h_r(x)$ ) returns an output of the following procedure

1. Randomly choose  $x_0 \in X$  according to distribution  $D$ .
2. Randomly uniformly choose  $r_0 \in R$ .
3. if  $f(x_0) \neq h_{r_0}(x_0)$  then output  $(x_0, f(x_0))$  else goto (1).

This procedure returns the first example  $x_0$  that  $h_r$  err on.

When  $X$  is finite or countable infinite, each  $x_0$  is received with probability

$$\frac{D(x_0) \Pr_r[h_r(x_0) \neq f(x_0)]}{\sum_x D(x) \Pr_r[h_r(x) \neq f(x)]}.$$

**EQ<sub>S,f</sub>** **Equivalence Query according to a sequence  $S$**  The answers to equivalence queries for a function  $f \in C$  and is the first counterexample of a sequence that is determined before the learning process.

### 4 Query Models

The learning models we will consider here are

**CH** (Consistent Hypothesis) In the CH learning model we say that algorithm  $\mathcal{A}$  of the learner *CH-learns* the class  $C$  if there is a constant  $\alpha < 1$  such that for any set  $S \subseteq X \times \{0, 1\}$  that is consistent with some  $f \in C$  ( $f$  consistent with  $S$  if for all  $(x, b) \in S$ ,  $f(x) = b$ ) and for every  $\delta$ , algorithm  $\mathcal{A}(S, \delta)$  with probability at least  $1 - \delta$  outputs a polynomial size circuit  $h$  of size at most  $|S|^{\alpha} poly(I_f)$  that is consistent with  $S$ . We say that  $C$  is *CH-learnable* if there is an algorithm that CH-learn  $C$  in time  $poly(\log(1/\delta), I_f, |S|)$ .

---

<sup>1</sup>For infinite domains  $X$ , the definition of “circuit” depends on the setting in which the elements of  $C$  are represented. The hypothesis  $h$  must have polynomial size in this setting. E.g., if  $X = \mathcal{R}^n$  we may ask of  $h$  to be a polynomial size *arithmetic* circuit

**PAC** (Probably Approximately Correct)[V85] In the PAC learning model we say that an algorithm  $\mathcal{A}$  of the learner *PAC-learns* the class  $C$  if for any  $f \in C$ , any probability distribution  $D$  and any  $\varepsilon, \delta > 0$  the algorithm  $\mathcal{A}(\varepsilon, \delta)$  asks example queries according to  $D$ ,  $\text{Ex}_D$ , and with probability at least  $1 - \delta$ , outputs a polynomial size circuit  $h$  that  $\varepsilon$ -approximates  $f$  with respect to  $D$ . That is  $\Pr_D[X_{f\Delta h}] \leq \varepsilon$ . We say that  $C$  is *PAC-learnable* if there is an algorithm that PAC-learns  $C$  in time  $\text{poly}(1/\varepsilon, \log(1/\delta), I_f)$ .

**Exact** (Exactly Correct) [A88] In the Exact-learning model we say that an algorithm  $\mathcal{A}$  of the learner *Exact-learns* the class  $C$  if for any  $f \in C$  and any  $\delta$  the algorithm  $\mathcal{A}(\delta)$  asks equivalence queries and with probability at least  $1 - \delta$  outputs a polynomial size circuit  $h$  that is equivalent to  $f$ . We say that  $C$  is *Exact-learnable* if there is an algorithm that Exact-learns  $C$  in time  $\text{poly}(\log(1/\delta), I_f)$ .

## 5 \*Other Query Models

**PExact** (Probably Exactly Correct) [B97] In the Probably Exact model we say that an algorithm  $\mathcal{A}$  of the learner *PExact-learns* the class  $C$  if for any  $f \in C$  and any  $\delta$  the algorithm  $\mathcal{A}(\delta)$  asks equivalence queries with respect to  $D$ ,  $\text{EQ}_D$ , and with probability at least  $1 - \delta$  outputs a polynomial size circuit  $h$  that satisfies  $\Pr_D[X_{f\Delta h}] = 0$ . We say that  $C$  is *PExact-learnable* if there is an algorithm that Probably Exact-learns  $C$  in time  $\text{poly}(\log(1/\delta), I_f)$ .

If the hypothesis in the equivalence query can also be randomized then we refer to this model as  $\text{PExact}_R$ .

**PAExact** (Probably Almost Exactly Correct) [BJT02] In the Almost Exact model we say that an algorithm  $\mathcal{A}$  of the learner *PAExact-learns* the class  $C$  if for any  $f \in C$  and any  $\delta$  the algorithm  $\mathcal{A}(\delta)$  asks equivalence queries with respect to  $D$ ,  $\text{EQ}_D$ , and with probability at least  $1 - \delta$  outputs a polynomial size circuit  $h$  that satisfies  $\Pr_D[X_{f\Delta h}] = 1/\omega(\text{poly}(I_f))$ . We say that  $C$  is *PAExact-learnable* if there is an algorithm that PAExact-learns  $C$  in time  $\text{poly}(\log(1/\delta), I_f)$ .

If the hypothesis in the equivalence query can also be randomized then we refer to this model as  $\text{PAExact}_R$ .

We say that  $C$  is  $\eta$ -*PAExact-learnable* if it is  $\text{PAExact}$ -learnable and it achieves error at most  $\eta$ . That is,  $\Pr_D[X_{f\Delta h}] \leq \eta$ . When randomized hypotheses are used in the equivalence query then we say that  $C$  is  $\eta$ -*PAExact<sub>R</sub>-learnable*.

## 6 Online Models

In the online learning model [L88] the teacher at each trial sends a point  $x \in X$  to the learner and the learner has to predict  $f(x)$ . The learner returns to the teacher the prediction  $y$ . If  $f(x) \neq y$  then the teacher returns “mistake” to the learner. The goal of the learner is to minimize the number of prediction mistakes. The teacher can be regarded as an adversary with

unlimited computational power that must answer honestly but also wants to fail the learner from learning quickly. The teacher does not know the coin tosses of the learner.

**Online** [L88] In the online model we say that algorithm  $\mathcal{A}$  of the learner *Online-learns* the class  $C$  if for any  $f \in C$  and for any  $\delta$ , algorithm  $\mathcal{A}(\delta)$  with probability at least  $1 - \delta$  makes bounded number of mistakes. We say that  $C$  is *Online-learnable* if the number of mistakes and the running time of the learner for each prediction is  $\text{poly}(\log(1/\delta), I_f)$ .

## 7 \*Other Online Model

**Online<sub>f</sub>** This is the online model where the teacher fixes the target function before running the learning algorithm.

**Online<sub>f,S</sub>** [M91] This is the online model where the teacher fixes the target function and the sequence of points that will be sent to the learner before running the learning algorithm.

**Probabilistic Prediction (PP)** [HLW94] In the Probabilistic Prediction model the points sent to the learner are chosen from  $X$  according to some distribution  $D$ . We say an algorithm  $\mathcal{A}$  of the learner  $\eta$ -*PP-learns* the class  $C$  if for any  $f \in C$  and for any  $\delta$  the algorithm  $\mathcal{A}(\delta)$  with probability at least  $1 - \delta$  after bounded number of mistakes can predict the answer with probability greater than  $1 - \eta$ . We say that  $C$  is  $\eta$ -*PP-learnable* if the number of mistakes and the running time of the learner at each trial is  $\text{poly}(\log(1/\delta), I_f)$ .

**Bibliography.** Many papers investigated the online model ignoring computational efficiency. Maass in [M91] investigated connections between different oblivious and adaptive online models, and, randomized vs non-randomized online models.

## 8 \*Models with Other Queries

### 8.1 Other Queries

We will also consider the following queries

**MQ Membership query** [A88]. In the membership query the learner asks  $\text{MQ}(a)$  where  $a \in X$  and the teacher returns  $f(a)$ .

**SQ<sub>D</sub> Statistical query according to  $D$**  [K93]. In the statistical query the learner asks  $\text{SQ}_D(g, \tau)$  where  $g$  is a polynomial size circuit  $g : X \times \{0, 1\} \rightarrow \{0, 1\}$  and  $\tau$  is a real number called the *tolerance* of the query. The teacher returns some value  $v$  such that

$$E_D[g(x, f(x))] - \tau \leq v \leq E_D[g(x, f(x))] + \tau.$$

**UEx<sub>D</sub>** **Unlabeled Example Query according to  $D$**  For the unlabeled example query the teacher chooses  $x \in X$  according to the probability distribution  $D$  and returns  $x$  to the learner (without its label).

## 8.2 Other Models

The learning models we will consider here are

**SQ** (Statistical Query Model)[K93] In the SQ learning model we say that an algorithm  $\mathcal{A}$  of the learner *SQ-learns* the class  $C$  if for any  $f \in C$ , any probability distribution  $D$  and any  $\varepsilon, \delta > 0$  the algorithm  $\mathcal{A}(\varepsilon, \delta)$  asks statistical queries according to  $D$  with tolerant  $\tau = 1/\text{poly}(I_f, 1/\varepsilon)$  and unlabeled queries according to  $D$ , and with probability at least  $1 - \delta$ , outputs a polynomial size circuit  $h$  that  $\varepsilon$ -approximates  $f$  with respect to  $D$ . That is  $\Pr_D[X_{f\Delta h}] \leq \varepsilon$ . We say that  $C$  is *SQ-learnable* if there is an algorithm that SQ-learns  $C$  in time  $\text{poly}(1/\varepsilon, \log(1/\delta), I_f)$ .

**CH(MQ)** (Consistent Hypothesis with Membership Query) Is the same as the CH-model where the learner is also allowed to ask MQ.

**PAC(MQ)** (Probably Approximately Correct with Membership queries) Is the same as the PAC model where the learner is also allowed to ask MQ.

**Exact(MQ)** (Exactly Correct with Membership queries) [A88] Is the same as the Exact model where the learner is also allowed to ask MQ.

**PExact(MQ)** (Probably Exactly Correct with Membership queries) [B97] Is the same as the PExact model where the learner is also allowed to ask MQ.

**PAExact(MQ)** (Probably Almost Exactly Correct with Membership queries) [BJT02] Is the same as the PAExact model where the learner is also allowed to ask MQ.

**Online(MQ)** [L88] Is the same as the Online model where the learner is also allowed to ask MQ.

**PP(MQ)** [HLW94] (Probabilistic Prediction with Membership queries) [HLW94] Is the same as the Probabilistic Prediction model where the learner is also allowed to ask MQ.

## 9 \*Models with Noise

**Example Query (Ex)** [V85] For the example query the teacher chooses  $x \in X$  according to a distribution  $D$  and returns  $(x, f(x))$  to the learner.

**Example Query with Malicious Errors (ExMAL <sub>$\beta$</sub> )** [V85] For the example query with malicious error  $\beta$  the teacher chooses  $x \in X$  according to the distribution  $D$ . Then with probability  $1 - \beta$  the teacher returns  $(x, f(x))$  and with probability  $\beta$  returns arbitrary  $(z, y)$  with  $z \in X$  and  $y \in \{0, 1\}$ .

**Example Query with Classification Noise (ExCN $_{\alpha}$ )** [AL88] For the example query with classification noise  $\alpha$  the teacher chooses  $x \in X$  according to the distribution  $D$ . Then with probability  $1 - \alpha$  the teacher returns  $(x, f(x))$  and with probability  $\alpha$  returns  $(x, 1 - f(x))$ .

We say that  $h \in C$  is  $\epsilon$ -good hypothesis with respect to  $f$  and  $D$  if

$$\Pr_{x \in_D X} [f(x) \neq h(x)] \leq \epsilon.$$

The goal of the learner is to output with high probability (probability greater than  $1 - \delta$ ) an  $\epsilon$ -good hypothesis with respect to  $f$  and  $D$ . The learning models we will consider in this paper are

**PAC** (Probably Approximately Correct) In the PAC learning model we say that an algorithm  $\mathcal{A}$  of the learner *PAC-learns* the class  $C$  if for any  $f \in C$ , any distribution  $D$  and for any  $\epsilon, \delta > 0$  the algorithm  $\mathcal{A}(\epsilon, \delta)$  asks queries from the Ex oracle and, with probability at least  $1 - \delta$ , outputs a hypothesis  $h \in C$  that is an  $\epsilon$ -good hypothesis with respect to  $f$  and  $D$ . If  $\mathcal{A}$  runs in time  $T$  (e.g. polynomial, exponential, etc.) in  $1/\epsilon, 1/\delta$  and  $|f|$  then we say that  $C$  is *PAC-learnable* in time  $T$ . Here  $|f|$  also includes  $\log |X|$  when  $X$  is finite. For example when  $X = B_n$  then  $|f|$  is defined as the length of the representation of  $f$  plus  $n$ . If we allow  $h$  to be from a larger class  $H \supset C$  then we say that  $C$  is *PAC-learnable from  $H$*  in time  $T$ .

**PAC with malicious errors** In the PAC learning model with malicious errors, for  $0 \leq \beta < 1/3$  and  $\epsilon > 0$ , we say that an algorithm  $\mathcal{A}$  of the learner *PAC-learns* the class  $C$  with malicious error rate  $\beta$  and error  $\epsilon$  if for any  $f \in C$  and any distribution  $D$  and for any  $\delta > 0$ , the algorithm  $\mathcal{A}(\delta)$  asks queries from the ExMAL $_{\beta}$  oracle and, with probability at least  $1 - \delta$ , outputs a hypothesis  $h \in C$  that is an  $\epsilon$ -good hypothesis with respect to  $f$  and  $D$ . The algorithm knows  $\epsilon$  and  $\delta$  but does not know  $\beta$ . In this model we define  $\epsilon_{MAL}^T(C, \beta)$  to be the minimal possible  $\epsilon$  such that an algorithm that runs in time  $T$  in  $1/\delta, |f|$  and  $1/\eta$  exists that learns with error  $\epsilon_{MAL}^T(C, \beta) + \eta$  for any  $\eta > 0$ . If we allow  $h$  to be from a larger class  $H \supset C$  then we write  $\epsilon_{MAL}^T(C/H, \beta)$ . It is known from [?] that  $\epsilon_{MAL}^{\infty}(C, \beta) > \min(\beta/(1 - \beta), 1/2)$ . Notice that if  $\beta \geq 1/3$  then  $\epsilon_{MAL}^{\infty}(C, \beta) = 1/2$  and learning with any  $\epsilon$  is impossible. When  $T$  is polynomial time we write  $\epsilon_{MAL}(C, \beta)$ .

**Agnostic PAC** [Ha92, KSS92] In the Agnostic PAC learning model we have a distribution  $P$  on  $X \times \{0, 1\}$  and an oracle  $\text{Exp}$  that produces examples according to this distribution. We say that an algorithm  $\mathcal{A}$  (of the learner) *Agnostic PAC-learns* the class  $C$  if for any distribution  $P$  and for any  $\epsilon, \delta > 0$  the algorithm  $\mathcal{A}(\epsilon, \delta)$  asks queries from  $\text{Exp}$  and, with probability at least  $1 - \delta$ , outputs a hypothesis  $h \in C$  that satisfies

$$\Pr_{(x,y) \in_P X \times \{0,1\}} [h(x) \neq y] \leq \min_{f \in C} \Pr_{(x,y) \in_P X \times \{0,1\}} [f(x) \neq y] + \epsilon.$$

If  $\mathcal{A}$  runs in time  $T$  in  $1/\epsilon, 1/\delta$  and  $|f|$  then we say that  $C$  is *Agnostic PAC-learnable* in time  $T$ . If we allow  $h$  to be from a larger class  $H$  then we say that  $C$  is *Agnostic*

*PAC-learnable* from  $H$  in time  $T$ . We make a relaxation of the Agnostic PAC-learning and define  $\alpha$ -Agnostic PAC-learning and  $\alpha$ -CoAgnostic PAC-learning. We say that  $C$  is  $\alpha$ -Agnostic PAC-learnable from  $H$  in time  $T$  if

$$\Pr_{(x,y) \in_P X \times \{0,1\}} [h(x) \neq y] \leq \alpha \min_{f \in C} \Pr_{(x,y) \in_P X \times \{0,1\}} [f(x) \neq y] + \epsilon.$$

See also [HSV95] for some other extensions.

We say that  $C$  is  $\alpha$ -CoAgnostic PAC-learnable from  $H$  in time  $T$  if

$$\Pr_{(x,y) \in_P X \times \{0,1\}} [h(x) = y] \geq \alpha \min_{f \in C} \Pr_{(x,y) \in_P X \times \{0,1\}} [f(x) = y] - \epsilon.$$

## References

- [A88] D. Angluin. newblock Queries and concept learning. *Machine Learning*, 2:319-342, 1987.
- [AL88] D. Angluin and P. D. Laird, Learning from noisy examples, *Machine Learning*, 2:343-370,1988.
- [B97] N. H. Bshouty, Exact learning of formulas in parallel. *Machine Learning*, 26,pp. 25-41,1997.
- [BG02] N. H. Bshouty and D. Gavinsky, PAC=PAExact and other equivalent models in learning Proceedings of the 43rd Ann. Symp. on Foundation of Computer Science. (FOCS) 2002.
- [BJT02] N. H. Bshouty, J. Jackson and C. Tamon, Exploring learnability between exact and PAC, Proceedings of the 15th Annual Conference on Computational Learning Theory, 2002.
- [Ha92] D. Haussler, Decision theoretic generalization of the PAC model for neural net and other learning applications. *Inform. Comput.*, 100(1):78-150, Sept. 1992.
- [HLW94] D. Haussler, N. Littlestone and M. K. Warmuth, Predicting 0,1-functions on randomly drawn points, *Information and Computation*, 115,pp. 248-292,1994.
- [HSV95] Klaus-U Höffgen, Hans-U. Simon and Kevin S. Van Horn, Robust trainability of single neurons, *J. of Comput. Syst. Sci.*, 50(1): 114-125, 1995.
- [K93] Michael J. Kearns, Efficient Noise-Tolerant Learning from Statistical Queries. *JACM* 45(6): 983-1006 (1998) and *STOC* 1993: 392-401.
- [KSS92] M. Kearns, R. E. Schapire and L. M. Sellie. Toward efficient agnostic learning. *Proceedings of the Fifth Annual ACM Workshop on Computational Learning Theory*, 341-352, 1992.
- [L88] N. Littlestone. Learning when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

- [M91] W. Maass. On-line learning with an oblivious environment and the power of randomization. In Proceedings of the 4th Annual ACM Workshop on Computational Learning Theory, 167–175. 1991.
- [V84] L. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.
- [V85] L. G. Valiant, Learning disjunctions of conjunctions *Proceeding of the 9th International Joint Conference on Artificial Intelligence*, 560-566, 1985.