Seminar in Computational Learning 28.12.1999

Lecture 10: A New Composition Theorem for Learning Algorithms

Authors: Nader Bshouty
Scribe: Ittai Balaban and Lior Malka

Abstract

An algorithm is presented which implements a composition of exact-learning algorithms for "simple" classes, into learning algorithms for more "complicated" classes. It is shown that if a class with constant VC-Dimension $\mathcal{C}$ is learnable in time t, then a function F which is a composition of functions in $\mathcal{C}$ is learnable in time polynomial in t and the size of F.

1 Preliminaries

1.1 VC-Dimension

Definition 1 *For any concept class $\mathcal{C}$ over domain X, and any $S \subseteq X$:*
If $S = \{x_1, \ldots, x_m\}$ then $\Pi_{\mathcal{C}}(S)$ is a set of binary vectors:

$$\Pi_{\mathcal{C}}(S) = \{(c(x_1), \ldots, c(x_m)) : c \in \mathcal{C}\}$$

Usually we'll use $\Pi(S)$ instead of $\Pi_{\mathcal{C}}(S)$ when the meaning of $\mathcal{C}$ is clear from the context. $\Pi(S)$ represents all the behaviors of $\mathcal{C}$ on S. Each member vector of $\Pi(S)$ represents a single function $c \in \mathcal{C}$, reduced to S. If $\mathcal{C}$ contains all the possible behaviors on S, then this means that $\Pi(S)$ contains all possible binary words of length m. This leads us to the following definition.

Definition 2 *If $|\Pi(S)| = 2^m$ then we say that S is* ***shattered*** *by $\mathcal{C}$.*

Definition 3 *The* **Vapnik-Chervonenkis Dimension** *of $\mathcal{C}$, or $VC-dim(\mathcal{C})$, is the cardinality of the largest set S shattered by $\mathcal{C}$.*

1.2 Dual Class

Let $X = \{x_1, \ldots, x_m\}$ be a set and $\mathcal{H} \subseteq \{0,1\}^X$ a class of functions $h : X \to \{0,1\}$.

Definition 4 *The* dual *class of $\mathcal{H}$, denoted $\mathcal{H}^\perp \subseteq \{0,1\}^{\mathcal{H}}$, contains all functions $x_i^\perp : \mathcal{H} \to \{0,1\}$ where for every $h \subseteq \mathcal{H}$ we have $x_i^\perp(h) = h(x)$.*

The following relation between VC-dim($\mathcal{H}$) and VC-dim($\mathcal{H}^\perp$) is well known:

Claim 5 *For every class $\mathcal{H}$, VC-dim($\mathcal{H}$) $\geq \log \lfloor$ VC-dim($\mathcal{H}^\perp$)$\rfloor$.*

For the concept classes of half spaces in two dimensions, we will use the following relation (although no proof is provided here):

$$\text{VC-dim}(\mathcal{H}) = \text{VC-dim}(\mathcal{H}^\perp) = 3.$$

2 The Composition Class and Halfspaces

Let $X_n = \{0, 1, \ldots, n\}^d$ for some integers n and d. We consider classes of boolean functions over X_n, and, in particular, the class of *halfspaces* over X_n. Such a halfspace is a function of the form

$$h(x) = \begin{cases} 1 & \text{if } c_1x_1 + \cdots + c_dx_d \geq b \\ 0 & \text{otherwise} \end{cases}$$

where $c_1, \ldots, c_d, b$ are integers.

A hyperplane is defined as

$$h(x) = \begin{cases} 1 & \text{if } c_1x_1 + \cdots + c_dx_d = b \\ 0 & \text{otherwise} \end{cases}$$

where $c_1, \ldots, c_d, b$ are integers.

Definition 6 *Let $\mathcal{C}$ be the class of halfspaces over X_n. Define $\mathcal{C}^r$ as:*

$$\mathcal{C}^r = \left\{ F : X_n \to \{0,1\} : \begin{array}{l} F(x) = f(g_1(x), \ldots, g_r(x)) \\ \textit{where } g_1, \ldots, g_r \in \mathcal{C} \\ \textit{and } f : \{0,\ 1\}^r \to \{0,\ 1\} \textit{ is some boolean function} \end{array} \right\}$$

It can be demonstrated that any geometric object bounded by r hyperplanes is in $\mathcal{C}^r$.

Definition 7 $\mathcal{C}^* = \bigcup_r \mathcal{C}^r$

The *size* of a geometric object G in $\mathcal{C}^*$ is the minimal r such that $G \in \mathcal{C}^r$. For example, the size of a 2-dimensional cube is 4. The size of the union of k such cubes is $4k$.

For a given $\mathcal{C}$, we will be learning $\mathcal{C}^*$, its *Composition Class.*

3 Consistency Algorithms

Let P and N be sets of points in X. A boolean function $h : X \to \{0,1\}$ is *consistent* with (P, N) if $\forall x \in P : h(x) = 1$ and $\forall x \in N : h(x) = 0$.

For two classes $\mathcal{C}$ and $\mathcal{H}$, the pair $(\mathcal{C}, \mathcal{H})$ has a *consistency algorithm* if there is an algorithm that takes (P, N) as input, runs in time polynomial in $|P \cup N|$, and satisfies the following: If there is a function $h \in \mathcal{H}$ that is consistent with (P, N) then the algorithm outputs "YES" and a consistent h. If there is no consistent h the algorithm outputs "NO".

Claim 8 *If class $\mathcal{C}$ is exact-learnable from $\mathcal{H}$ with equivalence queries, then $(\mathcal{C}, \mathcal{H})$ has a consistency algorithm.*

Proof:

Let ALG be the algorithm that learns $\mathcal{C}$ from $\mathcal{H}$. We define a consistency algorithm CON which utilizes ALG, as follows:

CON runs ALG, where CON simulates the teacher. All the counterexamples given to ALG will be from $|P \cup N|$. Specifically, each time ALG makes an equivalence query with an h, CON calculates h over all points in $|P \cup N|$. If there exists a point where h is mistaken then CON passes it as a counterexample to ALG. Otherwise, ALG has found a consistent hypothesis.

If there is no consistent $h \in \mathcal{H}$ then either ALG will either get stuck, run longer than it should, output an inconsistent h, or output an $h \notin \mathcal{H}$. All these can be verified in polynomial time. ■

Let Q be a set of points, and CON be a consistency algorithm for $(\mathcal{C}, \mathcal{H})$. We would like to be able to generate the set $\Pi_{\mathcal{H}}(Q)$ (as defined in Definition 1). Algorithm $S_{CON}(Q)$ in Figure 1 does just this in polynomial time.

The following lemma is given without proof. It gives us an upper bound on the size $|\Pi(Q)|$.

Lemma 9 $|\Pi(Q)| \leq |Q|^{VC\text{-}dim(\mathcal{H})}$.

Lemma 9 provides for the polynomiality of algorithm 1 and the main algorithm further on.

Claim 10 *Algorithm $S_{CON}(Q)$ outputs $\Pi(Q)$ and runs in time polynomial in $|Q|$ and the running time of CON.*

Proof: Before the main loop has begun, S is defined, in the first line, as $\Pi(\emptyset)$. In the inductive step, Assume that for some $0 < i \leq |Q|$, $S = \Pi(\{x_1, \ldots, x_{i-1}\})$. $\mathcal{T}$ is first initialized to $\emptyset$. For every $((P, N), h) \in S$, CON is run on $(P \cup \{x_i\}, N)$ and on $(P, N \cup \{x_i\})$. Because there is a consistent h for $((P, N), h)$, there is also a consistent hypothesis for at least one of the two options tried. For any of of the two, if CON answered ("YES", h') then the new pair is added to $\mathcal{T}$. Thus, at the end of the inner loop $\mathcal{T} = \Pi(\{x_1, \ldots, x_i\})$. This $\mathcal{T}$ is then assigned to S. In the next

Algorithm $S_{CON}(Q)$

$S \leftarrow \{((\emptyset,\emptyset),0)\}$
Let $Q = \{x_1,\ldots,x_q\}$
for $i = 1$ to q do
  $\mathcal{T} \leftarrow \emptyset$
  for all $((P,N),h) \in S$ do
    run CON$(P \cup \{x_i\}, N)$ and if the answer is (“YES”, h') do
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{((P \cup \{x_i\}, N), h')\}$
    run CON$(P, N \cup \{x_i\})$ and if the answer is (“YES”, h') do
      $\mathcal{T} \leftarrow \mathcal{T} \cup \{((P, N \cup \{x_i\}), h')\}$
  $S \leftarrow \mathcal{T}$
Output(S)

Figure 1: An algorithm for generating $|\Pi(Q)|$.

main loop iteration, the assumption will hold for $i+1$. Therefore $S_{CON}(Q)$ outputs the correct S.

The complexity of $S_{CON}(Q)$ is as follows: The main loop runs $|Q|$ iterations. At each iteration the inner loop runs at most $|\Pi(Q)|$ iterations. At each such iteration, CON is invoked twice. Thus the running time is at most $2\,|\Pi(Q)|\,|Q|\,T(CON)$, where $T(CON)$ is the running time of CON on a (P,N) such that $P \cup N \subseteq Q$. With Lemma 9 we have $O(|Q|^{\text{VC-dim}(\mathcal{H})}\, T(CON))$. ■

4 The Composition Theorem

Theorem 11 *Let $\mathcal{C}$ and $\mathcal{H}$ be classes of boolean functions over a domain X where $\mathcal{C} \subseteq \mathcal{H}$. If the concept class $\mathcal{C}$ is learnable from $\mathcal{H}$ using q equivalence queries in time T then the concept class $\mathcal{C}^*$ is learnable using*

$$p = (2mq)^{\text{VC-dim}(\mathcal{H})\,\text{VC-dim}(\mathcal{H}^{\perp})}$$

equivalence queries, where m is the size of the target function F in $\mathcal{C}^*$, i.e., the number of functions in $\mathcal{C}$ on which F is based.
The running time is polynomial in p and T.

Proof:

Claim 12 *We have $|Q| \leq 2qm$.*

Proof: We show that for every r, after executing steps 2-9 r times there are $r_1, \ldots, r_m$ such that: For every g_i there is a hypothesis h_i equivalent to what we would get from running ALG for r_i steps for the target g_i, and $r \leq r_1 + \cdots + r_m$. A single "step" means running ALG until an equivalence query is asked. This is shown by showing that one execution of steps 2-9 is equivalent to running ALG at least one more step for some g_i.

Suppose we get to step 8, and $M(h_1(a), \ldots, h_t(a)) \neq *$. There is an a' such that

$$(h_1(a), \ldots, h_t(a)) = (h_1(a'), \ldots, h_t(a'))$$

and

$$f(g_1(a), \ldots, g_m(a)) \neq f(g_1(a'), \ldots, g_m(a'))$$

Thus we must have $g_i(a) \neq g_i(a')$ for at least one g_i. This means that either a or a' is a counterexample for h_i. Therefore, the next generation of $\Pi(Q)$ for $Q \leftarrow Q \cup \{a, a'\}$ will contain a hypothesis generated by ALG in one more step, a hypothesis which differentiates between a and a'.

Thus steps 2-9 are executed at most qm times. At each execution 2 elements are added to Q. ■

In order to show the running time we show the following.

Claim 13 *For every t (which is the number of hypothesis in $\Pi(Q)$), the number of elements $(y_1, \ldots, y_t) \in \{0,1\}^t$ such that $M(y_1, \ldots, y_t) \neq *$ is at most*

$$t^{\text{VC-dim}(\mathcal{H}^{\perp})}$$

Proof: For every $y = (y_1, \ldots, y_t)$ for which $M(y) \neq *$ we have $y = h_1(x), \ldots, h_t(x)$ for some x. Therefore, the number of non-star entries in M is at most the number of different values of $y = h_1(x), \ldots, h_t(x)$ for every $x \in X$. These are exactly functions $x^\perp \in \mathcal{H}^\perp$. Therefore, the number of non-star entries in M is $|\Pi_{\mathcal{H}^\perp}(\{h_1, \ldots, h_t\})|$. Using Lemma 9, the result follows. ■

Now, by Lemma 9 We have

$$t = |\Pi(Q)| \leq (2qm)^{\text{VC-dim}(\mathcal{H})}.$$

From Claim 13 we have that the number of equivalence queries is

$$t^{\text{VC-dim}(\mathcal{H}^\perp)} \leq (2qm)^{\text{VC-dim}(\mathcal{H})\text{VC-dim}(\mathcal{H}^\perp)}.$$

That is, at most

$$(2qm)^{\text{VC-dim}(\mathcal{H})\text{VC-dim}(\mathcal{H}^\perp)}.$$

■

In a two-dimensional domain X_n, there exists an algorithm for learning the class $\mathcal{C}$ of halfplanes over X_n from $\mathcal{H}$, where $\mathcal{C} \subseteq \mathcal{H}$, using $O(\log n)$ equivalence queries [1]. Given that VC-dim$(\mathcal{H})$ = VC-dim$(\mathcal{H}^\perp) = 3$, learning an object in $\mathcal{C}^*$ using ALG* can be done in $O((m \log n)^{3 \cdot 3})$.

References

[1] W. Maass and G. Turán. How fast can a threshold gate learn? In S. J. Hanson, G. A. Drastal, and R. L. Rivest, editors, *Computational learning theory and natural learning systems*, volume 1, pages 381–414. MIT Press, 1994.

[2] N. H. Bshouty. A new composition theorem for learning algorithms. In Proc. of the 30th Symp. on Theory of Computing (STOC), pages 583–589, 1998.

Algorithm ALG*

1. $Q \leftarrow \emptyset$
2. Let $\Pi(Q) = \{((P_1, N_1), h_1), \ldots, ((P_t, N_t), h_t)\}$
3. Define a table $M(y_1, \ldots, y_t) = *$ for $(y_1, \ldots, y_t) \in \{0, 1\}^t$
4. $A \leftarrow \emptyset$
5. Define a hypothesis

$$H(y_1, \ldots, y_t) = \begin{cases} M(y_1, \ldots, y_t) & \text{if } M(y_1, \ldots, y_t) \neq * \\ 0 & \text{otherwise} \end{cases}$$

6. $a \leftarrow EQ(H(h_1(x), \ldots, h_t(x)))$
   If the answer was "yes" then return $H(h_1(x), \ldots, h_t(x))$ and halt.
7. If $M(h_1(a), \ldots, h_t(a)) = *$ then
   set $M(h_1(a), \ldots, h_t(a))$ to 1
   set $A \leftarrow A \cup \{a\}$
   goto 5
8. If $M(h_1(a), \ldots, h_t(a)) \neq *$ then there exists $a' \in A$ such that

$$(h_1(a), \ldots, h_t(a)) = (h_1(a'), \ldots, h_t(a'))$$

9. $Q \leftarrow Q \cup \{a, a'\}$
10. goto 2

Figure 2: An algorithm for learning $\mathcal{C}^*$.