Probably learning Approximately Correct
(The PAC Learning Model)
Overview

- Introduction/Computational Learning Theory
- PAC Learning Model
- Consistent Learning Algorithms and Potential Learnability
- Proofs
Computability/Complexity

- Computability theory
  - What can/can not be computed by a machine?

- Complexity theory
  - What classes of problems can be computed efficiently by a machine?
Develop general mathematical models of learning.

What classes of problems can be learned within these models?

How efficiently can classes of problems be learned?

Under what conditions is successful learning possible and impossible?

Identify the computational limits of the models.
Overview

- Introduction/Computational Learning Theory
- PAC Learning Model
- Consistent Learning Algorithms and Potential Learnability
- Proofs
The PAC Learning Model

- PAC: „Probably Approximately Correct“
- Introduced in 1984 by L.G. Valiant [1]

Problem Setting

Let $X$ be a set called the instance space.

A target concept over $X$ is a subset $c \subseteq X$ of the instance space.

Each target concept $c \subseteq X$ corresponds to a boolean-valued function $c_b : X \rightarrow \{0, 1\}$, where $c_b(x) = 1$ indicates that $x$ is a positive example of $c$ and $c_b(x) = 0$ that $x$ is a negative example.

$$c_b(x) = \begin{cases} 1; & \forall x \in c \\ 0; & \forall x \not\in c \end{cases}$$
Problem Setting

A concept class $C$ over $X$ is a collection of target concepts over $X$.

The goal of a learning algorithm $L$ within this environment is to produce a hypothesis concept $h \subseteq X$ (or $h : X \rightarrow \{0, 1\}$) that can accurately classify instances as positive or negative examples of an unknown target concept $c$, where $c$ is member a known concept class $C$.

For identifying $h$, $L$ considers some set $H$ of possible hypotheses. $H$ is called hypothesis space.
Problem Setting

- Remark: „Known“ concept class $C$ means, that the designer of the learning algorithm is guaranteed that the target concept will be chosen from $C$.
- The algorithm must work for any $c \in C$.
- That means that there is a strong constraint on the target concept!
- This is often criticised as not realistic and there exists a refined model, called „agnostic PAC-Learning“ where this constraint has been removed.
The learning algorithm will have access to positive and negative examples (training examples) of the unknown target concept $c$, which are drawn randomly according to a probability distribution $D$.

$D$ is referred to as the target distribution.

There are no contradictory examples.
Problem Setting

Where do the positive and negative examples of $c$ come from?

It is assumed that there exists a procedure $EX(c, D)$ which runs in unit time and on each call returns a labeled example $\langle x, c(x) \rangle$, where $x$ is drawn randomly and independently according to $D$. This procedure is called Oracle.
Problem Setting

Of course one is interested in how closely the learner’s hypothesis \( h \) approximates the actual target concept \( c \).

If \( c \) is the target concept and \( h \) is any concept over \( X \) then the distribution \( D \) provides an error-measure between \( h \) and \( c \), the true error of \( h \):

\[
error(h) = \Pr_{x \in D} \left[ c(x) \neq h(x) \right]
\]

which is the probability that for a randomly drawn \( x \) (according to \( D \)) \( c \) and \( h \) disagree.
**Problem Setting**

\[ \text{error}(h) = \Pr_{x \in D} \left[ c(x) \neq h(x) \right] \]

\(\text{error}(h)\) depends on \(c\) and \(D\).

It is not observable by the learner \(L\), because it has no access to \(D\).

\(L\) can only observe the performance of \(h\) over the training examples.
Problem Setting

Ideal learning algorithm $L$ – first try:

- $L$ produces a hypothesis concept $h$ for which $\text{error}(h) = 0$.

- This is unreasonable, because $L$ would have to see every possible instance in $X$ (unrealistic).

- Otherwise there is always a non-zero probability that $h$ and $c$ disagree on an unseen $x$.

- Additionally the provided training examples can be misleading.
Problem Setting

Ideal learning algorithm $L$ – weakened demands:

- $L$ produces a hypothesis concept $h$ such that $\text{error}(h)$ is bounded by some constant $\varepsilon$.

- It is not required that $L$ succeeds for every sequence of randomly drawn training example. Rather the failure probability of $L$ is bound by some constant $\delta$.

- $\varepsilon$ and $\delta$ can be made arbitrarily small.

- We require that $L$ probably ($\delta$... error parameter) learns an approximately correct ($\varepsilon$... confidence parameter) hypothesis.
Problem Setting

Additional requirements (efficiency):

- $L$ makes a small number of calls to $EX(c,D)$
- the amount of computation that $L$ performs is small
- Small means, bounded by a fixed polynomial by some parameters.
Problem Setting

What are these parameters?

- $1/\epsilon$ and $1/\delta$
- $n$ which stands for the size of the instances in $X$.
- $\text{size}(c)$ that measures the size (or complexity) of the smallest representation of each $c$ in $C$
  - $\text{size}(c)$ is the size of the smallest possible representation of $c$.
  - the size of representation of $h$ must be bound by a polynomial of $\text{size}(c)$
  - $L$ needs to „write down“ the representation of $h$!!
Definition of the PAC Learning Model

Let \( C \) be a concept class over a instance space \( X \) consisting of instances of length \( n \).

We say that \( C \) is **PAC-learnable** using \( H \) if there exists an algorithm \( L \) with the following property:

For every concept \( c \in C \), for every distribution \( D \) on \( X \), and for all \( 0 < \varepsilon \leq 1 \) and \( 0 < \delta < 1 \), if \( L \) is given access to \( EX(c,D) \) and inputs \( \varepsilon, \delta, n \) and \( \text{size}(c) \), then with probability at least \((1-\delta)\), \( L \) outputs a hypothesis concept \( h \in H \) satisfying \( \text{error}(h) \leq \varepsilon \).

This probability is taken over the random examples drawn by calls to \( EX(c,D) \) and by any internal randomization of \( L \).

If \( L \) runs in time polynomial in \( 1/\varepsilon \), \( 1/\delta \), \( n \) and \( \text{size}(c) \), we say that \( C \) is **efficiently PAC-learnable**.

\( L \) is called a **PAC learning algorithm** for \( C \).
Remarks

- We implicitly assume that the learner’s hypothesis space $H$ contains a hypothesis with arbitrarily small error for every target concept in $C$.

- In practice this is difficult to assure, since generally one doesn‘t know $C$.

- What is the concept class for a learning algorithm that has to learn to recognize faces from images?
Remarks

- The demand that a PAC learning algorithm performs well with respect to any distribution $D$ is a strong requirement.
- It is moderated by the fact that the evaluation of the produced hypothesis is done with respect to the same distribution.
- That means that instances with low probability of being drawn (according to $D$) as training example the distribution also gives small weight on such instances for computing the true error.
Remarks

- The definition appears to be concerned only with the computational resources required for learning and not with the number of training examples required.

- These two properties are closely related: If $L$ requires some minimum processing time per training example then for $C$ to be PAC learnable by $L$, $L$ must be able to learn from a polynomial number of training examples.

- The growth in the number of required training examples with problem size is called the sample complexity of the problem.
Overview

- Introduction/Computational Learning Theory
- PAC Learning Model
- Consistent Learning Algorithms and Potential Learnability
- Proofs
Training Sample

- Let $c$ be a target concept on the instance space $X$ and let $H$ be a hypothesis space consisting of hypotheses $h : X \to \{0, 1\}$ as defined.

- Define a training sample (training set) $s$ of size $m$ of $c$ as:

$$s = \left\{ \langle x_1, c(x_1) \rangle, ..., \langle x_i, c(x_i) \rangle, ..., \langle x_m, c(x_m) \rangle \right\} \subseteq X \times \{0, 1\}$$

where $x_j \in X; 1 \leq j \leq m$.
A learning algorithm $L$ for $H$ is called consistent if, given any $s$ for a target concept $c \in H$, $L$’s output-hypothesis $h \in H$ agrees with $c$ on all examples in $s$, i.e. $h(x_j) = c(x_j); 1 \leq j \leq m$.

Let $H[s]$ be the set of all consistent hypothesis in $H$ for a given $s$:

$$H[s] = \{h \in H \mid h(x_j) = c(x_j); 1 \leq j \leq m\}$$

Let, for $\varepsilon \in (0,1)$, $B_\varepsilon$ be the set of all hypothesis in $H$ that produce an error $> \varepsilon$:

$$B_\varepsilon = \{h \in H \mid \text{error}(h) > \varepsilon\}$$
Potential learnability

- Let $S_m$ be the collection of all training sets $s$ with size $m$.
- A hypothesis class $H$ is potentially learnable iff for all $\varepsilon (0 < \varepsilon < 1)$, $\delta (0 < \delta < 1)$, for all distributions $D$ on $X$ and for all concepts $c$ in $H$ there exists a positive integer $m_0 = m_0(\delta, \varepsilon)$ such that, whenever $m \geq m_0$ the following holds true:

$$
\Pr_{s \in S_m, D} \left[ H[s] \cap B_\varepsilon = \emptyset \right] > 1 - \delta
$$

- That means that every hypothesis in $H$ that agrees with all examples in $s$ has an error $\leq \varepsilon$, i.e., each consistent hypothesis on with respect to $s$ also has a true error $\leq \varepsilon$.
- I.e. there is no consistent hypothesis in $H$ that has an error $> \varepsilon$.
Overview

- Introduction/Computational Learning Theory
- PAC Learning Model
- Consistent Learning Algorithms and Potential Learnability
- Proofs
Potential learnability

Theorem 1:

If $H$ is potentially learnable and $L$ is a consistent learning algorithm for $H$ then $L$ is a PAC learning algorithm for $H$.
Let $S_m$ be the collection of all training sets $s$ with size $m$.

A hypothesis class $H$ is potentially learnable iff for all $\varepsilon (0 < \varepsilon < 1)$, $\delta (0 < \delta < 1)$, for all distributions $D$ on $X$ and for all concepts $c$ in $H$ there exists a positive integer $m_0 = m_0(\delta, \varepsilon)$ such that, whenever $m \geq m_0$ the following holds true:

That means that every hypothesis in $H$ that agrees with all examples in $s$ has an error $\leq \varepsilon$, i.e. each consistent hypothesis on with respect to $s$ also has a true error $\leq \varepsilon$.

\[
H[s] = \{h \in H \mid h(x_j) = c(x_j); 1 \leq j \leq m\} \quad (1)
\]

\[
B_{\varepsilon} = \{h \in H \mid error(h) > \varepsilon\} \quad (2)
\]

\[
Pr_{s \in S_m, D} [H[s] \cap B_{\varepsilon} = \emptyset] > 1 - \delta \quad (3)
\]

Theorem 1: If $H$ is potentially learnable and $L$ is a consistent learning algorithm for $H$ then $L$ is a PAC learning algorithm for $H$. 


Potential learnability

Theorem 2:

Any finite hypothesis space $H$ is potentially learnable.
Let $S_m$ be the collection of all training sets $s$ with size $m$.

A hypothesis class $H$ is potentially learnable iff for all $\varepsilon (0 < \varepsilon < 1)$, $\delta (0 < \delta < 1)$, for all distributions $D$ on $X$ and for all concepts $c$ in $H$ there exists a positive integer $m_0 = m_0(\delta, \varepsilon)$ such that, whenever $m \geq m_0$ the following holds true:

That means that every hypothesis in $H$ that agrees with all examples in $s$ has an error $\leq \varepsilon$, i.e. each consistent hypothesis on with respect to $s$ also has a true error $\leq \varepsilon$.

Theorem 2: Any finite hypothesis space $H$ is potentially learnable.

\[ H[s] = \{ h \in H \mid h(x_j) = c(x_j); 1 \leq j \leq m \} \quad (1) \]
\[ B_\varepsilon = \{ h \in H \mid error(h) > \varepsilon \} \quad (2) \]
\[ \Pr_{s \in S_m, D} \left[ H[s] \cap B_\varepsilon = \emptyset \right] > 1 - \delta \quad (3) \]
Potential learnability

That means that if you can show that a given learning algorithm is consistent and has a finite hypothesis space, the learning algorithm is a PAC learning algorithm.
THANK YOU!
Sources