Incomparable Degrees in PACi and PAC Learning

Gihanee Senadheera

senadheerad@winthrop.edu

Department of Mathematics
Winthrop University

New England Recursion and Definability Seminar 2023
Wellesley College

October 14, 2023

PAC Learning Model

- PAC stands for **P**robably **A**pproximately **C**orrect
- It is a Machine learning model.
- It was introduced by Leslie Valiant in 1984.

Why this is important

- PAC and PACi reducibilities give partial ordering and a degree structure.
- Helps to understand the structure of PAC learnability.
- If it is a linear ordering, then it has only one way of non-learnability.
- If incomparable degrees exist then there are at least two different ways of non-learnability.
- If there exists an embedding from known degrees to PACi or PAC degrees, then all the properties true for the known degree will be true for PACi or PAC degrees.

PAC Learning Model (Valiant 1984) Cont.

Definition

1. Let X be a set, called the *instance space*.

PAC Learning Model (Valiant 1984) Cont.

Definition

1. Let X be a set, called the *instance space*.
2. Let C be a subset of $P(X)$ the power set of X, called a *concept class*.

PAC Learning Model (Valiant 1984) Cont.

Definition

1. Let X be a set, called the *instance space*.
2. Let C be a subset of $P(X)$ the power set of X, called a *concept class*.
3. The elements of C are called *concepts*.

PAC Learning Model (Valiant 1984)

Definition

We say that C is *PAC Learnable* if and only if there is an algorithm L such that for every $c \in C$, every $\epsilon, \delta \in (0, \frac{1}{2})$ and every probability distribution $\mathfrak{D}$ on X, the algorithm L behaves as follows:

PAC Learning Model (Valiant 1984)

Definition

We say that C is *PAC Learnable* if and only if there is an algorithm L such that for every $c \in C$, every $\epsilon, \delta \in (0, \frac{1}{2})$ and every probability distribution $\mathfrak{D}$ on X, the algorithm L behaves as follows:

On input (ϵ, δ), the algorithm L will ask for some number n of examples, and will be given $\{(x_1, i_1), ..., (x_n, i_n)\}$ where x_k are independently randomly drawn from $\mathfrak{D}$ and $i_k = \chi_c(x_k)$.

PAC Learning Model (Valiant 1984)

Definition

We say that C is *PAC Learnable* if and only if there is an algorithm L such that for every $c \in C$, every $\epsilon, \delta \in (0, \frac{1}{2})$ and every probability distribution $\mathfrak{D}$ on X, the algorithm L behaves as follows:

On input (ϵ, δ), the algorithm L will ask for some number n of examples, and will be given $\{(x_1, i_1), ..., (x_n, i_n)\}$ where x_k are independently randomly drawn from $\mathfrak{D}$ and $i_k = \chi_c(x_k)$.

The algorithm will then output some $h \in C$ with probability at least $1 - \delta$ in $\mathfrak{D}$, the symmetric difference of h and c has the probability at most ϵ in $\mathfrak{D}$.

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

Given the inputs $\epsilon, \delta \in (0, \frac{1}{2})$

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

Given the inputs $\epsilon, \delta \in (0, \frac{1}{2})$

find m, large enough such that $(1-\epsilon)^m < \delta$ is satisfied.

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

Given the inputs $\epsilon, \delta \in (0, \frac{1}{2})$

find m, large enough such that $(1-\epsilon)^m < \delta$ is satisfied.

Ask for m examples. Denoted as $A = \{(x_1, i_1), ..., (x_m, i_m)\}$ where x_k are independently randomly drawn from $\mathfrak{D}$ and $i_k = \chi_c(x_k)$.

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

Given the inputs $\epsilon, \delta \in (0, \frac{1}{2})$

find m, large enough such that $(1 - \epsilon)^m < \delta$ is satisfied.

Ask for m examples. Denoted as $A = \{(x_1, i_1), ..., (x_m, i_m)\}$ where x_k are independently randomly drawn from $\mathfrak{D}$ and $i_k = \chi_c(x_k)$.

Define $B = \{x_k | (x_k, i_k) \in A \ and \ i_k = 1\}$. Define $h = inf \ B$.

PAC Learning Model Examples

Suppose X is the real line. Let C be the set of positive half lines then C is PAC learnable.

The set X is called the *instance space*, the set C is called the *concept class* and elements of C are called the *concepts*.

Given the inputs $\epsilon, \delta \in (0, \frac{1}{2})$

find m, large enough such that $(1-\epsilon)^m < \delta$ is satisfied.

Ask for m examples. Denoted as $A = \{(x_1, i_1), ..., (x_m, i_m)\}$ where x_k are independently randomly drawn from $\mathfrak{D}$ and $i_k = \chi_c(x_k)$.

Define $B = \{x_k | (x_k, i_k) \in A \ and \ i_k = 1\}$. Define $h = inf \ B$.

Return the hypothesis $H = (h, \infty)$.

PAC Learning Model Examples

Why does above mentioned algorithm works?

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

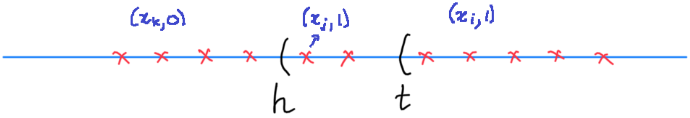

Figure 1: Training data error

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

Then $(t, \infty) \triangle (h, \infty) = (t, h)$.

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

Then $(t, \infty) \triangle (h, \infty) = (t, h)$.

Define b such that $\mathfrak{D}(t, b) < \epsilon$.

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

Then $(t, \infty) \triangle (h, \infty) = (t, h)$.

Define b such that $\mathfrak{D}(t, b) < \epsilon$.

We can show that probability of $h < b$ is less than δ.

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

Then $(t, \infty) \triangle (h, \infty) = (t, h)$.

Define b such that $\mathfrak{D}(t, b) < \epsilon$.

We can show that probability of $h < b$ is less than δ.

This probability is bounded by $(1 - \epsilon)^m$ and $(1 - \epsilon)^m < \delta$.

PAC Learning Model Examples

Why does above mentioned algorithm works?

Suppose the target is (t, ∞).

Notice that $t \leq h$. Otherwise, our training data is wrong.

Then $(t, \infty) \triangle (h, \infty) = (t, h)$.

Define b such that $\mathfrak{D}(t, b) < \epsilon$.

We can show that probability of $h < b$ is less than δ.

This probability is bounded by $(1 - \epsilon)^m$ and $(1 - \epsilon)^m < \delta$.

Since one example missing (t, b) has the probability $(1 - \epsilon)$.

Then m examples missing (t, b) has the probability $(1 - \epsilon)^m$.

More Examples

Example

Suppose X is the real line.

- Let C be the set of positive half lines then C is PAC learnable.
- Let C be the set of negative half lines then C is PAC learnable.
- Let C be the set of intervals then C is PAC learnable.

Suppose X is $\mathbb{R}^2$.

- Let C be the set of axis aligned rectangles then C is PAC learnable.
- Let C be the set of convex d-gons then C is PAC learnable for any d.

Suppose $X = \mathbb{R}^d$.

- Let C be the set of linear-half spaces. Then C is PAC learnable.

Weakly effective concept class

Definition

A weakly effective concept class is a computable enumeration $\varphi_e : \mathbb{N} \to \mathbb{N}$ such that $\varphi_e(n)$ is a Π^0_1 index for a Π^0_1 tree $T_{e,n}$.

An effective concept class

Definition

An effective concept class is a weakly effective concept class $\varphi_e(n)$ such that for each n, the set c_n of paths through $T_{e,n}$ is computable in the sense that there is a computable function $f_{c_n}(\sigma, r) : 2^{<\omega} \times \mathbb{Q} \to \{0, 1\}$ such that

$$f_{c_n}(\sigma, r) = \begin{cases} 1 & \text{if } B_r(\sigma) \cap c_n \neq \varnothing \\ 0 & \text{if } B_{2r}(\sigma) \cap c_n = \varnothing \\ 0 \textit{ or } 1 & \text{otherwise} \end{cases}$$

where $B_r(\sigma)$ is the set of all paths that either extend σ or first differ from it at the $-\lceil lg(r) \rceil$ place or later.

Figure

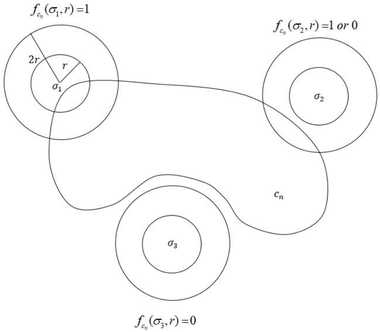

Figure 2: Computable function $f_{c_n}(\sigma, r)$

An effective concept class

Definition

An effective concept class is a weakly effective concept class $\varphi_e(n)$ such that for each n, the set c_n of paths through $T_{e,n}$ is computable in the sense that there is a computable function $f_{c_n}(\sigma, r) : 2^{<\omega} \times \mathbb{Q} \to \{0, 1\}$ such that

$$f_{c_n}(\sigma, r) = \begin{cases} 1 & \text{if } B_r(\sigma) \cap c_n \neq \varnothing \\ 0 & \text{if } B_{2r}(\sigma) \cap c_n = \varnothing \\ 0 \textit{ or } 1 & \text{otherwise} \end{cases}$$

where $B_r(\sigma)$ is the set of all paths that either extend σ or first differ from it at the $-\lceil lg(r) \rceil$ place or later.

We can say that an effective concept class is a set of Π^0_1 classes. A Π^0_1 class is expressed as the set of infinite paths through a computable tree or the set of infinite paths through a Π^0_1 tree.

Example

Example

The class C of linear half-spaces in $\mathbb{R}^d$ bounded by hyper-planes with computable coefficients is an effective concept class.

Example

Example

The class C of linear half-spaces in $\mathbb{R}^d$ bounded by hyper-planes with computable coefficients is an effective concept class.

Since the distance of a point from the boundary can be computed, the linear half-spaces with computable coefficients is a computable set.

Consider $\mathbb{R}^2$. There are algorithms to compute the distance from a point to a line. The line has computable coefficients. Here no need to use the full precision reals.

PACi Reducibility (Calvert 2018)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PACi reduces to D, which we denote by $C \leq_{PACi} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

PACi Reducibility (Calvert 2018)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PACi reduces to D, which we denote by $C \leq_{PACi} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional

PACi Reducibility (Calvert 2018)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PACi reduces to D, which we denote by $C \leq_{PACi} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional
2. h is a computable function on indices

PACi Reducibility (Calvert 2018)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PACi reduces to D, which we denote by $C \leq_{PACi} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional
2. h is a computable function on indices
3. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$.

PACi Reducibility (Calvert 2018)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PACi reduces to D, which we denote by $C \leq_{PACi} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional
2. h is a computable function on indices
3. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$.

The "i" indicates the independence of this definition from size and computation time.

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional and computable in polynomial time,

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional and computable in polynomial time,
2. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional and computable in polynomial time,
2. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$
3. There is a polynomial p such that for any $x \in X$ of size n, the element $g(x)$ is of size at most $p(n)$, and

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional and computable in polynomial time,
2. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$
3. There is a polynomial p such that for any $x \in X$ of size n, the element $g(x)$ is of size at most $p(n)$, and
4. There is a polynomial q such that for every $c \in C$ of size n, the concept $h(c)$ is of size at most $q(n)$.

Learnability and Reducibility

Theorem

Let C and D be concept classes. Then if C PAC-reduces to D, and D is PAC learnable, C is PAC learnable.

Learnability and Reducibility

Theorem

Let C and D be concept classes. Then if C PAC-reduces to D, and D is PAC learnable, C is PAC learnable.

Proof:

Let L' be the learning algorithm for D.
We use L' to learn C.

Learnability and Reducibility

Theorem

Let C and D be concept classes. Then if C PAC-reduces to D, and D is PAC learnable, C is PAC learnable.

Proof:

Let L' be the learning algorithm for D.
We use L' to learn C.
For a random example (x, c) of the unknown target concept $c \in C$, we can compute the labeled example $(g(x), h(c))$ and give it to L'.

Learnability and Reducibility

Theorem

Let C and D be concept classes. Then if C PAC-reduces to D, and D is PAC learnable, C is PAC learnable.

Proof:

Let L' be the learning algorithm for D.
We use L' to learn C.
For a random example (x, c) of the unknown target concept $c \in C$, we can compute the labeled example $(g(x), h(c))$ and give it to L'.
If the instance $x \in X$ are drawn according to $\mathfrak{D}$, then the instances $g(x) \in Y$ are drawn according to some induced distribution $\mathfrak{D}'$.

Proof cont.

Although we do not know the target concept c, our definition of reduction guarantees that the computed examples $(g(x), h(c))$ are consistent with some $d \in D$, and thus L' will output a hypothesis t' that has an error at most ϵ with respect to $\mathfrak{D}'$.

Proof cont.

Although we do not know the target concept c, our definition of reduction guarantees that the computed examples $(g(x), h(c))$ are consistent with some $d \in D$, and thus L' will output a hypothesis t' that has an error at most ϵ with respect to $\mathfrak{D}'$.

Our hypothesis for c becomes $t(x) = t'(g(x))$, which has at most ϵ error with respect to $\mathfrak{D}$.

Kolmogorov Complexity

As the size of the effective concept class, we can use the Kolmogorov complexity.

Kolmogorov Complexity

As the size of the effective concept class, we can use the Kolmogorov complexity.
The quantification of the amount of absolute information in individual objects that are invariant up to an additive constant is known as the Kolmogorov Complexity.

Example

Let $X = 2^{\omega}$ be the instance spaces.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros.

Example

Let $X = 2^\omega$ be the instance spaces.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros.

In this sequence, each of these trees c_n consists of a single infinite path. Let C be the concept class consisting of the above sequence of trees.

Example

Let $X = 2^\omega$ be the instance spaces.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros.

In this sequence, each of these trees c_n consists of a single infinite path. Let C be the concept class consisting of the above sequence of trees. Since c_n has only a single infinite path, identify each tree c_n by this single path.

Example

Let $X = 2^\omega$ be the instance spaces.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros.

In this sequence, each of these trees c_n consists of a single infinite path. Let C be the concept class consisting of the above sequence of trees. Since c_n has only a single infinite path, identify each tree c_n by this single path.

We calculate the Kolmogorov Complexity of the initial segment of the path of the tree c_n.

Example

Let $X = 2^\omega$ be the instance spaces.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros.

In this sequence, each of these trees c_n consists of a single infinite path. Let C be the concept class consisting of the above sequence of trees. Since c_n has only a single infinite path, identify each tree c_n by this single path.

We calculate the Kolmogorov Complexity of the initial segment of the path of the tree c_n.

Thus size of c_n is given by $size(c_n) = K((c_n)_{1:n}|n) \leq k$ where k is a constant. This is possible since the finite segment of the tree is computable.

- Observe that the empty concept class on the empty instance space is reducible to any other concept class.

PAC Reducibility (Kearns, Vazirani 1994)

Definition

Let C be an effective concept class over the instance space X and D an effective concept class over the instance space Y.

We say that C PAC reduces to D, denoted $C \leq_{PAC} D$ exactly when there are functions $g : X \to Y$ and $h : C \to D$ such that

1. g is a Turing functional and computable in polynomial time,
2. for all $x \in X$ and for all $c \in C$, we have $x \in c$ if and only if $g(x) \in h(c)$
3. There is a polynomial p such that for any $x \in X$ of size n, the element $g(x)$ is of size at most $p(n)$, and
4. There is a polynomial q such that for every $c \in C$ of size n, the concept $h(c)$ is of size at most $q(n)$.

- Observe that the empty concept class on the empty instance space is reducible to any other concept class.
- Also any concept class is reducible to itself through the identity function.

- Observe that the empty concept class on the empty instance space is reducible to any other concept class.
- Also any concept class is reducible to itself through the identity function.
- We can infer that there are $\leq_{PAC}$ incomparable concept classes since there are continuum many concept classes on a countably infinite instance space.

- Observe that the empty concept class on the empty instance space is reducible to any other concept class.
- Also any concept class is reducible to itself through the identity function.
- We can infer that there are $\leq_{PAC}$ incomparable concept classes since there are continuum many concept classes on a countably infinite instance space.
- This degree structure is analogous to Turing degrees and their structures. So, we can expect the effective concept classes to behave in a similar manner to computably enumerable degrees.

PACi Degree and PAC Degree

Definition

We say $C \sim_{PACi} D$ if $C \leq_{PAC_i} D$ and $D \leq_{PAC_i} C$, the relation $\sim$ is an equivalence relation. The PACi degree of concept class C is $deg(C) = \{D : D \sim_{PACi} C\}$

PACi Degree and PAC Degree

Definition

We say $C \sim_{PACi} D$ if $C \leq_{PAC_i} D$ and $D \leq_{PAC_i} C$, the relation $\sim$ is an equivalence relation. The PACi degree of concept class C is $deg(C) = \{D : D \sim_{PACi} C\}$

Definition

We say $C \sim_{PAC} D$ if $C \leq_{PAC} D$ and $D \leq_{PAC} C$, the relation $\sim$ is an equivalence relation. The PAC degree of concept class C is $deg(C) = \{D : D \sim_{PAC} C\}$

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

The positive half lines are bounded below. If positive half lines are bounded below by a computable lower bound then the concept class C is an effective concept class.

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

The positive half lines are bounded below. If positive half lines are bounded below by a computable lower bound then the concept class C is an effective concept class.

Similarly we can show that C' is also an effective concept class.

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

The positive half lines are bounded below. If positive half lines are bounded below by a computable lower bound then the concept class C is an effective concept class.

Similarly we can show that C' is also an effective concept class.

Define $g : \mathbb{R} \to \mathbb{R}$ by $g(x) = -x$ and $h : C \to C'$ by $h((a, \infty)) = (-\infty, -a)$.

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

The positive half lines are bounded below. If positive half lines are bounded below by a computable lower bound then the concept class C is an effective concept class.

Similarly we can show that C' is also an effective concept class.

Define $g : \mathbb{R} \to \mathbb{R}$ by $g(x) = -x$ and $h : C \to C'$ by $h((a, \infty)) = (-\infty, -a)$.

Now we can show that *for all* $x \in \mathbb{R}$ and *for all* positive half lines $c = (a, \infty)$ in C we have $x \in c$ iff $g(x) \in h(c)$ where $h(c)$ is a negative half line.

Examples

Example

Let $X = X' = \mathbb{R}$ be the two instance spaces.
Let C be the set of positive half lines and C' be the set of negative half lines.

The positive half lines are bounded below. If positive half lines are bounded below by a computable lower bound then the concept class C is an effective concept class.

Similarly we can show that C' is also an effective concept class.

Define $g : \mathbb{R} \to \mathbb{R}$ by $g(x) = -x$ and $h : C \to C'$ by $h((a, \infty)) = (-\infty, -a)$.

Now we can show that *for all* $x \in \mathbb{R}$ and *for all* positive half lines $c = (a, \infty)$ in C we have $x \in c$ iff $g(x) \in h(c)$ where $h(c)$ is a negative half line.

This will give us $C \leq_{PACi} C'$. With appropriate functionals, we can show that $C' \leq_{PACi} C$. Thus $C \sim C'$.

PACi Incomparable Degrees

Theorem

There exist effective concept classes C and D over the instance space $X = Y = 2^\omega$ such that C does not PACi reduce to D and also D does not PACi reduce to C. (i.e. $C \nleq_{PACi} D$ and $D \nleq_{PACi} C$).

Sketch of Proof:

The two concept classes C and D are constructed over the instance spaces X and Y respectively. Let $\{h_t | t \in \mathbb{N}\}$ enumerate the set of all partial computable functions from $\mathbb{N} \to \mathbb{N}$.

Requirements : R_{2t} : there exists $c \in C$ such that $h_t(c) \notin D$

R_{2t+1} : there exists $d \in D$ such that $h_t(d) \notin C$

Satisfying one requirement

Consider R_{2t}.

To satisfy the requirement R_{2t} we will attach a potential witness c: a concept, to R_{2t} which is not yet enumerated in C.

We choose c such that c is an index for a tree.

Satisfying one requirement

Consider R_{2t}.

To satisfy the requirement R_{2t} we will attach a potential witness c: a concept, to R_{2t} which is not yet enumerated in C.

We choose c such that c is an index for a tree.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros. In this sequence each of these trees c_n, consists of a single infinite path.

Satisfying one requirement cont.

We will use B_s to keep track of the set of all trees that we plan not to enumerate in C.

We will use A_s to keep track of the set of all trees that we plan not to enumerate in D.

Satisfying one requirement cont.

We will use B_s to keep track of the set of all trees that we plan not to enumerate in C.

We will use A_s to keep track of the set of all trees that we plan not to enumerate in D.

At stage s pick a c such that $c \notin B_s$ and $c \notin C_s$ and $h_t(c) \notin D_s$.

Satisfying one requirement cont.

We will use B_s to keep track of the set of all trees that we plan not to enumerate in C.

We will use A_s to keep track of the set of all trees that we plan not to enumerate in D.

At stage s pick a c such that $c \notin B_s$ and $c \notin C_s$ and $h_t(c) \notin D_s$.

We will enumerate c in C_s and enumerate $h_t(c)$ in A_s.

Thus we restrain the tree $h_t(c)$ from later entering to D by checking the condition, $d \notin A_{s+1}$, and selecting the next indexed concept from the sequence.

Satisfying one requirement cont.

We have $C \nleq_{PAC_i} D$.

Satisfying one requirement cont.

We have $C \not\leq_{PAC_i} D$.

The strategy for R_{2t+1} is the same but with roles of C_s and D_s reversed.

Satisfying one requirement cont.

We have $C \not\leq_{PAC_i} D$.

The strategy for R_{2t+1} is the same but with roles of C_s and D_s reversed.

We call the sets A and B restraint sets.

Construction of the two concept classes, C and D.

Let $X = Y = 2^{\omega}$.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros. In this sequence each of these trees c_n, consists of a single infinite path.

Construction of the two concept classes, C and D.

Let $X = Y = 2^\omega$.

Let $\{c_n\}_{n=1}^{\infty}$ be a family of trees, where c_n has n number of 1's and followed by zeros. In this sequence each of these trees c_n, consists of a single infinite path.

Stage $s = 0$: Let $C_0 = D_0 = \phi$ and $A_0 = B_0 = \phi$.

Stage $s + 1$:

Requirement R_{2t} requires attention, if we have not enumerated a witness, $c \in C$ for the requirement R_{2t}.

Requirement R_{2t+1} requires attention, if we have not enumerated a witness, $d \in D$ for the requirement R_{2t+1}.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$**.** Now R_{2t} receives attention.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$**.** Now R_{2t} receives attention.

Pick a tree c from the family $\{c_n\}$ defined above with $c = c_n$ for some $n < s$ such that $c \notin C_s$ and $c \notin B_s$ and $h_t(c) \notin D_s$.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$**.** Now R_{2t} receives attention.

Pick a tree c from the family $\{c_n\}$ defined above with $c = c_n$ for some $n < s$ such that $c \notin C_s$ and $c \notin B_s$ and $h_t(c) \notin D_s$.

If such a c exists enumerate $c \in C_{s+1}$ and $h_t(c)$ in A_{s+1}. If such a c does not exist then do nothing.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$**.** Now R_{2t} receives attention.

Pick a tree c from the family $\{c_n\}$ defined above with $c = c_n$ for some $n < s$ such that $c \notin C_s$ and $c \notin B_s$ and $h_t(c) \notin D_s$.

If such a c exists enumerate $c \in C_{s+1}$ and $h_t(c)$ in A_{s+1}. If such a c does not exist then do nothing.

Suppose $i = 2t + 1$**.** Now R_{2t+1} receives attention.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$. Now R_{2t} receives attention.

Pick a tree c from the family $\{c_n\}$ defined above with $c = c_n$ for some $n < s$ such that $c \notin C_s$ and $c \notin B_s$ and $h_t(c) \notin D_s$.

If such a c exists enumerate $c \in C_{s+1}$ and $h_t(c)$ in A_{s+1}. If such a c does not exist then do nothing.

Suppose $i = 2t + 1$. Now R_{2t+1} receives attention.

Pick a tree d from the family $\{c_n\}$ with $d = c_n$ for some $n < s$ such that $d \notin D_s$ and $d \notin A_s$ and $h_t(d) \notin C_s$.

Construction of the two concept classes, C and D.

Choose least $i \leq s$ such that R_i requires attention.

Suppose $i = 2t$. Now R_{2t} receives attention.

Pick a tree c from the family $\{c_n\}$ defined above with $c = c_n$ for some $n < s$ such that $c \notin C_s$ and $c \notin B_s$ and $h_t(c) \notin D_s$.

If such a c exists enumerate $c \in C_{s+1}$ and $h_t(c)$ in A_{s+1}. If such a c does not exist then do nothing.

Suppose $i = 2t + 1$. Now R_{2t+1} receives attention.

Pick a tree d from the family $\{c_n\}$ with $d = c_n$ for some $n < s$ such that $d \notin D_s$ and $d \notin A_s$ and $h_t(d) \notin C_s$.

If d exists then enumerate $d \in D_{s+1}$ and $h_t(d)$ in B_{s+1}. Do nothing if such a d does not exist.

Construction of the two concept classes, C and D.

At each stage, we will be checking through a finite amount of trees in C_s, D_s, A_s, or B_s.

When a requirement is satisfied at stage s it will remain satisfied forever. Thus we have $C \not\leq_{PAC_i} D$ and $D \not\leq_{PAC_i} C$

PAC Incomparable Degrees

Theorem

There exist effective concept classes C and D over the instance space $X = Y = 2^\omega$ such that C does not PAC reduce to D and also D does not PAC reduce to C. (i.e. $C \nleq_{PAC} D$ and $D \nleq_{PAC} C$).

Sketch of the proof:

Requirements :

R_{2t} : there exists $\sigma \in c$ where $c \in C$ s.t. $g_t(\sigma) \notin d, \forall\ d \in D$

R_{2t+1} : there exists $\tau \in d$ where $d \in D$ s.t. $g_t(\tau) \notin c, \forall\ c \in C$

A Greatest Effective Concept Class

We can define a jump for the effective concept classes also.

A Greatest Effective Concept Class

We can define a jump for the effective concept classes also.

We can show that $deg(A \oplus B)$ is the least upper bound for the $deg(A)$ and $deg(B)$ in $(P, <)$ where P is the class of all PAC degrees. The degree P forms a partially ordered set under relation $deg(A) \leq deg(B)$.

A Greatest Effective Concept Class

We can define a jump for the effective concept classes also.

We can show that $deg(A \oplus B)$ is the least upper bound for the $deg(A)$ and $deg(B)$ in $(P, <)$ where P is the class of all PAC degrees. The degree P forms a partially ordered set under relation $deg(A) \leq deg(B)$.

We can show that $\bigoplus_{i \in \omega} C_i$ is an effective concept class.

$$(\oplus_{i \in \omega} C_i)_e = \{1^i 0\sigma \ |\sigma \in (C_i)_e, i, e \in \omega\}$$

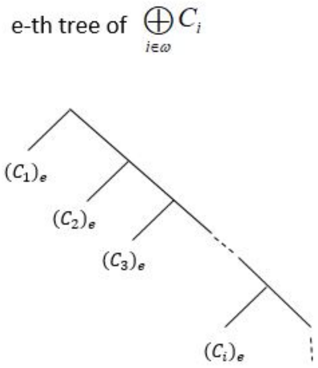

Figure 3: The e-th tree of the concept class $\bigoplus_{i\in\omega} C_i$

A Greatest Effective Concept Class

Theorem

$\bigoplus_{i \in \omega} C_i$ is an effective concept class.

A Greatest Effective Concept Class

Theorem

$\bigoplus_{i \in \omega} C_i$ is an effective concept class.

Proof:
Since each effective concept class, C_i is computable, there exist a computable $f^i_{c_n}(\sigma, r)$ for each i. Then define $f_{c_n}(\sigma, r) : 2^{<\omega} \times \mathbb{Q} \to \{0, 1\}$

$$f_{c_n}(\sigma, r) = \begin{cases} 1 & \text{if } \sigma = 1^{|\sigma|} \\ f^i_{c_n}(\tau, r') & \text{if } \sigma = 1^i 0 \tau \end{cases}$$

A Greatest Effective Concept Class

Theorem

deg $\left(\bigoplus_{j\in\omega} A_j\right)$ *is the least upper bound for* $\{deg(A_y)|y \in \omega\}$.

A Greatest Effective Concept Class

Theorem

$deg\left(\bigoplus_{j\in\omega} A_j\right)$ is the least upper bound for $\{deg(A_y)|y\in\omega\}$.

Proof:

$$g_y : 2^\omega \to 2^\omega \qquad and \qquad h_y : A_y \to \bigoplus_{j\in\omega} A_j \qquad by$$

$$\sigma \mapsto 1^y 0\sigma \qquad and \qquad (A_y)_e \mapsto \left(\bigoplus_{j\in\omega} A_j\right)_e \tag{1}$$

Hence $A_y \leq_{PACi} \bigoplus_{j\in\omega} A_j$ for all y. Therefore

$$deg(A_y) \leq deg\left(\bigoplus_{j\in\omega} A_j\right) \quad for\ all\ y \tag{2}$$

Embedding 1-degrees to PACi degrees

Let $\mathcal{P}$ be the PACi degrees of effective concept class. Let $\mathcal{C}$ be the 1-degree of c.e. sets.

Is there $\phi : \mathcal{C} \to \mathcal{P}$ such that $\mathbf{a} \leq_1 \mathbf{b}$ iff $\phi(\mathbf{a}) \leq_{PACi} \phi(\mathbf{b})$?

Embedding 1-degrees to PACi degrees

Let $\mathcal{P}$ be the PACi degrees of effective concept class. Let $\mathcal{C}$ be the 1-degree of c.e. sets.

Is there $\phi : \mathcal{C} \to \mathcal{P}$ such that $\mathbf{a} \leq_1 \mathbf{b}$ iff $\phi(\mathbf{a}) \leq_{PACi} \phi(\mathbf{b})$?

Sketch of the proof:

There are c.e. sets $W_{e_1} \in \mathbf{a}$ and $W_{e_2} \in \mathbf{b}$.

Embedding 1-degrees to PACi degrees

Let $\mathcal{P}$ be the PACi degrees of effective concept class. Let $\mathcal{C}$ be the 1-degree of c.e. sets.

Is there $\phi : \mathcal{C} \to \mathcal{P}$ such that $\mathbf{a} \leq_1 \mathbf{b}$ iff $\phi(\mathbf{a}) \leq_{PACi} \phi(\mathbf{b})$?

Sketch of the proof:

There are c.e. sets $W_{e_1} \in \mathbf{a}$ and $W_{e_2} \in \mathbf{b}$.

$W_{e_1} \mapsto \{c_n | n \in W_{e_1}\} = C_{\phi(e_1)}$ where $c_n = 1^n\bar{0}$.

$W_{e_2} \mapsto \{c_n | n \in W_{e_2}\} = C_{\phi(e_2)}$.

Continuing sketch of the proof:

Since $W_{e_1} \leq_1 W_{e_2}$ there is a 1-1 function ψ.

$$\begin{tikzcd} \psi : W_{e_1} \arrow[r] & W_{e_2} \\ n \arrow[r] \arrow[d, "\phi"'] & \psi(n) \arrow[d, "\phi"] \\ c_n \arrow[r] & c_{\psi(n)} \end{tikzcd} \tag{3}$$

Continuing sketch of the proof:

Since $W_{e_1} \leq_1 W_{e_2}$ there is a 1-1 function ψ.

$$\begin{tikzcd} \psi : W_{e_1} \arrow[r] & W_{e_2} \\ n \arrow[r] \arrow[d, "\phi"'] & \psi(n) \arrow[d, "\phi"] \\ c_n \arrow[r] & c_{\psi(n)} \end{tikzcd} \tag{3}$$

$$g : X = 2^\omega \to Y = 2^\omega \qquad \textit{and} \qquad h : C_{\phi(e_1)} \to C_{\phi(e_2)}$$

$$\sigma \mapsto \begin{cases} \tau & \text{if } \sigma = c_n \textit{ where } \tau = c_{\psi(n)} \\ \sigma & \text{otherwise} \end{cases} \qquad c_n \mapsto c_{\psi(n)} \tag{4}$$

Lemma

If $W_{e_1} \not\leq_1 W_{e_2}$ *then* $C_{\phi(e_1)} \not\leq_{PACi} C_{\phi(e_2)}$.

Sketch of the proof:

we will use the contrapositive of this statement. That is, if $C_{\phi(e_1)} \leq_{PACi} C_{\phi(e_2)}$ then $W_{e_1} \leq_1 W_{e_2}$.

Since $C_{e_1} \leq_{PACi} C_{e_2}$ there exist g and h but we will consider only h.

$$\begin{array}{ccc} h : C_{e_1} & \longrightarrow & C_{e_2} \\ c_n & \longrightarrow & h(c_n) \\ \phi' \downarrow & & \downarrow \phi' \\ n & \longrightarrow & h(n) \end{array} \tag{5}$$

Sketch of the proof continues

To obtain the 1-reduction between $W_{\phi'(e_1)}$ and $W_{\phi'(e_2)}$ define ψ as follows. Each $n \in W_{\phi'(e_1)}$ will be mapped to $h(n)$ as in Equation 6.

$$\begin{aligned} \psi : W_{\phi'(e_1)} &\rightarrow W_{\phi'(e_2)} \\ n &\longmapsto \ h(n) \end{aligned} \tag{6}$$

Thank you!