# Computational Learning Theory
# FoPPS Logic and Learning School

James Worrell
University of Oxford

# What is Machine Learning?

# What is Learning Learning?

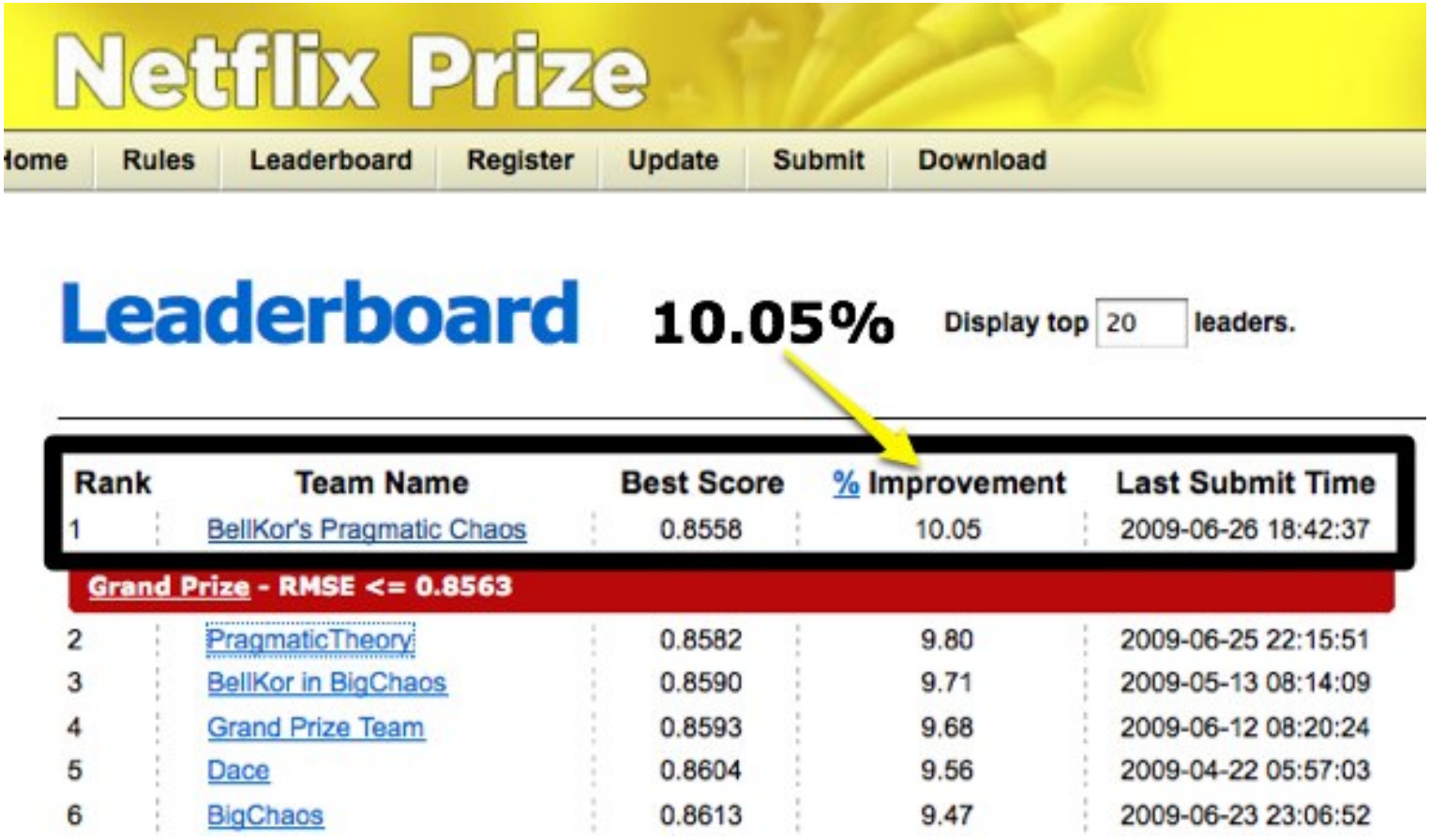
Term coined by Arthur Samuel in 1959:

# What is Learning Learning?

Term coined by Arthur Samuel in 1959:

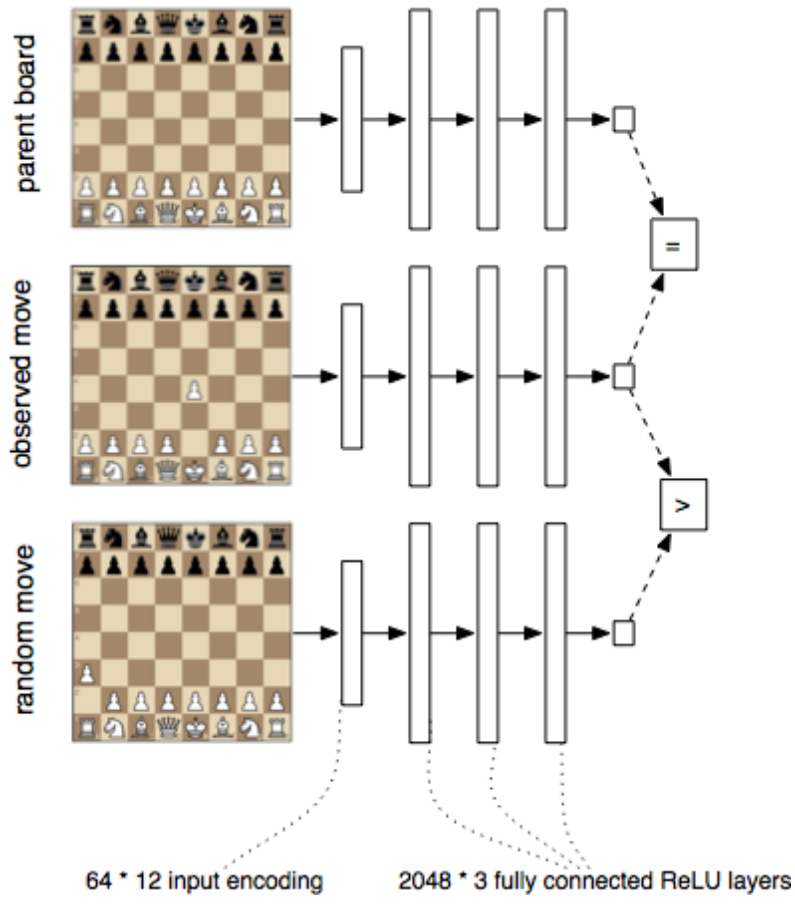
Have a computer solve problems by learning from data rather than being explicitly programmed.

# Filtering




Best algorithm for predicting user ratings, based on previous ratings, without any other information about the users or films.

# Planning and Strategy




Neural network trained to GM level using database of chess positions

# Machine Learning Overview

- ***Classification:*** assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;

# Machine Learning Overview

- ***Classification:*** assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;
- ***Regression:*** predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;

# Machine Learning Overview

- ***Classification:*** assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;
- ***Regression:*** predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;
- ***Clustering:*** partition a set of items into classes, e.g., identify communities among users of a social network;

# Machine Learning Overview

- ***Classification:*** assign a category to each data item, e.g., classify newspaper articles according to the topics *current events*, *sports*, *politics*, *entertainment*, *business*, and *other*;
- ***Regression:*** predict a numerical value for each data item, e.g., predict the value of a stock given various data about a company;
- ***Clustering:*** partition a set of items into classes, e.g., identify communities among users of a social network;
- ***Ranking:*** order items according to several criteria, e.g., order the results of a search query by relevance to a particular user.

# What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

# What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

> . . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them

# What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

. . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them

. . . what types of guarantees we might hope to achieve (error bounds, complexity bounds)

# What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

. . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them

. . . what types of guarantees we might hope to achieve (error bounds, complexity bounds)

. . . why particular algorithms may or may not perform well under various conditions

# What is Learning Theory?

The goal of learning theory is to develop and analyse formal models that help us understand . . .

. . . what concepts we can hope to learn efficiently, and how much data is necessary to learn them

. . . what types of guarantees we might hope to achieve (error bounds, complexity bounds)

. . . why particular algorithms may or may not perform well under various conditions

**Applications to and connections with logic and verification!**

# This Mini-Course

- Definition of the PAC learning model

# This Mini-Course

- Definition of the PAC learning model
- VC Dimension with examples (neural nets and definable families of sets)

# This Mini-Course

- Definition of the PAC learning model
- VC Dimension with examples (neural nets and definable families of sets)
- Finite VC dimension is equivalent to PAC learnability

# This Mini-Course

- Definition of the PAC learning model
- VC Dimension with examples (neural nets and definable families of sets)
- Finite VC dimension is equivalent to PAC learnability
- Sample compression schemes and Littlestone-Warmuth conjecture

# This Mini-Course

- Definition of the PAC learning model
- VC Dimension with examples (neural nets and definable families of sets)
- Finite VC dimension is equivalent to PAC learnability
- Sample compression schemes and Littlestone-Warmuth conjecture
- Concept classes that are hard to learn (time permitting)

# This Mini-Course

- Definition of the PAC learning model
- VC Dimension with examples (neural nets and definable families of sets)
- Finite VC dimension is equivalent to PAC learnability
- Sample compression schemes and Littlestone-Warmuth conjecture
- Concept classes that are hard to learn (time permitting)
- Learning with membership queries

# The PAC Model

# The Basic Set Up

- A learning problem is specified by an **input space** $\mathcal{X}$ and **concept class** $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$.

# The Basic Set Up

- A learning problem is specified by an **input space** $\mathcal{X}$ and **concept class** $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$.
- An instance is determined by an unknown distribution $D$ on $\mathcal{X}$ and **target concept** $c \in \mathcal{C}$.

# The Basic Set Up

- A learning problem is specified by an **input space** $\mathcal{X}$ and **concept class** $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$.
- An instance is determined by an unknown distribution $D$ on $\mathcal{X}$ and **target concept** $c \in \mathcal{C}$.
- Sample **training data** $S \in \mathcal{X}^m$ i.i.d. from $D$

# The Basic Set Up

- A learning problem is specified by an **input space** $\mathcal{X}$ and **concept class** $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$.
- An instance is determined by an unknown distribution $D$ on $\mathcal{X}$ and **target concept** $c \in \mathcal{C}$.
- Sample **training data** $S \in \mathcal{X}^m$ i.i.d. from $D$
- The desired output is a **hypothesis** $h \in \{0,1\}^{\mathcal{X}}$ such that

$$\mathrm{err}(h) \stackrel{\mathrm{def}}{=} \Pr_{x \sim D}\big(h(x) \neq c(x)\big).$$

  is "small".

# Example - Spam Filtering

```
Date:  Mon, 12 Oct 2015 19:36:45
From:  FSTTCS 2015
Subject:  Paper Submission 25
```
Not spam

```
Date:  13 Oct 2015 13:39:14
From:  HITACHI Personal Finance
Your Latest Money Saving Tips
```
Spam

```
Date:  Thu, 17 Sep 2015 05:00:12
From:  registration@it.ox.ac.uk
Subject:  Nexus mailbox expiry 01/10/15
```
???

# Example - Spam Filtering

Represent data as set of **labelled feature vectors**:

| Bad Spelling | Attachment? | "Ben" | "Viagra" | **Spam?** |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | **1** |
| 0 | 1 | 1 | 0 | **1** |
| 0 | 1 | 0 | 1 | **0** |
| 1 | 1 | 1 | 0 | **0** |
| 0 | 0 | 1 | 1 | **0** |

# Example - Spam Filtering

Represent data as set of **labelled feature vectors**:

| Bad Spelling | Attachment? | "Ben" | "Viagra" | **Spam?** |
|---|---|---|---|---|
| 0 | 1 | 1 | 1 | **1** |
| 0 | 1 | 1 | 0 | **1** |
| 0 | 1 | 0 | 1 | **0** |
| 1 | 1 | 1 | 0 | **0** |
| 0 | 0 | 1 | 1 | **0** |

Input space is $\mathcal{X} = \{0,1\}^4$ and concept class $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ is set of linear classifiers, e.g.,

$$2 \cdot \mathrm{BadSpell} - 3 \cdot \mathrm{Ben} + 1 \cdot \mathrm{Viagra} > 2 \ ?$$

# The PAC Model

**PAC** : **P**robably **A**pproximately **C**orrect

# The PAC Model

**PAC** : **P**robably **A**pproximately **C**orrect

Given target concept $c \in \mathcal{C}$, define $L_{\mathcal{C}}(m)$ to be the collection of **labelled samples** $(S, c|_S)$ where $S \in \mathcal{X}^m$.

# The PAC Model

**PAC** : **P**robably **A**pproximately **C**orrect

Given target concept $c \in \mathcal{C}$, define $L_{\mathcal{C}}(m)$ to be the collection of **labelled samples** $(S, c|_S)$ where $S \in \mathcal{X}^m$.

We say that $\mathcal{C}$ is **PAC learnable with sample complexity** $m$, **accuracy** $\varepsilon$, **and confidence** $\delta$ if there is a **learning map**

$$H : L_{\mathcal{C}}(m) \to \{0,1\}^{\mathcal{X}}$$

s.t.

# The PAC Model

**PAC** : **P**robably **A**pproximately **C**orrect

Given target concept $c \in \mathcal{C}$, define $L_{\mathcal{C}}(m)$ to be the collection of **labelled samples** $(S, c|_S)$ where $S \in \mathcal{X}^m$.

We say that $\mathcal{C}$ is **PAC learnable with sample complexity** $m$, **accuracy** $\varepsilon$, **and confidence** $\delta$ if there is a **learning map**

$$H : L_{\mathcal{C}}(m) \to \{0,1\}^{\mathcal{X}}$$

s.t. for any target concept $c \in \mathcal{C}$ and distribution $D$ on $\mathcal{X}$,

$$\Pr_{S \sim D^m} \left( \mathrm{err}(H(S, c|_S)) \leq \varepsilon \right) \geq 1 - \delta \, .$$

# Remarks on the Definition

- We assume the existence of a target concept

# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting

# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting
  - Say nothing about model selection, bias-variance trade-off, SRM, regularization,...

# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting
  - Say nothing about model selection, bias-variance trade-off, SRM, regularization,...
- The number of examples does not depend on the distribution on input space: the model is **distribution independent**.

# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting
  - Say nothing about model selection, bias-variance trade-off, SRM, regularization,...
- The number of examples does not depend on the distribution on input space: the model is **distribution independent**.
- The definition allows for **improper learning**. It says nothing about the form or representation of the hypothesis.

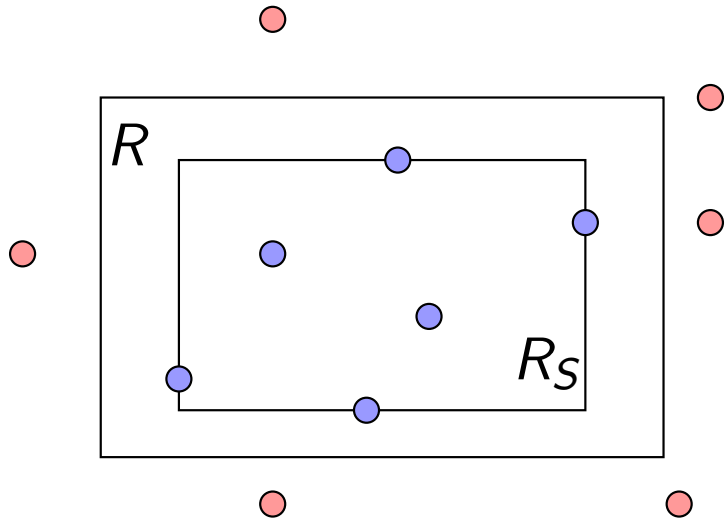
# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting
  - Say nothing about model selection, bias-variance trade-off, SRM, regularization,...
- The number of examples does not depend on the distribution on input space: the model is **distribution independent**.
- The definition allows for **improper learning**. It says nothing about the form or representation of the hypothesis.
- PAC model presents learning as a problem of **prediction**.

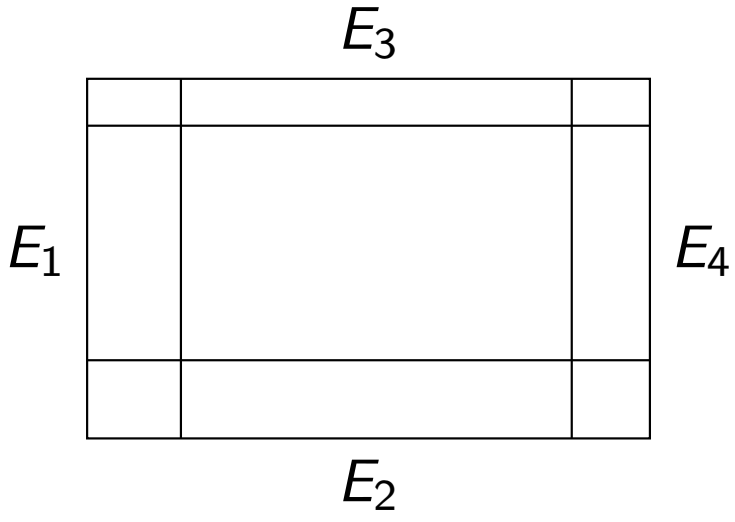
# Remarks on the Definition

- We assume the existence of a target concept
  - Work in realisable rather than agnostic setting
  - Say nothing about model selection, bias-variance trade-off, SRM, regularization,...
- The number of examples does not depend on the distribution on input space: the model is **distribution independent**.
- The definition allows for **improper learning**. It says nothing about the form or representation of the hypothesis.
- PAC model presents learning as a problem of **prediction**.
- Doesn't explicitly reflect Occam's razor.

# Hypothesis Rectangle



# Border Regions




$$\Pr(E_1) = \Pr(E_2) = \Pr(E_3) = \Pr(E_4) = \varepsilon/4$$

# Error Estimation

*Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $err(R_S) \leq \varepsilon$ with probability at least $1 - \delta$?*

# Error Estimation

*Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $err(R_S) \leq \varepsilon$ with probability at least $1 - \delta$?*

If the sample hits all four "border regions", then:

$$\begin{aligned} err(R_S) &= \Pr(R \setminus R_S) \\ &\leq \Pr(E_1 \cup E_2 \cup E_3 \cup E_4) \\ &\leq \sum_{i=1}^{4} \Pr(E_i) \leq \varepsilon\,, \end{aligned}$$

# Error Estimation

*Given $\epsilon > 0$ and $\delta > 0$, how many samples are needed such that $err(R_S) \leq \varepsilon$ with probability at least $1 - \delta$?*

If the sample hits all four "border regions", then:

$$\begin{aligned} err(R_S) &= \Pr(R \setminus R_S) \\ &\leq \Pr(E_1 \cup E_2 \cup E_3 \cup E_4) \\ &\leq \sum_{i=1}^{4} \Pr(E_i) \leq \varepsilon\,, \end{aligned}$$

*How many samples are need to hit all border regions with probability at least $1 - \delta$?*

# A Sample Bound

# A Sample Bound

- Pr(all $m$ samples miss $E_1$) $= \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.

# A Sample Bound

- Pr(all $m$ samples miss $E_1$) $= \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.
- Pr(some $E_i$ missed by all $m$ samples) $\leq 4e^{-\varepsilon m/4}$.

# A Sample Bound

- Pr(all $m$ samples miss $E_1$) $= \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.
- Pr(some $E_i$ missed by all $m$ samples) $\leq 4e^{-\varepsilon m/4}$.
- But $4e^{-\varepsilon m/4} \leq \delta$ iff $m \geq (4/\varepsilon)\log(4/\delta)$.

# A Sample Bound

- Pr(all $m$ samples miss $E_1$) $= \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.
- Pr(some $E_i$ missed by all $m$ samples) $\leq 4e^{-\varepsilon m/4}$.
- But $4e^{-\varepsilon m/4} \leq \delta$ iff $m \geq (4/\varepsilon)\log(4/\delta)$.
- Sample complexity is polynomial in $1/\varepsilon$ and $1/\delta$.

# A Sample Bound

- Pr(all $m$ samples miss $E_1$) $= \left(1 - \frac{\varepsilon}{4}\right)^m \leq e^{-\varepsilon m/4}$.
- Pr(some $E_i$ missed by all $m$ samples) $\leq 4e^{-\varepsilon m/4}$.
- But $4e^{-\varepsilon m/4} \leq \delta$ iff $m \geq (4/\varepsilon)\log(4/\delta)$.
- Sample complexity is polynomial in $1/\varepsilon$ and $1/\delta$.

**Learning map: output the "smallest" consistent hypothesis.**

# Building More Expressive Hypothesis Classes

# Perceptrons

A **linear classifier** is a function $f : \mathbb{R}^n \to \{-1, 1\}$ given by

$$f(\vec{x}) = \begin{cases} +1 & \text{if } \vec{a} \cdot \vec{x} \geq b \\ -1 & \text{otherwise} \end{cases}$$

# Perceptrons

A **linear classifier** is a function $f : \mathbb{R}^n \to \{-1, 1\}$ given by

$$f(\vec{x}) = \begin{cases} +1 & \text{if } \vec{a} \cdot \vec{x} \geq b \\ -1 & \text{otherwise} \end{cases}$$

Consistency with $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_k, y_k)\} \subseteq \mathbb{R}^n \times \{-1, 1\}$ formulated as a linear program:

$$\begin{aligned} \text{find} \quad & \vec{a} \in \mathbb{R}^n, b \in \mathbb{R} \text{ s.t.} \\ & y_i(\vec{a} \cdot \vec{x}_i - b) \geq 1, \; i = 1, \ldots, k. \end{aligned}$$

# Perceptrons

A **linear classifier** is a function $f : \mathbb{R}^n \to \{-1, 1\}$ given by

$$f(\vec{x}) = \begin{cases} +1 & \text{if } \vec{a} \cdot \vec{x} \geq b \\ -1 & \text{otherwise} \end{cases}$$

Consistency with $S = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_k, y_k)\} \subseteq \mathbb{R}^n \times \{-1, 1\}$ formulated as a linear program:

$$\begin{aligned} \text{find} \quad & \vec{a} \in \mathbb{R}^n, b \in \mathbb{R} \text{ s.t.} \\ & y_i(\vec{a} \cdot \vec{x}_i - b) \geq 1, \; i = 1, \ldots, k. \end{aligned}$$

If a consistent linear classifier exists then we say that $S$ is **linearly separable**.

# Combining Perceptrons

## Exercise

Given $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^d$ and $F : S \to \{-1, +1\}$, how can we can realise $f$ as a composition of perceptrons?

# Combining Perceptrons

## Exercise

Given $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}^d$ and $F : S \to \{-1, +1\}$, how can we can realise $f$ as a composition of perceptrons?

$$h(x) = \operatorname{sgn}\left(-1 + \sum_{i:F(x_i)=1} f_i(x) + g_i(x)\right)$$

# Layered Feedforward Neural Nets

- Directed acyclic, layered graph with inputs in $\mathbb{R}^n$ and output in $\{0, 1\}$ (the architecture)

# Layered Feedforward Neural Nets

- Directed acyclic, layered graph with inputs in $\mathbb{R}^n$ and output in $\{0, 1\}$ (the architecture)
- Each node computes function

$$\boldsymbol{x} \mapsto \sigma\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)$$

for **weights** $w_i$ and **activation function** $\sigma$

# Layered Feedforward Neural Nets

- Directed acyclic, layered graph with inputs in $\mathbb{R}^n$ and output in $\{0, 1\}$ (the architecture)
- Each node computes function

$$\boldsymbol{x} \mapsto \sigma\left(w_0 + \sum_{i=1}^{d} w_i x_i\right)$$

  for **weights** $w_i$ and **activation function** $\sigma$
- Activation functions:
  - Step $\sigma(z) = \mathbb{I}_{z \geq 0}$
  - Logistic sigmoidal $\sigma(z) = \frac{1}{1+e^{-z}}$
  - Rectified linear $\sigma(z) = \max(0, z)$

# But . . .

## Proposition

*The following problem is NP-hard. Given a feedforward neural network with $d$ inputs, a single hidden layer with 3 neurons, and step activation function, is there some weight setting that is consistent with a given labelled sample $S \subseteq \{0,1\}^d \times \{-1,+1\}$?*

# VC Dimension

# VC Dimension

Let $\mathcal{C}$ be a concept class on input set $\mathcal{X}$. We say that $S \subseteq \mathcal{X}$ is *shattered* by $\mathcal{C}$ if every function from $S$ to $\{0, 1\}$ arises as the restriction of some $c \in \mathcal{C}$.

# VC Dimension

Let $\mathcal{C}$ be a concept class on input set $\mathcal{X}$. We say that $S \subseteq \mathcal{X}$ is *shattered* by $\mathcal{C}$ if every function from $S$ to $\{0,1\}$ arises as the restriction of some $c \in \mathcal{C}$.

The *VC dimension* of $\mathcal{C}$ is defined by

$$\mathrm{VC}(\mathcal{C}) = \sup\left\{|S| : S \subseteq_{\mathrm{fin}} \mathcal{X} \text{ shattered by } \mathcal{C}\right\}.$$

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | |
| $\mathbb{R}^2$ | axis-aligned rectangles | |
| $\mathbb{R}^2$ | convex $k$-gons | |
| $\mathbb{R}^n$ | half spaces | |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | |
| $\mathbb{R}^2$ | axis-aligned rectangles | |
| $\mathbb{R}^2$ | convex $k$-gons | |
| $\mathbb{R}^n$ | half spaces | |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | |
| $\mathbb{R}^2$ | convex $k$-gons | |
| $\mathbb{R}^n$ | half spaces | |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | 4 |
| $\mathbb{R}^2$ | convex $k$-gons | |
| $\mathbb{R}^n$ | half spaces | |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | 4 |
| $\mathbb{R}^2$ | convex $k$-gons | $2k+1$ |
| $\mathbb{R}^n$ | half spaces | |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | 4 |
| $\mathbb{R}^2$ | convex $k$-gons | $2k+1$ |
| $\mathbb{R}^n$ | half spaces | $n+1$ |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | 4 |
| $\mathbb{R}^2$ | convex $k$-gons | $2k+1$ |
| $\mathbb{R}^n$ | half spaces | $n+1$ |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | $\infty$ |

# VC Dimension Workout

| $\mathcal{X}$ | $\mathcal{C}$ | $\mathrm{VC}(\mathcal{C})$ |
|---|---|---|
| $\{0,1\}^n$ | boolean formulas | $2^n$ |
| $\{0,1\}^n$ | boolean circuits with $n^3$ gates | $\mathrm{poly}(n)$ |
| $\mathbb{R}^2$ | axis-aligned rectangles | 4 |
| $\mathbb{R}^2$ | convex $k$-gons | $2k+1$ |
| $\mathbb{R}^n$ | half spaces | $n+1$ |
| $\mathbb{R}$ | $\{x \mapsto \mathrm{sgn}(\sin \alpha x) : \alpha \in \mathbb{R}\}$ | $\infty$ |

For "tame" models, VC dimension $\sim$ number of parameters

# Radon's Theorem

## Theorem (Radon's Theorem)

*Any set of $n+2$ points $S \subseteq \mathbb{R}^n$ can be partitioned into two subsets $S_1$ and $S_2$ such that the convex hulls of $S_1$ and $S_2$ intersect.*

# Radon's Theorem

## Theorem (Radon's Theorem)

*Any set of $n+2$ points $S \subseteq \mathbb{R}^n$ can be partitioned into two subsets $S_1$ and $S_2$ such that the convex hulls of $S_1$ and $S_2$ intersect.*

## Corollary

*No set of $n+2$ points in $\mathbb{R}^n$ can be shattered by halfspaces.*

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

Define $\alpha := \frac{\pi}{2}\left(1 + \sum_{i=1}^{m} 2^i(1 - f(x_i))\right)$.

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

Define $\alpha := \frac{\pi}{2}\left(1 + \sum_{i=1}^{m} 2^i(1 - f(x_i))\right)$. Then

$$\alpha x_k \bmod 2\pi \quad = \quad \alpha 2^{-k} \bmod 2\pi$$

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

Define $\alpha := \frac{\pi}{2}\left(1 + \sum_{i=1}^{m} 2^i(1 - f(x_i))\right)$. Then

$$\begin{aligned} \alpha x_k \bmod 2\pi &= \alpha 2^{-k} \bmod 2\pi \\ &= \frac{\pi}{2}\left(2^{-k} + \sum_{i=1}^{k-1}(1 - f(x_i))2^{i-k}\right) + \frac{\pi}{2}(1 - f(x_k)) \end{aligned}$$

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

Define $\alpha := \frac{\pi}{2}\left(1 + \sum_{i=1}^{m} 2^i(1 - f(x_i))\right)$. Then

$$\begin{aligned} \alpha x_k \bmod 2\pi &= \alpha 2^{-k} \bmod 2\pi \\ &= \frac{\pi}{2}\left(2^{-k} + \sum_{i=1}^{k-1}(1 - f(x_i))2^{i-k}\right) + \frac{\pi}{2}(1 - f(x_k)) \\ &= c\pi + \frac{\pi}{2}(1 - f(x_k)), \end{aligned}$$

where $c \in (0, 1)$.

# Concepts from Functions that are not Tame

Consider $S = \{x_1, \ldots, x_m\} \subseteq \mathbb{R}$, where $x_i = 2^{-i}$ for $i = 1, \ldots, m$, and an arbitrary labelling $f : S \to \{-1, +1\}$.

Define $\alpha := \frac{\pi}{2}\left(1 + \sum_{i=1}^{m} 2^i(1 - f(x_i))\right)$. Then

$$\begin{aligned}
\alpha x_k \bmod 2\pi &= \alpha 2^{-k} \bmod 2\pi \\
&= \frac{\pi}{2}\left(2^{-k} + \sum_{i=1}^{k-1}(1 - f(x_i))2^{i-k}\right) + \frac{\pi}{2}(1 - f(x_k)) \\
&= c\pi + \frac{\pi}{2}(1 - f(x_k)),
\end{aligned}$$

where $c \in (0, 1)$. Thus $\operatorname{sgn}(\sin \alpha x_k) = f(x_k)$ for $k = 1, \ldots, m$.

# Dual Classes

Given $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$, the **dual class** $\mathcal{C}^* \subseteq \{0,1\}^{\mathcal{C}}$ consists of $f_x : \mathcal{C} \to \{0,1\}$, $x \in \mathcal{X}$, such that $f_x(c) = c(x)$ for all $c \in \mathcal{C}$.

# Dual Classes

Given $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$, the **dual class** $\mathcal{C}^* \subseteq \{0,1\}^{\mathcal{C}}$ consists of $f_x : \mathcal{C} \to \{0,1\}$, $x \in \mathcal{X}$, such that $f_x(c) = c(x)$ for all $c \in \mathcal{C}$.

## Proposition

$\mathrm{VC}(\mathcal{C}) \leq 2^{\mathrm{VC}(\mathcal{C}^*)}$.

# Dual Classes

Given $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$, the **dual class** $\mathcal{C}^* \subseteq \{0,1\}^{\mathcal{C}}$ consists of $f_x : \mathcal{C} \to \{0,1\}$, $x \in \mathcal{X}$, such that $f_x(c) = c(x)$ for all $c \in \mathcal{C}$.

## Proposition

$\mathrm{VC}(\mathcal{C}) \leq 2^{\mathrm{VC}(\mathcal{C}^*)}$.

## Proof.

Suppose $\{c_1, \ldots, c_{2^n}\} \subseteq \mathcal{C}$ is shattered by $\mathcal{C}^*$ for some $n \in \mathbb{N}$. Then for each $i \in \{1, \ldots, n\}$ there exists $x_i \in \mathcal{X}$ such that $f_{x_i}(c_j) = 1$ if and only if the $i$-th bit of $j$ is 1. But then $\{c_1, \ldots, c_{2^n}\}$ shatters $\{x_1, \ldots, x_n\}$. $\square$

# The Growth Function

Consider a concept class $\mathcal{C}$ on input set $\mathcal{X}$. Given a finite sample $S \subseteq \mathcal{X}$, define

$$\Pi_{\mathcal{C}}(S) = \{c|_S : c \in \mathcal{C}\},$$

where $c|_S$ denotes the restriction of a function $c : \mathcal{X} \to \{0, 1\}$ to the set $S$.

# The Growth Function

Consider a concept class $\mathcal{C}$ on input set $\mathcal{X}$. Given a finite sample $S \subseteq \mathcal{X}$, define

$$\Pi_{\mathcal{C}}(S) = \{c|_S : c \in \mathcal{C}\},$$

where $c|_S$ denotes the restriction of a function $c : \mathcal{X} \to \{0, 1\}$ to the set $S$.

The *growth function* of $\mathcal{C}$ is defined as

$$\Pi_{\mathcal{C}}(m) = \max_{S:|S|=m} |\Pi_{\mathcal{C}}(S)|.$$

# Growth-Function Workout

Let $\mathcal{C}$ be the class of annular disks in $\mathbb{R}^2$ of the form $\{(x, y) : r \leq x^2 + y^2 \leq R\}$, where $r, R \in \mathbb{R}$. Describe the growth function $\Pi_{\mathcal{C}}(m)$.

# Sauer's Lemma

## Lemma

*Let $\mathcal{C}$ be a hypothesis set with finite VC dimension $d$. Then for all $m \geq d$,*

$$\Pi_{\mathcal{C}}(m) \leq \sum_{k=0}^{d} \binom{m}{k} \leq \left(\frac{em}{d}\right)^{d}.$$

# Sauer's Lemma

## Lemma

*Let $\mathcal{C}$ be a hypothesis set with finite VC dimension $d$. Then for all $m \geq d$,*

$$\Pi_{\mathcal{C}}(m) \leq \sum_{k=0}^{d} \binom{m}{k} \leq \left(\frac{em}{d}\right)^{d}.$$

Considering $\mathcal{X} = \mathbb{N}$ and $\mathcal{C}$ the collection of subsets of $\mathbb{N}$ of cardinality at most $d$, one sees that the upper bound in Lemma 3 is tight.

# VC Dimension of Neural Nets with Step Activation

**Theorem**

*Fix an architecture with $n_0$ inputs and $\omega$ parameters and write $\mathcal{C} \subseteq \{0,1\}^{\mathbb{R}^{n_0}}$ the for class of functions that can be implemented by instantiating the parameters. Then*

$$\mathrm{VC}(\mathcal{C}) \leq 2\omega \log_2(e\omega)\,.$$

# Concept Classes from Predicate Logic

Given a $\sigma$-formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, $\sigma$-structure $\mathcal{A}$, and elements $b_1, \ldots, b_n \in A$, write

$$\varphi(\mathcal{A}, b_1, \ldots, b_n) := \{(a_1, \ldots, a_m) \in A^m : \mathcal{A} \models \varphi(a_1, \ldots, a_m, b_1, \ldots, b_n)\}$$

# Concept Classes from Predicate Logic

Given a $\sigma$-formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, $\sigma$-structure $\mathcal{A}$, and elements $b_1, \ldots, b_n \in A$, write

$$\varphi(\mathcal{A}, b_1, \ldots, b_n) := \{(a_1, \ldots, a_m) \in A^m : \mathcal{A} \models \varphi(a_1, \ldots, a_m, b_1, \ldots, b_n)\}$$

Then the **definable family of sets**

$$\mathcal{C}(\varphi, \mathcal{A}) = \{\varphi(\mathcal{A}, b_1, \ldots, b_n) : b_1 \ldots, b_n \in A\}$$

specifies a concept class on the input space $\mathcal{X} = A^m$.

# Concept Classes from Predicate Logic

Given a $\sigma$-formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, $\sigma$-structure $\mathcal{A}$, and elements $b_1, \ldots, b_n \in A$, write

$$\varphi(\mathcal{A}, b_1, \ldots, b_n) := \{(a_1, \ldots, a_m) \in A^m : \mathcal{A} \models \varphi(a_1, \ldots, a_m, b_1, \ldots, b_n)\}$$

Then the **definable family of sets**

$$\mathcal{C}(\varphi, \mathcal{A}) = \{\varphi(\mathcal{A}, b_1, \ldots, b_n) : b_1 \ldots, b_n \in A\}$$

specifies a concept class on the input space $\mathcal{X} = A^m$.

Write $\mathrm{VC}(\varphi, \mathcal{A})$ for $\mathrm{VC}(\mathcal{C}(\varphi, \mathcal{A}))$.

# Semi-Algebraic Families (I)

Let $\mathcal{A} = (\mathbb{R}, 0, 1, +, \times)$.

# Semi-Algebraic Families (I)

Let $\mathcal{A} = (\mathbb{R}, 0, 1, +, \times)$.

Give bounds on $\mathrm{VC}(\varphi, \mathcal{A})$ in terms of the "complexity" of $\varphi$.

# Semi-Algebraic Families (I)

Let $\mathcal{A} = (\mathbb{R}, 0, 1, +, \times)$.

Give bounds on $\mathrm{VC}(\varphi, \mathcal{A})$ in terms of the "complexity" of $\varphi$.

Key tool is:

## Theorem (Warren 68, Goldberg and Jerrum 95)

*Let $P_1, \ldots, P_\ell$ be polynomials of degree at most $d$ in $k$ real variables with $\ell \geq k$. Then the number of realisable sign assignments (each $P_i$ is positive, negative, or zero) is at most $(8ed\ell/k)^k$.*

# Semi-Algebraic Families (II)

## Proposition

*Let* $\mathcal{A} = (\mathbb{R}, 0, 1, +, \times)$ *and let* $\varphi(x_1, \ldots, x_n, y_1, \ldots, y_k)$ *be a Boolean combination of* $s$ *polynomial equalities and inequalities, with each polynomial mentioned in* $\varphi$ *having degree at most* $d$*. Then* $\mathrm{VCD}(\varphi, \mathcal{A})$ *is at most* $2k \log(8eds)$*.*

# Useful Lemma

## Lemma (Shelah 71, Laskowski 92)

*Let $K$ be a class of structures such that for every formula $\varphi(x, y_1, \ldots, y_n)$ the set $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded. Then it also holds that $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded for every first-order formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$.*

# o-Minimal Structures

## Proposition

*If $\mathcal{A}$ is an o-minimal structure then $\mathrm{VC}(\varphi, \mathcal{A}) < \infty$ for every first-order formula $\varphi$.*

# o-Minimal Structures

## Proposition

*If $\mathcal{A}$ is an o-minimal structure then $\mathrm{VC}(\varphi, \mathcal{A}) < \infty$ for every first-order formula $\varphi$.*

## Proof.

By Shelah's result, it suffices to prove that $\mathrm{VC}(\varphi, \mathcal{A}) < \infty$) for every formula $\varphi(x_1, y_1, \ldots, y_n)$.

# o-Minimal Structures

## Proposition

*If $\mathcal{A}$ is an o-minimal structure then $\mathrm{VC}(\varphi, \mathcal{A}) < \infty$ for every first-order formula $\varphi$.*

## Proof.

By Shelah's result, it suffices to prove that $\mathrm{VC}(\varphi, \mathcal{A}) < \infty$) for every formula $\varphi(x_1, y_1, \ldots, y_n)$.
But by general results about o-minimal structures, there is a bound on the number intervals comprising $\varphi(\mathcal{A}, b_1, \ldots, b_n)$ that is independent of $b_1, \ldots, b_n \in A$. $\square$

# Neural Nets - Sigmoidal and Piecewise Polynomial Activation

**Theorem (Goldberg and Jerrum 95, Karpinski and Macintyre 97)**

*Consider a fixed architecture of layered feed-forward neural net with $N$ neurons and $\omega$ weights. The VC dimension of the corresponding class of functions is:*

# Neural Nets - Sigmoidal and Piecewise Polynomial Activation

**Theorem (Goldberg and Jerrum 95, Karpinski and Macintyre 97)**

*Consider a fixed architecture of layered feed-forward neural net with $N$ neurons and $\omega$ weights. The VC dimension of the corresponding class of functions is:*

- $O(\omega N)$ *with piecewise polynomial activation functions and an absolute bound on number of pieces (e.g., ReLU)*

# Neural Nets - Sigmoidal and Piecewise Polynomial Activation

**Theorem (Goldberg and Jerrum 95, Karpinski and Macintyre 97)**

*Consider a fixed architecture of layered feed-forward neural net with $N$ neurons and $\omega$ weights. The VC dimension of the corresponding class of functions is:*

- $O(\omega N)$ *with piecewise polynomial activation functions and an absolute bound on number of pieces (e.g., ReLU)*
- $O(\omega^2 N^2)$ *with sigmoidal activation function.*

# Classes of Structures

Want bounds on $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ for $K$ a class of structures.

# Classes of Structures

Want bounds on $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ for $K$ a class of structures.

**Example.** Consider formula $\varphi := E(x, y)$ and $\mathcal{G}_n = (U, V, E)$ a bipartite graph, where $U = \{1, \ldots, n\}$ and $V = 2^{\{1,\ldots,n\}}$ and

$$E = \{(i, \alpha) : i \in U, \alpha \in V, i \in \alpha\}.$$

Then $\mathrm{VC}(\varphi, \mathcal{G}_n) \geq n$.

# Forests

Consider a signature with a single relation symbol. Let $K$ be the class of $\sigma$ structures that are **forests**.

# Forests

Consider a signature with a single relation symbol. Let $K$ be the class of $\sigma$ structures that are **forests**.

## Theorem (Maass and Turán 95)

*For any formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, the set $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded.*

# Forests

Consider a signature with a single relation symbol. Let $K$ be the class of $\sigma$ structures that are **forests**.

## Theorem (Maass and Turán 95)

*For any formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, the set $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded.*

## Proof.

Suppose $\varphi(x, y_1, \ldots, y_n)$ has quantifier-depth $k$. Consider two types of sets that cannot be shattered:

# Forests

Consider a signature with a single relation symbol. Let $K$ be the class of $\sigma$ structures that are **forests**.

## Theorem (Maass and Turán 95)

*For any formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, the set $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded.*

## Proof.

Suppose $\varphi(x, y_1, \ldots, y_n)$ has quantifier-depth $k$. Consider two types of sets that cannot be shattered:

- A sufficiently large set $S \subseteq A$ such that for distinct $a, b \in S$ we have $d(a, b) > 4k$.

# Forests

Consider a signature with a single relation symbol. Let $K$ be the class of $\sigma$ structures that are **forests**.

## Theorem (Maass and Turán 95)

*For any formula $\varphi(x_1, \ldots, x_m, y_1, \ldots, y_n)$, the set $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded.*

## Proof.

Suppose $\varphi(x, y_1, \ldots, y_n)$ has quantifier-depth $k$. Consider two types of sets that cannot be shattered:

- A sufficiently large set $S \subseteq A$ such that for distinct $a, b \in S$ we have $d(a, b) > 4k$.
- A sufficiently large set $S \subseteq A$ such that for distinct $a, b \in S$ we have $d(a, b) \leq 4k$.

□

# Bounds for MSO

## Theorem (Grohe and Turán 04)

*Let $\sigma$ be a relational signature and $K$ a class of structures of bounded tree-width. Then for any formula $\varphi$ of MSO, $\{\mathrm{VC}(\varphi, \mathcal{A}) : \mathcal{A} \in K\}$ is bounded.*

# Grid Graphs

Consider the $2^{\times} n$ grid $\mathcal{G}_{n,2^n}$. We describe a formula $\varphi(x, y)$ s.t. $\mathrm{VC}(\varphi, \mathcal{G}_{n,2^n}) \geq n$. Formula $\varphi$ is such that $\mathcal{G}_{n,2^n} \models \varphi((i, 1), (1, j))$ if the $i$-th bit in the binary expansion of $j$ is 1.

# Grid Graphs

Consider the $2^{\times} n$ grid $\mathcal{G}_{n,2^n}$. We describe a formula $\varphi(x, y)$ s.t. $\mathrm{VC}(\varphi, \mathcal{G}_{n,2^n}) \geq n$. Formula $\varphi$ is such that $\mathcal{G}_{n,2^n} \models \varphi((i,1),(1,j))$ if the $i$-th bit in the binary expansion of $j$ is 1.

Then $\mathcal{C}(\varphi, \mathcal{G}_{n,2^n})$ shatters the set $\{(1,1), \ldots, (n,1)\}$.

# VC Classes are PAC Learnable

# VC Classes are PAC Learnable

## Theorem

*Let $\mathcal{C}$ be a concept class of VC dimension $d$ on input set $\mathcal{X}$. Let $\varepsilon, \delta > 0$ and $m \in \mathbb{N}$ be such that*

$$m \geq \frac{8}{\varepsilon}\left(d \log \frac{8}{\varepsilon} + \log \frac{2}{\delta}\right) \tag{1}$$

*Then for any target concept $c \in \mathcal{C}$ and distribution $D$ on $\mathcal{X}$, the probability that a sample of size $m$ is consistent with some hypothesis $h \in \mathcal{C}$ with $\mathrm{err}(h) \geq \varepsilon$ is at most $\delta$.*

# VC Classes are PAC Learnable

## Theorem

*Let $\mathcal{C}$ be a concept class of VC dimension $d$ on input set $\mathcal{X}$. Let $\varepsilon, \delta > 0$ and $m \in \mathbb{N}$ be such that*

$$m \geq \frac{8}{\varepsilon}\left(d \log \frac{8}{\varepsilon} + \log \frac{2}{\delta}\right) \tag{1}$$

*Then for any target concept $c \in \mathcal{C}$ and distribution $D$ on $\mathcal{X}$, the probability that a sample of size $m$ is consistent with some hypothesis $h \in \mathcal{C}$ with $\mathrm{err}(h) \geq \varepsilon$ is at most $\delta$.*

## Corollary

*VC classes are PAC learnable.*

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

## Proposition

$$\Pr_{S_1, S_2 \sim D^m} \left( \exists h \in \mathcal{C} \left( \mathrm{err}_{S_1}(h) = 0 \wedge \mathrm{err}_{S_2}(h) \geq \frac{\varepsilon}{2} \right) \right) \leq \frac{\delta}{2}$$

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

## Proposition

$$\Pr_{S_1, S_2 \sim D^m}\left(\exists h \in \mathcal{C}\left(\mathrm{err}_{S_1}(h) = 0 \wedge \mathrm{err}_{S_2}(h) \geq \frac{\varepsilon}{2}\right)\right) \leq \frac{\delta}{2}$$

- Choose $S = \{x_1, \ldots, x_{2m}\}$ i.i.d. from $D$.

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

## Proposition

$$\Pr_{S_1,S_2\sim D^m}\left(\exists h\in\mathcal{C}\left(\mathrm{err}_{S_1}(h)=0\wedge\mathrm{err}_{S_2}(h)\geq\frac{\varepsilon}{2}\right)\right)\leq\frac{\delta}{2}$$

- Choose $S=\{x_1,\ldots,x_{2m}\}$ i.i.d. from $D$.
- For $i=1,\ldots,2m$ swap $x_i$ with $x_{i+m}$ with prob $1/2$.

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

## Proposition

$$\Pr_{S_1,S_2\sim D^m}\left(\exists h\in\mathcal{C}\left(\mathrm{err}_{S_1}(h)=0\wedge\mathrm{err}_{S_2}(h)\geq\frac{\varepsilon}{2}\right)\right)\leq\frac{\delta}{2}$$

- Choose $S=\{x_1,\ldots,x_{2m}\}$ i.i.d. from $D$.
- For $i=1,\ldots,2m$ swap $x_i$ with $x_{i+m}$ with prob $1/2$.
- Set $S_1:=\{x_1,\ldots,x_m\}$ and $S_2:=\{x_{m+1},\ldots,x_{2m}\}$.

# Step I - The Double Sampling Trick

Two samples $S_1$ and $S_2$ of size $m$ drawn i.i.d. from $D$ behave similarly w.r.t. every statistical test in $\mathcal{C}$:

## Proposition

$$\Pr_{S_1,S_2\sim D^m}\left(\exists h\in\mathcal{C}\left(\mathrm{err}_{S_1}(h)=0\wedge\mathrm{err}_{S_2}(h)\geq\frac{\varepsilon}{2}\right)\right)\leq\frac{\delta}{2}$$

- Choose $S=\{x_1,\ldots,x_{2m}\}$ i.i.d. from $D$.
- For $i=1,\ldots,2m$ swap $x_i$ with $x_{i+m}$ with prob $1/2$.
- Set $S_1:=\{x_1,\ldots,x_m\}$ and $S_2:=\{x_{m+1},\ldots,x_{2m}\}$.
- What is the distribution on $S_1$ and $S_2$?

# Interlude - Chernoff Bound

Let $X_1, \ldots, X_m$ be $m$ independent Bernoulli trials with mean $p$. Write $S = \frac{1}{m}(X_1 + \ldots + X_m)$.

# Interlude - Chernoff Bound

Let $X_1, \ldots, X_m$ be $m$ independent Bernoulli trials with mean $p$. Write $S = \frac{1}{m}(X_1 + \ldots + X_m)$.

## Theorem

*Given* $0 \leq \gamma \leq 1$ *the following bounds hold:*

$$\begin{aligned} \Pr\left(S > (1+\gamma)p\right) &\leq e^{-mp\gamma^2/3} \\ \Pr\left(S < (1-\gamma)p\right) &\leq e^{-mp\gamma^2/2} \end{aligned}$$

# Step II - Back to Double Sampling

Recall the "double sampling" experiment:

- Event $A$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \varepsilon$ and $h$ consistent on $S_1$.

# Step II - Back to Double Sampling

Recall the "double sampling" experiment:

- Event $A$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \varepsilon$ and $h$ consistent on $S_1$.
- Event $B$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \epsilon$, $h$ consistent on $S_1$ and has at least $m\varepsilon/2$ errors on $S_2$

# Step II - Back to Double Sampling

Recall the "double sampling" experiment:

- Event $A$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \varepsilon$ and $h$ consistent on $S_1$.
- Event $B$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \epsilon$, $h$ consistent on $S_1$ and has at least $m\varepsilon/2$ errors on $S_2$
- By Chernoff, $\Pr(B \mid A) \geq \frac{1}{2}$.

# Step II - Back to Double Sampling

Recall the "double sampling" experiment:

- Event $A$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \varepsilon$ and $h$ consistent on $S_1$.
- Event $B$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \epsilon$, $h$ consistent on $S_1$ and has at least $m\varepsilon/2$ errors on $S_2$
- By Chernoff, $\Pr(B \mid A) \geq \frac{1}{2}$.
- Thus

$$Pr(B) = \Pr(A \cap B) = \Pr(B \mid A) \Pr(A) \geq \frac{1}{2} \Pr(A)$$

# Step II - Back to Double Sampling

Recall the "double sampling" experiment:

- Event $A$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \varepsilon$ and $h$ consistent on $S_1$.
- Event $B$: $\exists h \in \mathcal{C}$, $\mathrm{err}(h) \geq \epsilon$, $h$ consistent on $S_1$ and has at least $m\varepsilon/2$ errors on $S_2$
- By Chernoff, $\Pr(B \mid A) \geq \frac{1}{2}$.
- Thus

$$Pr(B) = \Pr(A \cap B) = \Pr(B \mid A) \Pr(A) \geq \frac{1}{2} \Pr(A)$$

But we know $\Pr(B) \leq \delta/2$, so $\Pr(A) \leq \delta$.

# Lower Bounds

**Theorem**

*Let $\mathcal{C}$ be a concept class that has VC dimension at least $d$. Then for any learning algorithm there exists a target concept $c \in \mathcal{D}$ and distribution $D$ on the input space $\mathcal{X}$ such that if the algorithm is given $d/2$ examples then the output hypothesis $h \in \mathcal{C}$ is such that*

$$\Pr(\mathrm{err}(h) > 1/8) > 1/8\,.$$

# Remember this Slide for Later!

**Theorem**

*Let $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ have VC dimension $d$. Let $D$ be a distribution on $\mathcal{X}$ and $\varepsilon > 0$. Then there exists a multiset $S \subseteq \mathcal{X}$ of cardinality $O(d/\varepsilon^2)$ such that for all $c \in \mathcal{C}$,*

$$\left| \Pr_{x \sim D}(c(x) = 1) - \frac{1}{|S|} \sum_{x \in S} \mathbb{I}(c(x) = 1) \right| < \varepsilon \,.$$

# Remember this Slide for Later!

**Theorem**

*Let $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ have VC dimension $d$. Let $D$ be a distribution on $\mathcal{X}$ and $\varepsilon > 0$. Then there exists a multiset $S \subseteq \mathcal{X}$ of cardinality $O(d/\varepsilon^2)$ such that for all $c \in \mathcal{C}$,*

$$\left| \Pr_{x \sim D}(c(x) = 1) - \frac{1}{|S|} \sum_{x \in S} \mathbb{I}(c(x) = 1) \right| < \varepsilon \,.$$

If $\mathcal{C}$ contains the zero function then a corollary of sample complexity bounds.

# Sample Compression Schemes

# Sample Compression Schemes

A *sample compression scheme* for $\mathcal{C}$ consists $k \in \mathbb{N}$ (the *kernel size*) and a finite set $I$, together with:

# Sample Compression Schemes

A *sample compression scheme* for $\mathcal{C}$ consists $k \in \mathbb{N}$ (the *kernel size*) and a finite set $I$, together with:

- A *compression map*

$$\kappa : \bigcup_{m \in \mathbb{N}} L_{\mathcal{C}}(m) \to L_{\mathcal{C}}(k) \times I$$

mapping $(S, f)$ to $((S', f'), \sigma)$, where $S' \subseteq S$ and $f' = f|_{S'}$.

# Sample Compression Schemes

A *sample compression scheme* for $\mathcal{C}$ consists $k \in \mathbb{N}$ (the *kernel size*) and a finite set $I$, together with:

- A *compression map*

$$\kappa : \bigcup_{m \in \mathbb{N}} L_{\mathcal{C}}(m) \to L_{\mathcal{C}}(k) \times I$$

  mapping $(S, f)$ to $((S', f'), \sigma)$, where $S' \subseteq S$ and $f' = f|_{S'}$.

- A *reconstruction map*

$$\rho : L_{\mathcal{C}}(k) \times I \to \{0,1\}^{\mathcal{X}},$$

  such that $\rho(\kappa(S, f))|_S = f$ for all $(S, f) \in L_{\mathcal{C}}(m)$, $m \geq k$.

# Sample Compression Scheme Workout

- Rectangles in the plane

# Sample Compression Scheme Workout

- Rectangles in the plane
- Intervals in the real line

# Sample Compression Scheme Workout

- Rectangles in the plane
- Intervals in the real line
- Halfspaces

# Sample Compression Scheme Workout

- Rectangles in the plane
- Intervals in the real line
- Halfspaces
- Mistake-bounded mistake-driven online learning algorithms

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

1. Initialise all weights $w_1, \ldots, w_n$ to 1.

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

1. Initialise all weights $w_1, \ldots, w_n$ to 1.
2. Given an example $\vec{x} \in \{0,1\}^n$, Output 1 if

$$w_1 x_1 + \ldots + w_n x_n \geq n$$

and output 0 otherwise.

# Excursion into Online Learning

**Winnow for Monotone Disjunctions**

1. Initialise all weights $w_1, \ldots, w_n$ to 1.
2. Given an example $\vec{x} \in \{0,1\}^n$, Output 1 if

$$w_1 x_1 + \ldots + w_n x_n \geq n$$

   and output 0 otherwise.
3. If the algorithm makes a mistake:

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

1. Initialise all weights $w_1, \ldots, w_n$ to 1.
2. Given an example $\vec{x} \in \{0,1\}^n$, Output 1 if

$$w_1 x_1 + \ldots + w_n x_n \geq n$$

   and output 0 otherwise.
3. If the algorithm makes a mistake:
   (a) If the algorithm predicts negative on a positive example, then for each $x_i$ equal to 1, double the value of $w_i$.

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

1. Initialise all weights $w_1, \ldots, w_n$ to 1.
2. Given an example $\vec{x} \in \{0,1\}^n$, Output 1 if

$$w_1 x_1 + \ldots + w_n x_n \geq n$$

   and output 0 otherwise.
3. If the algorithm makes a mistake:
   (a) If the algorithm predicts negative on a positive example, then for each $x_i$ equal to 1, double the value of $w_i$.
   (b) If the algorithm predicts positive on a negative example, then for each $x_i$ equal to 1, halve the value of $w_i$.

# Excursion into Online Learning

## Winnow for Monotone Disjunctions

1. Initialise all weights $w_1, \ldots, w_n$ to 1.
2. Given an example $\vec{x} \in \{0,1\}^n$, Output 1 if

$$w_1 x_1 + \ldots + w_n x_n \geq n$$

   and output 0 otherwise.
3. If the algorithm makes a mistake:
   (a) If the algorithm predicts negative on a positive example, then for each $x_i$ equal to 1, double the value of $w_i$.
   (b) If the algorithm predicts positive on a negative example, then for each $x_i$ equal to 1, halve the value of $w_i$.
4. Goto 2.

# Mistake Bounds for Winnow

Suppose that the target concept is a disjunction of $r$ variables.

1. The algorithm makes at most $r(1 + \log_2 n)$ mistakes on positive examples.

# Mistake Bounds for Winnow

Suppose that the target concept is a disjunction of $r$ variables.

1. The algorithm makes at most $r(1 + \log_2 n)$ mistakes on positive examples.

2. The algorithm makes at most twice the number of mistakes on negative examples as on positive examples. (Consider effect of positive and negative mistakes on total weight $\sum_{i=1}^{n} w_i$.)

# Mistake Bounds for Winnow

Suppose that the target concept is a disjunction of $r$ variables.

1. The algorithm makes at most $r(1 + \log_2 n)$ mistakes on positive examples.

2. The algorithm makes at most twice the number of mistakes on negative examples as on positive examples. (Consider effect of positive and negative mistakes on total weight $\sum_{i=1}^{n} w_i$.)

3. Total number of mistakes is at most $2 + 3r(1 + \log_2 n)$.

# Sample Compression Implies Learnability

## Theorem

*Let $\mathcal{C}$ be a concept class on instance space $\mathcal{X}$ that has a sample compression scheme $\kappa, \rho$ of kernel size $k$. Given $\varepsilon > 0$ and $\delta > 0$, let*

$$m \geq \max\left(\frac{2}{\varepsilon}\left(\log\left(\frac{2}{\delta}\right) + \log|I|\right), \frac{4k}{\varepsilon}\log\left(\frac{4k}{\varepsilon}\right) + 2k\right).$$

*Then the function $H : L_{\mathcal{C}}(m) \to \{0,1\}^{\mathcal{X}}$ defined by $H(S, f) = \rho(\kappa(S, f))$ is a PAC learning map with generalisation error $\varepsilon$ and failure probability $\delta$.*

# Sample Compression Implies Learnability

## Theorem

*Let $\mathcal{C}$ be a concept class on instance space $\mathcal{X}$ that has a sample compression scheme $\kappa, \rho$ of kernel size $k$. Given $\varepsilon > 0$ and $\delta > 0$, let*

$$m \geq \max\left(\frac{2}{\varepsilon}\left(\log\left(\frac{2}{\delta}\right) + \log|I|\right), \frac{4k}{\varepsilon}\log\left(\frac{4k}{\varepsilon}\right) + 2k\right).$$

*Then the function $H : L_{\mathcal{C}}(m) \to \{0,1\}^{\mathcal{X}}$ defined by $H(S, f) = \rho(\kappa(S, f))$ is a PAC learning map with generalisation error $\varepsilon$ and failure probability $\delta$.*

Reconstruction map receives only a small fraction of sample and so cannot overfit.

# The Sample Compression Conjecture

*Does every concept class of VC dimension $d$ have a sample compression scheme of kernel size $d$?*

Littlestone and Warmuth, 1986

# The Sample Compression Conjecture

*Does every concept class of VC dimension $d$ have a sample compression scheme of kernel size $d$?*

Littlestone and Warmuth, 1986

**Theorem (Johnson, Laskowski, Livni, Simon, Chernikov)**

*A definable family $\mathcal{C}(\varphi, \mathcal{A})$ has a sample compression scheme if either $\varphi$ is stable or $\mathcal{A}$ is an NIP structure.*

# The Sample Compression Conjecture

*Does every concept class of VC dimension $d$ have a sample compression scheme of kernel size $d$?*

Littlestone and Warmuth, 1986

**Theorem (Johnson, Laskowski, Livni, Simon, Chernikov)**

*A definable family $\mathcal{C}(\varphi, \mathcal{A})$ has a sample compression scheme if either $\varphi$ is stable or $\mathcal{A}$ is an NIP structure.*

*If $\mathcal{A}$ is o-minimal then $\mathcal{C}(\varphi, \mathcal{A})$ has a compression scheme of kernel size equal to the number of parameters of $\varphi$.*

# The Sample Compression Conjecture

*Does every concept class of VC dimension $d$ have a sample compression scheme of kernel size $d$?*

Littlestone and Warmuth, 1986

**Theorem (Johnson, Laskowski, Livni, Simon, Chernikov)**

*A definable family $\mathcal{C}(\varphi, \mathcal{A})$ has a sample compression scheme if either $\varphi$ is stable or $\mathcal{A}$ is an NIP structure.*

*If $\mathcal{A}$ is o-minimal then $\mathcal{C}(\varphi, \mathcal{A})$ has a compression scheme of kernel size equal to the number of parameters of $\varphi$.*

**Theorem (Moran, Yehudayoff, 2016)**

*A concept class of VC dimension $d$ and dual VC dimension $d^*$ admits a sample compression scheme of kernel size $O(d \cdot d^*)$.*

# A "Caratheodory Theorem" for $\mathcal{C}$

## Proposition

*Let $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ be a concept class such that the dual class $\mathcal{C}^* \subseteq \{0,1\}^{\mathcal{C}}$ has VC dimension $d^*$. Let $D^*$ be a distribution over $\mathcal{C}$ and $\varepsilon > 0$. Then there is a multiset $\mathcal{F} \subseteq \mathcal{C}$ of size $O(d^*/\varepsilon^2)$ such that for all $x \in \mathcal{X}$,*

$$\left| \Pr_{c \sim D^*}(c(x) = 1) - \frac{1}{|\mathcal{F}|} \sum_{c \in \mathcal{F}} \mathbb{I}\{c(x) = 1\} \right| \leq \varepsilon \,.$$

# A "Caratheodory Theorem" for $\mathcal{C}$

## Proposition

*Let $\mathcal{C} \subseteq \{0,1\}^{\mathcal{X}}$ be a concept class such that the dual class $\mathcal{C}^* \subseteq \{0,1\}^{\mathcal{C}}$ has VC dimension $d^*$. Let $D^*$ be a distribution over $\mathcal{C}$ and $\varepsilon > 0$. Then there is a multiset $\mathcal{F} \subseteq \mathcal{C}$ of size $O(d^*/\varepsilon^2)$ such that for all $x \in \mathcal{X}$,*

$$\left| \Pr_{c \sim D^*}(c(x) = 1) - \frac{1}{|\mathcal{F}|} \sum_{c \in \mathcal{F}} \mathbb{I}\{c(x) = 1\} \right| \leq \varepsilon \,.$$

What is the one-line proof?

# Zero-Sum Matrix Games

Rock-Paper-Scissors:

| | R | P | S |
|---|---|---|---|
| R | 0 | $+1$ | $-1$ |
| P | $-1$ | 0 | $+1$ |
| S | $+1$ | $-1$ | 0 |

# Minimax Theorem

## Theorem (Von Neumann Minimax Theorem)

*For any $m \times n$ matrix $M$,*

$$\min_{\vec{p} \in \Delta_m} \max_{\vec{q} \in \Delta_n} p^\top M_{i,j} q = \max_{\vec{q} \in \Delta_n} \min_{\vec{p} \in \Delta_m} p^\top M_{i,j} q \,.$$

# Main Result

## Theorem

*Let $\mathcal{C}$ be a concept class of VC dimension $d$ and dual VC dimension $d^*$. Then $\mathcal{C}$ admits a sample compression scheme of kernel size $O(d \cdot d^*)$.*

# Main Result

## Theorem

*Let $\mathcal{C}$ be a concept class of VC dimension $d$ and dual VC dimension $d^*$. Then $\mathcal{C}$ admits a sample compression scheme of kernel size $O(d \cdot d^*)$.*

*Proof.*

- There exists $s \in \mathbb{N}$ and a learning map $H : L_{\mathcal{C}}(s) \to \mathcal{C}$ s.t. for all $c \in C$ and every distribution $D$ on $\mathcal{X}$

$$\Pr_{T \sim D^s} \operatorname{err}(H(T, c|_T)) \leq 1/3\,.$$

# Main Result

## Theorem

*Let $\mathcal{C}$ be a concept class of VC dimension $d$ and dual VC dimension $d^*$. Then $\mathcal{C}$ admits a sample compression scheme of kernel size $O(d \cdot d^*)$.*

*Proof.*

- There exists $s \in \mathbb{N}$ and a learning map $H : L_{\mathcal{C}}(s) \to \mathcal{C}$ s.t. for all $c \in C$ and every distribution $D$ on $\mathcal{X}$

$$\Pr_{T \sim D^s} \operatorname{err}(H(T, c|_T)) \leq 1/3\,.$$

- Let $(S, c|_S)$ be a labelled sample to be compressed, where $S \subseteq \mathcal{X}$. Let $\mathcal{H} = \{H(T, c|_T) : T \subseteq S, |T| \leq s\}$ be the image of the learning map on subsets of $S$ of cardinality at most $s$.

# Central Claim

**Claim.** There are $k = O(d^*)$ functions $f_1 \ldots, f_k \in \mathcal{H}$ such that

$$|\{i : f_i(x) = c(x)\}| > k/2$$

for all $x \in S$.

# Compression and Reconstruction

**Compression.** Let $Z_i \subseteq S$ be such $f_i = H(Z_i, c|_{Z_i})$ for $i = 1, \ldots, k$. Define $\kappa(S, c|_S) = ((Z, c|_Z), \sigma)$ where $Z = \cup_{i=1}^{k} Z_i$ and $\sigma$ identifies which elements of $Z$ lie in each of the subsets $Z_1, \ldots, Z_k$.

# Compression and Reconstruction

**Compression.** Let $Z_i \subseteq S$ be such $f_i = H(Z_i, c|_{Z_i})$ for $i = 1, \ldots, k$. Define $\kappa(S, c|_S) = ((Z, c|_Z), \sigma)$ where $Z = \cup_{i=1}^{k} Z_i$ and $\sigma$ identifies which elements of $Z$ lie in each of the subsets $Z_1, \ldots, Z_k$.

**Reconstruction.** To $h := \rho((Z, f), \sigma)$, consider the sets $Z_1, \ldots, Z_k$ determined by $Z$ and $\sigma$ and define $f_i = H(Z_i, c|_{Z_i})$ for $i = 1, \ldots, k$. For every $x \in \mathcal{X}$, we define $h(x)$ to be the majority element in the list $f_1(x), \ldots, f_k(x)$.

# Concepts that are Hard to Learn

# Concepts that are Hard to Learn

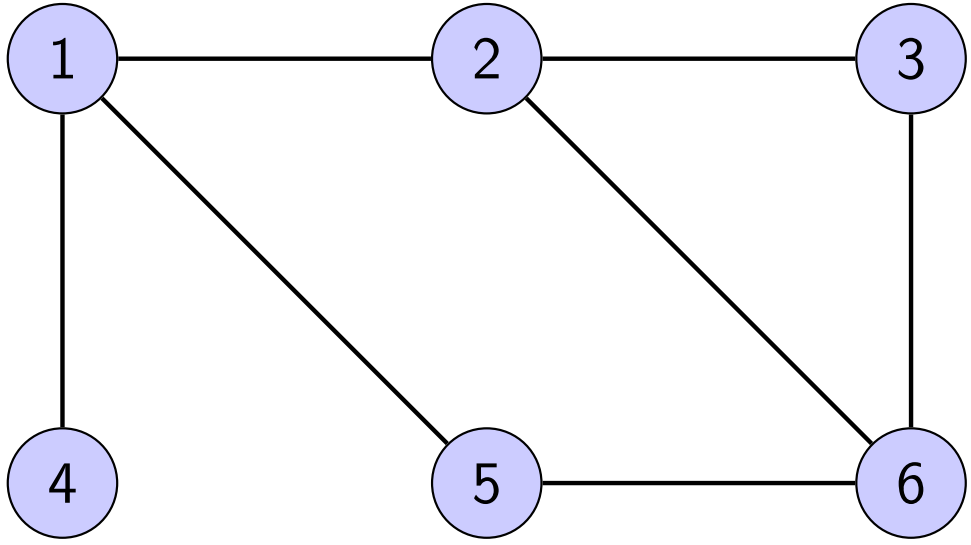
Is there a parameterized family of concept classes $\mathcal{C} = \{\mathcal{C}_n : n \in \mathbb{N}\}$, with $\mathrm{VC}(\mathcal{C}_n) = n^{O(1)}$, such that $\mathcal{C}$ admits no PAC learning algorithm that runs in polynomial-time in $n$ and the length of the representation of the target concept?

# Concepts that are Hard to Learn

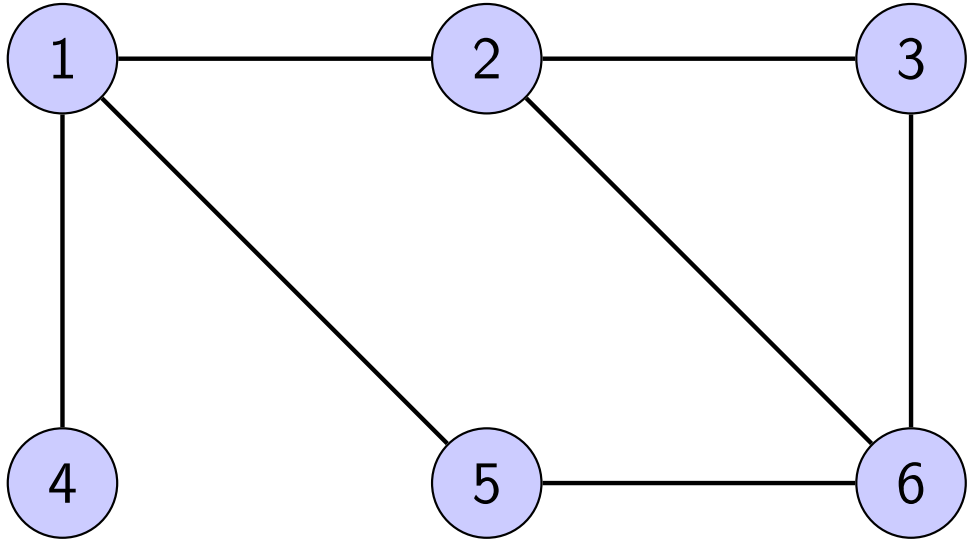
Is there a parameterized family of concept classes $\mathcal{C} = \{\mathcal{C}_n : n \in \mathbb{N}\}$, with $\mathrm{VC}(\mathcal{C}_n) = n^{O(1)}$, such that $\mathcal{C}$ admits no PAC learning algorithm that runs in polynomial-time in $n$ and the length of the representation of the target concept?

For example, what about $\mathcal{C}_n \subseteq \{0,1\}^{\Sigma^*}$ the class of regular languages whose minimal DFA has $n$ states?

# Consistency Problem for 3-Term DNF



# Consistency Problem for 3-Term DNF




| Positive examples |
|---|
| $(100000, \mathbf{1})$ |
| $(010000, \mathbf{1})$ |
| $(001000, \mathbf{1})$ |
| $(000100, \mathbf{1})$ |
| $(000010, \mathbf{1})$ |
| $(000001, \mathbf{1})$ |

| Negative examples |
|---|
| $(110000, \mathbf{0})$ |
| $(100100, \mathbf{0})$ |
| $(100010, \mathbf{0})$ |
| $(011000, \mathbf{0})$ |
| $(010001, \mathbf{0})$ |
| $(001001, \mathbf{0})$ |
| $(000011, \mathbf{0})$ |

# Finding a Consistent Hypothesis

Suppose $G$ is 3-colourable then

$$\bigwedge_{i:i \text{ not red}} \neg P_i \ \vee \bigwedge_{i:i \text{ not blue}} \neg P_i \ \vee \bigwedge_{i:i \text{ not green}} \neg P_i$$

is a consistent hypothesis.

# Improper Learning via 3-Term DNF

- By distributivity, every 3-term DNF $\tau_1 \vee \tau_2 \vee \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \vee \tau_2 \vee \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \vee \ell_2 \vee \ell_3)\,.$$

# Improper Learning via 3-Term DNF

- By distributivity, every 3-term DNF $\tau_1 \vee \tau_2 \vee \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \vee \tau_2 \vee \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \vee \ell_2 \vee \ell_3)\,.$$

- We can find a consistent hypothesis in 3-CNF in polynomial time!

# Improper Learning via 3-Term DNF

- By distributivity, every 3-term DNF $\tau_1 \vee \tau_2 \vee \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \vee \tau_2 \vee \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \vee \ell_2 \vee \ell_3)\,.$$

- We can find a consistent hypothesis in 3-CNF in polynomial time!
  - Start with conjunction of all 3-literal clauses and delete any clause not satisfied by some positive example.

# Improper Learning via 3-Term DNF

- By distributivity, every 3-term DNF $\tau_1 \vee \tau_2 \vee \tau_3$ has an equivalent 3-CNF:

$$\tau_1 \vee \tau_2 \vee \tau_3 \equiv \bigwedge_{\ell_1 \in \tau_1, \ell_2 \in \tau_2, \ell_3 \in \tau_3} (\ell_1 \vee \ell_2 \vee \ell_3)\,.$$

- We can find a consistent hypothesis in 3-CNF in polynomial time!
  - Start with conjunction of all 3-literal clauses and delete any clause not satisfied by some positive example.
  - What is the running time?

# Some Elementary Number Theory

- Let $N = pq$ where primes $p, q \equiv 2 \bmod 3$. Then $\mathbb{Z}_N^*$ has order $\varphi(N) = (p-1)(q-1)$.

# Some Elementary Number Theory

- Let $N = pq$ where primes $p, q \equiv 2 \bmod 3$. Then $\mathbb{Z}_N^*$ has order $\varphi(N) = (p-1)(q-1)$.
- Consider $f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, $f_N(x) = x^3 \bmod N$.

# Some Elementary Number Theory

- Let $N = pq$ where primes $p, q \equiv 2 \bmod 3$. Then $\mathbb{Z}_N^*$ has order $\varphi(N) = (p-1)(q-1)$.
- Consider $f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, $f_N(x) = x^3 \bmod N$.
- The inverse function is $f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, where $g_N(y) = y^d \bmod N$ with $d = 3^{-1} \bmod \varphi(N)$.

# Some Elementary Number Theory

- Let $N = pq$ where primes $p, q \equiv 2 \bmod 3$. Then $\mathbb{Z}_N^*$ has order $\varphi(N) = (p-1)(q-1)$.
- Consider $f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, $f_N(x) = x^3 \bmod N$.
- The inverse function is $f_N : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, where $g_N(y) = y^d \bmod N$ with $d = 3^{-1} \bmod \varphi(N)$.
- Given $N$ and $x$, computing $g_N(x)$ is in polynomial time if we know both $p$ and $q$.

# Concepts that are Hard to Learn

## Definition (Discrete Cube Root Assumption)

For any polynomial $P(\cdot)$, there does not exist any algorithm, $A$, that runs in time $P(n)$ and on input $N$ and $x$, where $N$ is the product of two random $n$-bit primes $p, q \equiv 2 \bmod 3$ and $x$ is chosen randomly from $\mathbb{Z}_N^*$, outputs $g_N(x)$ with probability at least $1/P(n)$. The probability is over the random choices of $p$, $q$, $x$ and any internal randomisation of $A$.

# Concepts that are Hard to Learn

## Definition (Discrete Cube Root Assumption)

For any polynomial $P(\cdot)$, there does not exist any algorithm, $A$, that runs in time $P(n)$ and on input $N$ and $x$, where $N$ is the product of two random $n$-bit primes $p, q \equiv 2 \bmod 3$ and $x$ is chosen randomly from $\mathbb{Z}_N^*$, outputs $g_N(x)$ with probability at least $1/P(n)$. The probability is over the random choices of $p$, $q$, $x$ and any internal randomisation of $A$.

Discrete cube roots are hard to find on average.

# Discrete Cube Roots as a Learning Problem

Suppose that we have access to a sample, $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, where $y_i = g_N(x_i)$ for $i = 1, \ldots, m$, and $x_i$ are drawn uniformly at random from $\mathbb{Z}_N^*$. Can we obtain $h : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, such that for $x$ drawn uniformly at random from $\mathbb{Z}_N^*$, it holds with probability at least $1 - \delta$ over the sample, that $\Pr(h(x) \neq g_N(x)) \leq \epsilon$?

# Discrete Cube Roots as a Learning Problem

Suppose that we have access to a sample, $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, where $y_i = g_N(x_i)$ for $i = 1, \ldots, m$, and $x_i$ are drawn uniformly at random from $\mathbb{Z}_N^*$. Can we obtain $h : \mathbb{Z}_N^* \to \mathbb{Z}_N^*$, such that for $x$ drawn uniformly at random from $\mathbb{Z}_N^*$, it holds with probability at least $1 - \delta$ over the sample, that $\Pr(h(x) \neq g_N(x)) \leq \epsilon$?

If our learning algorithm had polynomial sample complexity and running time (in the bit length of of $N$) then we would violate DCRA. Why???

# Concepts that are Hard to Learn

But learning cube roots is not a classification problem:

# Concepts that are Hard to Learn

But learning cube roots is not a classification problem:

- Let $g_{N,i} : \mathbb{Z}_N^* \to \{0,1\}$ give the $i$-th bit of $g_N$ for $i = 1, \ldots, 2n$.

# Concepts that are Hard to Learn

But learning cube roots is not a classification problem:

- Let $g_{N,i} : \mathbb{Z}_N^* \to \{0,1\}$ give the $i$-th bit of $g_N$ for $i = 1, \ldots, 2n$.

- A concept class that admits polynomial-size representations of $g_{N,i}$ and is evaluable in polynomial time is hard to learn under DCRA.

# Concepts that are Hard to Learn

**Theorem**

*There exists a fixed polynomial $P(\cdot)$, such that class of circuits, $\mathcal{C}$, where $\mathcal{C}_n$ consists of circuits of size at most $P(n)$ on $n$ inputs, is not PAC-learnable under the discrete cube root assumption.*

# Concepts that are Hard to Learn

**Theorem**

*There exists a fixed polynomial $P(\cdot)$, such that class of circuits, $\mathcal{C}$, where $\mathcal{C}_n$ consists of circuits of size at most $P(n)$ on $n$ inputs, is not PAC-learnable under the discrete cube root assumption.*

With some work one can strengthen this to the sub-case $\mathcal{C}_n$ consists of boolean formulas on $n$ inputs.

# Concepts that are Hard to Learn

**Theorem**

*There exists a fixed polynomial $P(\cdot)$, such that class of circuits, $\mathcal{C}$, where $\mathcal{C}_n$ consists of circuits of size at most $P(n)$ on $n$ inputs, is not PAC-learnable under the discrete cube root assumption.*

With some work one can strengthen this to the sub-case $\mathcal{C}_n$ consists of boolean formulas on $n$ inputs.

What about DFA?

# Learning with Membership and Equivalence Queries

# Learning with Membership and Equivalence Queries

Angluin (1987) devised the $L^*$ algorithm for **exactly** learning regular languages using the following two types of query:

# Learning with Membership and Equivalence Queries

Angluin (1987) devised the $L^*$ algorithm for **exactly** learning regular languages using the following two types of query:

- **Membership queries.** Learner selects word $w \in \Sigma^*$ and the teacher says whether or not $w$ lies in target language.

# Learning with Membership and Equivalence Queries

Angluin (1987) devised the $L^*$ algorithm for **exactly** learning regular languages using the following two types of query:

- **Membership queries.** Learner selects word $w \in \Sigma^*$ and the teacher says whether or not $w$ lies in target language.
- **Equivalence queries.** Learner formulates hypothesis language $L'$ and teacher responds whether or not $L'$ is identical to the target language and, if not, gives a counterexample word.

# Learning with Membership and Equivalence Queries

Angluin (1987) devised the $L^*$ algorithm for **exactly** learning regular languages using the following two types of query:

- **Membership queries.** Learner selects word $w \in \Sigma^*$ and the teacher says whether or not $w$ lies in target language.
- **Equivalence queries.** Learner formulates hypothesis language $L'$ and teacher responds whether or not $L'$ is identical to the target language and, if not, gives a counterexample word.

Number of queries is polynomial in the size of the minimal automaton for the target and the logarithm of the longest counterexample given by the teacher.

# Weighted Automata

- Weighted automata recognise functions $f : \Sigma^* \to \mathbb{K}$.

# Weighted Automata

- Weighted automata recognise functions $f : \Sigma^* \to \mathbb{K}$.
- Weighted automata over $\mathrm{GF}(2)$ are NFA that accept by parity.

# Weighted Automata

- Weighted automata recognise functions $f : \Sigma^* \to \mathbb{K}$.
- Weighted automata over $\mathrm{GF}(2)$ are NFA that accept by parity.
- Hankel matrix of $f : \Sigma^* \to \mathbb{K}$ is $\Sigma^* \times \Sigma^*$ matrix whose $(u, v)$-th entry is $f(uv)$.

# Weighted Automata

- Weighted automata recognise functions $f : \Sigma^* \to \mathbb{K}$.
- Weighted automata over $\mathrm{GF}(2)$ are NFA that accept by parity.
- Hankel matrix of $f : \Sigma^* \to \mathbb{K}$ is $\Sigma^* \times \Sigma^*$ matrix whose $(u, v)$-th entry is $f(uv)$.

## Theorem (Carlyle and Paz, Fleiss)

*A function $f : \Sigma^* \to \mathbb{K}$ is recognizable by a weighted automaton if and only if its Hankel matrix has finite rank.*

# A Learning Algorithm

At each stage the algorithm maintains the following data:

# A Learning Algorithm

At each stage the algorithm maintains the following data:

- A set of $n$ "rows" $X = \{x_1, \ldots, x_n\} \subseteq \Sigma^*$, where $x_1 = \varepsilon$.

# A Learning Algorithm

At each stage the algorithm maintains the following data:

- A set of $n$ "rows" $X = \{x_1, \ldots, x_n\} \subseteq \Sigma^*$, where $x_1 = \varepsilon$.
- A set of $n$ "columns" $Y = \{y_1, \ldots, y_n\} \subseteq \Sigma^*$, where $y_1 = \varepsilon$.

# A Learning Algorithm

At each stage the algorithm maintains the following data:

- A set of $n$ "rows" $X = \{x_1, \ldots, x_n\} \subseteq \Sigma^*$, where $x_1 = \varepsilon$.
- A set of $n$ "columns" $Y = \{y_1, \ldots, y_n\} \subseteq \Sigma^*$, where $y_1 = \varepsilon$.
- A full-rank $n \times n$ submatrix of Hankel matrix of target function:

$$H = \begin{bmatrix} f(x_1y_1) & f(x_1y_2) & \cdots & f(x_1y_n) \\ f(x_2y_1) & f(x_2y_2) & \cdots & f(x_2y_n) \\ \vdots & \vdots & \ddots & \vdots \\ f(x_ny_1) & f(x_ny_2) & \cdots & f(x_ny_n) \end{bmatrix}$$

# Building a Hypothesis

The hypothesis is $(n, \Sigma, \boldsymbol{\alpha}, \{M(\sigma)\}_{\sigma\in\Sigma}, \boldsymbol{\eta})$, where

# Building a Hypothesis

The hypothesis is $(n, \Sigma, \boldsymbol{\alpha}, \{M(\sigma)\}_{\sigma\in\Sigma}, \boldsymbol{\eta})$, where

- $\boldsymbol{\alpha} = \boldsymbol{e}_1^\top H$.

# Building a Hypothesis

The hypothesis is $(n, \Sigma, \boldsymbol{\alpha}, \{M(\sigma)\}_{\sigma\in\Sigma}, \boldsymbol{\eta})$, where

- $\boldsymbol{\alpha} = \boldsymbol{e}_1^\top H$.
- $\boldsymbol{\eta} = \boldsymbol{e}_1$.

# Building a Hypothesis

The hypothesis is $(n, \Sigma, \boldsymbol{\alpha}, \{M(\sigma)\}_{\sigma\in\Sigma}, \boldsymbol{\eta})$, where

- $\boldsymbol{\alpha} = \boldsymbol{e}_1^\top H$.
- $\boldsymbol{\eta} = \boldsymbol{e}_1$.
- Each $M(\sigma)$ is defined by

$$HM(\sigma) = \begin{bmatrix} f(x_1\sigma y_1) & f(x_1\sigma y_2) & \cdots & f(x_1\sigma y_n) \\ f(x_2\sigma y_1) & f(x_2\sigma y_2) & \cdots & f(x_2\sigma y_n) \\ \vdots & \vdots & \ddots & \vdots \\ f(x_n\sigma y_1) & f(x_n\sigma y_2) & \cdots & f(x_n\sigma y_n) \end{bmatrix}$$

# Parsing a Counterexample

## Proposition

*A counterexample $z$ has a prefix $w\sigma$, where $\sigma \in \Sigma$ and $w \in \Sigma^*$, such that for some $i \in \{1, \ldots, n\}$ the assignment $X \leftarrow X \cup \{w\}$, $Y \leftarrow Y \cup \{\sigma y_i\}$ increases the rank of $H$ by one.*

# Parsing a Counterexample

## Proposition

*A counterexample $z$ has a prefix $w\sigma$, where $\sigma \in \Sigma$ and $w \in \Sigma^*$, such that for some $i \in \{1, \ldots, n\}$ the assignment $X \leftarrow X \cup \{w\}$, $Y \leftarrow Y \cup \{\sigma y_i\}$ increases the rank of $H$ by one.*

## Theorem

*The concept class of weighted automata over a field $\mathbb{K}$ is exactly learnable with number of queries $O((|\Sigma| + \log m)r^2)$ where $r$ is the rank of the Hankel matrix of the target function and $m$ is the length of the longest counterexample returned by the teacher.*

# Summary

- Introduced a basic theoretical framework for supervised learning, emphasizing connections with logic and automata
- Further developments this week:
  - Statistical learning theory
  - Automata learning
  - Further connections with logic and verification
- Branching out:
  - Reinforcement learning
  - Bayesian learning