Machine Teaching

Adish Singla

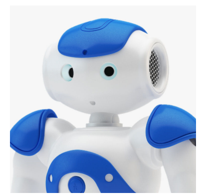
CMMRS, August 2019

MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

MAX-PLANCK-GESELLSCHAFT

Machine Learning vs. Teaching

Training data D → **Learner** → Learnt hypothesis $\hat{h}$



Target hypothesis h^* →

Learner model  →

Teacher → Teaching data D^*

Why Machine Teaching?

Adversarial settings
aka training-set poisoning

Educational settings

Applications: Online Education via MOOCs

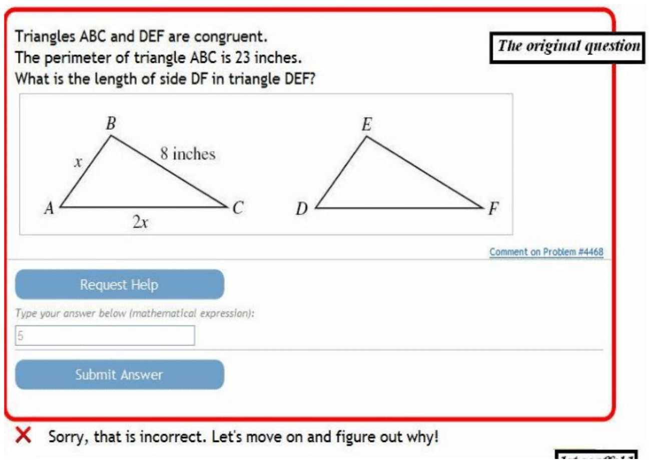
- Astronomical growth with over 100 million students
- Over 10,000 courses offered online

Key challenge: Dropout rate of over 95%

Applications: Skill Assessment and Practice

ASSISTments

HOUR OF CODE



- Over 10 million problems solved per year on ASSISTments
- Over 0.8 billion hours of code by 100 million students

Key limitation: No automated or personalized curriculum of problems

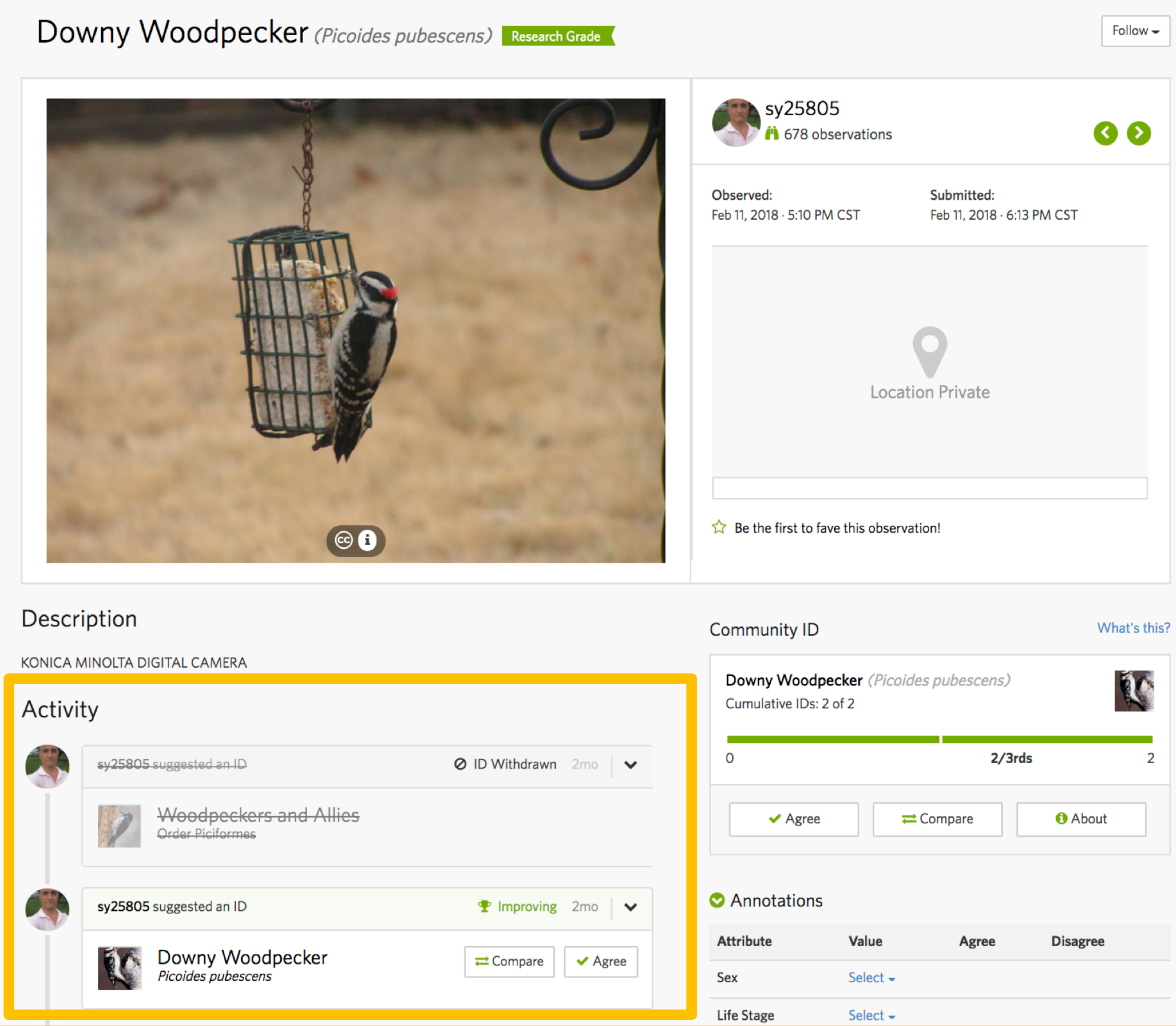
Applications: Training Simulators

Applications: Language Learning

- Over 300+ million students
- Based on **spaced repetition** of flash cards

Can we compute **optimal personalized schedule** of repetition?

Applications: Biodiversity Monitoring

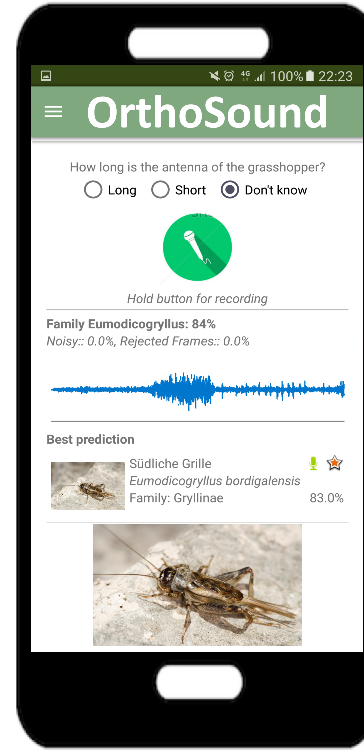


Key challenge: Noise in the annotations

Image credits: iNaturalist

Applications: Biodiversity Monitoring

iNaturalist
Explore Learn Record



- Teaching helps increase awareness and engagement
- Labeled data is crucial for training machine learning systems

Can we **teach** participants to label more accurately?

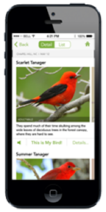
Machine Teaching: Applications

Educational settings

Online education via MOOCs

Skill assessment and practice

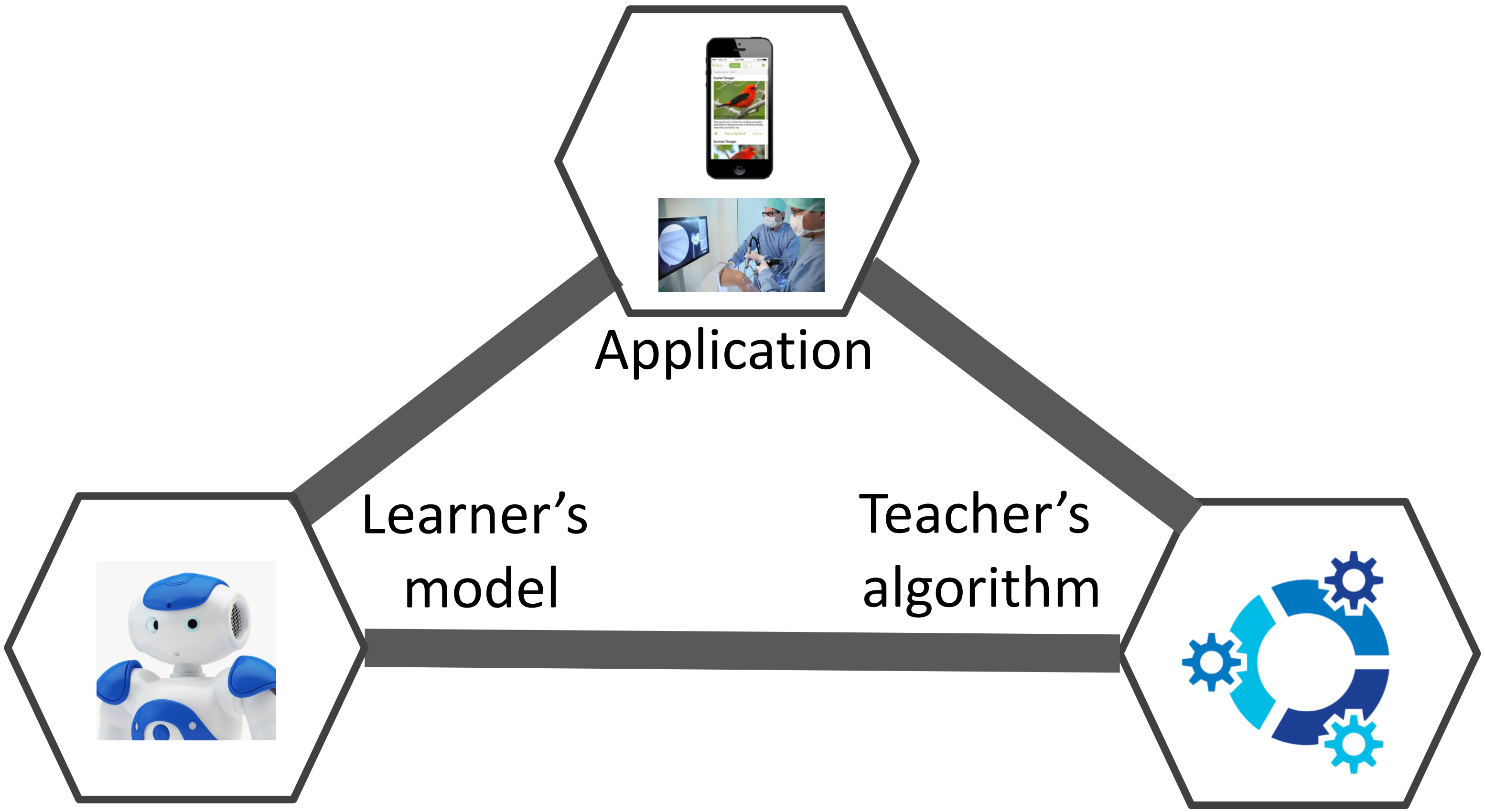
Language learning



Biodiversity monitoring

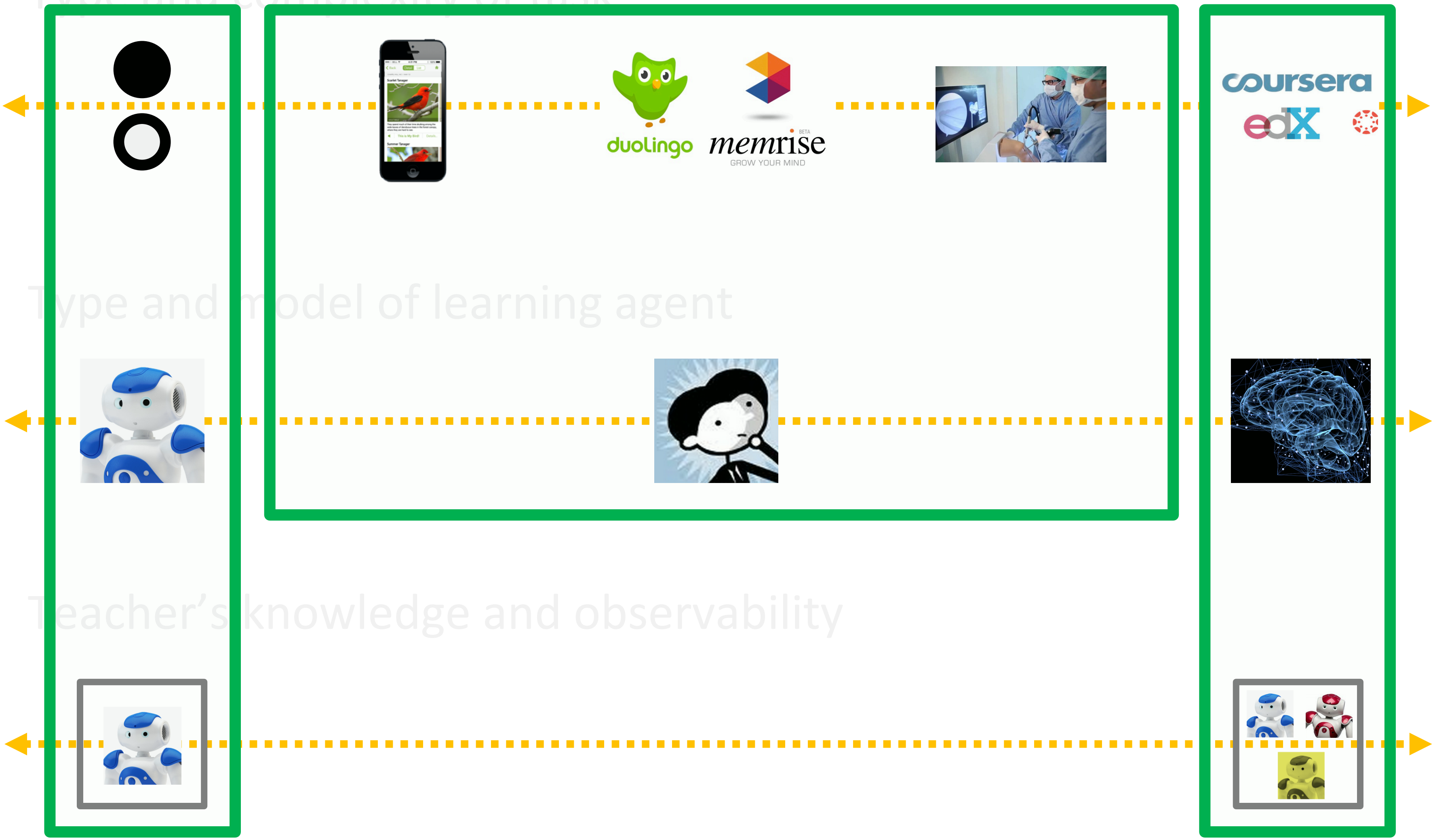
Training simulators

Machine Teaching: Key Components



Machine Teaching: Problem Space

- Type and complexity of task
- Type and model of learning agent
- Teacher's knowledge and observability



Course Outline

Part 1: Different viewpoints of the problem space

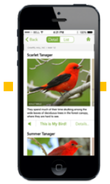
- **Information-theoretic** models of teaching
- **Cognitive** models of teaching

Part 2: Designing algorithms for teaching people

- **Classification** rules for biodiversity monitoring
- **Vocabulary** for language learning
- **Policies** for performing sequential tasks

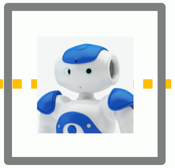
Machine Teaching: Problem Space

- Type and complexity of task



- Type and model of learning agent

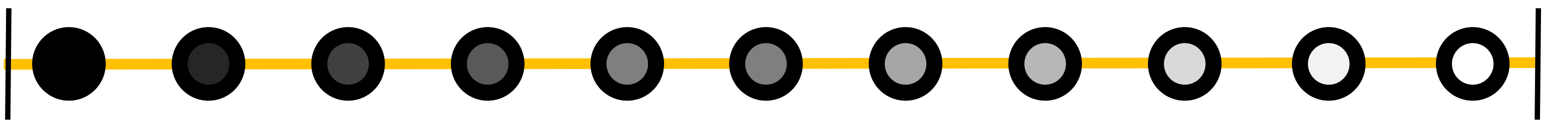
- Teacher's knowledge and observability



An Example: 1-D Threshold Function

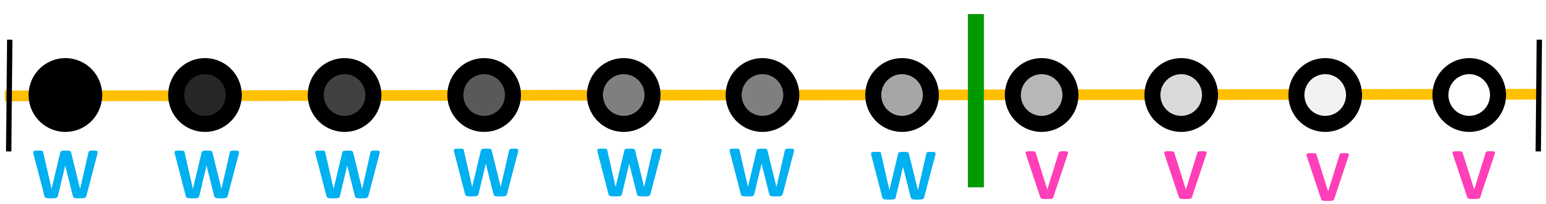
- Task: Classify animal image as **Weevil** or **Vespula**

- $\mathcal{X}$: Set of images, each $x \in \mathcal{X}$ is associated with a contrast level



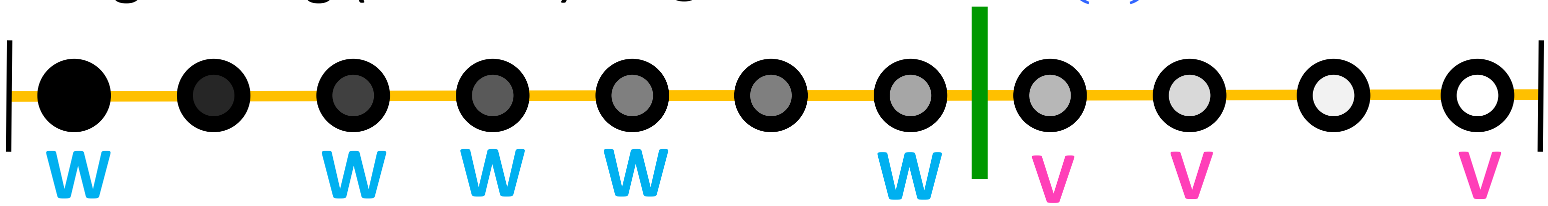
- $\mathcal{H}$: Set of hypotheses, each $h \in \mathcal{H}$ is a binary threshold classifier

- h^*: True classifier

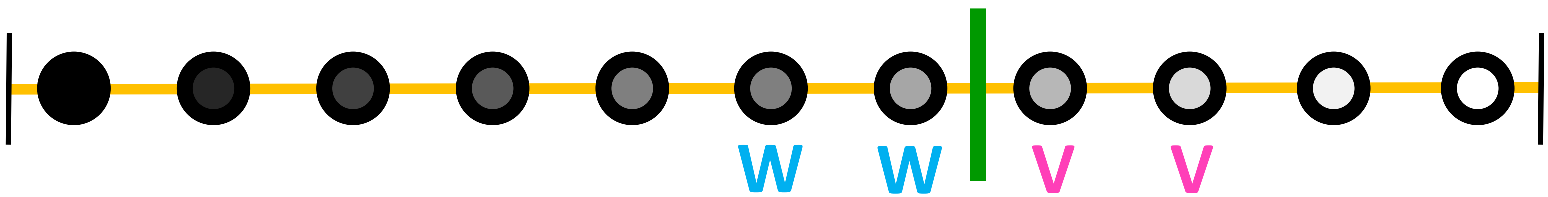


An Example: 1-D Threshold Function

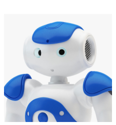
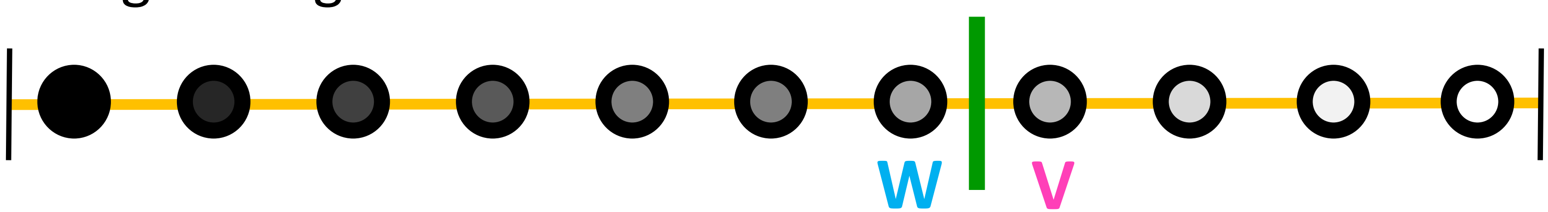
- Learning setting (Passive): avg. size of D is $\Theta(n)$



- Learning setting (Active): size of D is $\Theta(\log n)$



- Teaching setting: size of D is 2

Teaching Binary Functions

- Set of unlabeled examples $\mathcal{X}$
- Hypotheses class $\mathcal{H}$ as a set of binary functions $h : \mathcal{X} \to \{0,1\}$
- Target hypothesis $h^* \in \mathcal{H}$

$$
\begin{array}{cc|c|c|c|c|c|}
 & & \multicolumn{5}{c}{\mathcal{X}} \\
 & & x_1 & x_2 & x_3 & x_4 & x_5 \\
\hline
 & h_1 & 1 & 1 & 1 & 1 & 1 \\
\hline
 & h_2 & 0 & 1 & 1 & 1 & 1 \\
\hline
\mathcal{H} & h_3 & 0 & 0 & 1 & 1 & 1 \\
\hline
h^* = & h_4 & \boxed{0} & \boxed{0} & \boxed{0} & \boxed{1} & \boxed{1} \\
\hline
 & h_5 & 0 & 0 & 0 & 0 & 1 \\
\hline
\end{array}
\qquad
\begin{array}{cc|c|c|c|c|}
 & & \multicolumn{4}{c}{\mathcal{X}} \\
 & & x_1 & x_2 & x_3 & x_4 \\
\hline
 & h_1 & 1 & 0 & 0 & 0 \\
\hline
 & h_2 & 0 & 1 & 0 & 0 \\
\hline
\mathcal{H} & h_3 & 0 & 0 & 1 & 0 \\
\hline
 & h_4 & 0 & 0 & 0 & 1 \\
\hline
h^* = & h_5 & \boxed{0} & \boxed{0} & \boxed{0} & \boxed{0} \\
\hline
\end{array}
$$

Teaching Interaction

Start

- Learner starts at $h_0 \in \mathcal{H}$

At time t

Teacher

$\mathcal{X}, \mathcal{H}, h^*$

Teacher receives an estimate of h_{t-1}

Teacher selects x_t, provides $x_t, h^*(x_t)$

Learner updates to h_t

Learner

$\mathcal{X}, \mathcal{H}, h_{t-1}$

Stop

- When $h_t = h^*$

Learner Model: Version space learning

Notion of version space

- Maintain a set of eligible hypotheses
  - Start with $H_0 = \mathcal{H}$
- At time t, remove hypothesis inconsistent with $x_t, h^*(x_t)$
  - $H_t = H_{t-1} \setminus \{h \in \mathcal{H} \mid h(x_t) \neq h^*(x_t)\}$

Version space learner

- Learner starts at $h_0 \in \mathcal{H}$, $H_0 = \mathcal{H}$
- At time t:
  - Learner receives $x_t, h^*(x_t)$ and updates H_t
  - Learner selects a new hypothesis $h_t \in H_t$ at random

Teacher: Optimization Problem

Analysis setting

- **Worst-case** vs. average case
- **Finite** vs. infinite/continuous $\mathcal{H}$
- **Exact** vs. approximate teaching

Optimization problem

- Find smallest sequence $\overleftrightarrow{S} = (x_1, x_2, \dots)$ to achieve desired objective

$$\overleftrightarrow{S}^{\text{opt}} = \underset{\overleftrightarrow{S}}{\operatorname{argmin}} |\overleftrightarrow{S}| \qquad \text{s.t.} \quad \boxed{\begin{array}{c} h_t = h^* \\ \text{equivalent to} \\ H_t = \{h^*\} \end{array}}$$

Teacher: Optimization Problem

Teaching problem is equivalent to Set Cover problem

- $\mathcal{H} \setminus \{h^*\}$ is the set of elements to remove or "cover"
- Each x covers a subset $\mathcal{H}(x) = \{h \in \mathcal{H} \mid h(x) \neq h^*(x)\}$
- Find smallest set $S = \{x_1, x_2, \dots\}$ to cover $\mathcal{H} \setminus h^*$

Complexity of optimization

Theorem: Finding optimal teaching sequence $\overleftrightarrow{S}^{\text{opt}}$ is NP-hard.

Teacher: Optimization Problem

Teaching problem is a Submodular Coverage problem

- Define set function $F\colon 2^{\mathcal{X}} \to \mathbb{R}_+$ as

$$F(S) = |\cup_{x \in S} \mathcal{H}(x)| \text{ where } S \subseteq \mathcal{X}$$

- Rewrite teaching problem as

$$S^{\text{opt}} = \underset{S}{\operatorname{argmin}} |S| \quad \text{s.t.} \quad F(S) \geq |\mathcal{H}| - 1$$

Submodular Coverage problem

- $F(.)$ satisfies submodularity: A notion of diminishing returns

$$F(\{a\} \cup S) - F(S) \geq F(\{a\} \cup \{b\} \cup S) - F(\{b\} \cup S)$$

We can optimize using a greedy algorithm with provable guarantees

Teacher: Algorithm

Iterative greedy algorithm

- **Input:** $\mathcal{H}, \mathcal{X}, h^*$
- **Initialize:** set $S \leftarrow \emptyset$
- While $F(S) < |\mathcal{H}| - 1$:
  - Select $x \leftarrow \mathrm{argmax}_{x' \in \mathcal{X}} F(x' \cup S) - F(S)$
  - Provide $x, h^*(x)$ to learner
  - Update $S \leftarrow S \cup \{x\}$

Approximation guarantees

Theorem: Let S^{gr} be the set provided by the algorithm and $\overleftrightarrow{S}^{\mathrm{opt}}$ denote the optimal teaching sequence. Then, $|S^{\mathrm{gr}}| \leq |\overleftrightarrow{S}^{\mathrm{opt}}| \cdot \log(|\mathcal{H}|)$.

Complexity Measures: TD

Notion of teaching complexity: Teaching dimension TD

- Introduced by [Goldman, Kearns ’95]
- Analysis setting
  - randomized version space learner
  - worst-case analysis
  - finite size hypothesis class
  - exact teaching

Formal definition of TD

- Length of optimal teaching sequence for h^* is $|TS(h^*; \mathcal{H}, \mathcal{X})|$
- Teaching dimension is defined as

$$TD(\mathcal{H}, \mathcal{X}) := \max_{h^* \in \mathcal{H}} |TS(h^*; \mathcal{H}, \mathcal{X})|$$

Complexity Measures: TD

Examples for computing TD

	x_1	x_2	x_3	x_4	x_5	$\lvert TS(h^*)\rvert$
h_1	1	1	1	1	1	1
h_2	0	1	1	1	1	2
h_3	0	0	1	1	1	2
h_4	0	0	0	1	1	2
h_5	0	0	0	0	1	2

$$TD(\mathcal{H}, \mathcal{X}) = 2$$

	x_1	x_2	x_3	x_4	$\lvert TS(h^*)\rvert$
h_1	1	0	0	0	1
h_2	0	1	0	0	1
h_3	0	0	1	0	1
h_4	0	0	0	1	1
h_5	0	0	0	0	4

$$TD(\mathcal{H}, \mathcal{X}) = 4$$

Complexity Measures: TD vs. VCD

Notion of learning complexity: VCD

- Introduced by [Vapnik, Chervonenkis '71]
- Sample complexity bounds for learning grow as $\Theta(VCD(\mathcal{H}, \mathcal{X}))$

A fundamental question: TD vs. VCD?

- $TD(\mathcal{H}, \mathcal{X})$ is $O(VCD(\mathcal{H}, \mathcal{X}))$?
- There exists problems with
  - $TD(\mathcal{H}, \mathcal{X}) \ll O(VCD(\mathcal{H}, \mathcal{X}))$
  - $TD(\mathcal{H}, \mathcal{X}) \gg O(VCD(\mathcal{H}, \mathcal{X}))$

	x_1	x_2	x_3	x_4
h_1	1	0	0	0
h_2	0	1	0	0
h_3	0	0	1	0
h_4	0	0	0	1
h_5	0	0	0	0

$TD(\mathcal{H}, \mathcal{X}) = 4$

$VCD(\mathcal{H}, \mathcal{X}) = 1$

Improved Notions of TD: RTD

Teaching an "adversarial" learner: Classic TD

- Simple classes can be difficult to teach

Teaching a "cooperative" learner: Recursive TD (RTD)

- Introduced by [Zilles et al. @ COLT'08]
- $RTD(\mathcal{H}, \mathcal{X})$ is $O\big(VCD(\mathcal{H}, \mathcal{X})\big)$? [Simon, Zilles @ COLT'15]
- An active area of research
  - $O\big(d\ 2^d \log\log|\mathcal{H}|\big)$ [Moran et al. @ FOCS'15]
  - $O\big(d\ 2^d\big)$ [Chen et al. @ NIPS' 16]
  - $O(d^2)$ [Hu et al. @ COLT' 17]

where d denotes $VCD(\mathcal{H}, \mathcal{X})$

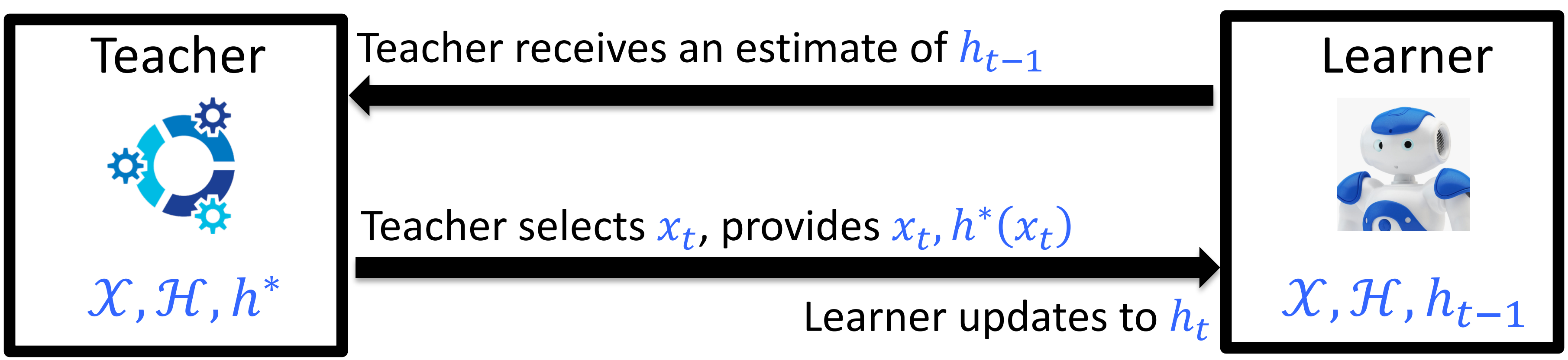
Improved Notions of TD: TD_σ

Teaching models for classic TD or RTD

- Order of examples and learner's feedback does not matter

Teaching a "state-dependent" learner: TD_σ

- Introduced in our recent work [NeurIPS'18, arXiv'19]
- Generalizes existing notions of teaching dimensions
- Provides necessary conditions when feedback matters

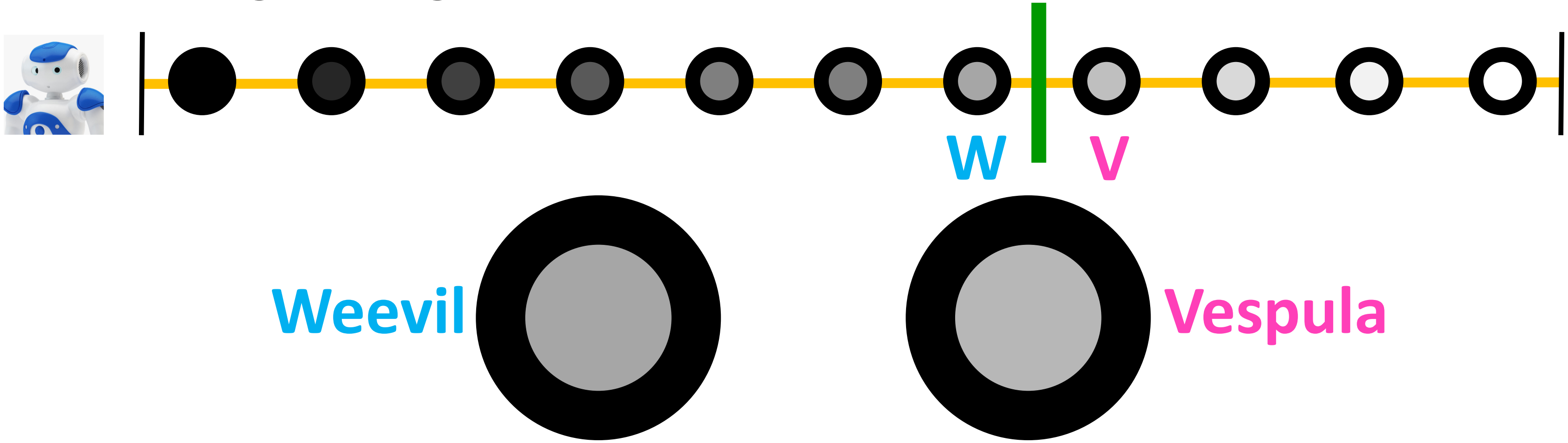


Teaching Binary Functions

- Understanding TD vs. VCD relation
  - see work by Sandra Zilles: http://www2.cs.uregina.ca/~zilles/
- Teaching complexity for ML models (e.g., SVM)
  - see work by Jerry Zhu: http://pages.cs.wisc.edu/~jerryzhu/

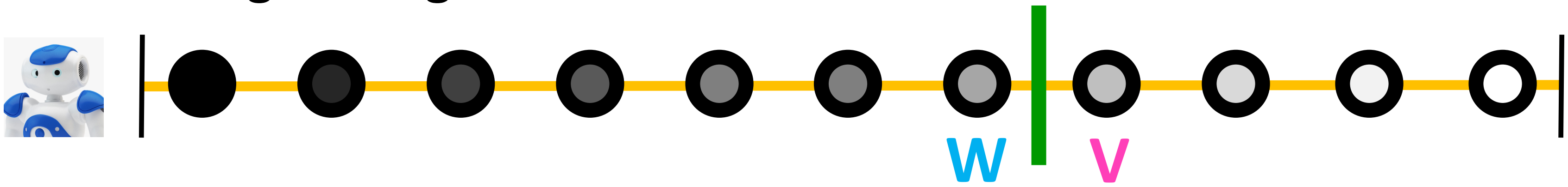
Teaching Binary Functions to People

- Teaching setting: size of D is 2



Teaching Binary Functions to People

- Teaching setting: size of D is 2

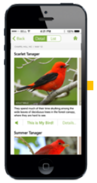


- Limited inference power and noise
- Mismatch in representation for $\mathcal{X}$, $\mathcal{H}$
- Limited memory
- Engagement
- Interpretability (e.g., teaching via labels vs. features)
- Safety (e.g., when teaching physical tasks)
- Fairness (e.g., when teaching a class)

More suitable for poisoning attacks, less for educational settings

Machine Teaching: Problem Space

- Type and complexity of task



- Type and model of learning agent

- Teacher’s knowledge and observability