Seminar on Machine Teaching

Saarland University – Winter Semester 2019

Adish Singla

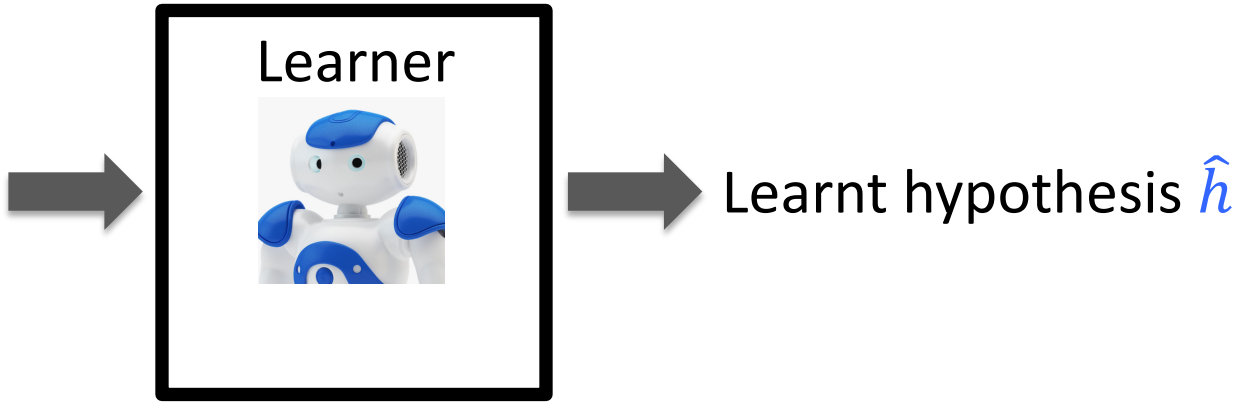
4th Nov 2019

MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

MAX-PLANCK-GESELLSCHAFT

Machine Learning vs. Teaching

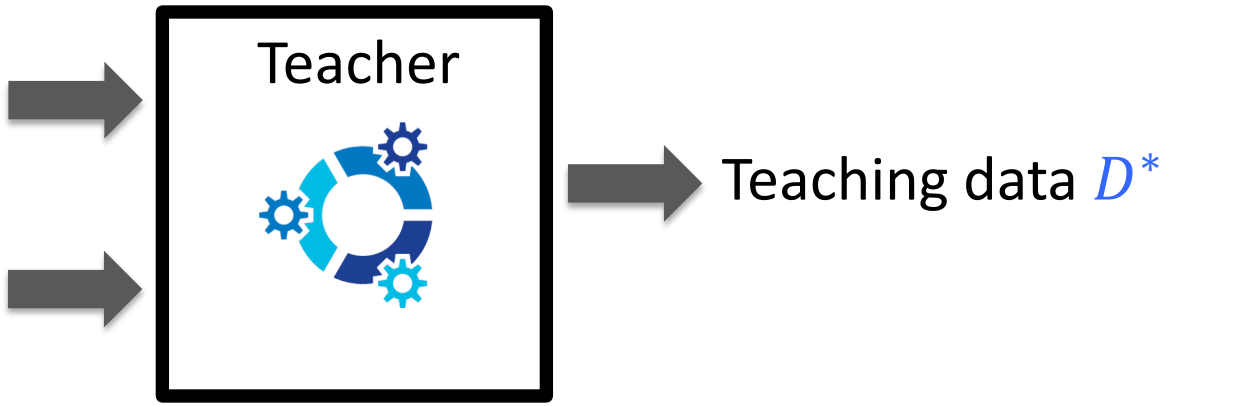
Training data D



Learnt hypothesis $\hat{h}$

Target hypothesis h^*

Learner model



Teaching data D^*

Why Machine Teaching?

Adversarial settings
aka training-set poisoning

coursera

edX

iNaturalist
Explore Learn Record

Educational settings

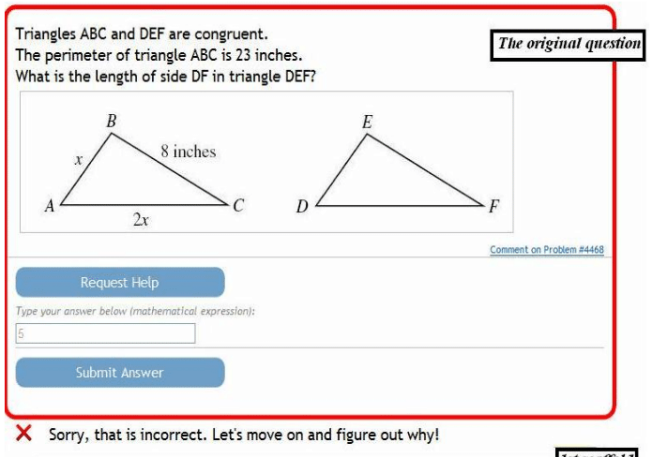
Applications: Online Education via MOOCs

- 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

VIRTAMED
WE SIMULATE REALITY

0:00 / 2:17

Video credits: Virtamed – Zurich, Switzerland

Applications: Language Learning

duolingo

BETA

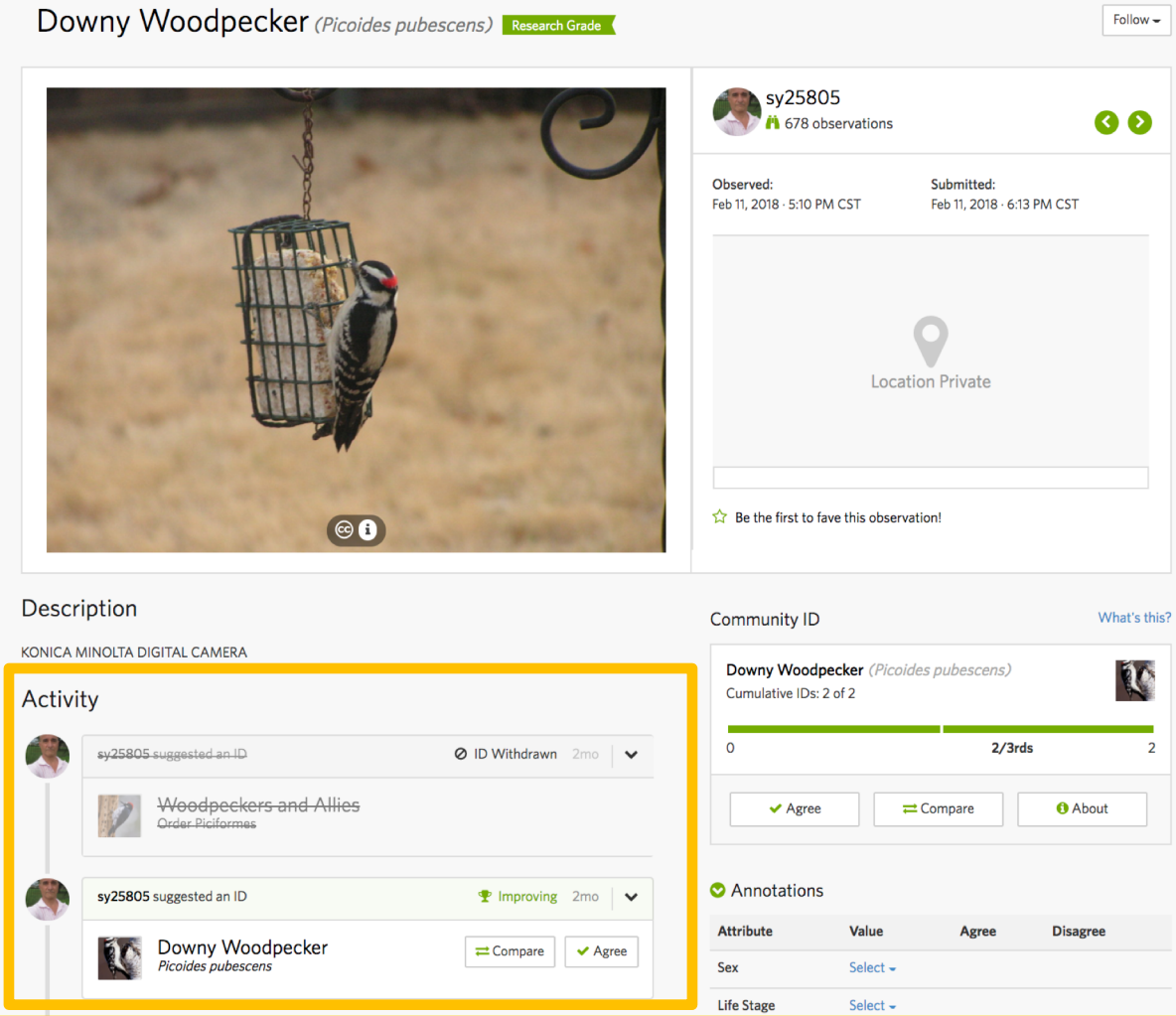
memrise

GROW YOUR MIND

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

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

Applications: Biodiversity Monitoring

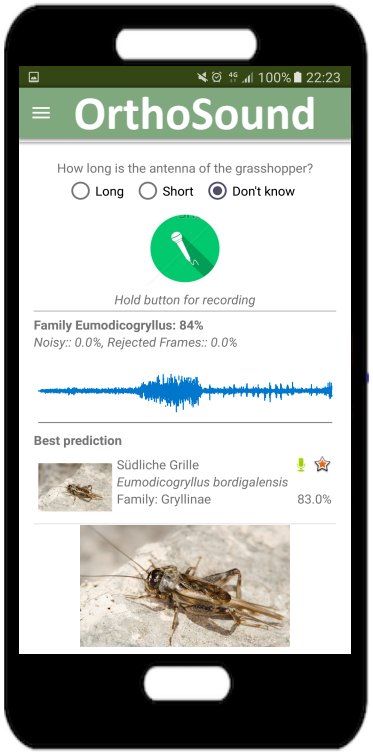


Key challenge: Noise in the annotations

Image credits: CornellLab – Global Big Day

Applications: Biodiversity Monitoring

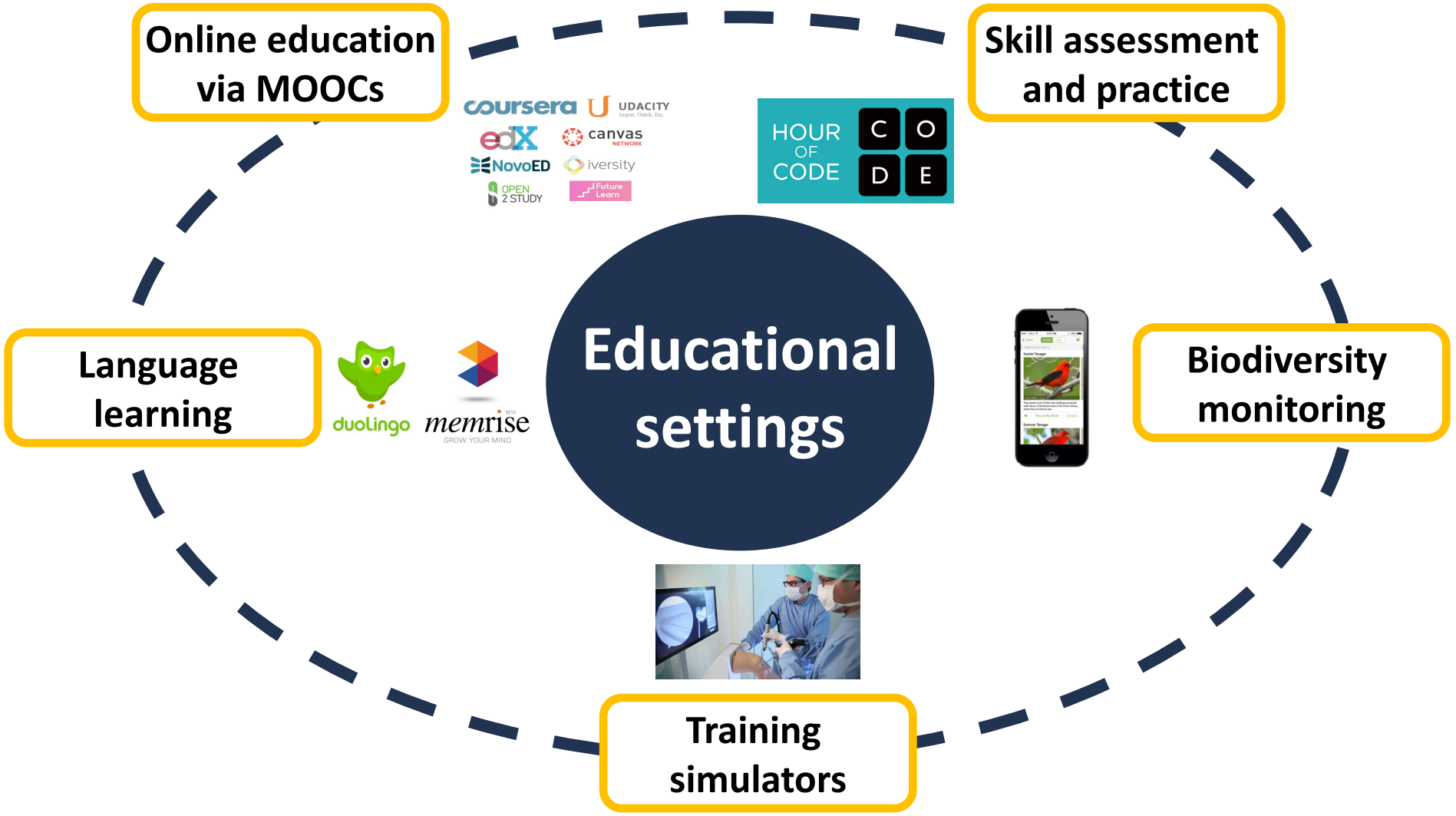
iNaturalist
Explore Learn Record



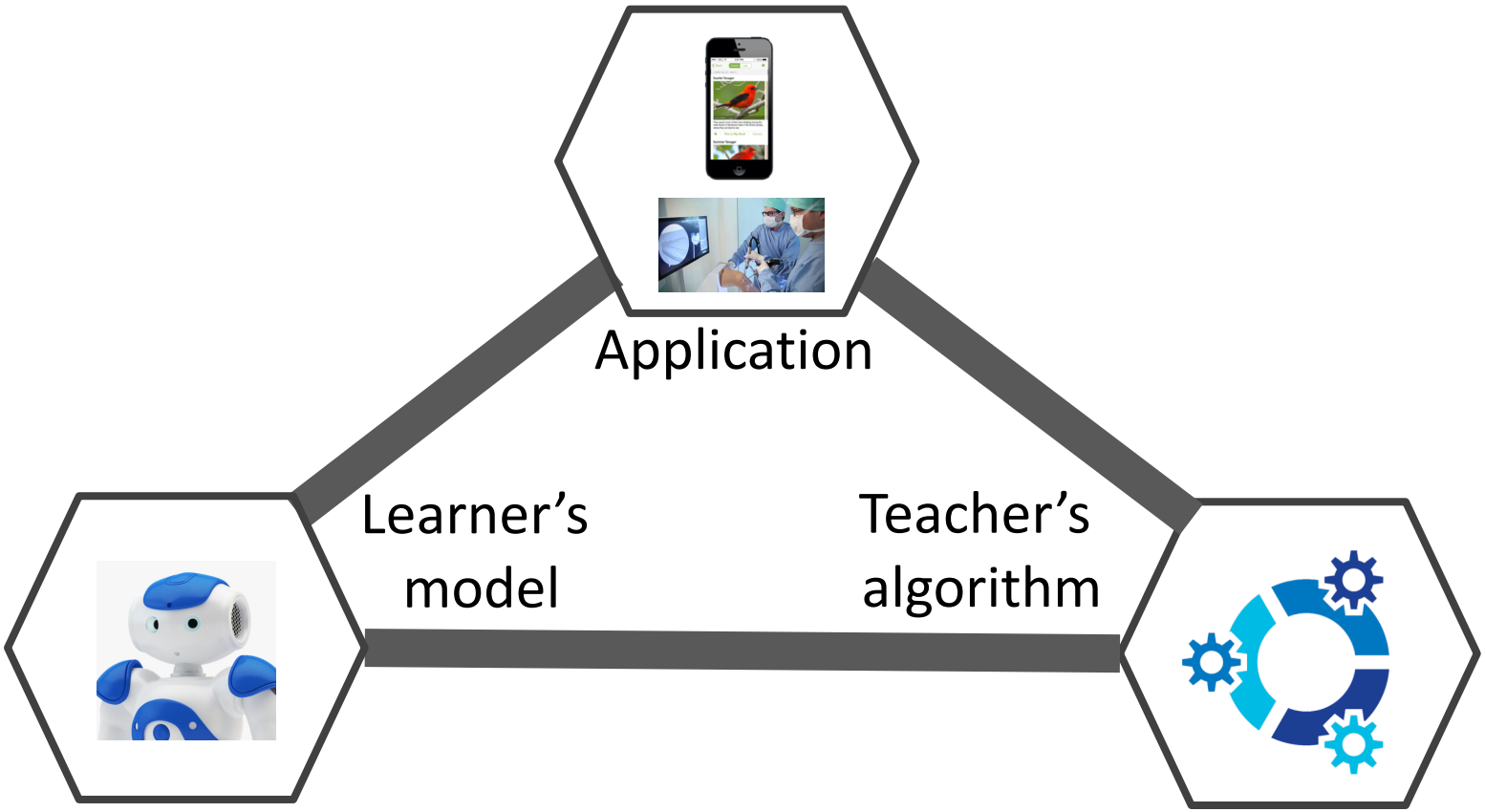
- Teaching helps increase awareness and engagement
- Labeled data is crucial for training machine learning systems

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

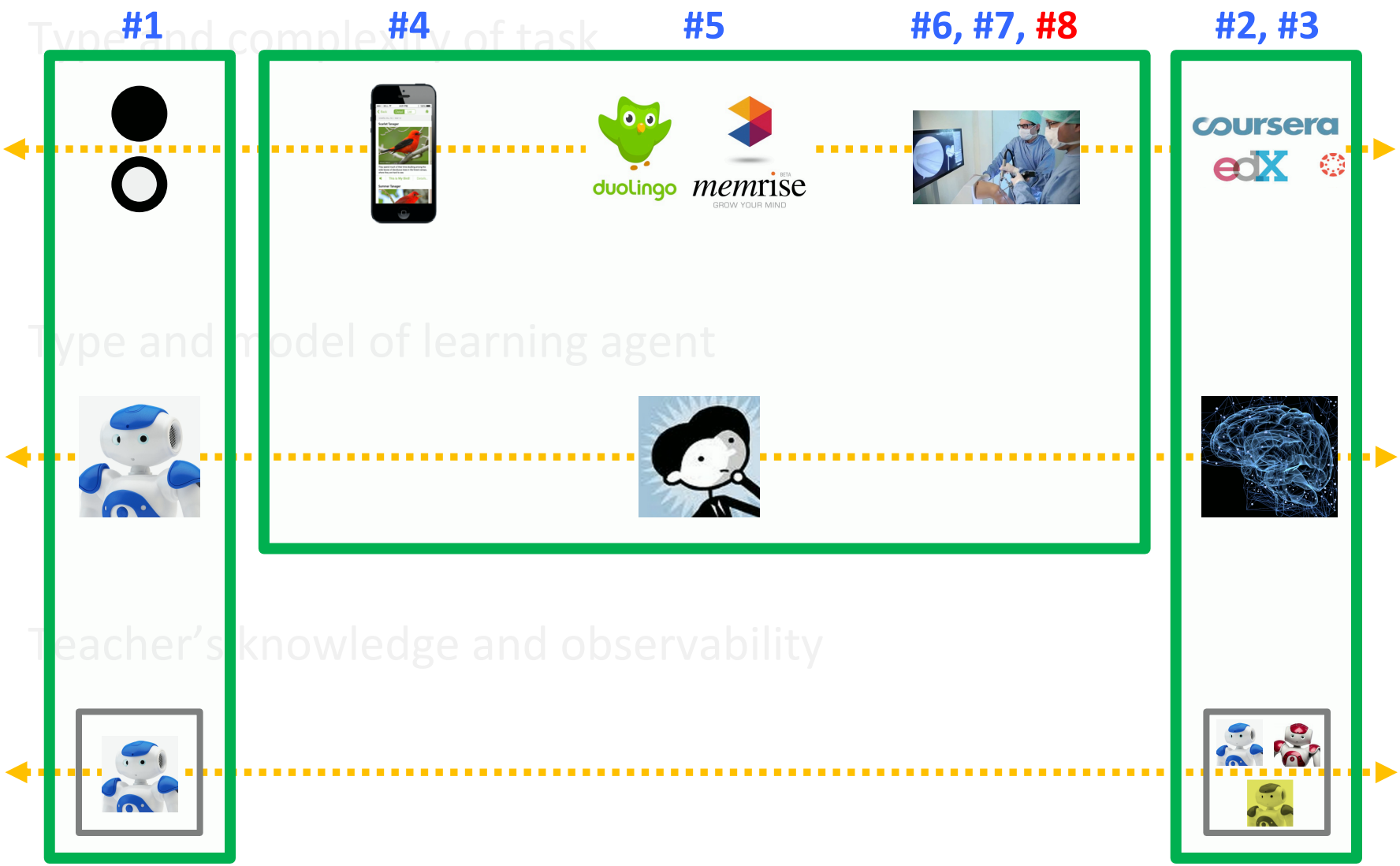
Machine Teaching: Applications



Machine Teaching: Key Components



Machine Teaching: Problem Space



Course Logistics: Different Parts

Part 1: Research Papers

- 1/3 of the score
- No formal feedback (each submission gets full points)
- Reports due by 10th Dec'19 (#1, 2, 3, 4) and by 10th Jan'20 (#5, 6, 7, 8)

Part 2: **Slides Preparation**

- 1/3 of the score
- Paper assignment on 11th Jan'20
- Final slides due by 15th Feb'20

Part 3: Final Talk

- 1/3 of the score
- Presentations to be scheduled between mid-Feb'20 to mid-Mar'20

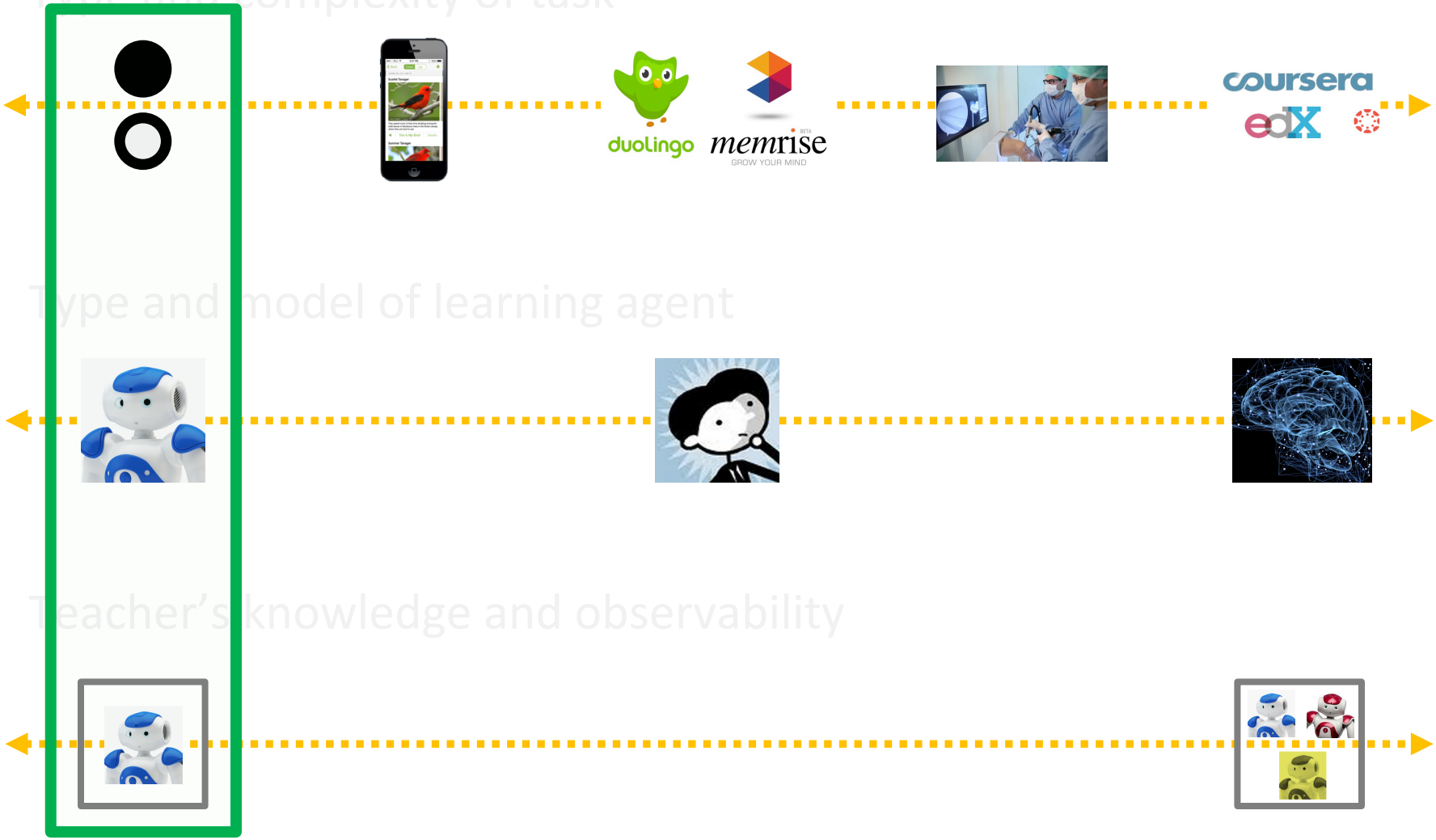
Course Logistics: Research Papers

- Each report submitted as **lastname_paper#.pdf**
  - singla_1.pdf, singla_2.pdf, …
- Typeset in latex using NeurIPS style files
  - https://neurips.cc/Conferences/2019/PaperInformation/StyleFiles
  - \usepackage[**preprint**]{neurips_2019} (Non-anonymous preprints)
- Structure the report as an extended review, e.g.,
  - Summarize the paper
  - Write down main strengths of this paper
  - Write down main weaknesses of this paper
  - Write down ways in which this paper could be improved
  - Write down ideas in which this work could be extended
  - …

Machine Teaching: Problem Space

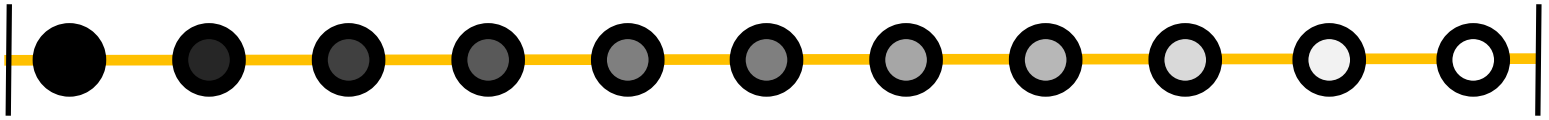
#1

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

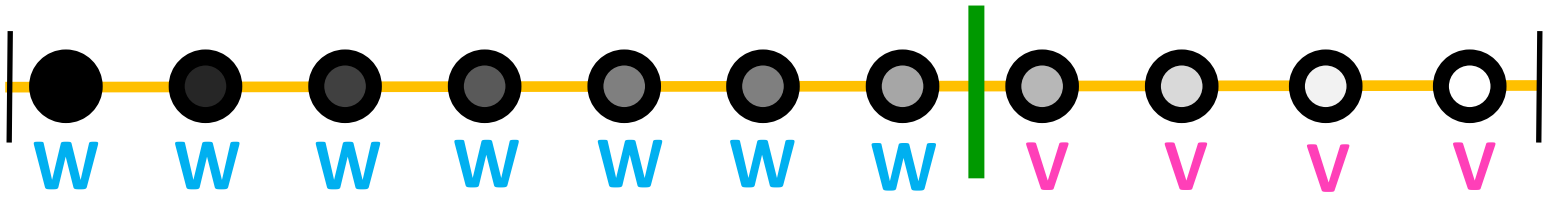


An Example: 1-D Threshold Function

- Task: Classify animal image as **Weevil**  or **Vespula** 
- $\mathcal{X}$: Set of images, each $x \in \mathcal{X}$ is associated with a contrast level

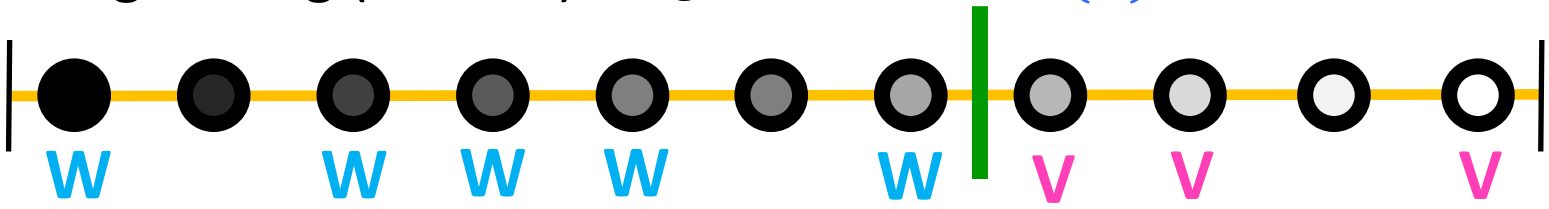


- $\mathcal{H}$: Set of hypotheses, each $h \in \mathcal{H}$ is a binary threshold classifier
- h^*: True classifier

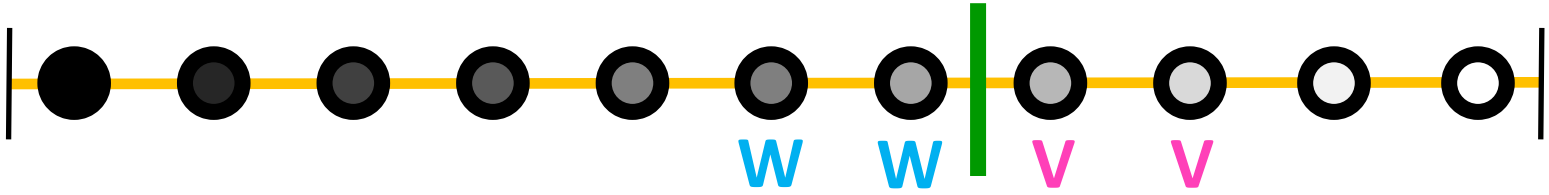


An Example: 1-D Threshold Function

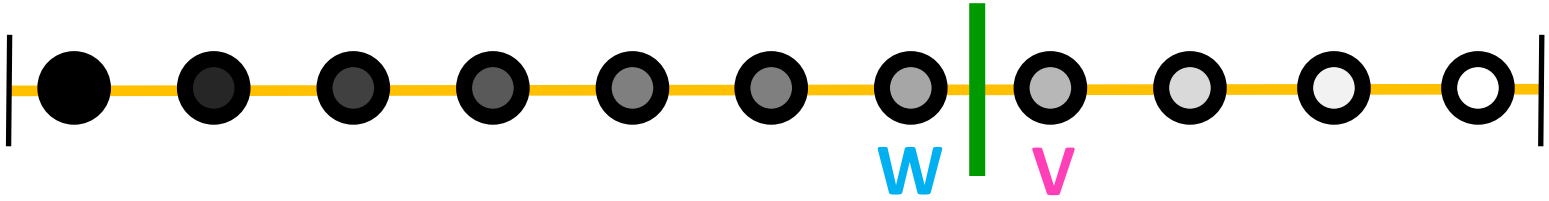
- Learning setting (Passive): avg. size of D is $\Theta(n)$



- 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 $h^* \in \mathcal{H}$

$\mathcal{X}$

$\mathcal{H}$	x_1	x_2	x_3	x_4	x_5
h_1	1	1	1	1	1
h_2	0	1	1	1	1
h_3	0	0	1	1	1
$h^* = h_4$	0	0	0	1	1
h_5	0	0	0	0	1

$\mathcal{X}$

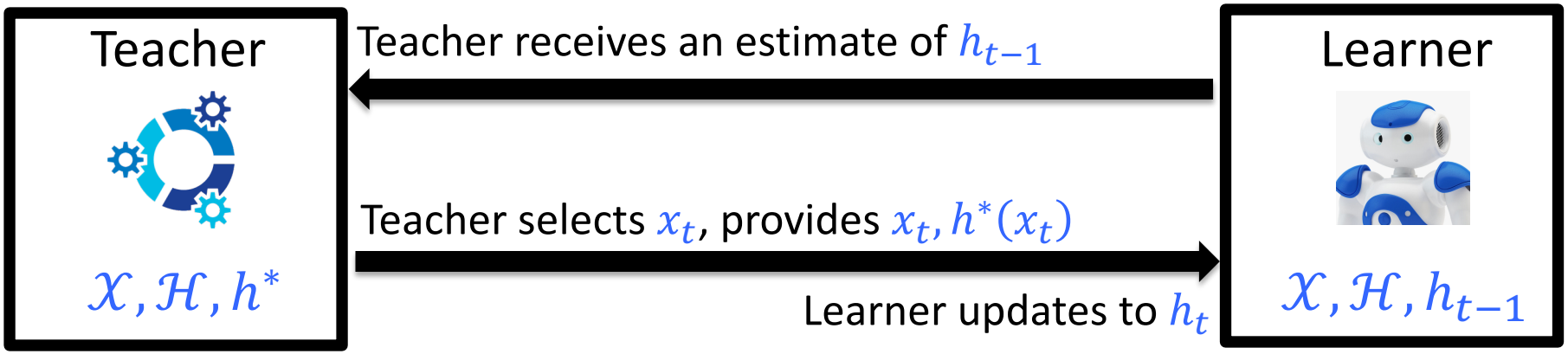
$\mathcal{H}$	x_1	x_2	x_3	x_4
h_1	1	0	0	0
h_2	0	1	0	0
h_3	0	0	1	0
h_4	0	0	0	1
$h^* = h_5$	0	0	0	0

Teaching Interaction

Start

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

At time t



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

$$\overleftrightarrow{S}^{\text{opt}} = \underset{\overleftrightarrow{S}}{\operatorname{argmin}} |\overleftrightarrow{S}| \quad \text{s.t.} \quad h_t = h^*$$

equivalent to

$$H_t = \{h^*\}$$

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 $\mathcal{H} \backslash h^*$

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 $F: 2^{\mathcal{X}} \rightarrow \mathbb{R}_+$ as

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

- Rewrite teaching problem as

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

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 $x \leftarrow \text{argmax}_{x' \in \mathcal{X}} F(x' \cup S) - F(S)$
  - 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 $\overleftrightarrow{S}^{\text{opt}}$ denote the optimal teaching sequence. Then, $|S^{\text{gr}}| \leq |\overleftrightarrow{S}^{\text{opt}}| \cdot \log(|\mathcal{H}|)$.