Understanding the Role of Adaptivity in Machine Teaching: The Case of Version Space Learners

Yuxin Chen†  Adish Singla‡  Oisin Mac Aodha†
Pietro Perona†  Yisong Yue†

†Caltech, {chenyux, macaodha, perona, yyue}@caltech.edu,
‡MPI-SWS, adish@mpi-sws.org

Abstract

In real-world applications of education, an effective teacher adaptively chooses the next example to teach based on the learner’s current state. However, most existing work in algorithmic machine teaching focuses on the batch setting, where adaptivity plays no role. In this paper, we study the case of teaching consistent, version space learners in an interactive setting. At any time step, the teacher provides an example, the learner performs an update, and the teacher observes the learner’s new state. We highlight that adaptivity does not speed up the teaching process when considering existing models of version space learners, such as the “worst-case” model (the learner picks the next hypothesis randomly from the version space) and the “preference-based” model (the learner picks hypothesis according to some global preference). Inspired by human teaching, we propose a new model where the learner picks hypotheses according to some local preference defined by the current hypothesis. We show that our model exhibits several desirable properties, e.g., adaptivity plays a key role, and the learner’s transitions over hypotheses are smooth/interpretable. We develop adaptive teaching algorithms, and demonstrate our results via simulation and user studies.

1 Introduction

Algorithmic machine teaching studies the interaction between a teacher and a student/learner where the teacher’s objective is to find an optimal training sequence to steer the learner towards a desired goal [36]. Recently, there has been a surge of interest in machine teaching as several different communities have found connections to this problem setting: (i) machine teaching provides a rigorous formalism for a number of real-world applications including personalized educational systems [35], adversarial attacks [24], imitation learning [6, 14], and program synthesis [18]; (ii) the complexity of teaching (“Teaching-dimension”) has strong connections with the information complexity of learning (“VC-dimension”) [9]; and (iii) the optimal teaching sequence has properties captured by new models of interactive machine learning, such as curriculum learning [4] and self-paced learning [28].

In the above-mentioned applications, adaptivity clearly plays an important role. For instance, in automated tutoring, adaptivity enables personalization of the content based on the student’s current knowledge [31, 55, 17]. In this paper, we explore the adaptivity gain in algorithmic machine teaching, i.e., how much speedup a teacher can achieve via adaptively selecting the next example based on the learner’s current state? While this question has been well-studied in the context of active learning and sequential decision making [15], the role of adaptivity is much less understood in algorithmic machine teaching. A deeper understanding would, in turn, enable us to develop better teaching algorithms and more realistic learner models to exploit the adaptivity gain.

We consider the well-studied case of teaching a consistent, version space learner. A learner in this model class maintains a version space (i.e., a subset of hypotheses that are consistent with the examples received from a teacher) and outputs a hypothesis from this version space. Here, a hypothesis can be viewed as a function that assigns a label to any unlabeled example. Existing work has studied this class of learner model to establish theoretical connections between the information complexity of teaching vs. learning [13, 37, 11]. Our main objective is to understand, when and by how much, a teacher can benefit by adapting the next example based on the learner’s current hypothesis. We compare two types of teachers: (i) an adaptive teacher that observes the learner’s hypothesis at every time step, and (ii) a non-adaptive teacher that only knows the initial hypothesis of the learner and does not receive any feedback during teaching. The non-adaptive teacher operates in a batch setting where the complete sequence of examples can be constructed before teaching begins.

Inspired by real-world teaching scenarios and as a generalization of the global “preference-based” model [11], we propose a new model where the learner’s choice of next hypothesis $h'$ depends on some local preferences defined by the current hypothesis $h$. For instance, the local preference could encode that the learner prefers to make smooth transitions by picking a consistent hypothesis $h'$ which is “close” to $h$. Local preferences, as seen in Fig. 1, are an important aspect of many machine learning algorithms (e.g., incremental or online learning algorithms [27, 28]) in order to increase robustness and reliability. We present results in the context of two different hypotheses classes, and show through simulation and user studies that adaptivity can play a crucial role when teaching learners with local preferences.

2 Related Work

Models of version space learners Within the model class of version space learners, there are different variants of learner models depending upon their anticipated behavior, and these models lead to different notions of teaching complexity. For instance, (i) the “worst-case” model [13] essentially assumes nothing and the learner’s behavior is completely unpredictable, (ii) the “cooperative” model [37] assumes a smart learner who anticipates that she is being taught, and (iii) the “preference-based” model [11] assumes that she has a global preference over the hypotheses. Recently, some teaching complexity results have been extended beyond version space learners, such as Bayesian learners [34], probabilistic/randomized learners [30, 3], learners implementing an optimization algorithm [22], and for iterative learning algorithms based on gradient updates [23]. Here, we focus on the case of version space learners, leaving the extension to other types of learners for future work.

Batch vs. sequential teaching Most existing work on algorithmic machine teaching has focused on the batch setting, where the teacher constructs a set of examples and provides it to the learner at the beginning of teaching [13, 37, 11, 7]. There has been some work on sequential teaching models that are more suitable for understanding the role of adaptivity. Recently, [23] studied the problem of iteratively teaching a gradient learner by providing a sequence of carefully constructed examples. However, since the learner’s update rule is completely deterministic, a non-adaptive teacher with knowledge of the learner’s initial hypothesis $h^0$ would behave exactly the same as an adaptive teacher (i.e., the adaptivity gain is zero). [5] studied randomized version-space learners with limited memory, and demonstrated the power of adaptivity for a specific class of hypotheses. Sequential teaching has also been studied in the context of crowdsourcing applications by [19] and [29], empirically demonstrating the improved performance of adaptive vs. non-adaptive teachers. However, these approaches do not provide any theoretical understanding of the adaptivity gain as done in our work.

Incremental learning and teaching Our learner model with local preferences is quite natural in real-world applications. A large class of iterative machine learning algorithms are based on the idea of incremental updates which in turn is important for the robustness and generalization of learning [27, 28]. From the perspective of a human learner, the notion of incremental learning aligns well with the concept of the “Zone of Proximal Development (ZPD)” in the educational research and psychology literature [32]. The ZPD suggests that teaching is most effective when focusing on a task slightly beyond the current abilities of the student as the human learning process is inherently
We now introduce a generic model of the learner by formalizing our assumptions about how she
acts. In our model, the learner has a preference function that encodes her transition preferences.

\[ \sigma(h'; h) \]

The preference function \( \sigma \) encodes the learner’s transition preferences. Consider that the
learner’s current hypothesis is \( h \), and there are two hypotheses \( h', h'' \) that she could possibly pick as
the next hypothesis. We want to encode whether the learner has any preference in choosing
between \( h' \) and \( h'' \). Formally, we define the preference function as
\[ \sigma : \mathcal{H} \times \mathcal{H} \rightarrow \mathbb{R}_+ \].

Intuitively, the preference function encodes the learner’s transition preferences. Consider that the
learner’s current hypothesis is \( h \), and there are two hypotheses \( h', h'' \) that she could possibly pick as
the next hypothesis. We want to encode whether the learner has any preference in choosing
between \( h' \) and \( h'' \). Formally, we define the preference function as
\[ \sigma : \mathcal{H} \times \mathcal{H} \rightarrow \mathbb{R}_+ \].

Consider that the learner’s current hypothesis is \( h \), and there are two hypotheses \( h', h'' \) that she could possibly pick as
the next hypothesis. We want to encode whether the learner has any preference in choosing
between \( h' \) and \( h'' \). Formally, we define the preference function as
\[ \sigma : \mathcal{H} \times \mathcal{H} \rightarrow \mathbb{R}_+ \].

\[ \sigma(h'; h) = \sigma(h''; h) \]

The learner starts with an initial hypothesis \( h_0 \in \mathcal{H} \) before receiving any labeled examples from
the teacher. Then, the interaction between the teacher and the learner proceeds in discrete time steps. At
any time step \( t \), let us denote the labeled examples received by the learner up to (but not including)
time step \( t \) via a set \( Z_t \). At time step \( t \), the learner’s version space as \( \mathcal{H}_t = \mathcal{H}(Z_t) \), and the current hypothesis as \( h_t \).

At time step \( t \), we model the learning dynamics as follows: (i) the learner receives a new example
\( z_t \); and (ii) the learner updates the version space \( \mathcal{H}_{t+1} \), and picks the next hypothesis based on the
current hypothesis \( h_t \), version space \( \mathcal{H}_{t+1} \), and the preference function \( \sigma \):
\[ h_{t+1} \in \{ h \in \mathcal{H}_{t+1} : \sigma(h; h_t) = \frac{1}{\sum_{h \\in \mathcal{H}_{t+1}} \sigma(h'; h_t)} \}. \]  (3.1)
finishes at time step $t$ if the learner’s hypothesis $h_t = h^*$. Our objective is to design teaching algorithms that can achieve this goal in a minimal number of time steps. We study the \textit{worst-case} number of steps needed as is common when measuring the information complexity of teaching \cite{13,37,11}.

We assume that the teacher knows the learner’s initial hypothesis $h_0$ as well as the preference function $\sigma(\cdot; \cdot)$. In order to quantify the gain from adaptivity, we compare two types of teachers: (i) an \textit{adaptive teacher} who observes the learner’s hypothesis $h_t$ before providing the next labeled example $z_t$ at any time step $t$; and (ii) a \textit{non-adaptive teacher} who only knows the initial hypothesis of the learner and does not receive any feedback from the learner during the teaching process. Given these two types of teachers, we want to measure the \textit{adaptivity gain} by quantifying the difference in teaching complexity of the \textit{optimal adaptive} teacher compared to the \textit{optimal non-adaptive} teacher.

\section{The Role of Adaptivity}

In this section, we study different variants of the learner’s preference function, and formally state the adaptivity gain with two concrete problem instances.

\subsection{State-independent Preferences}

We first consider a class of preference models where the learner’s preference about the next hypothesis does not depend on her current hypothesis. The simplest state-independent preference is captured by the “worst-case” model \cite{13}, where the learner’s preference over all hypotheses is uniform, i.e., $\forall h, h', \sigma(h'; h) = c$, where $c$ is some constant.

A more generic state-independent preference model is captured by non-uniform, global preferences. More concretely, for any $h' \in \mathcal{H}$, we have $\sigma(h'; h) = c_{h'} \forall h \in \mathcal{H}$, a constant dependent only on $h'$. This is similar to the notion of the global “preference-based” version space learner introduced by \cite{11}.

\textbf{Proposition 1} \textit{For the state-independent preference, adaptivity plays no role, i.e., the sample complexities of the optimal adaptive teacher and the optimal non-adaptive teacher are the same.}

In fact, for the uniform preference model, the teaching complexity of the adaptive teacher is the same as the teaching dimension of the hypothesis class with respect to teaching $h^*$, given by

$$TD(h^*, \mathcal{H}) := \min_Z |Z|, \text{ s.t. } \mathcal{H}(Z) = \{h^*\}. \quad (4.1)$$

For the global preference model, similar to the notion of \textit{preference-based teaching dimension} \cite{11}, the teaching complexity of the adaptive teacher is given by

$$\min_Z |Z|, \text{ s.t. } \forall h \in \mathcal{H}(Z) \backslash \{h^*\}, \sigma(h; \cdot) > \sigma(h^*; \cdot). \quad (4.2)$$

\subsection{State-dependent Preferences}

In real-world teaching scenarios, human learners incrementally build up their knowledge of the world, and their preference of the next hypothesis naturally depends on their current state. To better understand the behavior of an adaptive teacher under a state-dependent preference model, we investigate the following two concrete examples:

\textbf{Example 1 (2-REC)} \textit{$\mathcal{H}$ consists of up to two disjoint rectangles on a grid and $X$ represents the grid cells (cf. Fig. 1 and Fig. 3a). Consider an example $z = (x_z, y_z) \in Z$: $y_z = 1$ (positive) if the grid cell $x_z$ lies inside the target hypothesis, and 0 (negative) elsewhere.}

The 2-REC hypothesis class consists of two subclasses, namely $\mathcal{H}^1$: all hypotheses with one rectangle, and $\mathcal{H}^2$: those with exactly two (disjoint) rectangles. The 2-REC class is inspired by teaching a union of disjoint objects. Here, objects correspond to rectangles and any $h \in \mathcal{H}$ represents one or two rectangles. Furthermore, each hypothesis $h$ is associated with a complexity measure given by the number of objects in the hypothesis. \cite{10} recently studied the problem of teaching a union of disjoint geometric objects, and \cite{11} studied the problem of teaching a union of monomials. Their results show

\footnote{For simplicity of discussion, we assume that for the 2-REC hypothesis that contains two rectangles, the edges of the two rectangles do not overlap.}
that, in general, teaching a target hypothesis of lower complexity from higher complexity hypotheses is the most challenging task.

For the 2-REC class, we assume the following local preferences: (i) in general, the learner prefers to transition to a hypothesis with the same complexity as the current one (i.e., $\mathcal{H}_1 \rightarrow \mathcal{H}_1$ or $\mathcal{H}_2 \rightarrow \mathcal{H}_2$), (ii) when transitioning within the same subclass, the learner prefers small edits, e.g., by moving the smallest number of edges possible when changing their hypothesis, and (iii) the learner could switch to a subclass of lower complexity (i.e., $\mathcal{H}_2 \rightarrow \mathcal{H}_1$) in specific cases. We provide a detailed description of the preference function in the extended version of this paper \cite{8}.

**Example 2 (LATTICE)** $\mathcal{H}$ and $\mathcal{X}$ both correspond to nodes in a 2-dimensional integer lattice of length $n$. For a node $v$ in the grid, we have an associated $h_v \in \mathcal{H}$ and $x_v \in \mathcal{X}$. Consider an example $z_v = (x_{z_v}, y_{z_v}) \in \mathbb{Z}$: $y_{z_v} = 0$ (negative) if the target hypothesis corresponds to the same node $v$, and $1$ (positive) elsewhere. We consider the problem of teaching with positive-only examples.

LATTICE class is inspired by teaching in a physical world from positive-only (or negative-only) reinforcements, for instance, teaching a robot to navigate to a target state by signaling that the current location is not the target. The problem of learning and teaching with positive-only examples is an important question with applications to learning languages and reinforcement learning tasks \cite{12, 20}. For the LATTICE class, we assume that the learner prefers to move to a close-by hypothesis measured via $L_1$ (Manhattan) distance, and when hypotheses have equal distances we assume that the learner prefers hypotheses with larger coordinates.

**Theorem 2** For teaching the 2-REC class, the ratio between the cost of the optimal non-adaptive teacher and the optimal adaptive teacher is $\Omega((|h_0| \log |h_0|))$, where $|h_0|$ denotes the number of positive examples induced by the learner’s initial hypothesis $h_0$; for teaching the LATTICE class, the difference between the cost of the optimal non-adaptive teacher and the optimal adaptive teacher is $\Omega(n)$.

In the above theorem, we show that for both problems, under natural behavior of an incremental learner, adaptivity plays a key role. The proof of Theorem 2 is provided in the extended version of this paper \cite{8}. Specifically, we show the teaching sequences for an adaptive teacher which matches the above bounds for the 2-REC and LATTICE classes. We also provide lower bounds for any non-adaptive algorithms for these two classes. Here, we highlight two necessary conditions under which adaptivity can possibly help: (i) preferences are local and (ii) there are ties among the learner’s preference over hypotheses. The learner’s current hypothesis, combined with the local preference structure, gives the teacher a handle to steer the learner in a controlled way.

5 Adaptive Teaching Algorithms

In this section, we first characterize the optimal teaching algorithm, and then propose a non-myopic adaptive teaching framework.

5.1 The Optimality Condition

Assume that the learner’s current hypothesis is $h$, and the current version space is $H \subseteq \mathcal{H}$. Let $D^*(h, H, h^*)$ denote the minimal number of examples required in the worst-case to teach $h^*$. We identify the following optimality condition for an adaptive teacher:

**Proposition 3** A teacher achieves the minimal teaching cost, if and only if for all states $(h, H)$ of the learner, it picks an example such that

$$z^* \in \arg \min_z \left( 1 + \max_{h' \in \mathcal{C}(h, H, \sigma, z)} D^*(h', H \cap \mathcal{H}(\{z\}), h^*) \right)$$

where $\mathcal{C}(h, H, \sigma, z)$ denotes the set of candidate hypotheses in the next round as defined in (3.1), and for all $(h, H)$, it holds that

$$D^*(h, H, h^*) = \min_z \left( 1 + \max_{h' \in \mathcal{C}(h, H, \sigma, z)} D^*(h', H \cap \mathcal{H}(\{z\}), h^*) \right)$$

5
For any condition provided in Theorem 4 do not hold, the myopic heuristic (5.1) could perform poorly. An important observation from Theorem 4 is that, when $D(h, h^\ast)$ is small, i.e., $h^\ast$ is close to $h$, the cost of the myopic algorithm that greedily minimizes Eq. (5.1) to attain provable guarantees:

$$D(h, h^\ast) = |\{h' \in H : \sigma(h'; h) \leq \sigma(h^\ast; h)\}|$$

In words, $D$ denotes the index of the target hypothesis $h^\ast$ in the preference vector associated with $h$ in the version space $H$. Notice that for the uniform (resp. global) preference model, the function $D$ reduces to $D_u$ (resp. $D_g$). In the following theorem, we provide a sufficient condition for the myopic adaptive algorithm that greedily minimizes Eq. (5.1) to attain provable guarantees:

**Theorem 4** Let $h_0 \in \mathcal{H}$ be the learner’s initial hypothesis, and $h^\ast \in \mathcal{H}$ be the target hypothesis. For any $H \subseteq \mathcal{H}$, let $H(\{z\}) = \{h' \in H : h'(x_z) \neq y_z\}$ be the set of hypotheses in $H$ which are inconsistent with the teaching example $z \in \mathcal{Z}$. If for all learner’s states $(h, H)$, the preference and the structure of the teaching examples satisfy:

1. $\forall h_i, h_j \in H$, $\sigma(h_i; h) \leq \sigma(h_j; h) \leq \sigma(h^\ast; h) \implies \sigma(h_j; h_i) \leq \sigma(h^\ast; h_i)$
2. $\forall H' \subseteq H(\{z\})$, there exists $z' \in \mathcal{Z}$, s.t., $H(\{z'\}) = H'$,

then, the cost of the myopic algorithm that greedily minimizes Eq. (5.1) is within a factor of $2(\log D(h_0, H, h^\ast) + 1)$ approximation of the cost of the optimal adaptive algorithm.

We defer the proof of the theorem to the extended version of this paper [3]. Note that both the uniform preference model and the global preference model satisfy Condition 1. Intuitively, the first condition states that there does not exist any hypothesis between $h$ and $h^\ast$ that provides a “short-cut” to the target. Condition 2 implies that we can always find teaching examples that ensure smooth updates of the version space. For instance, a feasible setting that fits Condition 2 is where we assume that the teacher can synthesize an example to remove any subset of hypotheses of size at most $k$, where $k$ is some constant.

### 5.2 Non-Myopic Teaching Algorithms

When the conditions provided in Theorem 4 do not hold, the myopic heuristic (5.1) could perform poorly. An important observation from Theorem 4 is that, when $D(h, H, h^\ast)$ is small, i.e., $h^\ast$ is close to $h$, the cost of the myopic algorithm that greedily minimizes Eq. (5.1) is within a factor of $2(\log D(h_0, H, h^\ast) + 1)$ approximation of the cost of the optimal adaptive algorithm.
to the learner’s current hypothesis in terms of preference ordering, we need less stringent constraints on the preference function. This motivates adaptively devising intermediate target hypotheses to ground the teaching task into multiple, separate sub-tasks. Such divide-and-conquer approaches have proven useful for many practical problems, e.g., constructing a hierarchical decomposition for reinforcement learning tasks [16]. In the context of machine teaching, we assume that there is an oracle, Oracle($h, H, h^*$) that maps the learner’s state ($h, H$) and the target hypothesis $h^*$ to an intermediate target hypothesis, which defines the current sub-task.

We outline the non-myopic adaptive teaching framework in Algorithm 1. Here, the subroutine Teacher aims to provide teaching examples that bring the learner closer to the intermediate target hypothesis. As an example, let us consider the 2-REC hypothesis class. In particular, we consider the challenging case where the target hypothesis $h^* \in H^1$ represents a single rectangle $r^*$, and the learner’s initial hypothesis $h_0 \in H^2$ has two rectangles $(r_1, r_2)$. Imagine that the first rectangle $r_1$ is overlapping with $r^*$, and the second rectangle $r_2$ is disjoint from $r^*$. To teach the hypothesis $h^*$, the first sub-task (as provided by the oracle) is to eliminate the rectangle $r_2$ by providing negative examples so that the learner’s hypothesis represents a single rectangle $r_1$. Then, the next sub-task (as provided by the oracle) is to teach $h^*$ from $r_1$. We illustrate the sub-tasks in Fig. 2 and provide the full details of the adaptive teaching algorithm (i.e., Ada-R as used in our experiments) in the extended version of this paper [8].

6 Experiments

In this section, we empirically evaluate our teaching algorithms on the 2-REC hypothesis class via simulated learners.

6.1 Experimental Setup

For the 2-REC hypothesis class (cf. Fig. 3a and Example 1), we consider a grid with size varying from $5 \times 5$ to $20 \times 20$. The ground set of unlabeled teaching examples $X$ consists of all grid cells. In our simulations, we consider all four possible teaching scenarios, $H^{1\rightarrow1}, H^{1\rightarrow2}, H^{2\rightarrow1}, H^{2\rightarrow2}$, where $i, j$ in $H^{i\rightarrow j}$ specify the subclasses of the learner’s initial hypothesis $h_0$ and the target hypothesis $h^*$. In each simulated teaching session, we sample a random pair of hypotheses $(h_0, h^*)$ from the corresponding subclasses.

Teaching algorithms We consider three different teaching algorithms as described below. The first algorithm, SC, is a greedy set cover algorithm, where the teacher greedily minimizes $D_n = |H|$ (see §5.1). In words, the teacher acts according to the uniform preference model, and greedily picks the teaching example that eliminates the most inconsistent hypotheses in the version space. The second algorithm, denoted by Non-R for the class 2-REC, represents the non-adaptive teaching algorithm that matches the non-adaptive lower bounds provided in Theorem 2 with implementation details provided in the extended version of this paper [8]. Note that both SC and Non-R are non-adaptive. The third algorithm, Ada-R, represents the non-myopic adaptive teaching algorithm instantiated from Algorithm 1. The details of the subroutines Oracle and Teacher for Ada-R are provided in the extended version of this paper [8]. We note that all teaching algorithms have the same stopping criterion: the teacher stops when the learner reaches the target hypothesis, that is, $h_\tau = h^*$.

6.2 Results

We measure the performance of the teaching algorithms by their teaching complexity, and all results are averaged over 50 trials with random samples of $(h_0, h^*)$.

Noise-free setting Here, we consider the “noise-free” setting, i.e., the learner acts according to the state-dependent preference models as described in §4.2. In Fig. 3b, we show the results for 2-REC class with a fixed grid size $15 \times 15$ for all four teaching scenarios. As we can see from Fig. 3b, Ada-R has a consistent advantage over the non-adaptive baselines across all four scenarios. As expected, teaching $H^{1\rightarrow1}, H^{1\rightarrow2}, H^{2\rightarrow1}$ is easier, and the non-adaptive algorithms (SC and Non-R) perform well. In contrast, when teaching $H^{2\rightarrow2}$, we see a significant gain from Ada-R over the non-adaptive baselines. In the worst case, SC has to explore all the negative examples to teach $h^*$, whereas Non-R needs to consider all negative examples within the learner’s initial hypothesis.
We tested 215 participants, where each individual performed 10 trials on a grid of size $12 \times 12$. For each trial, we randomly selected one of the five types of sessions as discussed above. In Fig. 4a we see

---

1 In general, the teaching sequence constructed by the non-adaptive algorithms Non-R (resp. Non-L) would not be sufficient to reach the target under the noisy setting. Hence, we did not include the results of these non-adaptive algorithms in the robustness plots. Note that one can tweak Non-R (resp. Non-L) by concatenating the teaching sequence with teaching examples generated by SC; however, in general, in a worst-case sense, any non-adaptive algorithm in the noisy setting will not perform better than SC.

2 The performance of SC is non-monotone w.r.t. the noise-level. This is attributed to the stopping criteria of the algorithm as the increase in the noise level increases the chance for the learner to randomly jump to $h^*$. 

---

**Robustness in a noisy setting** In real-world teaching tasks, the learner’s preference may deviate from the preference $\sigma$ of an “ideal” learner that the teacher is modeling. In this experiment, we consider a more realistic scenario, where we simulate the noisy learners by randomly perturbing the preference of the “ideal” learner at each time step. With probability $1 - \varepsilon$ the learner follows $\sigma$, and with probability $\varepsilon$, the learner switches to a random hypothesis in the version space. In Fig. 3a we show the results for the 2-REC hypothesis class with different noise levels $\varepsilon \in [0, 1]$. We observe that even for highly noisy learners e.g., $\varepsilon = 0.9$, our algorithm Ada-R performs much better than SC.

---

**User Study**

We consider a two-step process for elicit the preference over hypotheses of human participants, and to (ii) evaluate our adaptive algorithm when teaching human learners.

---

In general, the teaching sequence constructed by the non-adaptive algorithms Non-R (resp. Non-L) would not be sufficient to reach the target under the noisy setting. Hence, we did not include the results of these non-adaptive algorithms in the robustness plots. Note that one can tweak Non-R (resp. Non-L) by concatenating the teaching sequence with teaching examples generated by SC; however, in general, in a worst-case sense, any non-adaptive algorithm in the noisy setting will not perform better than SC.

---

1 The performance of SC is non-monotone w.r.t. the noise-level. This is attributed to the stopping criteria of the algorithm as the increase in the noise level increases the chance for the learner to randomly jump to $h^*$. 

---

Figure 3: Illustration and simulation results for 2-REC. (a) illustrates the 2-REC hypothesis class. The initial hypothesis $h_0 \in \mathcal{H}^2$ is represented by the orange rectangles, and the target hypothesis $h^* \in \mathcal{H}^1$ is represented by the green rectangle. The green and blue cells represent a positive and a negative teaching example, respectively. Simulation results are shown in (b)-(d).
Teaching human learners  
Next we evaluate our teaching algorithms on human learners. As in the simulations, we consider four teaching scenarios $\mathcal{H}^{1\rightarrow1}$, $\mathcal{H}^{1\rightarrow2}$, $\mathcal{H}^{2\rightarrow1}$, and $\mathcal{H}^{2\rightarrow2}$. At the beginning of the teaching session, a participant was shown a blank $8 \times 8$ grid with either one or two initial rectangles, corresponding to $\mathcal{H}^1$. At every iteration, the participants were provided with a new teaching example (i.e., a new green or blue cell is revealed), and were asked to update the current hypothesis.

We evaluate three algorithms, namely Ada-R, SC, and Rand, where Rand denotes a teaching strategy that picks examples at random. The non-adaptive algorithm Non-R was not included in the user study for the same reasons as explained in Footnote \textsuperscript{3}. We enlisted 200 participants to evaluate teaching algorithms and this was repeated five times for each participant. For each trial, we randomly selected one of the three teaching algorithms and one of the four teaching scenarios. Then, we recorded the number of examples required to learn the target hypothesis. Teaching was terminated when 60\% of the cells were revealed. If the learner did not reach the target hypothesis by this time we set the number of teaching examples to this upper limit. We illustrate a teaching session in the extended version of this paper \cite{extended}.

Fig. \textsuperscript{4c} illustrates the superiority of the adaptive teacher Ada-R, while Rand performs the worst. In both cases where the target hypothesis is in $\mathcal{H}^2$, the SC teacher performs nearly as well as the adaptive teacher, as at most 12 teaching examples are required to fully characterize the location of both rectangles. However, we observe a large gain from the adaptive teacher for the scenario $\mathcal{H}^{2\rightarrow1}$.

8 Conclusions

We explored the role of adaptivity in algorithmic machine teaching and showed that the adaptivity gain is zero when considering well-studied learner models (e.g., “worst-case” and “preference-based”) for the case of version space learners. This is in stark contrast to real-life scenarios where adaptivity is an important ingredient for effective teaching. We highlighted the importance of local preferences (i.e., dependent on the current hypothesis) when the learner transitions to the next hypothesis. We presented hypotheses classes where such local preferences arise naturally, given that machines and humans have a tendency to learn incrementally. Furthermore, we characterized the structure of optimal adaptive teaching algorithms, designed near-optimal general purpose and application-specific adaptive algorithms, and validated these algorithms via simulation and user studies.

\textsuperscript{3}Given that a participant is allowed to move edges when updating the hypothesis, our interface could bias the participants’ choice of the next hypothesis towards a preference structure that favors local edits as assumed by our algorithm. As future work, one could consider an alternative interface which enforces participants to draw the rectangle(s) from scratch at every step.
Acknowledgments  This work was supported in part by Northrop Grumman, Bloomberg, AWS Research Credits, Google as part of the Visipedia project, and a Swiss NSF Early Mobility Postdoctoral Fellowship.

References


