Near-Optimally Teaching the Crowd to Classify

Adish Singla[†] ADISH.SINGLA@INF.ETHZ.CH
Ilija Bogunovic[*] ILIJA.BOGUNOVIC@EPFL.CH
Gábor Bartók[†] BARTOK@INF.ETHZ.CH
Amin Karbasi[†] AMIN.KARBASI@INF.ETHZ.CH
Andreas Krause[†] KRAUSEA@ETHZ.CH
† ETH Zürich, Zürich, Switzerland
* School of Computer and Communication Sciences, EPFL, Lausanne, Switzerland

Abstract

How should we present training examples to learners to teach them classification rules? This is a natural problem when training workers for crowdsourcing labeling tasks, and is also motivated by challenges in data-driven online education. We propose a natural stochastic model of the learners, modeling them as randomly switching among hypotheses based on observed feedback. We then develop STRICT, an efficient algorithm for selecting examples to teach to workers. Our solution greedily maximizes a submodular surrogate objective function in order to select examples to show to the learners. We prove that our strategy is competitive with the optimal teaching policy. Moreover, for the special case of linear separators, we prove that an exponential reduction in error probability can be achieved. Our experiments on simulated workers as well as three real image annotation tasks on Amazon Mechanical Turk show the effectiveness of our teaching algorithm.

1. Introduction

Crowdsourcing services, such as Amazon's Mechanical Turk platform (henceforth MTurk), are becoming vital for outsourcing information processing to large groups of workers. Machine learning, AI, and citizen science systems can hugely benefit from the use of these services as large-scale annotated data is often of crucial importance (Snow et al., 2008; Sorokin & Forsyth, 2008; Lintott et al., 2008). Data collected from such services however is often noisy, e.g., due to spamming, inexpert or careless workers (Sorokin & Forsyth, 2008). As the accuracy of the annotated data is often crucial, the problem of tackling noise from crowdsourcing services has received considerable attention. Most of the work so far has focused on methods for combining labels from many annotators (Welinder et al., 2010; Gomes et al., 2011; Dalvi et al., 2013) or in designing control measures by estimating the worker's reliabilities through "gold standard" questions (Snow et al., 2008).

In this paper, we explore an orthogonal direction: *can we teach workers in crowdsourcing services in order to improve their accuracy?* That is, instead of designing models and methods for determining workers' reliability, can we develop intelligent systems that teach workers to be more effective? While we focus on crowdsourcing in this paper, similar challenges arise in other areas of data-driven education. As running examples, in this paper we focus on crowdsourcing image labeling. In particular, we consider the task of classifying animal species, an important component in several citizen science projects such as the eBird project (Sullivan et al., 2009).

We start with a high-level overview of our approach. Suppose we wish to teach the crowd to label a large set of images (e.g., distinguishing butterflies from moths). How can this be done without already having access to the labels, or a set of informative features, for all the images (in which case crowdsourcing would be useless)? We suppose we have ground truth labels only for a small "teaching set" of examples. Our premise is that if we can teach a worker to classify this teaching set well, she can generalize to new images. In our approach, we first elicit—on the teaching set—a set of candidate features as well as a collection of hypotheses (e.g., linear classifiers) that the crowd may be using. We will describe the concrete procedure used in our experimental setup in Section 5. Having access to this information we use a teaching algorithm to select training examples and steer the learner towards the target hypothesis.

Classical work on teaching classifiers (reviewed in Section 2.2), assumes that learners are *noise-free*: Hypotheses are immediately eliminated from consideration upon ob-

Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 2014. JMLR: W&CP volume 32.

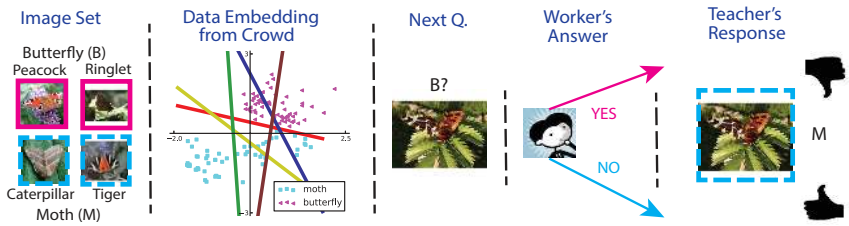


Figure 1. Illustration of crowd-teaching. Given a large set of images, the teacher (randomly) picks a small "teaching set". For this set, expert labels, as well as candidate features and hypotheses used by the crowd are elicited (see Section 5). The teacher then uses this information to teach the rest of the crowd to label the rest of the data, for which no features or labels are available. The teacher sequentially provides an unlabeled example from the teaching set to the worker, who attempts an answer. Upon receipt of the correct label, the learner may update her hypothesis before the next example is shown.

servation of an inconsistent training example. As we see in our experiments (Section 6), such approaches can be brittle. In contrast, we propose a *noise-tolerant* stochastic model of the learners, capturing our assumptions on how they incorporate training examples. We then (Section 4) propose STRICT (*Submodular Teaching for cRowdsourcIng ClassificaTion*), a novel teaching algorithm that selects a sequence of training examples to the workers in order to steer them towards the true hypothesis. We theoretically analyze our approach, proving strong approximation guarantees and teaching complexity results. Lastly, we demonstrate the effectiveness of our model and STRICT policy on three real image annotation tasks, carried out on the MTurk platform.

2. Background and Teaching Process

We now describe our learning domain and teaching protocol. As a running example, we consider the task of teaching to classify images, e.g., to distinguish butterflies from moths (see Figure 1).

2.1. The domain and the teaching protocol

Let $\mathcal{X}$ denote a set of examples (e.g., images), called the *teaching set*. We use (x, y) to denote a labeled example where $x \in \mathcal{X}$ and $y \in \{-1, 1\}$. We denote by $\mathcal{H}$ a finite class of hypotheses. Each element of $\mathcal{H}$ is a function $h : \mathcal{X} \mapsto \mathbb{R}$. The label assigned to x by hypothesis h is $\text{sgn}(h(x))$. The magnitude $|h(x)|$ indicates the confidence hypothesis h has in the label of x. For now, let us assume that $\mathcal{X}$ and $\mathcal{H}$ are known to both the teacher and the learner. In our image classification example, each image may be given by a feature vector x, and each hypothesis $h(x) = w_h^T x$ could be a linear function. In Section 5, we discuss the concrete hypothesis spaces used in our crowdsourcing tasks, and how we can elicit them from the crowd.

The teacher has access to the labels $y(x)$ of all the examples x in $\mathcal{X}$. We consider the *realizable* setting where $\mathcal{H}$ contains a hypothesis h^* (known to the teacher, but not the learner) for which $\text{sgn}(h^*(x)) = y(x)$ for all $x \in \mathcal{X}$. The goal of the teacher is to teach the correct hypothesis h^* to the learner. The basic assumption behind our approach is that if we can teach the workers to classify $\mathcal{X}$ correctly, then they will be able to generalize to new examples drawn from the same distribution as $\mathcal{X}$ (for which we neither have ground truth labels nor features). We will verify this assumption experimentally in Section 6. In the following, we review existing approaches to teaching classifiers, and then present our novel teaching method.

2.2. Existing teaching models

In existing methods, a broad separation can be made about assumptions that learners use to process training examples. *Noise-free* models assume learners immediately discard hypotheses inconsistent with observed examples. As our experiments in Section 6 show, such models can be brittle in practice. In contrast, *noise-tolerant* models make less strict assumptions on how workers treat inconsistent hypotheses.

Noise-free teaching: In their seminal work, Goldman & Kearns (1992) consider the *non-interactive* model: The teacher reveals a sequence of labeled examples, and the learner discards any inconsistent hypotheses (i.e., for which $h(x) \neq y$ for any example (x, y) shown). For a given hypothesis class, the *Teaching Dimension* is the smallest number of examples required to ensure that all inconsistent hypotheses are eliminated. More recent work (Balbach & Zeugmann, 2009; Zilles et al., 2011; Doliwa et al., 2010; Du & Ling, 2011) consider models of *interactive* teaching, where the teacher, after showing each example, obtains feedback about the hypothesis that the learner is currently entertaining. Such feedback can be used to select future teaching examples in a more informed way. While theoretically intriguing, in this paper we focus on *non-interactive* models, which are typically easier to deploy in practice.

Noise-tolerant teaching: In contrast to the noise-free setting, the practically extremely important *noise-tolerant* setting is theoretically much less understood. Very recently, Zhu (2013) investigates the optimization problem of generating a set of teaching examples that trades off between the expected future error of the learner and the "effort" (i.e., number of examples) taken by the teacher, in the special case when the prior of the learner falls into the exponential family, and the learner performs Bayesian inference. Their algorithmic approach does not apply to the problem addressed in this paper. Further, the approach

is based on heuristically rounding the solution of a convex program, with no bounds on the integrality gap.

Basu & Christensen (2013) study a similar problem of teaching workers to classify images. The authors empirically investigate a variety of heuristic teaching policies on a set of human subjects for a synthetically generated data set. Lindsey et al. (2013) propose a method for evaluating and optimizing over parametrized policies with different orderings of positive and negative examples. None of these approaches offer theoretical performance guarantees of the kind provided in this paper.

3. Model of the Learner

We now introduce our model of the learner, by formalizing our assumptions about how she adapts her hypothesis based on the training examples she receives from the teacher. Generally, we assume that the learner is not aware that she is being taught. We assume that she carries out a random walk in the hypothesis space $\mathcal{H}$: She starts at some hypothesis, stays there as long as the training examples received are consistent with it, and randomly jumps to an alternative hypothesis upon an observed inconsistency. Hereby, preference will be given to hypotheses that better agree with the received training.

More formally, we model the learner via a stochastic process, in particular a (non-stationary) Markov chain. Before the first example, the learner randomly chooses a hypothesis h_1, drawn from a prior distribution P_0. Then, in every round t there are two possibilities: If the example (x_t, y_t) received agrees with the label implied by the learner's current hypothesis (i.e., $\operatorname{sgn}(h_t(x_t)) = y_t$), she sticks to it: $h_{t+1} = h_t$. On the other hand, if the label y_t disagrees with the learner's prediction $\operatorname{sgn}(h_t(x_t))$, she draws a new hypothesis h_{t+1} based on a distribution P_t constructed in a way that reduces the probability of hypotheses that disagreed with the true labels in the previous steps:

$$P_t(h) = \frac{1}{Z_t} P_0(h) \prod_{\substack{s=1 \\ y_s \neq \operatorname{sgn}(h(x_s))}}^{t} P(y_s|h, x_s) \qquad (1)$$

with normalization factor

$$Z_t = \sum_{h \in \mathcal{H}} P_0(h) \prod_{\substack{s=1 \\ y_s \neq \operatorname{sgn}(h(x_s))}}^{t} P(y_s|h, x_s).$$

In Equation (1), for some $\alpha > 0$, the term

$$P(y_s|h, x_s) = \frac{1}{1 + \exp(-\alpha h(x_s) y_s)}$$

models a likelihood function, encoding the confidence that hypothesis h places in example x_s. Thus, if the example (x_s, y_s) is "strongly inconsistent" with h (i.e., $h(x_s)y_s$ takes a large negative value and consequently $P(y_s \mid h, x_s)$ is very small), then the learner will be very unlikely to jump to hypothesis h. The scaling parameter α allows to control the effect of observing inconsistent examples. The limit $\alpha \to \infty$ results in a behavior where inconsistent hypotheses are completely removed from consideration. This case precisely coincides with the *noise-free* learner models classically considered in the literature (Goldman & Kearns, 1992).

It can be shown (see Lemma 1 in the supplementary material), that the marginal probability that the learner implements some hypothesis h in step t is equal to $P_t(h)$, even when the true label and the predicted label agreed in the previous step.

4. Teaching Algorithm

Given the learner's prior over the hypotheses $P_0(h)$, how should the teacher choose examples to help the learner narrow down her belief to accurate hypotheses? By carefully showing examples, the teacher can control the learner's progress by steering her posterior towards h^*.

With a slight abuse of notation, if the teacher showed the set of examples $A = \{x_1, \ldots, x_t\}$ we denote the posterior distribution by $P_t(\cdot)$ and $P(\cdot|A)$ interchangeably. We use the latter notation when we want to emphasize that the examples shown are the elements of A. With the new notation, we can write the learner's posterior after showing A as

$$P(h|A) = \frac{1}{Z(A)} P_0(h) \prod_{\substack{x \in A \\ y(x) \neq \operatorname{sgn}(h(x))}} P(y(x)|h, x)\,.$$

The ultimate goal of the teacher is to steer the learner towards a distribution with which she makes few mistakes. The expected error-rate of the learner after seeing examples $A = \{x_1, \ldots, x_t\}$ together with their labels $y_i = \operatorname{sgn}(h^*(x_i))$ can be expressed as

$$\mathbb{E}[\mathrm{err}_L \mid A] = \sum_{h \in \mathcal{H}} P(h|A)\, \mathrm{err}(h, h^*)\,, \text{where}$$

$$\mathrm{err}(h, h^*) = \frac{|\{x \in \mathcal{X} : \operatorname{sgn}(h(x)) \neq \operatorname{sgn}(h^*(x))\}|}{|\mathcal{X}|}$$

is the fraction of examples x from the teaching set $\mathcal{X}$ on which h and h^* disagree about the label. We use the notation $\mathbb{E}[\mathrm{err}_L] = \mathbb{E}[\mathrm{err}_L \mid \{\}]$ as shorthand to refer to the learner's error before receiving training.

Given an allowed tolerance ϵ for the learner's error, a natural objective for the teacher is to find the smallest set of examples A^* achieving this error, i.e.:

$$A^*_\varepsilon = \arg\min_{A \subseteq \mathcal{X}} |A| \text{ s.t. } \mathbb{E}[\mathrm{err}_L \mid A] \leq \epsilon. \qquad (2)$$

We will use the notation $\mathrm{OPT}(\epsilon) = |A^*_\epsilon|$ to refer to the size of the optimal solution achieving error ϵ. Unfortunately, Problem (2) is a difficult combinatorial optimization problem. The following proposition, proved in the supplement, establishes hardness via a reduction from set cover.

Policy 1 Teaching Policy STRICT

Input: examples $\mathcal{X}$, hyp. $\mathcal{H}$, prior P_0, error ϵ.
Output: teaching set A
$A \leftarrow \emptyset$
while $F(A) < \mathbb{E}[\mathrm{err}_L] - P_0(h^*)\epsilon$ **do**
  $x \leftarrow \arg\max_{x \in \mathcal{X}} (F(A \cup \{x\}))$
  $A \leftarrow A \cup \{x\}$
end while

Proposition 1. *Problem (2) is NP-hard.*

Given this hardness, in the following, we introduce an efficient approximation algorithm for Problem (2).

The first observation is that, in order to solve Problem (2), we can look at the objective function

$$R(A) = \mathbb{E}[\mathrm{err}_L] - \mathbb{E}[\mathrm{err}_L \mid A] = \sum_{h \in \mathcal{H}} (P_0(h) - P(h|A)) \, \mathrm{err}(h, h^*) \,,$$

quantifying the expected *reduction in error* upon teaching A. Solving Problem (2) is equivalent to finding the smallest set A achieving error reduction $\mathbb{E}[\mathrm{err}_L] - \epsilon$. Thus, if we could, for each k, find a set A of size k maximizing $R(A)$, we could solve Problem (2), contradicting the hardness.

The key idea is to replace the objective $R(A)$ with the following surrogate function:

$$F(A) = \sum_{h \in \mathcal{H}} (Q(h) - Q(h|A)) \, \mathrm{err}(h, h^*) \,, \text{where}$$

$$Q(h|A) = P_0(h) \prod_{\substack{x \in A \\ y(x) \neq \mathrm{sgn}(h(x))}} P(y(x)|h, x)$$

is the *unnormalized posterior* of the learner. As shown in the supplementary material, this surrogate objective function satisfies *submodularity*, a natural diminishing returns condition. Submodular functions can be effectively optimized using a greedy algorithm, which, at every iteration, adds the example that maximally increases the surrogate function F (Nemhauser et al., 1978). We will show that maximizing $F(A)$ gives us good results in terms of the original, normalized objective function $R(A)$, that is, the expected error reduction of the learner. In fact, we show that running the algorithm until $F(A) \geq \mathbb{E}[\mathrm{err}_L] - P_0(h^*)\epsilon$ is sufficient to produce a feasible solution to Problem (2), providing a natural stopping condition. We call the greedy algorithm for $F(A)$ STRICT, and describe it in Policy 1.

Note that in the limit $\alpha \to \infty$, $F(A)$ quantifies the prior mass of all hypotheses h (weighted by $\mathrm{err}(h, h^*)$) that are inconsistent with the examples A. Thus, in this case, $F(A)$ is simply a weighted coverage function, consistent with classical work in *noise-free* teaching (Goldman & Kearns, 1992).

4.1. Approximation Guarantees

The following theorem ensures that if we choose the examples in a greedy manner to maximize our surrogate objective function $F(A)$, as done by Policy 1, we are close to being optimal in some sense.

Theorem 1. *Fix $\epsilon > 0$. The* STRICT *Policy 1 terminates after at most $\mathrm{OPT}(P_0(h^*)\epsilon/2) \log \frac{1}{P_0(h^*)\epsilon}$ steps with a set A such that $\mathbb{E}[\mathrm{err}_L \mid A] \leq \epsilon$.*

Thus, informally, Policy 1 uses a near-minimal number of examples when compared to *any* policy achieving $O(\epsilon)$ error (viewing $P_0(h^*)$ as a constant).

The main idea behind the proof of this theorem is that we first observe that $F(A)$ is submodular and thus the greedy algorithm gives a set reasonably close to F's optimum. Then we analyze the connection between maximizing $F(A)$ and minimizing the expected error of the learner, $\mathbb{E}[\mathrm{err}_L \mid A]$. A detailed proof can be found in the supplementary material.

Note that maximizing $F(A)$ is not only sufficient, but also *necessary* to achieve ϵ precision. Indeed, it is immediate that $P(h|A) \geq Q(h|A)$, which in turn leads to

$$\mathbb{E}[\mathrm{err}_L \,|A] = \sum_{h \in \mathcal{H}} P(h|A) \, \mathrm{err}(h, h^*) \geq \sum_{h \in \mathcal{H}} Q(h|A) \, \mathrm{err}(h, h^*) = \mathbb{E}[\mathrm{err}_L] - F(A) \,.$$

Thus, if $\mathbb{E}[\mathrm{err}_L] - F(A) > \epsilon$, then the expected posterior error $\mathbb{E}[\mathrm{err}_L \mid A]$ of the learner is also greater than ϵ.

4.2. Teaching Complexity for Linear Separators

Theorem 1 shows that greedily optimizing $F(A)$ leads to low error with a number of examples not far away from the optimal. Now we show that, under some additional assumptions, the optimal number of examples is not too large.

We consider the important case where the set of hypotheses $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$ consists of linear separators $h(x) = w_h^T x + b_h$ for some weight vector $w_h \in \mathbb{R}^d$ and offset $b_h \in \mathbb{R}$. The label predicted by h for example x is $\mathrm{sgn}(w_h^T x + b_h)$.

We introduce an additional assumption, namely λ-richness of $(\mathcal{X}, \mathcal{H})$. First notice that $\mathcal{H}$ partitions $\mathcal{X}$ into polytopes (intersections of half-spaces), where within one polytope, all examples are labeled the same by every hypothesis, that is, within a polytope $\mathcal{P}$, for every $x, x' \in \mathcal{P} \subseteq \mathbb{R}^d$ and $h \in \mathcal{H}$, $\mathrm{sgn}(h(x)) = \mathrm{sgn}(h(x'))$. We say that $\mathcal{X}$ is λ-rich if any $\mathcal{P}$ contains at least λ examples. In other words, if the teacher needs to show (up to) λ distinct examples to the learner from the same polytope in order to reduce her error below some level, this can be done.

Theorem 2. *Fix $\varepsilon > 0$. Suppose that the hypotheses are hyperplanes in $\mathbb{R}^d$ and that $(\mathcal{X}, \mathcal{H})$ is $(8 \log^2 \frac{2}{\epsilon})$-rich. Then the* STRICT *policy achieves learner error less than ϵ after at most $m = 8 \log^2 \frac{2}{\epsilon}$ teaching examples.*

Figure 2. Sample images of all the three data sets used for the experiments.

The proof of this theorem is in the supplementary material. In a nutshell, the proof works by establishing the existence (via the probabilistic method) of a teaching policy for which the number of examples needed can be bounded – hence also bounding the optimal policy – and then using Theorem 1.

5. Experimental Setup

In our experiments, we consider three different image classification tasks: i) classification of synthetic insect images into two hypothetical species *Vespula and Weevil* (*VW*); ii) distinguishing *butterflies and moths* on real images (*BM*); and iii) identification of birds belonging to an *endangered species of woodpeckers* from real images (*WP*). Our teaching process requires a known feature space for image dataset $\mathcal{X}$ (*i.e.* the teaching set of images) and a hypothesis class $\mathcal{H}$. While $\mathcal{X}$ and $\mathcal{H}$ can be controlled by design for the synthetic images, we illustrate different ways on how to automatically obtain a crowd-based embedding of the data for real images. We now discuss in detail the experimental setup of obtaining $\mathcal{X}$ with its feature space and $\mathcal{H}$ for the three different data sets used in our classification tasks.

5.1. Vespula vs. Weevil

We first generate a classification problem using synthetic images $\mathcal{X}$ in order to allow controlled experimentation. As a crucial advantage, in this setting the hypothesis class $\mathcal{H}$ is known by design, and the task difficulty can be controlled. Furthermore, this setting ensures that workers have no prior knowledge of the image categories.

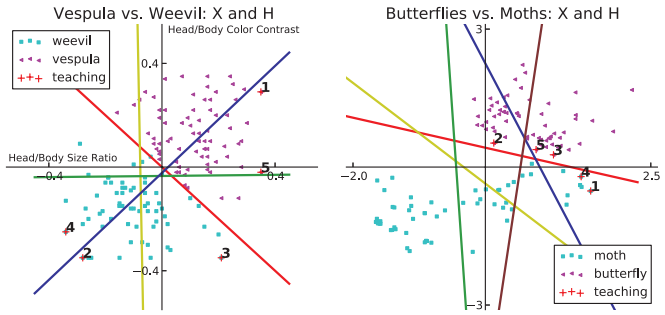
Dataset $\mathcal{X}$ and feature space: We generated synthetic images of insects belonging to two hypothetical species: *Weevil* and *Vespula*. The task is to classify whether a given image contains a Vespula or not. The images were generated by varying body size and color as well as head size and color. A given image x_i can be distinguished based on the following two-dimensional feature vector $x_i = [x_{i,1} = f_1, x_{i,2} = f_2]$ – i) f_1: the head/body size ratio, ii) f_2: head/body color contrast. Fig. 3(a) shows the embedding of this data set in a two-dimensional space based on these two features. Fig. 2 shows sample images of the two species and illustrates that Weevils have short heads with color similar to their body, whereas Vespula are distinguished by their big and contrasting heads. A total of 80 images per species were generated by sampling the features f_1 and f_2 from two bivariate Gaussian distributions: $(\mu = [0.10, 0.13], \Sigma = [0.12, 0; 0, 0.12])$ for Vespula and $(\mu = [-0.10, -0.13], \Sigma = [0.12, 0; 0, 0.12])$ for Weevil. A separate test set of 20 images per species were generated as well, for evaluating learning performance.

Hypothesis class $\mathcal{H}$: As we know the exact feature space of $\mathcal{X}$, we can use any parametrized class of functions $\mathcal{H}$ on $\mathcal{X}$. In our experiments, we use a class of linear functions for $\mathcal{H}$, and further restrict $\mathcal{H}$ to eight clusters of hypotheses, centered at the origin and rotated by $\pi/4$ from each other. Specifically, we sampled the parameters of the linear hypotheses from the following multivariate Gaussian distribution: $(\mu_i = [\pi/4 \cdot i, 0], \Sigma_i = [2, 0; 0, 0.005])$, where i varies from 0 to 7. Each hypothesis captures a different set of cues about the features that workers could reasonably have: i) ignoring a feature, ii) using it as a positive signal for Vespula, and iii) using it as a negative signal for Vespula. Amongst the generated hypotheses, we picked the target hypothesis h^* as the one with minimal error on teaching set $\mathcal{X}$. In order to ensure realizability, we then removed any data points $x \in \mathcal{X}$ where $\text{sgn}(h^*(x)) \neq y(x)$. Fig. 3(a) shows a subset of four of these hypotheses, with the target hypothesis h^* represented in red. The prior distribution P_0 is chosen as uniform.

5.2. Butterflies vs. Moths

Dataset images $\mathcal{X}$: As our second dataset, we used a collection of 200 real images of four species of butterflies and moths from publicly available images (Imagenet) : i) *Peacock Butterfly*, ii) *Ringlet Butterfly*, iii) *Caterpillar Moth*, iv) *Tiger Moth*, as shown in Fig. 2. The task is to classify whether a given image contains a butterfly or not. While Peacock Butterfly and Caterpillar Moth are clearly distinguishable as butterflies and moths, Tiger Moth and Ringlet Butterfly are often considered hard to classify correctly. We used 160 of these images (40 per sub-species) as teaching set $\mathcal{X}$ and the remaining 40 (10 per sub-species) for testing.

Crowd-embedding of $\mathcal{X}$: A Euclidean embedding of $\mathcal{X}$ for this image set is not readily available. Human-perceptible features for such real images may be difficult to compute. In fact, this challenge is one major motivation for using crowdsourcing in image annotation. However, several techniques do exist that allow estimating such an embedding from a small set of images and a limited number of crowd labels. In particular, we used the approach of Welinder et al. (2010) as a preprocessing step. Welinder et al. propose a generative Bayesian model for the annotation process of the images by the workers and then use an inference algorithm to jointly estimate a low-dimensional embedding of the data, as well as a collection of linear hypotheses – one for each annotator – that best explain their provided labels. We requested binary labels (of whether the image contains a butterfly) for



(a) $\mathcal{X}$ and $\mathcal{H}$ for VW (b) $\mathcal{X}$ and $\mathcal{H}$ for BM

Features	% feature presence in species			h*
	Red-cockaded	Red-bellied	Downy	
has_breast_pattern: *solid*	22	88	90	-1
has_upper_tail_color: *white*	55	91	53	0
has_bill_length: *same_as_head*	78	93	25	0
has_bill_length: *shorter_than_head*	23	7	76	0
has_forehead_color: *black*	91	0	96	0
has_forehead_color: *red*	0	78	0	0
has_nape_color: *black*	85	0	93	0
has_nape_color: *red*	0	87	15	0
has_back_pattern: *spotted*	75	27	29	0
has_back_pattern: *striped*	18	69	7	0
has_belly_pattern: *solid*	25	81	94	-1
has_crown_color: *black*	98	0	96	0
has_crown_color: *red*	2	86	36	-1

(c) Features for WP

Figure 3. (a) shows the 2-D embedding of synthetic images for *Weevil* and *Vespula* for the features: head/body size proportion (f_1) and head/body color contrast (f_2), normalized around origin. It shows four of the hypotheses in $\mathcal{H}$, with the target hypothesis h^* in red. (b) shows the 2-D embedding of images for the *Moth* and *Butterfly* data set, and the hypothesis for a small set of workers, as obtained using the approach of Welinder et al. (2010). (c) shows the 13 features used for representation of woodpecker images and the w_{h^*} vector of the target hypotheses. It also lists the average number of times a particular feature is present in the images of a given species.

our teaching set $\mathcal{X}$, $|\mathcal{X}| = 160$, from a set of 60 workers. By using the software CUBAM, implementing the approach of Welinder et al., we inferred a 2-D embedding of the data, as well as linear hypotheses corresponding to each of the 60 workers who provided the labels. Fig. 3(b) shows this embedding of the data, as well as a small subset of workers' hypotheses as colored lines.

Hypothesis class $\mathcal{H}$: The 60 hypothesis obtained through the crowd-embedding provide a prior distribution over linear hypotheses that the workers in the crowd may have been using. Note that these hypotheses capture various idiosyncrasies (termed "schools of thought" by Welinder et al.) in the workers' annotation behavior – i.e., some workers were more likely to classify certain moths as butterflies and vice versa. To create our hypothesis class $\mathcal{H}$, we randomly sampled 15 hypotheses from these. Additionally, we fitted a linear classifier that best separates the classes and used it as target hypothesis h^*, shown in red in Fig. 3(b). The few examples in $\mathcal{X}$ that disagreed with h^* were removed from our teaching set, to ensure realizability.

Teaching the rest of the crowd: The teacher then uses this embedding and hypotheses in order to teach the rest of the crowd. We emphasize that – crucially – the embedding is *not* required for test images. Neither the workers nor the system used any information about sub-species in the images.

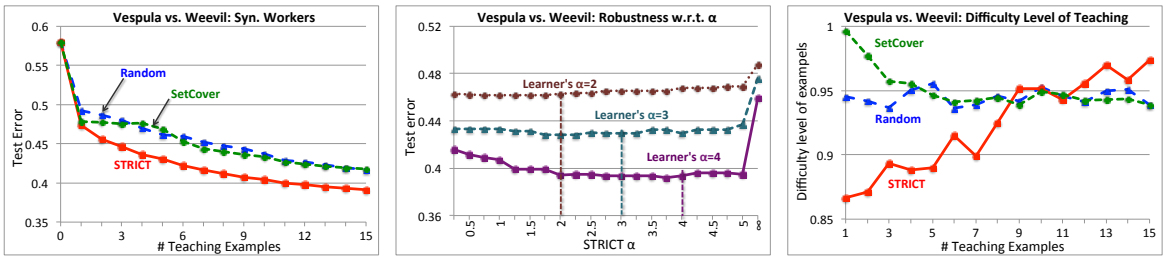
5.3. Endangered Woodpecker Bird Species

Dataset images $\mathcal{X}$: Our third classification task is inspired from the eBird citizen science project (Sullivan et al., 2009) and the goal of this task is to identify birds belonging to an endangered species of woodpeckers. We used a collection of 150 real images belonging to three species of woodpeckers from a publicly available dataset (Wah et al., 2011), with one endangered species: i) *Red-cockaded* woodpecker and other two species belonging to the least-concerned category: ii) *Red-bellied woodpecker*, iii) *Downy woodpecker*. On this dataset, the task is to classify whether a given image contains a red-cockaded woodpecker or not. We used 80 of these images (40 per red-cockaded, and 20 each per the other two species of the least-concerned categories) for teaching (i.e., dataset $\mathcal{X}$). We also created a testing set of 20 images (10 for red-cockaded, and 5 each for the other two species).

Crowd-embedding of $\mathcal{X}$: We need to infer an embedding and hypothesis space of the teaching set for our teaching process. While an approach similar to the one used for the BM task is applicable here as well, we considered an alternate option of using metadata associated with these images, elicited from the crowd, as further explained below.

Each image in this dataset is annotated with 312 binary attributes, for example, *has_forehead_color:black*, or *has_bill_length:same_as_head*, through workers on MTurk. The features can take values {+1, -1, 0} indicating the presence or absence of an attribute, or uncertainty (when the annotator is not sure or the answer cannot be inferred from the image given). Hence, this gives us an embedding of the data in $\mathbb{R}^{312}$. To further reduce the dimensionality of the feature space, we pruned the features which are not informative enough for the woodpecker species. We considered all the species of woodpeckers present in the dataset (total of 6), simply computed the average number of times a given species is associated positively with a feature, and then looked for features with maximal variance among the various species. By applying a simple cutoff of 60 on the variance, we picked the top $d = 13$ features as shown in Fig 3(c), also listing the average number of times the feature is associated positively with the three species.

Hypothesis class $\mathcal{H}$: We considered a simple set of linear hypotheses $h(x) = w^T x$ for $w \in \{+1, 0, -1\}^d$, which place a weight of {+1, 0, -1} on any given feature and passing through the origin. The intuition behind these simple hypotheses is to capture the cues that workers could possibly use or learn for different features: ignoring a feature (0),



(a) Data set VW: Test Error (b) Data set VW: Robustness w.r.t α (c) Data set VW: Difficulty Level

Figure 4. (a) compares the algorithms' teaching performance in terms of simulated workers' test error (VW task). (b) shows the robustness of STRICT w.r.t. unknown α parameters of the learners. Thus, a noise-tolerant teacher (i.e., $\alpha < \infty$) performs much better than noise-free *SetCover* teaching, even with misspecified α. (c) shows how the difficulty of STRICT's examples naturally increase during teaching.

using it as a positive signal ($+1$), and using it as a negative signal (-1). Another set of simple hypotheses that we explored are conjunctions and disjunctions of these features that can be created by setting the appropriate offset factor b_h (Anthony et al., 1992). Assuming that workers focus only on a small set of features, we considered sparse hypotheses with non-zero weight on only a small set of features. To obtain the target hypothesis, we enumerated all possible hypotheses that have non-zero weight for at most three features. We then picked as h^* the hypothesis with minimal error on $\mathcal{X}$ (shown in Fig 3(c)). Again, we pruned the few examples in $\mathcal{X}$ which disagreed with h^* to ensure realizability. As hypothesis class $\mathcal{H}$, we considered all hypotheses with a non-zero weight for at most two features along with the target h^*, resulting in a hypothesis class of size 339.

Teaching the rest of the crowd: Given this embedding and hypothesis class, the teacher then uses the same approach as in the two previous datasets to teach the rest of the crowd. Importantly, this embedding is *not* required for test images.

6. Experimental Results

Now we present our experimental results, consisting of simulations and actual annotation tasks on MTurk.

Metrics and baselines: Our primary performance metric is the test error (avg. classification error of the learners), of simulated or MTurk workers on a hold-out test data set. We compare STRICT against two baseline teachers: *Random* (picking uniformly random examples), and *SetCover* (the classical noise-free teaching model introduced in Section 3).

6.1. Results on Simulated Learners

We start with simulated learners and report results only on the VW dataset here for brevity. The simulations allow us to control the problem (parameters of the learner, size of the hypothesis space, *etc.*), and hence gain more insight into the teaching process. Additionally, we can observe how robust our teaching algorithm is against misspecified parameters.

Test error. We simulated 100 learners with varying α parameters chosen randomly from the set $\{2, 3, 4\}$ and different initial hypotheses of the learners, sampled from $\mathcal{H}$. We varied the experimental setting by changing the size of the hypothesis space and the α value used by STRICT. Fig. 4(a) reports results with $\alpha = 2$ for STRICT and size of hypothesis class 96 (2 hypotheses per each of the eight clusters, described in Section 5 for the VW dataset.

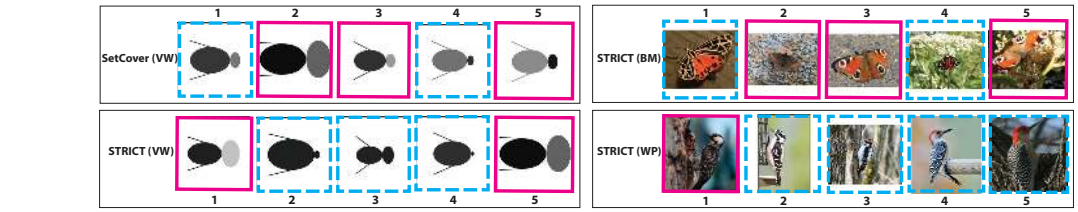
How robust is STRICT for a mismatched α? In real-world annotation tasks, the learner's α parameter is not known. In this experiment, we vary the α values used by the teaching algorithm STRICT against three learners with values of $\alpha = 1, 2$ and 3. Fig. 4(b) shows that a conservative teacher using α bounded in the range 1 to 5 performs as good as the one knowing the true α value.

On the difficulty level of teaching. Fig. 4(c) shows the difficulty of examples picked by different algorithms during the process of teaching, where difficulty is measured in terms of expected uncertainty (entropy) that a learner would face for the shown example, assuming that the expectation is taken w.r.t. the learners current posterior distribution over the hypotheses. *SetCover* starts with difficult examples assuming that the learner is perfect. STRICT starts with easy examples, followed by more difficult ones, as also illustrated in the experiments in Fig. 5(a). Recent results of Basu & Christensen (2013) show that such curriculum-based learning (where the difficulty level of teaching increases with time) indeed is a useful teaching mechanism. Note that our teaching process inherently incorporates this behavior, without requiring explicit heuristic choices. Also, the transition of *SetCover* to easier examples is just an artifact as *SetCover* randomly starts selecting examples once it (incorrectly) infers that the learner has adopted the target hypothesis. The difficulty can be easily seen when comparing the examples picked by *SetCover* and STRICT in Fig. 5(a).

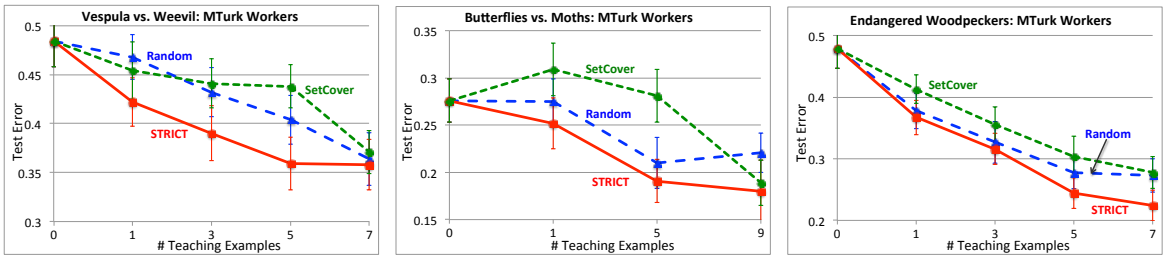
6.2. Results on MTurk Workers

Next, we measure the performance of our algorithms when deployed on the actual MTurk platform.

Generating the teaching sequence. We generate sequences of teaching examples for STRICT, as well as *Random* and *SetCover*. We used the feature spaces $\mathcal{X}$ and hypothesis spaces $\mathcal{H}$ as explained in Section 5. We chose $\alpha = 2$ for our algorithm STRICT. To better understand the execution of the algorithms, we illustrate the exam-



(a) Examples in teaching sequence



(b) Test error on VW dataset (c) Test error on BM dataset (d) Test error on WP dataset

Figure 5. (a) shows the order of examples picked by the teaching algorithms. For the VW and BM tasks, we embed the examples in the 2D-feature space in Figs. 3(a) and 3(b). (b-d) show the teaching performance of our algorithm measured in terms of test error of humans learners (MTurk workers) on hold out data. STRICT is compared against *SetCover* and *Random* teaching, as we vary the length of teaching.

ples picked by our algorithm as part of teaching, shown in Fig. 5(a). We further show these examples in the 2-D embedding for the VW and BM datasets in Figs. 3(a) and 3(b).

Workers on MTurk and the teaching task. We recruited workers from the MTurk platform by posting the tasks on the MTurk platform. Workers were split into different control groups, depending on the algorithm and the length of teaching used (each control group corresponds to a point in the plots of Fig. 5). Fig. 1 provides a high level overview of how the teaching algorithm interacted with the worker. Teaching is followed by a phase of testing examples without providing feedback, for which we report the classification error. For the VW dataset, a total of 780 workers participated (60 workers per control group). For BM, a total of 300 workers participated, and 520 participated in the WP task. The length of the teaching phase was varied as shown in Fig. 5. The test phase was set to 10 examples for the VW and BM tasks, and 16 examples for the WP task. The workers were given a fixed payment for participation and completion, additionally a bonus payment was reserved for the top 10% performing workers within each control group.

Does teaching help? Considering the worker's test set classification performance in Fig. 5, we can consistently see an accuracy improvement as workers classify unseen images. This aligns with the results from simulated learners and shows that teaching is indeed helpful in practice. Furthermore, the improvement is monotonic w.r.t. the length of teaching phase used by STRICT. In order to understand the significance of these results, we carried out Welch's t-test comparing the workers who received teaching by STRICT to the control group of workers without any teaching. The hypothesis that STRICT significantly improves the classification accuracy has two-tailed p-values of $p < 0.001$ for VW and WP tasks, and $p = 0.01$ for the BM task.

Does our teaching algorithm outperform baselines? Fig. 5 demonstrates that our algorithm STRICT outperforms both *Random* and *SetCover* teaching qualitatively in all studies. We check the significance by performing a paired-t test, by computing the average performance of the workers in a given control group and pairing the control groups with same length of teaching for a given task. For the VW task, STRICT is significantly better than *SetCover* and *Random* (at $p = 0.05$ and $p = 0.05$). For WP, STRICT is significantly better than SetCover ($p = 0.002$) whereas comparing with *Random*, the p-value is $p = 0.07$.

7. Conclusions

We proposed a noise-tolerant stochastic model of the workers' learning process in crowdsourcing classification tasks. We then developed a novel teaching algorithm STRICT that exploits this model to teach the workers efficiently. Our model generalizes existing models of teaching in order to increase robustness. We proved strong theoretical approximation guarantees on the convergence to a desired error rate. Our extensive experiments on simulated workers as well as on three real annotation tasks on the Mechanical Turk platform demonstrate the effectiveness of our teaching approach. More generally, our approach goes beyond solving the problem of teaching workers in crowdsourcing services. With the recent growth of online education and tutoring systems (e.g., Coursera), algorithms such as STRICT can be envisioned to aid in supporting data-driven online education (Weld et al., 2012; Dow et al., 2013).

Acknowledgments. This research was supported in part by SNSF grant 200021_137971, ERC StG 307036 and a Microsoft Research Faculty Fellowship. Ilija Bogunovic is partly supported by the European Commission under Grant MIRG-268398.

References

Anthony, Martin, Brightwell, Graham, Cohen, Dave, and Shawe-Taylor, John. On exact specification by examples. In *Proc. of COLT*, pp. 311–318. ACM, 1992.

Balbach, F. J. and Zeugmann, T. Recent developments in algorithmic teaching. In *Proceedings of the 3rd International Conference on Language and Automata Theory and Applications*, pp. 1–18, 2009.

Basu, S. and Christensen, J. Teaching classification boundaries to humans. In *Proc. of AAAI*, 2013.

Coursera. https://www.coursera.org/.

CUBAM. Caltech UCSD binary annotation model. https://github.com/welinder/cubam.

Dalvi, N., Dasgupta, A., Kumar, R., and Rastogi, V. Aggregating crowdsourced binary ratings. In *Proc. of WWW*, pp. 285–294, 2013.

Doliwa, T., Simon, H. U., and Zilles, S. Recursive teaching dimension, learning complexity, and maximum classes. In *Proc. of ALT*, pp. 209–223, 2010.

Dow, S., Gerber, E., and Wong, A. A pilot study of using crowds in the classroom. In *Proceedings of Human Factors in Computing Systems (CHI)*, 2013.

Du, Jun and Ling, Charles X. Active teaching for inductive learners. In *Proceedings of the Eleventh SIAM International Conference on Data Mining*, pp. 851–861, 2011.

Goldman, S. A. and Kearns, M. J. On the complexity of teaching. *Journal of Computer and System Sciences*, 50: 303–314, 1992.

Gomes, R., Welinder, P., Krause, A., and Perona, P. Crowdclustering. In *Proc. of NIPS*, 2011.

Imagenet. http://www.image-net.org/.

Lindsey, Robert, Mozer, Michael, Huggins, William J, and Pashler, Harold. Optimizing instructional policies. In *Proc. of NIPS*, pp. 2778–2786, 2013.

Lintott, C. J., Schawinski, K., Slosar, A., Land, K., Bamford, S., Thomas, D., Raddick, M. J., Nichol, R. C., Szalay, A., Andreescu, D., Murray, P., and Vandenberg, J. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. *Monthly Notices of the Royal Astronomical Society*, 389: 1179–1189, 2008.

MTurk. Mechanical Turk platform. https://www.mturk.com/.

Nemhauser, G.L., Wolsey, L.A., and Fisher, M. An analysis of the approximations for maximizing submodular set functions. *Mathematical Programming*, 14:265–294, 1978.

Snow, R., O'Connor, B., Jurafsky, D., and Ng, A. Y. Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 254–263, 2008.

Sorokin, A. and Forsyth, D. Utility data annotation with amazon mechanical turk. In *First IEEE Workshop on Internet Vision*, 2008.

Sullivan, Brian L, Wood, Christopher L, Iliff, Marshall J, Bonney, Rick E, Fink, Daniel, and Kelling, Steve. ebird: A citizen-based bird observation network in the biological sciences. *Biological Conservation*, 142(10): 2282–2292, 2009.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Weld, D. S, Adar, E., Chilton, L., Hoffmann, R., and Horvitz, E. Personalized online education a crowdsourcing challenge. *Workshop on Human Computation*, 2012.

Welinder, P., Branson, S., Belongie, S., and Perona, P. The multidimensional wisdom of crowds. In *Proc. of NIPS*, 2010.

Zhu, Xiaojin. Machine teaching for bayesian learners in the exponential family. In *Proc. of NIPS*, pp. 1905–1913, 2013.

Zilles, S., Lange, S., Holte, R., and Zinkevich, M. Models of cooperative teaching and learning. *Journal of Machine Learning Research*, 12:349–384, 2011.