Bidirectional Active Learning with Gold-Instance-Based Human Training

Feilong Tang
Department of Computer Science and Engineering, Shanghai Jiao Tong University, China
tang_feilong@sjtu.edu.cn

Abstract

Active learning was proposed to improve learning performance and reduce labeling cost. However, traditional relabeling-based schemes seriously limit the ability of active learning because human may repeatedly make similar mistakes, without improving their expertise. In this paper, we propose a *Bidirectional Active Learning with human Training* (BALT) model that can enhance human related expertise during labeling and improve relabeling quality accordingly. We quantitatively capture how gold instances can be used to both estimate labelers' previous performance and improve their future correctness ratio. Then, we propose the backward relabeling scheme that actively selects the most likely incorrectly labeled instances for relabeling. Experimental results on three real datasets demonstrate that our BALT algorithm significantly outperforms representative related proposals.

1 Introduction

Recently, active learning has been attracting increasing attention due to its ability to reduce labeling efforts and costs [Settles, 2010]. In traditional active learning, labelers are required to label only the most informative instances that predominantly determine classification performance. However, labelers often make mistakes repeatedly because they can not gain enough training [Golovin *et al.*, 2010; Harris, 2011].

Existing solutions to such the error-sensitivity problem can be divided into two categories. One is to estimate the label reliability or expertise level of labelers [Cakmak and Thomaz, 2014; Donmez *et al.*, 2009] and then *eliminate error-like answers*. However, filtering answers may remove useful information; filtering labelers is not feasible in scenarios with few labelers; and labelers may show varying performance in different tasks [Deng *et al.*, 2013]. Moreover, methods based on posterior distributions [Whitehill *et al.*, 2009] did not consider how to improve the ability of labelers during labeling.

Another category of approaches require labelers to relabel error-like labeled instances, called "*repeated labeling*" [Li et al., 2016; Ipeirotis *et al.*, 2014], which can improve learning performance to some extend [Lin and Weld, 2016; Zhang *et al.*, 2015]. In these approaches, however, labelers are asked to relabel their own previously labeled instances so that they may produce only slightly different labels even after many rounds of relabeling.

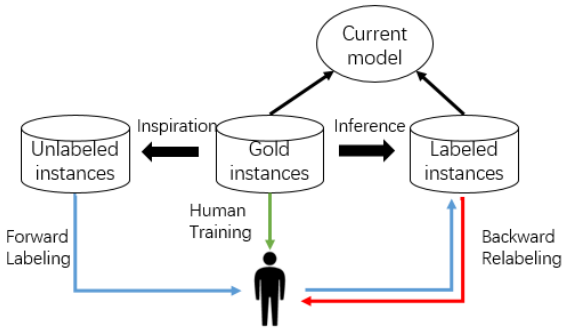


Figure 1: The framework of our BALT model.

One straightforward approach to improve labeling performance is enhancing labelers' expertise by training human with domain knowledge [Singla *et al.*, 2014]. Such methods have been widely used [Amir *et al.*, 2016] and usually follow the *training-before-working* scheme [Servajean *et al.*, 2016]. However, they are inefficient and possibly even undesirable because labelers are unable to distinguish key knowledge and may easily forget important information while working.

To solve this problem, we propose a novel *training-while-working* scheme in this paper, where labelers are adaptively trained with gold instances during labeling. Gold instances [Oleson *et al.*, 2011] are ones with known ground-truth labels or some explanations of why labels have been chosen. The most common approach to gold instance sampling is to randomly distribute gold instances among unlabeled instances [Li *et al.*, 2013] and reveal the result only after such an instance has been assigned a label. In this paper, we use this approach as both an indicator of performance and a training method for labelers; and propose a *Bidirectional Active Learning with human Training (BALT)* model. It consists of three processes: *forward labeling*, *human training* and *backward relabeling*, to enhance the ability of classifiers using gold instances, as shown in Figure 1. The main contributions of this paper can be summarized as follows.

- We propose a novel *training-while-working* scheme that adaptively determines gold instances to train labelers

during labeling. This scheme can improve labelers' expertise and future labeling performance.

- We quantitatively capture the characteristics of gold instances to model the labeling behavior of labelers. Based on an estimation of correctness probability, we propose effective method to compare the contributions of three types of sampling, and all of which can considerably reduce the number of potentially incorrect labels.
- We fuse active learning with human training and propose the *BALT* algorithm. Experimental results on three real datasets demonstrate that our BALT outperforms related proposals, both greatly reducing the proportion of falsely labeled instances and considerably improving the expertise of labelers.

2 Labeler Accuracy Estimation

Before we discuss the human training and backward relabeling processes, we would like to firstly introduce our labeler accuracy estimation model.

For any worker, we use $s(x_i)$ to denote that he or she is assigned an instance x_i. The probability that the instance x_i has been correctly labeled can be formulated as

$$P^{(t)}(s(x_i) = y_i) = \frac{1}{1 + (b_{x_i} - 1)e^{-(c^{(t)}(x_i)-1)/\alpha_i}} \quad (1)$$

where b_{x_i} is the number of options queried for x_i; $c^{(t)}(x_i)\geq 1$ is the *competence estimate* for the labeler on x_i in labeling round $t \geq 0$; and $\alpha_i>0$ denotes the difficulty of instance x_i. As t increases, the competence estimation should become more accurate since more gold instances have been labeled. In the general case, b_{x_i} can be set to the total number of classes, which are assumed to be equally likely [Dontcheva *et al.*, 2014]. Strategies also exist for querying only the most closely related options, which may greatly improve the correctness ratio and reduce the labeling time [Sheng *et al.*, 2008]. For a situation in which the labeler has the lowest competence, $c^{(t)}(x_i)=1$, or the task is extremely difficult, $\alpha_i\to\infty$, the label assignment will be essentially a random guess; and at the other extreme, if the worker's competency is very high or the task is very easy, $P^{(t)}(s(x_i, t) = y_i)$ will tend toward 1.

A labeler may be influenced by the results displayed for a gold instance including its true label and explanation, potentially causing his or her competence to change and affecting future labeling activities. We call this process *inspiration*. The labeler's performance on later gold instances will not affect his or her previous answers; however, it can be used to estimate the labeler's ability and the correctness of those previous answers. We call this process *inference*.

We define two impact factors, $r_{ins}(x_i, x_j)$ and $r_{inf}(x_i, x_j)$, to represent the impact exerted on $x_i \in \mathcal{X}_L$ by the labeling of instance x_j

$$r_{ins}(x_i, x_j)=\begin{cases} 1 & , x_t\in\mathcal{X}_U \cup \mathcal{X}_L \\ \dfrac{1}{1 - e^{(-\sigma\delta(x_i,x_j))}} & , x_j\in\mathcal{X}_G \end{cases} \quad (2)$$

$$r_{inf}(x_i, x_j)=\begin{cases} 1 & , \quad x_j\in\mathcal{X}_U \cup \mathcal{X}_L \\ \dfrac{1-\lambda_1\mathbf{I}[s(x_i)=s(x_j)]-\lambda_2\mathbf{I}[s(x_i)=y_j]}{1 + e^{(-\sigma\delta(x_i,x_j))}} & , x_j\in\mathcal{X}_G \wedge s(x_j)\neq y_j \\ \dfrac{1 + \lambda\mathbf{I}[s(x_i) = y_j]}{1 - e^{(-\sigma\delta(x_i,x_j))}} & , x_j\in\mathcal{X}_G \wedge s(x_j)=y_j \end{cases} \quad (3)$$

where $\delta(x_i, x_j)$ represents the distance between two feature vectors and $\sigma>0$ is a coefficient. We can simply adopt the Euclidean distance to measure $\delta(x_i, x_j)$. If x_j was labeled before x_i, we believe that it will exert a positive impact, whereas if x_j was labeled after x_i, the impact may differ depending on the correctness of the label assigned to x_j.

The above definition is based on the assumption that a worker's hypotheses in image classification tasks do not change after he or she has labeled an unlabeled instance. By contrast, as expressed in (3), if the labeler incorrectly labels $x_j\in\mathcal{X}_G$, that incorrect label will exert a negative impact on all $x_i\in\mathcal{X}_L$; the resulting $r_{inf}(x_i, x_j)<1$ indicates that the labeler is more likely to have assigned an incorrect label to x_i and therefore decreases the competence estimate for x_i. Besides the distance between two instances, several other factors may also theoretically exert some influences, such as whether the label assigned to the previously labeled instance was the same as the ground-truth label of the gold instance or the same as the label that was incorrectly assigned to the gold instance. λ, λ_1 and λ_2 are parameters that control the relations among these factors. Otherwise, the function given in (3) results in a positive impact, with $r_{inf}(x_i, x_j)>1$.

Thus, after T rounds of labeling (of both unlabeled instances and gold instances), we can obtain the current competence estimate for any previously labeled instance x_i as follows, considering the constraint of $c(x_i, t)\geq 1$

$$c^{(t)}(x_i) = \max\left(1, c^{(0)}(x_i) \prod_{j=1}^{i} r_{ins}(x_i, x_j) \prod_{j=i+1}^{t} r_{inf}(x_i, x_j)\right) \quad (4)$$

$c^{(0)}(x_i)$ is the initial competence estimate for the labeler, which can be obtained based on the overall distribution for other labelers on the gold instances

$$c^{(0)}(x_i) = 1 - \frac{\log\left(\frac{\frac{1}{P^{(0)}(s(x_i)=y_i)}-1}{b_{x_i}-1}\right)}{\alpha_i} \quad (5)$$

Here, $P^{(0)}(s(x_i) = y_i)$ can be estimated using

$$P^{(0)}(s(x_i)=y_i) = \frac{1}{|\mathcal{X}_G|} \sum_{x_j\in\mathcal{X}_G} \mathbf{I}[s(x_j) = y(j)]g(x_i, x_j) \quad (6)$$

where $g(\cdot, \cdot)$ is a Gaussian kernel function. At the very beginning, when no responses from other labelers are available, we simply set $c^{(0)}(x_i) = 1$.

For the estimation of the parameters $\boldsymbol{\alpha}=\{\alpha_1, ..., \alpha_n\}$, we use an *expectation maximization (EM)* algorithm similar to that adopted in [Whitehill *et al.*, 2009], described as follows.

E-step: In the expectation step, the posterior probability of y_i given $\boldsymbol{\alpha}$ is computed as follows

$$p(y_i|\boldsymbol{\alpha}) \propto p(y_i) \prod_{x_i \in \mathcal{X}_L \cup \mathcal{X}_G} p(s(x_i), \alpha_i) \tag{7}$$

M-step: In the maximization step, we use the cost function $Q(\boldsymbol{\alpha})$ to estimate a locally optimal solution using (8)

$$Q(\boldsymbol{\alpha}) = E(\ln p(y_i)) + \sum_{x_i \in \mathcal{X}_L \cup \mathcal{X}_G} E(\ln p(s(x_i)|y_i, \alpha_i)) \tag{8}$$

Finally, the EM algorithm returns the parameters $\boldsymbol{\alpha}^*$ that maximize $Q(\boldsymbol{\alpha})$, i.e., $\boldsymbol{\alpha}^*$=$\arg\max_{\boldsymbol{\alpha}} Q(\boldsymbol{\alpha})$.

3 BALT Model

We present the three key mechanisms of our BALT model in Figure 1. Specifically, we investigate two basic questions: 1) which type of instances (unlabeled/gold/labeled) should be sampled? and 2) which instance should be sampled? Finally, we propose the BALT algorithm.

3.1 Forward Labeling

We adopt the uncertainty-based sampling strategy that selects samples with the maximal uncertainty, where the degree of uncertainty is measured in terms of entropy

$$x_u = \arg\max_{x \in \mathcal{X}_U} -\sum_{y \in Y} P(y|x, \theta_{LG}) \log P(y|x, \theta_{LG}) \tag{9}$$

where $P(y|x_i, \theta_{LG})$ is the conditional distribution of label y and θ_{LG} is the model trained on the instances in $\mathcal{X}_L \cup \mathcal{X}_G$.

3.2 Human Training

To choose the most representative gold instances, we consider two types of information: the information contained in the instance itself and the information related to instances labeled previously.

We measure the correlation between a gold instance and all currently labeled instances by means of

$$l(x_i) = \frac{l(x_i)}{\sum_{x \in \mathcal{X}_G} l(x)} l_0(x_i) \prod_{x \in \mathcal{X}_L} (1 + e^{(-\sigma\delta(x_i, x))}) \tag{10}$$

where the initial correlation is set as $l_0(x_i)$=$\frac{1}{|\mathcal{X}_G|}$.

Considering the above "self-information" and "related information" simultaneously, we select gold instances in each round in terms of

$$x_t = \arg\max_{x \in \mathcal{X}_G} H^{\gamma}(x, \theta_{LG}) l^{(1-\gamma)}(x) \tag{11}$$

where γ is a parameter that determines the weight of the two impact factors and satisfies $0 \leq \gamma \leq 1$. Whenever a gold instance x_t is selected, we set $l(x_t)$=0 to prevent that gold instance from being sampled multiple times.

3.3 Backward Relabeling

The key point here is that we should find an error-like instance that the current labeler can revise with a higher confidence. Formula (1) reveals the probability that an instance x_i has been correctly labeled. So, we can estimate the future correctness probability and competence estimate for an instance according to the *inspiration* factor, using

$$\widetilde{c}^{(t)}(x_i) = c^{(0)}(x_i) \prod_{j=1}^{t} r_{ins}(x_i, x_j) \tag{12}$$

$$\widetilde{P}^{(t)}(s(x_i) = y_i) = \frac{1}{1 + (b_{x_i} - 1) e^{-(\widetilde{c}^{(t)}(x_i) - 1)}} \tag{13}$$

Then, the sample with the maximum revision confidence can be chosen for relabeling according to

$$x_r = \arg\max_{x_i \in \mathcal{X}_L} \left((1 - P^{(t)}(s(x_i) = y_i)) \widetilde{P}^{(t)}(s(x_i) = y_i) \right) \tag{14}$$

After relabeling, we do not delete original label and aggregate final label through confidence-weighted majority voting such that $\arg\max_{s(x_r)} \sum_{s(x_r)} P^{(t+1)}(s(x_r) = y)$. Using confidence-weighted majority voting can prevent too many noisy labelers from generating incorrect labels. This approach gives the ability to rely more strongly on an answer given by a single labeler with a higher confidence than an answer given by two labelers with a lower confidence.

3.4 BALT Algorithm

Our BALT algorithm provides a solution to how often a worker is trained and how often a labeled instance is relabeled. We use a greedy approach to select the most feasible process in every labeling round, calculating contributions from three types of operations and choosing the most desirable one.

For each of the three sampling strategies, we calculate the expected reduction in the total number of errors when a candidate instance x^* is selected using that strategy as follows

$$\Delta E_u = \left(\sum_{x \in \mathcal{X}_U} 1 - P(\widehat{y}|x, \mathcal{X}_{LG}) \right) - \sum_i P(y_i|x^*, \theta_{LG}) \left(\sum_{x \in \mathcal{X}_U} 1 - P(\widehat{y}|x, \theta_{LG+(x^*, y_i)}) \right) \tag{15}$$

$$\Delta E_g = P^{(t)}(s(x_g) = y_g)(\sum_{x \in \mathcal{X}_U} (P(\widehat{y}|x, \theta_{LG \setminus (x^*, y^*)}) - P(\widehat{y}|x, \theta_{LG}))) \tag{16}$$

$$\Delta E_r = (1 - P^{(t)}(s(x^*) = y^*) \widetilde{P}^{(t)}(s(x^*) = y^*)) \times [(\sum_{x \in \mathcal{X}_U} 1 - P(\widehat{y}|x, \mathcal{X}_{LG})) - \sum_{y_i \neq s(x^*)} P(y_i|x^*, \theta_{LG}) (\sum_{x \in \mathcal{X}_U} 1 - P(\widehat{y}|x^*, \theta_{LG \setminus (x^*, s(x^*)) + (x^*, y_i)}))] \tag{17}$$

where $\widehat{y}$=$\arg\max_y P(y|x, \theta)$ denotes the class label with the highest posterior probability under model θ, $\theta_{LG} \setminus (x, y)$ denotes the model trained on $\mathcal{X}_L \cup \mathcal{X}_G$ excluding instance (x, y),

Algorithm 1: BALT Algorithm

Input: $\mathcal{X}_U$,$\mathcal{X}_L$,$\mathcal{X}_G$,$\mathcal{T}$,initial gold instance probability $\rho^{(0)}$

```
for t = 1 to T
    select x_u∈X_U, x_g∈X_G, x_g∈X_G using (9) (11) (14)
    compute ΔE_u, ΔE_g and ΔE_r using (15)-(17)
    if (1 − ρ^(t))ΔE_r + ρ^(t)ΔE_g > ΔE_r
        if(random(0,1)> ρ^(t))
            ask labeler to label x_u
            X_U = X_U − x_u
            X_L = X_L ∪ {x_u}, S_L = S_L ∪ {s(x_u)}
            retrain classifier
        else
            ask labeler to label x_g
            update ρ^(t+1) according to (18)
        end if
    else
        relabel x_r and retrain classifier if s'(x_r)≠s(x_r)
    end if
end for
```

and $\theta_{LG}+(x,y)$ denotes the model trained on $\mathcal{X}_L \cup \mathcal{X}_G$ with instance (x,y) included.

Further investigation reveals that ΔE_g is usually much smaller than the other two factors. Therefore, we use a parameter ρ to fix the proportion of gold instances sampled relative to the number of unlabeled instances sampled. Different people may have different levels of expertise. Intuitively, a good-quality labeler should not require training on many gold instances, while a poor-quality worker should be trained on more gold instances to achieve a higher accuracy. So, we can adaptively adjust the value of ρ according to the expected number of correct answers among a labeler's previous labels.

$$\rho^{(t)} = \rho^{(t-1)} \frac{\sum_{x_i \in \mathcal{X}_L} \frac{1}{b_{x_i}}}{\sum_{x_i \in \mathcal{X}_L} P^{(t)}(s(x_i) = y_i)} \tag{18}$$

It is easy to demonstrate that $E(\mathcal{X}_L) \geq \sum_{x_i \in \mathcal{X}_L} \frac{1}{b_{x_i}}$, where the equality holds in the case of a labeler who has never been trained on gold instances or who has assigned all incorrect labels to gold instances. In such a case, we believe that the labeler has not received enough training and should therefore be trained with a higher probability. Here, $\rho^{(0)}$ is one of key factors in determining the training frequency; and we further investigate this parameter in experiments presented later.

Our BALT is described in Algorithm 1. As in most active learning settings, the number of labeling rounds is constrained by a budget or to a fixed number. In each sampling process, the BALT may dynamically decide to perform either forward sampling or backward relabeling depending on the relative expectation values. The classifier will be retrained only when the set of labeled instances changes. The process of unlabeled instance sampling has a time complexity of $O(k|\mathcal{X}_U|)$, and gold instance sampling has a time complexity of $O(|\Gamma||\mathcal{X}_U||\mathcal{X}_L|)$. Both the computation of $\rho^{(t)}$ and backward instance relabeling have a time complexity of $O(t|\mathcal{X}_G||\mathcal{X}_L|)$. The computation of the expected reduction in the total number of errors has the highest complexity in the case of the backward expectation, i.e., $O(t^2|\mathcal{X}_G||\mathcal{X}_U|)$. In general, $t \ll |\mathcal{X}_U|, |\mathcal{X}_L|$. The most time-consuming process is the training of the classifier after the addition of new labeled instances. In addition to the last possible training process at the end of the algorithm execution, unlabeled instance sampling requires one additional round of classifier training, and the computation of the expected reduction in the total number of errors requires three additional rounds of classifier training.

4 Performance Evaluations

To validate our proposed BALT model and algorithm, we conducted experiments with real human on three real datasets. We developed PHP-based Web interfaces encapsulating various models and algorithms. 260 participants were recruited from CrowdFlower, and they executed three classification tasks through our Web interfaces. We used dense SIFT features as the input features for the used instances.

4.1 Datasets

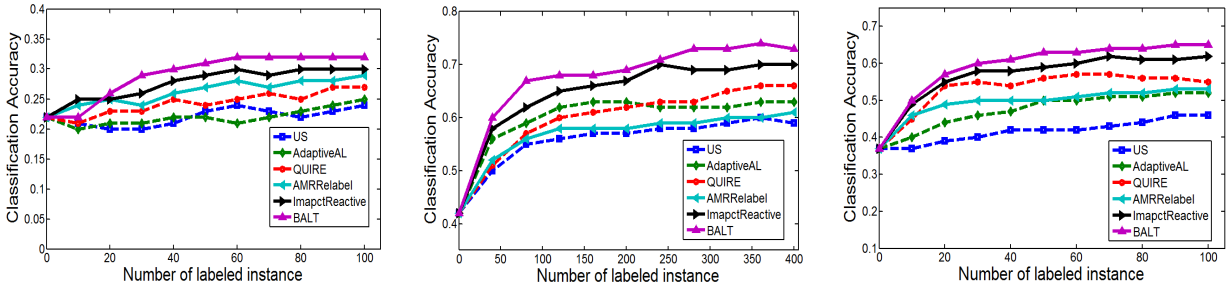
To evaluate the effectiveness of labeler training, we chose species classification tasks, which are challenging for non-domain experts. The labelers were informed of the total number of label classes before labeling. The experiments focused on the classification of various real images in three datasets.

- Bird species classification on the *Caltech-UCSD Birds* dataset [Welinder *et al.*, 2010], with 200 bird species and 6033 images. We extracted 13 bird species with 477 images. Each class contains more than 35 instances.
- Leaf species classification on the *Leafsnap* dataset [Kumar *et al.*, 2012], which contains 185 leaf (tree) species and 30866 images. Similarly, we chose a custom subset with 20 species and 3323 images in total.
- Butterfly species classification on the *Butterflies* dataset. We manually selected a total of 306 butterfly images representing 8 species from ImageNet.

4.2 Settings

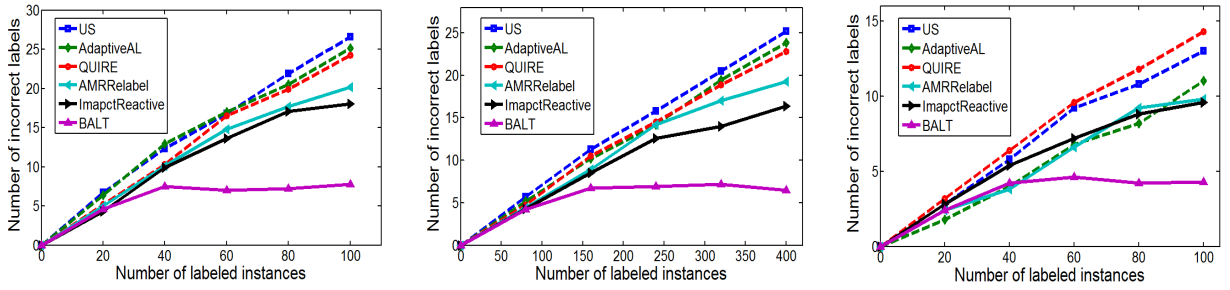
We chose five mainstream algorithms designed for *active learning* and *active learning with relabeling* for comparison with our BALT algorithm to enable its comprehensive evaluation. These algorithms include almost all mainstream types of sampling approaches used in image classification, including

- *Uncertainty Sampling (US)*, the most popular active learning algorithm, which is based on the maximum uncertainty [Lewis *et al.*, 1994];
- *AdaptiveAL* is an active learning approach that adaptively combines uncertainty measurements with information density measurements [Li and Guo, 2013];
- *QUIRE* represents another category of active learning algorithm in which instance label pairs are selected based on informativeness and representativeness [Huang *et al.*, 2014];
- *AMRRelabel* is an Absolute Majority Relabeling strategy for active learning, in which sampling is performed based on the uncertainty and inconsistency of instances [Zhao *et al.*, 2011]; and



(a) Birds (b) Leafsnap (c) Butterflies

Figure 2: Classification accuracy on the three datasets.



(a) Birds (b) Leafsnap (c) Butterflies

Figure 3: Numbers of incorrect labels on the three datasets.

- *ImpactReactive* is an impact (re)active sampling algorithm with relabeling, which chooses instances that most strongly impact the classifier [Lin and Weld, 2016].

We used the basic logistic regression classifier with L2 regularization implemented in Weka in combination with all of these methods. 180 participants were asked to complete all three tasks, and they were evenly divided into different groups corresponding to the different algorithms, each with 30 participants. For each group, we randomly partitioned all labelers into 10 sub-groups. Each small group performed an independent experiment, and we report the average performance values in the following evaluation. We compared the models under the same total number of labeling rounds, where each gold instance sampled and each instance chosen for relabeling was also considered in same round.

For each dataset, we partitioned the samples into three subsets: *gold instances*, *unlabeled instances* and *testing instances*, with proportions of 5%, 65% and 30%, respectively. The gold instances are instances that have ground-truth labels and some simple annotation, and they cover all species in the classified image sets. For the five algorithms considered for comparison with our algorithm, the gold instances can be regarded as previously prepared labeled instances. For all six algorithms, we started by training the classifiers on the gold instances. The remaining 65% and 30% of the samples were used for training and testing, respectively.

In our BALT algorithm, we set the initial probability of gold instance sampling as $\rho^{(0)}$=0.2 and set the following parameter weights: σ=0.1 and λ=λ_1=λ_2=0.05. Note that we used a serial sampling strategy in which the classifier was updated online whenever the training sample set changed [Zhang *et al.*, 2015]. This strategy may increase the computation complexity to some extent; however, it allows instances to be selected more effectively.

In all experiments, we continuously sampled 100, 400 and 100 instances on the Birds, Leafsnap and Butterflies datasets, respectively.

4.3 Classification Accuracy and Labeler Expertise

Figures 2(a), 2(b) and 2(c) show the classification accuracy results for the six algorithms on the three datasets. It is clear that our BALT algorithm significantly outperforms the other five on all three datasets. This is true even though, for the same number of rounds, fewer of the unlabeled instances have actually been labeled in the case of the BALT. Thus, these findings demonstrate that the accuracy of the labels is sometimes more important than the number of labels. Our BALT can improve labelers' expertise through training on gold instances, thereby causing the proportion of correct labels to increase continuously. These results also reveal the adaptability of the BALT to multiple datasets with different features.

In Figure 2(a), there is a brief initial period in which the BALT does not show the highest performance. This occurs because during the initial stage, the BALT has not yet identified incorrectly labeled instances. Moreover, during initial labeling, the labelers have not yet received sufficient training on gold instances. However, labelers gain higher and higher expertise in our BALT, enabling them to generate higher-quality labels.

Figures 3(a), 3(b) and 3(c) record the number of instances

labeled incorrectly versus the number of labeling rounds. The results show that our BALT always results in the lowest number of incorrect labels for all three tasks. Specifically, in other five algorithms, the number of instances labeled incorrectly increases proportionally to the number of labeled instances, which demonstrates that without training on specific related knowledge, labelers will tend to continue to make mistakes at a nearly constant rate. By contrast, in our BALT, the number of instances with incorrect labels remains relatively stable once the labelers have received sufficient training. At some points, the number of incorrect labels even decreases because the BALT gives labelers the ability to later revise some incorrect labels once they have been further trained.

Although AMRRelabel and ImpactReactive also include a relabeling process and consequently generate fewer instances with incorrect labels compared with the other three algorithms, they enable the revision of only a small number of incorrect labels. On one hand, without receiving external information, a labeler will have great difficulty correctly revising his or her own answers to instances labeled previously. On the other hand, if such instances are relabeled by different labelers, the results are difficult to determine, since these two algorithms do not estimate worker reliability and the new labeler may generate even worse labels. By contrast, with the improvement in labeler expertise facilitated by our BALT algorithm, labelers will be much more likely to correctly revise their own previous answers, and they will also achieve higher correctness ratio in subsequent labeling activities, regardless of whether they are labeling new instances or relabeling instances that have previously been labeled by others.

From the perspective of dataset construction, we find that the number of incorrect labels is positively correlated with the difficulty of the classification task. According to the numbers of incorrect labels shown in Figures 3, in the case of the other five algorithms, the labelers generated 24.73%, 6.025% and 12.98% incorrect labels on average on the Birds, Leafsnap and Butterflies datasets, respectively, whereas our BALT algorithm resulted in only 19.25%, 4.75% and 11.5% incorrect labels, respectively, on these three datasets during labeling. The reason for this finding is that the human training process can significantly improve labelers' expertise and thus their probability of correctly labeling later instances. Instead, the other five algorithms do not train the labelers. These results demonstrate that the three tasks can be ranked as *Birds*>*Butterflies*>*Leafsnap* in terms of relative difficulty.

There are several reasons for this ranking. One is that leaves have fewer features than birds and butterflies. Moreover, some birds in our dataset belong to the same subcategory and therefore are more difficult to identify. The probably of incorrect labeling for butterfly images is lower than that for bird images because the Butterflies dataset contains fewer classes. Consequently, the bird classification task generated the most incorrect labels, as shown in Figure 2(a). These findings demonstrate that active learning algorithms are quite sensitive to errors and cannot perform well in the presence of too many incorrect labels. Figure 2(b) reveals that fewer incorrect labels were generated in the leaf classification task, for which the curves are similar to those for error-free active learning algorithms [Settles, 2010]. In general, reducing the number of incorrect labels is more important than the complexity of the classification algorithm.

4.4 Average Labeling Time

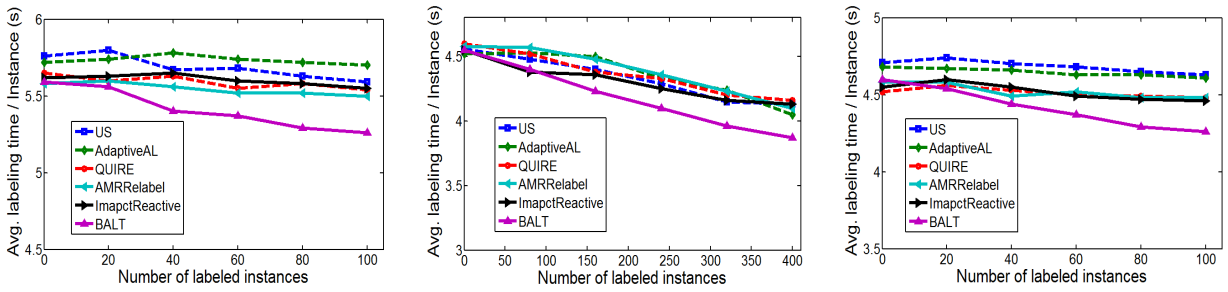
In this section, we compare the labeling times of the labelers for the six algorithms on the three datasets to capture the algorithms' labeling efficiency.

As illustrated in Figure 4, our BALT always uses the shortest labeling times on all three datasets. For all six algorithms, the average labeling time decreases as the number of labeled instances increases with different rates. These results indicate that labelers generally work increasingly quickly as time passes, because they both become more proficient and may naturally lose patience. The leaf classification task was both simpler than the other two tasks and included more labeling rounds. Therefore, the labelers spent the least average time on labeling the leaf images, and the corresponding curves fall at the fastest rates. For the initial stage of each of the three tasks, the six algorithms show similar labeling times because of the similar skill levels of the labelers. However, as the number of labeled instances increases, our BALT significantly outperforms the others, reflecting the round-by-round improvement in expertise promoted by our BALT algorithm for the same level of patience on the part of the labelers.

4.5 Gold Instance Sampling Probability

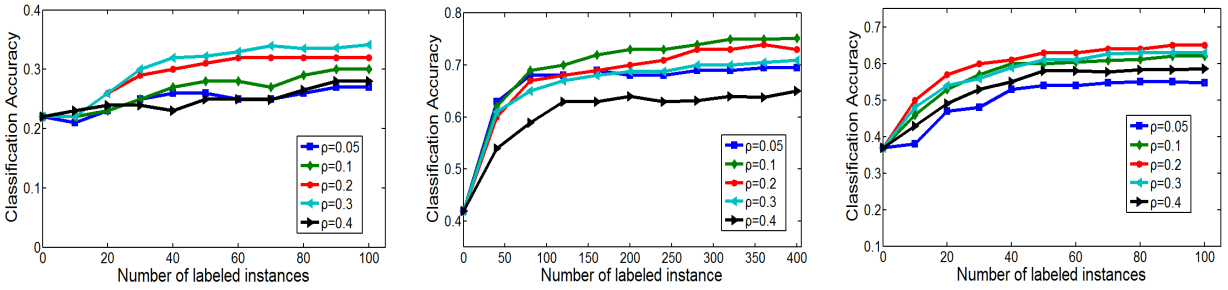
The gold instance sampling probability ρ, which plays a key role in determining the efficiency and effectiveness of training, is largely determined by the initial sampling probability $\rho^{(0)}$. Here, we report experiments conducted to identify the optimal probability $\rho^{(0)}$ that can guarantee both a high correct labeling ratio and a high labeling efficiency. We recruited additional 80 participants to test our BALT algorithm on the three tasks with the following settings: $\rho^{(0)} = [0.05, 0.1, 0.3, 0.4]$. For all parameters except $\rho^{(0)}$, the settings were the same as above.

Figure 5 illustrates how the initial sampling probability affects the classifier performance on the three tasks. From these results, we can conclude that the classifier performance is not proportional to the value of $\rho^{(0)}$. Too large or too small $\rho^{(0)}$ will result in poor classification accuracy. If $\rho^{(0)}$ is too small, the labelers will not receive sufficient training and will consequently generate more noisy labels. Instead, if $\rho^{(0)}$ is too large, the classifier will be trained with fewer samples, and the labelers may also receive redundant training. For the Birds, Leafsnap and Butterflies datasets, the highest performances are achieved with $\rho^{(0)}$=0.3, $\rho^{(0)}$=0.1 and $\rho^{(0)}$=0.2, respectively, and these optimal values can be considered to be correlated with the difficulty of the different tasks. As mentioned previously, the bird classification task is relatively difficult for labelers so that potential errors cannot be effectively reduced without sufficient training. By contrast, leaf classification is the easiest task and already generates relatively few incorrect labels, and excessive training on gold instances is a waste of the labelers' time. Therefore, we select $\rho^{(0)} = 0.2$ as an optimal and feasible choice for all of the experiments.



(a) Birds (b) Leafsnap (c) Butterflies

Figure 4: Execution time per instance versus the number of instances.



(a) Birds (b) Leafsnap (c) Butterflies

Figure 5: Classification accuracy for different initial gold instance sampling probabilities ρ on the three datasets.

5 Related Work

Active learning can reduce labeling costs [Sachan *et al.*, 2015; Qian *et al.*, 2013]. However, it is also more sensitive to incorrectly labeled instances, since it requires fewer training samples [Lin *et al.*, 2014].

Some researchers have attempted to mitigate this error-sensitivity problem by asking labelers to relabel error-like instances. The related methods can be divided into two categories. The first one treats relabeling as an individual process that is executed either in every labeling round or based on a certain triggering condition. [Zhao *et al.*, 2011] proposed an active learning method based on the MaxMin Margin strategy. [Zhang *et al.*, 2015] used a bidirectional active learning framework to relabel some previously labeled instances in every round. This framework allows labeled data to be effectively reevaluated and provides guidance for unlabeled data sampling.

The second one combines relabeling using forward sampling [Kamar and Horvitz, 2015]. [Lin and Weld, 2016] generalized the process of relabeling in active learning as *reactive learning* and proposed several new algorithms. [Ipeirotis *et al.*, 2014] considered the quality of both labels and labelers when proposing an uncertainty-preserving sampling strategy. However, these studies did not consider the fact that labelers without domain expertise are have difficulty making correct revisions during relabeling. Due to this phenomenon, crowd training is necessary for many crowdsourcing systems that involve complex tasks.

Crowd training is an effective way to help crowd workers to gain relevant domain expertise and improve task performance [Truby *et al.*, 2014]. The authors in [Johns *et al.*, 2015] proposed an interactive machine teaching algorithm to maximize students' classification ability. [Zhu, 2013] employed Bayesian models to investigate an optimization problem by good examples. However, to the best of our knowledge, no previous work has studied the training of labelers in the active learning scenario.

6 Conclusion

In this paper, we combine active learning with human training and propose the *BALT* model to address the error-sensitive problem in active learning. Under this model, we propose a *training-while-working* scheme that allows labelers to effectively gain domain expertise through training on gold instances during their labeling work, thereby improving their subsequent performance. We estimate the probabilities that instances are incorrectly labeled based on labelers' performance on gold instances and select error-like instances for relabeling. Experimental results on three different datasets demonstrate that our BALT algorithm significantly outperforms other related proposals in reducing errors and improving labelers' expertise.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China projects under Grants 61832013 and 61672351, and in part by the Huawei Technologies Co., Ltd project under Grant YBN2018125107. Feilong Tang is the corresponding author of this paper.

References

[Amir *et al.*, 2016] Ofra Amir, Ece Kamar, Andrey Kolobov, and Barbara Grosz. Interactive teaching strategies for agent training. In *IJCAI*, pages 804-811, 2016.

[Cakmak and Thomaz, 2014] Maya Cakmak and Andrea L Thomaz. Eliciting good teaching from humans for machine learners. *Artificial Intelligence*, 217:198–215, 2014.

[Deng *et al.*, 2013] Jia Deng, Jonathan Krause, and Li Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *CVPR*, pages 580-587, 2013.

[Donmez *et al.*, 2009] Pinar Donmez, Jaime G Carbonell. Efficiently learning the accuracy of labeling sources for selective sampling. In *SIGKDD*, pages 259-268, 2009.

[Dontcheva *et al.*, 2014] Mira Dontcheva, R Morris. Combining crowdsourcing and learning to improve engagement and performance. In *CHI*, pages 3379-3388, 2014.

[Whitehill *et al.*, 2009] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R. Movellan, Paul L. Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In *Advances in Neural Information Processing Systems*, 2009, pp. 2035-2043.

[Gadiraju *et al.*, 2015] Ujwal Gadiraju, Besnik Fetahu. Training workers for improving performance in crowdsourcing microtasks. In *EC-TEL*, pages 100-114, 2015.

[Golovin *et al.*, 2010] Daniel Golovin, Andreas Krause, and Debajyoti Ray. Near-optimal bayesian active learning with noisy observations. In *NIPS*, pages 766-774, 2010.

[Li et al., 2016] Qi Li, Fenglong Ma, Jing Gao, Lu Su, Christopher J. Quinn. Crowdsourcing high quality labels with a tight budget In *ACM International Conference on Web Search and Data Mining*, pages 237-246, 2016.

[Harris, 2011] Christopher Harris. Youre hired! an examination of crowdsourcing incentive models in human resource tasks. In *CSDM*, pages 15-18, 2011.

[Huang *et al.*, 2014] Sheng-Jun Huang, Rong Jin, and Zhihua Zhou. Active learning by querying informative & representative examples. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(10):1936-1949, 2014.

[Ipeirotis *et al.*, 2014] Panagiotis G. Ipeirotis, Foster Provost, Victor S. Sheng, and Jing Wang. Repeated labeling using multiple noisy labelers. *Data Mining and Knowledge Discovery*, 28(2):402–441, 2014.

[Johns *et al.*, 2015] Edward Johns, Oisin Mac Aodha, and Gabriel J Brostow. Becoming the expert-interactive multiclass machine teaching. In *CVPR*, pages 1-9, 2015.

[Kamar and Horvitz, 2015] Ece Kamar and Eric Horvitz. Planning for crowdsourcing hierarchical tasks. In *AAMAS*, pages 1191-1199, 2015.

[Kumar *et al.*, 2012] Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, and João VB Soares. Leafsnap: A computer vision system for automatic plant species identification. In *ECCV*. 2012.

[Li and Guo, 2013] Xin Li and Y Guo.Adaptive active learning for image classification.In *CVPR*,pages 859-866,2013.

[Li *et al.*, 2013] Shoushan Li, Yunxia Xue, Zhongqing Wang, and G Zhou. Active learning for cross-domain sentiment classification. In *IJCAI*, pages 2127-2133, 2013.

[Lin and Weld, 2016] Christopher H Lin and Daniel S Weld. Re-active learning: Active learning with relabeling. In *AAAI*, pages 1845-1852, 2016.

[Lin *et al.*, 2014] Christopher H Lin, Daniel S Weld, et al. To re (label), or not to re (label). In *HCOMP*, 2014.

[Oleson *et al.*, 2011] David Oleson, Alexander Sorokin, Greg P Laughlin, Vaughn Hester, John Le. Programmatic gold: Targeted and scalable quality assurance in crowdsourcing. *Human computation*, 2011.

[Qian *et al.*, 2013] Buyue Qian, Xiang Wang, Fei Wang, Hongfei Li, Jieping Ye, and Ian Davidson. Active learning from relative queries. In *IJCAI*, pages 1614-1620, 2013.

[Roy and McCallum, 2001] Nicholas Roy and Andrew McCallum. Toward optimal active learning through monte carlo estimation of error reduction. *ICML*, 2001.

[Sachan *et al.*, 2015] Mrinmaya Sachan, Eduard Hovy, and Eric P Xing. An active learning approach to coreference resolution. In *IJCAI*, pages 1312-1318, 2015.

[Servajean *et al.*, 2016] Maximilien Servajean, Alexis Joly, Dennis Shasha, Julien Champ, and Esther Pacitti. Theplantgame: Actively training human annotators for domain-specific crowdsourcing. In *ACMMM*, 2016.

[Settles, 2010] Burr Settles. Active learning literature survey. *University of Wisconsin, Madison*, 2010.

[Sheng *et al.*, 2008] Victor S Sheng, Foster Provost, and Panagiotis G Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In *SIGKDD*, pages 614-622, 2008.

[Lewis *et al.*, 1994] David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *SIGIR*, pages 3-12, 1994.

[Singla *et al.*, 2014] Adish Singla, Ilija Bogunovic, Gábor Bartók, Amin Karbasi, and Andreas Krause. Near-optimally teaching the crowd to classify. In *ICML*, 2014.

[Truby *et al.*, 2014] Katherine Truby, Meredith L Weiss, and David L Rousseau. Teaching the unfamiliar to a crowd. *PS: Political Science & Politics*, 47(1):189–194, 2014.

[Welinder *et al.*, 2010] Peter Welinder, Steve Branson, Takeshi Mita, Catherine Wah, Florian Schroff, Serge Belongie, and P Perona. Caltech-ucsd birds 200. 2010.

[Zhang *et al.*, 2015] Xiao-Yu Zhang, Shupeng Wang, and Xiaochun Yun. Bidirectional active learning: a two-way exploration into unlabeled and labeled data set. *IEEE Trans Neural Netw Learn Syst.*, 26(12):3034-44, 2015.

[Zhao *et al.*, 2011] Liyue Zhao, Gita Sukthankar, and Rahul Sukthankar. Incremental relabeling for active learning with noisy crowdsourced annotations. In *PASSAT/SocialCom*, pages 728-733, 2011.

[Zhu, 2013] Xiaojin Zhu. Machine teaching for bayesian learners in the exponential family. In *NIPS*, pages 1905-1913, 2013.