Improve Learning from Crowds via Generative Augmentation

Zhendong Chu, Hongning Wang
Department of Computer Science
University of Virginia
Charlottesville, VA 22903, USA
{zc9uy,hw5x}@virginia.edu

ABSTRACT

Crowdsourcing provides an efficient label collection schema for supervised machine learning. However, to control annotation cost, each instance in the crowdsourced data is typically annotated by a small number of annotators. This creates a *sparsity* issue and limits the quality of machine learning models trained on such data. In this paper, we study how to handle sparsity in crowdsourced data using data augmentation. Specifically, we propose to directly learn a classifier by augmenting the raw sparse annotations. We implement two principles of high-quality augmentation using Generative Adversarial Networks: 1) the generated annotations should follow the distribution of authentic ones, which is measured by a discriminator; 2) the generated annotations should have high mutual information with the ground-truth labels, which is measured by an auxiliary network. Extensive experiments and comparisons against an array of state-of-the-art learning from crowds methods on three real-world datasets proved the effectiveness of our data augmentation framework. It shows the potential of our algorithm for low-budget crowdsourcing in general.

CCS CONCEPTS

• **Information systems** → **Crowdsourcing**; • **Computing methodologies** → *Adversarial learning*.

KEYWORDS

Crowdsourcing, generative adversarial nets, label noise

ACM Reference Format:
Zhendong Chu, Hongning Wang. 2021. Improve Learning from Crowds via Generative Augmentation. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3447548.3467409

1 INTRODUCTION

Modern machine learning systems are data hungry, especially for labeled data, which unfortunately is expensive to acquire at scale. Crowdsourcing provides a label collection schema that is both cost- and time-efficient [4]. It spurs the growing research efforts in directly learning a classifier with only crowdsourced annotations, aka the *learning from crowds* problem.

In practice, to minimize annotation cost, the instances in crowdsourced data are typically labeled by a small number of annotators; and each annotator will only be assigned to a few instances. This introduces serious sparsity in crowdsourced data. We looked into two widely-used public crowdsourced datasets for multi-class classification, one for image labeling (referred to as LabelMe [25, 27]) and one for music genre classification (referred to as Music [26]). On the LabelMe dataset, each instance is only labeled by 2.5 annotators on average (out of 59 annotators), while 88% annotators provide less than 100 annotations (out of 1,000 instances). On the Music dataset, each instance is labeled by 4.2 annotators on average (out of 44 annotators), while 87.5% annotators provide less than 100 annotations (out of 700 instances). Such severe sparsity hinders the utility of crowdsourced labels. On the instance side, annotations provided by non-experts are noisy, which are expected to be improved by redundant annotations. But subject to the budget constraint, redundancy is also to be minimized. This conflict directly limits the quality of crowdsourced labels. On the annotator side, most existing crowdsourcing algorithms model annotator-specific confusions, which are used for label aggregation [9], task assignment [10, 21] and annotator education [29]. But due to the limited observations per annotator, such modeling can hardly be inaccurate, and thus various approximations (e.g., strong independence assumptions [9]) have to be devised.

A straightforward solution to address annotation sparsity is to recruit more annotators or increase their assignments, at the cost of an increasing budget. This however is against the goal of crowdsourcing, i.e., to collect labeled data at a low cost. We approach the problem from a different perspective: we perform data augmentation using generative models to fill in the missing annotations. Instead of collecting more real annotations, we generate annotations by modeling the annotation distribution on instances and annotators. Given our end goal is to obtain an accurate classifier, the key is to figure out *what annotations best help the classifier's training*. We propose two important criteria. First, the generated annotations should follow the distribution of authentic ones, such that they will be consistent with the label confusion patterns observed in the original annotations. Second, the generated annotations should well align with the ground-truth labels, e.g., with high mutual information [13, 38], so that they will be informative about ground-truth labels to the classifier.

We realize our criteria for annotation augmentation in crowdsourced data using Generative Adversarial Networks (GAN) [11]. The end product of our solution is a classifier, which predicts the label of a given instance. We set a discriminative model to judge

KDD '21, August 14–18, 2021, Virtual Event, Singapore

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467409

whether an annotation is authentic or generated. Meanwhile, a generative model aims to generate annotations following the distribution of authentic annotations under the guidance of the discriminative model. On a given instance, the generator takes the classifier's output and the annotator and instance features as input to generate the corresponding annotation. To ensure the informativeness of generated annotations, we maximize the mutual information between the classifier's predicted label and the generated annotation on each instance [7]. A two-step training strategy is proposed to avoid model collapse. We name our framework as *CrowdInG* - learning with **Crowd**sourced data through **In**formative **G**enerative augmentation. Extensive experiments on three real-world datasets demonstrated the feasibility of data augmentation for the problem of learning from crowds. Our solution outperformed a set of state-of-the-art crowdsourcing algorithms; and its advantage becomes especially evident with extremely sparse annotations. It provides a new opportunity for low-budget crowdsourcing in general.

2 RELATED WORKS

Our work studies the learning from crowds problem. Raykar et al. [24] employed an EM algorithm to jointly estimate the expertise of different annotators and a logistic regression classifier on crowdsourced data. They followed the well-known Dawid and Skene (DS) model [9] to model the observed annotations. Albarqouni et al. [1] extended this solution by replacing the logistic classifier with a deep neural network classifier. Rodrigues and Pereira [25] further extended the solution by replacing the confusion matrix in the DS model with a neural network to model annotators' expertise, and trained the model in an end-to-end manner. Guan et al. [12] used a neural classifier to model each annotator, and aggregated the predictions from the classifiers by a weighted majority vote. Cao et al. [5] proposed an information-theoretical deep learning solution to handle the correlated mistakes across annotators. However, all the mentioned solutions only use the observed annotations, such that their practical performance is limited by the sparsity of annotations.

Another research line focuses on modeling the annotators. Whitehill et al. [37] proposed a probabilistic model which considers both annotator accuracy and instance difficulty. Rodrigues et al. [26] modeled the annotation process by a Gaussian process. Imamura et al. [14] and Venanzi et al. [33] extended the DS model by sharing the confusion matrices among similar annotators to improve annotator modeling with limited observations. Confusions of annotators with few annotations are hard to be modeled accurately, and Kamar et al. [17] proposed to address the issue with a shared global confusion matrix. Chu et al. [8] also set a global confusion matrix, which is used to capture the common confusions beyond individual ones. However, the success of the aforementioned models relies on the assumed structures among annotators or annotations. Such strong assumptions are needed, because the sparsity in the annotations does not support more complicated models. But they also restrict the modeling of crowdsourced data, e.g., introducing bias in the learnt model. We lift such restrictions by directly generating annotations, such that our modeling of crowdsourced data even does not make any class- or annotator-dependent assumptions.

Benefiting from their powerful modeling capabilities, deep generative models have been popularly used for data augmentation purposes. Most efforts have been spent on problems in a continuous space, such as image and video generations. Semi-supervised GAN [2, 22, 30] augments training data by generating new instances from labeled ones. Chae et al. [6] employed GAN to address the data sparsity in content recommendation, with their proposed real-value, vector-wise recommendation model training. Recently, GAN has also been adopted in data augmentation for discrete problems. Wang et al. [36] designed a two-step solution to perform GAN training for collaborative filtering. Wang et al. [34] unified generative and discriminative graph neural networks in a GAN framework to enhance the graph representation learning. Irissappane et al. [15] reduced the needed labeled data to fine-tune the BERT-like text classification models via GAN-generated examples.

3 METHODOLOGY

In this section, we begin our discussion with the background techniques of our augmentation framework, including GAN and InfoGAN. Then we present our CrowdInG solution and discuss its detailed design of each component. Finally, we describe our two-step training strategy for the proposed framework.

3.1 Background

3.1.1 Generative Adversarial Networks. Goodfellow et al. [11] introduced the GAN framework for training deep generative models as a *minimax* game, whose goal is to learn a generative distribution $P_G(x)$ that aligns with the real data distribution $P_{\text{true}}(x)$. The generative distribution is imposed by a generative model G, which transforms a noise variable $\boldsymbol{\varepsilon} \sim P_{noise}(\boldsymbol{\varepsilon})$ into a sample $G(\boldsymbol{\varepsilon})$. A discriminative model D is set to distinguish between the authentic and generated samples. The generator G is trained by playing against the discriminator D. Formally, G and D play the following two-player minimax game with value function $V(G, D)$:

$$\min_G \max_D V(G,D) = \mathbb{E}_{x\sim P_{\text{true}}}[\phi(D(x))] + \mathbb{E}_{\boldsymbol{\varepsilon}\sim P_{\text{noise}}}[\phi(1 - D(G(\boldsymbol{\varepsilon})))],$$

where ϕ is a function of choice and $\log(\cdot)$ is typically the choice. The optimal parameters of the generator and the discriminator can be learned by alternately maximizing and minimizing the value function $V(G, D)$. In this paper, we adopt this idea to model the annotation distribution: a generator is used to generate annotations on specific instances and annotators; and a discriminator is set to distinguish the authentic annotations from the generated ones.

3.1.2 Information Maximizing Generative Adversarial Networks. Chen et al. [7] extended GAN with an information-theoretic loss to learn disentangled representations for improved data generation. Aside from the value function $V(G, D)$, InfoGAN also maximizes the mutual information between a small subset of latent variables (referred to as latent code z) and the generated data. The generator takes both random noise $\boldsymbol{\varepsilon}$ and latent code z as input, where the latent code is expected to capture the salient structure in the data distribution. The minimax game then turns into an information-regularized form,

$$\min_G \max_D V_I(G,D) = V(G,D) - \lambda I(z; G(\boldsymbol{\varepsilon}, z)),$$

where $I(x; y)$ is the mutual information between random variables x and y, and λ is the regularization coefficient.

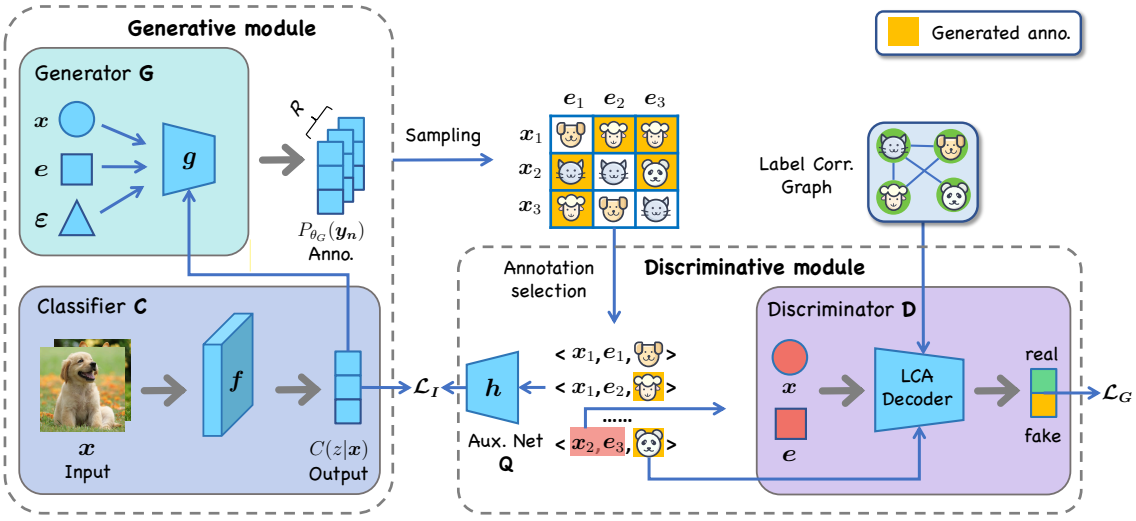


Figure 1: Overview of CrowdInG framework. We first sample annotations from annotation distributions provided by the generator. The discriminator and the auxiliary network are trained on the selected annotations. Then, the classifier is first fixed and the generator is updated according to $\mathcal{L}_G$ and $\mathcal{L}_I$. The generator is fixed and the classifier is updated according to $\mathcal{L}_G$.

3.2 The CrowdInG framework

Let $\mathcal{S} = \{\boldsymbol{x}_n, \boldsymbol{y}_n\}_{n=1}^N$ denote a set of N instances labeled by R annotators out of $|C|$ possible classes. We define $\boldsymbol{x}_n \in \mathbb{R}^d$ as the feature vector of the n-th instance and $y_n^r \in C$ as its annotation provide by the r-th annotator. $\boldsymbol{y}_n$ is thus the annotation vector (with missing values) from R annotators for the n-th instance. When available, the feature vector of the r-th annotator is denoted as $\boldsymbol{e}_r$; otherwise, we use a one-hot vector to represent an annotator. Each instance is associated with an unobserved ground-truth label $z \in C$. The goal of learning from crowds is to obtain a classifier $C(z|\boldsymbol{x})$ that is directly estimated from $\mathcal{S}$.

The framework of CrowdInG is depicted in Figure 1. It consists of two main components: 1) a generative module, including a classifier and a generator; and 2) a discriminative module, including a discriminator and an auxiliary network. In the generative module, the classifier first takes an instance $\boldsymbol{x}_n$ as input and outputs its predicted label distribution $P_{\theta_C}(z_n|\boldsymbol{x}_n)$. For simplicity, we collectively denote classifier's output for an instance $\boldsymbol{x}_n$ as $\hat{\boldsymbol{z}}_n$. And then the generator takes the instance $\boldsymbol{x}_n$, annotator $\boldsymbol{e}_r$, the classifier's output $\hat{\boldsymbol{z}}_n$, together with a random noise vector $\boldsymbol{\varepsilon}$, to generate the corresponding annotation distribution $P_{\theta_G}(y_n^r|\boldsymbol{x}_n, \boldsymbol{e}_r, \hat{\boldsymbol{z}}_n, \boldsymbol{\varepsilon})$. The discriminative module is designed based on our criteria of high-quality annotations to evaluate the generations. On one hand, the discriminative module uses a discriminator to differentiate whether the annotation triplet $(\boldsymbol{x}_n, \boldsymbol{e}_r, y_n^r)$ is authentic or generated. On the other hand, the discriminative module penalizes the generation based on the mutual information between the generated annotation and classifier's output measured by an auxiliary network. Following the idea of InfoGAN, we treat the classifier's output $\hat{\boldsymbol{z}}$ as the latent code in our annotation generation. And the auxiliary network measures the mutual information between $\hat{\boldsymbol{z}}$ and y. The two modules play a minimax game in CrowdInG. A better classifier is expected as the discriminative module faces more difficulties in recognizing the generated annotations during training.

3.2.1 Generative module. The output of the generative module is an annotation distribution for a given annotator-instance pair $(\boldsymbol{x}_n, \boldsymbol{e}_r)$. Sampling is applied to obtain the final annotations. As shown in Figure 1, this is a two-step procedure. First, the classifier $C(z_n|\boldsymbol{x}_n; \theta_C)$ predicts the label of a given instance $\boldsymbol{x}_n$ by

$$P_{\theta_C}(z_n = c|\boldsymbol{x}_n) \propto \exp[f(\boldsymbol{x}_n, z_n = c)],$$

where $f(\cdot)$ is a learnable scoring function chosen according to the specific classification tasks. Then the generator G takes the classifier's output $\hat{\boldsymbol{z}}$ as input to predict the underlying annotation distribution for the given annotator-instance pair. Moving beyond the classical class-dependent annotation confusion assumption [9, 25], we impose a much more relaxed generative process about the annotations. We consider the confusions can be caused by instance difficulty, or annotator expertise, or true labels of the instances (e.g., different annotation difficulty in different label categories), or even some random noise. To realize the idea, we provide the feature vector $\boldsymbol{x}_n$ of the instance, the annotator $\boldsymbol{e}_r$ and the classifier's output $\hat{\boldsymbol{z}}_n$ to the generator as input, and the corresponding annotation distribution is modeled as,

$$P_{\theta_G}(y_n^r = c|\boldsymbol{x}_n, \boldsymbol{e}_r, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n) \propto \exp[g(y_n^r = c, \boldsymbol{x}_n, \boldsymbol{e}_r, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n)], \quad (1)$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, 1)$ is a random noise vector, $g(\cdot)$ is a learnable scoring function implemented via a neural network. The generated annotations are sampled from the resulting distribution P_{θ_G}. To simplify our notations, we use $G(\boldsymbol{x}_n, \boldsymbol{e}_r, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n)$ to represent the predicted annotation distribution; and when no ambiguity is invoked, we denote $G(y_n^r)$ as its c-th entry when $y_n^r = c$. Thanks to our data augmentation framework, we can afford a more flexible modeling of the annotation noise, e.g., dropping the hard independence assumptions made in previous works [9, 25]. This in turn helps us boost the quality of generated annotations.

3.2.2 Discriminative module. We realize our principles of high-quality annotations in the discriminative module. First, the discriminator D aims to differentiate whether an annotation y_n^r is authentic from annotator $\boldsymbol{e}_r$ to instance $\boldsymbol{x}_n$, i.e., $D(y_n^r|\boldsymbol{x}_n, \boldsymbol{e}_r; \theta_D)$ predicts the probability of annotation y_n^r being authentic. In a crowdsourcing task, an annotator might confuse a ground-truth label with several classes, such that all of the confused classes could be independently authentic. For example, if an annotator always confuses "birds" with "airplanes" in low resolution images, his/her annotations might be random between these two categories. And thus both types of annotations should be considered as valid, as there is no way to tell which annotation is "correct" only based on the observations of his/her annotations. As a result, we realize the discriminator as a multi-label classifier, which takes an annotation triplet $(\boldsymbol{x}_n, \boldsymbol{e}_r, y_n^r)$ as input and calculates the discriminative score by a bilinear model,

$$D(y_n^r = c|\boldsymbol{x}_n, \boldsymbol{e}_r; \theta_D) = \sigma(\boldsymbol{u}_r^\top \boldsymbol{M}_c \boldsymbol{v}_n), \tag{2}$$

$$\boldsymbol{u}_r = \boldsymbol{W}_u \boldsymbol{e}_r + b_u, \boldsymbol{v}_n = \boldsymbol{W}_v \boldsymbol{x}_n + b_v,$$

where $\sigma(\cdot)$ is the sigmoid function, $\boldsymbol{M}_c$ is the weight matrix for class c, $(\boldsymbol{W}_v, b_v)$ and $(\boldsymbol{W}_u, b_u)$ are weight matrices and bias terms for annotator and instance embedding layers. For simplicity, we denote $D(y_n^r)$ as the discriminator's output on annotation y_n^r.

However, Eq (2) does not consider the correlation among different classes in the annotations, as it still evaluates each possible label independently. The situation becomes even worse with sparse observations in individual annotators. For example, when an annotator confuses "bird" with "airplanes", the discriminator might decide the label of "bird" is more authentic for this annotator, simply because this category appears more often in the annotator's observed annotations. To capture such "*equally plausible*" annotations, we equip the discriminator with additional label correlation information [20]. Specifically, we use a graph convolution network (GCN) [18] to model label correlation. Two labels are more likely to be correlated if they are provided to the same instance (by different annotators) in the authentic annotations. We calculate the frequency of label co-occurrence in the observed annotations to construct the adjacency matrix $\boldsymbol{A}$. Then we extend the weight matrix $\boldsymbol{M}_c$ in Eq (2) by $\hat{\boldsymbol{M}}_c = \hat{\boldsymbol{D}}^{-\frac{1}{2}} \hat{\boldsymbol{A}} \hat{\boldsymbol{D}}^{-\frac{1}{2}} \boldsymbol{M}_c \boldsymbol{W}$, with $\hat{\boldsymbol{A}} = \boldsymbol{A} + \boldsymbol{I}$ where $\boldsymbol{I}$ is the identity matrix, $\hat{D}$ is the diagonal node degree matrix of $\hat{\boldsymbol{A}}$. We name this component as the label correlation aggregation (LCA) decoder. We also enforce sparsity on the discriminator by applying L2 norm to its outputs.

To realize our second criterion, an auxiliary network Q is used to measure the mutual information between the classifier's prediction $\hat{\boldsymbol{z}}_n$ and the generated annotation y_n^r on instance $\boldsymbol{x}_n$. To simplify our notations in the subsequent discussions, we denote $G(\boldsymbol{x}_n, \boldsymbol{e}_r, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n)$ as $G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}})$ to represent the annotation distribution predicted on pair $(\boldsymbol{x}_n, \boldsymbol{e}_r)$. As our generator design is very flexible to model complex confusions, it however becomes useless for classifier training if the learnt confusions are independent from the classifier's outputs. For example, if the generator learnt to generate a particular annotation only by the annotator's features (e.g., the most frequently observed label in this annotator), such a generation contributes no information to classifier training. We propose to penalize such generations by maximizing the mutual information between classifier's output and the generated annotations for an instance, i.e., $I(\hat{\boldsymbol{z}}; G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}))$.

In practice, mutual information is generally difficult to optimize, because it requires the knowledge of posterior $P(\hat{z}|y)$. We follow the design in [7] to maximize the variational lower bound of $I(\hat{\boldsymbol{z}}; G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}))$ by utilizing an auxiliary distribution P_Q to approximate $P(\hat{z}|y)$:

$$\begin{aligned}\mathcal{L}_I(G, Q) &= \mathbb{E}_{\hat{z}\sim P(\hat{z}), y\sim G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}})}[\log P_Q(\hat{z}|y)] + H(\hat{z}) \\ &\leq I(\hat{\boldsymbol{z}}; G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}})).\end{aligned} \tag{3}$$

We refer to $\mathcal{L}_I$ as the information loss, which can be viewed as an information-theoretical regularization to the original minimax game. The auxiliary distribution $P_Q(\hat{z}|y)$ is parameterized by the auxiliary network Q. In our implementation, we devise a two-step training strategy for the entire pipeline (details in Section 3.3.3), where we fix the classifier when updating the generator. As a result, $H(\hat{z})$ becomes a constant when updating the generator by Eq (3). Since the posterior $P(\hat{z}|y)$ can be different when the annotations are given by different annotators on different instances, we also provide the instance and annotator features to the auxiliary network,

$$P_{\theta_Q}(\hat{z}_n = c|\boldsymbol{x}_n, \boldsymbol{e}_r, y_n^r) \propto \exp[h(\hat{z}_n = c, \boldsymbol{x}_n, \boldsymbol{e}_r, y_n^r)],$$

where $h(\cdot)$ is a learnable scoring function. To reduce model complexity, we reuse the annotator and instance encoding layers from the discriminator here. The class-related weight matrix $\hat{\boldsymbol{M}}_c$ is flatten and transformed to a low-dimension embedding $\boldsymbol{m}_c$ by an embedding layer for each annotation type $y_n^r = c$.

Putting the generative and discriminative modules together, we formalize the value function of our minimax game for learning from crowds in CrowdInG as,

$$\min_{C,G,Q} \max_{D} V_{\text{CrowdInG}}(C, G, D, Q) = V(C, G, D) - \lambda \mathcal{L}_I(G, Q) \tag{4}$$

$$V(C, G, D) = \mathbb{E}_{y\sim P_{\text{true}}}[\log(D(y))] + \mathbb{E}_{\boldsymbol{\varepsilon}\sim P_{\text{noise}}, y\sim P_{G(\boldsymbol{\varepsilon}, \hat{\boldsymbol{z}})}}[\log(1 - D(y))],$$

where λ is a hyper-parameter to control the regularization. The value function is maximized by updating the discriminator to improve its ability in differentiating the authentic annotations from the generated ones, and minimized by updating the classifier, generator and the auxiliary network to generate more high-quality annotations.

3.3 Model optimization

In this section, we introduce the training strategy for CrowdInG, which cannot be simply performed via vanilla end-to-end training. First, the number of unobserved annotator-instance pairs is much larger than the observed ones. Blindly using all the generated annotations overwhelms the training of our discriminative module, and simply leads to trivial solutions (e.g., classifying all annotations as generated). As our solution, we present an entropy-based annotation selection strategy to select informative annotations for discriminative module update. Second, due to the required sampling procedure when generating the annotations, there are non-differentiable steps in our generative module. We resort to an effective counterfactual risk minimization (CRM) method to address the difficulty. Finally, the classifier and the generator in the generative module might change dramatically to fit the complex training signals, which can easily cause model collapse. We propose a two-step training strategy to prevent it in practice.

3.3.1 Entropy-based annotation selection. We borrow the idea from active learning [28]: *the discriminator should learn to distinguish the most difficult annotations.* A generated annotation is more difficult for the discriminator if the generator is more confident about it. Formally, the selection strategy is designed as,

$$P(y_n^r) \propto \frac{1}{H(G(\boldsymbol{x_n}, \boldsymbol{e_r}, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n))},$$

where $H(G(\boldsymbol{x_n}, \boldsymbol{e_r}, \boldsymbol{\varepsilon}, \hat{\boldsymbol{z}}_n))$ is the entropy of the annotation distribution. To reduce training bias caused by annotation sparsity in individual annotators, we sample the same number of generated annotations as the authentic ones in each annotator. As a by-product, our instance selection also greatly reduces the size of training data for the discriminative module. It makes discriminator training a lot more efficient. To fully utilize the power of discriminative module, we use all generated annotations for the generator updating.

3.3.2 Gradient-based optimization. The gradient for the discriminator and the auxiliary network is easy to compute by calculating the derivative on trainable parameters. However, due to the required sampling steps for generating specific annotations, there are non-differentiable steps in the generative module. Previous works [34, 36] use Gumbel-softmax trick or policy gradient to handle the non-differentiable functions. However, once the generator is updated, we need to re-sample the annotations and evaluate them again using the discriminative module, which is time-consuming. To accelerate our model training, we perform batch learning from logged bandit feedback [16, 31]. In each epoch, we treat the generative module from the last epoch as the logging policy G_0, and sample annotations from it. Because the discriminator only evaluates the sampled annotations from the (last) generative module, rather than the entire distribution of annotations predicted by the module, training signals received on the generative module side are in the form of logged bandit feedback.

When updating the generator, the training signals are from both the discriminator $\mathcal{L}_G = \log\big(1 - D(y)\big)$ and the information loss $-\lambda\mathcal{L}_I$. We collectively denote them as loss $\delta = \mathcal{L}_G - \lambda\mathcal{L}_I$. In each epoch, we update the generator G_{θ_G} as follows,

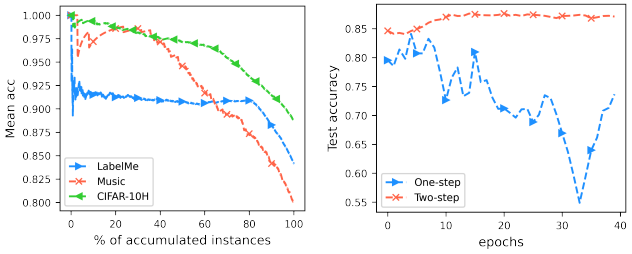
$$\theta_G = \underset{\theta_G}{\operatorname{argmin}} \frac{1}{NR} \sum_{n=1}^{N} \sum_{r=1}^{R} \frac{(\delta(y_n^r) - \mu) G_{\theta_G}(y_n^r)}{G_0(y_n^r)}, \tag{5}$$

where μ is a Lagrange multiplier introduced to avoid overfitting to the logging policy [16]. The optimization of Eq (5) can be easily solved by gradient descent. When updating the classifier, we only use the discriminator's signals. Intuitively, even though annotations should contain the information about the true labels, the inverse is not necessary. The classifier is updated in a similar fashion,

$$\theta_C = \underset{\theta_C}{\operatorname{argmin}} \frac{1}{NR} \sum_{n=1}^{N} \sum_{r=1}^{R} \frac{(\mathcal{L}_G(y_n^r) - \mu) G_{\theta_C}(y_n^r)}{G_0(y_n^r)}. \tag{6}$$

We follow the suggestions in [16] to search the best μ in practice.

3.3.3 Two-step update for the generative module. The generative process is controlled by the generator and the classifier. However, the coupling between the two components introduces difficulties in the estimation of them. For example, one component might overfit a particular pattern in the discriminative signal, and cause model collapse in the entire pipeline. In our empirical studies reported in Figure 2(b), we observed test accuracy fluctuated a lot when we simply used gradients calculated by Eq (5) and (6) to update these two components together. Details of our experiment setup can be found in Section 4.



(a) (b)

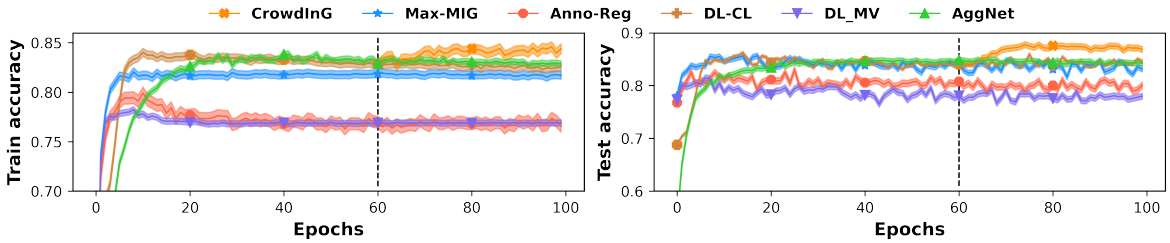
Figure 2: Performance of two-step strategy. (a) Mean accuracy of accumulated instances with ascending order of entropy on three real-world datasets. (b) Comparison between one-step and two-step strategy on LabelMe dataset.

Based on this finding, we adopt a two-step strategy to update the generator and the classifier alternatively. First, we found that the principle behind our annotation selection also applied to our classifier: the entropy of the classifier's output strongly correlates with its accuracy. According to Figure 2(a), the classifier obtains higher accuracy on instances with lower prediction entropy. Therefore, we decided to use the instances with low classification entropy to update the generator by Eq (5), as there the classifier's predictions are more likely to be accurate. Then, we use the updated generator on the rest of instances to update the classifier by Eq (6), where the classifier still has a high uncertainty to handle them.

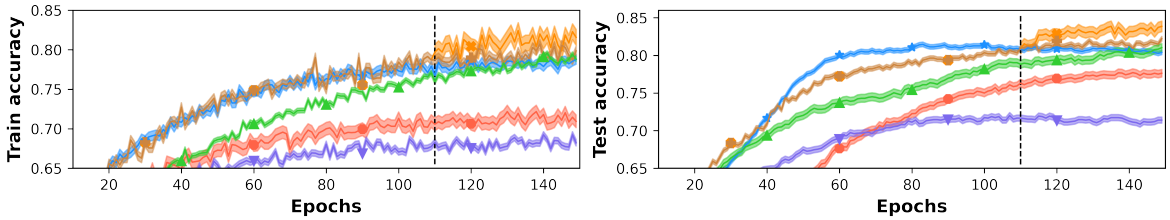
A threshold t is pre-selected to separate the instances; and we will discuss its influence on model training in Section 4.5. Besides, to make the entire training process stable, we pre-train the classifier with the observed annotations using neural crowdsourcing algorithm proposed in [25], which is included as one of our baselines. With the initialized classifier, we also pre-train the generator and discriminator to provide good initialization of these components.

4 EXPERIMENTS

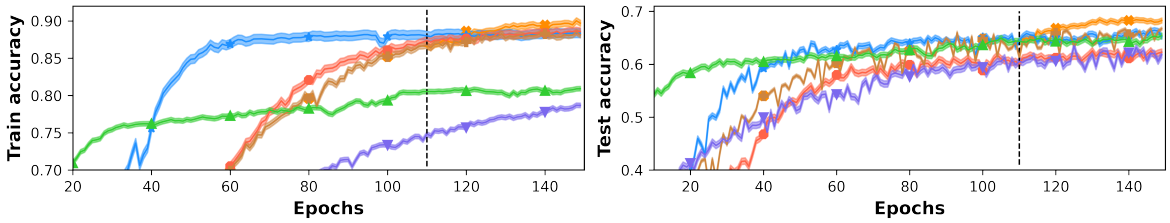
In this section, we evaluate our proposed solution framework on three real-world datasets. The annotations were originally collected from Amazon Mechanical Turk (AMT) by the dataset creators. We compared with a rich set of state-of-the-art crowdsourcing algorithms that estimate the classifiers only with observed annotations. We are particularly interested in investigating *how much human labor can be saved by our data augmentation solution?* We gradually removed an increasing number of annotations and compared with baselines. The result suggests significant annotation cost can be reduced with our generated annotations, while still maintaining the quality of the learnt classifier. Besides, since our model is the first effort to augment crowdsourced data for classifier training, we compared with models trained with annotations from other generative models for crowdsourced data. Finally, we performed extensive ablation analysis about our proposed model components and hyper-parameters to better understand the model's behavior.



(a) Results on LabelMe dataset.



(b) Results on Music dataset.

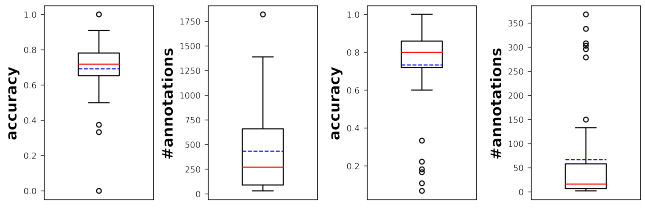


(c) Results on CIFAR-10H dataset.

Figure 3: Results on three real-world datasets. Full CrowdInG training is applied after the dashed line.

4.1 Datasets & Implementation details

We employed three real-world datasets for evaluations. **LabelMe** [25, 27] is an image classification dataset, which consists of 2,688 images from 8 classes, e.g., *inside city*, *street*, *forest*, etc. 1,000 of them are labeled by 59 AMT annotators and the rest are used for validation and testing. Each image is labeled by 2.5 annotators on average. To enrich the training set, standard data augmentation techniques are applied on the training set, including horizontal flips, rescaling and shearing, following the setting in [25]. We created 10,000 images for training eventually. **Music** [26] is a music genre classification dataset, which consists of 1,000 samples of songs with 30 seconds in length from 10 classes, e.g., *classical*, *country*, *jazz*, etc. 700 of them are labeled by 44 AMT annotators and the rest are left for testing. Each sample is labeled by 4.2 annotators on average. Figure 4 shows several important statistics of these two datasets. Specifically, we report the annotation accuracy and the number of annotations among the annotators. Both statistics vary considerably across annotators in these two datasets, which cause serious difficulties in classical crowdsourcing algorithms. **CIFAR-10H** [23] is another image classification dataset, which consists of 10,000 images from 10 classes, e.g., *airplane*, *bird*, *cat*, etc., collected from the CIFAR-10 image dataset [19]. There were 2,571 annotators recruited and each annotator was asked to label 200 images. However, such large-scale annotations are typically expensive and rare in practice. To make this dataset closer to a realistic and challenging setting, we only selected a subset of low-quality annotators. The modified dataset has 8,687 images annotated by 103 AMT annotators. Each annotator still has 200 annotations with an average accuracy of 78.2%; and each image has 2.37 annotations on average. The original 10,000 images validation set of CIFAR-10 is used as our testing set.



(a) LabelMe (b) Music

Figure 4: Boxplots for the number of annotations and the accuracy of the AMT annotators for two real-world crowdsourcing datasets.

To make the comparisons fair, all evaluated methods used the same classifier design (in both CrowdInG and baselines). On the LabelMe dataset, we adopted the same setting as in [25]: we applied a pre-trained VGG-16 network followed by a fully connected

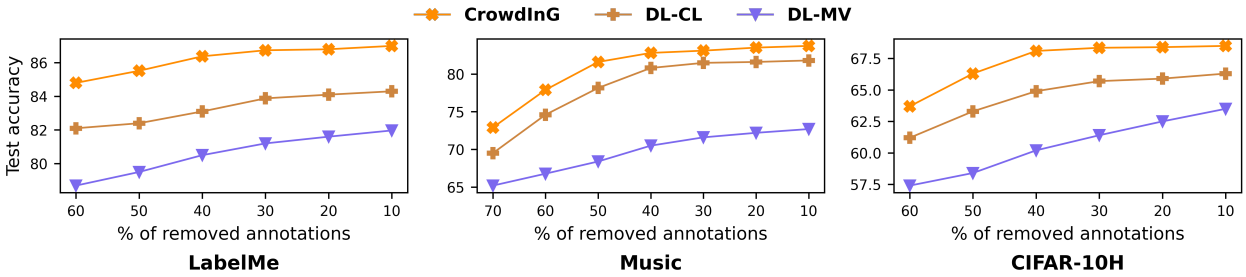


Figure 5: Test accuracy with various proportion of removed annotations.

(FC) layer with 128 units and ReLU activations, and a softmax output layer, using 50% dropout. On the Music dataset, we also used a 128 units FC layer and softmax output layer. Batch normalization was performed in each layer. We disabled LCA on Music since there is no meaningful label correlation patterns. On the CIFAR-10H dataset, we used a VGG-16 network for the classifier. We used Adam optimizer with learning rates searched from $\{3.0 \times 10^4, 2.0 \times 10^4, 1.0 \times 10^4, 1.0 \times 10^5\}$ for both generative and discriminative modules. Scoring functions $g(\cdot)$ and $h(\cdot)$ are implemented by two-layer neural networks with 64 and 128 hidden units. In each epoch, we update the generative and discriminative modules for 5 times. With pre-training, we execute the training procedures for CrowdInG in the last 40 epochs. All experiments are repeated 5 times with different random seeds, and mean accuracy and standard derivation are reported.

Table 1: Test accuracy of different augmentation methods.

	LabelMe	Music	CIFAR-10H
Doctor Net	82.12±0.43	75.41±0.42	67.23±0.54
DL-CL$_{+\text{Self}}$	85.24±0.51	82.56±0.49	64.94±0.84
DL-CL$_{+\text{GCN}}$	82.74±0.34	81.42±0.74	65.02±0.61
DL-CL$_{+\text{GAN}}$	85.16±0.26	83.17±0.48	65.34±0.32
DL-CL$_{+\text{InG}}$	85.42±0.57	83.38±0.59	66.17±0.35
CrowdInG	**87.03**±0.55	**83.73**±0.62	**68.85**±0.47

4.2 Classification performance

4.2.1 Baselines. We compared with a rich set of state-of-the-art baselines, which we briefly introduce here. **DL-MV**: annotations are first aggregated by majority vote, and then it trains a classifier based on the aggregated labels. **DL-CL** [25]: a set of designated layers that capture annotators' confusions (the so-called Crowd Layer) are connected to the classifier, aiming to transform the predicted classifier's outputs to annotation distributions. **Anno-Reg** [32]: trace regularization on confusion matrices is applied to improve the confusion estimation. **Max-MIG** [5]: a neural classifier and a label aggregation network are jointly trained using an information-theoretical loss function, correlated confusions among annotators are captured. **AggNet** [1]: an EM-based deep model considering annotator sensitivity and specificity.

4.2.2 Results & analysis. The classification accuracy of the learnt classifiers from different models on the three datasets are reported in Figure 3. Two things we should highlight: 1) as all models are learnt from crowdsourced data, the ground-truth labels on instances are *unrevealed* to them in training. Therefore, a classifier's accuracy on training set is still a meaningful performance metric. 2) CrowdInG starts with the same classifier as obtained in DL-CL (as we used DL-CL to pre-train our classifier). On all datasets, we observe that even though DL-CL did not outperform the other baselines, after the training in CrowdInG starts, the classifier's performance got significantly improved. This proves the utility of our generated annotations for classifier training. Besides, we also looked into the accuracy in individual classes and found by generating more annotations, CrowdInG's performance on those easily confused classes got more improvement than the baselines. For example, for the class of *open country* on LabelMe, the original annotation accuracy was only 51.5%. DL-CL achieved 49.6% (i.e., the starting point of CrowdInG), and it was improved to 58.9% after CrowdInG training. Compared with models that are designed for complex confusions, such as Max-MIG and AggNet, CrowdInG still outperformed them with a large margin. This indicates our generator has a stronger advantage in capturing complex confusions.

4.3 Utility of augmented annotations

4.3.1 Experiment setup. We study the utility of augmented annotations from CrowdInG. On each dataset, we gradually removed an increasing number of observed annotations to investigate how different models' performance changes. We ensure that each instance has at least one annotation, such that we will only remove annotations rather than instances for classifier training. We compared with two representative baselines: 1) DL-MV, a typical majority-vote-based method, and 2) DL-CL, a typical DS-model-based method, to study their sensitivity on the sparsity of annotations.

4.3.2 Results & analysis. We present the results in Figure 5. All models suffered from extreme sparsity when we removed a large portion of annotations (e.g., 60%), but CrowdInG still enjoyed a consistent improvement against all baselines. DL-MV performed the worst, because with less redundant annotations, the quality of its aggregated labels deteriorated seriously. When we looked into the detailed model update trace of CrowdInG, we found that the performance gain became larger after CrowdInG training. Again, because we used the classifier obtained from DL-CL as our starting point for CrowdInG, low-quality annotations were generated at the beginning of CrowdInG update. However, CrowdInG quickly improved once its discriminative module started to penalize those low-quality annotations. The results strongly support that a great

deal of human labor can be saved. On LabelMe and CIFAR-10H, CrowdInG performed closely to the baselines' best performance even with 60% less annotations. Even on the most difficult dataset Music, about 10% annotations can be saved by CrowdInG to achieve similar performance as DL-CL.

4.4 Comparison with other augmentations

4.4.1 Baselines. As no existing method explicitly performs data augmentation for crowdsourced data, we consider several alternative data augmentation methods using various self-training or generative modeling techniques. Arguably, any generative model for crowdsourced data can be used for this purpose.

In particular, we chose the following baselines. **Doctor Net** [12]: each annotator is modeled by an individual neural network. When testing, annotations are predicted by annotator networks and then aggregated by weighted majority vote. **DL-CL$_{+\text{Self}}$**: we complete the missing annotations using a pre-trained DL-CL model, and then train another DL-CL model based on the completed annotations. **DL-CL$_{+\text{GCN}}$**: we construct an annotator-instance bipartite graph based on the observed annotations, and fill in the missing links using a Graph Convolution Network (GCN) [3, 18]. Then we train a DL-CL model using the expanded annotations. **DL-CL$_{+\text{GAN}}$**: we follow the same design in [35], which unifies generative and discriminative models into a GAN framework. We use DL-CL as the generative model. **DL-CL$_{+\text{InG}}$**: we directly train a DL-CL model on the expanded dataset provided by CrowdInG.

4.4.2 Results & analysis. We present the test accuracy on all three datasets in Table 1. Doctor Net trains individual classifiers for each annotator, so that on datasets where annotations from each annotator are sufficient, such as CIFAR-10H, this model obtained satisfactory performance with the generated annotations. But on the other datasets where annotations are sparse in each annotator, its performance dropped a lot. In DL-CL type methods, the performance is generally improved. However, due to the simple class-dependent confusion assumption, such models' capacity to capture complex confusions is limited. As a result, even though GCN could capture more complex annotator-instance interactions, DL-CL still failed to benefit from it in DL-CL$_{+\text{GCN}}$. The added discriminator in DL-CL$_{+\text{GAN}}$ improved the performance; however, DL-CL still could not fully utilize the complex discriminative signals and failed to further improve the performance. DL-CL$_{+\text{InG}}$ performed better than the other baselines by directly using the annotations generated by CrowdInG, which suggests the annotations generated under our criteria are generically helpful for other crowdsouring algorithms.

Table 2: Test accuracy of different variants of CrowdInG

	LabelMe	Music	CIFAR-10H
CrowdG	85.89±0.47	83.14±0.28	66.15±0.34
CrowdInG$_{\text{U}}$	83.12±0.39	81.28±0.51	67.12±0.59
CrowdInG$_{\text{I}}$	84.34±0.72	82.24±0.47	66.90±0.31
CrowdInG$_{\text{R}}$	86.17±0.44	82.74±0.58	67.88±0.62
CrowdInG	**87.03**±0.55	**83.73**±0.62	**68.85**±0.47

4.5 Ablation study

4.5.1 Analysis of different components in CrowdInG. To show the contributions of different components in CrowdInG, we varied the setting of our solution. We already showed the one-step training variant in Figure 2, which suffered from serious model collapsing. To investigate the other components, we created the following variants. **CrowdG**: the information loss defined in Eq (3) is removed. **CrowdInG$_{\text{U}}$**: the generator only considers classifier's outputs, annotator features and random noise, but not the instance features. **CrowdInG$_{\text{I}}$**: the generator only considers classifier's outputs, instance features and random noise, but not the annotator features. **CrowdInG$_{\text{R}}$**: the annotation selection is kept, but instead we randomly select an equal number of generated annotations as the authentic ones for discriminator update.

We reported the test accuracy on three datasets in Table 2. By maximizing the mutual information, CrowdInG outperformed CrowdG with a considerable margin. We further investigated the generated annotations and found the annotations generated by CrowdG were more random, which could not be easily linked to the classifier's output. CrowdInG$_{\text{U}}$ performed poorly when the number of annotations per annotator was limited, such as on LabelMe and Music datasets, but worked better when annotations per annotator are adequate, such as on CIFAR-10H. This again proves more annotations are needed to better model annotators' confusions. CrowdInG$_{\text{I}}$ performed better because by taking instance features, the generator can model more complicated confusions with respect to instance features. CrowdInG$_{\text{R}}$ bypassed the data imbalance issue; but without focusing on difficult annotations, it still cannot fully unleash the potential of generated annotations.

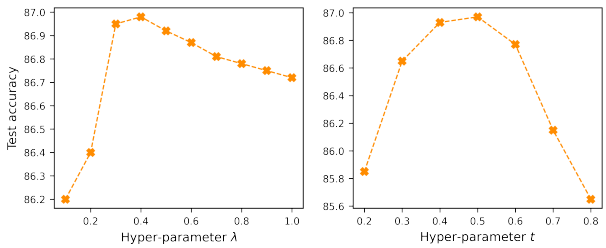


Figure 6: Performance under different hyper-parameter settings on LabelMe dataset.

4.5.2 Hyper-parameter analysis. We studied the sensitivity of hyper-parameters λ and t in CrowdInG. Specifically, λ controls the degree of the information regularization in Eq (4), we varied it from 0.1 to 1. t controls the grouping of instances used for classifier update; and we varied it from 0.2 to 0.8. Due to space limit, we only report the results on LabelMe, similar observations were also obtained on the other two datasets.

The model's performance under different hyper-parameter settings is illustrated in Figure 6. We can clearly observe that the performance is boosted when appropriate hyper-parameters are chosen. Small λ poses weak information regularization to the generator, and thus the generated annotations are less informative for classifier training. Large λ slightly hurts the performance because strong regularization weakens the ability of the generator

to capture complex confusions related to instance and annotator features. We can observe similar trend on t. To avoid model collapse, a moderate t is needed to restrict the classifier training, but a large t will hurt the performance more. Because with a large t, very few instances will be selected for classifier training, so that the classifier can hardly be updated.

5 CONCLUSIONS & FUTURE WORKS

Data sparsity poses a serious challenge to current learning from crowds solutions. We present a data augmentation solution using generative adversarial networks to handle the issue. We proposed two important principles in generating high-quality annotations: 1) the generated annotations should follow the distribution of authentic ones; and 2) the generated annotations should have high mutual information with the ground-truth labels. We implemented these principles in our discriminative model design. Extensive experiment results demonstrated the effectiveness of our data augmentation solution in improving the performance of the classifier learned from crowds, and it sheds light on our solution's potential in low-budget crowdsourcing in general.

Our exploration also opens up a series interesting future directions. As our generative module captures annotator- and instance-specific confusions, it can be used for annotator education [29], e.g., inform individual annotators about their potential confusions. Our solution can also be used for interactive labeling with annotators [39], e.g., only acquire annotations on which our generative module currently has a low confidence. Also, the instance-level confusion modeling can better support fine-grained task assignment [10], e.g., gather senior annotators for specific tasks.

ACKNOWLEDGEMENT

This work was partially supported by the National Science Foundation under award NSF IIS-1718216 and NSF IIS-1553568, and the Department of Energy under the award DoE-EE0008227.

REFERENCES

[1] Shadi Albarqouni, Christoph Baur, Felix Achilles, Vasileios Belagiannis, Stefanie Demirci, and Nassir Navab. 2016. Aggnet: deep learning from crowds for mitosis detection in breast cancer histology images. *IEEE transactions on medical imaging* 35, 5 (2016), 1313–1321.
[2] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340* (2017).
[3] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2017. Graph convolutional matrix completion. *arXiv preprint arXiv:1706.02263* (2017).
[4] Thierry Buecheler, Jan Henrik Sieg, Rudolf Marcel Füchslin, and Rolf Pfeifer. 2010. Crowdsourcing, open innovation and collective intelligence in the scientific method: a research agenda and operational framework. In *The 12th International Conference on the Synthesis and Simulation of Living Systems, Odense, Denmark, 19-23 August 2010*. MIT Press, 679–686.
[5] Peng Cao, Yilun Xu, Yuqing Kong, and Yizhou Wang. 2019. Max-MIG: an Information Theoretic Approach for Joint Learning from Crowds. In *ICLR*. https://openreview.net/forum?id=BJg9DoR9t7
[6] Dong-Kyu Chae, Jin-Soo Kang, Sang-Wook Kim, and Jung-Tae Lee. 2018. Cfgan: A generic collaborative filtering framework based on generative adversarial networks. In *Proceedings of the 27th ACM CIKM*. 137–146.
[7] Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657* (2016).
[8] Zhendong Chu, Jing Ma, and Hongning Wang. 2020. Learning from Crowds by Modeling Common Confusions. *arXiv preprint arXiv:2012.13052* (2020).
[9] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 28, 1 (1979), 20–28.
[10] Jia Deng, Jonathan Krause, and Li Fei-Fei. 2013. Fine-grained crowdsourcing for fine-grained recognition. In *CVPR*. 580–587.
[11] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial networks. *arXiv preprint arXiv:1406.2661* (2014).
[12] Melody Guan, Varun Gulshan, Andrew Dai, and Geoffrey Hinton. 2018. Who said what: Modeling individual labelers improves classification. In *AAAI*, Vol. 32.
[13] Hrayr Harutyunyan, Kyle Reing, Greg Ver Steeg, and Aram Galstyan. 2020. Improving generalization by controlling label-noise information in neural network weights. In *International Conference on Machine Learning*. PMLR, 4071–4081.
[14] Hideaki Imamura, Issei Sato, and Masashi Sugiyama. 2018. Analysis of minimax error rate for crowdsourcing and its application to worker clustering model. In *International Conference on Machine Learning*. PMLR, 2147–2156.
[15] Athirai A Irissappane, Hanfei Yu, Yankun Shen, Anubha Agrawal, and Gray Stanton. 2020. Leveraging GPT-2 for Classifying Spam Reviews with Limited Labeled Data via Adversarial Training. *arXiv preprint arXiv:2012.13400* (2020).
[16] Thorsten Joachims, Adith Swaminathan, and Maarten de Rijke. 2018. Deep learning with logged bandit feedback. In *ICLR*.
[17] Ece Kamar, Ashish Kapoor, and Eric Horvitz. 2015. Identifying and accounting for task-dependent bias in crowdsourcing. In *Proceedings of the AAAI Conference on Human Computation and Crowdsourcing*, Vol. 3.
[18] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).
[19] Alex Krizhevsky, Geoffrey Hinton, et al. 2009. Learning multiple layers of features from tiny images. (2009).
[20] Jack Lanchantin, Arshdeep Sekhon, and Yanjun Qi. 2019. Neural message passing for multi-label classification. In *ECML-PKDD*. Springer, 138–163.
[21] Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael J Franklin. 2016. Crowdsourced data management: A survey. *IEEE TKDE* 28, 9 (2016), 2296–2319.
[22] Augustus Odena. 2016. Semi-supervised learning with generative adversarial networks. *arXiv preprint arXiv:1606.01583* (2016).
[23] Joshua C Peterson, Ruairidh M Battleday, Thomas L Griffiths, and Olga Russakovsky. 2019. Human uncertainty makes classification more robust. In *Proceedings of the IEEE International Conference on Computer Vision*. 9617–9626.
[24] Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. 2010. Learning from crowds. *Journal of Machine Learning Research* 11, 4 (2010).
[25] Filipe Rodrigues and Francisco Pereira. 2018. Deep learning from crowds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.
[26] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Gaussian process classification and active learning with multiple annotators. In *International conference on machine learning*. PMLR, 433–441.
[27] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. 2008. LabelMe: a database and web-based tool for image annotation. *International journal of computer vision* 77, 1-3 (2008), 157–173.
[28] Burr Settles. 2009. Active learning literature survey. (2009).
[29] Adish Singla, Ilija Bogunovic, Gábor Bartók, Amin Karbasi, and Andreas Krause. 2014. Near-optimally teaching the crowd to classify. In *ICML*. PMLR, 154–162.
[30] Jost Tobias Springenberg. 2015. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390* (2015).
[31] Adith Swaminathan and Thorsten Joachims. 2015. Batch learning from logged bandit feedback through counterfactual risk minimization. *The Journal of Machine Learning Research* 16, 1 (2015), 1731–1755.
[32] Ryutaro Tanno, Ardavan Saeedi, Swami Sankaranarayanan, Daniel C Alexander, and Nathan Silberman. 2019. Learning from noisy labels by regularized estimation of annotator confusion. In *CVPR*. 11244–11253.
[33] Matteo Venanzi, John Guiver, Gabriella Kazai, Pushmeet Kohli, and Milad Shokouhi. 2014. Community-based bayesian aggregation models for crowdsourcing. In *Proceedings of the 23rd international conference on World wide web*. 155–164.
[34] Hongwei Wang, Jia Wang, Jialin Wang, Miao Zhao, Weinan Zhang, Fuzheng Zhang, Xing Xie, and Minyi Guo. 2018. Graphgan: Graph representation learning with generative adversarial nets. In *AAAI*, Vol. 32.
[35] Jun Wang, Lantao Yu, Weinan Zhang, Yu Gong, Yinghui Xu, Benyou Wang, Peng Zhang, and Dell Zhang. 2017. Irgan: A minimax game for unifying generative and discriminative information retrieval models. In *Proceedings of the 40th International ACM SIGIR conference*. 515–524.
[36] Qinyong Wang, Hongzhi Yin, Hao Wang, Quoc Viet Hung Nguyen, Zi Huang, and Lizhen Cui. 2019. Enhancing collaborative filtering with generative augmentation. In *Proceedings of the 25th ACM SIGKDD conference*. 548–556.
[37] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier Movellan, and Paul Ruvolo. 2009. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. *NeurIPS* 22 (2009), 2035–2043.
[38] Yilun Xu, Peng Cao, Yuqing Kong, and Yizhou Wang. 2019. L_DMI: A Novel Information-theoretic Loss Function for Training Deep Nets Robust to Label Noise.. In *NeurIPS*. 6222–6233.
[39] Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. 2011. Active learning from crowds. In *International Conference on Machine Learning*.