Extending Label Aggregation Models with a Gaussian Process to Denoise Crowdsourcing Labels

Dan Li
University of Amsterdam & Elsevier
Amsterdam, The Netherlands
d.li1@elsevier.com

Maarten de Rijke
University of Amsterdam
Amsterdam, The Netherlands
m.derijke@uva.nl

ABSTRACT

Label aggregation (LA) is the task of inferring a high-quality label for an example from multiple noisy labels generated by either human annotators or model predictions. Existing work on LA assumes a label generation process and designs a probabilistic graphical model (PGM) to learn latent true labels from observed crowd labels. However, the performance of PGM-based LA models is easily affected by the noise of crowd labels. As a consequence, the performance of LA models differs on different datasets and no single LA model outperforms the others on all datasets.

We extend PGM-based LA models by integrating a Gaussian process (GP) prior on the true labels. The advantage of LA models extended with a GP prior is that they can take as input crowd labels, example features, and existing pre-trained label prediction models to infer the true labels, while the original LA can only leverage crowd labels. Experimental results on both synthetic and real datasets show that any LA model extended with a GP prior and a suitable mean function achieves better performance than the underlying LA model, demonstrating the effectiveness of using a GP prior.

CCS CONCEPTS

• **Information systems → Information retrieval**; • **Computing methodologies → Bayesian network models**.

KEYWORDS

Label aggregation, Crowdsourcing

ACM Reference Format:
Dan Li and Maarten de Rijke. 2023. Extending Label Aggregation Models with a Gaussian Process to Denoise Crowdsourcing Labels. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '23), July 23–27, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3539618.3591685

1 INTRODUCTION

Crowdsourcing has been widely adopted as a time-efficient and cost-effective solution for dataset construction [4, 9, 14, 28, 33, 35, 37, 44]. Typically, *examples* in a dataset, e.g., query-document pairs that need to be labeled with a relevance label, are distributed on crowdsourcing platforms to multiple *annotators* (or *workers*) to quickly obtain *crowd labels*. Despite its efficiency and effectiveness, crowdsourcing comes with a challenge: noisy labels [16, 19].

Various label aggregation (LA) models have been proposed to infer true labels of examples from noisy crowd labels. Existing work on LA [8, 11, 18, 23, 24, 39, 40, 43] models the true labels and the crowd labels as random variables and use a probabilistic graphical model (PGM) to calculate the probability of generating the crowd labels and the probability of generating the true labels. The *latent true labels* are usually assumed to be generated independently from each other; and given the latent true label of an example, its *observed crowd labels* are also assumed to be generated independently from other crowd labels. Thus, model parameters can easily be learned using expectation maximization (EM). However, PGM-based LA models do not generalize well to different datasets. Relative model performance is easily affected by datasets. Empirical results show that no single PGM-based LA model outperforms all others [24, 43].

We propose to integrate LA models with a Gaussian process (GP) prior to make them generalizable to different datasets. A GP is defined by a covariance function and a mean function: its covariance function takes example features as input to model example correlation, and its mean function can be initialized by a label prediction model that is trained on other datasets and can predict example true labels. We detail extensions with a GP for four representative models: (i) the confusion matrix-based Dawid and Skene model (DS) [6], and the 1-coin models (ii) ZenCrowd (ZC) [7], (iii) generative model of labels, abilities, and difficulties (GLAD) [38], and (iv) multi-annotator competence estimation (MACE) [15]. GP-extensions of other LA models can easily be derived following this paper.

We use *GPLA model* to refer to a LA model that has been extended with a GP prior. The advantage of a GPLA model over its underlying LA model is that it can take as input crowd labels, example features, and existing label prediction models to infer the true labels, while a LA model can only leverage crowd labels. An important challenge of GPLA models is their optimization. The EM algorithm used for LA models cannot be used by GPLA models because of the GP prior. Instead, we use the variational expectation maximization (VEM) [29] algorithm for optimization.

The main contributions of this work are the following:

- We propose a way of extending LA models with a GP prior, which results in Gaussian process-based label aggregation models that can take as input crowd labels, example features, and existing label prediction models to infer the true labels, while the original LA models can only leverage crowd labels.

SIGIR '23, July 23–27, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-9408-6/23/07…$15.00
https://doi.org/10.1145/3539618.3591685

- We empirically demonstrate that any of the GPLA models achieves better label inference performance than all the LA models on different datasets.

2 RELATED WORK AND BACKGROUND

2.1 PGMs for label aggregation

Mainstream PGM-based LA models model the label generation process using a joint distribution of the observed crowd labels $\boldsymbol{Y}$ and the latent true labels $\boldsymbol{z}$, defined as $p(Y, z) = \prod_{i=1}^{N} p(z_i) \prod_{j=1}^{M_i} p(y_i^j \mid z_i)$. N is the number of examples, M_i is the number of annotators for example i, z_i is the true label of example i, and y_i^j is the observed crowd label given by annotator j. The decomposition of $p(z_i)$ over i and $p(y_i^j \mid z_i)$ over j is based on the *independence assumption*. The likelihood of the observed labels is $p(Y) = \sum_z p(Y, z)p(z)$. Model parameters can be learned using the EM algorithm [26]. All LA models use a categorical distribution to model $p(z_i)$; the main difference between them is how they model $p(y_i^j \mid z_i)$, i.e., what assumption they make about the *label generation process*, based on which they can be classified into confusion matrix-based models such as DS [6], and 1-coin models such as ZC [7], GLAD [38], and MACE [15].

The DS model assumes a parameterized confusion matrix $\lambda_{ol}^j = p(y_i^j = l \mid z_i = o)$ for each annotator j where $l, o \in \{0, 1, \ldots, L-1\}$. L is the number of class. The confusion matrix can be understood as an annotator competence matrix. It contains $M \times L \times (L-1)$ parameters, which needs a large number of crowd labels to learn. The ZC model models $p(y_i^j \mid z_i)$ with a parameter η_j, which determines a Bernoulli variable representing an annotator giving a correct label: $\eta_j = p(y_i^j = z_i)$; η_j can be understood as annotator competence similar to the confusion matrix of the DS model. The GLAD model models $p(y_i^j \mid z_i)$ by a logistic function $\sigma_i^j = (1 + e^{-\alpha_i \beta_j})^{-1}$, where α_i is example difficulty and β_j is annotator competence. σ_i^j represents the probability an annotator gives a correct label. The MACE model models $p(y_i^j \mid z_i)$ using a scalar parameter ϵ_j and a parameterized distribution ξ^j. *Bernoulli*(ϵ_j) determines whether the annotator is *not spamming*. If so, he copies the true label; if not, he randomly picks one label from the distribution *Categorial*(ξ^j). It reduces the number of parameters to $M \times (L-1)$. There are also Bayesian extensions of the above models. E.g., Raykar et al. [31] add Dirichlet priors to the parameters of the DS model, Kim and Ghahramani [20] propose a full Bayesian model of DS and use Gibbs sampling for parameter optimization.

There is also work that learns a downstream classification model using the crowd labels. Cao et al. [3] work on medical image class annotation. They observe that annotators from the same group (a hospital in their setting) makes highly correlated mistakes and modeling annotator weights helps. They jointly learn a classification model and an annotator weighting model. Albarqouni et al. [1] handle label aggregation directly as part of the learning process of the convolutional neural network via additional crowdsourcing layer. A survey of more work can be found in [18, 24, 43].

2.2 Gaussian processes for label aggregation

GPs have previously been applied to aggregate multiple data [12, 13, 21, 22, 27, 32, 34, 36, 41]. Groot et al. [12] average multiple crowd labels to one single label and apply a vanilla GP regression model for continuous label aggregation. Rodrigues et al. [32] propose a label generation process that consists of a GP prior for latent true labels and multiple Bernoulli variables for annotators; they propose a expectation propagation (EP) algorithm for model optimization. Ruiz et al. [34] propose a different label generation process that also consists of a GP prior but a sensitivity-specificity model for annotators; they propose a variational Bayes (VB) algorithm for model optimization. The model has been extended to large-scale datasets [27]. Li et al. [22] propose a GP-based label aggregation model that consists of a GP to model correlation between examples, Gaussian variables to model examples and to model annotators; they apply the VEM algorithm [29] for optimization.

In this work, we take advantage of the GP prior in modeling label correlation. The main difference with prior work [e.g., 32, 34] is that we utilize both a non-zero mean function and a covariance function while they only utilize a zero mean function and a covariance function. For model optimization, we also use the VEM algorithm similar to Li et al. [22]. Unlike the EP and VB algorithms in [32, 34], which depend on the label generation assumption in the LA models, VEM is agnostic to the label generation assumption and thus is easy to apply to many different LA models.

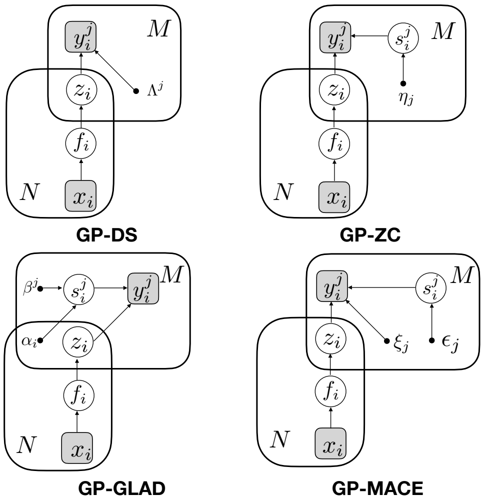
2.3 Gaussian process classification

We recall the Gaussian process classification (GPC) model as it is helpful to understand our proposal of extending LA models with a GP prior; for a thorough introduction, see [30].

Given training samples $(\boldsymbol{X}, \boldsymbol{y}) \triangleq \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{N}$, where N is the number of samples, $\boldsymbol{x}_i$ is a feature vector, y_i is a label. The goal is to predict a label for a new point $\boldsymbol{x}_*$. A GPC model assumes the labels are generated through the following process. First, the latent variable $\boldsymbol{f} \triangleq [f_1, f_2, \ldots, f_N]$ follows a GP, a stochastic process with the important characteristic that any finite number of random variables follow a joint Gaussian distribution. It is denoted by $\boldsymbol{f} \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}'))$, where $m(\boldsymbol{x})$ and $k(\boldsymbol{x}, \boldsymbol{x}')$ are the mean function and covariance function. E.g., $m(\boldsymbol{x})$ is usually a zero function or a linear function, $k(\boldsymbol{x}, \boldsymbol{x}')$ can be the Euclidean distance function. Second, each observed variable y_i follows a Bernoulli distribution conditioned on its corresponding latent variable f_i, and the positive probability is defined using a *probit* function, $p(y_i = 1) = \Phi(f_i) = \int_{-\infty}^{f_i} \mathcal{N}(t \mid 0, 1)\, \mathrm{d}t$.

GPC is optimized to maximize the likelihood. This is not trivial because the likelihood is not a Gaussian distribution due to the multiplication of the Bernoulli likelihood and the GP prior. The solution is to use a parameterized Gaussian distribution $q(\boldsymbol{f} \mid \boldsymbol{\mu}_\psi, \Sigma_\psi)$ to make an evidence lower bound (ELBO) of the likelihood. Given a new point $\boldsymbol{x}_*$, the predictive distribution of the latent variable f_* can be calculated using $p(f_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \int p(f_* \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{f})\, q(\boldsymbol{f} \mid \boldsymbol{\mu}_\psi, \Sigma_\psi)\, \mathrm{d}\boldsymbol{f} = \mathcal{N}(\mu_*, \sigma_*)$, where

$$
\begin{aligned}
\mu_* &= m(\boldsymbol{x}_*) + K_* K^{-1}(\boldsymbol{\mu}_\psi - m(\boldsymbol{X})), \\
\sigma_* &= K_{**} - K_*^T (K^{-1} - K^{-1}\Sigma_\psi K^{-1}) K_*, \\
K &= \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix},
\end{aligned}
\tag{1}
$$



GP-DS

GP-ZC

GP-GLAD

GP-MACE

Figure 1: Graphical model representations of GP-DS, GP-ZC, GP-GLAD, and GP-MACE.

$$K_*^T = [k(\boldsymbol{x}_1, \boldsymbol{x}_*), \ldots, k(\boldsymbol{x}_N, \boldsymbol{x}_*)],$$
$$K_{**} = k(\boldsymbol{x}_*, \boldsymbol{x}_*).$$

The probability of the label to be positive is $p(y_* = 1 \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y}) \approx \Phi(\mathbb{E}(f_*)) = \Phi(\mu_*)$. Following [30, page 44–45], there are two ways of calculating $p(y_* = 1 \mid \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})$: the full Bayesian treatment, *averaged predictive probability* defined as $\int \Phi(f_*) p(f_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})\, \mathrm{d}f_*$, and the simpler *maximum a posteriori (MAP) prediction* defined is $\Phi\left(\int f_* p(f_* | \boldsymbol{x}_*, \boldsymbol{X}, \boldsymbol{y})\, \mathrm{d}f_*\right)$. We will use the MAP prediction because the integral of the first one is difficult to calculate while the second one is easy to calculate.

3 A GAUSSIAN PROCESS FOR LABEL AGGREGATION

3.1 Problem formulation

Different from Gaussian process classification (GPC), where each example in the observed data has one label, there are multiple labels for each example in the crowd annotation data. Assume that there are N examples, M annotators, and L classes in the crowd annotation data. We use $\boldsymbol{X} \triangleq [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N]$ to denote the feature vectors and $Y \triangleq [\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N]$ to denote crowd labels for all the N examples, where $\boldsymbol{y}_i \triangleq [y_i^1, y_i^2, \ldots, y_i^j, \ldots, y_i^{M_i}]$ denotes the multiple labels of the i-th example, M_i is the number of labels of the i-th example. Our goal is to infer the true labels for all examples, which we denote using $\boldsymbol{z} \triangleq [z_1, z_2, \ldots, z_N]$. Note that we work on binary labels, thus $z_i, y_i^j \in \{0, 1\}$ and $L = 2$ in this section.[1]

3.2 Modeling example correlation and example label priors with a Gaussian process

We use a Gaussian process (GP) to model example correlation and the prior knowledge of example true labels. A GP model is formally represented as:

$$\boldsymbol{f} \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')). \tag{2}$$

$\boldsymbol{f} \triangleq [f_1, f_2, \ldots, f_N]$ are continuous random variables which correspond to the N examples and determine N Bernoulli distributions which generate the latent true labels $\boldsymbol{z}$. For example i, the likelihood of obeserving z_i is defined as

$$p(z_i \mid f_i) = \Phi\left((-1)^{(1-z_i)} f_i\right). \tag{3}$$

Note that we work on the binary class problem in this paper. One possible way to extend to the multi-class problem is by using multiple GPs and change the *probit* function $\Phi(\cdot)$ to a *softmax* function.

The covariance function $k(\boldsymbol{x}, \boldsymbol{x}')$ captures the correlation between examples. A higher value indicates that the two examples are more correlated and that they are more likely to have the same latent true label. In this work, we use the radial basis function (RBF), defined as $k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{x}'\|^2}{l^2}\right)$.

The mean function $m(\boldsymbol{x})$ maps a vector to a real value. It can capture the prior information for example true labels. Previous work on GPs often uses a zero mean function, i.e., $m(\boldsymbol{x}) \equiv 0$. This is a convenient choice in terms of modeling but introduces no prior information. There is also the linear mean function, defined as $m(\boldsymbol{x}) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$, where $\boldsymbol{w}$ is the weight parameter to be learned during model optimization. Optionally, we can use a pre-trained label prediction model as the mean function of which the parameters are fixed. It can introduce prior knowledge of example true labels from some similar source domain to the target domain to which the crowd annotation data belong. In this work, we use a logistic regression (LR) model that is trained using example features and labels aggregated by majority voting. Given an example feature as the input, the output logits are used as the mean function value.

3.3 Modeling label generation processes with a probabilistic graphical model

Most PGM-based LA models model the joint distribution of Y and $\boldsymbol{z}$ as $p(Y, \boldsymbol{z}) = \prod_{i=1}^{N} p(z_i) \prod_{j=1}^{M_i} p(y_i^j \mid z_i)$. In this work, we model the joint distribution of Y, $\boldsymbol{z}$, and $\boldsymbol{f}$. $\boldsymbol{f}$ denotes a latent function following a GP, defined in Eq. (2). The generation of $\boldsymbol{z}$ is no longer independent with each other but conditioned on $\boldsymbol{f}$, defined in Eq. (3). The joint distribution is represented as

$$p(Y, \boldsymbol{z}, \boldsymbol{f}) = \\ p(\boldsymbol{f}) p(\boldsymbol{z} \mid \boldsymbol{f}) p(Y \mid \boldsymbol{z}) = p(\boldsymbol{f}) \prod_{i=1}^{N} p(z_i \mid f_i) \prod_{j=1}^{M_i} p(y_i^j \mid z_i), \tag{4}$$

where $p(\boldsymbol{f}) = \mathcal{N}(m(\boldsymbol{X}), k(\boldsymbol{X}, \boldsymbol{X}))$ is a Gaussian distribution modelling example correlation and true label prior information; $m(\boldsymbol{X}) \in \mathbb{R}^N$ is an N-dimensional mean vector, $k(\boldsymbol{X}) \in \mathbb{R}^N \times \mathbb{R}^N$ is a covariance matrix; $p(z_i \mid f_i)$ is defined in Eq. (3) and models the true labels, and $p(y_i^j \mid z_i)$ is defined by different LA models (see Section 3.3.1–3.3.4). The label generation process is formally summarized in Algorithm 1.

3.3.1 GP-DS. The original Dawid and Skene model (DS) [6] is for multi-class tasks. It models $p(z_i)$ by a categorical distribution $p(z_i = o) = \tau_o$. It models $p(y_i^j \mid z_i)$ by a parameterized confusion matrix Λ^j, and its element $\lambda_{ol}^j = p(y_i^j = l \mid z_i = o)$ for each

[1] Throughout the paper, we use lowercase for scalars, lowercase bold for vectors, and bold uppercase for matrices.

Algorithm 1 Label generation process for GPLA models

```
Variable: $Y, z, f$.
1: $f \sim \mathcal{GP}(m(X), k(X, X))$
2: for $i = 1, \ldots, N$ do
3:     $z_i \sim Bernoulli(\Phi(f_i))$
4:     for $j = 1, \ldots, M_i$ do
5:         Sample $y_i^j$ based on the label generation assumption of
           different LA models.
6:     end for
7: end for
```

annotator j, which can be interpreted as an annotator competence matrix, $l, o \in \{0, 1\}$. Formally, $p(y_i^j \mid z_i)$ can be written as:

$$p(y_i^j \mid z_i) = \prod_{o=0}^{1} \prod_{l=0}^{1} {\lambda_{ol}^{j}}^{\mathbb{I}(z_i=o, y_i^j=l)}. \tag{5}$$

We extend the vanilla DS model by adding a latent function $\boldsymbol{f}$ as the prior of $\boldsymbol{z}$. We refer to the model as GP-DS. The label generation process is formally summarized in Algorithm 1. In line 1, a set of random variables $[f_1, \ldots, f_i, \ldots, f_N]$ are sampled from the Gaussian distribution $p(\boldsymbol{f}) = (m(\boldsymbol{X}), k(\boldsymbol{X}, \boldsymbol{X}))$. In line 2–3, for each example i, the latent true label z_i is sampled from $Bernoulli(\Phi(f_i))$. In line 4–5, for each annotator j, the observed label y_i^j is sampled from $Bernoulli(\lambda_{z_i 1}^j)$. Note $\lambda_{z_i}^j$ is in the z_i-th row of $\boldsymbol{\Lambda}^j$ and $\lambda_{z_i 0}^j + \lambda_{z_i 1}^j = 1$.

For example, suppose we get a sample $[f_1, \ldots, f_i, \ldots, f_N]$. Let us focus on f_1 and assume $f_1 = 1.96$, thus $\Phi(f_1) = 0.97$. Then we sample z_1 from $Bernoulli(0.97)$, let us assume $z_1 = 1$ because it has a chance of 97% to be 1. Finally, for annotator j, we know the annotator's confusion matrix is $\boldsymbol{\Lambda}^j$. We will sample $y_1^j = 0$ with a probability of λ_{10}^j and $y_1^j = 1$ with a probability of λ_{11}^j.

3.3.2 GP-ZC. The original ZC model [7] models $p(z_i)$ by a categorical distribution $p(z_i = o) = \tau_o$. It models $p(y_i^j \mid z_i)$ by a parameter η_j, which can be understood as annotator competence similar with the confusion matrix of DS. But the parameters of ZC are reduced to M (compared to $M \times 2 \times 1$ in DS). Formally, $p(y_i^j \mid z_i)$ can be written as:

$$p(y_i^j \mid z_i) = {\eta_j}^{\mathbb{I}\left(z_i = y_i^j\right)} (1 - \eta_j)^{\mathbb{I}\left(z_i \neq y_i^j\right)}. \tag{6}$$

Similar to GP-DS, we extend it by adding a latent function $\boldsymbol{f}$ as the prior of $\boldsymbol{z}$. We name the model GP-ZC. In line 4–5 of Algorithm 1, for each annotator j, s_i^j is sampled from $Bernoulli(\eta_j)$. The variable $s_i^j = 1$ indicates the annotator gives a correct label and $s_i^j = 0$ a wrong label.

As an example, suppose we already get a sample $f_1 = 1.96$ and $z_1 = 1$. For annotator j, we know the annotator's competence is η_j. We will get a correct label $y_1^j = z_1 = 1$ with probability η_j and a wrong label $y_1^j = 0$ with probability $1 - \eta_j$.

3.3.3 GP-GLAD. The original GLAD model [38] is for bi-class tasks. It models $p(z_i)$ by a categorical distribution $p(z_i = o) = \tau_o$. It assumes that both the difficulty of the example and the competence of the annotator affect the observed crowd labels and models $p(y_i^j \mid z_i)$ by a logistic function $\sigma_i^j = (1 + e^{-\alpha_i \beta_j})^{-1}$, where α_i is the difficulty of example i and β_j is the competence of annotator j. Model parameters are largely compressed to $M{+}N$. Formally, $p(y_i^j \mid z_i)$ can be written as:

$$p(y_i^j \mid z_i) = {\sigma_i^j}^{\mathbb{I}(y_i^j = z_i)} (1 - \sigma_i^j)^{\mathbb{I}(y_i^j \neq z_i)}. \tag{7}$$

Similar to GP-DS, we extend GLAD by adding a latent function $\boldsymbol{f}$ as the prior of $\boldsymbol{z}$. We name the model GP-GLAD. In line 4–5 of Algorithm 1, for each annotator j, s_i^j is sampled from $Bernoulli(\sigma_i^j)$. The variable s_i^j indicates whether the annotator gives the correct label.

As an example, suppose we already get a sample $f_1 = 1.96$ and $z_1 = 1$. For annotator j, we will get a correct label $y_1^j = z_1 = 1$ with the probability of σ_{i}^j and we will get a wrong label $y_1^j = 0$ with the probability of $1 - \sigma_i^j$.

3.3.4 GP-MACE. The original MACE model [15] is for multi-class tasks. It models $p(y_i^j = l \mid z_i = o)$ using a confusion matrix similar to the DS model but reducing the number of parameters to $2M$.

Similar to GP-DS, we extend MACE by adding a latent function $\boldsymbol{f}$ as the prior of $\boldsymbol{z}$. We name the model GP-MACE. In line 4–5 of Algorithm 1, for each annotator j, s_i^j is sampled from $Bernoulli(\epsilon_j)$. The variable s_i^j indicates whether the annotator is *not spamming* on the example. When the annotator is not spamming on the example ($s_i^j = 1$), he copies the true label to produce the annotation y_i^j; when he is spamming on the example ($s_i^j = 0$), he produces the annotation y_i^j from $Bernoulli(\xi^j)$. Note that ξ^j indicates annotator i's label preference when he is spamming. Formally, $p(y_i^j \mid z_i)$ can be written as:

$$\begin{aligned}
p(y_i^j \mid z_i) &= \left(p(s_i^j) + (1 - p(s_i^j))p(z_i = y_i^j)\right)^{\mathbb{I}(z_i = y_i^j)} \\
&\qquad \left((1 - \epsilon_j)p(z_i \neq y_i^j)\right)^{\mathbb{I}(z_i \neq y_i^j)} \\
&= \left(\epsilon_j + (1 - \epsilon_j){\xi^j}^{z_i^j} (1 - \xi^j)^{(1 - z_i^j)}\right)^{\mathbb{I}(z_i = y_i^j)} \\
&\qquad \left((1 - p(s_i^j)){\xi^j}^{(1 - z_i^j)} (1 - \xi^j)^{z_i^j}\right)^{\mathbb{I}(z_i \neq y_i^j)}.
\end{aligned} \tag{8}$$

E.g., suppose we already get a sample $f_1 = 1.96$ and $z_1 = 1$. For annotator j, we first sample s_i^j from $Bernoulli(\epsilon_j)$ and assume $s_{i_j}^j = 0$. Thus the annotator is spamming and he will randomly label $y_1^j = 1$ with probability ξ^j and $y_1^j = 0$ with probability $1 - \xi^j$.

3.4 Model optimization and label inference

The extended models GP-DS, GP-ZC, GP-GLAD, GP-MACE contain parameters from the mean function and covariance function of the GP prior $p(\boldsymbol{f})$, and parameters from the corresponding likelihood $p(Y \mid \boldsymbol{z})$. We use $\boldsymbol{\theta}$ to denote all the parameters. In order to learn the parameters and infer latent true labels, similar to the vanilla GPC, we adopt a Bayesian view and maximize the log-likelihood of the observed data plus the log of the parameter prior. We formally write the optimization problem as:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} \left\{\log p\left(Y \mid \boldsymbol{X}, \boldsymbol{\theta}\right) + \log p\left(\boldsymbol{\theta}\right)\right\}. \tag{9}$$

As the first part $\log p(Y \mid \boldsymbol{X}, \boldsymbol{\theta}) = \int p(Y \mid \boldsymbol{f}) p(\boldsymbol{f}) \, \mathrm{d}\boldsymbol{f}$ is intractable due to the multiplication of a non-Gaussian likelihood and a Gaussian prior, we instead maximize its ELBO, which is tractable. The derivation of ELBO is as follows:

$$\begin{aligned}
\log p(Y \mid X, \boldsymbol{\theta}) &\triangleq \log p(Y) \\
&= \int q(\boldsymbol{f}) \log \frac{p(Y, \boldsymbol{f})}{q(\boldsymbol{f})} \mathrm{d}\boldsymbol{f} + \mathbb{KL}\left(q(\boldsymbol{f}) \,\|\, p(\boldsymbol{f} \mid Y)\right) \\
&\geqslant \int q(\boldsymbol{f}) \log \frac{p(Y, \boldsymbol{f})}{q(\boldsymbol{f})} \mathrm{d}\boldsymbol{f} \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log p(Y \mid \boldsymbol{f})\right] - \mathbb{KL}\left[q(\boldsymbol{f}) \,\|\, p(\boldsymbol{f})\right] \\
&= \mathbb{E}_{q(\boldsymbol{f})}\left[\log\left(\prod_{i=1}^{N} \sum_{z_i=0}^{1} \prod_{j=1}^{M} p(y_i^j \mid z_i) p(z_i \mid f_i)\right)\right] \\
&\quad - \mathbb{KL}\left[q(\boldsymbol{f}) \,\|\, p(\boldsymbol{f})\right],
\end{aligned} \tag{10}$$

where $q(\boldsymbol{f}) \triangleq q(\boldsymbol{f} \mid \boldsymbol{\psi}) \triangleq \mathcal{N}(\boldsymbol{\mu}_\psi, \Sigma_\psi)$ is a parameterized multivariate Gaussian distribution approximating $p(\boldsymbol{f} \mid Y)$; $p(\boldsymbol{f})$ is the prior Gaussian distribution; $p(y_i^j \mid z_i)$ is defined in Eq. (5), (6), (7), (8), and $p(z_i \mid f_i)$ is defined in Eq. (3).

Finally, we apply the variational expectation maximization (VEM) algorithm [29] to maximize the objective function, which means to $ELBO(\boldsymbol{\psi}, \boldsymbol{\theta}) + \log p(\boldsymbol{\theta})$. Both the E and M steps maximize the same function, the difference is that the E step maximizes it with respect to the parameters of $q(\boldsymbol{f})$ while the M step maximizes it with respect to the model parameters $\boldsymbol{\theta}$.

So far, we have found the Gaussian approximation $q(\boldsymbol{f} \mid \boldsymbol{\mu}_\psi, \Sigma_\psi)$. We can infer the latent true label z_* for a new point or an existing point $\boldsymbol{x}_*$. The derivation is similar to the vanilla GPC model:

$$p(z_* = 1 \mid \boldsymbol{x}_*, X, Y) = \Phi(\mu_*). \tag{11}$$

3.5 Model selection

We have introduced different GPLA models and their optimization. A natural question left is how to choose between different GPLA models. We propose to use the Bayes' rule for model selection. Suppose there is a set of different GPLA models under consideration, denoted as $\{\mathcal{M}_i \mid i = 1, 2, \ldots\}$. The best model should be

$$\underset{i}{\operatorname{argmax}}\, p(\mathcal{M}_i \mid Y). \tag{12}$$

Using Bayes' rule, we have

$$p(\mathcal{M}_i \mid Y) = \frac{p(Y \mid \mathcal{M}_i)\, p(\mathcal{M}_i)}{p(Y)} \tag{13}$$

Since we have no prior knowledge about the models, it is reasonable to assume $p(\mathcal{M}_i)$ is the same for each model. $p(Y)$ is the marginalized likelihood and is the same for each model. Therefore, optimizing Eq. (12) is equivalent to optimizing

$$\underset{i}{\operatorname{argmax}}\, p(Y \mid \mathcal{M}_i). \tag{14}$$

Given the model $\mathcal{M}_i$, note that $p(Y \mid \mathcal{M}_i)$ is Eq. (9), which is the objective function for optimizing the model.

To sum up, model selection is to select the model that has the highest likelihood value on the observed data, meaning to select the model that most fits the observed data.

3.6 Complexity analysis

The model parameters are from the likelihood part and the GP prior part. The number of parameter for the likelihood are $2M$, M, $N + M$, and $2M$ for GP-DS, GP-ZC, GP-GLAD, and GP-MACE, respectively. In addition, the GP prior contains N (for $\boldsymbol{\mu}_\psi$) and $\frac{N^2}{2}$ (for Σ_ψ) parameters for $q(\boldsymbol{f})$. It also contains N parameters if a linear mean function is used. For a constant zero mean function and a pre-trained model as mean function, there is no parameter to learn.

Table 1: Statistics of the CS2010 and CS2011 datasets.

Data set	CS2010	CS2011
# Examples	3,275	711
# Rel	1,775	589
# Nonrel	1,500	122
# Annotators	722	181
# Annotations	18,479	2,181

4 EXPERIMENTAL SETUP

4.1 Research questions

The overall question we want to answer is *whether the introduction of a Gaussian process (GP) prior can improve the performance of label aggregation (LA) models in inferring latent true labels.* Specifically, we split the question into three aspects: (RQ1) Do the Gaussian process-based label aggregation models outperform the underlying LA models? (RQ2) What is the influence of label quality and the number of labels per example on the performance of GPLA models? (RQ3) How stable is the relative performance of GPLA models under different crowd label generation assumptions?

4.2 Datasets

4.2.1 *Real data.*

We evaluate our models on two real datasets. They are from the TREC crowdsourcing track in 2010 (CS2010[2]) and 2011 (CS2011[3]) and contain crowdsourcing relevance labels between queries and documents. Each example is a tuple of query ID, document ID, annotator ID, ground truth label, and crowd label. We remove invalid examples that have broken document links, or that have no ground truth labels, or that have no document texts available. We also turn the original ternary scale into a binary scale by mapping highly relevant or relevant labels to relevant labels and the rest to non-relevant labels, i.e., we map label value 2 or 1 to 1 and keep 0 as the same. Consequently, 3,275 query-document pairs with 18,479 crowd labels remain for CS2010 and 711 unique query-document pairs with 2,181 crowd labels remain for CS2011. The statistics of the two datasets after preprocessing are shown in Table 1.

As the GPLA models require feature vectors from examples in addition to their crowd labels, we augment each example in the original data with a feature vector. We use the pre-trained *BERT-FirstP* neural ranking model [5] because it uses the same document collection as the two crowdsourcing datasets and thus can generate query-document feature vectors correlating well regarding relevance labels. We input `query [SEP] document` and output the vector of the `[CLS]` token as feature vectors.

4.2.2 *Simulated data.*

We also evaluate our LA models on simulated data. The benefits of simulated data are two-fold: (i) we have ground truth values for all the parameters to estimate how accurate

[2] https://www.ischool.utexas.edu/~ml/data/trec-rf10-crowd.tgz

[3] https://sites.google.com/site/treccrowd/2011

the models are in learning parameters; and (ii) we can control label quality and the number of labels per example to study their impact on model performance.

We propose a data simulation algorithm for each of the four baseline LA models. See Algorithm 2 in the Appendix. First, for all four LA models, we generate example features and example true labels from two multi-dimensional Gaussian distributions, one as positive and the other as negative. Second, we generate workers and crowd labels based on the label generation process assumed in DS, ZC, GLAD, and MACE, respectively. We adjust their corresponding parameters to make sure around 10% annotators annotate 80% examples because it is empirically shown more than 80% of the examples are annotated by only 10% of the workforce [17]. The parameters are summarized in Table 6 in the Appendix.

4.3 Evaluation metric

The mainstream work on label aggregation uses standard classification metrics like accuracy and F1. But we found it does not reflect model performance well, the performance is not consistent among different datasets. This is because a hard threshold, usually 0.5, is used to discriminate between positive and negative predictions. We claim that AUC is a better metric in such a case [10]. Instead of the accuracy score or the F1 score which considers only one threshold (e.g., 0.5), AUC provides an aggregate measure of performance across all possible thresholds. Overall, AUC measures the probability that the model ranks a random positive example more highly than a random negative example. We report the AUC score in our experiments.

4.4 Experiments

We conduct three experiments. (i) Our first experiment addresses RQ1 by comparing the baseline LA models with their corresponding GPLA extensions. In order to understand the impact of the input on the performance of GPLA models, we also compare three variants: (a) a zero mean function that uses crowd labels and example features as input, (b) a linear mean function that uses crowd labels and example features as input plus an extra parameter $\boldsymbol{w}$ that needs to be learned, and (c) a pre-trained mean function that uses crowd labels, example features, and a label prediction model as input. We use two real crowd datasets, CS2010 and CS2011. (ii) Our second experiment addresses RQ2 by comparing the performance of LA and GPLA models on simulated datasets with varying label quality and numbers of labels. We choose a data simulation algorithm that has the same label generation assumptions as the model to be evaluated, making sure that label quality and the number of labels are the only changing factors. For example, we use the DS-sim dataset to evaluate the DS and GP-DS models. For the number of labels, we set $P = 1, 2, 3, 5$; for label quality, we ensure that the percentage of correct crowd labels per example is within the range of $[0.5, 0.55]$, $[0.6, 0.65]$, $[0.7, 0.75]$, or $[0.8, 0.85]$ by controlling the corresponding parameters of data simulation algorithms. We set $N = 1,000$ and $M = 10$. We report the result of the GPLA models with the zero function, where the main difference between a GPLA model and the underlying LA model is whether example correlation is modeled (using the covariance function of GP). (iii) Our third experiment addresses RQ3 by comparing the performance of LA and GPLA models under different crowd label generation assumptions. There are four crowd label generation assumptions. For each, we set $P = 3$ and the label accuracy in $[0.7, 0.75]$. Thus, we have four simulated datasets: DS-sim, ZC-sim, GLAD-sim, and MACE-sim. We evaluate the LA and GPLA models on all the simulated datasets. As in our second experiment, we use the zero function as the mean function of the GP prior. Each experiment is repeated 5 times; we report the average AUC scores over those repetitions.

4.5 Implementation

We use the implementation from Zheng et al. [43] for the four baselines, DS, ZC, GLAD, MACE. We use GPflow [25] to implement our extensions with a GP prior, GP-DS, GP-ZC, GP-GLAD, and GP-MACE. When optimizing the GPLA models, as we adopt a Bayesian view, we need to specify the prior distributions for model parameters, $p(\boldsymbol{\theta})$. We use $Gamma(1, 1)$ as the prior for all positive scalar parameters, $\mathcal{N}(0, 1)$ as the prior for all non-constrained scalar parameters, and $Beta(1, 1)$ as the prior for all distributional parameters. The number of optimization epochs is set to 5 on simulated data and 100 on real data to ensure model convergence. The code that we used to produce our experimental results is publicly available.[4]

5 RESULTS

5.1 Performance of label aggregation models extended with a Gaussian process prior

Table 2 lists the mean AUC scores of the baseline LA models and their extensions with a GP prior. GPLA models consistently outperform the underlying baseline models. The Gaussian process-based label aggregation models are able to outperform the underlying LA models by integrating example features and pre-trained label prediction model.

Table 2: Mean AUC scores on real datasets of baseline LA methods, their extensions with a GP prior.

	Dataset	
Model	CS2010	CS2011
DS	0.7074	0.6451
ZC	0.5559	0.3917
GLAD	0.6307	0.5180
MACE	0.6919	0.5925
GP-DS	0.7368 ↑	0.7522 ↑
GP-ZC	0.7368 ↑	0.7522 ↑
GP-GLAD	0.7368 ↑	0.7522 ↑
GP-MACE	0.7368 ↑	0.7522 ↑

5.2 Quality of crowd labels

To answer RQ2, we study whether the extended LA models GP-X perform better than the underlying LA models X on crowd sourced data with different label quality (% of correct labels per example) and different numbers of labels.

Table 3 shows the mean AUC scores of four groups of LA models, where every group consists of one of the baseline LA models, DS,

[4] https://github.com/dli1/gp4la

ZC, GLAD, and MACE, plus its extension with a GP prior. We discuss the results for DS and GP-DS. The results for GLAD, ZC, and MACE are qualitatively similar. First, the performance of both DS and GP-DS increases as the number of labels increases and as label accuracy increases. Second, GP-DS performs better than DS in most cases. When the label accuracy is within the range of $[0.5, 0.55]$, $[0.6, 0.65]$, and $[0.7, 0.75]$, GP-DS outperforms DS; when the label accuracy is within the range of $[0.8, 0.85]$, GP-DS outperforms DS if the number of labels is 1 or 2, and slightly worse than DS when the number of labels is 3 or 5. We believe that this is due to the fact that when label quality is high, the likelihood part $p(Y \mid z)$ plays a more important role than the prior $p(f)p(z \mid f)$. Overall, the results indicate that modeling example correlation through a GP prior can help improve the performance on label aggregation in many settings.

Thus, the performance of the GPLA models and the LA models increases when label accuracy increases and the number of labels increases; the GPLA models outperform the LA models in most cases and the improvement is especially large if there are only 1 or 2 labels per example or the example accuracy is less than 0.8.

5.3 Crowd data generation

Table 4 shows the AUC scores of LA and GPLA. First, the four LA models do not generalize well across the four datasets. This finding is similar to [24, 43]. For example, DS wins on the datasets of DS-sim, ZC wins on the datasets of ZC-sim, GLAD-sim, and MACE-sim. It also indicates that the assumption of label generation process is a bottleneck that prevents these models generalizable to different datasets. Second, the GPLA models generalize well to the four datasets. On the four simulated datasets, the GPLA models perform better than the corresponding LA models.

To sum up, the GP prior of the GPLA models makes them easy to generalize to different datasets. This is a key capability of GPLA: to introduce prior knowledge of true labels and model example correlation; while the LA models do not have this capability, they can only use the crowdsourcing labels to infer the true labels.

5.4 Discussion

5.4.1 Example feature. Example features play an important role for GPLA. From the formula $\mu_* = m(\boldsymbol{x}_*) + K_* K^{-1}(\boldsymbol{\mu}_\psi - m(X))$ in Eq. (1), we know that the true label of a target example $\boldsymbol{x}_*$ is roughly determined by the weighted average of the of its neighbor example labels, i.e., $\boldsymbol{\mu}_\psi$ indicates the labels of neighbor examples and $K_* K^{-1}$ indicates the weights. Ideally, if the example features for the binary classes have clearly separate clusters in their vector space, it will be likely that the true label of the target example is the same with its neighbors. However, if not clustered well, the target example is surrounded with examples of different classes and the it is not confident for the model to determine the true label.

Based on the discussion of example features, we can examine the results on real data (Table 2) and simulated data (Table 3) again. Figure 2 plots positive examples in blue and negative in yellow. It shows that the example features for real data are not clustered well, while the example features on simulated data are clustered well. As a sequence, we observe that the performance improvement of GPLA on real data is not as much as on simulated data.

Table 3: The impact of crowd label quality of simulated data; mean AUC scores; ↑ indicates that the AUC score of an extension GP-X is higher than of the underlying model X.

	Number	% range of correct labels per example			
Model	of labels	0.5–0.55	0.6–0.65	0.7–0.75	0.8–0.85
DS	1	0.5291	0.6259	0.7246	0.8283
	2	0.5453	0.6832	0.8143	0.9197
	3	0.5495	0.7176	0.8600	0.9560
	5	0.5689	0.7831	0.9317	0.9890
GP-DS	1	0.5481 ↑	0.7932 ↑	0.8870 ↑	0.9229 ↑
	2	0.6458 ↑	0.8518 ↑	0.9028 ↑	0.9349 ↑
	3	0.6040 ↑	0.8723 ↑	0.9215 ↑	0.9434
	5	0.6679 ↑	0.8877 ↑	0.9350 ↑	0.9489
ZC	1	0.5211	0.6201	0.7189	0.8222
	2	0.5263	0.6797	0.8144	0.9253
	3	0.5479	0.7208	0.8711	0.9632
	5	0.5627	0.7851	0.9309	0.9902
GP-ZC	1	0.6023 ↑	0.8069 ↑	0.8888 ↑	0.9241 ↑
	2	0.5863 ↑	0.8501 ↑	0.9071 ↑	0.9325 ↑
	3	0.6482 ↑	0.8797 ↑	0.9241 ↑	0.9438
	5	0.6409 ↑	0.8907 ↑	0.9328 ↑	0.9472
GLAD	1	0.5066	0.6029	0.7074	0.8250
	2	0.5190	0.6523	0.8159	0.9204
	3	0.5026	0.6716	0.8898	0.9692
	5	0.5215	0.7745	0.9618	0.9940
GP-GLAD	1	0.5552 ↑	0.7875 ↑	0.8763 ↑	0.9192 ↑
	2	0.5130	0.8189 ↑	0.8939 ↑	0.9298 ↑
	3	0.5512 ↑	0.8400 ↑	0.9197 ↑	0.9389
	5	0.5494 ↑	0.8739 ↑	0.9310	0.9460
MACE	1	0.5408	0.6489	0.7446	0.8560
	2	0.5296	0.6533	0.7580	0.8441
	3	0.5539	0.6970	0.8316	0.9330
	5	0.5647	0.7389	0.8897	0.9732
GP-MACE	1	0.6510 ↑	0.8279 ↑	0.8929 ↑	0.9299 ↑
	2	0.6476 ↑	0.8714 ↑	0.9151 ↑	0.9375 ↑
	3	0.6955 ↑	0.8870 ↑	0.9317 ↑	0.9428 ↑
	5	0.7394 ↑	0.9074 ↑	0.9377 ↑	0.9486

An interesting future direction is to jointly optimize for both the example features and the label aggregation model. A similar problem has been studied for *node classification on graphs*. For example, Zhao et al. [42] proposed a variational expectation maximization framework for text-attributed graphs which iteratively updates the language model to generate better node representation and update the graph neural network for better node classification. It will be interesting to study its applicability in the label aggregation task.

5.4.2 Mean function. One advantage of a GPLA model over its underlying LA model is that it has a mean function that can provide prior knowledge of the true labels. Common options for a mean function include the constant zero function, a linear function, and any other functions such as a pre-trained label prediction model. We compare three mean functions to understand their impact on model performance.

Table 4: The impact of the crowd label generation assumptions of simulated data; mean AUC scores; ↑ indicates that the AUC score of an extension GP-X is higher than of the underlying model X.

	Dataset			
Model	DS-sim	ZC-sim	GLAD-sim	MACE-sim
DS	0.8600	0.8708	0.9023	0.8957
ZC	0.8593	0.8711	0.9053	0.8966
GLAD	0.8430	0.8615	0.8898	0.8889
MACE	0.8005	0.8046	0.8289	0.8316
GP-DS	0.9215 ↑	0.9241 ↑	0.9195 ↑	0.9314 ↑
GP-ZC	0.9215 ↑	0.9241 ↑	0.9195 ↑	0.9314 ↑
GP-GLAD	0.9214 ↑	0.9241 ↑	0.9197 ↑	0.9314 ↑
GP-MACE	0.9215 ↑	0.9243 ↑	0.9199 ↑	0.9317 ↑

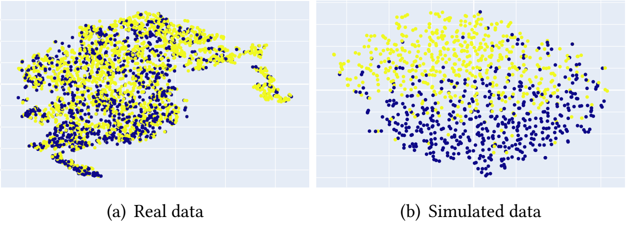

(a) Real data (b) Simulated data

Figure 2: Visualization of example features from real data (CS2010) and simulated data.

Results are shown in Table 5. First, GPLA models with a zero mean function perform on par with their underlying LA models. This indicates that example features may not correlate well with true example labels. We leave the selection of example features as future work. GPLA models with a linear mean function do not perform better than LA. A possible reason is that the linear function introduces extra parameters and thus makes the model under-fit to the crowdsourcing data. To see this, the crowdsourcing labels used for training are $N \times M$, in practice, usually between 3N and 5N; the GPLA models have $N + \frac{N^2}{2}$ parameters for the GP prior part, plus the number of parameters of the likelihood part, which is different for each LA model. Finally, a pre-trained mean function consistently outperforms other mean functions. Using a label prediction model trained on existing data provides useful prior knowledge of true labels and is important to learn an effective GPLA model.

5.4.3 Limitation of label aggregation. Throughout the paper, we are assuming that the ground truth labels are always correct. However, in practical applications, the ground truth labels from experts can also contain noise. Generally speaking, the experts are not necessarily the experts on certain task by academic standards, but very often simply the people the author of the annotation work knows. The expert annotation becomes a matter of availability of human resources to perform the annotation task. As a consequence, the quality of the ground truth labels affects the training and evaluation of the label aggregation models. We leave this issue for future work.

Furthermore, label aggregation models are helpful for relatively objective tasks, but have limitations for subjective tasks requiring human perception [2]. In these cases, a single ground truth label does not even exist. For example, in the annotation of offensive language, the same message can be perceived as abusive by one annotator and not abusive by another annotator. The labels are inherently associated with each individual annotator. It may make more sense to model the downstream task directly using the pre-aggregated data.

Table 5: Impact of the mean function on GPLA; mean AUC scores; ↑ indicates that the AUC score of an extension GP-X is higher than of the underlying model X.

		Dataset	
Model	**Mean function**	CS2010	CS2011
GP-DS	zero	0.6819	0.5646
GP-ZC	zero	0.6819 ↑	0.5646 ↑
GP-GLAD	zero	0.6779 ↑	0.5282 ↑
GP-MACE	zero	0.5923	0.5424
GP-DS	linear	0.5821	0.4751
GP-ZC	linear	0.5959 ↑	0.4749 ↑
GP-GLAD	linear	0.5921	0.4755
GP-MACE	linear	0.5923	0.4733
GP-DS	pre-train	0.7368 ↑	0.7522 ↑
GP-ZC	pre-train	0.7368 ↑	0.7522 ↑
GP-GLAD	pre-train	0.7368 ↑	0.7522 ↑
GP-MACE	pre-train	0.7368 ↑	0.7522 ↑

6 CONCLUSION

In this work, we have addressed the label aggregation task. We have proposed a way of extending LA models by integrating a GP prior, which results in GPLA models that can take as input crowd labels, example features, and existing label prediction models to infer the true labels, while the original LA can only leverage crowd labels. We have presented GP extensions of four baseline label aggregation models, DS, ZC, GLAD, and MACE. GP extensions of other LA models can be derived following these examples.

We have conducted experiments on both simulated and real data. We find that with different label accuracy and numbers of labels, the GPLA models perform better than the LA models in most cases; the improvement is especially large for small numbers of labels or low label accuracy. We also find that the GPLA models outperform the original LA models on multiple datasets with different label generation assumptions.

An important direction for future work is to adapt the GPLA for multi-class labels. The key idea is to replace the single GP by multiple GPs and replace the probit function by a softmax function. Since we observed that example features plays an important role for GPLA, another interesting direction is to jointly optimize for both the example features and the label aggregation model.

ACKNOWLEDGMENTS

This research was (partially) funded by the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organization for Scientific Research, https://hybrid-intelligence-centre.nl. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

A SIMULATED DATA

In this appendix we describe the algorithm used to generate simulated data and the parameters used to generate the simulated datasets in our experiments.

Algorithm 2 Simulated crowd label generation.

Input: Number of examples N, number of workers M, number of labels per example P, example feature dimension D.
Parameter: Union distribution parameters $l, h \in [0, 1]$, GLAD parameter μ.
Out: List of tuples (example id, example feature, worker id, true label, crowd label).

```
# Generate N examples.
for $i = 1$ to N do
    Sample a true label $z_i$ from $Bernoulli(0.5)$.
    Sample an example feature $\boldsymbol{x}_i$ from $\mathcal{N}(I[z_i, :], I)$.
end for
// Generate M workers.
for $j = 1$ to $M$ do
    if DS then
        Sample $p_j$ from $\mathcal{U}(l, h)$ and set
        $\Lambda^j = [[p_j, 1 - p_j], [1 - p_j, p_j]]$ as the confusion matrix.
    else if ZC then
        Sample the worker competence value $\eta_j$ from $\mathcal{U}(l, h)$.
    else if GLAD then
        Sample the worker competence value $\beta_j$ from $\mathcal{N}(\mu, 1)$.
    else if MACE then
        Set the worker label preference $\xi_j = 0.5$.
    end if
end for
// Generate N × P crowd labels.
for $i = 1$ to $N$ do
    Sample $P$ workers $\{i_1, \ldots, i_P\}$ from $p(j) = \frac{j^{-1.5}}{\sum_{j=1}^{M} j^{-1.5}}$.
    for $j = i_1$ to $i_P$ do
        if DS then
            Sample a crowd label $y_i^j$ from $Bernoulli(\lambda_{z_i 1}^j)$.
        else if ZC then
            Sample $s_i^j$ from $Bernoulli(\eta_j)$. If $s_i^j = 1$, the worker
            gives a correct label, $y_i^j = z_i$; else $y_i^j = 1 - z_i$.
        else if GLAD then
            Sample $s_i^j$ from $Bernoulli(\frac{1}{1+e^{-\alpha_i \beta_j}})$. If $s_i^j = 1$, the
            worker gives a correct label, $y_i^j = z_i$; else $y_i^j = 1 - z_i$.
        else if MACE then
            Sample $s_i^j$ from $Bernoulli(\epsilon_j)$. If $s_i^j = 1$, the worker
            gives a correct label, $y_i^j = z_i$; else, the worker guesses a
            label $y_i^j$ from $Bernoulli(\xi_j)$.
        end if
    end for
end for
```

Table 6: Parameters for generating the simulated data.

	% range of correct labels per example			
#labels	0.5–0.55	0.6–0.65	0.7–0.75	0.8–0.85
DS	Parameters: (l, h)			
1	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
2	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
3	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
5	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
	% of correct labels per example			
1	0.52	0.62	0.71	0.82
2	0.55	0.64	0.74	0.83
3	0.53	0.63	0.72	0.82
5	0.53	0.63	0.72	0.82
ZC	Parameters: (l, h)			
1	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
2	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
3	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
5	(0.5, 0.55)	(0.6, 0.65)	(0.7, 0.75)	(0.8, 0.85)
	% of correct labels per example			
1	0.55	0.65	0.74	0.83
2	0.55	0.64	0.73	0.83
3	0.52	0.63	0.72	0.82
5	0.53	0.64	0.73	0.82
GLAD	Parameters: (l, h, μ)			
1	(0.2, 0.4, 0.2)	(0.4, 1.0, 1.0)	(1.0, 1.4, 1.0)	(1.4, 2.0, 1.0)
2	(0.4, 1.0, 0.2)	(0.4, 1.0, 1.0)	(1.0, 1.4, 1.0)	(1.4, 2.0, 1.2)
3	(1.0, 1.4, 0.2)	(0.4, 1.0, 1.0)	(1.0, 1.4, 1.0)	(1.4, 2.0, 1.2)
5	(1.4, 2.0, 1.0)	(0.4, 1.0, 1.0)	(1.0, 1.4, 1.0)	(1.4, 2.0, 1.2)
	% of correct label per example			
1	0.51	0.62	0.75	0.82
2	0.54	0.63	0.74	0.83
3	0.52	0.61	0.73	0.82
5	0.51	0.61	0.72	0.80
MACE	Parameters: (l, h)			
1	(0.85, 1.0)	(0.6, 0.8)	(0.4, 0.6)	(0.2, 0.4)
2	(0.85, 1.0)	(0.6, 0.8)	(0.4, 0.6)	(0.2, 0.4)
3	(0.85, 1.0)	(0.6, 0.8)	(0.4, 0.6)	(0.2, 0.4)
5	(0.85, 1.0)	(0.6, 0.8)	(0.4, 0.6)	(0.2, 0.4)
	% of correct labels per example			
1	0.53	0.64	0.72	0.86
2	0.52	0.64	0.73	0.83
3	0.53	0.64	0.73	0.82
5	0.54	0.62	0.73	0.83

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] Valerio Basile. 2023. The Perspectivist Data Manifesto. https://pdai.info/.

[3] Peng Cao, Yilun Xu, Yuqing Kong, and Yizhou Wang. 2019. Max-MIG: an Information Theoretic Approach for Joint Learning from Crowds. In *International Conference on Learning Representations*.

[4] Xi Chen, Paul N Bennett, Kevyn Collins-Thompson, and Eric Horvitz. 2013. Pairwise Ranking Aggregation in a Crowdsourced Setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*. 193–202.

[5] Zhuyun Dai and Jamie Callan. 2019. Deeper Text Understanding for IR with Contextual Neural Language Modeling. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 985–988.

[6] Alexander Philip Dawid and Allan M Skene. 1979. Maximum Likelihood Estimation of Observer Error-rates using the EM Algorithm. *Applied statistics* (1979), 20–28.

[7] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2012. ZenCrowd: Leveraging Probabilistic Reasoning and Crowdsourcing Techniques for Large-Scale Entity Linking. In *Proceedings of the 21st International Conference on World Wide Web*. 469–478.

[8] Djellel Difallah and Alessandro Checco. 2021. Aggregation Techniques in Crowdsourcing: Multiple Choice Questions and Beyond. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*. 4842–4844.

[9] Alexey Drutsa, Valentina Fedorova, Dmitry Ustalov, Olga Megorskaya, Evfrosiniya Zerminova, and Daria Baidakova. 2020. Practice of Efficient Data Collection via Crowdsourcing: Aggregation, Incremental Relabelling, and Pricing. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 873–876.

[10] Peter A Flach, José Hernández-Orallo, and Cèsar Ferri Ramirez. 2011. A Coherent Interpretation of AUC as a Measure of Aggregated Classification Performance. In *ICML*.

[11] Meric Altug Gemalmaz and Ming Yin. 2021. Accounting for Confirmation Bias in Crowdsourced Label Aggregation.. In *IJCAI*. 1729–1735.

[12] Perry Groot, Adriana Birlutiu, and Tom Heskes. 2011. Learning from Multiple Annotators with Gaussian Processes. In *International Conference on Artificial Neural Networks*. Springer, 159–164.

[13] Oliver Hamelijnck, Theodoros Damoulas, Kangrui Wang, and Mark Girolami. 2019. Multi-resolution Multi-task Gaussian Processes. *Advances in Neural Information Processing Systems* 32 (2019).

[14] Lei Han, Eddy Maddalena, Alessandro Checco, Cristina Sarasua, Ujwal Gadiraju, Kevin Roitero, and Gianluca Demartini. 2020. Crowd Worker Strategies in Relevance Judgment Tasks. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 241–249.

[15] Dirk Hovy, Taylor Berg-Kirkpatrick, Ashish Vaswani, and Eduard Hovy. 2013. Learning Whom to Trust with MACE. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 1120–1130.

[16] Oana Inel, Giannis Haralabopoulos, Dan Li, Christophe Van Gysel, Zoltán Szlávik, Elena Simperl, Evangelos Kanoulas, and Lora Aroyo. 2018. Studying Topical Relevance with Evidence-based Crowdsourcing. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1253–1262.

[17] Ayush Jain, Akash Das Sarma, Aditya Parameswaran, and Jennifer Widom. 2017. Understanding Workers, Developing Effective Tasks, and Enhancing Marketplace Dynamics: A Study of a Large Crowdsourcing Marketplace. *Proceedings of the VLDB Endowment* 10, 7 (2017), 829–840.

[18] Yuan Jin, Mark Carman, Ye Zhu, and Yong Xiang. 2020. A Technical Survey on Statistical Modelling and Design Methods for Crowdsourcing Quality Control. *Artificial Intelligence* 287 (2020), 103351.

[19] Gabriella Kazai, Jaap Kamps, and Natasa Milic-Frayling. 2013. An Analysis of Human Factors and Label Accuracy in Crowdsourcing Relevance Judgments. *Information retrieval* 16, 2 (2013), 138–178.

[20] Hyun-Chul Kim and Zoubin Ghahramani. 2012. Bayesian Classifier Combination. In *Artificial Intelligence and Statistics*. 619–627.

[21] Ho Chung Law, Dino Sejdinovic, Ewan Cameron, Tim Lucas, Seth Flaxman, Katherine Battle, and Kenji Fukumizu. 2018. Variational Learning on Aggregate Outputs with Gaussian Processes. *Advances in neural information processing systems* 31 (2018).

[22] Dan Li, Zhaochun Ren, and Evangelos Kanoulas. 2021. CrowdGP: A Gaussian Process Model for Inferring Relevance from Crowd Annotations. In *Proceedings of the Web Conference 2021*. 1821–1832.

[23] Shao-Yuan Li, Sheng-Jun Huang, and Songcan Chen. 2021. Crowdsourcing Aggregation with Deep Bayesian Learning. *Science China Information Sciences* 64, 3 (2021), 1–11.

[24] Yuan Li. 2019. *Probabilistic Models for Aggregating Crowdsourced Annotations*. Ph. D. Dissertation. University of Melbourne, Parkville, Victoria, Australia.

[25] Alexander G. de G. Matthews, Mark van der Wilk, Tom Nickson, Keisuke Fujii, Alexis Boukouvalas, Pablo León-Villagrá, Zoubin Ghahramani, and James Hensman. 2017. GPflow: A Gaussian Process Library Using TensorFlow. *The Journal of Machine Learning Research* 18, 1 (2017), 1299–1304.

[26] Geoffrey J McLachlan and Thriyambakam Krishnan. 2007. *The EM algorithm and extensions*. John Wiley & Sons.

[27] Pablo Morales-Álvarez, Pablo Ruiz, Raúl Santos-Rodríguez, Rafael Molina, and Aggelos K Katsaggelos. 2019. Scalable and Efficient Learning from Crowds with Gaussian Processes. *Information Fusion* 52 (2019), 110–127.

[28] Yashar Moshfeghi and Alvaro Francisco Huertas-Rosero. 2021. A Game Theory Approach for Estimating Reliability of Crowdsourced Relevance Assessments. *ACM Transactions on Information Systems (TOIS)* 40, 3 (2021), 1–29.

[29] Radford M Neal and Geoffrey E Hinton. 1998. A View of the EM Algorithm that Justifies Incremental, Sparse, and Other Variants. In *Learning in graphical models*. Springer, 355–368.

[30] Carl Edward Rasmussen. 2004. Gaussian Processes in Machine Learning. In *Advanced lectures on machine learning*. Springer, 63–71.

[31] 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, Apr (2010), 1297–1322.

[32] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Gaussian Process Classification and Active Learning with Multiple Annotators. In *International Conference on Machine Learning*. 433–441.

[33] Kevin Roitero, Alessandro Checco, Stefano Mizzaro, and Gianluca Demartini. 2022. Preferences on a Budget: Prioritizing Document Pairs when Crowdsourcing Relevance Judgments. In *Proceedings of the ACM Web Conference 2022*. 319–327.

[34] Pablo Ruiz, Pablo Morales-Álvarez, Rafael Molina, and Aggelos K Katsaggelos. 2019. Learning from Crowds with Variational Gaussian Processes. *Pattern Recognition* 88 (2019), 298–311.

[35] Michael Soprano, Kevin Roitero, Francesco Bombassei De Bona, and Stefano Mizzaro. 2022. Crowd Frame: A Simple and Complete Framework to Deploy Complex Crowdsourcing Tasks Off-the-shelf. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*. 1605–1608.

[36] Yusuke Tanaka, Toshiyuki Tanaka, Tomoharu Iwata, Takeshi Kurashima, Maya Okawa, Yasunori Akagi, and Hiroyuki Toda. 2019. Spatially Aggregated Gaussian Processes with Multivariate Areal Outputs. *Advances in Neural Information Processing Systems* 32 (2019).

[37] Jeroen Vuurens, Arjen P de Vries, and Carsten Eickhoff. 2011. How Much Spam Can You Take? An Analysis of Crowdsourcing Results to Increase Accuracy. In *Proc. ACM SIGIR Workshop on Crowdsourcing for Information Retrieval (CIR?11)*. 21–26.

[38] Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. 2009. Whose Vote Should Count More: Optimal Integration of Labels from Labelers of Unknown Expertise. In *Advances in Neural Information Processing Systems*. 2035–2043.

[39] Hanlu Wu, Tengfei Ma, Lingfei Wu, Fangli Xu, and Shouling Ji. 2021. Exploiting Heterogeneous Graph Neural Networks with Latent Worker/Task Correlation Information for Label Aggregation in Crowdsourcing. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 16, 2 (2021), 1–18.

[40] Ming Wu, Qianmu Li, Jing Zhang, and Jun Hou. 2022. Label Aggregation with Clustering for Biased Crowdsourced Labeling. In *2022 14th International Conference on Machine Learning and Computing (ICMLC)*. 165–169.

[41] Fariba Yousefi, Michael T Smith, and Mauricio Alvarez. 2019. Multi-task Learning for Aggregated Data Using Gaussian Processes. *Advances in Neural Information Processing Systems* 32 (2019).

[42] Jianan Zhao, Meng Qu, Chaozhuo Li, Hao Yan, Qian Liu, Rui Li, Xing Xie, and Jian Tang. 2023. Learning on Large-scale Text-attributed Graphs via Variational Inference. In *The Eleventh International Conference on Learning Representations*. https://openreview.net/forum?id=q0nmYciuuZN

[43] Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. 2017. Truth Inference in Crowdsourcing: Is the Problem Solved? *Proceedings of the VLDB Endowment* 10, 5 (2017), 541–552.

[44] Yao Zhou, Fenglong Ma, Jing Gao, and Jingrui He. 2019. Optimizing the Wisdom of the Crowd: Inference, Learning, and Teaching. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 3231–3232.