Collusion-resistant Spatial Phenomena Crowdsourcing via Mixture of Gaussian Processes Regression

Qikun Xiang[1], Ido Nevat[2], Pengfei Zhang[2], Jie Zhang[1]

[1] Nanyang Technological University, Singapore.
[2] Institute for Infocomm Research (I^2R), Singapore.

Abstract

With the rapid development of mobile devices, *spatial location-based crowdsourcing* applications have attracted much attention. These applications also introduce new security risks due to untrustworthy data sources. In the context of crowdsourcing applications for spatial interpolation (i.e. spatial regression) using crowdsourced data, the results can be seriously affected if malicious data sources initiate a colluding (collaborate) attacks which purposely alter some of the measurements. To combat this serious detrimental effect, and to mitigate such attacks, we develop a robust version via a *Gaussian Process mixture* model and develop a computationally efficient algorithm which utilises a Markov chain Monte Carlo (MCMC)-based methodology to produce an accurate predictive inference in the presence of collusion attacks. The algorithm is fully Bayesian and produces posterior predictive distribution for any point-of-interest in the input space. It also assesses the trustworthiness of each worker, i.e. the probability of each worker being honest (trustworthy). Simulation results demonstrate the accuracy of this algorithm.

1 Introduction

Crowdsourcing, as defined by Howe [1], is often characterized by the participation of large networks of people from the public in tasks given by companies or institutes. It has been broadly used in data acquisition, which replaces the traditional data gathering processes that are costly. Recent works have shown many successful applications in crowdsourced information gathering from various different domains. In [2], the authors introduce a crowdsourcing framework for acquisition and analysis of mobile videos to monitor rapidly changing disaster sites that optimally utilise bandwidth and resources. In a study done by Meier et al. [3], crowdsourced atmospheric data in Berlin, Germany was used for urban climate research. In [4], a privacy-preserving community sensing method is developed for real-time road traffic monitoring. Another study by Kim et al. [5] uses crowdsourcing and regression techniques to automatically analyse the conflict level in television political debates.

While decentralized spatial information gathering (crowdsourcing) techniques have clear benefits, such as being inexpensive and easy to implement, their quality and reliability are not as good as data collected from

In: J. Zhang, R. Cohen, and M. Sensoy (eds.): Proceedings of the 18th International Workshop on Trust in Agent Societies, Singapore, 09-MAY-2016, published at http://ceur-ws.org

domain experts. The issue of trust is one of the major impediments to the success of real-world crowdsourcing applications [6]. Apart from inherent noises, untrustworthiness in crowdsourced data can originate from various ways. For example the sensor used for measurement can be faulty (if the data is in the form of sensor reading). Data that come from humans can be subjective and highly biased [7]. For example, workers might put very little effort into the work they are assigned in the hope that they will not be noticed and still get the reward. Other workers might lack the essential skills to perform the task, hence produce erroneous data [7]. In addition, there can be adversaries that report malicious data strategically to interrupt the crowdsourcing application [8].

The challenge of identifying and filtering untrustworthy information before summarization (prediction, decision making, etc.) has been addressed by a number of researchers in various domains. Some of these works focus on binary/multi-class labelling tasks. Groot et at. [9] applied a Gaussian Process for regression model to merge information from multiple inaccurate annotators to produce reliable labels for objects. Tarasov et at. [10] proposed a method to dynamically estimate reliability of workers in crowdsourcing for regression by adopting a multi-armed bandits (MAB) model.

Some studies are applied to problems of estimation. Reece et al. [11] studied the problem of tackling sensor faults in applications based on sensor networks. Their method consists of two stages. Firstly the model-based step detects and filters sensor faults with known types, then the consensus-based step removes unanticipated faults that passed the first step. Venanzi et al. [12] developed an algorithm called MaxTrust that jointly estimates the target attribute and the trustworthiness of each worker using maximum likelihood paradigm.

Not much attention has been given to address the problem of spatial regression (or the estimation of a spatial process/function) with the presence of untrustworthy data sources. Venanzi et al. [13] proposed a heteroskedastic Gaussian Process (HGP) model with independent noises and designed an algorithm called TrustHGP to jointly estimate the spatial function and the trustworthiness of each worker. Their model assumes that all reports are noisy observations from the same underlying function with heterogeneous and independent noises. Malicious observations are associated with higher noises compared to honest observations. These assumptions of their model are too intuitive to be able to handle complex real-world situations. Especially, this method is vulnerable against collusion attacks.

In collusion attack scenarios in crowdsourcing, multiple untrustworthy workers can collaborate to inject deceptive false data to skew the regression results. More specifically, an effective strategy for attackers is to make a number of similar reports that are far from the ground-truth (or follow a spatial function that is far from the underlying spatial function) with similar input values. This often results in the regression algorithm mistakenly accepting the false value provided by attackers. We refer to this type of attacks and similar types as collusion attacks. We will formally present the attacker model in the model specification in Section 3 of this paper.

1.1 The Proposed Gaussian Processes Mixture Model

In order to address the issue of collusion attacks, we propose a Gaussian Processes mixture model where malicious data and non-malicious (honest) data are modelled to be from two different realizations of Gaussian Processes. We develop a Markov chain Monte Carlo (MCMC) based algorithm that takes all data as input and produces statistical prediction of function value at any spatial location of interest. The predictions produced by the algorithm is based on honest data. It minimizes the impact of malicious data by excluding them before performing posterior inference with high accuracy. The algorithm also assesses the trustworthiness of each worker in terms of the probability that the data it produces being honest. We show that this method is truly collusion-resistant by performing experiment on synthetic dataset.

1.2 Contribution

The contribution of this paper includes:

- We develop a unified statistical framework for collusion attacks in the regression setting via a Gaussian Process mixture model.
- We develop a novel MCMC-based algorithm to produce comprehensive summary of data in terms of posterior predictive distribution and trustworthiness of each worker (data point). The inference method is fully Bayesian.
- We conduct a proof of concept on synthetic dataset and show that our method is truly resistant to collusion attacks.

2 Problem Statement

In the field of spatial monitoring, n sets of data are gathered from n different workers that are potentially untrustworthy. Each set of data comes in the form of a tuple $(\mathbf{x}_i, y_i),\ i = 1, \ldots, n$, where $\mathbf{x}_i \in \mathbb{R}^d$ represents the independent variables (spatial location, time, input features, etc.), and $y_i \in \mathbb{R}$ represents the response (dependent) variable that depends on $\mathbf{x}_i$. Let $\mathbf{x} := \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, and let $\mathbf{y} := \{y_1, \ldots, y_n\}$. We refer to a tuple of $(\mathbf{x}_i, y_i)$ as a data point. Each data point is either honest or malicious, depending on whether it comes from a trustworthy or untrustworthy worker. Honest data points are noisy observations from the underlying function $f_{\mathrm{H}} : \mathbb{R}^d \to \mathbb{R}$, while malicious data points are not (and will be defined later).

The task of the regression problem is that given $\mathbf{x}$ and y, produce a reliable estimation of $f_{\mathrm{H}}(\mathbf{x}_*)$ with its probability distribution, for any given $\mathbf{x}_* \in \mathbb{R}^d$, that is free from the influence of malicious data. The probability density function $p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, y)$ is referred to as *posterior predictive distribution* and contains all the information about $f_{\mathrm{H}}(\mathbf{x}_*)$, given all the observations.

2.1 Illustrative Example of Spatial Phenomena Crowdsourcing

To motivate our model, we present a real-world example of the stated problem. Suppose a meteorological study aims to estimate the spatial structure of atmospheric parameters (e.g. temperature, humidity, air quality, etc.) of a particular region of interest by using crowdsourced data collected from workers in the region by hand-held devices equipped with sensors. In this example, each report $(\mathbf{x}_i, y_i)$ consists of: $\mathbf{x}_i$, the geographical location of the i-th worker (longitude and latitude), and y_i, the reading of the atmospheric parameter from the sensor y_i, e.g. air quality. The underlying function of interest f_{H} corresponds to the inherent spatial structure of the parameter, i.e. for a given location $\mathbf{x}_*$, $f_{\mathrm{H}}(\mathbf{x}_*)$ corresponds to the ground-truth of the value of the parameter at the location. Suppose that, for some reasons, a worker reports dishonest data at location $\mathbf{x}_{\mathrm{m}}$ that does not correspond to the actual parameter value $f_{\mathrm{H}}(\mathbf{x}_{\mathrm{m}})$. This piece of data is deemed as malicious. There can be practical reasons why malicious data exists, such as to defame a spatial location by claiming that the air quality in the vicinity is poor. There can be multiple malicious workers reporting dishonestly to achieve their objectives. Thus, the data reported by these workers cannot be trusted and should be regarded as unrelated to f_{H}.

3 Crowdsourcing Statistical Model Development

In this section, we formally present the statistical crowdsourcing model which uses Gaussian Process mixture model.

3.1 Gaussian Process, Covariance Functions and Hyperparameters

We model the physical phenomena as spatially dependent continuous processes with a spatial correlation structure and are independent from each other. The degree of the spatial correlation in the process increases with the decrease of the separation between two observing locations and can be accurately modelled as a Gaussian random field [1]. A Gaussian process (GP) defines a distribution over a space of functions and it is completely specified by the equivalent of sufficient statistics for such a process, and is formally defined as follows.

Definition 1 *(Gaussian process [14]): Let $\mathcal{X} \subset \mathbb{R}^D$ be some bounded domain of a D-dimensional real valued vector space. Denote by $f(\mathbf{x}) : \mathcal{X} \mapsto \mathbb{R}$ a stochastic process parametrized by $\mathbf{x} \in \mathcal{X}$. Then, the random function $f(\mathbf{x})$ is a Gaussian process if all its finite dimensional distributions are Gaussian, where for any $m \in \mathbb{N}$, the random variables $(f(\mathbf{x}_1), \cdots, f(\mathbf{x}_m))$ are normally distributed.*

We can therefore interpret a GP as formally defined by the following class of random functions:

$$\begin{aligned}
\mathcal{F} := \{ f(\cdot) : \mathcal{X} \mapsto \mathbb{R} \text{ s.t. } f(\cdot) \sim \mathcal{GP}\left(\mu(\cdot;\boldsymbol{\theta}), \mathcal{C}(\cdot,\cdot;\boldsymbol{\Psi})\right), \text{ with} \\
\mu(\mathbf{x};\boldsymbol{\theta}) := \mathbb{E}[f(\mathbf{x})] : \mathcal{X} \mapsto \mathbb{R}, \\
\mathcal{C}(\mathbf{x}_i, \mathbf{x}_j; \boldsymbol{\Psi}) := \mathbb{E}\left[(f(\mathbf{x}_i) - \mu(\mathbf{x}_i;\boldsymbol{\theta}))(f(\mathbf{x}_j) - \mu(\mathbf{x}_j;\boldsymbol{\theta}))\right] : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}^+ \},
\end{aligned}$$

where at each point the mean of the function is $\mu(\cdot;\boldsymbol{\theta})$, parameterised by $\boldsymbol{\theta}$, and the spatial dependence between any two points is given by the covariance function (Mercer kernel) $\mathcal{C}(\cdot,\cdot;\boldsymbol{\Psi})$, parameterised by $\boldsymbol{\Psi}$ (see detailed discussion in [14]).

[1]We use Gaussian Process and Gaussian random field interchangeably.

3.2 Specification of the Spatial Processes

We assume that f_H can be accurately modelled a Gaussian Process prior with zero mean and square exponential covariance function parameterised by $\sigma_\mathrm{H}^2, w_{\mathrm{H}1}^2, \cdots, w_{\mathrm{H}d}^2$.

$$f_\mathrm{H} \sim \mathcal{GP}(0, \mathcal{C}_\mathrm{H}),$$

$$\mathcal{C}_\mathrm{H}(\mathbf{x}_i, \mathbf{x}_{i'}) = \sigma_\mathrm{H}^2 \exp\left(-\frac{1}{2}\sum_{m=1}^{d}\frac{(x_{im} - x_{i'm})^2}{w_{\mathrm{H}m}^2}\right).$$

Similarly, f_M is also modelled a Gaussian Process prior with zero mean and square exponential covariance function parameterised by $\sigma_\mathrm{M}^2, w_{\mathrm{M}1}^2, \cdots, w_{\mathrm{M}d}^2$.

$$f_\mathrm{M} \sim \mathcal{GP}(0, \mathcal{C}_\mathrm{M}),$$

$$\mathcal{C}_\mathrm{M}(\mathbf{x}_i, \mathbf{x}_{i'}) = \sigma_\mathrm{M}^2 \exp\left(-\frac{1}{2}\sum_{m=1}^{d}\frac{(x_{im} - x_{i'm})^2}{w_{\mathrm{M}m}^2}\right).$$

3.3 Partition of Data Points and Hyperparameters of Gaussian Processes

Data points are partitioned into 2 mutually exclusive sets, H and M, denoting honest data points and malicious data points. In other words, either $i \in \mathrm{H}$ or $i \in \mathrm{M}$. Without prior knowledge, H is indistinguishable from M. Hence, we make the assumption that H contains the majority of data points, that is, $|\mathrm{H}| > |\mathrm{M}|$. The case where malicious data points become the majority is unrealistic in real applications. When it happens, there is no way to distinguish which set is honest without informative prior knowledge.

Since each data point belongs either to H or M, we use $\mathbf{c} \in \{\mathrm{H}, \mathrm{M}\}^n$ to denote the configuration vector, where each component c_i denotes the set which i belongs to. In other words,

$$c_i = \begin{cases} \mathrm{H} & \text{if } i \in \mathrm{H} \\ \mathrm{M} & \text{if } i \in \mathrm{M} \end{cases}.$$

As mentioned above, honest data points are noisy observations from underlying function $f_\mathrm{H} : \mathbb{R}^d \to \mathbb{R}$. Similarly, malicious data points are noisy observations from a different function $f_\mathrm{M} : \mathbb{R}^d \to \mathbb{R}$. Hence,

$$y_i = \begin{cases} f_\mathrm{H}(\mathbf{x}_i) + \epsilon_i, & \text{if } c_i = \mathrm{H} \\ f_\mathrm{M}(\mathbf{x}_i) + \omega_i, & \text{if } c_i = \mathrm{M} \end{cases}, \tag{1}$$

where

$$\epsilon_i \overset{iid}{\sim} N(0, \sigma_n^2),\ \omega_i \overset{iid}{\sim} N(0, \sigma_\omega^2). \tag{2}$$

Within each set, the Gaussian noises are assumed to be independently and identically distributed (iid). The assumption of homoscedasticity is made to simplify derivation. In future work we will explore the possibility to extended it to incorporate input-dependent noises such as in [15–17].

Let θ denote hyperparameters of above GPs (including variances of noise),

$$\theta = (\sigma_\mathrm{H}^2, \sigma_\mathrm{M}^2, w_{\mathrm{H}1}^2, \cdots, w_{\mathrm{H}d}^2, w_{\mathrm{M}1}^2, \cdots, w_{\mathrm{M}d}^2, \sigma_n^2, \sigma_\omega^2).$$

In these hyperparameters, signal variances σ_H^2 and σ_M^2, noise variances σ_n^2 and σ_ω^2 are given inverse-gamma priors,

$$\sigma_\mathrm{H}^2, \sigma_\mathrm{M}^2 \overset{iid}{\sim} \text{Inv-Gamma}(\alpha_f, \beta_f),\ \sigma_n^2, \sigma_\omega^2 \overset{iid}{\sim} \text{Inv-Gamma}(\alpha_n, \beta_n),$$

and length-scales $w_{\mathrm{H}1}^2, \cdots, w_{\mathrm{H}d}^2, w_{\mathrm{M}1}^2, \cdots, w_{\mathrm{M}d}^2$ are given log-normal prior,

$$w_{\mathrm{H}1}^2, \cdots, w_{\mathrm{H}d}^2, w_{\mathrm{M}1}^2, \cdots, w_{\mathrm{M}d}^2 \overset{iid}{\sim} \ln N(\mu_w, \sigma_w^2).$$

Placing priors over hyperparameters makes the method fully Bayesian and provides extra flexibility to the model. Parameters $\alpha_f, \beta_f, \alpha_n, \beta_n, \mu_w, \sigma_w^2$ are pre-set values to make these priors non-informative. Alternatively, when prior knowledge about hyperparameters is available, these parameters can be tuned in order to achieve better accuracy.

In addition, we define the following notations that are used in the following parts of the paper.

$$\mathbf{x}_{\mathrm{H}} = \{\mathbf{x}_i : c_i = \mathrm{H}\},\ \mathbf{x}_{\mathrm{M}} = \{\mathbf{x}_i : c_i = \mathrm{M}\},$$
$$\mathbf{y}_{\mathrm{H}} = \{y_i : c_i = \mathrm{H}\},\ \mathbf{y}_{\mathrm{M}} = \{y_i : c_i = \mathrm{M}\},$$
$$n_{\mathrm{H}} = |\mathrm{H}| = \sum_{i=1}^{n} \delta(c_i, \mathrm{H}),\ n_{\mathrm{M}} = |\mathrm{M}| = \sum_{i=1}^{n} \delta(c_i, \mathrm{M}).$$

3.4 Distribution of Independent Variables

In our assumptions, $\mathbf{x}_{\mathrm{H}}$ is distributed uniformly throughout the domain of $\mathbf{x}$ since no region is preferred over other regions when gathering data. When prior knowledge about distribution of independent variables is present, the uniformity assumption can be replaced with no difficulty.

$$x_{\mathrm{H}_{im}} \overset{iid}{\sim} U(l_m, u_m),\ \text{for } i = 1 \cdots n, m = 1 \cdots d,$$

where $x_{\mathrm{H}_{im}}$ denotes the mth dimension of ith honest data point, l_m, u_m are the lower bound and upper bound of the mth dimension.

It is assumed that $\mathbf{x}_{\mathrm{M}}$ follows a d-variate Gaussian distribution with mean $\boldsymbol{\mu}_x$ and covariance Σ_x. This is a practical assumption since in the presence of collusion attacks, the damage maximizing strategy for attackers is to concentrate malicious data points in a particular region. In addition, attackers are assumed to possess limited resources, making it difficult for them to inject malicious data from a broad region (for example, when the independent variable is geographical locations). In the case of a distributed attack where independent variables of malicious data spread through a broad region, Σ_x corresponds to a large value that makes the distribution of $\mathbf{x}_{\mathrm{M}}$ flat.

$$\mathbf{x}_{\mathrm{M}_i} \overset{iid}{\sim} N(\boldsymbol{\mu}_x, \Sigma_x),\ \text{for } i = 1 \cdots n,$$

where $\mathbf{x}_{\mathrm{M}_i}$ denotes the ith malicious data point.

Let ϕ denote parameters of the input distribution,

$$\phi = (\boldsymbol{\mu}_x, \Sigma_x),$$

where $\boldsymbol{\mu}_x$ is given uniform prior,

$$\mu_{xm} \sim U(l_m, u_m),\ \text{for } m = 1 \cdots d,$$

and Σ_x is given inverse-Wishart prior,

$$\Sigma_x \sim W^{-1}(\Psi, \nu),$$

and the parameters Ψ, ν are set to be non-informative.

3.5 Prior Over Configurations

We assume there is no a priori information about the trustworthiness of each worker. Hence there is an identical probability ρ of each data point being honest. Therefore, $\mathbf{c}$ has a n-variate Bernoulli distribution with parameter ρ (H corresponds to the positive outcome and M corresponds to the negative outcome),

$$c_i \overset{iid}{\sim} \text{Bernoulli}(\rho),\ \text{for } i = 1 \cdots n.$$

It is obvious that n_{H} follows a binomial distribution with parameters n and ρ.

$$n_{\mathrm{H}} \sim B(n, \rho).$$

Strictly speaking, since $n_{\mathrm{H}} > n_{\mathrm{M}}$, n_{H} should follow a truncated binomial distribution where $n_{\mathrm{H}} > \lceil \frac{n}{2} \rceil$. For simplicity, this condition is ignored here. As later discussed in the algorithm, if a posterior sample violates the condition, we simply ignore the sample.

ρ is given a beta prior, with known parameters α_ρ and β_ρ,

$$\rho \sim \text{Beta}(\alpha_\rho, \beta_\rho).$$

Here α_ρ and β_ρ should be set such that $\alpha_\rho \gg \beta_\rho$, because before making any observations, all workers should be trusted as being honest.

4 Posterior Predictive Inference and MCMC Sampler Design

Based on the model we propose, the goal is to obtain the posterior predictive distribution, which is derived by marginalizing over $\mathbf{c}$ and θ, i.e.

$$p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}) = \sum_{\mathbf{c}} \int_{\theta} p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}, \mathbf{c}, \theta) p(\mathbf{c}, \theta|\mathbf{x}, \mathbf{y}) d\theta.$$

The first term inside the integral $p(f(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}, \mathbf{c}, \theta)$ is the posterior distribution of Gaussian Process regression and is Gaussian:

$$p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}, \mathbf{c}, \theta) = N(f_{\mathrm{H}}(\mathbf{x}_*); \boldsymbol{\mu}_f, \Sigma_f), \tag{3}$$

where

$$\begin{aligned} \boldsymbol{\mu}_f &= \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*)^T [\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_{\mathrm{H}}) + \sigma_n^2 I]^{-1} \mathbf{y}_{\mathrm{H}}, \\ \Sigma_f &= \mathcal{C}_{\mathrm{H}}(\mathbf{x}_*, \mathbf{x}_*) - \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*)^T [\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_{\mathrm{H}}) + \sigma_n^2 I]^{-1} \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*). \end{aligned} \tag{4}$$

The second term inside the integral $p(\mathbf{c}, \theta|\mathbf{x}, \mathbf{y})$ is the marginal posterior distribution of $\mathbf{c}$ and θ obtained by marginalizing out ϕ and ρ from the full posterior $p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})$.

$$\begin{gathered} p(\mathbf{c}, \theta|\mathbf{x}, \mathbf{y}) = \int_0^1 \int_{\phi} p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y}) d\phi d\rho, \\ p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y}) \propto p(\mathbf{y}|\mathbf{x}, \mathbf{c}, \theta) p(\mathbf{x}|\mathbf{c}, \phi) p(\mathbf{c}|\rho) p(\rho) p(\theta) p(\phi). \end{gathered} \tag{5}$$

This full posterior $p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})$ is clearly intractable. However, if we could sample from this posterior $p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})$ and get S samples $\{(\mathbf{c}^{(1)}, \theta^{(1)}, \phi^{(1)}, \rho^{(1)}), \cdots, (\mathbf{c}^{(S)}, \theta^{(S)}, \phi^{(S)}, \rho^{(S)})\}$, we could approximate the posterior predictive distribution by:

$$p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}) \approx \frac{1}{S} \sum_{r=1}^{S} p(f_{\mathrm{H}}(\mathbf{x}_*)|\mathbf{x}_*, \mathbf{x}, \mathbf{y}, \mathbf{c}^{(r)}, \theta^{(r)}) \tag{6}$$

This posterior predictive distribution is a mixture of multivariate Gaussian distribution, whose mean $\boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}$ and covariance matrix $\Sigma_{f_{\mathrm{H}}(\mathbf{x}_*)}$ are given by:

$$\begin{aligned} \boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)} &= \frac{1}{S} \sum_{r=1}^{S} \boldsymbol{\mu}^{(r)}, \\ \Sigma_{f_{\mathrm{H}}(\mathbf{x}_*)} &= \sum_{r=1}^{S} (\Sigma^{(r)} + \boldsymbol{\mu}^{(r)} \boldsymbol{\mu}^{(r)^T}) - \boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)} \boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}^T, \end{aligned} \tag{7}$$

where $\boldsymbol{\mu}^{(r)}$ and $\Sigma^{(r)}$ are the mean and covariance matrix of the rth component of the mixture, respectively. Inference of other characteristics of the posterior predictive distribution, such as median, mode, percentiles, etc. can be done by generating T samples from each of the S components from the approximate posterior predictive distribution and use the histogram of ST samples to approximate the posterior predictive distribution.

Therefore, we develop a MCMC sampler algorithm to generate samples from the posterior distribution $p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})$. We can use Gibbs Sampling to generate samples from conditional posterior distribution of each component of the parameter space.

4.1 Sampling c

Since it is difficult to sample $\mathbf{c}$ as a whole from its conditional posterior distribution, we sample it component by component. Let $\mathbf{c}_{-i}$ denote the vector resulting from removing i-th component from $\mathbf{c}$.

$$\begin{aligned} p(c_i = j|\mathbf{c}_{-i}, \mathbf{x}, \mathbf{y}, \theta, \phi, \rho) &\propto p\left(y_i|c_i = j, \mathbf{c}_{-i}, \mathbf{x}_{j_{-i}}, \mathbf{y}_{j_{-i}}, \theta\right) p(\mathbf{x}_i|\mathbf{x}_{j_{-i}}, c_i = j, \mathbf{c}_{-i}, \phi) p(c_i = j|\mathbf{c}_{-i}, \rho) \\ &= N(y_i; \boldsymbol{\mu}_j, \Sigma_j) p(\mathbf{x}_i|\mathbf{x}_{j_{-i}}, c_i = j, \mathbf{c}_{-i}, \phi) p(c_i = j|\mathbf{c}_{-i}, \rho), \text{ for } j \in \{\mathrm{H}, \mathrm{M}\}, \end{aligned} \tag{8}$$

where

$$\begin{aligned}
\boldsymbol{\mu}_{\mathrm{H}} &= \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}_{-i}}, \mathbf{x}_i)^T[\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}_{-i}}, \mathbf{x}_{\mathrm{H}_{-i}}) + \sigma_n^2 I]^{-1}\mathbf{y}_{\mathrm{H}_{-i}}, \\
\Sigma_{\mathrm{H}} &= [\mathcal{C}_{\mathrm{H}}(\mathbf{x}_i, \mathbf{x}_i) + \sigma_n^2] - \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}_{-i}}, \mathbf{x}_i)^T[\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}_{-i}}, \mathbf{x}_{\mathrm{H}_{-i}}) + \sigma_n^2 I]^{-1}\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}_{-i}}, \mathbf{x}_i), \\
\boldsymbol{\mu}_{\mathrm{M}} &= \mathcal{C}_{\mathrm{M}}(\mathbf{x}_{\mathrm{M}_{-i}}, \mathbf{x}_i)^T[\mathcal{C}_{\mathrm{M}}(\mathbf{x}_{\mathrm{M}_{-i}}, \mathbf{x}_{\mathrm{M}_{-i}}) + \sigma_\omega^2 I]^{-1}\mathbf{y}_{\mathrm{M}_{-i}}, \\
\Sigma_{\mathrm{M}} &= [\mathcal{C}_{\mathrm{M}}(\mathbf{x}_i, \mathbf{x}_i) + \sigma_\omega^2] - \mathcal{C}_{\mathrm{M}}(\mathbf{x}_{\mathrm{M}_{-i}}, \mathbf{x}_i)^T[\mathcal{C}_{\mathrm{M}}(\mathbf{x}_{\mathrm{M}_{-i}}, \mathbf{x}_{\mathrm{M}_{-i}}) + \sigma_\omega^2 I]^{-1}\mathcal{C}_{\mathrm{M}}(\mathbf{x}_{\mathrm{M}_{-i}}, \mathbf{x}_i),
\end{aligned}$$

$$\begin{aligned}
p(\mathbf{x}_i|\mathbf{x}_{j_{-i}}, c_i = \mathrm{H}, \mathbf{c}_{-i}, \phi) &= \frac{1}{\prod_{m=1}^{d}(u_m - l_m)}, \\
p(\mathbf{x}_i|\mathbf{x}_{j_{-i}}, c_i = \mathrm{M}, \mathbf{c}_{-i}, \phi) &= N(\mathbf{x}_i; \boldsymbol{\mu}_x, \Sigma_x), \\
p(c_i = \mathrm{H}|\mathbf{c}_{-i}, \rho) &= \rho, \\
p(c_i = \mathrm{M}|\mathbf{c}_{-i}, \rho) &= 1 - \rho.
\end{aligned} \tag{9}$$

In above equations, $\mathbf{x}_{j_{-i}} = \mathbf{x} \setminus \mathbf{x}_i$, $\mathbf{y}_{j_{-i}} = \mathbf{y} \setminus y_i$ for $j \in \{\mathrm{H}, \mathrm{M}\}$. Note that rank-one updates to the inverse covariance matrix can be performed to reduce the number of inversion needed.

4.2 Sampling θ

We can sample the entire θ from its conditional posterior distribution.

$$\begin{aligned}
p(\theta|\mathbf{x}, \mathbf{y}, \mathbf{c}, \phi, \rho) &\propto p(\mathbf{y}|\mathbf{x}, \mathbf{c}, \theta)p(\theta) \\
&= p(\mathbf{y}|\mathbf{x}, \mathbf{c}, \theta)p(\sigma_{\mathrm{H}}^2)p(\sigma_{\mathrm{M}}^2)p(\sigma_n^2)p(\sigma_\omega^2)\prod_{m=1}^{d} p(w_{\mathrm{H}m}^2)p(w_{\mathrm{M}m}^2)
\end{aligned} \tag{10}$$

The unnormalized distribution can be derived analytically, along with its partial derivative w.r.t. each parameter. Hence Hamiltonian Monte Carlo (HMC) [19] can be used to efficiently sample from this conditional posterior.

4.3 Complementary Metropolis-Hastings Jump

When sampling $\mathbf{c}$ and θ, a convergence issue may occur. Intuitively, the joint posterior distribution $p(\mathbf{c}, \theta, \phi, \rho|x, y)$ should be bimodal, since the model specification of $\mathbf{c}$ and θ is highly symmetric except for the condition that $n_{\mathrm{H}} > n_{\mathrm{M}}$, as reflected in the prior of c, and the distribution of independent variables $\mathbf{x}_{\mathrm{H}}$ and $\mathbf{x}_{\mathrm{M}}$. Thus, switching the labels of all data points (labelling all honest data points as M and all malicious data points as H) will result in an inferior mode in the posterior distribution (besides the major mode we are interested in where all data points are correctly classified). This mode has much less probability mass as compared to the major mode, and since its violation of the condition $n_{\mathrm{H}} > n_{\mathrm{M}}$, it will eventually be rejected when producing posterior predictive inference. Nonetheless, its existence will make the MCMC sampler algorithm highly sensitive to initial state. Due to the nature of Gibbs samplers, $\mathbf{c}$ is updated one component at a time. The slow movement of the Gibbs sampler in the state space will make it extremely unlikely to jump from a state close to inferior mode to a state close to the major mode, due to low probability mass between two modes. Hence, apart from the Gibbs sampler for $\mathbf{c}$ and HMC sampler for θ mentioned above, an auxiliary Metropolis-Hastings sampler is introduced to facilitate the movement between two modes. The Metropolis-Hastings proposal distribution will propose a switch of labels as well as their associated GP hyperparameters θ. Below shows the proposal distribution $q(\mathbf{c}_*|\mathbf{c})$ and $q(\theta_*|\theta)$ of the Metropolis-Hastings jump.

$$\begin{aligned}
q(c_{i*} \neq c_i|c_i) &= p_c, \\
q(\ln\sigma_{\mathrm{H}*}|\theta) &= N(\ln\sigma_{\mathrm{H}*}; \ln\sigma_{\mathrm{M}}, \sigma_F^2), \\
q(\ln\sigma_{\mathrm{M}*}|\theta) &= N(\ln\sigma_{\mathrm{M}*}; \ln\sigma_{\mathrm{H}}, \sigma_F^2), \\
q(\ln\sigma_{n*}|\theta) &= N(\ln\sigma_{n*}; \ln\sigma_\omega, \sigma_N^2), \\
q(\ln\sigma_{\omega*}|\theta) &= N(\ln\sigma_{\omega*}; \ln\sigma_n, \sigma_N^2), \\
q(\ln w_{\mathrm{H}m*}|\theta) &= N(\ln w_{\mathrm{H}m*}; \ln w_{\mathrm{M}m}, \sigma_W^2), \text{ for } m = 1\cdots d, \\
q(\ln w_{\mathrm{M}m*}|\theta) &= N(\ln w_{\mathrm{M}m*}; \ln w_{\mathrm{H}m}, \sigma_W^2), \text{ for } m = 1\cdots d.
\end{aligned} \tag{11}$$

Here, p_c is set to be close to 1 and σ_F^2, σ_N^2 and σ_W^2 are set close to 0 so that the Metropolis-Hastings jump roughly switches labels of all data points along with their corresponding hyperparameters. The update is accepted with probability shown below:

$$p(\text{accept}) = \min\left(\frac{p(\mathbf{c}_*, \theta_*, \phi, \rho|\mathbf{x}, \mathbf{y})q(\mathbf{c}|\mathbf{c}_*)q(\theta|\theta_*)}{p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})q(\mathbf{c}_*|\mathbf{c})q(\theta_*|\theta)}, 1\right). \tag{12}$$

$\dfrac{q(\theta|\theta_*)}{q(\theta_*|\theta)}$ is equal to 1 since the part of the Metropolis-Hastings update related to θ is symmetrical. Therefore, the sampler for $\mathbf{c}$ and θ becomes a mixture of two MCMC samplers. One is the Gibbs sampler for $\mathbf{c}$ and HMC sampler for θ, and the other is a Metropolis-Hastings sampler that jointly updates $\mathbf{c}$ and θ. At each iteration of the MCMC sampler algorithm, with probability p_{MH}, the Metropolis-Hastings sampler is selected. Since the sole purpose of this sampler is to get the chain out of the inferior mode, p_{MH} should be set close to 0. Now when the chain walks into states close to the inferior mode, the Metropolis-Hastings sampler will take the chain to states close to the major mode. When the chain is already close to the major mode, the Metropolis-Hastings jumps will almost never be accepted.

4.4 Sampling ϕ

To sample ϕ, we generate a random sample from the conditional posterior of each component, i.e. from $p(\boldsymbol{\mu}_x|\mathbf{x}_{\text{M}}, \mathbf{c}, \Sigma_x)$ and $p(\Sigma_x|\mathbf{x}_{\text{M}}, \mathbf{c}, \boldsymbol{\mu}_x)$, which are analytically tractable.

For $\boldsymbol{\mu}_x$,

$$\begin{aligned}
p(\boldsymbol{\mu}_x|\mathbf{x}_{\text{M}}, \mathbf{c}, \Sigma_x) &= \frac{p(\mathbf{x}_{\text{M}}|\mathbf{c}, \boldsymbol{\mu}_x, \Sigma_x)p(\boldsymbol{\mu}_x)}{\int_{\boldsymbol{\mu}_x} p(\mathbf{x}_{\text{M}}|\mathbf{c}, \boldsymbol{\mu}_x, \Sigma_x)p(\boldsymbol{\mu}_x)d\boldsymbol{\mu}_x} \\
&= \frac{\frac{1}{C}\prod\limits_{\{i:c_i=\text{M}\}} N(\boldsymbol{\mu}_x; \mathbf{x}_i, \Sigma_x)\prod\limits_{m=1}^{d}\frac{1}{(u_m-l_m)}}{\int_{\boldsymbol{\mu}_x}\frac{1}{C}\prod\limits_{\{i:c_i=\text{M}\}} N(\boldsymbol{\mu}_x; \mathbf{x}_i, \Sigma_x)\prod\limits_{m=1}^{d}\frac{1}{(u_m-l_m)}d\boldsymbol{\mu}_x} \\
&= \frac{N\left(\boldsymbol{\mu}_x; \overline{\mathbf{x}_{\text{M}}}, \frac{\Sigma_x}{n_{\text{M}}}\right)}{\int_{\boldsymbol{\mu}_x} N\left(\boldsymbol{\mu}_x; \overline{\mathbf{x}_{\text{M}}}, \frac{\Sigma_x}{n_{\text{M}}}\right)d\boldsymbol{\mu}_x},
\end{aligned} \tag{13}$$

where $C = \prod\limits_{\{i:c_i=\text{M}\}}\int_{\boldsymbol{\mu}_x} N(\boldsymbol{\mu}_x; \mathbf{x}_i, \Sigma_x)d\boldsymbol{\mu}_x$ is the product of all the normalizing constants to truncate the Gaussian distribution. Since C appears in both the denominator and the numerator, it cancels out. Note that this conditional posterior distribution is a truncated Gaussian distribution and can be efficiently generated.

For Σ_x, we have that

$$\begin{aligned}
p(\Sigma_x|\mathbf{x}_{\text{M}}, \mathbf{c}, \boldsymbol{\mu}_x) &= \frac{p(\mathbf{x}_{\text{M}}|\mathbf{c}, \boldsymbol{\mu}_x, \Sigma_x)p(\Sigma_x)}{\int_{\Sigma_x} p(\mathbf{x}_{\text{M}}|\mathbf{c}, \boldsymbol{\mu}_x, \Sigma_x)p(\Sigma_x)d\Sigma_x} \\
&= \frac{\left(\frac{1}{C}\prod\limits_{\{i:c_i=\text{M}\}}^{d} N(\boldsymbol{\mu}_x; \mathbf{x}_i, \Sigma_x)\right)W^{-1}(\Sigma_x; \Psi, \nu)}{\int_{\Sigma_x}\left(\frac{1}{C}\prod\limits_{\{i:c_i=\text{M}\}}^{d} N(\boldsymbol{\mu}_x; \mathbf{x}_i, \Sigma_x)\right)W^{-1}(\Sigma_x; \Psi, \nu)d\Sigma_x} \\
&= W^{-1}(\Sigma_x; Z + \Psi, n_{\text{M}} + \nu),
\end{aligned} \tag{14}$$

where $Z = \sum\limits_{\{i:c_i=\text{M}\}}(\mathbf{x}_i - \boldsymbol{\mu}_x)(\mathbf{x}_i - \boldsymbol{\mu}_x)^T$. This conditional posterior distribution is an inverse-Wishart distribution and can be directly generated.

Algorithm 1: MCMC Sampler

```
    Input: $S$, $\mathbf{x}$, $\mathbf{y}$ and parameters of priors
    Output: $S$ samples of posterior distribution $(\mathbf{c}^{(1)}, \theta^{(1)}, \phi^{(1)}, \rho^{(1)}), \cdots, (\mathbf{c}^{(S)}, \theta^{(S)}, \phi^{(S)}, \rho^{(S)})$
1   Initialize $\theta$, $\phi$ and $\rho$ by randomly drawing from their respective priors.
2   Initialize $\mathbf{c}$.
3   for $r = 1 \cdots S$ do
4       Randomly sample $u_1 \sim U(0,1)$.
5       if $u_1 < p_{\mathrm{MH}}$ (Probability of proposing a Metropolis-Hastings jump) then
6           Propose a jump to $\mathbf{c}_*, \theta_*$ by $q(\mathbf{c}_*|\mathbf{c})$ and $q(\theta_*|\theta)$.
7           Compute acceptance probability $p(\mathrm{accept})$.
8           Randomly sample $u_2 \sim U(0,1)$.
9           if $u_2 < p(\mathrm{accept})$ then
10              $\mathbf{c} \leftarrow \mathbf{c}_*$.
11              $\theta \leftarrow \theta_*$.
        else
12          for $i = 1 \cdots n$ do
13              Sample $c_i$ from $p(c_i|\mathbf{c}_{-i}, \mathbf{x}, \mathbf{y}, \theta, \phi, \rho)$.
14          Sample $\theta$ from $p(\theta|\mathbf{x}, \mathbf{y}, \mathbf{c}, \phi, \rho)$ by HMC.
15      Sample $\boldsymbol{\mu}_x$ from $p(\boldsymbol{\mu}_x|\mathbf{x}, \mathbf{y}, \mathbf{c}, \theta, \Sigma_x, \rho)$.
16      Sample $\Sigma_x$ from $p(\Sigma_x|\mathbf{x}, \mathbf{y}, \mathbf{c}, \theta, \boldsymbol{\mu}_x, \rho)$.
17      Sample $\rho$ from $p(\rho|\mathbf{x}, \mathbf{y}, \mathbf{c}, \theta, \phi)$.
18      Save current values of $\mathbf{c}$, $\theta$, $\phi$, $\rho$ in $(\mathbf{c}^{(r)}, \theta^{(r)}, \phi^{(r)}, \rho^{(r)})$.
19  return $\{(\mathbf{c}^{(1)}, \theta^{(1)}, \phi^{(1)}, \rho^{(1)}), \cdots, (\mathbf{c}^{(S)}, \theta^{(S)}, \phi^{(S)}, \rho^{(S)})\}$.
```

4.5 Sampling ρ

Finally, to sample ρ we generate a random sample from its conditional posterior, which is also a beta distribution.

$$\begin{aligned} p(\rho|\mathbf{x}, \mathbf{y}, \mathbf{c}, \theta, \phi) &= p(\rho|\mathbf{c}) \\ &= \frac{p(\mathbf{c}|\rho)p(\rho)}{\int_0^1 p(\mathbf{c}|\rho)p(\rho)d\rho} \\ &= \text{Beta}\left(\rho; \alpha_\rho + n_{\mathrm{H}}, \beta_\rho + n - n_{\mathrm{H}}\right). \end{aligned} \tag{15}$$

5 The Algorithms

Our methods consists of two algorithms. The first, presented in Algorithm 1, is a MCMC sampler used to generate samples of model parameters from the posterior distribution $p(\mathbf{c}, \theta, \phi, \rho|\mathbf{x}, \mathbf{y})$. The second algorithm, presented in Algorithm 2, is used to summarize the posterior predictive distribution by calculating the posterior mean and variance. Details of both algorithms are shown. S is set sufficiently large to allow the Markov chain to approximately converge to the target posterior distribution. To assess convergence, we used the method introduced by Gelman et al. in [18].

Apart from mean and variance of the posterior predictive distribution, we also obtain the trustworthiness of each worker (each data point) in the form of the posterior probability of each data point being honest, or alternatively, $p(c_i|\mathbf{x}, \mathbf{y})$ for $i = 1 \cdots n$. This information can be used to further model the behavior of each worker in crowdsourcing applications. However, this is not the main focus of this paper and will not be further discussed. Derivation of posterior probability $p(c_i|\mathbf{x}, \mathbf{y})$ is also straightforward and thus omitted from the algorithm specification.

Assuming $d \ll n$, the computational complexity of Algorithm 1 is limited by the HMC step to sample θ, in which L leapfrog steps are taken, each requires an inverse operation that has $\mathcal{O}(n^3)$ complexity. This results in Algorithm 1 having an overall complexity of $\mathcal{O}(SLn^3)$. The computational complexity of Algorithm 2 is $\mathcal{O}(S'(n_*^2 + n_* n^2))$, where n_* is the length of $\mathbf{x}_*$.

Algorithm 2: Posterior Predictive Inference

```
   Input: $S$ samples of posterior distribution $(\mathbf{c}^{(1)}, \theta^{(1)}, \phi^{(1)}, \rho^{(1)}), \cdots, (\mathbf{c}^{(S)}, \theta^{(S)}, \phi^{(S)}, \rho^{(S)}), \mathbf{x}_*$
   Output: $\boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}$, $\Sigma_{f_{\mathrm{H}}(\mathbf{x}_*)}$
1  Discard the first half of samples as warm-up. The number of samples used for approximating posterior
   predictive distribution is $S'$.
2  $r' \leftarrow 1$.
3  for $r = 1 \cdots S'$ do
4      $n_{\mathrm{H}} \leftarrow \sum_{i=1}^{n} \delta(c_i, \mathrm{H})$.
5      if $2n_{\mathrm{H}} > n$ then
6          $\boldsymbol{\mu}^{(r')} \leftarrow \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*)^T[\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_{\mathrm{H}}) + \sigma_n^2 I]^{-1}\mathbf{y}_{\mathrm{H}}$.
7          $\Sigma^{(r')} \leftarrow \mathcal{C}_{\mathrm{H}}(\mathbf{x}_*, \mathbf{x}_*) - \mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*)^T[\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_{\mathrm{H}}) + \sigma_n^2 I]^{-1}\mathcal{C}_{\mathrm{H}}(\mathbf{x}_{\mathrm{H}}, \mathbf{x}_*)$.
8          $r' \leftarrow r' + 1$.
9  $\boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)} \leftarrow \frac{1}{S} \sum_{r'=1}^{S'} \boldsymbol{\mu}^{(r')}$.
10 $\Sigma_{f_{\mathrm{H}}(x_*)} \leftarrow \sum_{r'=1}^{S'} (\Sigma^{(r')} + \boldsymbol{\mu}^{(r')}\boldsymbol{\mu}^{(r')^T}) - \boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}\boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}^T$.
11 return $(\boldsymbol{\mu}_{f_{\mathrm{H}}(\mathbf{x}_*)}, \Sigma_{f_{\mathrm{H}}(\mathbf{x}_*)})$.
```

6 Simulation Results

To demonstrate that the algorithms are able to make accurate inference from data, we performed experiments on one-dimensional synthetic dataset, where $n = 100$ and $d = 1$, generated in accordance to our assumptions.

In the synthetic dataset, the underlying function is $f_{\mathrm{H}}(x) = \sin \frac{\pi}{2} x$ and the malicious function is $f_{\mathrm{M}}(x) = \frac{1}{2} \tan \frac{\pi}{4} x$. 60 noisy observations are sampled from f_{H}, with homogeneous Gaussian noise with standard deviation 0.10. The distribution of independent variables is uniform in $[0, 1]$. 40 noisy observations are sampled from f_{M}, with homogeneous Gaussian noise with standard deviation 0.12. The distribution of independent variables is Gaussian with mean 0.80 and standard deviation 0.10, truncated to range $[0, 1]$.

The choices of priors in terms of their parameters are given by:

$$\begin{aligned}
\alpha_f &= 2.0, \beta_f = 2.0, \\
\alpha_n &= 2.0, \beta_n = 1.0, \\
\mu_w &= 0.7, \sigma_w^2 = 0.3, \\
\Psi &= 0.0025, \nu = 1.
\end{aligned}$$

For the Metropolis-Hastings sampler, the probability of proposing a jump is $p_{\mathrm{MH}} = 0.05$. Parameters of the proposal distribution is shown below.

$$p_c = 0.95, \sigma_F^2 = 1 \times 10^{-4}, \sigma_N^2 = 1 \times 10^{-4}, \sigma_W^2 = 1 \times 10^{-4}$$

For the sampler of θ, hyperparameters related to H and M are sampled separately, each using an HMC algorithm with $L = 10$ (leapfrog steps).

Finally, for this particular dataset, we set $S = 2000$ since the Markov chain has converged sufficiently after 2000 iterations of the MCMC sampler. Then we pass the second half of the resulting samples to make posterior predictive inference. The results are shown in Fig. 1. Using the posterior mean as the estimator results in a Mean Square Error (MSE) of 0.0012. The posterior probability $p(c_i|\mathbf{x}, \mathbf{y})$ (trustworthiness) of each data point is also computed. In this example, the average trustworthiness of truly honest data points is 0.9906, while the average trustworthiness of truly malicious data points is 5×10^{-5}. It is thus clear that for this set of data, the accuracy of the proposed algorithm is quite high.

As for the efficiency of the algorithm, we measure the average time cost by running the algorithm 50 times using the above dataset on a MacBook Pro equipped with Intel(R) Core(TM) i7-4980HQ CPU @ 2.80GHz. The resulted average time cost is 27.43 seconds per run.

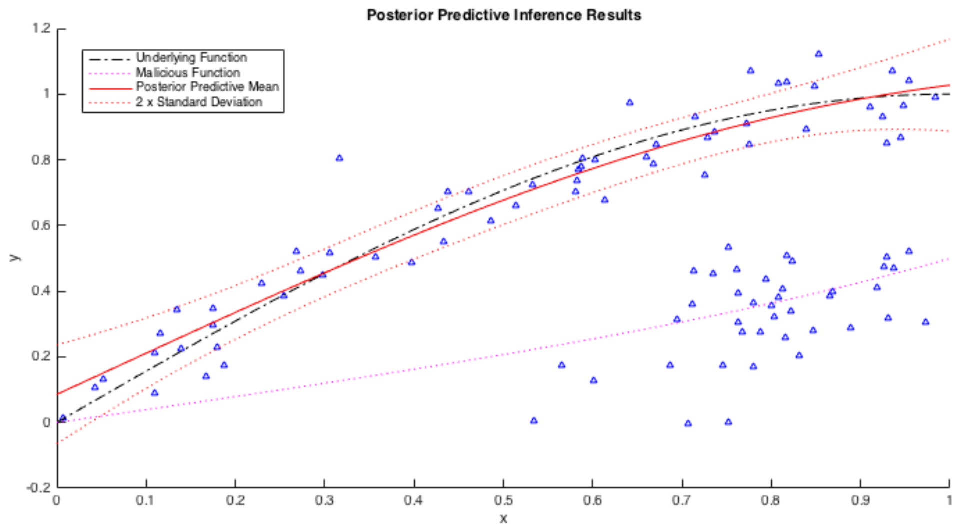

Figure 1: Posterior Predictive Inference Results of Synthetic Dataset

7 Conclusion and Future Works

In this paper, we build a Gaussian Process mixture model and design a MCMC-based algorithm to address the issue of collusion attacks in the regression setting. Honest data from observing underlying function and malicious data are modelled as two separate Gaussian Processes. Compared to [13], where malicious data are assumed to be observed from the same underlying function with extra noise, we claim that the mixture model is a more realistic and natural choice. We use synthetic dataset to show that our algorithm is able to produce accurate posterior predictive inference and is computationally efficient.

As part of an ongoing study, the results of this paper suggest several directions for future studies. Firstly, a single malicious function (class) is too restrictive to handle more sophisticated real-world scenarios and the algorithm currently lacks the ability to handle cases with multiple malicious groups. It is therefore reasonable to extend this model to incorporate multiple malicious classes. One way of modelling is to build a finite mixture of Gaussian Processes model with unknown number of components, and design a MCMC sampler using reversible jump techniques [20]. Another choice is to build an infinite mixture of Gaussian Processes model similar to [21]. Secondly, this model is unable to handle heteroscedasticity as mentioned before, as well as non-Gaussian noises. We will explore approaches to handle these situations in future studies. Lastly, our model is not build specifically to model trustworthiness of workers, which is a significant issue in crowdsourcing applications. In our future work we will improve the model to make robust, collusion-resistant predictions while maintaining trustworthiness of workers for future predictions.

References

[1] Howe, Jeff. *"Crowdsourcing: A definition." Crowdsourcing: Tracking the rise of the amateur* (2006).

[2] To, H., Kim, S.H. and Shahabi, C., 2015, October. Effectively crowdsourcing the acquisition and analysis of visual data for disaster response. In *Big Data (Big Data), 2015 IEEE International Conference on* (pp. 697-706). IEEE.

[3] Meier, Fred, Daniel Fenner, Tom Grassmann, Britta Jnicke, Marco Otto, and Dieter Scherer. "Challenges and benefits from crowd-sourced atmospheric data for urban climate research using Berlin, Germany, as testbed."

[4] Krause, Andreas, Eric Horvitz, Aman Kansal, and Feng Zhao. "Toward community sensing." In *Proceedings of the 7th international conference on Information processing in sensor networks*, pp. 481-492. IEEE Computer Society, 2008.

[5] Kim, Samuel, Maurizio Filippone, Fabio Valente, and Alessandro Vinciarelli. ”Predicting the conflict level in television political debates: an approach based on crowdsourcing, nonverbal communication and gaussian processes.” In *Proceedings of the 20th ACM international conference on Multimedia*, pp. 793-796. ACM, 2012.

[6] Shahabi, Cyrus. ”Towards a generic framework for trustworthy spatial crowdsourcing.” In *Proceedings of the 12th International ACM Workshop on Data Engineering for Wireless and Mobile Acess*, pp. 1-4. ACM, 2013.

[7] Welinder, Peter, and Pietro Perona. ”Online crowdsourcing: rating annotators and obtaining cost-effective labels.” (2010): 25-32.

[8] Hall, David L., and John M. Jordan. *Human-centered information fusion*. Artech House, 2014.

[9] Groot, Perry, Adriana Birlutiu, and Tom Heskes. ”Learning from multiple annotators with Gaussian processes.” In *Artificial Neural Networks and Machine LearningICANN 2011*, pp. 159-164. Springer Berlin Heidelberg, 2011.

[10] Tarasov, Alexey, Sarah Jane Delany, and Brian Mac Namee. ”Dynamic estimation of worker reliability in crowdsourcing for regression tasks: Making it work.” *Expert Systems with Applications* 41, no. 14 (2014): 6190-6210.

[11] Reece, Steven, Stephen Roberts, Christopher Claxton, and David Nicholson. ”Multi-sensor fault recovery in the presence of known and unknown fault types.” In *Information Fusion, 2009. FUSION'09. 12th International Conference on*, pp. 1695-1703. IEEE, 2009.

[12] Venanzi, Matteo, Alex Rogers, and Nicholas R. Jennings. ”Trust-based fusion of untrustworthy information in crowdsourcing applications.” In *Proceedings of the 2013 international conference on autonomous agents and multi-agent systems*, pp. 829-836. International Foundation for Autonomous Agents and Multiagent Systems, 2013.

[13] Venanzi, Matteo, Alex Rogers, and Nicholas R. Jennings. ”Crowdsourcing spatial phenomena using trust-based heteroskedastic gaussian processes.” In *First AAAI Conference on Human Computation and Crowdsourcing*. 2013.

[14] Rasmussen, Carl Edward. ”Gaussian processes for machine learning.” (2006).

[15] Goldberg, Paul W., Christopher KI Williams, and Christopher M. Bishop. ”Regression with input-dependent noise: A Gaussian process treatment.” *Advances in neural information processing systems* 10 (1997): 493-499.

[16] Titsias, Michalis K., and Miguel Lzaro-gredilla. ”Variational heteroscedastic Gaussian process regression.” In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 841-848. 2011.

[17] Kersting, Kristian, Christian Plagemann, Patrick Pfaff, and Wolfram Burgard. ”Most likely heteroscedastic Gaussian process regression.” In *Proceedings of the 24th international conference on Machine learning*, pp. 393-400. ACM, 2007.

[18] Gelman, Andrew, John B. Carlin, Hal S. Stern, and Donald B. Rubin. *Bayesian data analysis*. Vol. 2. Boca Raton, FL, USA: Chapman & Hall/CRC, 2014.

[19] Duane, Simon, Anthony D. Kennedy, Brian J. Pendleton, and Duncan Roweth. ”Hybrid monte carlo.” *Physics letters* B 195, no. 2 (1987): 216-222.

[20] Green, Peter J. ”Reversible jump Markov chain Monte Carlo computation and Bayesian model determination.” *Biometrika* 82, no. 4 (1995): 711-732.

[21] Rasmussen, Carl Edward, and Zoubin Ghahramani. ”Infinite mixtures of Gaussian process experts.” *Advances in neural information processing systems* 2 (2002): 881-888.