Entry Dependent Expert Selection in Distributed Gaussian Processes Using Multilabel Classification

Hamed Jalali, Gjergji Kasneci

Abstract—By distributing the training process, local approximation reduces the cost of the standard Gaussian Process. An ensemble technique combines local predictions from Gaussian experts trained on different partitions of the data. Ensemble methods aggregate models’ predictions by assuming a perfect diversity of local predictors. Although it keeps the aggregation tractable, this assumption is often violated in practice. Even though ensemble methods provide consistent results by assuming dependencies between experts, they have a high computational cost, which is cubic in the number of experts involved. By implementing an expert selection strategy, the final aggregation step uses fewer experts and is more efficient. However, a selection approach that assigns a fixed set of experts to each new data point cannot encode the specific properties of each unique data point. This paper proposes a flexible expert selection approach based on the characteristics of entry data points. To this end, we investigate the selection task as a multi-label classification problem where the experts define labels, and each entry point is assigned to some experts. The proposed solution’s prediction quality, efficiency, and asymptotic properties are discussed in detail. We demonstrate the efficacy of our method through extensive numerical experiments using synthetic and real-world data sets.

Index Terms—Distributed Gaussian Process, Multi-label Classification, Conditional Dependency, Ensemble learning

I. INTRODUCTION

Gaussian processes (GPs) [1] are interpretable and powerful Bayesian non-parametric methods for non-linear regression. A Gaussian process is a stochastic process where every finite collection of those random variables has a multivariate Gaussian distribution. By applying Bayes’ theorem for inference, the posterior predictive distribution of a GP is the best linear unbiased estimator (BLUE) under the assumed model and provides proper quantification of the prediction error uncertainty. GPs do not need restrictive assumptions of the model and can estimate complex linear and non-linear structures. While GPs are extensively used in practical cases [2]–[7], their cubic training and quadratic prediction costs limit their application to big data use cases [8].

For GP regression, the major computational hurdle is the need to estimate the kernel inversion and determinant, which is prohibitively expensive when $n$ is large. Because of this issue, GPs are typically restricted to relatively small training data sets in the range of $O(10^4)$.

To reduce computational expense, sparse approximation techniques use a subset of the training sets (called inducing points) and Nyström approximation to estimate posterior distributions [9]–[12]. In this case, this approach provides a full probabilistic model and appropriate predictions based on the Bayesian framework. Despite its advantages, this method cannot handle large data sets since its capacity is limited by the number of inducing points [13, 14].

Unlike the sparse approximation, which uses only the inducing points, the prominent distributed Gaussian processes (DGPs) use the full training set. This method uses centralized distributed learning, which means that the training data is partitioned into numerous subsets, the local inference is conducted for each partition separately, and then the local estimations are combined through ensemble learning [15]–[17]. A local GP with expertise in a particular partition is called an expert. Experts share the same hyper-parameters, thus accounting for implicit regularisation and encountering overfitting [8, 18].

In a DGP, the conditional independence (CI) assumption between partitions (i.e., between experts given the target) allows for factorizing the global posterior distribution as a product of local distributions. While this assumption reduces the computational cost, it results in inconsistencies and suboptimal solutions [19] caused by the partitioning of the data set, such that when $N \rightarrow \infty$, the CI-based posterior approximations do not converge to the full GP posterior.

Relaxing the independence assumption raises the aggregation’s theoretical properties. If the experts’ predictions are assumed to be random variables, their relative correlations define dependencies between experts. The aggregated posterior distribution, in this case, provides high-quality forecasts and is capable of returning consistent results [20]–[22]. However, solutions that deal with the consistency problem suffer from extra computational costs induced by the need to find the inverse of the covariance matrix between experts for each test point. It means the complexity of this model cubically depends on the number of experts (say $M$), and therefore it can become computationally prohibitive when $M$ is large.

Few works have considered boosting the efficiency of dependency-based aggregation. In [23, 24], authors discuss complexity reduction as an expert selection scenario that excludes a subset of original experts and considers only the valuable experts in the aggregation. For this purpose, the precision matrix of the experts’ predictions is estimated using a Gaussian graphical model (GGM). Experts are the nodes in the obtained undirected sparse graph, and their interactions are the edges. The nodes with fewer interactions are defined as unimportant experts and excluded from the model.
This approach can lower the complexity and provide a good approximation for the original estimator. However, it is not flexible concerning new entries, and the selected experts are fixed for all test points. If new entry data points have specific behavior or are close to excluded partitions, the prediction error increases.

The critical contribution of our work lies in selecting a subset of local experts for each new data point using multi-label classification. Unlike the static expert selection by GGM [23], the proposed method does not assign a fixed set of local experts for all test points. A dynamic and flexible mechanism for each new observation designates related experts to provide local predictions. Multi-label classification [25] is a generalization of multi-class classification, where multiple labels may be assigned to each instance. It originates from the investigation of the text categorization problem, where each document may belong to several predefined topics simultaneously.

To transform the distributed learning case into a multi-label classification, the indices of the partitions/experts are the labels/classes. The task is to assign some experts to a new data point. Multi-classification problem would select an appropriate expert for predicting and would lead to a local approximation with only one expert per test point. This one-expert inductive model, however, produces discontinuous separation boundaries between sub-regions and therefore is not a proper solution for quantifying uncertainties [18], [26].

Two algorithms can be adapted to assign experts to data points: k-nearest neighbors (KNN) and deep neural networks (DNN). For the first one, we use the centroid of the partition as a substitute for the corresponding local expert. By estimating the distance between a new entry point and the centroids, we can find its K nearest neighboring experts. Due to the properties of the Gaussian process experts, if a test point is close to a GP expert, the expert can provide a reliable prediction for that test point.

For the second approach, we train a neural network with a soft-max output layer and log-loss (i.e., cross-entropy loss) using the training points and their related partition index that shows the partition they belong to. After training the DNN, we send a new test point through the network, and the experts share a single set of hyperparameters at each subset are called experts that are trained jointly and conquer the distance between a new entry point and the centroids, a substitute for the corresponding local expert. By estimating the related log-marginal likelihood,

$$
\log p(y|X) = -\frac{1}{2}y^T \Sigma^{-1} y - \frac{1}{2} \log |\Sigma| - \frac{N}{2} \log 2\pi,
$$

where $Z = k(X, X) + \sigma^2 I$. Optimization task in (1) scales as $O(N^3)$ because it requires to calculate the inversion of the $N \times N$ matrix $Z$. It inflicts limitations on the scalability of GPs, and the training step is time-consuming for large data sets.

### II. Problem Set-Up

#### A. Gaussian Process

We start with the basic non-linear regression problem $y = f(x) + \epsilon$, and the objective is to learn the latent function $f$ from a training set $\mathcal{P} = \{X, y\}$. Assume the training set contains $N$ observations, $X$ is a d-dimensional variable, $x \in \mathbb{R}^d$, and $\epsilon$ is a zero-mean Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma^2)$. The Gaussian process describes a prior distribution over the latent functions as $f \sim \mathcal{N}(0, k(x, x'))$, where $k(x, x')$ is the covariate function (kernel) with hyperparameters $\psi$, and $x, x' \in X$. The prior kernel is the well-known squared exponential (SE) covariance function equipped with automatic relevance determination (ARD),

$$
k(x, x') = \sigma^2 \exp\left(-\frac{1}{2} \sum_{i=1}^{d} \frac{(x_i - x'_i)^2}{T_i}\right),
$$

where $\sigma_i^2$ is the signal variance, and $T_i$ is a correlation length scale parameter along the $i$-th dimension. Let $\tau = \{\sigma^2, T_1, \ldots, T_d\}$, training the GP involves determining the hyperparameters $\theta = \{\sigma^2, \tau\}$ such that they maximise the related log-marginal likelihood.

The local predictive distribution of the i-th expert $\mathcal{E}_i$ on a test set $X^*$ of size $N_t$ is $\mathcal{P}(y^*|\mathcal{P}_i, X^*) \sim \mathcal{N}(\mu^*_i, \Sigma^*_i)$ with mean and covariance as:

$$
\mu^*_i = k^T_i(K_i + \sigma^2 I)^{-1}y_i,
$$

$$
\Sigma^*_i = k_{**} - k^T_i(K_i + \sigma^2 I)^{-1}k_{*i},
$$

where $K_i = k(X_i, X_i)$, $k_{**} = k(X_i, X^*)$, and $k_{*i} = k(X_i, X^*)$.

To divide the training data set $\mathcal{P}$ into $M$ partitions, two different strategies are used: random and disjoint partitioning. Although random partitioning is faster than disjoint partitioning, it has been widely accepted that disjoint partitioning can capture the local features of the data more accurately, see [23], [29]. Therefore, we assign training data to experts using a K-mean clustering approach in this work.

The structure of the paper is as follows. Section II introduces the problem formulation and related works. The proposed model and inference process are presented in Section III. Section IV discusses some associated details. Section V shows the experimental results, and we conclude in Section VI.
A DGP typically aggregates the local GP experts assuming perfect diversity between them which means they are conditionally independent, i.e., $E_i \perp E_j | y^*$ where $i, j \in \{1, \ldots, M\}$. Using CI assumption between experts $\{E\}_{i=1}^M$ allows us to factorize the predictive distribution of a DGP over all local predictive distributions. That is, for a test input $x^*$,

$$p(y^*|x^*) \propto \prod_{i=1}^M p_i(y^*|E_i, x^*).$$

Equation (4) shows that the aggregated predictive distribution can be defined as the product of local densities. The $\beta = \{\beta_1, \ldots, \beta_M\}$ describes the weights and influence of the experts. The most prevalent CI-based aggregation methods are product of experts (PoE) [30], generalization product of experts (GPoE) [31], Bayesian committee machine (BCM) [32], robust Bayesian committee machine (RBCM) [8], and generalized robust Bayesian committee machine (GRBCM) [29].

Figure 1 depicts the computational graph of the DGP strategy. It reveals the aggregation based on conditional independence assumption between experts $\{E_1, \ldots, E_{10}\}$. The CI assumption means two local experts $E_i$ and $E_j$ are connected only via the target variable $y^*$, i.e., $E_i \perp E_j | y^*$. Thus, there is no interaction between experts, and they can not affect each other.

### C. Beyond Conditional Independence Assumption

Ensemble methods extensively employ the conditional independence assumption for regression and classification problems [33], [34]. Although this assumption reduces the prediction cost of DGPs, it generally leads to a sub-optimal solution and their related predictions are not accurate enough [35]. In classification, modeling dependencies between classifiers has been considered in several works, for example, in [35]–[37]. However, few works have considered modeling expert dependencies in local approximate GPs. The nested point-wise aggregation of experts (NPAE) method [20], [21] defines an estimator using the interactions between experts and the target variable $y^*$.

For a given test point $x^* \in X^*$, assume the vector $\mu^*(x^*) = [\mu^*_1(x^*), \ldots, \mu^*_M(x^*)]^T$ contains the centered predictions of $M$ local GP experts $E = \{E_1, \ldots, E_M\}$, where $\mu^*_i(x^*)$, $i = 1, \ldots, M$ has been defined in (4). Each of the local Gaussian experts $E_i$ is a linear estimator because the related prediction $\mu^*_i$ is linear with respect to the observed values of the random variable $y_i$, i.e., $\mu^*_i = Q_i y_i$, where $Q_i = \frac{K_i}{\alpha^2 I + \sigma^2 I}$. Authors in [20] assumed that $y_i$ in (2) has not yet been observed. It allows us to consider $\mu^*_i(x^*)$ as a random variable. Therefore, the experts’ dependencies can be investigated in two ways; the correlations between the experts’ predictions and target variable, $\text{Cov}(\mu^*_i, y^*)$, and internal correlations between experts’ predictions, $\text{Cov}(\mu^*_i, \mu^*_j)$ where $i, j = 1, \ldots, M$. The analytical explanation of both covariances can be defined as:

$$\text{Cov}(\mu^*_i, y^*) = \text{Cov}(Q_i y_i, y^*) = Q_i k(X_i, X^*)$$

and

$$\text{Cov}(\mu^*_i, \mu^*_j) = \text{Cov}(Q_i y_i, Q_j y_j) = Q_i k(X_i, X_j) Q_j^T.$$

For a test point $x^* \in X^*$, the point-wise covariances are defined as $r(x^*) = \text{Cov}(\mu^*(x^*), y^*(x^*))$ and $R(x^*) = \text{Cov}(\mu^*(x^*), \mu^*(x^*))$, where $r(x^*)$ is a $1 \times 1$ vector and $R(x^*)$ is a $M \times M$ matrix. The joint distribution of random variables $(y^*, \mu^*_1, \ldots, \mu^*_M)$ is a multivariate normal distribution. This arises from the fact that every vector of linear combinations of normally distributed observations is itself a Gaussian vector. This fact is used to define the predictor $y^*_A(x^*)$ of $y^*(x^*)$ which aggregates variables $\mu^*_i(x^*)$, $i = 1, \ldots, M$, and leads to the subsequent aggregation:

**Definition 1 (Dependency-Based Aggregation).** The aggregated predictor for the test point $x^*$ and local predictions $\mu^*_1(x^*), \ldots, \mu^*_M(x^*)$ is defined as

$$y^*_A(x^*) = r(x^*)^T R(x^*)^{-1} \mu^*(x^*).$$

The NPAE estimator in (7) provides high-quality predictions. It is straightforward to show that this linear estimator is the best linear unbiased predictor (BLUP), see [20].

**Example 1 (Concrete Data Set).** Concrete Compressive Strength [3] data set contains 1030 observations of 9 attributes (8 independent variables and one response variable). We use 90% of the observations for training and the rest for testing, where disjoint partitioning is used to divide the data set into 5, 7, and 10 subsets. The prediction quality of CI and dependency-based aggregations are compared with standard GP. The quality of predictions is evaluated in two ways, standardized mean squared error (SMSE) and the mean standardized log loss (MSLL). The SMSE measures the accuracy of the prediction mean, while the MSLL evaluates the quality of predictive distribution [1].

Figure 2 shows the prediction quality of available baselines for the Concrete data set when the number of experts (M) changes. NPAE, which uses the dependencies between experts, provides consistently better results, confirming that modeling the experts’ interactions improves the quality of the aggregated predictive distribution. Especially, the quality of NPAE is not sensitive to changes in the number of experts (M). On the other hand, increasing M (reducing the size of the sub-partitions) lowers the prediction accuracy of existing CI-based DGP methods.

[https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength](https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength)
randomly partitioned. Sometimes yields poor results, particularly when the data is
uses the CI-based aggregation in the RBCM method and under some mild assumptions, can provide consistent results
between the base and other local experts, which, therefore the aggregation cost is \( O(N,M^3) \), which is not an efficient solution for complex real-world data sets with large \( M \) and \( N \) values.

In the next section, we propose a new expert selection approach using the multi-label classification model to assign test points to some proper experts instead of using the full experts set that modifies the aggregation estimator in [7].

III. EXPERT SELECTION IN LOCAL APPROXIMATION GPS

The current DGP baselines rely on experts’ weighting to quantify the experts’ importance. In [3], the authors discussed different weights for local experts and came to the conclusion that they can not outperform the linear mixture weights based on experts’ correlations defined in [7]. However, despite the high accuracy of the NPAE aggregation, its computational cost is a challenge for using the method on large data sets.

Expert selections can improve the performance of dependency-based aggregation in two ways. First, unrelated experts for a given data point can be excluded and only informative partitions can be considered to make the prediction. Second, mitigating the number of experts reduces the prediction cost and enables the ensemble to be used in large data sets.

A. Expert Selection Using Graphical Models

Gaussian graphical model (GGM) is the continuous form of pairwise Markov random fields. It assumes the nodes of an undirected graph are random variables, and the joint distribution of the random variables is multivariate Gaussian distribution with zero mean and precision matrix \( \Omega, N(0, \Omega^{-1}) \). The elements of the precision matrix are the unknown parameters and show interactions between experts (edges in the graph).

Let \( S \) be the sample covariance of local predictions \( \mu \), i.e. \( S = \text{Cov}(\mu) \). Then, the log-likelihood of the Gaussian multivariate distribution and precision matrix \( \Omega \) is equal to \( \log|\Omega| - \text{tr}(S \Omega) \). To estimate the precision matrix, Graphical Lasso (GLasso) [38] [39] is an efficient inference algorithm that maximizes this likelihood subject to an element-wise \( l_1 \)-norm penalty on \( \Omega \). More precisely, the objective function is,

\[
\bar{\Omega} = \arg\max_{\Omega} \log|\Omega| - \text{tr}(S \Omega) - \lambda \|\Omega\|_1 ,
\]

where the estimated expert network is then given by the non-zero elements of \( \bar{\Omega} \).

Modeling the dependency in distributed learning by GGMs has been studied in [22], [40], where the precision matrix

\[
\text{SMSE} = \frac{1}{M} \sum_{i=1}^{M} (\tilde{y}_i - y_i)^2
\]

\[
\text{MSLL} = -\frac{1}{2M} \sum_{i=1}^{M} \log |\Omega_i| + \frac{1}{2} \sum_{i=1}^{M} \text{tr}(S_i \Omega_i) + \frac{1}{2} \lambda \|\Omega_i\|_1
\]

Fig. 2: Ablation experiment1. Prediction quality of different aggregation baselines for the Concrete data set.

D. Asymptotic Properties

Conventional DGP baselines suffer from inconsistency. Since the local experts are trained on separated partitions, the aggregation produces inconsistent predictions that can not converge to the standard GP. Several works have investigated the asymptotic properties of the CI-based ensembles and confirmed the inconsistent and overconfident predictions of the PoE and (R)BCM methods. Besides, the GPoE with normalized equal weights [8] conservatively converges to the full GP distribution when \( N \to \infty \) [19], [21]. However, the authors in [29] showed that the GPoE produces consistent predictions using random partitioning under some mild assumptions.

The generalized robust Bayesian committee machine (GR-BCM) [29] introduces a base (global) expert and considers the covariance between the base and other local experts, which, under some mild assumptions, can provide consistent results using both random and disjoint partitioning. However, it still uses the CI-based aggregation in the RBCM method and sometimes yields poor results, particularly when the data is randomly partitioned.

The point-wise NPAE method is capable of providing consistent results. It benefits from both dependencies forms in Equations (5) and (6), and the aggregated predictor in (7) produces high-quality predictions employing the properties of conditional Gaussian distribution. Estimating the inverse of the internal correlation \( R(x^*) \) leads to two issues: the existence of the inversion matrix and computational cost. Using matrice’s pseudo-inverse can solve the first issue, but the second complicates employing the NPAE for large data sets. Calculating the inverse of the \( M \times M \) matrix \( R(x^*) \) has cubic time complexity in the number of local experts at each test point \( x^* \in X^* \). Therefore the aggregation cost is \( O(N,M^3) \), which is not an efficient solution for complex real-world data sets with large \( M \) and \( N \) values.
of selected experts is defined as \( K \) set. MI importance set selection task uses the first \( K \) experts in the descending sorted partitions, i.e., one for each expert, and the experts’ predictions concrete estimations. The unbiased estimator in (7) [23].

GGM-based aggregation can provide consistent results, and are fixed and used for prediction at any new entry point. The experts selected for the final aggregation. The experts in \( M \) a hyperparameter that indicates the percentage of original experts. Besides, expert selection by GGM has been proposed GGM-based related expert selection encodes the interactions between experts. The authors in [40] used the GLasso algorithm to detect dependencies between Gaussian experts and identify clusters of strongly dependent experts. Besides, expert selection by GGM has been proposed and investigated in [23], [24], which divides the experts into important and unimportant experts and excludes the unimportant experts in the final aggregation. The strength of an expert’s interactions in the related undirected graph defines the expert’s importance.

Definition 2 (GGM-related Expert Importance). The importance of expert \( E_i \) based on the estimated precision matrix \( \hat{\Omega} \) is defined as \( I_i = \sum_{j=1,j \neq i}^{M} |\hat{\Omega}_{ij}| \).

According to the interactions, the GGM-related expert selection task uses the first \( K \) experts in the descending sorted importance set \( I = \{ I_1, I_2, \ldots, I_M \} \), leading to a new expert set \( M_{GGM} = \{ M^1, \ldots, M^K \} \) and \( K < M \). The number of selected experts is defined as \( K = \alpha \times M \), where \( \alpha \) is a hyperparameter that indicates the percentage of original experts selected for the final aggregation. The experts in \( M_{GGM} \) are fixed and used for prediction at any new entry point. The GGM-based aggregation can provide consistent results, and its predictive distribution is a consistent approximation of the unbiased estimator in (7) [23].

Figure 3 depicts the related GGM of local experts’ predictions. The Concrete data set in Example 1 is divided into 10 partitions, i.e., one for each expert, and the experts’ predictions are used to quantify interactions between experts in this graph. Figure 3a presents the original GGM related to this distributed learning case. Figure 3b shows an expert selection scenario when only 60% of the experts are selected based on their importance. In [23], [24], the authors studied this selection method in CI-based baselines and reported remarkable improvements over state-of-the-art aggregation approaches in terms of prediction quality.

Algorithm 1 summarizes the aggregation procedure with GMM-based expert selection. Its primary input is given by the local predictions, meaning that this selection method is employed after individual experts’ predictions. Therefore, it does not depend on the entry points. Indeed, based on the Definition only absolute values of conditional dependencies are used for the importance calculation, i.e., \( |\hat{\Omega}| \), indicating that the importance is affected only by the amount of the dependency, and not its direction.

Although GGM-based expert selection provides an interpretable method, it suffers from two significant obstacles. First, it needs GLasso to obtain the precision matrix, and the cost of the GLasso is \( O(M^3) \). Hence, for massive data sets with large \( M \), its application is impractical. Besides, the algorithm is not flexible enough to capture the specific behavior of new data points because it provides a static method that selects a fixed set of experts for all test points. Even if an expert can provide accurate prediction for some part of the data set, it will be excluded from the model if it does not have high interactions with the other experts.

In the following subsections, we propose a novel approach that estimates the essential (i.e., data-related) experts for each new test point by converting the problem into a multi-label classification. The obtained labels for each test point define the data-point-wise selected experts.

B. Multi-label Classification for Flexible Expert Selection

Assigning experts to new entry points in a distributed learning model can be seen as a classification problem. Let’s assume that each expert is a class of estimators. The selection problem then for each test point \( x^* \) is defined as a multi-label classification task where each instance can be associated with some classes. The main advantage of this method is its flexibility because the selected experts depend on the given test

Algorithm 1 Aggregating Dependent Experts Using GGM

Input: Local predictions of GP experts \( \mu^* \), sparsity hyperparameter \( \lambda \), selection percentage \( \alpha \).

Output: Aggregated estimator \( y^*_A \).

1: Calculate sample covariance \( S \) of experts’ predictions.
2: Estimate \( \hat{\Omega} \) using GLasso (8).
3: Calculate the importance values \( \{ I_1, I_2, \ldots, I_M \} \)
4: Sort the importance values to find \( I \) as defined in Definition 2.
5: Select \( \alpha \) percent of most important experts.
6: Create the expert set for Aggregation, \( M^G \).
7: Aggregate selected experts \( M^G \) using (7).
point, and thus different experts can be assigned to different test points.

Assume \( x^* \) is a new test point and \( \mathcal{E} = \{ \mathcal{E}_1, \ldots, \mathcal{E}_M \} \) is the Gaussian experts set, and \( \mathcal{L} = \{ 1, \ldots, M \} \) is the label set. The task is to find \( \mathcal{M}^g(x^*) = \{ \mathcal{M}^g_1(x^*), \ldots, \mathcal{M}^g_K(x^*) \} \) that represents \( K \) selected experts to predict at \( x^* \). We adopt two prominent classification models to solve this multi-label task without requiring problem transformations, K-nearest neighbors (KNN) and conventional deep neural networks (DNN).

**Example 2 (Expert Selection Models).** Let’s consider an example with five local experts \( \mathcal{E} = \{ \mathcal{E}_1, \ldots, \mathcal{E}_5 \} \) that predict at 10 test points and \( K = 3 \). Figure 4 describes the difference between static and dynamic expert selection models in an example with synthetic data points. Figure 4(a) depicts the original aggregation where all experts are used, e.g., for the ensemble model in (7). The GGM-based expert selection in 4(b) proposes a fixed set of 3 experts \( \{ \mathcal{E}_2, \mathcal{E}_4, \mathcal{E}_5 \} \) for all new (i.e., 10) entry points even though they do not provide appropriate predictions in some of this 10 test points. The flexibility of the entry-based selection model is depicted in 4(c) where the model assigns different experts to each test point \( x^* \) and uses the ability of experts in a better way.

C. K-nearest neighbors (KNN)

The K-nearest neighbors algorithm \([41]\) is a lazy, non-parametric classification approach that uses proximity to classify an individual data point. It is a supervised machine learning algorithm, working off the assumption that similar data points are located near one another. Here, we adopt this algorithm for the experts’ assignment in a distributed learning scenario such that the raw training data set is not needed for the selection process and only the partitions’ information is used.

Let \( \mathcal{P}' = \{ \mathcal{P}_1, \ldots, \mathcal{P}_M \} \) be the partitions based on a disjoint partitioning strategy, i.e. K-Means clustering. Also, assume \( \mathcal{C} = \{ \mathcal{C}_1, \ldots, \mathcal{C}_M \} \) contains the related centroids of the clusters in \( \mathcal{P} \). For each test point \( x^* \), there is a \( 1 \times M \) vector \( \text{dist}(x^*, \mathcal{C}) \), in which the \( i \)th element is the distance between \( x^* \) and \( \mathcal{P}_i \), where \( \text{dist}(\cdot) \) is a distance metric. Therefore, the adopted KNN algorithm is defined as:

- calculate the distance between \( x^* \) and the centroids \( \text{dist}(x^*, \mathcal{C}) \)
- choose \( K \) experts with closest centroids to \( x^* \)
- return \( \mathcal{M}^g(x^*) \) based on the selected experts.

To determine which experts/partitions are closest to a given query point, the distance between the query point and the other data points will need to be calculated using a distance metric. The distance metric helps to form decision boundaries, which partition test points into different subsets/experts. Several distance measures can be chosen, e.g. Euclidean, Manhattan, Minkowski, and Hamming distances. In this work, we use the conventional Euclidean distance.

The value of \( K \) in the KNN algorithm defines how many clusters will be checked to determine the classification of a specific entry point. For example, if \( K = 1 \), the instance will be assigned to the same class as its nearest cluster, which due to discontinuity issues, is not the desired case. Lower values of \( K \) can have high variance, but low bias and larger values of \( K \) may lead to higher bias, and lower variance \([42]\).

Figure 5 schematically shows how the KNN method works.
used in machine learning problems, especially in classification.

**D. Neural Networks for Classification**

Conventional deep neural networks (DNNs) are widely used in machine learning problems, especially in classification tasks [43], [44]. By converting the expert selection task into a multi-label classification task, this supervised learning problem can be solved through DNNs. The capability of the neural networks can compensate for the possible weaknesses of KNN classifiers in dealing with high-dimensional data sets and underlying dependencies between labels.

Multi-label classification can be supported directly by neural networks simply by specifying the number of target labels in the problem as the number of nodes in the output layer. We will define a Multi-layer Perceptron (MLP) model for the multi-label classification task described in subsection III-B. The network requires an input layer that expects $D$ inputs to specify the dimension of $X$, $H$ nodes in the hidden layers, and $M$ nodes in the output layer, indicating the number of experts.

Each node in the output layer must use the softmax activation to predict the label’s class membership probability. Finally, the model must fit with the binary cross-entropy loss function and the Adam version of stochastic gradient descent.

To consider the expert selection task as a multi-label classification, a label set $\mathcal{L} = \{1, \ldots, M\}$ contains required classes related to the training data set. For each $x_i \in X$, $i = 1, \ldots, N$, the related partition label $l_i \in \mathcal{L}$ is available as an output of the training step in DGP. Therefore, instead of the original training set $(X, y)$, a new set of points and labels $(X, \mathcal{L})$ is constructed for training the DNN. After that, for each test point $x^* \in X^*$ the network will provide a probability vector $P^\mathcal{L}(x^*)$ where $P^\mathcal{L}(x^*)_j$ represents the probability that $x^*$ belongs to the $j$’th expert. The $K$ partitions with highest probabilities in $P^\mathcal{L}(x^*)$ are assigned to $x^*$.

Figure 6 depicts a simple network for a multi-label classification task that assigns local approximation Gaussian process experts to new entry points. The network receives training set $X$ as input and their clusters’ indices as output. After training the network, test points $x^* \in X^*$ are sent to the neural network, which returns the classifier’s raw output values.

Figure 7 depicts the prediction quality of expert selection methods on the Concrete data set with 10 partitions for

---

**Algorithm 2 Aggregating Dependent Experts Using KNN**

**Input:** Test point $x^* \in X^*$, centroids set $\mathcal{C}$, hyperparameter $K$, Local GPs moments, distance metric.

**Output:** Aggregated estimator $y^*_A(x^*)$

1. Calculate distance vector for $x^*$, i.e. $\text{dist}(x^*, \mathcal{C})$.
2. Sort the elements of $\text{dist}(x^*, \mathcal{C})$ ascendingly.
3. Select the first $K$ experts in the sorted list of expert distances to generate the set of related experts $\mathcal{M}^E(x^*)$.
4. Estimate local GPs by the experts in $\mathcal{M}^E(x^*)$ using (3) and (4).
5. Aggregate local predictions from Step 4 using (7).

---

**Figure 5:** Adopted K-nearest neighbors for multi-label classification.

**Figure 6:** Deep Neural Network Architecture for adopted multi-label classification in DGPs.
Algorithm 3 Aggregating Dependent Experts Using DNN

Input: Test point $x^* \in X^*$, training points $X$ and their indices, index set $L$, hyperparameter $K$, Local GPs moments.

Output: Aggregated estimator $y_A^*(x^*)$

1: Train the network parameters using $X$ and indices.
2: Return the classifier’s output values $P^L(x^*)$, i.e., as produced by the softmax layer.
3: Sort the elements of $P^L(x^*)$ descendingly.
4: Select the first $K$ indices of the sorted $P^L(x^*)$.
5: Create the expert set for $x^*$, $M^C(x^*)$.
6: Estimate local GPs by the experts in $M^C(x^*)$ using (2) and (3).
7: Aggregate local predictions from Step 6 using (7).

IV. DISCUSSION

This section considers some specific aspects of the expert selection models.

A. Restrictive Assumptions

The GGM approach assumes that the nodes in the graph are random, and their joint distribution is Gaussian. This normality assumption leads to a Gaussian likelihood, and GLasso solves this optimization problem. This assumption is strong, and in some cases, it can be restrictive. Various solutions have been proposed to relax this assumption by considering the model as a nonparametric problem and solving after some smooth monotone transformation [45]–[48] at the cost of requiring higher time complexity. On the other hand, expert selection using multi-label classification does not need any distributional assumption. Therefore, it can be used as a general expert selection method in distributed/federated learning models and not only in the context of local approximation of GPs.

Besides, the classification-based expert allocation can also be considered a self-attention mechanism that implicitly captures relationships between data points. Recently, the explicit modeling of self-attention between all data points has been shown to boost the classification performance [49], [50]. In our case, to explain the dependencies between training and test points, the expert ensembles do not use the original training points. Instead, the final prediction benefits from the critical information of training data captured by the partitions’ centroids and the corresponding indices for KNN and DNN, respectively.

B. Computational Costs of Expert Selection Models

The aggregation cost in all three selection methods, i.e., GGM, KNN, and DNN, is $O(N_tK^3)$ where $K$ is the number of selected experts and $N_t$ is the number of test observations. However, their selection strategies lead to different computational costs. GGM in Algorithm 1 needs GLasso to estimate the precision matrix, and its computational cost is $O(M^3)$, where the $M$ is the number of initial experts and is challenging for large $M$. Indeed, the sparsity parameter can affect the cost such that choosing a smaller value for $\lambda$ leads to a dense graph with a more considerable computational cost.

The cost of the KNN approach 2 is obtained by considering the cardinality of the training set, which refers to the number of possible labels that a feature can assume, in our case $M$, the dimension of each sample, i.e., $D$, and also the hyperparameter $K$. The computation time for calculating the distances is usually negligible compared to the rest of the algorithm. However, we consider this aspect as well in the overall cost estimation. Algorithm 2 computes the distance between the new observation and each centroid point, requiring $O(MD)$ work for an iteration and therefore $O(KMD)$ work overall to select $K$ closest centroids.
The cost of the DNN approach in Algorithm 3 depends on the network structure, i.e., the number of layers \( L \), the input dimension \( D \), the output dimension \( M \), and the number of hidden units. Let \( U_i \) represent the number of units in the \( i \)th layer (\( i = 1, \ldots, L \)), where \( U_1 \) and \( U_L \) represent the number of units in the input and output layers, respectively. The computational complexity is thus \( \mathcal{O}(N(U_1U_2 + \ldots + U_{L-1}U_L)) \).

In conclusion, both methods described in Algorithms 2 and 3 have linear complexity concerning \( M \). Therefore, they are more efficient when the number of partitions is an enormous value, unlike the cubical dependency in Algorithm 1.

C. Activation Functions in DNN

Using a softmax output layer for the DNN-based classification in Section III-D leads to a probability vector \( P^L(x^*) \) of the output values. Hence, when the probability of one class increases, the probability of at least one of the other classes has to decrease by an equivalent amount. Since the labels represent the interdependent experts, using the softmax function for the classification layer is reasonable.

Figure 8 explains how different activation functions can affect the prediction quality. It considers the Concrete data set with \( M = 10 \) and different mini-batch sizes \( N_b = \{16, 25, 50, 100\} \). Both activation functions softmax and sigmoid have been used to select \( K = 4 \) and \( K = 6 \) experts. The quality of the related aggregations confirms that due to the interaction between labels, the softmax activation leads to much better results, and its sensitivity to the size of the mini-batches is lower than for the sigmoid activation.

D. Expert Selection for CI-Based Baselines

In [23], the authors extended the GGM-based expert selection for CI-based ensembles. They introduce an extra step aiming to exclude the unimportant experts from the model before using the weight parameters \( \beta \). The same procedure can be used for entry-based expert selection methods. In this case, for a test point \( x^* \) in [4], only \( K \) experts are used, and the selection is based on the Algorithm 2 or Algorithm 3. Since these models are fast, the selection parameter \( K \) can be also set to relatively large values.

| Model | Expert Selection | SMSE | MSLL | Time (s) |
|-------|------------------|------|------|---------|
| GPoE  | KNN              | 0.138| -0.876| 0.03    |
| RBCM  | KNN              | 0.0993| 0.396| 0.03    |
| GRBCM | KNN              | 0.1093| -1.103| 0.06    |

Table I describes the effect of the selection scenario on CI-based ensembles using KNN on the Concrete data set with \( M = 10 \) and \( K = 6 \). Although this modification can not improve the asymptotic properties of the baselines, it raises their prediction quality. At the same time, the running times of both original and modified models are indistinguishable.
In this section, we investigate the influence of hyperparameters on the prediction quality and computational cost of the proposed methods and available baselines. First, we consider the aggregations of dependent experts with the selection step using a synthetic one-dimensional data set. Then, we use a medium-scale real-world data set to study how hyperparameters affect the results in a complex multi-dimensional data set.

1) Synthetic Example: The first experiment evaluates the effect of hyperparameters $M$ and $K$ on prediction quality and computation time in different selection scenarios. It is based on simulated data of a one-dimensional analytical function $f(x)$,

$$f(x) = 5x^2 \sin(12x) + (x^3 - 0.5)\sin(3x - 0.5) + 4\cos(2x) + \epsilon,$$

where $\epsilon \sim N(0,(0.2)^2)$. We generate $n$ training points in $[0,1]$, and $N_l = 0.1n$ test points in $[-0.2,1.2]$. The data is normalized to zero mean and unit variance. We vary the number of experts to consider different partition sizes. The $K$-means method is used for the partitioning to compare the prediction quality of the proposed selection methods with other baselines. Since the quality of CI-based methods is low, they are excluded from these experiments.

Figure 9 depicts the 99% confidence interval of NPAE, expert selection-based aggregations, and the full Gaussian process. In the experiment, $n = 3 \times 10^3$ training data points from Equation (9) are used. The training set is divided into $M = 10$ partitions, i.e., partition size $m_0 = 300$, with $K$-means clustering, and $K = 5$ agents are used for the final prediction. The confidence intervals of KNN and DNN are closer to the original baseline NPAE, and their predictions (mean of the predictive distribution) are close to the full GP. For test points out of the training set domain, the multi-label classification leads to accurate expert selection results; see for example the interval $[-0.2,0]$ in the GGM plot, which shows a significant deviation from the standard GP.

Figure 10 depicts the prediction quality of expert selection methods compared to NPAE for $3 \times 10^3$ training points and partition size $m_0 = 300$ (i.e., ten experts) with $K$-means partitioning. The x-axis shows the number of selected experts ($K$) used in the final aggregation. We vary the number of selected experts for selection methods and use the NPAE as a baseline that refers to $K = 10$. The plots in 10a and 10b indicate that multi-label classification leads to a selection
strategy with fast convergence to the original estimator.

When the number of experts increases, the prediction errors of KNN and DNN do not show significant changes, which is an indication of predictive stability. On the other hand, GGM needs more experts to provide closer results to NPAE and has a slow convergence procedure. For $K \geq 7$, the error values of all three methods are almost the same. Figure 10c indicates that the computational costs of the aggregations based on GGM, KNN, and DNN are approximately the same.

We evaluate the prediction quality of the expert selection methods using $n = 3 \times 10^3$ and $n = 5 \times 10^3$ training points and different numbers of experts, $M = \{10, 15, 20\}$. All related baselines are used in this experiment with $K = \{3, 5\}$ selected experts. Figure 11 depicts the results of both generated samples. KNN and DNN aggregations in both samples have remarkable prediction qualities, and their SMSE and MSLL values are close to each other, which means both classification methods return almost similar results.

On the other hand, when the ratio of the selected experts to the total number of experts $K_M = \frac{K}{M}$ decreases, the prediction quality of GGM-based models decreases drastically. For instance in Figure 11a, the differences between SMSE and MSLL become more significant as $M$ increases.
TABLE II: SMSE and MSLL of different baselines on the *Pumadyn* data set for different number of partitions strategies. Both dependent (D) and conditionally independent (CI) aggregation methods are used.

| Model   | Type | SMSE | MSLL     | SMSE | MSLL | SMSE | MSLL |
|---------|------|------|----------|------|------|------|------|
| GPOE    | CI   | 0.0487 | -1.5092 | 0.0489 | -1.5087 | 0.0501 | -1.4815 |
| GRBCM   | CI   | 0.049 | -1.5129 | 0.049 | -1.5083 | 0.0486 | -1.5133 |
| NPAE    | D    | 0.0462 | -1.5397 | 0.0473 | -1.5271 | 0.0470 | -1.5285 |
| GMM-5   | D    | 0.0477 | -1.5249 | 0.0485 | -1.5103 | 0.0481 | -1.5180 |
| GMM-7   | D    | 0.0471 | -1.5307 | 0.0481 | -1.5165 | 0.0478 | -1.5208 |
| KNN-5   | D    | 0.0467 | -1.5364 | 0.0477 | -1.5236 | 0.0475 | -1.5234 |
| KNN-7   | D    | 0.0462 | -1.5402 | 0.0474 | -1.5266 | 0.0470 | -1.5285 |
| DNN-5   | D    | 0.0465 | -1.5370 | 0.0477 | -1.5232 | 0.0476 | -1.5216 |
| DNN-7   | D    | 0.0460 | -1.5418 | 0.0475 | -1.5253 | 0.0470 | -1.5285 |

Fig. 12: Prediction Time (seconds) of different aggregation for disjoint partitioning in *Pumadyn* data set. The training data set is divided into 10, 15, and 20 subsets and a 50% and 70% of experts are selected for final aggregation.

values of *GGM-3* and *GGM-5* at *M* = 20 are almost twice the SMSE at *M* = 15. Indeed, GGM requires more experts to provide qualitative predictions, and the difference between the SMSE and MSLL of *GGM-3* and *GGM-5* indicates this fact. At the same time, the quality of KNN and DNN does not change significantly when *K* increases from 3 to 5.

Figures 11c and 11f show the running time of the baselines that consider the dependencies between experts. All expert selection-based aggregations share the same prediction process (of $O(NtK^3)$), differing only in the selection task. Besides enhancing the number of selected experts, *K* increases the computational cost of selection methods because it raises the prediction cost $O(NtK^3)$, see Section IV-B. In these experiments with smooth 1D data points, the running times of the GGM, the KNN, and the DNN are of the same rate.

2) Multi-Dimensional Real-World Data Set: The relative number of experts $K_M = \frac{K}{M}$ defined in Section V-A1 indicates the percentage of the initial experts selected to be used in the final predictive distribution. In this section, we use a medium-scale real-world data set and $K_M$ to appraise the efficacy of the data assignment strategy on the prediction quality. *Pumadyn* is a generated data set with 32 dimensions and 7,168 training points and 1,024 test points. The disjoint partitioning divides the data set into 10, 15, and 20 subsets. We consider the GPOE [8], GRBCM [29], NPAE [20], GGM-based aggregation [23], and proposed classification-based methods with K-means clustering. The penalty term $\lambda$ is 0.1 for GGM, and a neural network with a single hidden layer and 50 hidden units is used in this experiment.

The expert selection approaches based on GGM, KNN, and DNN, use $K_M = 0.5$ and $K_M = 0.7$, which means 50% and 70% of available experts are selected, respectively. Table II depicts the prediction quality of local approximation methods for the *Pumadyn* data set. The column Type shows

https://www.cs.toronto.edu/~delve/data/pumadyn/desc.html
the interactions between experts in the aggregation method, D for dependent experts, and CI for conditionally independent experts. The GGM-5, GGM-7, KNN-5, KNN-7, DNN-5, DNN-7, and NPAE are the dependency-based methods while GPoE and GRBCM are CI-based aggregations. The numbers after the names of the methods indicate the ratio. For instance, KNN-5 and KNN-7 refer to KNN with $K_M = 0.5$ and $K_M = 0.7$, respectively.

NPAE is a basis for comparison because it is the best linear unbiased predictor (BLUP). The multi-label classification methods provide accurate results, and their derivatives with $K_M = 0.5$ and $K_M = 0.7$ are close to the NPAE. This performance shows the fact that convergence occurs faster in these methods. However, the proficiency of the GGM method is sensitive to the number of agents and has more deviation from the NPAE when $K_M$ is small. Both KNN and DNN with 50% of the experts return appropriate approximations. They offer a significant improvement in prediction quality when $K_M = 0.7$, and for $M = 10$, they outperform the BLUP baseline, i.e., the NPAE method. It happens because the selection step properly excludes only weak experts at each test point.

Figure 12 depicts the running time of different aggregations with dependent experts for Pumadyn data set. The training data set is divided into $M = \{10, 15, 20\}$ partitions, and the prediction time of the related baselines is compared in two different cases, with 50% and 70% of original experts. In both cases, the NPAE method is used as a baseline that uses all dependent experts. Based on the plots, the prediction time of GGM, KNN, and DNN are almost at the same rate. However, Table III shows that GGM can not provide competitive prediction quality compared to the other selection methods. For instance, the SMSE and MSLL values of DNN-5 for all values of $M$ are lower than those of GGM-5 and GGM-7. This issue is also confirmed by Figures 10 and 11. The main reason for this issue is the low convergence rate of the GGM, which requires more experts.

B. Prediction Quality in Real-World Data Sets

In this section, we use real-world data sets to evaluate the prediction quality of proposed aggregations and compare with available baselines. The baselines that we use here are GPoE with uniform weights [8], RBCM [8], GRBCM [29], NPAE [20], GGM-based aggregation [23], KNN and DNN based ensemble methods. Various real-world data sets with different sizes and dimensions are used here which have been explained in Table III.

![Image](https://archive.ics.uci.edu/ml/datasets/airfoil+self-noise)

![Image](https://archive.ics.uci.edu/ml/datasets/parkinsons+telemonitoring)

![Image](https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure)

![Image](http://www.gaussianprocess.org/gpml/data/)

![Image](https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd)

We divide the observations in Airfoil, Parkinson, and Protein into training and test sets by extracting 85% of the sample as training and the rest as test points. In the other data sets, there are predefined training and test sets. Indeed, in Song data set, we extract the first $10^5$ songs from this data set for training and the first $10^4$ songs from the original test set for testing. We used disjoint partitioning (K-means) to divide the data sets into 5 (for Airfoil), 10 (for Parkinson and Pole Telecom), 70 (for Protein), 72 (for Sacros), and 80 (for Song) subsets.

Next, we compare SOTA baselines with classification-based aggregations. For the selection-based methods, we set $K_M$ to 0.5, which means 50% of experts are selected. For the Song data set only, we set $K_M = 0.2$ in. Since NPAE is computationally burdensome, especially when $M$ and $N_t$ are large, it is only used in small and medium-scale data sets. DNN uses a neural network with a hidden layer and 50 hidden units to evaluate the labels.

Tables IV and V reveal the SMSE and MSLL values of the baselines. Selecting the experts enables us to encode dependency between agents efficiently while the prediction quality is comparable with the original baseline NPAE. However, their running times are acceptable, especially when dealing with high-dimensional large data sets where using the NPAE is not feasible. Besides, the convergence rate of KNN and DNN is much faster than GGM. Even with 50% of experts, the results of the GGM still have a remarkable deviation from NPAE and cannot provide relevant results in the different data sets.

We consider Parkinson data set as an instance. GGM can not provide an accurate prediction when $K_M = 0.5$. Our experiments with $K_M = 0.7$ confirm that increasing the $K_M$ to 0.7 for GGM reduces the SMSE to 0.2208 and MSLL to ~0.8541 which are close to the SMSE and MSLL of NPAE. It indicates that GGM converges to NPAE, but the convergence is slower than KNN and DNN. Meanwhile, the SMSE values of the KNN and DNN methods for $K_M = 0.7$ are 0.2122 and 0.2117, respectively. Therefore, by including more experts in the final aggregation, KNN, and DNN can outperform the NPAE by excluding the effects of low-quality experts in Equation (7).

In CI-based methods, the conservative GPoE does not return acceptable results and can not outperform RBCM and GRBCM methods. The quality of the GRBCM is slightly better than RBCM because of the global communication expert. The global expert improves the quality measures of GRBCM compared to RBCM, especially in MSLL, where the values are always smaller the those of RBCM. Both methods provide competitive results with GGM when $K_M = 0.5$. However, their deviation from NPAE, KNN, and DNN is remarkable. Indeed, by increasing the $K_M$ GGM can easily outperform them.

### Table III: Real-World Data Sets

| Data Set      | (#) Observations | n  | $N_t$ | D  |
|---------------|------------------|----|-------|----|
| Airfoil       | 1,103            | 123| 300   | 5  |
| Parkinson     | 5,875            | 875| 20    |
| Pole Telecom  | 15,000           | 5,000| 5,000| 26 |
| Protein       | 45,730           | 45,730| 5,730| 9  |
| Sacros        | 48,938           | 44,489| 4,449| 21 |
| Song          | 515,345          | 463,715| 51,630| 91 |
TABLE IV: SMSE for various methods on real-world data sets. The table depicts SMSE values for SOTA baselines and the classification-based aggregations, i.e., KNN, and DNN. Both dependent (D) and conditionally independent (CI) aggregation methods are used.

| MODEL       | Airfoil | Parkinson | Pole Telecom | Protein | Sacros | Song |
|-------------|---------|-----------|--------------|---------|--------|------|
| GPoE (CI)   | 0.1305  | 0.2703    | 0.0727       | 0.8654  | 0.0461 | 0.9221|
| RBCM (CI)   | 0.0881  | 0.2339    | 0.0237       | 0.3569  | 0.0039 | 0.8127|
| GRBCM (CI)  | 0.0777  | 0.2395    | 0.0191       | 0.3540  | 0.0034 | 0.7762|
| NPAE (D)    | 0.0694  | 0.2121    | 0.0144       | -       | -      | -    |
| GMM-5 (D)   | 0.0765  | 0.2317    | 0.0182       | 0.3326  | 0.0025 | 0.7379|
| KNN-5 (D)   | 0.0694  | 0.2126    | 0.0145       | 0.2743  | 0.0025 | 0.7007|
| DNN-5 (D)   | 0.0694  | 0.2127    | 0.0141       | 0.2744  | 0.0025 | 0.6994|

TABLE V: MSLL for various methods on real-world data sets. The table depicts MSLL values for SOTA baselines and the classification-based aggregations, i.e., KNN, and DNN. Both dependent (D) and conditionally independent (CI) aggregation methods are used.

| MODEL       | Airfoil | Parkinson | Pole Telecom | Protein | Sacros | Song |
|-------------|---------|-----------|--------------|---------|--------|------|
| GPoE (CI)   | -1.1875 | -0.5862   | -1.5171      | -0.0759 | -1.165 | -0.0449|
| RBCM (CI)   | -1.3187 | -0.5433   | -1.5901      | -0.6164 | -2.5347 | -0.1266|
| GRBCM (CI)  | -1.4706 | -0.8123   | -2.293       | -0.6378 | -2.7985 | -0.1563|
| NPAE (D)    | -1.5207 | -0.8583   | -2.3537      | -       | -      | -    |
| GMM-5 (D)   | -1.4928 | -0.7937   | -2.1658      | -0.6173 | -2.8017 | -0.1613|
| KNN-5 (D)   | -1.5209 | -0.8563   | -2.3851      | -0.7348 | -2.8005 | -0.1913|
| DNN-5 (D)   | -1.5208 | -0.8569   | -2.3823      | -0.7349 | -2.8023 | -0.1926|

VI. CONCLUSION

In this work, we have proposed a novel expert selection approach for distributed learning with Gaussian agents, which leverages expert selection to aggregate dependent local experts’ predictions. The available ensemble baselines use all correlated experts in the aggregation step. It affects the final predictions by the local predictions of weak experts or leads to impractically high computational costs. Our proposed approach uses a multi-label classification model, and the allocation of data points to experts is defined by considering the experts as class labels.

Unlike the available expert selection method that assigns a fixed and static set of the selected experts to all new data points, the proposed model is more flexible to the model and data changes and chooses a related group of experts at each entry point. Excluding unrelated experts at each test point improves the prediction quality and reduces computational costs. Meanwhile, it keeps the original baseline’s asymptotic properties that use all exerts and provides consistent results when \( n \to \infty \). The classification methods in this work, i.e., KNN and DNN, can be replaced with recent and more efficient solutions proposed to solve the multi-label classification problem. The proposed approach can be used for distributed and federated learning and does not impose restricted assumptions. Through empirical analyses, we illustrated the superiority of our approach, which improves the prediction quality of existing SOTA aggregation methods while being highly efficient.

REFERENCES

[1] C. E. Rasmussen and C. K. Williams, Gaussian processes for machine learning. MIT Press, 2006.
[2] V. L. Deringer, A. P. Bartók, N. Bernstein, D. M. Wilkins, M. Ceriotti, and G. Csányi, “Gaussian process regression for materials and molecules,” Chemical Reviews, vol. 121, no. 16, pp. 10 073–10 141, 2021.
[3] A. Zeng, H. Ho, and Y. Yu, “Prediction of building electricity usage using gaussian process regression,” Journal of Building Engineering, vol. 28, no. 11, p. 101054, 2020.
[4] A. Denzel and J. Kästner, “Gaussian process regression for geometry optimization,” Journal of Chemical Physics, vol. 148, no. 9, p. 094114, 2018.
[5] R. Gramacy, Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences. Chapman and Hall/CRC, 2020.
[6] T. Le, K. Nguyen, V. Nguyen, T. D. Nguyen, and D. Phung, “Gogp: Fast online regression with gaussian processes,” IEEE International Conference on Data Mining, pp. 257–266, 2017.
[7] F. Tobar, T. D. Bui, and R. E. Turner, “Learning stationary time series using gaussian processes with nonparametric kernels,” in Advances in Neural Information Processing Systems, pp. 3501–3509, 2015.
[8] M. P. Deisenroth and J. W. Ng, “Distributed gaussian processes,” International Conference on Machine Learning, pp. 1481–1490, 2015.
[9] M. K. Titsias, “Variational learning of inducing variables in sparse gaussian processes,” *Artificial Intelligence and Statistics*, pp. 567–574, 2009.

[10] J. Hensman, N. Fusi, and N. D. Lawrence, “Gaussian processes for big data,” *Uncertainty in Artificial Intelligence, AUA Press*, pp. 282–290, 2013.

[11] C. Cheng and B. Boots, “Variational inference for gaussian process models with linear complexity,” *Advances in Neural Information Processing Systems 30*, pp. 5184–5194, 2017.

[12] D. R. Burt, C. E. Rasmussen, and M. van der Wilk, “Rates of convergence for sparse variational gaussian process regression,” *ICML*, 2019.

[13] T. D. Bui and R. E. Turner, “Tree-structured gaussian process approximations,” *Advances in Neural Information Processing Systems*, pp. 2213–2221, 2014.

[14] D. Moore and S. J. Russell, “Gaussian process random fields,” In *Advances in Neural Information Processing Systems*, pp. 3357–3365, 2015.

[15] D. Peteiro-Barral and B. Guijarro-Berdiniás, “A survey of methods for distributed machine learning,” *Progress in Artificial Intelligence*, vol. 2, no. 1, pp. 1–11, 2013.

[16] J. Zhang, M. Wang, Q. Li, S. Wang, X. Chang, and B. Wang, “A survey on distributed machine learning,” *ACM Computing Surveys*, vol. 53, no. 2, p. 1–33, 2020.

[17] J. Liu, J. Huang, Y. Zhou, X. Li, S. Ji, H. Xiong, and D. Dou, “From distributed machine learning to federated learning: A survey,” *arXiv preprint arXiv:2104.14362*, 2021.

[18] X. S. H. Liu, Y. S. Ong and J. Cai, “When gaussian process meets big data: A review of scalable gpcs,” *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 11, pp. 4405–4423, 2020.

[19] B. Szabó and H. van Zanten, “An asymptotic analysis of distributed non-parametric methods,” *Journal of Machine Learning Research*, vol. 20, no. 87, pp. 1–30, 2019.

[20] D. Rullière, N. Durrande, F. Bachoc, and C. Chevalier, “Nested kriging predictions for datasets with a large number of observations,” *Statistics and Computing*, vol. 26, no. 4, pp. 849–867, 2016.

[21] F. Bachoc, N. Durrande, D. Rullière, and C. Chevalier. “Some properties of nested kriging predictors,” *Technical report hal-01561747*, 2017.

[22] H. Jalali and G. Kasneci, “Aggregating the gaussian experts’ predictions via undirected graphical models,” *IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 23–26, 2022.

[23] H. Jalali, M. Pawelczyk, and G. Kasneci, “Model selection in local approximation gaussian processes: A markov random fields approach,” *IEEE International Conference on Big Data (Big Data)*, pp. 768–778, 2021.

[24] ——, “Gaussian experts selection using graphical models,” *arXiv preprint arXiv:2102.01496*, 2021.

[25] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification*. Springer International Publishing, 2016, pp. 17–31.

[26] C. Park and D. Apley, “Patchwork kriging for large-scale gaussian process regression,” *Journal of Machine Learning Research*, vol. 19, no. 7, pp. 1–43, 2018.

[27] P. Blanchard, E. E. Mhamdi, R. Guerraoui, and J. Stainer, “Machine learning with adversaries: Byzantine tolerant gradient descent,” vol. 30, 2017.

[28] D. Data and S. Diggavi, “Byzantine-resilient high-dimensional sgd with local iterations on heterogeneous data,” vol. 139, 2021, pp. 2478–2488.

[29] H. Liu, J. Cai, Y. Ong, and Y. Wang, “Generalized robust bayesian committee machine for large-scale gaussian process regression,” *International Conference on Machine Learning*, pp. 1–10, 2018.

[30] G. E. Hinton, “Training products of experts by minimizing contrastive divergence,” *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[31] Y. Cao and D. J. Fleet, “Generalized product of experts for automatic and principled fusion of gaussian process predictions,” *arXiv preprint arXiv:1410.7027*, 2014.

[32] V. Tresp, “A bayesian committee machine,” *Neural Computation*, vol. 12, no. 11, pp. 2719–2741, 2000.

[33] J. Mendes-Moreira, C. Soares, A. Jorge, and J. Sousa, “Ensemble approaches for regression: A survey,” *Acm computing surveys (csur)*, vol. 45, no. 4, pp. 1–40, 2012.

[34] F. Parisi, F. Strino, B. Nadler, and Y. Kluger, “Ranking and combining multiple predictors without labeled data,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 4, pp. 1233–1238, 2014.

[35] A. Jaffe, E. Fetaya, B. Nadler, T. Jiang, and Y. Kluger, “Unsupervised ensemble learning with dependent classifiers,” *in Artificial Intelligence and Statistics*, pp. 351–360, 2016.