entropy

IMPACT FACTOR 2.0

Indexed in: PubMed

CITESCORE 5.2

Article

Multilabel Classification for Entry-Dependent Expert Selection in Distributed Gaussian Processes

Hamed Jalali and Gjergji Kasneci

Special Issue

Gaussian Fields and Their Application in Computational Engineering and Mathematical Physics

Edited by

Prof. Dr. Robertas Alzbutas, Prof. Dr. Mark Girolami and Dr. Hussein Rappel

MDPI

https://doi.org/10.3390/e27030307

entropy

MDPI

Article

Multilabel Classification for Entry-Dependent Expert Selection in Distributed Gaussian Processes

Hamed Jalali [1,*] and Gjergji Kasneci [2]

[1] Center for Plant Molecular Biology (ZMBP), University of Tübingen, 72076 Tuebingen, Germany
[2] School of Social Sciences and Technology, Technical University of Munich, 80333 Munich, Germany; gjergji.kasneci@tum.de
* Correspondence: hamed.jalali@uni-tuebingen.de

Abstract: By distributing the training process, local approximation reduces the cost of the standard Gaussian process. An ensemble method aggregates predictions from local Gaussian experts, each trained on different data partitions, under the assumption of perfect diversity among them. While this assumption ensures tractable aggregation, it is frequently violated in practice. Although ensemble methods provide consistent results by modeling dependencies among experts, they incur a high computational cost, scaling cubically with the number of experts. Implementing an expert-selection strategy reduces the number of experts involved in the final aggregation step, thereby improving efficiency. However, selection approaches that assign a fixed set of experts to each data point cannot account for the unique properties of individual data points. This paper introduces a flexible expert-selection approach tailored to the characteristics of individual data points. To achieve this, we frame the selection task as a multi-label classification problem in which experts define the labels, and each data point is associated with specific experts. We discuss in detail the prediction quality, efficiency, and asymptotic properties of the proposed solution. We demonstrate the efficiency of the proposed method through extensive numerical experiments on synthetic and real-world datasets. This strategy is easily extendable to distributed learning scenarios and multi-agent models, regardless of Gaussian assumptions regarding the experts.

Keywords: distributed learning; Gaussian processes; multi-agent systems; multi-label classification; conditional dependency; ensemble learning

Academic Editors: Robertas Alzbutas, Mark Girolami and Hussein Rappel

Received: 16 December 2024
Revised: 8 March 2025
Accepted: 12 March 2025
Published: 14 March 2025

Citation: Jalali, H.; Kasneci, G. Multilabel Classification for Entry-Dependent Expert Selection in Distributed Gaussian Processes. *Entropy* **2025**, *27*, 307. https://doi.org/10.3390/e27030307

1. Introduction

Gaussian processes (GPs), as described by Rasmussen [1], are interpretable and powerful Bayesian non-parametric methods for non-linear regression. A Gaussian process is a stochastic process in which any finite subset of random variables follows a multivariate Gaussian distribution. Using Bayes' theorem, the posterior predictive distribution of a GP is the best linear unbiased estimator (BLUE) under the assumed model, offering accurate quantification of prediction uncertainty. GPs do not require restrictive model assumptions and can effectively capture complex linear and non-linear relationships. Although GPs are widely applied in practical settings [2–7], their cubic training complexity and quadratic prediction costs (concerning the training set size) restrict their scalability to large-scale data problems [8].

The major computational hurdle for GP regression is the need to estimate the kernel inversion and determinant, which is prohibitively expensive when n is large. Due to this issue, GPs are typically restricted to relatively small training datasets in the range of $\mathcal{O}(10^4)$.

To reduce computational expense, sparse approximation techniques use a subset of the training sets (called inducing points) and Nyström approximation to estimate posterior distributions [9–13]. 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 datasets since its capacity is limited by the number of inducing points [14,15].

Unlike sparse approximation methods, which rely solely on inducing points, distributed Gaussian processes (DGPs) utilize the entire training set. This method employs centralized distributed learning, where the training data are divided into subsets, the local inference is performed on each subset independently, and the local estimations are aggregated through ensemble learning [16–19]. A local GP who specializes in a particular data partition is referred to as an expert. Sharing the same hyperparameters among experts contributes to implicit regularization and helps mitigate overfitting [8,20].

In a DGP, the conditional independence (CI) assumption between partitions (i.e., between experts given the target) allows the factorization of the global posterior distribution as a product of local distributions. Although this assumption reduces the computational cost, it introduces inconsistencies and leads to suboptimal solutions [21] caused by the partitioning of the dataset, such that when $N \to \infty$, the CI-based posterior approximations fail to 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 [22–24]. 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. This 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 [25,26], the 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 the 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 [25], 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 [27] 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 or experts are the labels or classes. The objective is to assign specific experts to a new data point. A multi-class 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 [20,28].

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 train 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 with higher probability are assigned to this test point. Relative to consistent aggregation methods that use dependency information, our approach keeps all asymptotic properties of the original baseline and substantially provides competitive prediction performance while leading to better computational costs than other SOTA approaches, which use the dependency assumption. By extending the proposed method for CI-based ensembles, we can use it in federated learning problems, which do not consider dependencies between agents, see [29,30].

The structure of the paper is as follows. Section 2 introduces the problem formulation and related works. The proposed model and inference process are presented in Section 3. Section 4 discusses some associated details. Section 5 shows the experimental results, and we conclude in Section 6.

2. Problem Set Up

2.1. 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, $x \in R^d$, and ϵ 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 ψ, and $x, x' \in X$. The prior kernel is the well-known squared exponential (SE) covariance function equipped with automatic relevance determination (ARD),

$$k(x, x') = s^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{d}\frac{(x_i - x_i')^2}{\mathcal{T}_i}\right),$$

where σ_f^2 is the signal variance, and $\mathcal{T}_i$ is a correlation length scale parameter along the i-th dimension. Let $\tau = \{s^2, \mathcal{T}_1, \ldots, \mathcal{T}_d\}$; training the GP involves determining the hyperparameters $\theta = \{\sigma^2, \tau\}$ such that they maximize the related log-marginal likelihood,

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

where $\mathcal{Z} = k(X, X) + \sigma^2 I$. The optimization task in (1) scales as $\mathcal{O}(N^3)$ because it require calculating the inversion of the $N \times N$ matrix $\mathcal{Z}$. It inflicts limitations on the scalability of GPs, and the training step is time-consuming for large datasets.

2.2. Local Approximation Gaussian Process

The local approximation Gaussian process is a *divide-and-conquer* approach, which partitions the training dataset into M subsets $\mathcal{P}' = \{\mathcal{P}_1, \ldots, \mathcal{P}_M\}$ and trains standard GPs

on these subsets. It is also called the distributed Gaussian process (DGP for short), which builds on distributing the training of the standard GP among several computing units. Let X_i and y_i be the input and output of subset $\mathcal{P}_i$. The related local GPs at each subset are called experts that are trained jointly and share a single set of hyperparameters $\theta = \{\sigma^2, \psi\}$ [8].

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

$$\mu_i^* = k_{i*}^T (K_i + \sigma^2 I)^{-1} y_i, \tag{2}$$

$$\Sigma_i^* = k_{**} - k_{i*}^T (K_i + \sigma^2 I)^{-1} k_{i*}, \tag{3}$$

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

To divide the training dataset $\mathcal{P}$ 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 [25,31]. Therefore, we assign training data to experts using a K-mean clustering approach in this work.

A DGP typically aggregates the local GP experts assuming perfect diversity between them, which means they are conditionally independent, i.e., $\mathcal{E}_i \perp\!\!\!\perp \mathcal{E}_j | y^*$ where $i, j \in \{1, \ldots, M\}$. Using CI assumption between experts $\{\mathcal{E}\}_{i=1}^M$ allows us to factorize the predictive distribution of a DGP over all local predictive distributions. That is to say, for a test input x^*

$$p(y^*|\mathcal{P}, x^*) \propto \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{P}_i, x^*). \tag{4}$$

Equation (4) shows that the aggregated predictive distribution can be defined as the product of local densities. The $\beta = \{\beta_1, \ldots, \beta_M\}$ describe the weights and influence of the experts. The most prevalent CI-based aggregation methods are the product of experts (PoE) [32], generalized product of experts (GPoE) [33], Bayesian committee machine (BCM) [34], robust Bayesian committee machine (RBCM) [8] and generalized robust Bayesian committee machine (GRBCM) [31].

Figure 1 depicts the computational graph of the DGP strategy. It reveals the aggregation based on the conditional independence assumption between experts $\{\mathcal{E}_1, \ldots, \mathcal{E}_{10}\}$. The CI assumption means two local experts $\mathcal{E}_i$ and $\mathcal{E}_j$ are connected only via the target variable y^*, i.e., $\mathcal{E}_i \perp\!\!\!\perp \mathcal{E}_j \mid y^*$. Thus, there is no interaction between experts, and they cannot affect each other.

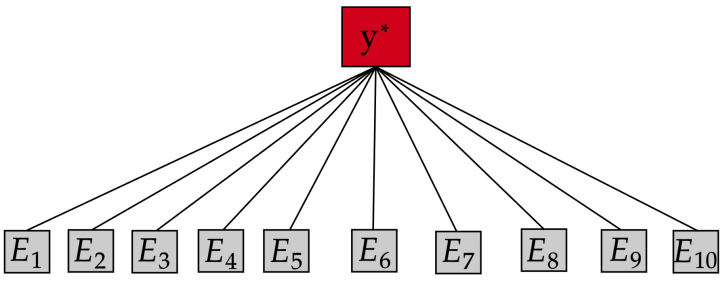


Figure 1. Computational graphs of an aggregation based on the conditional independence assumption between experts.

2.3. Beyond Conditional Independence Assumption

Ensemble methods extensively employ the conditional independence assumption for regression and classification problems [35,36]. 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 [37]. In classification, modeling dependencies between

classifiers has been considered in several works, for example, in [37–39]. However, few works have considered modeling expert dependencies in local approximation GPs. The nested pointwise aggregation of experts (NPAE) method [22,23] 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 $\mathcal{E} = \{\mathcal{E}_1, \ldots, \mathcal{E}_M\}$, where $\mu_i^*(x^*)$, $i = 1, \ldots, M$ has been defined in (2). Each local Gaussian expert, $\mathcal{E}_i$, is a linear estimator because its associated prediction, μ_i^*, is linear with respect to the observed values of the random variable y_i, i.e., $\mu_i^* = Q_i y_i$, where $Q_i = k_{i*}^T (K_i + \sigma^2 I)^{-1}$. Authors in [22] assumed that y_i in (2) has not yet been observed. This 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, $Cov(\mu_i^*, y^*)$, and internal correlations between experts' predictions, $Cov(\mu_i^*, \mu_j^*)$, where $i, j = 1, \ldots, M$. The analytical explanation of both covariances can be defined as:

$$Cov(\mu_i^*, y^*) = cov(Q_i y_i, y^*) = Q_i k(X_i, X^*) \tag{5}$$

$$Cov(\mu_i^*, \mu_j^*) = cov(Q_i y_i, Q_j y_j) = Q_i k(X_i, X_j) Q_j^T \tag{6}$$

For a test point $x^* \in X^*$, the point-wise covariances are defined as $r(x^*) = Cov(\mu^*(x^*), y^*(x^*))$ and $R(x^*) = Cov(\mu^*(x^*), \mu^*(x^*))$, where $r(x^*)$ is an $M \times 1$ vector and $R(x^*)$ is an $M \times M$ matrix. The joint distribution of random variables $(y^*, \mu_1^*, \ldots, \mu_M^*)$ is a multivariate normal distribution. This issue comes from the fact that any 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*

$$y_A^*(x^*) = r(x^*)^T R(x^*)^{-1} \mu^*(x^*). \tag{7}$$

This method is called the nested pointwise aggregation of experts (NPAE) and provides high-quality predictions. It is straightforward to show that this linear estimator is the *best linear unbiased predictor* (BLUP), see [22].

Example 1 (**Concrete Dataset**). *The Concrete Compressive Strength (available at: https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength, accessed on 10 March 2022) dataset 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 dataset into 5, 7, and 10 subsets. The prediction quality of CI and dependency-based aggregations is 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* dataset 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. The quality of NPAE is especially 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.

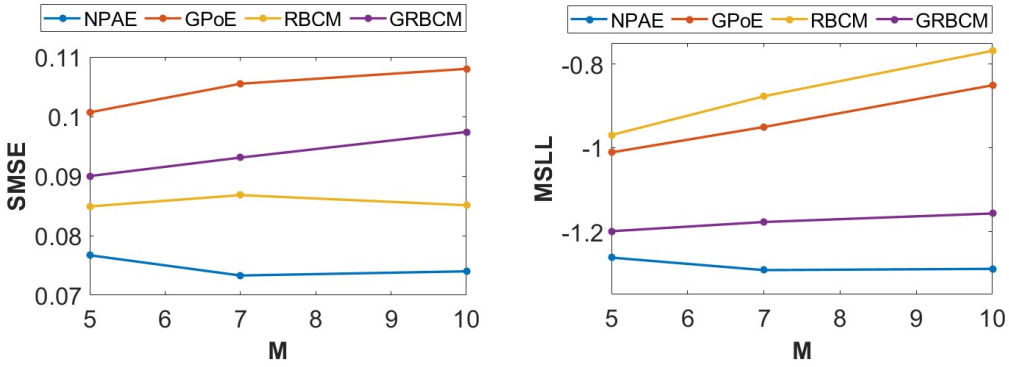


Figure 2. Ablation experiment. Prediction quality of different aggregation baselines for the *Concrete* dataset. (**a**) SMSE. (**b**) MSLL.

2.4. Asymptotic Properties

Conventional DGP baselines suffer from *inconsistency*. Since the local experts are trained on separate partitions, the aggregation produces inconsistent predictions that cannot converge to the standard GP. Several studies have investigated the asymptotic properties of the CI-based ensembles and confirmed the inconsistent and overconfident predictions of the PoE and (R)BCM methods. Moreover, the GPoE with normalized equal weights [8] conservatively converges to the full GP distribution as $N \to \infty$ [21,23]. However, the authors of [31] demonstrated that the GPoE produces consistent predictions using random partitioning under certain mild assumptions.

The generalized robust Bayesian committee machine (GRBCM) [31] 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 are randomly partitioned.

The point-wise NPAE method is capable of providing consistent results. It benefits from both dependency 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 a matrix's pseudo-inverse can solve the first issue, but the second complicates employing the NPAE for large datasets. 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 $\mathcal{O}(N_t M^3)$, which is not an efficient solution for complex real-world datasets with large M and N_t 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 expert set that modifies the aggregation estimator in (7).

3. Expert Selection in Local Approximation GPs

The current DGP baselines rely on weighting experts to quantify their importance. In [8], the authors analyzed various weighting schemes for local experts and concluded that none outperformed the linear mixture weights based on experts' correlations, as defined

in (7). However, despite the high accuracy of the NPAE aggregation, its computational cost poses a significant challenge for applying the method to large datasets.

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 datasets.

3.1. Expert Selection Using Graphical Models

The 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 Ω, $\mathcal{N}(0, \Omega^{-1})$. 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 μ^*, i.e., $S = Cov(\mu^*)$. Then, the log-likelihood of the Gaussian multivariate distribution and precision matrix Ω is equal to $\log|\Omega| - trace(S\Omega)$. To estimate the precision matrix, Graphical Lasso (GLasso) [40,41] is an efficient inference algorithm that maximizes this likelihood subject to an element-wise l_1-norm penalty on Ω. More precisely, the objective function is,

$$\widehat{\Omega} = \arg\max_{\Omega} \log|\Omega| - trace(S\Omega) - \lambda\|\Omega\|_1, \tag{8}$$

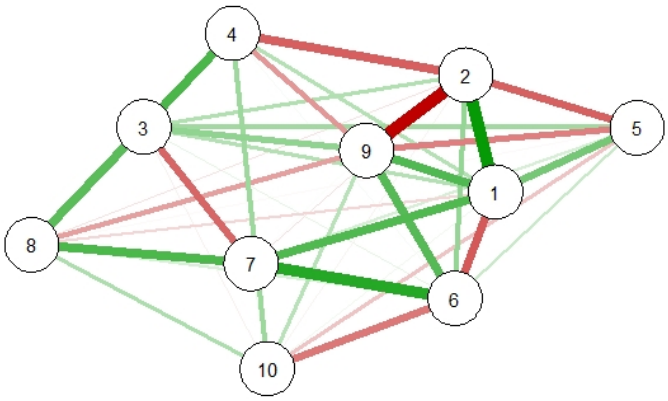
where the estimated expert network is then given by the non-zero elements of $\widehat{\Omega}$.

Modeling the dependency in distributed learning by GGMs has been studied in [24,42], where the precision matrix encodes the interactions between experts. The authors in [42] used the GLasso algorithm to detect dependencies between Gaussian experts and identify clusters of strongly dependent experts. In addition, expert selection by GGM has been proposed and investigated in [25,26], 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.

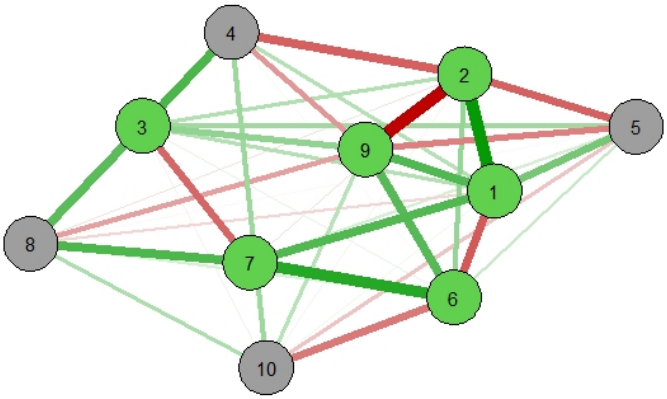
Definition 2 (GGM-related Expert Importance). *The importance of expert $\mathcal{E}_i$ based on the estimated precision matrix $\widehat{\Omega}$ is defined as $\mathcal{I}_i = \sum_{j=1, j\neq i}^{M} |\widehat{\Omega}_{ij}|$.*

According to the interactions, the GGM-related expert-selection task uses the first K experts in the descending sorted importance set $\mathcal{I} = \{\mathcal{I}_{i_1}, \mathcal{I}_{i_2}, \ldots, \mathcal{I}_{i_M}\}$, leading to a new expert set. $\mathcal{M}_{\mathcal{GGM}} = \{\mathcal{M}^{\mathcal{G}}{}_1, \ldots, \mathcal{M}^{\mathcal{G}}{}_K\}$ and $K < M$. The number of selected experts is defined as $K = \alpha \times M$, where α is a hyperparameter that indicates the percentage of original experts selected for the final aggregation. The experts in $\mathcal{M}_{\mathcal{GGM}}$ 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) [25].

Figure 3 depicts the related GGM of local experts' predictions. The *Concrete* dataset 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 [25,26], 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.



(**a**) GGM



(**b**) Selected Experts

Figure 3. Expert selectionusing GGM for a set of 10 local experts from the *Concrete* dataset: (**a**) the experts' graph and (**b**) selected experts based on 60% of most important experts (green nodes).

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 2, 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.

Algorithm 1 Aggregating Dependent Experts Using GGM

Require: Local predictions of GP experts μ^*, sparsity hyperparameter λ, selection percentage α.
Ensure: Aggregated estimator $y_A^\star$.
1: Calculate sample covariance S of experts' predictions.
2: Estimate $\hat{\Omega}$ using *GLasso* (8).
3: Calculate the importance values $\{\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_M\}$
4: Sort the importance values to find $\mathcal{I}$ as defined in Definition 2.
5: Select α percent of most important experts.
6: Create the expert set for Aggregation, $\mathcal{M}^{\mathcal{G}}$.
7: Aggregate selected experts $\mathcal{M}^{\mathcal{G}}$ using (7).

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 $\mathcal{O}(M^3)$. Hence, for massive datasets with large M, its application

is impractical. In addition, 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 dataset, 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.

3.2. 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 us 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 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 $\mathcal{M}^{\mathcal{C}}(x^*) = \{\mathcal{M}^{\mathcal{C}}{}_1(x^*), \ldots, \mathcal{M}^{\mathcal{C}}{}_K(x^*)\}$ which represents K selected experts to predict at x^*. We adapt 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 us 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 4a depicts the original aggregation where all experts are used, e.g., for the ensemble model in (7). The GGM-based expert selection in Figure 4b 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 these 10 test points. The flexibility of the entry-based selection model is depicted in Figure 4c, where the model assigns different experts to each test point x^* and uses the ability of experts in a better way.*

(**a**) $\mathcal{E}$ (**b**) $\mathcal{E}^{\mathcal{G}}$ (**c**) $\mathcal{E}^{\mathcal{C}}$

Figure 4. Expert-selection scheme of both static and entry-dependent models for a setting of 5 experts with 10 test points. Both selection models assign 3 experts (shown in red) to each test point: (**a**) original set of experts $\mathcal{E}$, (**b**) static assignment of experts $\mathcal{E}^{\mathcal{G}}$, and (**c**) entry-based selection of experts $\mathcal{E}^{\mathcal{C}}$.

3.3. Adopted K-Nearest Neighbors (KNN)

The K-nearest neighbors algorithm [43] 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 assignment of experts in a distributed learning scenario such that the raw training dataset 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 $dist(x^*, \mathcal{C})$, in which the ith element is the distance between x^* and $\mathcal{P}_i$, where $dist()$ is a distance metric. Therefore, the adopted KNN algorithm is defined as:

- Calculate the distance between x^* and the centroids $dist(x^*, \mathcal{C})$;
- Choose K experts with centroids closest to x^*;
- Return $\mathcal{M}^{\mathcal{C}}(x^*)$ 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 [44].

Figure 5 schematically shows how the KNN method works for the multi-label classification methods. It represents a KNN framework for a test point x^*. The red points are related training points assigned to each partition, and the blue points are the clusters' centroids. The lines between the x^* and the centroids show the distances. The proposed method suggests the orange lines, which are the shortest, and the related experts are assigned to this test point.

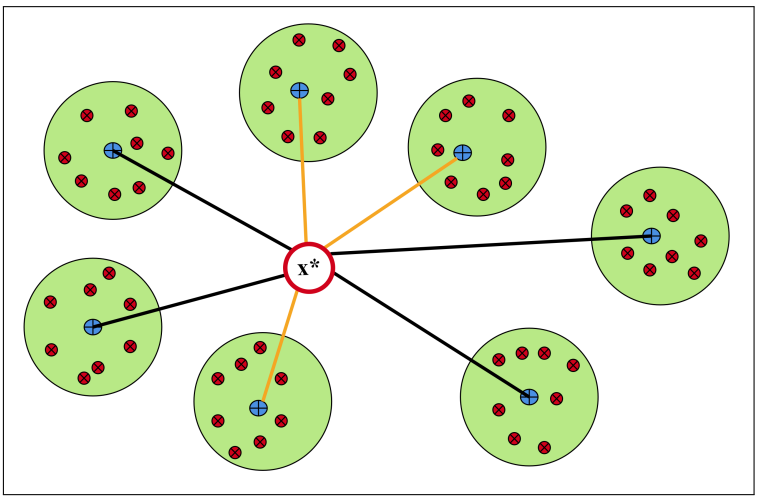


Figure 5. Adopted K-nearest neighbors for multi-label classification.

Algorithm 2 summarizes the whole procedure of the KNN-based aggregation. The main advantage of the KNN method is that it does not include another training period.

The only thing to be calculated is the distance between different points; therefore, it is straightforward to implement and accept new entry data at any time. In addition, instead of n training points, the modified KNN proposed in Algorithm 2 only uses the M centroids and scales the distance calculations in large datasets.

Algorithm 2 Aggregating Dependent Experts Using KNN

Require: Test point $x^* \in X^*$, centroids set $\mathcal{C}$, hyperparameter K, Local GPs moments, distance metric.
Ensure: Aggregated estimator $y_A^*(x^*)$
1: Calculate distance vector for x^*, i.e., $dist(x^*, \mathcal{C})$.
2: Sort the elements of $dist(x^*, \mathcal{C})$ ascendingly.
3: Select the first K experts in the sorted list of expert distances to generate the set of related experts $\mathcal{M}^{\mathcal{C}}(x^*)$.
4: Estimate local GPs by the experts in $\mathcal{M}^{\mathcal{C}}(x^*)$ using (2) and (3).
5: Aggregate local predictions from Step 4 using (7).

Generally, this selection method is defensible and justifiable because Gaussian processes predict better when a test point is close to them. Since the distance-based solution may have some drawbacks in high-dimensional datasets, we propose another multi-label classification solution in the next section that can be effective in high-dimensional cases.

3.4. Adapted Neural Networks for Classification

Conventional deep neural networks (DNNs) are widely used in machine learning problems, especially in classification tasks [45,46]. 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 datasets 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 Section 3.2. 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* or *sigmoid* 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 dataset. 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^*. This procedure is concisely presented in Algorithm 3 for clarity.

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 the 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* dataset with 10 partitions for different values of K. As can be seen, multi-label-based expert-selection models provide higher quality predictions with lower deviation from

the NPAE model. The quality of the classification-based aggregations changes less as the selection parameter K increases. The case $K = 10$ leads to the original NPAE baseline (the dashed green lines in Figure 7a,b), and both KNN and DNN return proper error values in both plots.

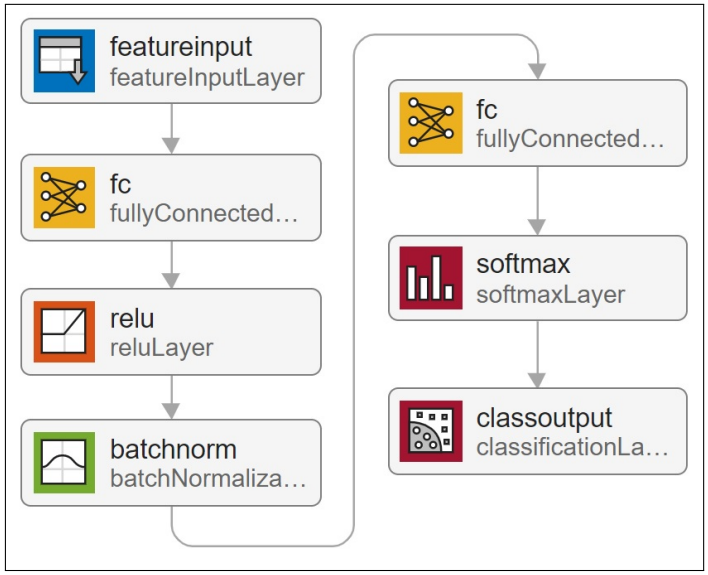


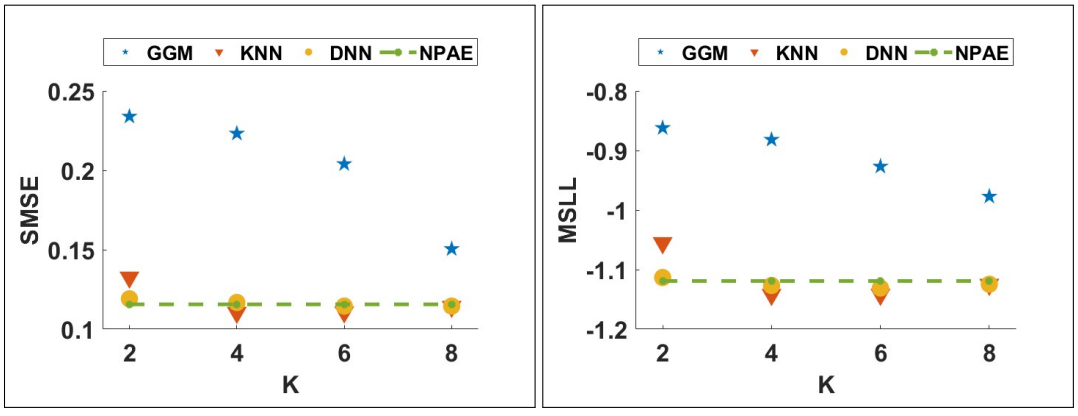
Figure 6. Deep neural network architecture for adopted multi-label classification in DGPs.

Algorithm 3 Aggregating Dependent Experts Using DNN

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

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

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



(**a**) (**b**)

Figure 7. Expert-selection-prediction qualities of different expert-selection methods compared to original baseline, NPAE, from *Concrete* dataset. (**a**) SMSE. (**b**) MSLL.

4. Discussion

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

4.1. 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 [47–50] 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.

In addition, 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 [51,52]. 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.

4.2. Computational Costs of Expert-Selection Models

The aggregation cost in all three selection methods, i.e., GGM, KNN, and DNN, is $\mathcal{O}(N_t K^3)$ 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 $\mathcal{O}(M^3)$, where 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 λ leads to a dense graph with a more considerable computational cost.

The cost of the KNN Algorithm 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 $\mathcal{O}(MD)$ work for an iteration and therefore $\mathcal{O}(KMD)$ 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 $\mathcal{O}(N(U_1U_2 + \ldots + U_{L-1}U_L))$.

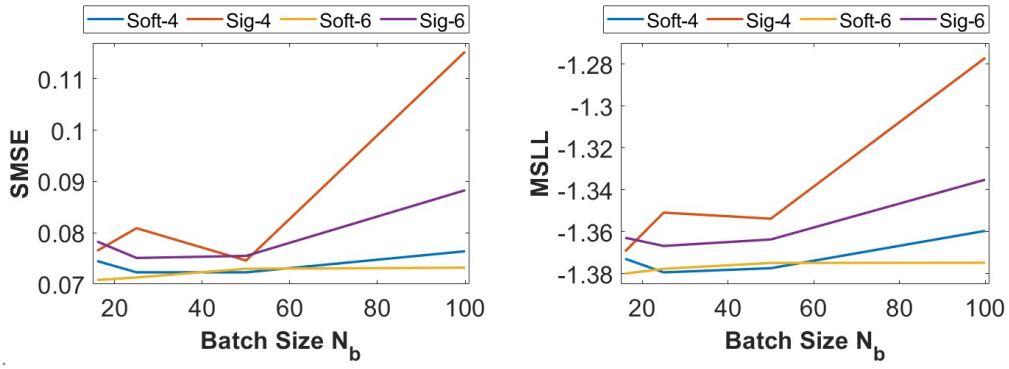
In conclusion, both methods described in Algorithms 2 and 3 have linear complexity concerning M. Therefore, they are more efficient when the number of partitions is an enormous value, unlike the cubical dependency in Algorithm 1.

4.3. Activation Functions in DNN

Using a *softmax* output layer for the DNN-based classification in Section 3.4 leads to a probability vector $P^{\mathcal{L}}(x^*)$ 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* dataset 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.



(**a**) (**b**)

Figure 8. Activation functions for the final (i.e., output) layer of the DNN classifier: prediction quality for $K = 4$ and $K = 6$ in an experiment from the *Concrete* dataset with 10 experts. (**a**) SMSE. (**b**) MSLL.

4.4. Expert Selection for CI-Based Baselines

In [25], 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 β. 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 Algorithm 2 or Algorithm 3. Since these models are fast, the selection parameter K can also be set to relatively large values.

Table 1 describes the effect of the selection scenario on CI-based ensembles using KNN on the *Concrete* dataset with $M = 10$ and $K = 6$. Although this modification cannot 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.

Table 1. Expert selection in CI-based baselines: The KNN method is employed for the selection task, and the results are highlighted in red.

Model	Expert Selection	SMSE	MSLL	Time (s)
GPoE	-	0.138	−0.876	0.03
	KNN	0.115	−0.916	0.03
RBCM	-	0.0993	0.396	0.03
	KNN	0.091	0.156	0.03
GRBCM	-	0.1093	−1.103	0.06
	KNN	0.089	−1.21	0.06

5. Experiments

The quality of the expert-selection methods is assessed in this section. We consider the prediction quality and the required prediction time of the proposed and state-of-the-art distributed GP models using both simulated and real datasets. The quality of predictions is evaluated by the standardized mean squared error (SMSE) and the mean

standardized log loss (MSLL). The standard squared exponential kernel with automatic relevance determination and a Gaussian likelihood is used. Since the disjoint partitioning of training data captures the local features more accurately and outperforms random partitioning [25,31], it is mainly used in our experiments. The sparsity parameter in the GGM-based expert-selection method is set to $\lambda = 0.1$, *Euclidean* norm measures the distances in KNN, and a neural network with a single hidden layer is used for the DNN classification. The experiments were conducted in MATLAB R2022b using GPML package: https://www.gaussianprocess.org/gpml/code/matlab/doc/ (accessed on 10 March 2022).

5.1. Sensitivity Analysis

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 dataset. Then, we use a medium-scale real-world dataset to study how hyperparameters affect the results in a complex multi-dimensional dataset.

5.1.1. 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 [31],

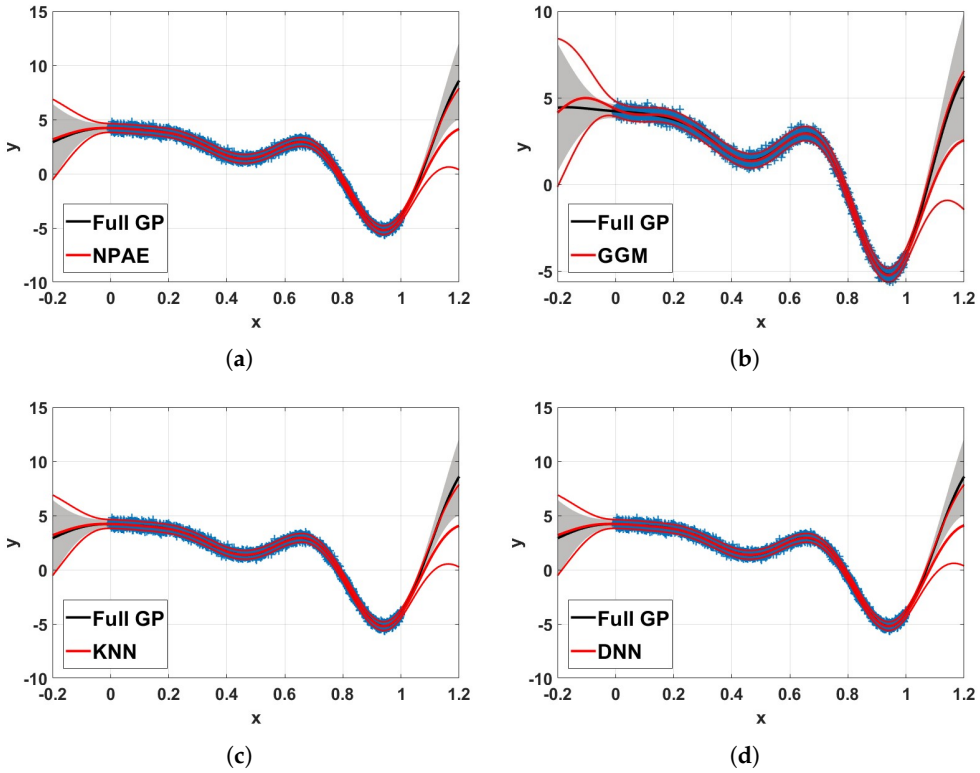
$$f(x) = 5x^2 sin(12x) + (x^3 - 0.5)sin(3x - 0.5) + 4cos(2x) + \epsilon, \tag{9}$$

where $\epsilon \sim \mathcal{N}\left(0, (0.2)^2\right)$. We generate n training points in $[0, 1]$, and $N_t = 0.1n$ test points in $[-0.2, 1.2]$. The data are 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 in 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 3×10^3 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 Figure 10a,b indicate that multi-label classification leads to a selection strategy with fast convergence to the original estimator.

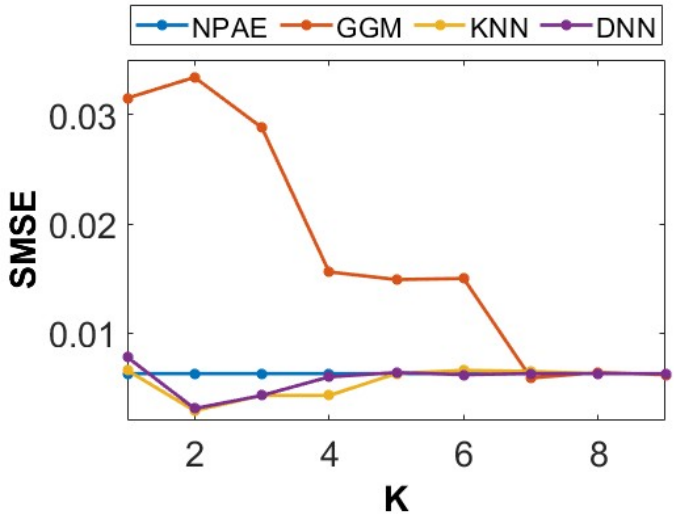
When the number of experts increases, the prediction errors of KNN and DNN do not show significant changes, indicating predictive stability. On the other hand, GGM needs more experts to provide closer results to NPAE and has a slow convergence procedure. For $K \geq 7$, the error values of all three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



(**a**) (**b**)

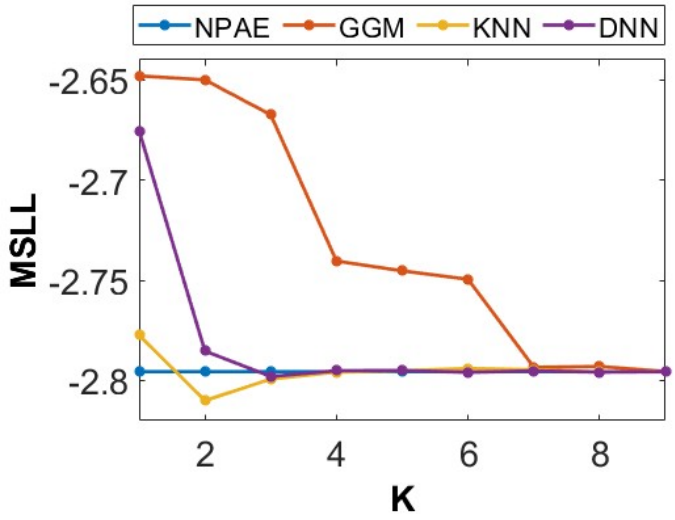
(**c**) (**d**)

Figure 9. The 99% confidence interval of NPAE, expert-selection methods, and full GP for $n = 3 \times 10^3$ training points from Equation (9) and $M = 10$ (partition size $m_0 = 300$) with K-means partitioning. The GGM, KNN, and DNN results are based on $K = 5$ selected experts. We see that the DNN and the KNN approximations to the full GP are practically indistinguishable from the NPAE approximation; all three techniques are quality-wise superior to the GGM approach. (**a**) NPAE. (**b**) GGM. (**c**) KNN. (**d**) DNN.

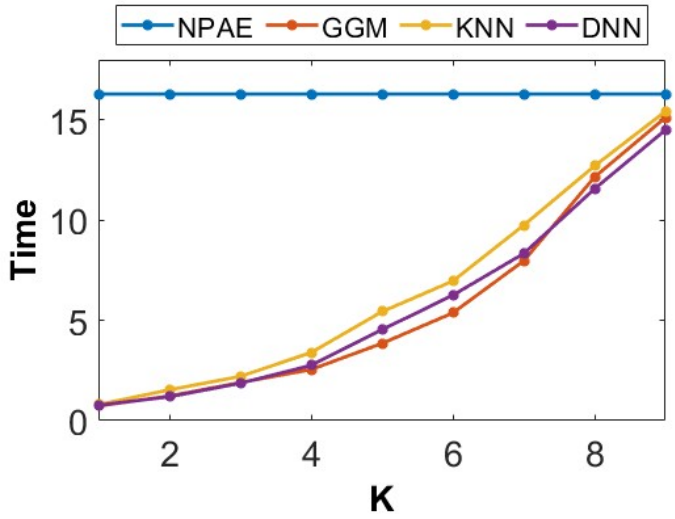


(**a**)

Figure 10. *Cont.*



(**b**)



(**c**)

Figure 10. Prediction quality of available baselines as a function of experts for 3×10^3 training points and partition size $m_0 = 300$ with K-means partitioning. (**a**) SMSE and $n = 3000$. (**b**) MSLL and $n = 3000$. (**c**) Time and $n = 3000$.

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 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.

Figure 11e,f show the baselines' running time considering the dependencies between experts. All expert-selection-based aggregations have the same prediction process (of $\mathcal{O}(N_t K^3)$), and their difference is 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 $\mathcal{O}(N_t K^3)$, see Section 4.2. In these experiments with smooth 1D data points, the running times of the GGM, the KNN, and the DNN are of the same rate.

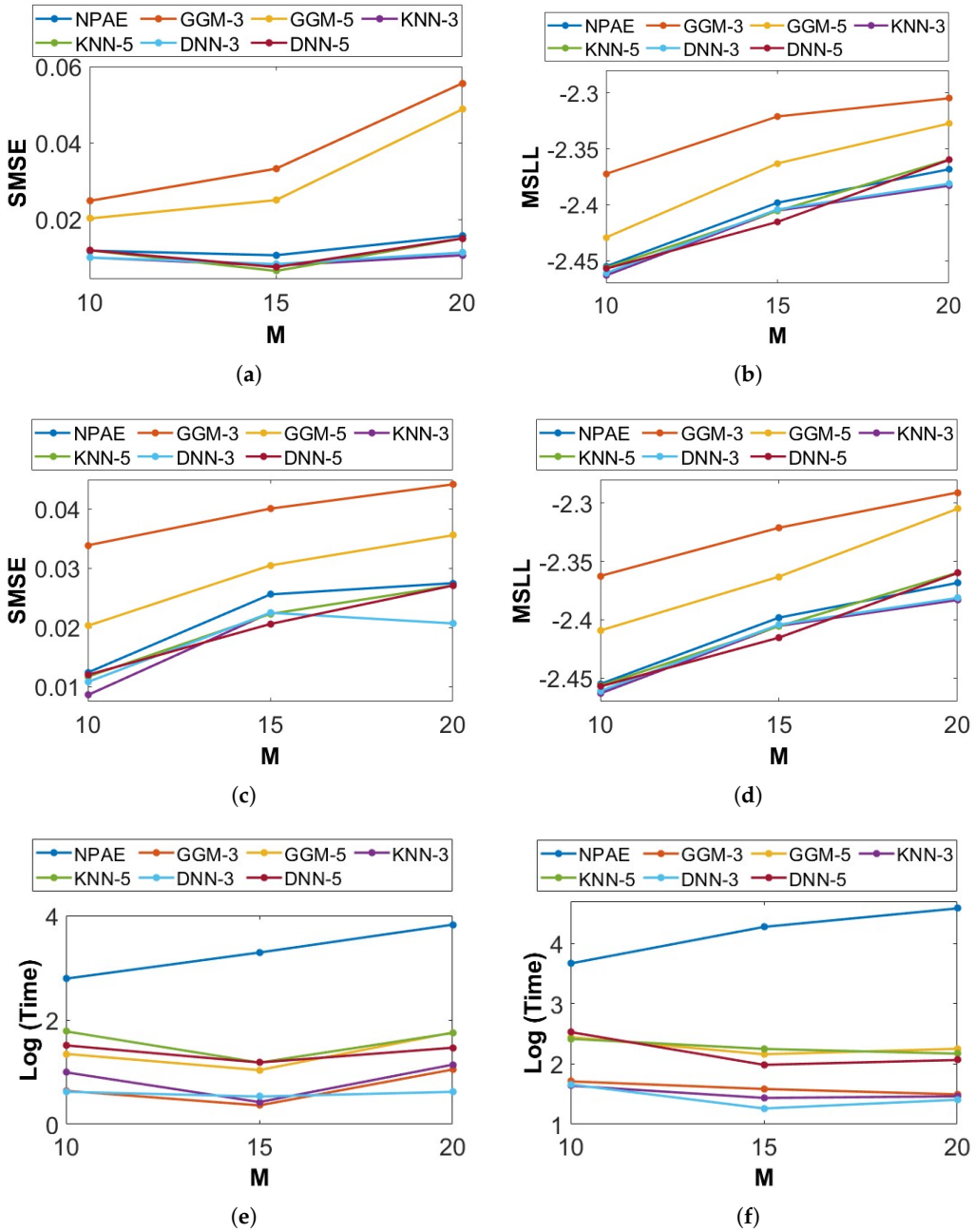


Figure 11. Prediction quality and running time of DGP baselines for 3000 and 5000 training points from Equation (9) with different numbers of partitions, $M = \{10, 15, 20\}$. For expert-selection-based methods, $K = \{3, 5\}$ experts were selected for the final aggregation. (**a**) SMSE and $n = 3000$. (**b**) MSLL and $n = 3000$. (**c**) SMSE and $n = 5000$. (**d**) MSLL and $n = 5000$. (**e**) Time and $n = 3000$. (**f**) Time and $n = 5000$.

5.1.2. Multi-Dimensional Real-World Dataset

The relative number of experts $K_M = \frac{K}{M}$ defined in Section 5.1.1 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 dataset and K_M to appraise the efficacy of the data-assignment strategy on the prediction quality. *Pumadyn* (https://www.cs.toronto.edu/~delve/data/pumadyn/desc.html, accessed on 10 March 2022) is a generated dataset with 32 dimensions and 7168 training points and 1024 test points. The disjoint partitioning divides the dataset into 10, 15, and 20 subsets. We considered the GPoE [8], GRBCM [31], NPAE [22], GGM-based aggregation [25], and proposed classification-based methods with K-means clustering. The penalty term λ 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 2 depicts the prediction quality of local approximation methods for the *Pumadyn* dataset. The column Type shows the interactions between experts in the aggregation method, D for dependent experts, and CI for conditionally independent experts. *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.

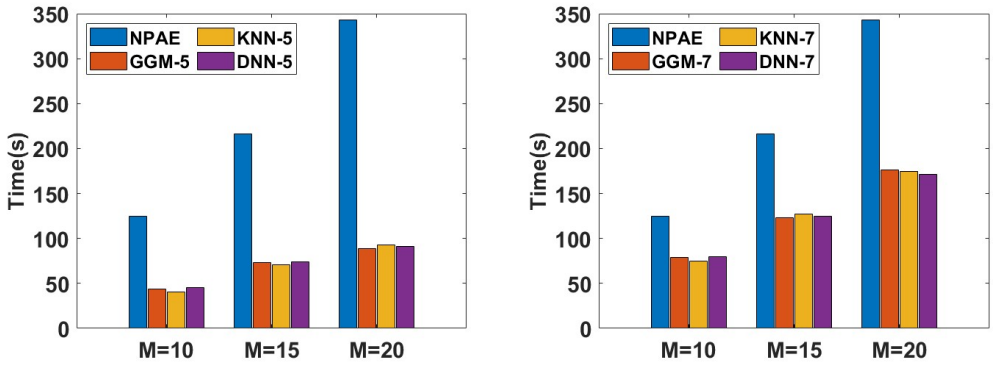
Table 2. SMSE and MSLL of different baselines on the *Pumadyn* dataset for different numbers of partition strategies. Both dependent (D) and conditionally independent (CI) aggregation methods are used. The best results are highlighted in **bold**. Methods based on expert selection are highlighted in blue in the table.

		$M = 10$		$M = 15$		$M = 20$	
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	0.0490	−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
GGM-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**

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. This 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 the *Pumadyn* dataset. The training dataset 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 times of GGM, KNN, and DNN are almost at the same rate. However, Table 2 shows that GGM cannot 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.



(**a**) (**b**)

Figure 12. Prediction time (seconds) of different aggregation for disjoint partitioning in the *Pumadyn* dataset. The training dataset is divided into 10, 15, and 20 subsets and 50% and 70% of experts are selected for final aggregation. (**a**) Aggregation with 50% of experts. (**b**) Aggregation with 70% of experts.

5.2. Prediction Quality in Real-World Datasets

In this section, we use real-world datasets to evaluate the prediction quality of proposed aggregations and compare them with available baselines. The baselines that we use here are GPoE with uniform weights [8], RBCM [8], GRBCM [31], NPAE [22], GGM-based aggregation [25], KNN and DNN-based ensemble methods. Various real-world datasets with different sizes and dimensions are used here, as explained in Table 3.

Table 3. Real-world datasets.

Dataset	(#) Observations	n	N_t	D
Airfoil [1]	1503	1203	300	5
Parkinson [2]	5875	5000	875	20
Pole Telecom [53]	15,000	10,000	5000	26
Protein [3]	45,730	40,000	5730	9
Sacros [4]	48,938	44,489	4449	21
Song [5]	515,345	463,715	51,630	91

[1] https://archive.ics.uci.edu/dataset/291/airfoil+self+noise (accessed on 10 March 2022); [2] https://archive.ics.uci.edu/ml/datasets/parkinsons+telemonitoring (accessed on 10 March 2022); [3] https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure (accessed on 10 March 2022); [4] http://www.gaussianprocess.org/gpml/data/ (accessed on 10 March 2022); [5] https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd (accessed on 10 March 2022).

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 datasets, there are predefined training and test sets. Indeed, in the *Song* dataset, we extract the first 10^5 songs from this dataset for training and the first 10^4 songs from the original test set for

testing. We used disjoint partitioning (K-means) to divide the datasets 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* dataset only, we set K_M = 0.2. Since NPAE is computationally burdensome, especially when M and N_t are large, it is only used in small- and medium-scale datasets. DNN uses a neural network with a hidden layer and 50 hidden units to evaluate the labels.

Tables 4 and 5 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 datasets where using the NPAE is not feasible. In addition, 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 datasets.

Table 4. SMSE for various methods on real-world datasets. 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. The best results are highlighted in **bold**. Methods based on expert selection are highlighted in blue in the table.

	SMSE					
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 5. MSLL for various methods on real-world datasets. 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. The best results are highlighted in **bold**. Methods based on expert selection are highlighted in blue in the table.

	MSLL					
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**

We consider the *Parkinson* dataset as an instance. GGM cannot provide accurate predictions when K_M = 0.5. Our experiments confirm that increasing 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. This 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, and (the MSLL values of the KNN and DNN methods for $K_M = 0.7$ are −0.8575 and −0.8587, 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).

On the other hand, a comparison of the running times of the aggregation methods confirms the results presented in Section 5.1. Consider the *Pole Telecom* dataset. The logarithm of the prediction times (in seconds) was calculated for $K_M = 0.5$. The NPAE method recorded a logarithm of 3.31, while GGM, KNN, and DNN yielded logarithms of 2.66, 2.42, and 2.41, respectively. These values indicate that NPAE has the longest running time in logarithmic terms, followed by GGM, KNN, and DNN in decreasing order. However, as previously discussed, GGM requires more experts to achieve higher prediction quality and therefore requires additional time to match the performance of KNN and DNN.

In CI-based methods, the conservative GPoE does not return acceptable results and cannot 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 than 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.

6. Conclusions

In this work, we propose a novel expert-selection approach for distributed learning with Gaussian agents, leveraging expert selection to aggregate the predictions of dependent local experts. Existing ensemble baselines incorporate all correlated experts during the aggregation step, which can negatively impact final predictions due to the influence of weak local experts or result in impractically high computational costs.

Our proposed approach employs a multi-label classification model, where data points are allocated to experts treated as class labels. Unlike existing expert-selection methods, which assign a fixed and static set of experts to all new data points, the proposed model adapts to changes in the model and data by selecting a relevant group of experts for each input. Excluding unrelated experts for each test point enhances prediction quality and reduces computational costs. Additionally, the approach retains the original baseline's asymptotic properties, ensuring consistent results as $n \to \infty$.

The classification methods used in this work, such as KNN and DNN, can be replaced with newer and more efficient solutions for addressing the multi-label classification problem. The proposed approach is applicable to both distributed and federated learning and imposes minimal assumptions. Empirical analyses demonstrate the superiority of our approach, which enhances the prediction quality of existing SOTA aggregation methods while maintaining high efficiency.

Author Contributions: Conceptualization, H.J. and G.K.; methodology, H.J. and G.K.; software, H.J.; validation, H.J.; formal analysis, H.J.; investigation, H.J.; resources, G.K.; writing—original draft preparation, H. J.; writing—review and editing, G.K.; visualization, H.J.; supervision, G.K.; project administration, G.K.; funding acquisition, G.K. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Rasmussen, C.E.; Williams, C.K. *Gaussian Processes for Machine Learning*; MIT Press: Cambridge, MA, USA, 2006.
2. Deringer, V.L.; Bartók, A.P.; Bernstein, N.; Wilkins, D.M.; Ceriotti, M.; Csányi, G. Gaussian Process Regression for Materials and Molecules. *Chem. Rev.* **2021**, *121*, 10073–10141. [CrossRef] [PubMed]
3. Zeng, A.; Ho, H.; Yu, Y. Prediction of building electricity usage using Gaussian Process Regression. *J. Build. Eng.* **2020**, *28*, 101054. [CrossRef]
4. Denzel, A.; Kästner, J. Gaussian process regression for geometry optimization. *J. Chem. Phys.* **2018**, *148*, 094114. [CrossRef]
5. Gramacy, R. *Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2020.
6. Le, T.; Nguyen, K.; Nguyen, V.; Nguyen, T.D.; Phung, D. GoGP: Fast online regression with Gaussian processes. In Proceedings of the IEEE International Conference on Data Mining, New Orleans, LA, USA, 18–21 November 2017 ; pp. 257–266.
7. Tobar, F.; Bui, T.D.; Turner, R.E. Learning stationary time series using Gaussian processes with nonparametric kernels. In Proceedings of the 29th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 3501–3509.
8. Deisenroth, M.P.; Ng, J.W. Distributed Gaussian processes. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; pp. 1481–1490.
9. Seeger, M.; Williams, C.; Lawrence, N. Fast Forward Selection to Speed Up Sparse Gaussian Process Regression. In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, USA, 3–6 January 2003; Volume R4, pp. 254–261.
10. Titsias, M.K. Variational learning of inducing variables in sparse Gaussian processes. In Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, Clearwater Beach, FL, USA, 16–18 April 2009; pp. 567–574.
11. Hensman, J.; Fusi, N.; Lawrence, N.D. Gaussian processes for big data. In *Uncertainty in Artificial Intelligence*; AUAI Press: Arlington, VA, USA, 2013; pp. 282–290.
12. Cheng, C.; Boots, B. Variational Inference for Gaussian Process Models with Linear Complexity. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017) , Long Beach, CA, USA, 4–9 December 2017; pp. 5184–5194.
13. Burt, D.R.; Rasmussen, C.E.; van der Wilk, M. Rates of Convergence for Sparse Variational Gaussian Process Regression. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019.
14. Bui, T.D.; Turner, R.E. Tree-structured Gaussian process approximations. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 2213–2221.
15. Moore, D.; Russell, S.J. Gaussian process random fields. In Proceedings of the 29th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 3357–3365.
16. Peteiro-Barral, D.; Guijarro-Berdiñas, B. A survey of methods for distributed machine learning. *Prog. Artif. Intell.* **2013**, *2*, 1–11. [CrossRef]
17. Zhang, J.; Wang, M.; Li, Q.; Wang, S.; Chang, X.; Wang, B. A Survey on Distributed Machine Learning. *ACM Comput. Surv.* **2020**, *53*, 30 .
18. Liu, J.; Huang, J.; Zhou, Y.; Li, X.; Ji, S.; Xiong, H.; Dou, D. From distributed machine learning to federated learning: A survey. *arXiv* **2021**, arXiv:2104.14362. [CrossRef]
19. Yang, Z.; Dai, X.; Dubey, A.; Hirche, S.; Hattab, G. Whom to Trust? Elective Learning for Distributed Gaussian Process Regression. In Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS, Auckland, New Zealand, 6–10 May 2024; pp. 2020–2028.
20. Liu, H.; Ong, Y.S.; Shen, X.; Cai, J. When Gaussian Process Meets Big Data: A Review of Scalable GPs. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *31*, 4405–4423. [CrossRef]
21. Szabó, B.; van Zanten, H. An asymptotic analysis of distributed nonparametric methods. *J. Mach. Learn. Res.* **2019**, *20*, 1–30.
22. Rullière, D.; Durrande, N.; Bachoc, F.; Chevalier, C. Nested Kriging predictions for datasets with a large number of observations. *Stat. Comput.* **2018**, *28*, 849–867. [CrossRef]
23. Bachoc, F.; Durrande, N.; Rullière, D.; Chevalier, C. Some Properties of Nested Kriging Predictors; *arXiv* **2017**, arXiv:1707.05708.

24. Jalali, H.; Kasneci, G. Aggregating the Gaussian Experts' Predictions via Undirected Graphical Models. In Proceedings of the IEEE International Conference on Big Data and Smart Computing (BigComp), Daegu, Republic of Korea, 17–20 January 2022; pp. 23–26.
25. Jalali, H.; Pawelczyk, M.; Kasneci, G. Model Selection in Local Approximation Gaussian Processes: A Markov Random Fields Approach. In Proceedings of the IEEE International Conference on Big Data (Big Data), Orlando, FL, USA, 15–18 December 2021; pp. 768–778.
26. Jalali, H.; Pawelczyk, M.; Kasneci, G. Gaussian Experts Selection using Graphical Models. *arXiv* **2021**, arXiv:2102.01496.
27. Herrera, F.; Charte, F.; Rivera, A.J.; del Jesus, M.J. Multilabel Classification. In *Multilabel Classification: Problem Analysis, Metrics and Techniques*; Springer International Publishing: Cham, Switzerland, 2016; pp. 17–31.
28. Park, C.; Apley, D. Patchwork kriging for large-scale Gaussian process regression. *J. Mach. Learn. Res.* **2018**, *19*, 1–43.
29. Blanchard, P.; Mhamdi, E.E.; Guerraoui, R.; Stainer, J. Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent. In Proceedings of the 31st International Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30.
30. Data, D.; Diggavi, S. Byzantine-Resilient High-Dimensional SGD with Local Iterations on Heterogeneous Data. In Proceedings of the 38th International Conference on Machine Learning, Virtual, 18–24 July 2021; Volume 139, pp. 2478–2488.
31. Liu, H.; Cai, J.; Ong, Y.; Wang, Y. Generalized robust Bayesian committee machine for large-scale Gaussian process regression. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; pp. 1–10.
32. Hinton, G.E. Training products of experts by minimizing contrastive divergence. *Neural Comput.* **2002**, *14*, 1771–1800. [CrossRef] [PubMed]
33. Cao, Y.; Fleet, D.J. Generalized product of experts for automatic and principled fusion of Gaussian process predictions. *arXiv* **2014**, arXiv:1410.7827.
34. Tresp, V. A Bayesian committee machine. *Neural Comput.* **2000**, *12*, 2719–2741. [CrossRef]
35. Mendes-Moreira, J.; Soares, C.; Jorge, A.; Sousa, J. Ensemble approaches for regression: A survey. *ACM Comput. Surv. (CSUR)* **2012**, *45*, 10. [CrossRef]
36. Parisi, F.; Strino, F.; Nadler, B.; Kluger, Y. Ranking and combining multiple predictors without labeled data. *Proc. Natl. Acad. Sci. USA* **2014**, *111*, 1253–1258. [CrossRef] [PubMed]
37. Jaffe, A.; Fetaya, E.; Nadler, B.; Jiang, T.; Kluger, Y. Unsupervised ensemble learning with dependent classifiers. In Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, Cadiz, Spain, 9–11 May 2016; pp. 351–360.
38. Donmez, P.; Lebanon, G.; Balasubramanian, K. Unsupervised supervised learning I: Estimating classification and regression errors with-out labels. *J. Mach. Learn. Res.* **2010**, *11*, 1323–1351.
39. Platanios, E.; Blum, A.; Mitchell, T. Estimating accuracy from unlabeled data. In Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, Quebec City, QC, Canada, 23– 27 July 2014; pp. 682–691.
40. Friedman, J.; Hastie, T.; Tibshirani, R. Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **2008**, *9*, 432–441. [CrossRef]
41. Witten, D.M.; Friedman, J.H.; Simon, N. New insights and faster computations for the graphical lasso. *J. Comput. Graph. Stat.* **2011**, *20*, 892–900. [CrossRef]
42. Jalali, H.; Kasneci, G. Aggregating Dependent Gaussian Experts in Local Approximation. In Proceedings of the 25th International Conference on Pattern Recognition (ICPR), Milan, Italy, 10–15 January 2021; pp. 9015–9022.
43. Altman, N.S. An Introduction to Kernel and Nearest-Neighbor Non-parametric Regression. *Am. Stat.* **1992**, *46*, 175–185. [CrossRef]
44. Everitt, B.; Landau, S.; Leese, M.; Stahl, D. Miscellaneous clustering methods. In *Cluster Analysis*; John Wiley & Sons : New York, NY, USA, 2011; pp. 215–255.
45. Read, J.; Perez-Cruz, F. Deep learning for multi-label classification. *arXiv* **2014**, arXiv:1502.05988.
46. Du, J.; Chen, Q.; Peng, Y.; Xiang, Y.; Tao, C.; Lu, Z. ML-Net: Multi-label classification of biomedical texts with deep neural networks. *J. Am. Med. Inform. Assoc.* **2019**, *26*, 1279–1285. [CrossRef]
47. Lafferty, J.; Liu, H.; Wasserman, L. Sparse Nonparametric Graphical Models. *Stat. Sci.* **2012**, *27*, 519–537. [CrossRef]
48. Mulgrave, J.J.; Ghosal, S. Bayesian Inference in Nonparanormal Graphical Models. *Bayesian Anal.* **2020**, *15*, 449–475. [CrossRef]
49. Li, B.; Solea, E. A Nonparametric Graphical Model for Functional Data with Application to Brain Networks Based on fMRI. *J. Am. Stat. Assoc.* **2018**, *113*, 1637–1655. [CrossRef]
50. Solea, E.; Dette, H. Nonparametric and high-dimensional functional graphical models. *arXiv* **2021**, arXiv:2103.10568s. [CrossRef]
51. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.; Kaiser, L.; Polosukhin, I. Attention is all you need. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2017; Volume 30.

52. Kossen, J.; Band, N.; Gomez, C.L.A.; Rainforth, T.; Gal, Y. Self-attention between datapoints: Going beyond individual input-output pairs in deep learning. In *Advances in Neural Information Processing Systems*; Curran Associates Inc.: Red Hook, NY, USA, 2021; Volume 34, pp. 28742–28756.
53. Nguyen, T.; Bonilla, E. Fast Allocation of Gaussian Process Experts. In Proceedings of the 31st International Conference on Machine Learning, Beijing, China, 21–26 June 2014; Volume 32, pp. 145–153.