Aggregation Strategies for Distributed Gaussian Processes

Aggregation Strategies for Distributed Gaussian Processes

Dissertation

der Mathematisch-Naturwissenschaftlichen Fakultät

der Eberhard Karls Universität Tübingen

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

(Dr. rer. nat.)

vorgelegt von

Hamed Jalali

aus Teheran, Iran

Tübingen

2022

Gedruckt mit Genehmigung der Mathematisch-Naturwissenschaftlichen Fakultät der Eberhard Karls Universität Tübingen.

Tag der mündlichen Qualifikation:	14.04.2023
Dekan:	Prof. Dr. Thilo Stehle
1. Berichterstatter:	Prof. Dr. Gjergji Kasneci
2. Berichterstatter:	Prof. Dr. Gerard Pons-Moll

”The only true wisdom is in knowing you know nothing.”

- Socrates

Abstract

Gaussian processes are robust and flexible non-parametric statistical models that benefit from the Bayes theorem by assigning a Gaussian prior distribution to the unknown function. Despite their capability to provide high-accuracy predictions, they suffer from high computational costs. Various solutions have been proposed in the literature to deal with computational complexity. The main idea is to reduce the training cost, which is cubic in the size of the training set.

A distributed Gaussian process is a divide-and-conquer approach that divides the entire training data set into several partitions and employs a local approximation scenario to train a Gaussian process at each data partition. An ensemble technique combines the local Gaussian experts to provide final aggregated predictions. Available baselines aggregate local predictions assuming perfect diversity between experts. However, this assumption is often violated in practice and leads to sub-optimal solutions.

This thesis deals with dependency issues between experts. Aggregation based on experts' interactions improves accuracy and can lead to statistically consistent results. Few works have considered modeling dependencies between experts. Despite their theoretical advantages, their prediction steps are costly and cubically depend on the number of experts. We benefit from the experts' interactions in both dependence and independence-based aggregations. In conventional aggregation methods that combine experts using a conditional independence assumption, we transform the available experts set into clusters of highly correlated experts using spectral clustering. The final aggregation uses these clusters instead of the original experts. It reduces the effect of the independence assumption in the ensemble technique. Moreover, we develop a novel aggregation method for dependent experts using the latent variable graphical model and define the target function as a latent variable in a connected undirected graph.

Besides, we propose two novel expert selection strategies in distributed learning.

They improve the efficiency and accuracy of the prediction step by excluding weak experts in the ensemble method. The first is a static selection method that assigns a fixed set of experts to all new entry points in the prediction step using the Markov random field model. The second solution increases the flexibility of the selection step by converting it into a multi-label classification problem. It provides an entry-dependent selection model and assigns the most relevant experts to each data point.

We address all related theoretical and practical aspects of the proposed solutions. The findings present valuable insights for distributed learning models and advance the state-of-the-art in several directions. Indeed, the proposed solutions do not need restricted assumptions and can be easily extended to non-Gaussian experts in distributed and federated learning.

Kurzfassung

Gaußsche Prozesse sind robuste und flexible nichtparametrische statistische Modelle, die Bayes-Theorem verwenden, um einer unbekannten Funktion eine Gaußsche Prior-Verteilung zuzuweisen. Trotz ihrer Fähigkeit, hochgenaue Vorhersagen zu liefern, leiden sie unter hohen Rechenkosten. In der Literatur wurden verschiedene Lösungen vorgeschlagen, um die Rechenkomplexität zu beherrschen. Die Hauptidee besteht darin, die Trainingskosten zu reduzieren, die in der Größe des Trainingssets kubisch sind.

Der verteilte Gaußsche Prozess ist ein Teile-und-Herrsche-Ansatz, der den gesamten Trainingsdatensatz in mehrere Partitionen unterteilt und ein lokales Näherungsszenario verwendet, um einen Gaußschen Prozess an jeder Datenpartition zu trainieren. Eine Ensemble-Technik kombiniert die lokalen Gaußschen Experten, um endgültige aggregierte Vorhersagen zu liefern. Verfügbare Basislösungen aggregieren lokale Vorhersagen unter der Annahme einer perfekten Diversität zwischen Experten. Diese Annahme wird jedoch in der Praxis oft verletzt und führt zu suboptimalen Lösungen.

Diese Arbeit beschäftigt sich mit Abhängigkeitsproblemen zwischen Experten. Die Aggregation basierend auf den Interaktionen von Experten verbessert die Genauigkeit und kann zu statistisch konsistenten Ergebnissen führen. Nur wenige Arbeiten haben die Modellierung von Abhängigkeiten zwischen Experten in Betracht gezogen. Trotz ihrer theoretischen Vorteile sind ihre Vorhersageschritte kostspielig und hängen kubisch von der Anzahl der Experten ab. Wir profitieren von den Interaktionen der Experten sowohl bei abhängigkeits- als auch bei unabhängigkeitsbasierten Aggregationen. In konventionellen Aggregationsverfahren, die Experten unter Verwendung einer bedingten Unabhängigkeitsannahme kombinieren, transformieren wir den verfügbaren Expertensatz in Cluster von hochgradig korrelierten Experten unter Verwendung von spektralem Clustering. Die endgültige Aggregation verwendet diese Cluster anstelle der ursprünglichen Experten. Diese Vorgehensweise reduziert den Effekt der Unabhängigkeits- annahme in

der Ensemble-Technik. Darüber hinaus entwickeln wir eine neuartige Aggregationsmethode für abhängige Experten unter Verwendung eines latenten Variablen-Grafikmodells und definieren die Zielfunktion als latente Variable in einem verbundenen ungerichteten Graphen.

Außerdem schlagen wir zwei neue Expertenauswahlstrategien für verteiltes Lernen vor. Sie verbessern die Effizienz und Genauigkeit des Vorhersageschritts, indem sie schwache Experten in der Ensemble-Methode ausschließen. Das erste ist ein statisches Auswahlverfahren, das allen neuen Eintrittspunkten im Vorhersageschritt unter Verwendung des Markov-Zufallsfeldmodells eine feste Gruppe von Experten zuweist. Die zweite Lösung erhöht die Flexibilität des Auswahlschritts, indem sie ihn in ein Klassifizierungsproblem mit mehreren Labels umwandelt. Es bietet ein eintragsabhängiges Auswahlmodell und ordnet jedem Datenpunkt die relevantesten Experten zu.

Wir gehen auf alle damit verbundenen theoretischen und praktischen Aspekte der vorgeschlagenen Lösungen ein. Die Ergebnisse stellen wertvolle Erkenntnisse für verteilte Lernmodelle dar und bringen den Stand der Technik in mehrere Richtungen voran. Tatsächlich benötigen sie keine eingeschränkten Annahmen und können leicht auf nicht-Gaußsche Experten für verteiltes und föderiertes Lernen erweitert werden.

Acknowledgments

First, I thank my supervisor Gjergji Kasneci for his guidance throughout the research work. Conducting the academic study regarding such a difficult topic couldn't be as simple as he made this for me. Despite the differences in research interests, his immense knowledge, motivation, and patience have given me more power and spirit to finish my thesis.

I am also grateful to my colleagues in the Data Science and Analytics group. In particular, I thank Vadim Borisov, Martin Pawelczyk, Johannes Christian Haug, and Tobias Leemann, with whom I had many insightful discussions about our research projects. Special thanks go to Shahram Eivazi, who kindly helped me with many beneficial recommendations. Furthermore, I thank Margot Reimold for helping me with all the bureaucratic issues during my stay in Tuebingen.

Moreover, I thank my parents and brother for their unconditional love, support, encouragement, and confidence.

Lastly and most importantly, I am particularly grateful to my wife, **Mahsa**, for her unfailing understanding and encouragement throughout my years of study, researching, and writing this thesis. Without her patience and support, this path couldn't end.

Tuebingen, December, 2022 Hamed Jalali

Contents

List of Tables

List of Figures

List of Algorithms

List of Publications

Accepted Publications

Paper 1

Hamed Jalali and Gjergji Kasneci.2020. **"Aggregating Dependent Gaussian Experts in Local Approximation"**. In 25th International Conference on Pattern Recognition (ICPR 2020). https://doi.org/10.1109/ICPR48806.2021.9413079[3].

Paper 2

Hamed Jalali, Martin Pawelczyk and Gjergji Kasneci.2021. **"Model Selection in Local Approximation Gaussian Processes: A Markov Random Fields Approach"**. In 2021 IEEE International Conference on Big Data (Big Data). https://doi.org/10.1109/BigData52589.2021.9672077[4].

Paper 3

Hamed Jalali and Gjergji Kasneci.2021. **"Gaussian Graphical Models as an Ensemble Method for Distributed Gaussian Processes"**. OPT2021: 13th Annual Workshop on Optimization for Machine Learning at 35th Conference on Neural Information Processing Systems (NeurIPS 2021). https://doi.org/10.48550/arXiv.2202.03287. (workshop proceedings at https://opt-ml.org/oldopt/opt21/papers.html)[5].

Paper 4

Hamed Jalali and Gjergji Kasneci.2022. **"Aggregating the Gaussian Experts' Predictions via Undirected Graphical Models"**. In IEEE International Conference on Big Data and Smart Computing (BigComp). https://doi.org/10.1109/BigComp54360.2022.00014[6].

Paper 5

Hamed Jalali and Gjergji Kasneci.2022. **"Expert Selection in Distributed Gaussian Processes: A Multi-label Classification Approach"**. GPSMDMS: NeurIPS Workshop on Gaussian Processes, Spatiotemporal Modeling, and Decision-making Systems at 36th Conference on Neural Information Processing Systems (NeurIPS 2022). [7].

arXiv Published Manuscripts

Paper 6

Hamed Jalali, Martin Pawelczyk and Gjergji Kasneci.2021. **"Gaussian experts selection using graphical models"**. Published on ArXiv. https://arxiv.org/abs/2102.01496[8].

Submitted Manuscripts

Paper 7

Hamed Jalali and Gjergji Kasneci.2022. **"Entry Dependent Expert Selection in Distributed Gaussian Processes Using Multilabel Classification"**. Under review at the IEEE Transactions on Neural Network and Learning Systems.

Additional Co-Authored Paper

Paper 8

Vadim Borisov, Johannes Meier, Johan van den Heuvel, Hamed Jalali, Gjergji Kasneci.2021. **"A Robust Unsupervised Ensemble of Feature-Based Explanations using Restricted Boltzmann Machines"**. Workshop on eXplainable AI approaches for debugging and diagnosis at 35th Conference on Neural Information Processing Systems (NeurIPS 2021). `https://doi.org/10.48550/arXiv.2111.07379`[9].

Chapter 1

Introduction

Gaussian processes (GPs) [10] are a flexible, interpretable, and powerful non-parametric statistical approach to learning and predicting under uncertainty. They provide accurate predictions with quantified uncertainty and apply Bayes' theorem for inference, allowing them to estimate complex linear and non-linear structures without the need for restrictive assumptions about the model. They have been extensively used in practical cases, e.g., optimization [11], data visualization, and manifold learning [12], building engineering [13], chemical engineering [14], reinforcement learning [15, 16], multitask learning [17, 18], online streaming models [19, 20] and time series analysis [21, 22]. The main bottleneck of using standard GPs is that they poorly scale with the size of the data set. Various machine learning solutions have been proposed to render GPs feasible for more complex and large data sets. In this thesis, we address local approximation GPs, also called distributed GPs, and discuss some of the most pressing issues in this area.

1.1 Problem Set-up

The basic nonlinear regression problem is of the form

$$y = f(x) + \varepsilon,$$

where $x \in R^D$ is a D-dimensional independent variable and the observation noise ε is assumed to be a Gaussian random variable with mean 0 and variance σ^2, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. Thus, the Gaussian likelihood is defined as $p(y|f) = \mathcal{N}(f, \sigma^2 I)$. The objective is to learn

the latent function $f : R^D \to R$ from a training set $\mathcal{D} = \{X, y\}$ of n observations, where X is a D-dimensional independent variable, and y is the response or dependent variable. The Gaussian process regression is a collection of random variables such that any finite subset of the variables has a joint Gaussian distribution. The GP then describes a prior distribution over the latent functions as $f \sim GP\big(m(x), k(x, x^{'})\big)$, where $m(x)$ is a mean function that is often assumed as zero, $k(x, x^{'})$ is the covariate function (kernel) with hyperparameter ψ, and $x, x' \in X$. Using the Bayesian perspective allows GP to provide the predictions and the uncertainties (variances) about the predictions at test points.

A kernel is a positive-definite function of two inputs x, x' and specifies which functions are likely under the GP prior to determining the model's generalization properties[23]. There are various kernel functions, e.g., squared exponential, periodic, and linear kernels. The most widely used kernel function is the squared exponential (SE) function equipped with automatic relevance determination (ARD). The SE kernel is stationary, meaning that its value only depends on the difference $(x - x')$ and not on the absolute locations of x and x':

$$k(x, x^{'}) = \sigma_f^2 \, exp\left(-\frac{1}{2} \sum_{i=1}^{D} \frac{(x_i - x_i^{'})^2}{\mathcal{L}_i} \right)$$

where σ_f^2 is a signal variance, and $\mathcal{L}_i$ is an input length-scale along the ith dimension, and $\psi = \{\sigma_f^2, \mathcal{L}_1, \ldots, \mathcal{L}_D\}$. To train the GP, the hyper-parameters $\theta = \{\sigma^2, \psi\}$ should be determined such that they maximise the log-marginal likelihood

$$log p(y|X) = -\frac{1}{2} y^T (K + \sigma^2 I)^{-1} y - \frac{1}{2} log|K + \sigma^2 I| - \frac{n}{2} log 2\pi. \tag{1.1}$$

Let X^* be the test set of size n_t. For a test point x^*, the joint distribution of the observed target values, y, and the test outputs y^* according to the prior is

$$\begin{bmatrix} y \\ y^* \end{bmatrix} \sim \mathcal{N}\left[0, \begin{bmatrix} k(X, X) & k(X, x^*) \\ k(x^*, X) & k(x^*, x^*) \end{bmatrix} \right].$$

Therefore, the predictive distribution is also a Gaussian distribution $p(y^*|y, X, x^*) =$

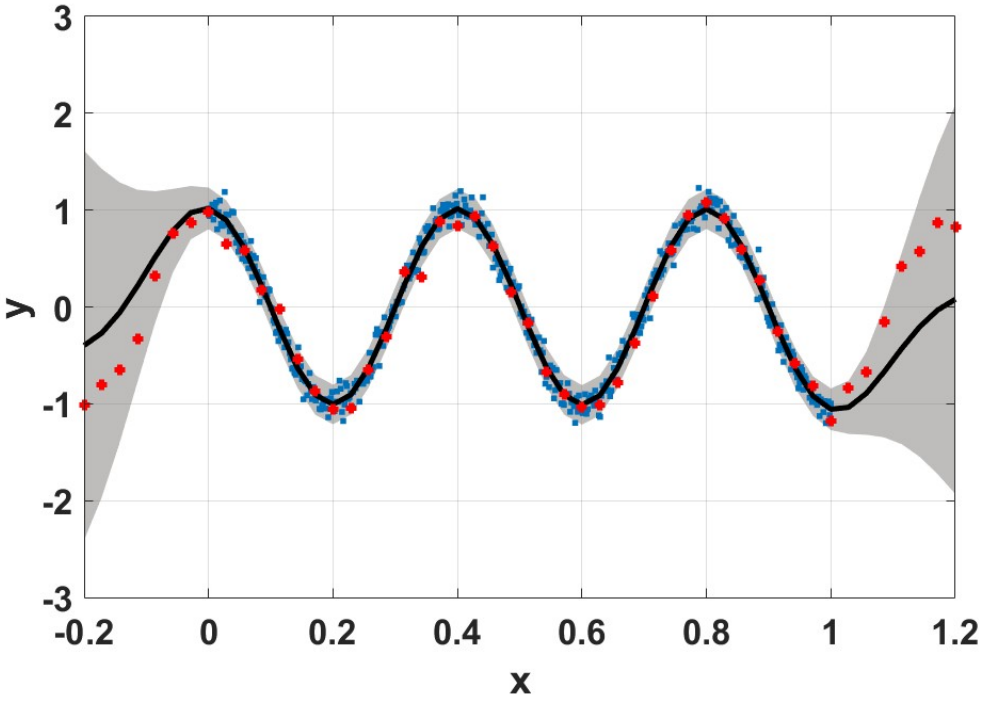


Figure 1.1: **95% Confidence Interval** of GP predictive distribution.

$\mathcal{N}(\mu^*, \Sigma^*)$, which its mean and covariance respectively given by

$$\mu^* = k_*^T (K + \sigma^2 I)^{-1} y, \tag{1.2}$$

$$\Sigma^* = k_{**} - k_*^T (K + \sigma^2 I)^{-1} k_*, \tag{1.3}$$

where $K = k(X,X)$, $k_* = k(X,x^*)$, and $k_{**} = k(x^*,x^*)$. Figure 1.1 depicts the mean of the predictive distribution and the related 95% confidence interval of the Gaussian process for 500 training observations and 50 test points from the analytical function $f(x) = cos(5\pi x)$ with Gaussian noise $\varepsilon \sim \mathcal{N}(0,(0.1)^2)$. The blue and red points are training and test points, respectively.

According to (1.1), the training task maximizes the log-marginal likelihood. This optimization task is affected by the inverse and determinant of $n \times n$ matrix $\mathcal{C} = K + \sigma^2 I$ and therefore scales as $\mathcal{O}(n^3)$. Indeed, the prediction cost is $\mathcal{O}(n^2)$ due to vector and matrix operations in Equations (1.2) and (1.3). Therefore, training and prediction steps are time-consuming tasks for large data sets and impose a limitation on the scalability

of GP. This issue currently restricts GPs to relatively small training data sets, the size of which is typically in the order of $\mathcal{O}(10^4)$.

1.2 Scaling Gaussian Processes for Large Data Sets

Various solutions have been proposed to mitigate the training cost of Gaussian processes in large data sets. The main difference between the solutions is how they deal with the entire training set and inference methods. Depending on their training set strategies, they can usually be classified into two main categories: global approximation and local approximation [24, 25].

Global approximation. This strategy operates on a small subset of the training data set. The methods that follow this strategy train a Gaussian process on the smaller subset and then generalize the results. A simple approach in this area is subset-of-data (SoD) [26, 27], which only uses a subset of size m of the training data set $\mathcal{D}$. To select the data points for the subset, one could randomly sample m data points from $\mathcal{D}$ or use clustering techniques to partition the data into m subsets and choose their centroids as subset points. Thus, the training complexity of SoD is $\mathcal{O}(m^3)$ where $m << n$. However, it has limited efficiency because it ignores the remaining data.

Another method, called sparse kernel or compactly supported kernel [28, 29], provides a sparse representation of the original kernel to reduce the training cost. It ignores the observations that are not correlated or show a covariance smaller than a threshold. In stationary kernels, if the distance between two different entries is more significant than a determined value, their covariance can be set to zero. Although the training cost of this method is $\mathcal{O}(\alpha n^3)$ with $0 < \alpha < 1$, there is no guarantee that the new modified kernel is positive semi-definite.

The most popular method in this area is the sparse approximation approach, which employs a global subset of the data (called inducing points) and low-rank matrix approximations (e.g., *Nyström* method) to approximate the prior, and posterior distributions [30, 31, 32]. For m inducing points, this approach reduces the training complexity to $\mathcal{O}(nm^2)$. Although the sparse approximation provides a full probabilistic model using the Bayesian framework, its capability is restricted by the number of inducing points. It

can not be used for large and high dimensional data sets [33, 34].

Local approximation. Unlike global approximation, this strategy uses the entire training data set. It divides the observations into some partitions, trains the local GPs in each partition, and then aggregates the local approximations [1, 35, 36, 37, 38]. Compared to global approximations, this divide-and-conquer approach can model complex systems and non-stationary data features. Besides, localized experts can capture the model's local patterns and specific behavior. If the entire training set is divided into M partitions, the local approximation can reduce the training cost to $\mathcal{O}(nm_0^2)$, where $m_0 = \frac{n}{M}$. Figure 1.2 summarizes the computational costs of both local and global approximation models.

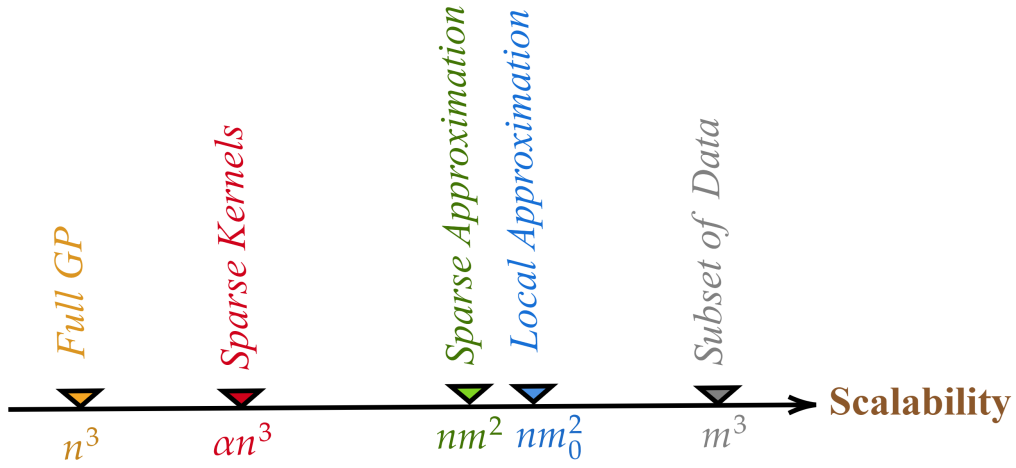


Figure 1.2: **Scalability** of local and global approximation models, where $0 < \alpha < 1$; m is the inducing size for sparse approximations and the subset size for subset-of-data, and m_0 is the partition size in local approximations.

The most popular local approximation methods are the mixture of experts (MoE), the product of experts (PoE), and the Bayesian committee machine (BCM), which inherit the advantages of naive-local experts but boost the predictions through model averaging. The MoE is a Gaussian mixture model that combines the local experts with their hyper-parameters and improves the overall predictive power [39, 40]. Although this joint training positively affects the predictive power and also helps to control the experts with poor performance but increases the complexity [37].

The product of experts (PoE) [41] and Bayesian committee machine (BCM) [42] methods provide a new framework for GPs. Independent experts are GPs that are learned

separately. Both approaches suffer from the discontinuity issue and the weak experts' problem. The generalized product of experts (GPoE) [37] and robust Bayesian committee machine (RBCM) [1] propose different aggregation criteria, which are more robust to weak experts' predictions.

However, the main limitation of local aggregations is that they cannot provide a complete probabilistic model, which results in the so-called *Kolmogorov* inconsistency [43]. It means that for $x_1^*, x_2^* \in X^*$, the predictive distribution $p(y_1^*|\mathcal{D}, x_1^*)$ provided by the aggregation procedure of the local approximation is not marginal over $p(y_2^*|\mathcal{D}, x_2^*)$ of the multivariate predictive distribution:

$$p(y_1^*|\mathcal{D}, x_1^*) \neq \int p(y_1^*, y_2^*|\mathcal{D}, x_1^*, x_2^*) dy_2^*.$$

1.3 Distributed Gaussian Processes

The term distributed Gaussian process (DGP), which includes PoE, BCM, and their derivatives, was proposed by [1]. It uses the fact that the computations of the standard GP can be distributed amongst individual computing units. To do that, one divides the full training data set $\mathcal{D}$ into M partitions (called experts) and trains standard GPs on these partitions. Let $\mathcal{D}' = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ be the partitions, and X_i and y_i be the input and output of partition $\mathcal{D}_i$. All GP experts are trained jointly and share a single set of hyper-parameters $\theta = \{\sigma^2, \psi\}$. Sharing hyper-parameters leads to automatic regularization for protection from local over-fitting. For a test set X^* of size n_t, the predictive distribution of the i-th expert $\mathcal{M}_i$ is $p_i(y^*|\mathcal{D}_i, X^*) = \mathcal{N}(\mu_i^*, \Sigma_i^*)$, where its mean and covariance are, respectively:

$$\mu_i^* = k_{i*}^T (K_i + \sigma^2 I)^{-1} y_i, \tag{1.4}$$

$$\Sigma_i^* = k_{**} - k_{i*}^T (K_i + \sigma^2 I)^{-1} k_{i*}, \tag{1.5}$$

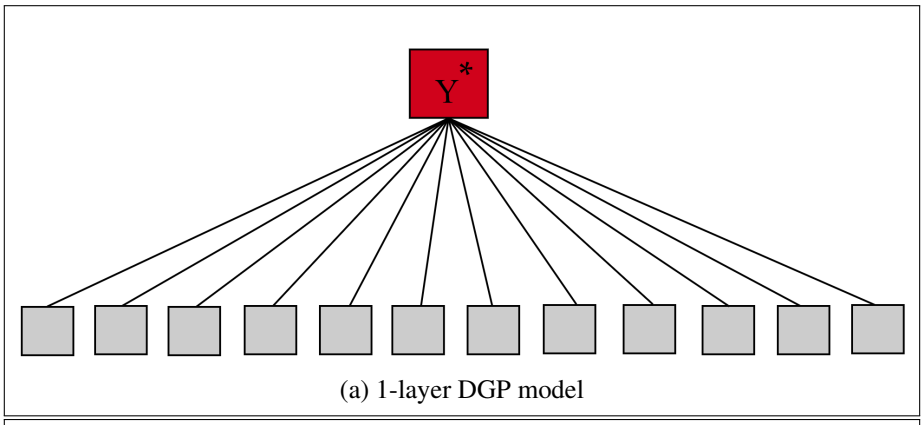
where $K_i = k(X_i, X_i)$, $k_{i*} = k(X_i, X^*)$, and $k_{**} = k(X^*, X^*)$.

Distributed Gaussian processes aggregate local predictive distribution assuming perfect diversity between experts, i.e., they are conditionally independent (CI). This as-

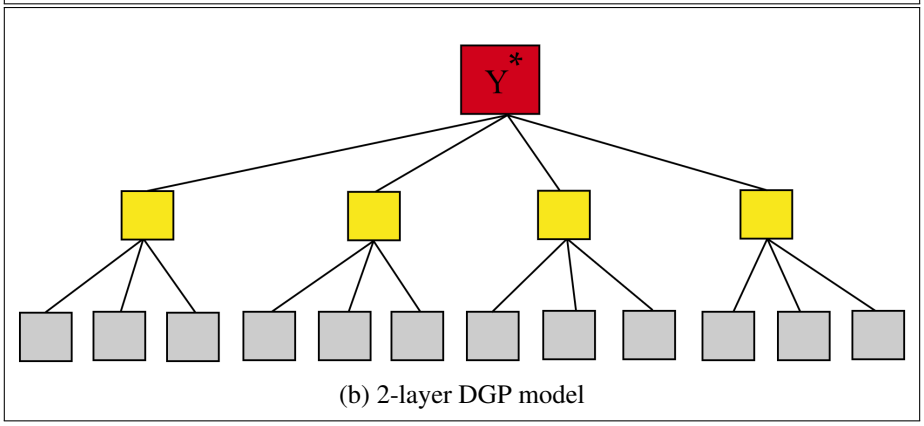
sumption factorizes the aggregated posterior distribution as the product of multiple local densities. That is to say for independent experts $\{\mathcal{M}\}_{i=1}^{M}$ and a test input x^*

$$p(y^*|\mathcal{D},x^*) \propto \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i,x^*), \tag{1.6}$$

where the weights $\beta = \{\beta_1,\ldots,\beta_M\}$ describe the importance and influence of the experts. The most popular aggregation methods are the product of experts (PoE) [41], the generalized product of experts (GPoE) [37], Bayesian committee machine (BCM) [42], robust Bayesian committee machine (RBCM) [1] and generalized robust Bayesian committee machine (GRBCM) [38]. Figure 1.3 depicts the computational graph of the standard and hierarchical local Gaussian process approximations.



(a) 1-layer DGP model



(b) 2-layer DGP model

Figure 1.3: **Computational graphs**: Computational graphs of the standard and hierarchical DGP models [1]. GP experts are at the leaf nodes (gray). All other nodes recombine computations from their direct children (experts). The top node (red) computes the overall prediction.

Two significant scenarios are used for partitioning training points: random and disjoint (e.g., clustering) partitioning. Although random partitioning is faster than clustering methods, it has been accepted that random partitions cannot describe the quickly varying characteristics of the target function. On the contrary, the disjoint partitions provide refined local information, which enables the model to capture the variability of the target function. The advantage of disjoint partitioning has been empirically confirmed in [44, 38].

1.3.1 Product of Experts

In the classical product of experts (PoE) models [41], the desired probability distribution is given as the product of multiple densities (i.e., the experts). If the experts $\{\mathcal{M}\}_{i=1}^{M}$ are independent, the predictive distribution for a test input x^* is

$$p(y^*|\mathcal{D},x^*) = \prod_{i=1}^{M} p_i(y^*|\mathcal{D}_i,x^*). \tag{1.7}$$

The experts are trained on different partitions $\mathcal{D}_i$. According to (1.7), weak experts can considerably affect the PoE's predictive distribution. To come up with this issue, authors in [37] proposed the generalized product of experts (GPoE) as

$$p(y^*|\mathcal{D},x^*) = \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i,x^*), \tag{1.8}$$

where the $\beta = \{\beta_1,\ldots,\beta_M\}$ describes the importance and influence of the experts. The product distribution in (1.8) is proportional to a Gaussian distribution with mean and precision, respectively:

$$\mu_D^* = \Sigma_D^* \sum_{i=1}^{M} \beta_i (\Sigma_i^*)^{-1} \mu_i^*, \tag{1.9}$$

$$(\Sigma_D^*)^{-1} = \sum_{i=1}^{M} \beta_i (\Sigma_i^*)^{-1}. \tag{1.10}$$

The PoE can be recovered for $\beta_i = 1$. The precision of PoE prediction in (1.9) is a linear sum of individual precision values. If the number of local GPs increases, it leads to a rise in precision and, therefore, a decrease in variance, which returns overconfident

predictions in areas with no training data. The authors in [37] suggested a varying value for β_i as the difference in differential entropy between the prior and posterior distribution of each expert, $\beta_i = \frac{1}{2}(\log k_{**} - \log \Sigma_i^*)$. With this, GPoE can increase or decrease the importance of experts based on their prediction uncertainty.

However, the varying weights may produce undesirable and unreasonable errors and high prediction variance for GPoE when leaving the training data [1, 38]. To describe this issue, assume x^* is a test point far away from the training data. Due to the definition of the local variance in Equation (1.5), $k_{i*} \to 0$, and thus $\Sigma_i^* \to k_{**}$. This results in $\beta_i \to 0$, and $\Sigma_D^* \to \infty$, see Equation (1.10). Besides, the predictive distribution of GPoE with uniform weights $\beta_i = \frac{1}{M}$ [1] asymptotically converges to the full Gaussian process distribution but is too conservative (it has a high prediction variance) [38, 45].

1.3.2 Bayesian Committee Machine

Another distributed GP model is the Bayesian committee machine (BCM) [42]. Unlike the PoE family, it uses the Gaussian process prior $p(y^*|x^*)$ when aggregating the experts' predictions. It imposes a conditional independence assumption, i.e., $\mathcal{D}_i \perp\!\!\!\perp \mathcal{D}_j | y^*$ for two experts i and j. Inspired by GPoE, robust Bayesian committee machine (RBCM) [1] improves the efficiency of BCM, especially on the regions with only few data points, by adding the importance weights β_i. The predictive distribution of this family is defined as

$$p(y^*|\mathcal{D}, x^*) = \frac{\prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i, x^*)}{p^{\sum_{i=1}^{M}\beta_i - 1}(y^*|x^*)}, \tag{1.11}$$

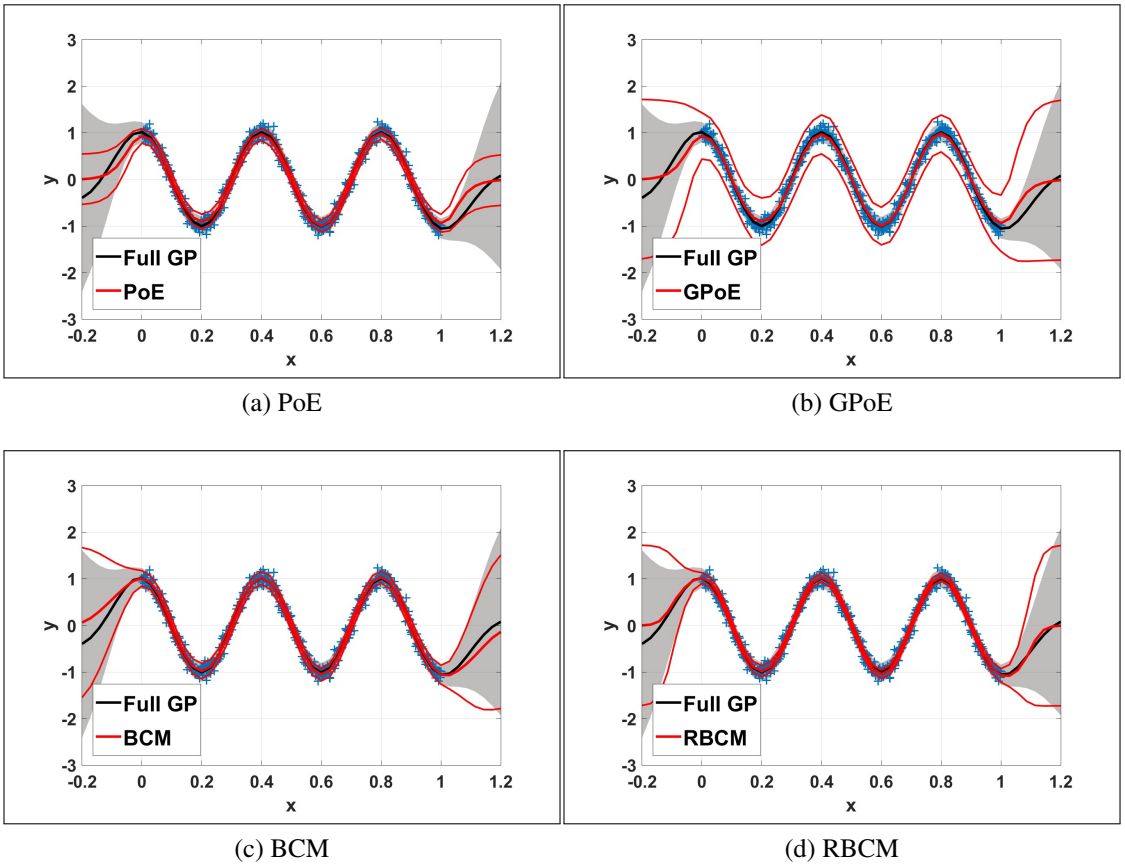
and its mean and precision respectively are

$$\mu_D^* = \Sigma_D^* \sum_{i=1}^{M} \beta_i (\Sigma_i^*)^{-1} \mu_i^*, \tag{1.12}$$

$$(\Sigma_D^*)^{-1} = \sum_{i=1}^{M} \beta_i (\Sigma_i^*)^{-1} + (1 - \sum_{i=1}^{M} \beta_i)(\Sigma^{**})^{-1} \tag{1.13}$$

where the $(\Sigma^{**})^{-1}$ is the prior precision of $p(y^*)$. The available choice of the weights is the difference in differential entropy between the prior $p(y^*|x^*)$ and the posterior $p(y^*|\mathcal{D}, x^*)$, i.e., $\beta_i = \frac{1}{2}(log\Sigma^{**} - log\Sigma_i^*)$, $i = 1, \ldots, M$. For $\beta_i = 1$ and $\beta_i = \frac{1}{M}$, the basic

BCM and GPoE models are recovered, respectively. In RBCM and GPoE with varying β_i, the predictions can not recover the full GP prediction when $M = 1$. In this case we should have $\beta = 1$, while usually $\beta = \frac{1}{2}(log\Sigma^{**} - log\Sigma^{*}_{full}) \neq 1$.



(a) PoE (b) GPoE

(c) BCM (d) RBCM

Figure 1.4: The characteristics of various local approximations compared with standard Gaussian process.

Figure 1.4 shows the prediction with four different distributed Gaussian process models. The standard GP to be approximated is shown by the shaded area, representing the 95% confidence interval of the GP. Training data is displayed as a blue plus sign. We assign five data points to each GP expert from the analytical function $f(x) = cos(5\pi x)$ with Gaussian noise $\varepsilon \sim \mathcal{N}(0, (0.1)^2)$. The standard PoE model does not fall back to the prior when leaving the training data. The generalized PoE model falls back to the prior distribution but overestimates the variances in the region of the training data. The BCM is close to the full GP in the range of the training data, but the predictive mean becomes

unreliable outside the range of the training data. The RBCM is more robust to weak experts than the (G)PoE and BCM and produces reasonable predictions.

The **generalized robust Bayesian committee machine (GRBCM)** is a modified version of RBCM proposed in [38], which introduces a base (global) expert and considers the covariance between the base and other local experts. For a global expert $\mathcal{M}_b$ in a base partition $\mathcal{D}_b$, the predictive distribution of GRBCM is

$$p(y^*|\mathcal{D},x^*) = \frac{\prod_{i=2}^{M} p_{bi}^{\beta_i}(y^*|\mathcal{D}_{bi},x^*)}{p_b^{\sum_{i=2}^{M}\beta_i-1}(y^*|\mathcal{D}_b,x^*)}, \tag{1.14}$$

where the $p_b(y^*|\mathcal{D}_b,x^*)$ is the predictive distribution of $\mathcal{M}_b$, and $p_{bi}(y^*|\mathcal{D}_{bi},x^*)$ is the predictive distribution of a new expert $\mathcal{M}_{bi}$ trained on the data set $\mathcal{D}_{bi} = \{\mathcal{D}_b,\mathcal{D}_i\}$.

The GRBCM is an RBCM method in which the partitions are enhanced with more data, and the Gaussian process prior $p(y^*|x^*)$ is replaced by the prior distribution of the global expert. The data points of the global partition $\mathcal{D}_b$ are always randomly sampled. Therefore, the experts trained at $\mathcal{D}_{bi}, i = 2,\ldots,M$ can almost cover the entire data range and improve the prediction quality of the RBCM. Due to the construction of new experts $\{\mathcal{M}_{bi}\}_{i=2}^{M}$, the proposed GRBCM has a higher time complexity in prediction, compared to (G)PoE and (R)BCM.

The prediction cost of the full GP, $\mathcal{O}(n^2)$, is reduced by DGP methods. Since they need the predictions of all the experts at each test point x^*, their prediction cost is $\mathcal{O}(Mm_0^2)$ or $\mathcal{O}(nm_0)$, where m_0 in the number of assigned points to each expert and $M = \frac{n}{m_0}$ [24]. Due to the construction of new experts, the prediction process in the GRBCM model has a higher computational cost. It needs to combine the predictions of the global expert $\mathcal{M}_{bi}$ and $(M-1)$ new experts $\mathcal{M}_{bi}, i = 2,\ldots,M$ at n_t test points.The time complexity in the prediction process of GRBCM is almost equivalent to $\mathcal{O}(8nm_0^2) + \mathcal{O}(4n_tnm_0)$, where n_t is the size of the test set, see discussion in [38].

1.3.3 Consistency

Conventional CI-based baselines (i.e., baselines that assume conditional independence between experts) for DGPs suffer from *inconsistency*. Since the local experts are trained on separated partitions, the aggregation produces inconsistent predictions which can

not converge to the standard GP. Several works have studied the asymptotic properties of the CI-based ensembles; see, for example [44, 45, 46]. It has been confirmed that the DGP methods, i.e., (G)PoE and (R)BCM, are inconsistent. Mainly, these families of ensembles provide overconfident results. Besides, GPoE with normalized equal weights [1] asymptotically converges to the full GP distribution. However, it is too conservative [45, 46].

The authors of [38] considered the consistency issue of DGP models according to the partitioning strategies, i.e., random partitioning and disjoint partitioning. They showed that in both partitioning scenarios, PoE and (R)BCM produce overconfident prediction variance, and their prediction variances will shrink to zero when $n \to \infty$. Besides, GPoE yields conservative prediction variance in disjoint partitioning while it produces consistent predictions using random partitions under some mild assumptions.

By introducing a base (global) expert, GRBCM considers the covariance between the base and other local experts, which, under some mild assumptions, can provide consistent results in disjoint and random partitioning scenarios. However, it still uses the conditional independence assumption between non-global experts, which sometimes yields poor results, particularly when the data is randomly partitioned.

The CI assumption is the primary cause for the asymptotic issues of DGP models. The following section considers dependencies between experts and related aggregation that uses experts' correlations.

1.3.4 Dependency Between Experts

The main reasons to use an ensemble over a single model are their performance and robustness. An ensemble method can provide better predictions and reduce the spread or dispersion of the model predictions [47]. The CI-based ensemble methods are widely used for regression and classification problems [48, 49]. Local approximation GPs use CI to reduce the computational costs of the training and prediction processes. However, this assumption is often violated in practice. Thus, their predictions are not accurate enough, and the aggregation based on this assumption generally returns a sub-optimal solution [2].

In classification, modeling the dependent classifiers has been considered in several

works, for example, in [2, 50, 51]. In local approximation GPs, the dependency between experts has been discussed in few works. The primary method in this area is the nested pointwise aggregation of experts (NPAE) [44, 46] that employs the internal correlation between experts and the target variable y^* to define an estimator. Figure 1.5 depicts the computational graphs of aggregation strategies with CI assumption and dependency.

Figure 1.5(a) reveals the aggregation based on the CI assumption between experts $\{\mathcal{M}_1, \mathcal{M}_2, \mathcal{M}_3, \mathcal{M}_4\}$. It means that two local experts $\mathcal{M}_i$ and $\mathcal{M}_j$ are independent of each other given the target variable y^*, i.e. $\mathcal{M}_i \perp\!\!\!\perp \mathcal{M}_j \mid y^*$. On the other hand, Figure 1.5(b) represents an aggregation with dependent experts where the interactions between experts show the dependencies.

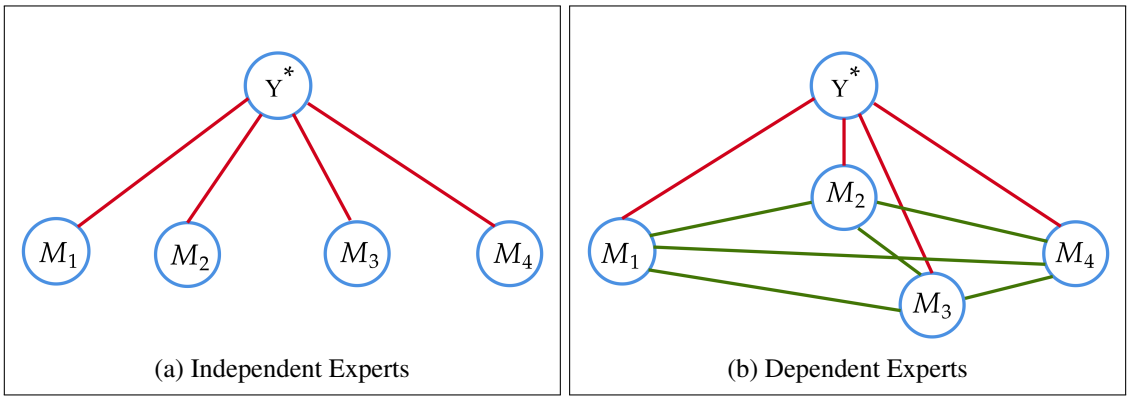


Figure 1.5: **Computational graphs** of (a) an aggregation based on conditional independence and (b) an aggregation based on conditional dependency between local experts.

Assume the Gaussian experts $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ have been trained on different partitions and the first and second moments of their local posterior distributions have been defined in (1.4) and (1.5). Let $\mu^*(x^*) = [\mu_1^*(x^*), \ldots, \mu_M^*(x^*)]^T$ be an $M \times 1$ vector that contains the centered predictions of M experts for a given test point $x^* \in X^*$. We first assume that y_i in (1.4) has not yet been observed. Therefore, the mean of the predictive distribution $\mu_i^*(x^*)$ can be considered as a *random variable*. This allows us to consider correlations between the experts' predictions and latent variable y^*, $cov(\mu_i^*, y^*)$, and also leverage internal correlations between experts, $cov(\mu_i^*, \mu_j^*)$ where $i, j = 1, \ldots, M$. According to (1.4), the local experts are linear estimators: $\mu_i^* = \Gamma_i y_i$, where $\Gamma_i = k_{i*}^T (K_i + \sigma^2 I)^{-1}$. Using this result, we can find the analytical expression for both covariances:

$$cov(\mu_i^*, y^*) = cov(\Gamma_i y_i, y^*) = \Gamma_i cov(y_i, y^*) = \Gamma_i k(X_i, X^*), \tag{1.15}$$

$$cov(\mu_i^*, \mu_j^*) = cov(\Gamma_i y_i, \Gamma_j y_j) = \Gamma_i cov(y_i, y_j)\Gamma_j^T = \Gamma_i k(X_i, X_j)\Gamma_j^T. \tag{1.16}$$

1.3.5 Dependency Based Aggregation of Experts

As we assumed, the means (predictions) of the local predictive distributions are random variables. Then $k_A(x^*) = cov(\mu^*(x^*), y^*(x^*))$ and $K_A(x^*) = cov(\mu^*(x^*), \mu^*(x^*))$ are the point-wise covariances at x^*. For each test point x^*, $k_A(x^*)$ is a $M \times 1$ vector and $K_A(x^*)$ is a $M \times M$ matrix and according to (1.15) and (1.16) their elements are defined as $k_A(x^*)_i = \Gamma_i k(X_i, x^*)$ and $K_A(x^*)_{ij} = \Gamma_i k(X_i, X_j)\Gamma_j^T$, where $\Gamma_i = k(X_i, x^*)^T (K_i + \sigma^2 I)^{-1}$ and $i, j = 1, \ldots, M$. The task is to aggregate variables $\mu_i^*(x^*), i = 1, \ldots, M$, into a unique predictor $y_A^*(x^*)$ of $y^*(x^*)$.

Due to the prior choice, the joint distribution of random variables $(y^*, \mu_1^*, \ldots, \mu_M^*)$ is a multivariate normal distribution because any vector of linear combinations of normally distributed observations is itself a Gaussian vector. This fact is used to define the aggregated predictor, but it also implies that the experts' predictions $(\mu_1^*, \ldots, \mu_M^*)$ follow a multivariate Gaussian. Employing properties of conditional Gaussian distributions for the centered random vector $(y^*(x^*), \mu^*(x^*))$ allows for the subsequent aggregation:

Definition 1 (**Aggregated predictor**). For the test point x^* and sub-model predictions $\mu_1^*(x^*), \ldots, \mu_M^*(x^*)$, the aggregated predictor is defined as

$$y_A^*(x^*) = k_A(x^*)^T K_A(x^*)^{-1} \mu^*(x^*). \tag{1.17}$$

Here, y_A^* is the mean of the conditional distribution of y^* given μ^*, i.e. $p(y^*|\mu^*)$. The following Proposition shows that the linear estimator in (1.17) is the *best linear unbiased predictor* (BLUP) of y^*.

Proposition 1 (**BLUP**). $y_A^*(x^*)$ is the best linear unbiased predictor of $y^*(x^*)$, i.e. for linear estimators of the form $\beta\mu^* = \sum_{i=1}^M \beta_i \mu_i^*(x^*)$, the mean square error $(y^* - \beta\mu^*(x^*))^2$ is minimized when $\beta = k_A(x^*)^T K_A(x^*)^{-1}$.

Proof. The proof is straightforward. We need to show that $var(y^*(x^*) - \beta\mu^*(x^*)) -$

$var\left(y^*(x^*) - y_A^*(x^*)\right)$ is positive semi-definite for all linear unbiased predictors $\beta\mu^*(x^*)$. To do that, we extend the $var(y^*(x^*) - \beta\mu^*(x^*))$:

$$var(y^*(x^*) - \beta\mu^*(x^*)) = var(y^*(x^*) - y_A^*(x^*) + y_A^*(x^*) - \beta\mu^*(x^*)) =$$
$$= var(y^*(x^*) - y_A^*(x^*)) + var(y_A^*(x^*) - \beta\mu^*(x^*)) + 2cov(y^*(x^*) - y_A^*(x^*), y_A^*(x^*) - \beta\mu^*(x^*)).$$

Now, we show $cov\left(y^*(x^*) - y_A^*(x^*), C\mu^*(x^*)\right) = 0, \forall C$.

$$\begin{aligned} &cov(y^*(x^*) - y_A^*(x^*), C\mu^*(x^*)) = \\ &= cov(y^*(x^*), C\mu^*(x^*)) - cov(y_A^*(x^*), C\mu^*(x^*)) = \\ &= k_A(x^*)^T C^T - k_A(x^*)^T K_A(x^*)^{-1} K_A(x^*) C^T = \\ &= k_A(x^*)^T C^T - k_A(x^*)^T C^T = 0. \end{aligned}$$

Therefore,

$$cov(y^*(x^*) - y_A^*(x^*), y_A^*(x^*) - \beta\mu^*(x^*)) = 0$$

where in this case, $C = k_A(x^*)^T K_A(x^*)^{-1} - \beta$. It means

$$var(y^*(x^*) - \beta\mu^*(x^*)) - var(y^*(x^*) - y_A^*(x^*)) = var(y_A^*(x^*) - \beta\mu^*(x^*)) \geq 0$$

because $var\left(y_A^*(x^*) - \beta\mu^*(x^*)\right)$ is positive semi-definite variance matrix. □

This method is known as the nested pointwise aggregation of experts (NPAE) and provides high-quality predictions. The non-invertibility condition on $K_A(x^*)$ can be avoided using pseudo-inverses. This aggregated predictor has Gaussian distribution, and its moments can easily be calculated using $k_A(x^*)$ and $K_A(x^*)$ as:

$$\mathbb{E}[y^*(x^*)|\mu^*(x^*)] = y_A^*(x^*) = k_A(x^*)^T K_A(x^*)^{-1} \mu^*(x^*) \tag{1.18}$$

$$\mathbb{E}[y^*(x^*)|\mu^*(x^*)] = k(x^*, x^*) - k_A(x^*)^T K_A(x^*)^{-1} k_A(x^*) \tag{1.19}$$

By using the dependencies between experts, NPAE theoretically provides consistent predictions [44, 46]. However, its aggregation step suffers from high time complexity

because it needs to determine the inverse of a $M \times M$ covariance matrix between experts at *each* test point x^*, i.e., $K_A(x^*)$. Therefore, the computational cost of NPAE depends on the number of experts M, and the size of the test set, i.e., almost $\mathcal{O}(n_t M^3)$, where n_t is the number of available test points.

The authors of [44] showed that the prediction cost of the NPAE can be defined as $\mathcal{O}(n_t n^2)$. It also can be reduced to some extent by employing a hierarchical computing structure that cannot provide identical predictions. However, NPAE leads to an impractical run time for many partitions and large test sets.

1.4 Gaussian Graphical Model

Undirected graphical models (known as pairwise Markov random fields (MRF)) provide a framework for specifying the joint distribution over large numbers of random variables. This framework uses a matrix of parameters to describe the graph structure and encodes the edges as parameters. If there is a connection between two nodes, a non-zero parameter indicates the existence of an edge between them in the graph. Since the graphical model is used in the next chapter to detect dependencies between GP experts, the fundamental aspects of this method are discussed here.

1.4.1 Essential Assumption

The Gaussian graphical model (GGM) is a prominent graphical model with continuous Gaussian random variables [52, 53, 54]. The basic assumption underlying GGMs is that all variables in the network follow a multivariate Gaussian distribution. The distribution for GGMs is

$$p(\mu^*|\xi,\Sigma) = \frac{1}{(2\pi)^{M/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(\mu^* - \xi)^T \Sigma^{-1}(\mu^* - \xi)\right\}, \tag{1.20}$$

where $\mu^* = \{\mu_1^*, \ldots, \mu_M^*\}$ are the variables, M is the number of variables, and ξ and Σ are the mean and covariance, respectively. The distribution in (1.20) can also be ex-

pressed using the precision matrix Ω:

$$
\begin{aligned}
p(\mu^*|\eta,\Omega) &= \frac{|\Omega|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mu^*-\eta)^T\Omega(\mu^*-\eta)\right\} \\
&\propto \exp\left\{-\frac{1}{2}\mu^{*T}\Omega\mu^* + \eta^T\mu^*\right\},
\end{aligned}
\tag{1.21}
$$

where $\Omega = \Sigma^{-1}$ and $\eta = \Omega\xi$. The matrix Ω is also known as the potential or information matrix.

Without loss of generality, let $\xi = 0$, then the distribution of a GGM shows the potentials defined on each node i as $exp\{-\Omega_{ii}(\mu_i^*)^2\}$ and on each edge (i,j) as $exp\{-\Omega_{ij}\mu_i^*\mu_j^*\}$. In a correlation network, if $\Sigma_{ij} = 0$, then μ_i^* ard μ_j^* are assumed to be independent. In a GGM, if $\Omega_{ij} = 0$, then μ_i^* ard μ_j^* are conditionally independent given all other variables, i.e., there is no edge between μ_i^* ard μ_j^* in the graph[1].

1.4.2 Network Learning and Graphical Lasso

Network learning results in the precision matrix Ω that reveals the conditional dependencies between experts. GGMs use the common sparsity assumption; there are only few edges in the network, and thus the precision matrix is sparse. This assumption usually makes sense in experts' networks because the interaction of one expert is limited to only a few other experts. Since the local predictions of this expert are close to the local predictions of its adjacent experts, it has stronger interactions (or similarities) with them; see [55] for more information about similarity and dissimilarity.

To this end, the Lasso regression [56] is used to estimate the network's precision matrix and neighborhood selection. The Meinshausen-Bühlmann algorithm [57] is one of the first algorithms in this area. [57] and [58] proved that with some assumptions, Lasso asymptotically recovers the correct relevant subsets of edges. In [59], the efficient graphical Lasso (GLasso) was proposed, which adopts a maximum likelihood approach subject to an $\mathcal{L}_1$ penalty on the coefficients of the precision matrix. The graphical Lasso has been improved in later works [60, 61, 62].

Let S_{μ^*} be the sample covariance of experts' predictions, i.e., $S_{\mu^*} = cov(\mu^*)$. Then,

[1] We would like to emphasize that this does not hold in general, but here it does hold since we assumed that all variables are jointly Gaussian distributed.

the Gaussian log-likelihood of the precision matrix Ω is equal to

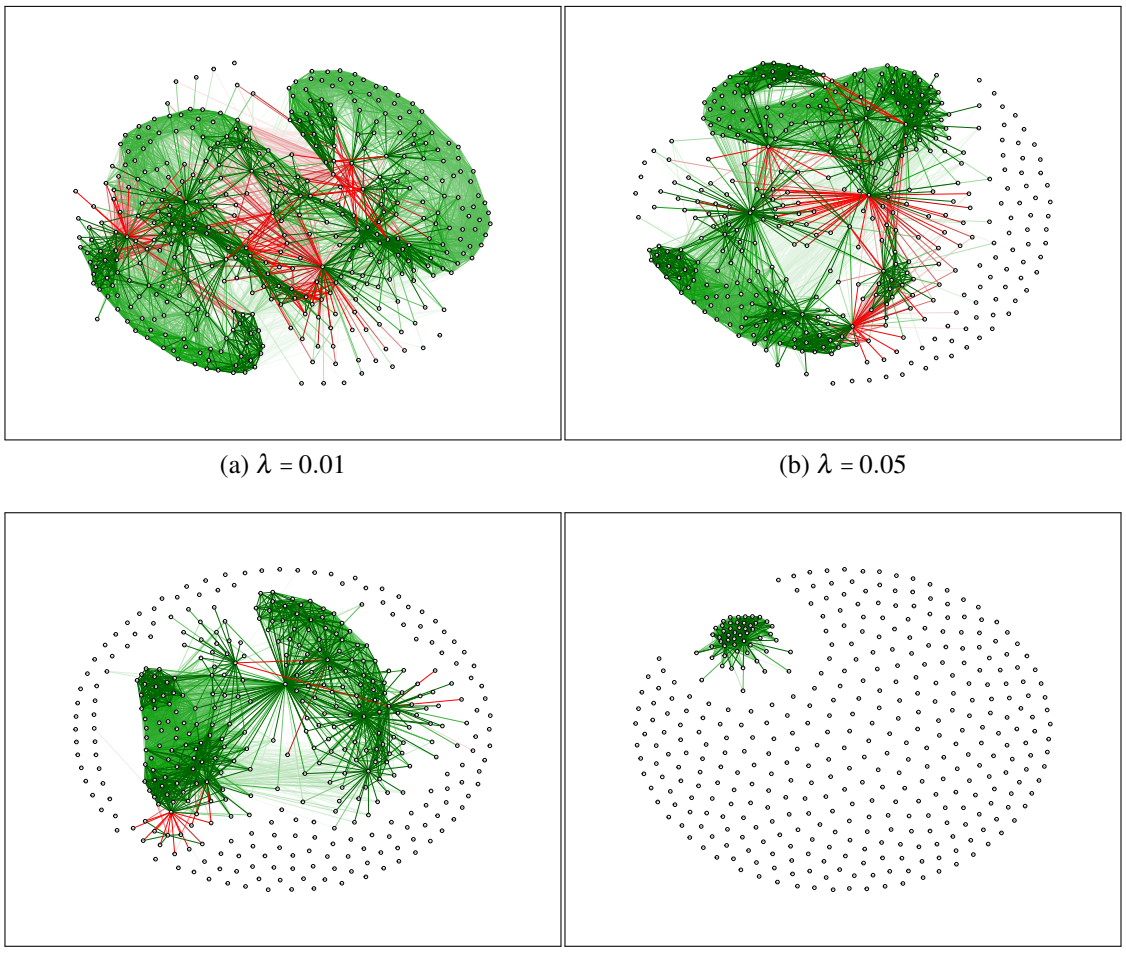
$$\mathcal{L}(\Omega; S_{\mu^*}) = log|\Omega| - trace(S_{\mu^*}\Omega).$$

The graphical Lasso maximizes this likelihood subject to an element-wise $\mathcal{L}_1$ norm penalty on Ω. Precisely, the objective function is

$$\hat{\Omega} = \underset{\Omega \in \mathcal{R}^{M \times M}}{\arg\min} \; -\mathcal{L}(\Omega; S_{\mu^*}) + \lambda \, \|\Omega\|_1 , \tag{1.22}$$

where the estimated neighborhood is then the non-zero elements of $\hat{\Omega}$ and λ is the sparsity parameter.

Figure 1.6 depicts the network structure in a GGM with varying penalty strength λ using 200 experts. The red coloured edges represent positive connections ($\Omega_{ij} > 0$), while the green edges reflect negative dependencies ($\Omega_{ij} < 0$). The network keeps the most relevant edges by increasing the penalty term. The graph in Figure 1.6(a) contains many edges with low thicknesses. On the other hand, the graph in Figure 1.6(d) only contains the most relevant nodes. For the disconnected expert (node) i, Ω_{ij} is approximately 0 for all all j.



(a) $\lambda = 0.01$ (b) $\lambda = 0.05$

(c) $\lambda = 0.1$ (d) $\lambda = 0.5$

Figure 1.6: Sparsity of a GGM for 5×10^4 synthetic data and $M = 200$ experts with varying penalty strength λ.

Chapter 2

Our Contribution

This chapter will summarize the papers we have authored on the direction of dependency in distributed Gaussian processes during this Ph.D. study. The motivation, methodology, and findings are according to the discussed principles laid out in Chapter 1. The perfect diversity between Gaussian experts is often violated in practice and, therefore, an aggregation may not provide high-quality predictions. Besides, modeling the experts' dependencies leads to impractically high computational costs. This chapter will consider these issues in both CI-based and dependency-based ensemble methods.

We first consider the CI-based aggregations and develop a novel strategy to combine local experts using clusters of experts. It can be explained as a boosted 2-layers DGP that uses spectral clustering to define the second layer. After that, we propose a novel aggregation technique for DGPs using the latent variable GGM, where the target variable is defined as a latent variable in an undirected graph. We employ Expectation-Maximization (EM) to estimate the latent variable and update the related parameters of the model. In both scenarios, dependencies between experts are detected using Gaussian graphical models.

The main drawback of the dependency-based ensembles is their complexity, which cubically depends on the number of experts. Therefore, an expert selection strategy that assigns a small set of related experts to each new entry point can significantly reduce the computational cost. Indeed, the expert selection method can improve the prediction quality of the final aggregation by excluding the weak experts at each data point. To come up with such a solution, we first consider the most important experts in a GGM as a set of selected experts and show the practical and theoretical advantages of the expert

selection model. Moreover, we convert the expert selection problem into a multi-label classification task, providing a flexible and entry-dependent expert assignment model. The detailed analysis of all proposed methods and the published papers are available in Appendices A, B, C, D.

2.1 Dependent Experts in Conditional Independence-Based Ensembles

In this section, we consider the dependency between experts in CI-based methods and improve the prediction quality of the aggregation by designing a two-layer computational graph. We mainly focus on GPoE and GRBCM models as aggregation techniques to guarantee that the final predictions converge to the true values.

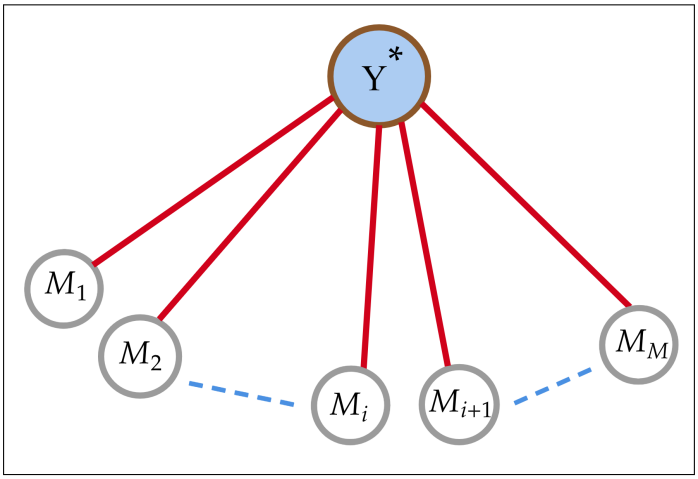
2.1.1 Motivation and Main Methodology

Distributed Gaussian processes can effectively reduce the computational costs of standard Gaussian processes. However, assuming perfect diversity or conditional independency between experts weakens the quality of the ensemble method. The conventional form of the DGP model considered a computational graph with multiple layers [1]. However, the authors only discussed the computational aspect of such a model and did not propose any solution to mitigate the drawbacks of the CI assumption. The modified model that uses a global communication expert [38] also assumes that non-global experts are conditionally independent.

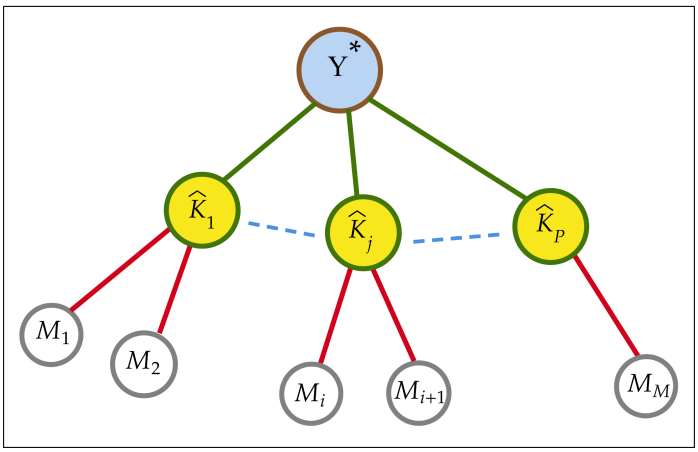
The critical contribution of this section lies in considering the dependency between Gaussian experts in CI-based ensembles and improving the prediction quality efficiently. To this end, we develop an approach to detect the conditional correlation between Gaussian experts to modify the final aggregation. The Gaussian graphical model 1.4 is used to infer and detect dependencies between experts. The precision matrix estimated by GLasso contains all the required information about experts' interactions, resulting in the related GGM. Besides, using this matrix, we can determine the clusters of strongly dependent experts. It means each cluster contains some highly correlated experts, and by gathering them in a cluster, we mitigate the effects of their dependency on the CI-based

ensembles.

Figure 2.1 depicts the computational graph of a standard DGP with CI assumption(a) and a DGP with clusters of correlated experts (b). The leaves $M_i, i = 1, \ldots, M$ are Gaussian experts trained on local partitions. The graph in Figure 2.1(a) assumes perfect diversity between leaves, and there is no experts' interactions. The predictive distribution of target variable y^* is estimated using Equation (1.6). A new layer of P new nodes $\mathcal{K} = \{K_i\}, i = 1, \ldots, P$ is defined in Figure 2.1(b) where each K_i is a cluster of some strongly correlated experts. According to the definition of the new experts, there is perfect diversity between experts, and this layer mitigates the effect of CI assumption on predictive distribution.



(a) Distributed GPs



(b) Clusters of correlated experts

Figure 2.1: **Computational graphs**: (a) DGP model with CI [1]; (b) DGP with clusters of dependent experts [2]

The new experts in $\mathcal{K}$ are defined using spectral clustering (SC) [55]. Performing spectral clustering on the precision matrix Ω returns the clusters of experts $\mathcal{K}$. Thus, each cluster K_i contains strongly dependent experts based on the precision matrix, i.e., the experts' interactions in GGM. Spectral clustering uses the relevant eigenvectors of the Laplacian matrix of the similarity matrix (here, the precision matrix) alongside a standard clustering method. The Laplacian matrix L is $L = D - \Omega$, where D is the degree matrix, a diagonal matrix that includes the sum of the values in each row of the precision matrix Ω.

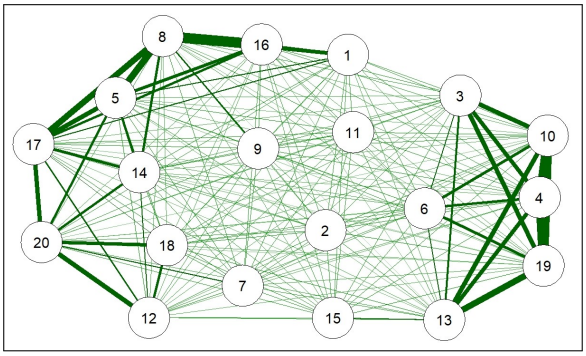
Figure 2.2 depicts the undirected graph of a GGM with 20 experts with sparsity parameter $\lambda = 0.1$. The sample covariance of local predictions is the main ingredient of the GLasso to construct the graph. The GGM in (a) shows the interactions between experts where the experts are divided into $P = 5$ clusters in (b). Although the black and green clusters include more experts, there are clusters with only one expert, e.g., the red cluster. In (c), we can see the amount of interaction in the green cluster. The thickness of the edges between the experts inside this cluster is significantly more than the thickness between these experts and the experts in other clusters, indicating a strong correlation between the nodes in this cluster.

Each cluster K_i consists of Gaussian experts, and the moments of the related predictive distribution can be estimated by the GRBCM method. We choose GRBCM [38] inside the clusters because, under some mild assumption, it can provide statistically consistent results in both random and disjoint partitioning strategies. To aggregate clusters, $K_i, i = 1, \ldots, P$, we employ GPoE [1] or GRBCM [38] because we can show that the moments of the related predictive distribution converge to the true moments of the target variable.

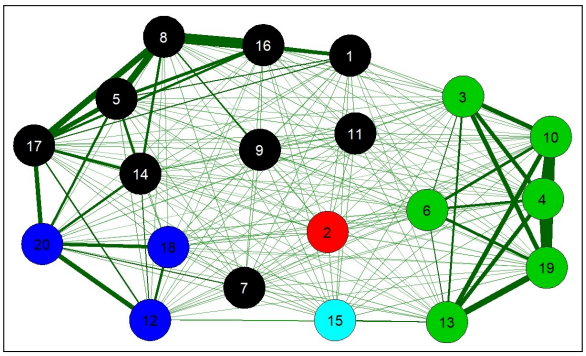
We propose the dependent Gaussian expert aggregation method (DGEA) by aggregating strongly correlated experts at each cluster. The proposed method has significantly higher prediction quality (lower SMSE and MSLL) than the other baselines. Let's consider this analytical function as a toy example:

$$f(x) = 5x^2 \sin(12x) + (x^3 - 0.5)\sin(3x - 0.5) + 4\cos(2x) + \varepsilon, \tag{2.1}$$

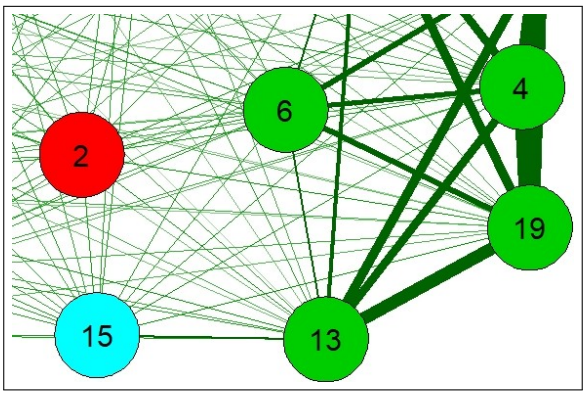
where $\varepsilon \sim \mathcal{N}(0, (0.2)^2)$. To evaluate the prediction quality of the proposed method and available baselines, we generated $n = 10^4$ training points in $[0, 1]$, and $n_t = 0.1\,n$ test points in $[-0.2, 1.2]$. The data is normalized to zero mean and unit variance. The training



(a) GGM, $\lambda = 0.1$



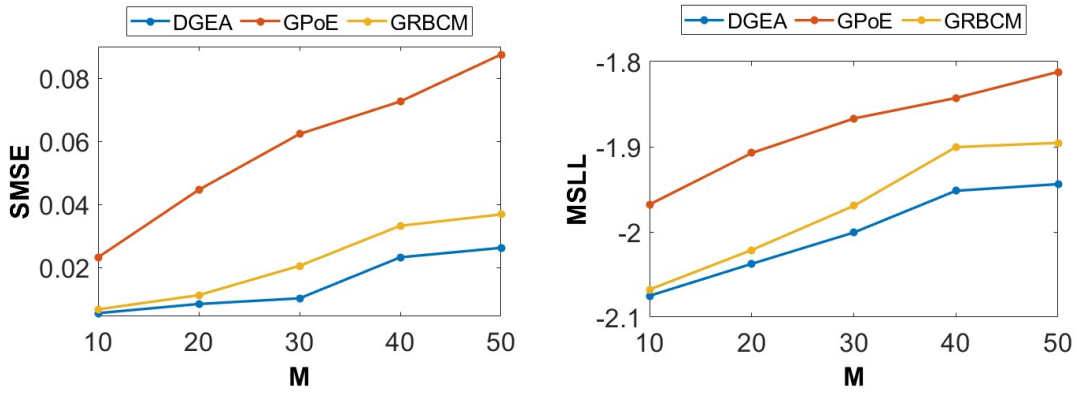
(b) GGM, $\lambda = 0.1$, $P = 5$



(c) Clusters and interactions between experts

Figure 2.2: **Gaussian graphical models**:(a) shows the interaction between experts in a GGM of 20 experts with a penalty term $\lambda = 0.1$, (b) reveals the GGM with 5 clusters of experts, and (c) depicts the interactions between experts in a cluster.

set is divided into $M = \{10, 20, 30, 40, 50\}$ experts. The experts inside the clusters are aggregated using GRBCM, and the clusters' predictions are combined using the GPoE method.

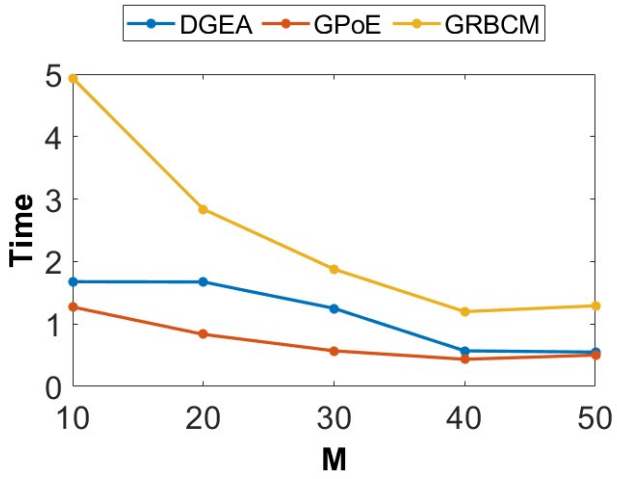


(a) SMSE (b) MSLL

Figure 2.3: **Prediction quality** of different DGP methods with respect for different number of experts in the simulated data of the analytic function by (2.1).

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 prediction mean, while the MSLL evaluates the quality of predictive distribution [10]. Figure 2.3 shows the sensitivity of different ensemble methods concerning the change in the number of experts. The figure indicates that clusters of dependent experts can improve the prediction quality of the final ensemble method. Although the prediction quality of the GPoE is low, the DGEA method that uses GPoE to aggregate clusters K_i significantly outperforms GPoE and GRBCM methods. Raising the number of experts increases the differences between prediction errors. However, DGEA shows better performance in all cases.

Although this approach performs the aggregation several times (for clusters and latent variable y^*), its computational cost is slightly smaller than standard GRBCM. It is because the approach uses few experts at each run. For instance, there are P clusters for final aggregation, which is much less than the number of original experts M. Therefore, the model distributed the aggregation in some clusters of experts, and each cluster has few experts. This advantageous behavior of DGEA over GRBCM is confirmed in Figure 2.4, which shows that DGEA is faster than GRBCM. In fact, the runtime of DGEA is comparable to that of GPoE (the most efficient aggregation approach).



(a) Time

Figure 2.4: **Prediction time** of different DGP methods with respect for different number of experts in the simulated data of the analytic function by (2.1).

2.1.2 Discussion and Future Works

The proposed model enables CI-based ensembles to mitigate the drawbacks of the independence assumption in the predictive distribution. Although the method can also be used for dependency-based aggregation (i.e., the NPAE method), it can not guarantee any improvement in computational cost. It is because spectral clustering divides the nodes into some clusters based on the similarity matrix. However, it is possible to have a cluster with more experts and some with only one expert; therefore, it still has some costly clusters. Even if we increase the number of clusters P to avoid having clusters with more experts, performing NPAE for the final aggregation is expensive because it needs to find the inverse of a $P \times P$ matrix at each test point. Since we increased P, it can become a costly task.

Indeed, the model must run GLasso once at the first step. Since performing GLasso for network learning is a costly method $\mathcal{O}(M^3)$, it can lead to high computational costs in large data sets when M is a high value. However, there are newer, faster methods to learn a GGM that can be used instead of the GLasso, see [63, 64]. For instance, the Fast and Scalable Inverse Covariance Estimator by Thresholding (FST) model [64] reduces the computational complexity of sparse Gaussian Graphical Model to a much lower order of magnitude ($\mathcal{O}(M^2)$).

According to the asymptotic properties of the GPoE and GRBCM model, we can show that our new local approximation approach provides consistent results when $n \to \infty$. Here, we integrated Gaussian graphical models into the generalized robust Bayesian committee machine and the generalized product of experts. Nevertheless, aggregating clusters K_i can be substituted by latent variable GGMs, assuming that the final predictor is a latent variable within the graph. This line of work has been studied in several works, e.g., [65, 66, 67], and can be another alternative for the available baselines. Besides, the GGM relies on the assumption that all experts are jointly Gaussian and cannot be used to explain complex models with the non-Gaussian distribution. However, the normality assumption for joint distribution is not restrictive. In practice, we can relax this assumption and consider random variables without resorting to multi-dimensional Gaussian distribution [68, 69, 70].

2.2 Latent Variable Gaussian Graphical Model as an Ensemble Technique

In the previous section, we considered how standard GGM could improve the prediction quality of the available CI-based baselines. This section proposes a new aggregation method using the latent variable Gaussian graphical model. The experts are assumed to be dependent, and the target variable is a latent variable in the GGM.

2.2.1 Motivation and Main Methodology

Let us consider the estimated means of local Gaussian experts in (1.4) again

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

where $K_i = k(X_i, X_i)$, $k_{i*} = k(X_i, X^*)$, and $k_{**} = k(X^*, X^*)$. Assuming that the y_i has not yet been observed helps to consider the experts' predictions μ_i^* as a *random variable*. It allows us to consider correlations between the experts' predictions and a target variable and leverage internal correlations between experts. Unlike the point-wise correlations in the NPAE method, the related dependencies can be approximated by a latent variable Gaussian graphical model (LVGGM). In particular, instead of only M random variables in standard GGM, the LVGGM considers $M+1$ random variables $\{\mu_1^*, \ldots, \mu_M^*, y^*\}$ where y^* is the random target variable.

Since the targeted expert y^* is unobserved, the set of nodes in GGM is divided into two subsets: *latent* expert and *observed* local experts. The goal is to find the joint distribution of $(y^*, \mu^*) = (y^*, \mu_1^*, \ldots, \mu_M^*)$. According to the choice of prior in GP and the experts' randomness assumption, the joint distribution of $(y^*, \mu^*) \in \mathcal{R}^{M+1}$ is normal with a covariance matrix Σ and precision matrix $\Omega = \Sigma^{-1}$ as

$$\Sigma = \begin{pmatrix} \Sigma_{y^*y^*} & \Sigma_{y^*\mu^*} \\ \Sigma_{\mu^*y^*} & \Sigma_{\mu^*\mu^*} \end{pmatrix}, \text{ and } \Omega = \begin{pmatrix} \Omega_{y^*y^*} & \Omega_{y^*\mu^*} \\ \Omega_{\mu^*y^*} & \Omega_{\mu^*\mu^*} \end{pmatrix}.$$

The GGMs with latent variables have been widely considered over the past decade. For instance, [65] proposed a regularized maximum likelihood approach to estimate

the precision matrix with sparse structure Ω_{μ^*} and the low-rank terms L^* where $L^* = \Omega_{\mu^* y^*} \Omega_{y^*}^{-1} \Omega_{y^* \mu^*}$. The precision matrix in this form is $\Omega_{\mu^*} - L^*$, and the related Gaussian log-likelihood $\mathcal{L}(\Omega_{\mu^*}, L^*; S_{\mu^*})$ is expressed in terms of the sample covariance matrix S_{μ^*}, the precision matrix of observed variables Ω_{μ^*}, and the low-rank term L^*.

It is a misspecified optimization problem because the precision matrix is the sum of two matrices. However, if Ω_{μ^*} is sparse and there are few latent variables, it is possible to decompose the precision matrix into its summands [65, 71]. It leads to a convex optimization problem called *Low-Rank Plus Sparse Decomposition* (LR+SD). The authors of [72] proposed a direct approach via *Expectation-Maximization* algorithm, which converts the LR+SD model to a conventional GGM. The authors of [66] accelerated the LR+SD model via a non-convex optimization and showed that it is faster than the convex relaxation-based methods. In another work [67], the authors considered the model selection task in GGMs in the presence of latent variables. They illustrated the conditions required for identifiability (i.e., under which conditions does the optimization problem admit a unique solution).

In GGMs, GLasso detects the interactions between the experts and returns the joint distribution of random variables. But for (y^*, μ^*), the input of the GLasso method is unknown because the targeted expert y^* is unobserved, and therefore the sample covariance of (y^*, μ^*) can not be calculated. Therefore, the first step is to approximate the sample covariance matrix. Let S be the sample covariance of the full data, i.e., $S = cov(y^*, \mu^*)$, then we can represent S as

$$S = \begin{pmatrix} S_{y^* y^*} & S_{y^* \mu^*} \\ S_{\mu^* y^*} & S_{\mu^* \mu^*} \end{pmatrix}$$

where $S_{\mu^* \mu^*}$ is a known $M \times M$ matrix of the sample covariance of the observed variables μ^*, $S_{y^* \mu^*}$ is an unknown $1 \times M$ vector that shows the sample covariance matrix between latent and observed expert, and $S_{y^* y^*}$ is the internal potential of the latent expert. Performing GLasso needs to estimate unknown partitions of S, i.e., $S_{y^* \mu^*}$ and $S_{y^* y^*}$. Here, the expected-maximization (EM) algorithm is used to estimate the unknown elements of S and Ω. At the end of the E-step, the estimated sample covariance $\hat{S}$ is obtained, i.e.,

$$\hat{S} = \begin{pmatrix} \hat{S}_{y^* y^*} & \hat{S}_{y^* \mu^*} \\ \hat{S}_{\mu^* y^*} & S_{\mu^* \mu^*} \end{pmatrix},$$

and it will be the input of the M-step where GLasso solves the following minimization problem:

$$\widehat{\Omega} = \arg\min_{\Omega} -\mathcal{L}(\Omega; \hat{S}) + \lambda \|\Omega\|_1 .$$

The EM algorithm iteratively applies the following two steps:

E-Step: The E-step Calculates $Q(\Omega \mid \Omega^{(t)})$, the expected value of the penalized log-likelihood function w.r.t. the conditional distribution of y^* given μ^* under the current estimate $\Omega^{(t)}$ of Ω.

$$\begin{aligned} Q(\Omega \mid \Omega^{(t)}) &= \mathrm{E}_{y^*|\mu^*,\Omega^{(t)}} \left[\mathcal{L}(\Omega; S) + \lambda \|\Omega\|_1\right] \\ &= E_{y^*|\mu^*,\Omega^{(t)}} \{\log|\Omega| - trace(S\,\Omega) + \lambda \|\Omega\|_1\} \\ &= \log|\Omega| - trace\{E_{y^*|\mu^*,\Omega^{(t)}}(S)\,\Omega\} + \lambda \|\Omega\|_1 . \end{aligned}$$

The output of the E-step is $\hat{S} = E_{y^*|\mu^*,\Omega^{(t)}}(S)$ and can be used as an input for GLasso in M-step.

M-Step: This step finds the parameters that maximize $Q(\Omega \mid \Omega^{(t)})$ over all $(M+1) \times (M+1)$ positive-definite matrices Ω:

$$\Omega^{(t+1)} = \arg\max_{\Omega} Q(\Omega \mid \Omega^{(t)}).$$

This maximization problem is a GLasso problem and is equivalent to this minimization problem:

$$\hat{\Omega} = \arg\min_{\Omega} -\log|\Omega| + trace\{\widehat{S}\,\Omega\} + \lambda \|\Omega\|_1 , \tag{2.2}$$

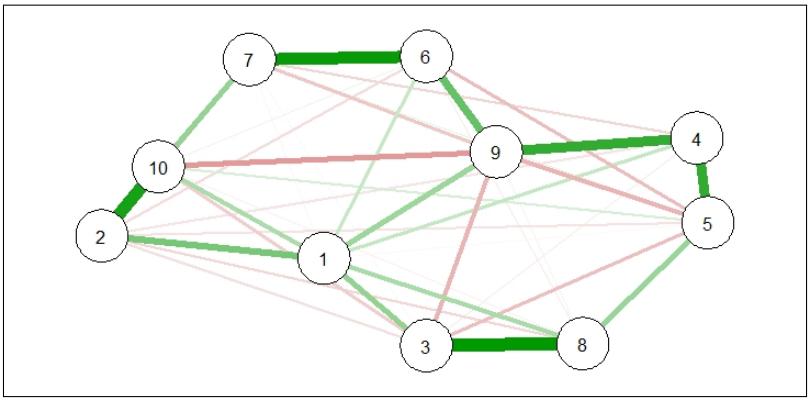
where $\Omega \in \mathcal{R}^{(M+1)\times(M+1)}$ and $\Omega > 0$. The authors in [72] indicate that the main difference between the optimization problem in (2.2) and GLasso problem is the coordinate corresponding to the latent variable should be left unpenalized. However, using a sparsity parameter for the latent variable can lead to an expert selection approach in the aggregation step.

The proposed EM algorithm reveals the covariance and precision matrix of the joint distribution of the full data (y^*, μ^*). The aggregated estimator can be defined according to the joint covariance and precision matrix. Assume the joint distribution of centered random variables (y^*, μ^*) is Gaussian with covariance matrix $\Sigma = (\Omega)^{-1}$, where Ω is the

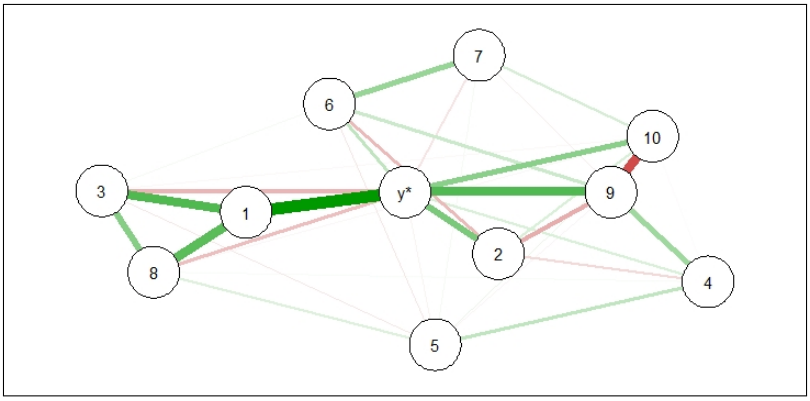
output of Equation ((2.2)). Then, the LVGGM-based aggregation for local predictions is given by:

$$y_A^* = \Sigma_{y^*\mu^*}^T (\Sigma_{\mu^*\mu^*})^{-1} \mu^*. \tag{2.3}$$

Figure 2.5 depicts the related graphs of ten Gaussian experts and a latent variable y^*. The sample is a synthetic data set containing 5000 observations from Equation (2.1) assigned to ten partitions. The graph in (a) is the standard GGM based on the precision matrix estimated by GLasso. The graph in (b) is a latent variable GGM where the sample covariance and precision matrix are calculated using the EM algorithm and GLasso. The interactions between y^* and the other nodes are used to define the aggregated predictor.



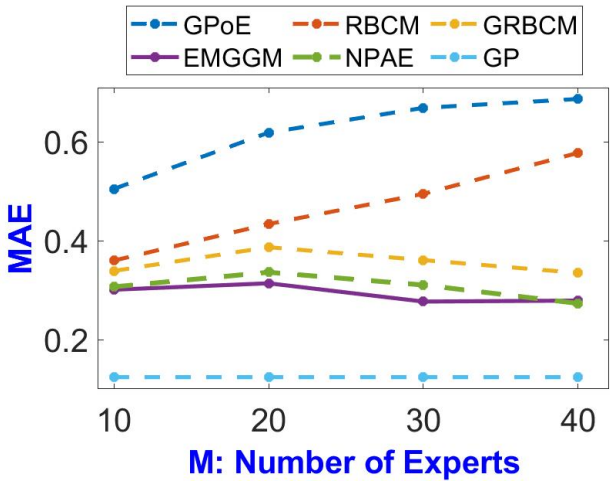
(a) GGM



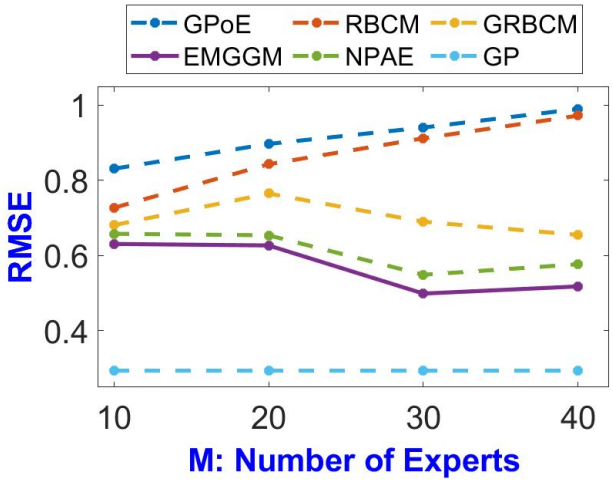
(b) LVGGM

Figure 2.5: **Ablation experiment** A Graph for a synthetic data set with 10 observed variables, a latent variable y^*, and sparsity parameter $\lambda = 0.01$. The green and red lines return positive and negative interaction based on the estimated precision matrix, respectively.

The conventional mean absolute error (MAE) and the root mean squared error (RMSE) are used to evaluate the accuracy of the aggregated estimator in Equation (2.3). A simulated data set is generated from the one-dimensional analytical function defined in Equation (2.1). It involves $n = 10^4$ training points in $[0,1]$, and $n_t = 10^3$ test points in $[-0.2, 1.2]$. To evaluate different partition sizes, we vary the number of experts, $M = \{10, 20, 30, 40\}$. The prediction quality of the proposed ensemble is compared with the other DGP baselines and the full GP.



(a) MAE



(b) RMSE

Figure 2.6: **Prediction quality** of DGP methods with respect for different number of experts M.

Figure 2.6 depicts the prediction error of EM-based aggregation (EMGGM) and other baselines compared to the standard Gaussian process. The ensemble methods that use experts' dependencies, i.e., EMGGM and NPAE, outperform the CI-based baselines. It results that employing the interactions between experts significantly raises the prediction quality. Indeed, the error values of the EMGGM method are lower than NPAE, which shows the ability and superiority of the LVGGM as an ensemble method.

2.2.2 Discussion and Future Works

The proposed ensemble is capable of aggregating local experts considering their dependencies. Unlike the NPAE method, which uses all training and test set information in the aggregation step, the EMGGM only uses local experts' predictions. The NPAE needs to store all auto-covariance and cross-covariance matrices between partitions, and therefore its storage cost is high. In contrast, the proposed method only requires a $n_t \times M$ matrix of local responses, where n_t is the size of the test set and M is the number of local experts.

Besides, the normality assumption in GGM for the joint distribution is not restrictive and can be relaxed. Due to the choice of the prior in GPs, the joint distribution of the local experts in DGP is Gaussian. However, we can relax this assumption and consider random variables without resorting to a multi-dimensional Gaussian distribution. The nonparanormal graphical model [68, 69, 70] and nonparametric functional graphical models [73, 74] are two primary types of nonparametric models that do not need normality assumption. It shows that the proposed strategy can provide an ensemble method for Gaussian and non-Gaussian experts.

The proposed method for latent variable GGM employs the EM algorithm to encode the interactions of the latent variables. An EM iteration does increase the likelihood function $\mathcal{L}(\Omega;S)$. However, no guarantee exists that the sequence converges to a maximum likelihood estimator. It is only guaranteed to converge to a point with a zero gradient concerning the parameters. So it can get stuck at saddle points, see [75]. The convergence property of the EM algorithm can be improved using a variety of heuristic or meta-heuristic approaches that enable EM to escape a local maximum, e.g., hill climbing and simulated annealing. The standard latent variable GGM (LVGGM) can solve the existing challenges in the convergence of EM. In the presence of latent vari-

ables, the non-linear optimization problem in Equation (1.22) can be solved by convex or non-convex optimization methods [65, 67].

2.3 Gaussian Process Experts Selection Using Gaussian Graphical Models

We discussed aggregating the Gaussian experts based on their dependencies in Definition 1. As we can see in Equation (1.17), the inverse of an $M \times M$ matrix has to be computed for each new test point, where M is the number of local experts. It makes aggregation inefficient for large data sets. Selecting the most relevant experts reduces the aggregation and storage costs. On the other hand, excluding the weak experts with lower-quality local predictions improves the accuracy of the final predictive distribution. In this section, we evaluate the importance of the experts via their related GGM.

2.3.1 Motivation and Main Methodology

To develop an expert selection strategy, we use the $n_t \times M$ local prediction matrix μ^*, which involves the local predictions of M experts at n_t test points. We still use the randomness assumption, and therefore the joint distribution of the local experts $\mu^* = (\mu_1^*, \ldots, \mu_M^*)$ is multivariate normal. Thus, they can be described through a GGM. The selection task is done based on their importance in the graphical model.

The selection task divides the full nodes in the graph into important and unimportant experts. The relevant precision matrix Ω obtained by the GLasso method is used to estimate the expert's importance. The elements of this matrix represent the interactions between the nodes. The nodes interacting more with the others are determined as important experts. Assume P is the number of important experts selected for final aggregation, where $P < M$. Thus, the complexity of the aggregation in Equation (1.17) is reduced from $\mathcal{O}(M^3)$ to $\mathcal{O}(P^3)$.

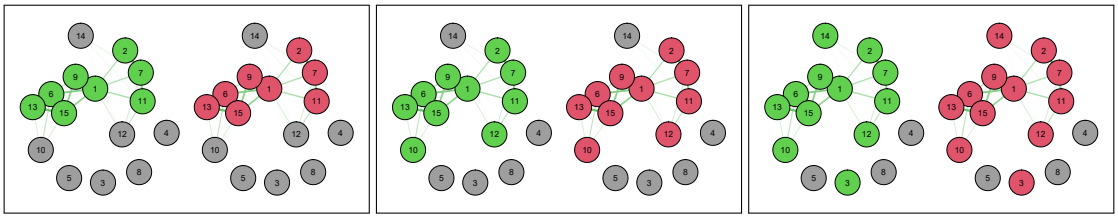
Expert Importance: The importance of expert i is defined as

$$\mathcal{I}_i = \sum_{j=1, j\neq i}^{M} |\widehat{\Omega}_{ij}|, \tag{2.4}$$

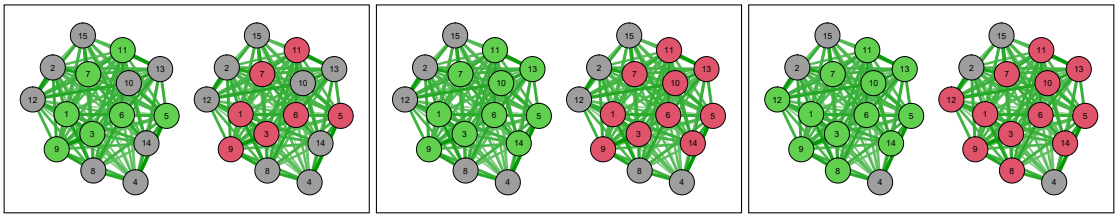
resulting in the sorted importance set $\mathcal{I} = \{\mathcal{I}_{i_1}, \mathcal{I}_{i_2}, \ldots, \mathcal{I}_{i_M}\}$. The $\widehat{\Omega}_{ij}$ is estimated by GLasso in Equation (1.22) and is affected by penalty term λ. The importance of the expert i $\mathcal{I}_i$ is the sum of all elements in row i of Ω, except the diagonal element. To

choose the important experts in the sorted importance set $\mathcal{I}$, we define a hyperparameter α, where $0 < \alpha \leq 1$, that indicates the percentage of the experts that should be selected for final aggregation, i.e., $P = \alpha \times M$. The α can be an almost large value, like 80%, in small data sets or a significantly small value, e.g., 50%, in large data sets.

Importance and Prediction Quality: Essentially, the importance measure $\mathcal{I}_i$ reflects the overall prediction quality of an expert. Let's consider the conventional mean square error (MSE) to evaluate the prediction quality of local experts. Then we compare α percent of best experts with smaller MSE with α percent of the most important experts. Our experiments confirm that both scenarios tend to select the same experts.



(a) Synthetic data set, $\alpha = 50\%$ (b) Synthetic data set, $\alpha = 70\%$ (c) Synthetic data set, $\alpha = 80\%$



(d) *Pumadyn* data set, $\alpha = 50\%$ (e) *Pumadyn* data set, $\alpha = 70\%$ (f) *Pumadyn* data set, $\alpha = 80\%$

Figure 2.7: **Ablation experiment.** Expert selection for synthetic data and a real-world dataset (*Pumadyn*) with $M = 15$, varying expert set strength $\alpha = \{50\%, 70\%, 80\%\}$, and $\lambda = 0.1$. The green nodes reveal the $\alpha\%$ of the best best experts w.r.t. their individual MSE errors while the red nodes are the most important experts according to Equation (2.4).

Figure 2.7 compares the selected experts based on the MSE criterion and importance measure defined in Equation (2.4) using synthetic and real-world data sets. For synthetic data, 5×10^3 training points and 500 test points are generated from (2.1). The real-world data set is *Pumadyn*, a 32D data set with 7,168 training points and 1,024 test points. The training points in both data sets are divided among 15 experts. We use the actual test

labels y^* to calculate the MSE between local predictions and actual values [1]. It results in the corresponding MSE values of our local experts, and the experts with a lower error are preferable. On the other hand, we estimate the importance values of all experts using Equation (2.4). Figure 2.7 depicts the results of the synthetic data set in (a), (b), and (c) and for *Pumadyn* data set in (d), (e), and (f). It confirms that the proposed selection method approach tends to choose the best experts as the important experts.

Aggregation Based on Important and Unimportant Experts: To show how important experts can provide better aggregation than unimportant experts, we consider again the Equation (1.17) that combines dependent experts:

$$y_A^*(x^*) = k_A(x^*)^T K_A(x^*)^{-1} \mu^*(x^*).$$

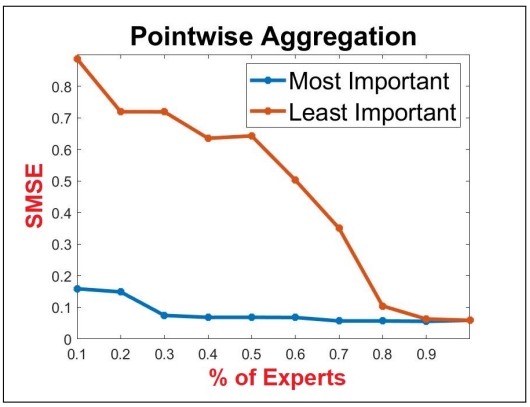
We first calculate the aggregated estimator y_A^* using $\alpha\%$ of important experts. Then we repeat the task with $\alpha\%$ of unimportant experts. In this way, we can investigate the influence of the importance measure in Equation (2.4) in the final aggregation.

Figure 2.8 presents the effect of important and unimportant experts on the prediction quality of the dependency-based aggregation defined in Equation (1.17). Here, 5×10^3 synthetic data points from (2.1) are used with varying expert set strength α and $\lambda = 0.1$. The quality of predictions is evaluated in two ways: the standardized mean squared error (SMSE) and the mean standardized log loss (MSLL). The blue line shows the prediction quality when $\alpha\%$ of the most important experts are chosen. The red line shows the prediction quality when the least important experts are chosen. We can see a significantly better aggregation quality based on the most important experts than the least important ones.

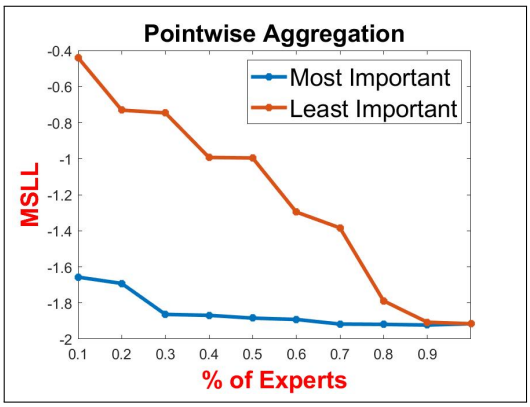
Figure 2.9 depicts the experiment in the CI-based aggregation method and confirms that the importance measure can also be used in this class of ensembles. It can be helpful when dealing with problems that do not allow experts to communicate, e.g., federated learning. In both figures, the case $\alpha = 1$ return the standard NPAE and GRBCM methods, respectively.

Expert's Weights vs Expert's Importance: Due to the property of the Gaussian processes, the prediction quality of GP at a test point far away from the training data set

[1] Here, we assume that the actual labels y^* are available, and then we can calculate the MSE values. It is an unrealistic assumption; in practice, we do not have the true labels.

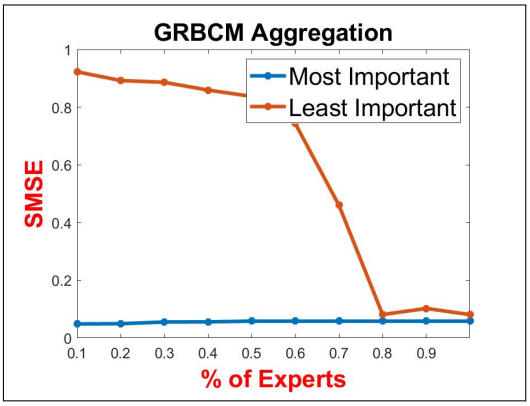


(a) SMSE: Dependency-based aggregation

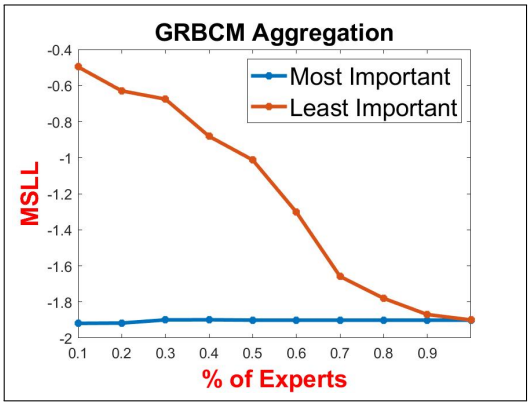


(b) MSLL: Dependency-based aggregation

Figure 2.8: Aggregating dependent important and unimportant experts.



(a) SMSE: CI-based aggregation



(b) MSLL: CI-based aggregation

Figure 2.9: Aggregation conditionally independent important and unimportant experts.

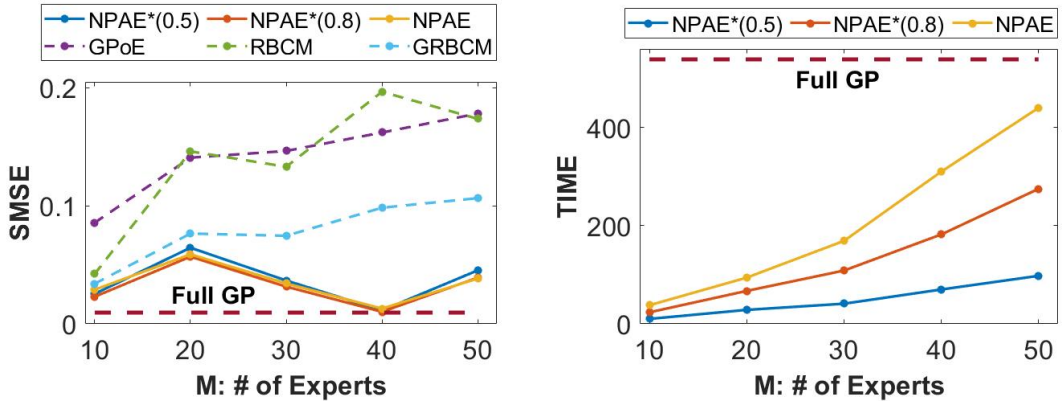
is less accurate than when it is closer. Therefore, for a test point, x^*, that is far from a partition $\mathcal{D}_i$, the prediction performance of the relative GP expert is generally poor. Besides, the local experts' covariance has been defined in Equation (1.5) as

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

Assume x^* is far away from the partition $\mathcal{D}_i$, then $k(x^*, X_i) \approx 0$, and $\Sigma_i^* \approx k_{**}$. Suppose the experts' weights are defined as the difference in differential entropy [37], i.e., $\beta_i = \frac{1}{2}(\log k_{**} - \log \Sigma_i^*)$. Then $\beta_i \approx 0$. Thus the distributed GPs tend to assign a small

weight to $\mathcal{D}_i$. On the other hand, $\mathcal{D}_i$ has weak dependencies with the other experts in Equation (1.16) because $k_{i*} \approx 0$, and therefore, $\Gamma_i \approx 0$. Hence, the importance measure recognizes this expert as an irrelevant or unimportant expert. Consequently, expert selection eliminates this expert, while distributed GPs tend to keep this expert and assign a small weight to $\mathcal{D}_i$.

Prediction Quality of Expert Selection Strategy: Essentially, the expert selection strategy provides an approximation for the consistent estimator in Equation (1.17). The selection task keeps all related asymptotic properties of the original estimator and reduces the computational costs of using all initial experts in the aggregation. Indeed, excluding the weak experts removes their adverse effects on the final ensemble. Asymptotically, the predictions provided by the selection method deliver results that are consistent to the dependency-based aggregation when $n \to \infty$.



(a) SMSE

(b) Time

Figure 2.10: Prediction error and running time as a function of experts. SMSE and prediction time of NPAE with 50%, 80%, and 100% of experts , CI-based baselines, and full GP for different partition sizes M.

Figure 2.10 depicts the SMSE values and running times of GPoE, RBCM, GRBCM methods and dependency-based aggregation with $\alpha = \{0.5, 0.8, 1\}$ for partition sizes $M = \{10, 20, 30, 40, 50\}$. As we can see here, the approximations with 50% and 80% of most relevant experts provide competitive prediction quality compared to NPAE in just a fraction of NPAE's running time. It confirms the influence of the importance measure in increasing the efficiency and accuracy of the dependency-based ensemble method. In-

deed, the selection strategy also has a similar effect in CI-based baselines by excluding the weak experts, while the cost of modified and original baselines is the same, see [4, 8].

Penalty Parameter: The importance measure in Equation (2.4) uses the precision matrix Ω. This matrix depends on the selected value for λ that determines the network sparsity. We experiment to evaluate the effect of different sparsity values on prediction quality. The NPAE method with $\alpha = 0.5$ and $\alpha = 0.8$ is used for 5×10^3 observations of the synthetic data set in (2.1) and $M = 20$.

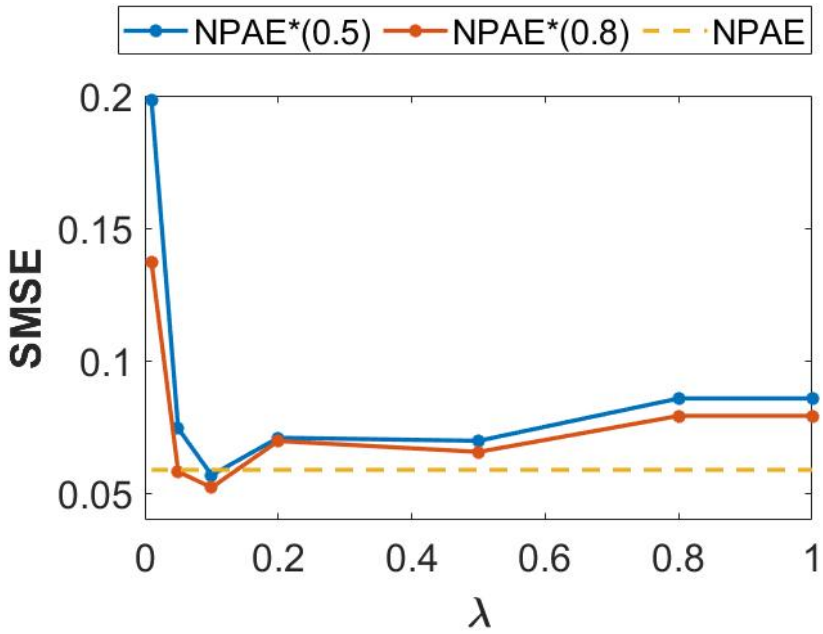


Figure 2.11: SMSE of NPAE method with $\alpha = \{0.5, 0.8\}$ for different λ values.

Figure 2.11 presents the varying sparsity parameter and depicts the prediction quality for different λ values. As the figure shows, small and large values (i.e., smaller than 0.05 or larger than 0.5) lead to slightly poor results, but for values between 0.05 and 0.5, the network shows stable results. The graph is dense with more edges for the small λ value, while large λ leads to a very sparse network with only nodes and very few edges. The figure demonstrates that $\lambda \approx 0.1$ is an appropriate choice for the expert selection step.

2.3.2 Discussion and Future Works

The Gaussian graphical model can provide a powerful method to represent the dependencies between experts in local approximation by capturing the interactions between local GP experts. The method does not use all local data points, and the inference is made by using only the provided local predictions. The interactions between experts in the related undirected graph are encoded in terms of similarities and differences between experts. The experts that return similar predictions are typically strongly dependent on each other and significantly influence the aggregated estimator defined in Equation (1.17). On the other hand, when the individual predictions of an expert are far away from all other experts, the related node in the graph does not have significant interaction with the other nodes, and it is not an influential variable in the final aggregation.

The expert selection step significantly reduces the prediction cost of the dependency-based aggregation method in Equation (1.17). The prediction cost of the original model is approximately equal to $\mathcal{O}(n_t M^3)$, where n_t is the number of available test points and M is the number of experts. GLasso also has cubic time complexity $\mathcal{O}(M^3)$. Therefore, the complexity of the expert selection-based aggregation is $\mathcal{O}(n_t M_\alpha^3 + M^3) = \mathcal{O}(n_t(\alpha M)^3)$. It means the cost of GLasso can be ignored when n_t is large. However, there are newer, faster methods to learn a GGM that reduce the computational complexity of sparse Gaussian Graphical Models to a much lower order of magnitude ($\mathcal{O}(M^2)$), see [63, 64]. Since $\alpha < 1$, the computational cost is significantly reduced when only $\alpha\%$ of the most relevant experts are used in the final ensemble. In CI-based baselines, the complexities of the original and modified versions are of the same rate, especially when the number of experts is large.

The most important experts selected for the aggregation step form a static set. The selection task assigns a fixed group of local experts to all test points. It is not a crucial problem in smooth data sets. However, this strategy is not appropriate to explain the data behavior in complex and quickly varying data sets. Assume there are new test points far from the selected experts and close to the excluded ones. In this case, all selected experts provide poor results for new entry points, affecting the final aggregation's prediction quality. The following subsection will discuss a flexible and entry-dependent selection approach that captures the properties of the new data entries.

2.4 Gaussian Process Experts Selection Using Multi-label Classification

Despite the advantages of the expert selection approach, the proposed method using GGM cannot capture the complex structure of quickly varying data sets. The previously presented GGM approach returns a static group of experts, which are fixed for prediction for every new entry point. We propose an entry-dependent selection approach that converts the selection task into a multi-label classification problem to solve this problem.

2.4.1 Motivation and Main Methodology

Assume that the training data set $\mathcal{D}$ has been divided into M partitions $\{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ and and $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ is the Gaussian experts set. The task is to assign only some relevant experts to each new entry point x^*. Let K indicates the number of the assigned experts, and $\mathcal{M}^{\mathcal{C}}(x^*) = \{\mathcal{M}^{\mathcal{C}}{}_1(x^*), \ldots, \mathcal{M}^{\mathcal{C}}{}_K(x^*)\}$ represents K selected experts to predict at x^*. In this scenario, $\mathcal{M}^{\mathcal{C}}(x^*)\}$ is selected using multi-label classification. Unlike the static expert selection by GGM, a dynamic and flexible mechanism designates related experts for each new observation.

Multi-label classification [76] 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 convert the expert selection task into multi-label classification, we define the indices of the partitions as labels. Although a multi-class classification problem would select an appropriate expert for predicting, it leads to a local approximation with only one expert per test point, which would produce discontinuous separation boundaries between sub-regions and would not be appropriate for quantifying uncertainties [25, 77].

Assume x^* is a new test point and $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ is the Gaussian experts set, and $\mathcal{L} = \{1, \ldots, M\}$ is the label set. The task is to find $\mathcal{M}^{\mathcal{C}}(x^*) = \{\mathcal{M}^{\mathcal{C}}{}_1(x^*), \ldots, \mathcal{M}^{\mathcal{C}}{}_K(x^*)\}$ that represents K selected experts to predict at x^*. Let $\mathcal{M}^{\mathcal{C}}$, and $\mathcal{M}^{\mathcal{G}}$ be the assigned experts using classification and GGM, respectively.

To compare the scheme of static (GGM-based) and entry-based (classification-based)

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

Figure 2.12: **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 to each test points: (a) original set of experts $\mathcal{M}$, (b) static assignment of experts $\mathcal{M}^{\mathcal{G}}$, and (c) entry-based selection of experts $\mathcal{M}^{\mathcal{C}}$.

selection strategies, let's consider both models in a setting with synthetic data points with five experts $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_5\}$ and their predictions at 10 test points. The task is to assign $K = 3$ experts to the test points. Figure 2.12 describes the difference between static and dynamic selection models. Figure 2.12 (a), (b), and (c) depict all initial experts $\mathcal{M}$, GGM-based static model $\mathcal{M}^{\mathcal{G}}$, and entry-based dynamic model $\mathcal{M}^{\mathcal{C}}$, respectively. As the scheme in (b) shows, the static model proposes a fixed set of 3 experts $\{\mathcal{M}_2, \mathcal{M}_4, \mathcal{M}_5\}$ for all new entry points even though they do not provide appropriate predictions in some of this 10 test points. On the other hand, the dynamic model in (c) assigns only the most relevant experts to each test point x^* and uses the ability of experts in a better way.

To solve this multi-label classification, we consider two prominent classification models, K-nearest neighbors (KNN) and conventional deep neural networks (DNN). In addition to the moments of local Gaussian experts defined in Equations (1.4) and (1.5), the classification task needs other information that describes the labels related to each partition. When each partition determines a label, there must be an explicit mechanism to represent the labels for the selection model. However, the details on label representation differ for our classification methods.

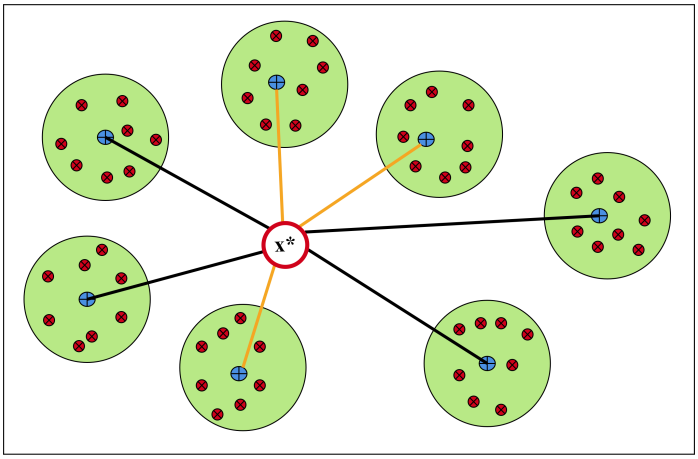
In KNN, the partitions can be explained by the centroids of the clusters. These centroids $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_M\}$ are the output of the partitioning step and contain all required

details about the clusters. We calculate the distances between the centroids and all new entries. 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. Therefore, by estimating the distance between a new entry point and the centroids, we can find its K nearest neighboring experts. The experts that are further away generally can not provide appropriate predictions at this point and can be ignored. However, unlike the static model, they are not excluded from the model because they may be pretty accurate for some new data points.

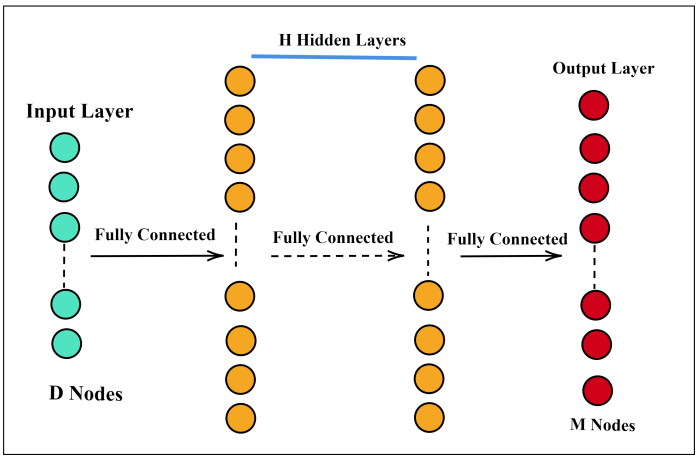
In DNN, instead of the centroids, the related labels of the training data set are used for assignment. If each partition represents a label, we can define the partition index set $\mathcal{L} = \{1, \ldots, M\}$ as the label set. Thus, indices can be attached to the training points and create a new pair (x_i, j) where $x_i \in \mathcal{D}_j$. In the next step, a multi-label classification task is performed to estimate the related label of the new test points. To this end, we train a neural network with a soft-max output layer and log-loss (i.e., cross-entropy loss) that uses the training points and their relevant partition indices and learns to which partitions the data points belong. For a test point x^*, the output layer in a DNN returns the related probabilities for each partition $(p_1(x^*), \ldots, p_M(x^*))$, where $p_j(x^*) = prob(x^* \in \mathcal{D}_j)$. The K experts that have higher probabilities are assigned to x^*.

The value of K defines how many clusters will be assigned to a specific entry point. For example, in KNN, if $K = 1$, the instance will be assigned to the same class as its nearest cluster, which is not the desired case due to discontinuity issues. Lower values of K can have high variance, but low bias and larger values of K may lead to higher bias and lower variance [78]. Like the static selection scenario, the dynamic selection method 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.

Figure 2.13 schematically shows the classification methods in this case. It represents a KNN framework for a test point x^* in (a). 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 that are the shortest. The second solution has been depicted in (b)—the conventional deep neural network with an input layer, H hidden layers, and an output



(a) KNN



(b) DNN

Figure 2.13: Conventional KNN and deep neural network for multi-label classification.

layer. Since the new entry point is a D-dimensional variable, $x^* \in \mathcal{R}^D$, the input layer has D nodes. The output layer contains M nodes because the label set has M labels. For each node in the output layer, the model estimates a value that describes the probability of the input belonging to the corresponding expert. The experts with the highest probability values are assigned to x^*.

To compare the prediction quality of different expert selection approaches, we use a real-world data set, *Concrete* [2]. It is a small data set containing 1030 observations (90% for training and the rest for testing) with nine attributes (8 independent variables and one

[2] `https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength`

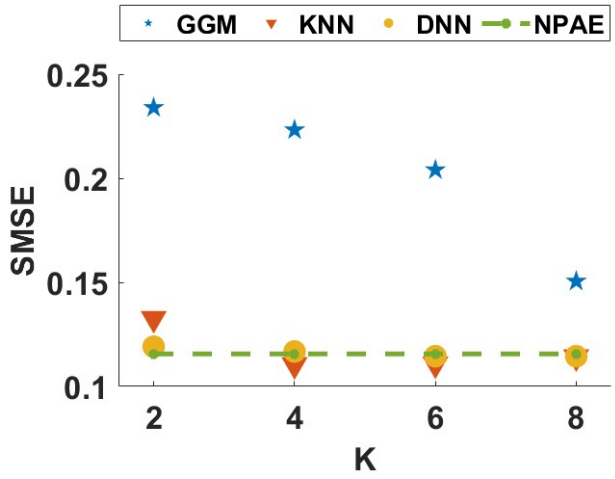


Figure 2.14: **Expert Selection** prediction qualities of different experts selection methods compared to original baseline, NPAE, from *Concrete* data set.

response variable). The disjoint partitioning is used to divide the data set into ten subsets. The prediction qualities of the static selection method (GGM-based), dynamic methods (KNN and DNN-based), and the original NPAE are compared using their SMSE values. The number of the selected experts K changes from 2 to 8 to consider the aggregation quality with the different numbers of partitions.

Figure 2.14 depicts the prediction quality of expert selection methods on the *Concrete* data set for different values of K. The case $K = 10$ refers to the NPAE method depicted with a green dashed line. We can see that the multi-label-based expert selection models provide higher quality predictions with lower deviation from the NPAE model, and both KNN and DNN return proper predictions with lower error values. Unlike the GGM-based aggregation, the quality of the aggregations that use KNN and DNN is not significantly affected by the values of K. However, the quality of the GGM-based ensemble improves by increasing the K. When the number of assigned experts rises, the quality of the dynamic methods is slightly better than that of the original baselines, which means that by excluding the weak experts at each test point, the final aggregation can provide better prediction using only the relevant experts.

We can also investigate the sensitivity of the selection model when the number of partitions, M, increases. Figure 2.15 explains the results using synthetic data points. It uses 3×10^3 training points from Equation (2.1) and divides them between $M = \{10, 15, 20\}$.

For the selection methods, 3 and 5 experts are selected for the final aggregation. As the figure shows, multi-label classification methods lead to better predictions and can also outperform the original baseline. When we increase M from 10 to 20, the partitions contain a smaller amount of training data, and therefore some weak experts can not provide a reliable estimation at a new test point. However, the quality of the static method is way more sensitive concerning the changes in M.

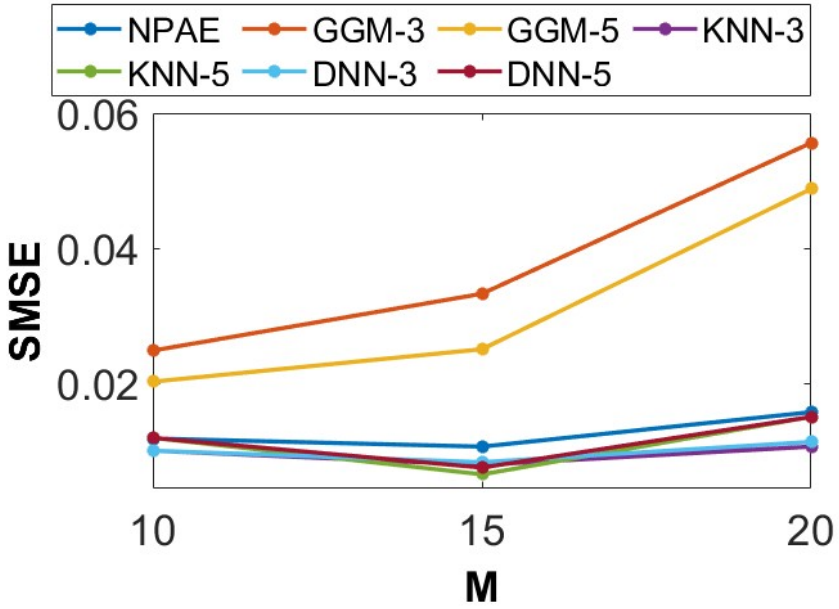
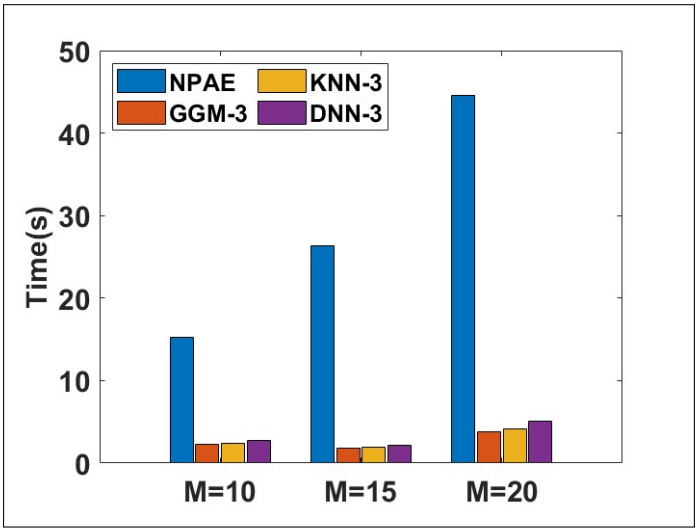
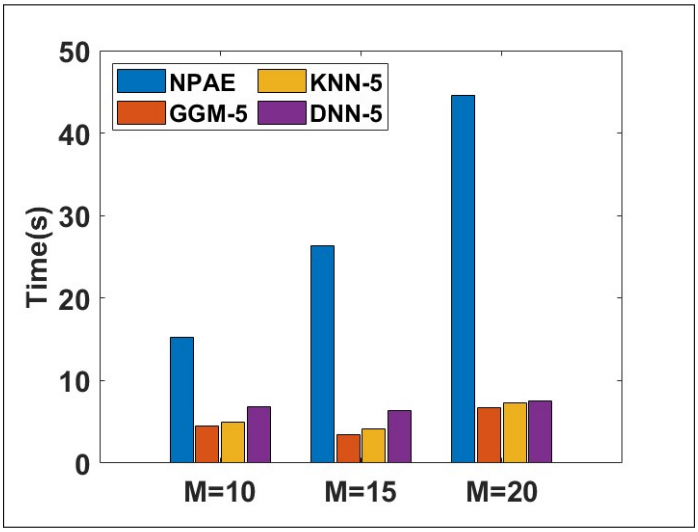


Figure 2.15: **Prediction Qualities** of different expert selection methods for different numbers of partitions using a synthetic data set.

The results can be explained by defining a new parameter: the relative number of selected experts to the total number of experts $K_M = \frac{K}{M}$. This ratio indicates the percentage of the initial experts selected to be used in the final predictive distribution. When K_M decreases, the prediction quality of GGM-based models decreases drastically. For instance, in Figure 2.15, the SMSE values of *GGM-3* and *GGM-5* at $M = 20$ are almost twice the SMSE at $M = 15$. The related K_M value with five selected experts ($K = 5$) for $M = 15$ is $\frac{1}{3}$ while for $M = 20$ is $\frac{1}{4}$. This means that the GGM method requires more experts to provide qualitative predictions, and the difference between the SMSE of *GGM-3* and *GGM-5* indicates this fact.



(a) Aggregation with 30% of Experts



(b) Aggregation with 50% of Experts

Figure 2.16: **Prediction Time** (seconds) of different aggregation in the synthetic data set.

It also can be seen when K changes. For instance, in $M = 15$, the difference between *GGM-3* and *GGM-5* is remarkable, and GGM with five experts significantly outperforms GGM with three experts. Figure 2.15 shows that GGM can not provide competitive prediction quality compared to the other selection methods. For instance, the SMSE values of *DNN-3* for all sizes of M are lower than those of *GGM-3* and *GGM-5*. This issue confirms the low convergence rate of the GGM, which requires more experts. Besides, the quality of KNN and DNN does not change when K increases from 3 to 5 or when

M changes from 10 to 20, which shows that they are not sensitive concerning K_M. The figure confirms that the convergence in classification-based expert selection methods occurs with smaller number of experts according to the difference between the SMSE lines of NPAE and the other lines.

Figure 2.16 depicts the prediction time of static and dynamic selection methods. The running times of the related baselines are compared in two cases, with 30% and 50% of original experts. In both cases, the NPAE method is a baseline that uses all experts. Based on the figure, the prediction times of GGM, KNN, and DNN are almost similar. However, it shows that static and dynamic selection strategies remarkably reduce the running time.

2.4.2 Discussion and Future Works

Multi-label classification problems can be used as an expert assignment mechanism. In this section, we proposed a novel method to select the most relevant experts for each test point. It determines the related experts by assigning a set of partitions' indices to new data points. This entry-dependent selection strategy does not have the drawbacks of the static expert selection method. The flexibility of this method leads to a significant improvement in prediction quality while computational costs of the proposed and GGM method are of the same rate. However, some related aspects of this method are discussed here.

Restrictive Assumptions: Unlike the GGM approach, expert selection using multi-label classification does not need any distributional assumption. The GGM approach assumes that the nodes in the graph are random and their joint distribution is Gaussian. Although, due to the choice of the prior in GPs, the normality assumption exists, it is not required in the dynamic selection approach. 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 GPs. Indeed, the entry-dependent 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 [79, 80]. The proposed ensembles (KNN and DNN) do not directly use the original training points. However, they benefit from

the critical information of training data captured by the partitions' centroids and the corresponding indices for KNN and DNN, respectively.

Activation Functions in DNN: In the classification model, two main activation functions are used to calculate the final probability values, *softmax* and *sigmoid*. The *sigmoid* looks at each raw output value separately; therefore, the produced probabilities are not constrained to sum up to one (labels are assumed to be independent). In contrast, the outputs of a *softmax* are all interrelated, and the probabilities produced by a softmax will always sum up to one by design. Hence, by raising the probability of one class, 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.

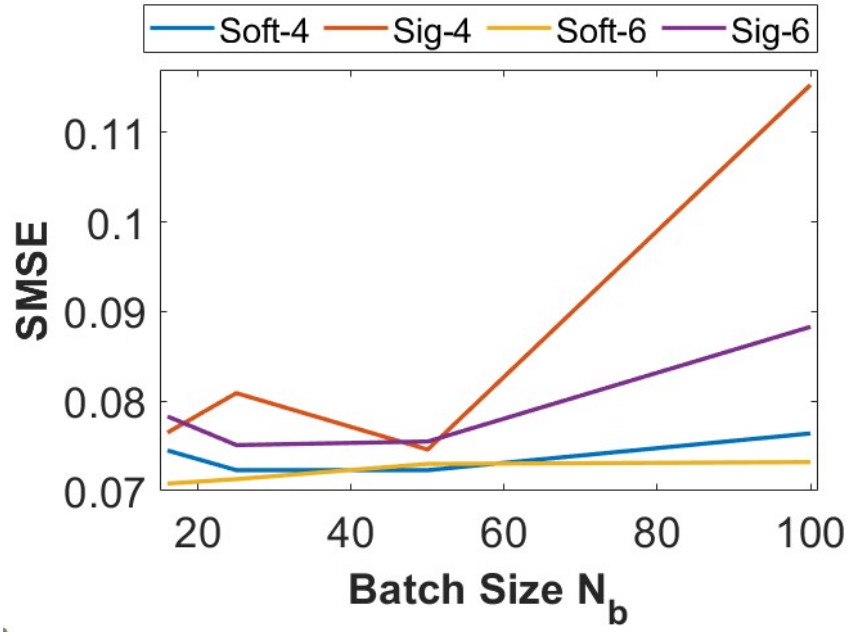


Figure 2.17: **Activation Functions** for the final (i.e., output) layer of the DNN classifier: prediction quality for $K = 4$ and $K = 6$ with 10 experts.

Figure 2.17 displays the prediction error of a DNN model with the *softmax* and *sigmoid* function in *Concrete* data set. We have used $M = 10$ experts 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 figure confirms that the *softmax* ac-

tivation leads to much better results due to the interaction between labels. Indeed, the sensitivity to the size of the mini-batches in the *softmax* activation function is lower than the sensitivity of the *sigmoid* activation.

Computational Costs and M: Using the selected experts (K experts) in GGM, KNN, and DNN leads to the aggregation cost $\mathcal{O}(n_t K^3)$ where K is the number of selected experts, and n_t is the number of test observations. Indeed, all three methods have a selection cost. The selection task in GGM needs GLasso to estimate the precision matrix, and its computational cost is $\mathcal{O}(M^3)$, which is challenging for a large number of initial experts M. Indeed, using a small sparsity parameter λ leads to a dense graph with a more considerable computational cost.

On the other hand, KNN and DNN methods have linear complexity concerning M. The cost of the KNN approach 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. KNN computes the distance between the new observation and each centroid point, which requires $\mathcal{O}(KMD)$ operations to select K closest centroids.

In DNN, the cost depends on the network parameters, 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 $\mathcal{O}\left(n(U_1U_2 + \ldots + U_{L-1}U_L)\right) = \mathcal{O}\left(n(DU_2 + \ldots + U_{L-1}M\right)$, which shows that the complexity linearly depends on the number of experts M.

Expert Selection for CI-Based Baselines: The benefits of the dynamic selection method for correlated experts have been discussed in this section. However, the proposed method can be easily extended to CI-based baselines. In this case, only K experts selected by KNN or DNN methods are used for final aggregation in Equation (1.6). Since these models are fast, the selection parameter K can also be set to relatively large values. It indicates that the proposed method can be used in federated learning models where the experts do not share any information and are independent.

Let us consider the *Concrete* data set again. Table 2.1 describes the effect of the selection scenario on CI-based ensembles using KNN with $M = 10$ and $K = 6$. Although this modification can not improve the asymptotic properties of the baselines, it raises their

Table 2.1: Expert Selection in CI-Based Baselines.

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

prediction quality. The selection method significantly enhances the prediction quality of GPoE, RBCM, and GRBCM methods. At the same time, the last column in the table shows that the running times of both original and modified models are indistinguishable.

Chapter 3

Outlook and Conclusion

In this thesis, we studied some aspects of decentralized learning when our local agents are Gaussian processes. In Gaussian process regression, the training data set is divided into several partitions to reduce the training cost. At each subset of the data, a local Gaussian process is trained, a so-called expert. The experts produce their predictions at each test point, and then an ensemble method should aggregate them. Assuming dependency between experts leads to high prediction and storage costs, while the assumption of conditional independence between experts (CI assumption) returns sub-optimal solutions with lower-quality predictions.

We considered different aspects of distributed Gaussian processes in this work. First, we convert the model into a two-layer computational graph to reduce the effect of perfect diversity in CI-based ensembles. The conventional one-layer model includes the local independent experts as leaves and the target variable as the parent. In the proposed method, presented in Section 2.1, the new experts in the middle layer of the graph are clusters of the highly correlated original experts. The Gaussian graphical model detects dependency, and spectral clustering is used to construct the new experts. Although the independence assumption is unrealistic and often violated in practice, the new experts are conditionally independent by design, and this solution significantly improves the prediction quality of the CI-based aggregations.

The Gaussian graphical model (GGM) detects dependency between random variables based on the assumption that the variables' joint distribution is Gaussian. The latent variable Gaussian graphical model (LVGGM) is a specific form of the GGM that considers latent and unobserved variables and encodes the relations between the variables.

Since the related precision matrix is the sum of two matrices, the relevant optimization problem is misspecified. We used the classical Expected-Maximization (EM) algorithm to estimate the precision matrix. We showed that by considering the target variable as latent and the local experts as observed experts, the LVGGM could be used as an ensemble method to aggregate the local predictions considering the dependencies between experts. This ensemble method returns the mean of the conditional distribution of the latent variable given the observed variables. The optimization employs the current precision matrix estimate and updates the conditional distribution's mean.

Both dependency-based and CI-based ensembles can benefit from an expert selection model in two ways: (1) through the selection of a small subset of local experts, which lowers the computational costs that depend on the number of experts; (2) through the exclusion of weak or irrelevant experts in the prediction step, which improves the prediction quality. When an expert produces low-quality predictions, the precision of the aggregation method is negatively affected. Thus, the expert selection task can improve the ensemble's efficiency and prediction quality by using only the most important experts. In this thesis, we proposed two different expert selection methods: general and entry-dependent selection.

The general expert selection strategy provides a static set of experts for all new entry points. In this work, we used the Gaussian graphical model to divide the initial experts into essential and unimportant experts. The importance measure is defined using the interaction between experts in the related undirected graph. The experts with high interactions constitute the selected experts. Using theoretical and experimental analysis, we showed that the important experts are almost identical to those with the highest local prediction quality. The proposed technique can improve the ensemble's performance and provide an insightful visual perspective on the experts' correlations, especially when the number of experts is not too large. Indeed, this strategy only uses the local experts' predictions and does not need the partitions' complete information. Thus, it can also be used for federated learning problems when we cannot share the partitions' details due to security requirements. Moreover, it keeps the asymptotic properties of the aggregation methods for both dependency and CI-based ensembles.

Despite the advantages and strengths of the general selection approach, it produces a static and fixed set of selected experts for all test points. Therefore, the model might not

be flexible enough to capture potentially unusual behavior of new data entries. When new observations are far away from the selected data partitions, the related experts provide poor local predictions for them, and, therefore, the final aggregation cannot return precise results. We converted the expert selection task to a multi-label classification problem to address this issue. To this end, we used the related indices of the experts as labels; the extended data set can then be used to train classification techniques on the available data points and the related indices. In this context, a sensible expert selection strategy should aim to find a set of the most relevant labels for each new entry point. The algorithms we devised find the most appropriate experts for each new entry point, and the final prediction is based on the selected experts. We developed two classification methods, one based on k-nearest neighbors (KNN) and another based on deep neural networks (DNN). We showed that this point-wise expert selection scenario significantly improves the prediction quality of the aggregation while its running time is at the same rate as the static expert selection approach.

This thesis focused on the dependency between experts in a distributed Gaussian process framework. However, the results can be easily extended to general decentralized learning and multi-agent decision-making problems. The normality assumption on the joint distribution of local experts is not a severe restriction. Moreover, this assumption can be relaxed so that the proposed solutions can work with non-Gaussian experts. Indeed, we introduced a novel solution for CI-based aggregation methods. The CI assumption is used in both distributed and federated learning problems. In federated learning, the experts do not share any information and are entirely independent, according to security issues. The aggregation step in this problem assumes the agents are independent. As discussed before, our solutions can be used by independent experts. Thus, the main ideas behind this thesis's aggregation and selection approaches can also be applied to federated learning scenarios. Adopting the proposed approaches to federated learning and multi-agent decision-making problems can be studied in future works.

Bibliography

[1] M. P. Deisenroth and J. W. Ng, "Distributed gaussian processes," *International Conference on Machine Learning*, pp. 1481–1490, 2015.

[2] 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.

[3] H. Jalali and G. Kasneci, "Aggregating dependent gaussian experts in local approximation," *25th International Conference on Pattern Recognition (ICPR)*, pp. 9015–9022, 2021.

[4] 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.

[5] H. Jalali and G. Kasneci, "Gaussian graphical models as an ensemble method for distributed gaussian processes," *NeurIPS 2021 Annual Workshop on Optimization for Machine Learning (OPT2021)*, 2021, doi: https://doi.org/10.48550/arXiv.2202.03287.

[6] ——, "Aggregating the gaussian experts' predictions via undirected graphical models," *IEEE International Conference on Big Data and Smart Computing (BigComp)*, pp. 23–26, 2022.

[7] ——, "Expert selection in distributed gaussian processes: A multi-label classification approach," *NeurIPS Workshop on Gaussian Processes, Spatiotemporal Modeling, and Decision-making Systems*, 2022, doi: .

[8] H. Jalali, M. Pawelczyk, and G. Kasneci, "Gaussian experts selection using graphical models," *arXiv preprint arXiv:2102.01496*, 2021.

[9] V. Borisov, J. Meier, J. Heuvel, H. Jalali, and G. Kasneci, "Ga robust unsupervised ensemble of feature-based explanations using restricted boltzmann machines," *NeurIPS 2021 Workshop on eXplainable AI approaches for debugging and diagnosis*, 2021, doi: https://doi.org/10.48550/arXiv.2111.07379.

[10] C. E. Rasmussen and C. K. Williams, *Gaussian processes for machine learning*. MIT Press, 2006.

[11] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. D. Freitas, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[12] N. Lawrence, "Taking the human out of the loop: A review of bayesian optimization," *Journal of Machine Learning Research*, vol. 6, pp. 1783–1816, 2005.

[13] A. Zeng, H. Ho, and Y. Yu, "Prediction of building electricity usage using gaussian process regression," *Journal of Building Engineering*, vol. 28, 2020.

[14] 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.

[15] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408—-423, 2013.

[16] I. Bogunovic, A. Krause, and J. Scarlett, "Corruption-tolerant gaussian process bandit optimization," vol. 108, 2020, pp. 1071–1081. [Online]. Available: https://proceedings.mlr.press/v108/bogunovic20a.html

[17] M. A. Alvarez, L. Rosasco, and N. D. Lawrence, "Kernels for vector-valued functions: A review," *Foundations and Trends in Machine Learning*, vol. 4, no. 3, pp. 195–266, 2012.

[18] O. Hamelijnck, T. Damoulas, K. Wang, and M. Girolami, "Multi-resolution multi-task gaussian processes," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[19] M. F. Huber, "Recursive gaussian process: On-line regression and learning," *Pattern Recognition Letters*, vol. 45, pp. 85–91, 2014.

[20] 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.

[21] D. Petelin and J. Kocijan, "Evolving gaussian process models for predicting chaotic time-series," *IEEE International Conference on Evolving and Adaptive Intelligent Systems*, pp. 1–8, 2014.

[22] 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.

[23] D. Duvenaud, "Automatic model construction with gaussian processes," Ph.D. dissertation, University of Cambridge, 2014.

[24] H. Liu, J. Cai, Y. Ong, and Y. Wang, "Understanding and comparing scalable gaussian process regression for big data," *arXiv preprint arXiv:1811.01159*, 2018.

[25] H. Liu, Y. Ong, X. Shen, and J. Cai, "When gaussian process meets big data: A review of scalable gps," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 4405–4423, 2020.

[26] K. Chalupka, C. K. Williams, and I. Murray, "A framework for evaluating approximation methods for gaussian process regression," *Journal of Machine Learning Research*, vol. 14, pp. 333–350, 2013.

[27] K. Hayashi, M. Imaizumi, and Y. Yoshida, "On random subsampling of gaussian process regression: A graphon-based analysis," vol. 108, 2020, pp. 2055–2065. [Online]. Available: https://proceedings.mlr.press/v108/hayashi20a.html

[28] T. Gneiting, "Compactly supported correlation functions," *Journal of Multivariate Analysis*, vol. 83, no. 2, pp. 493–504, 2002.

[29] A. Melkumyan and F. Ramos, "A sparse covariance function for exact gaussian process inference in large datasets," *International Joint Conferences on Artificial Intelligence*, vol. 9, pp. 1936–1942, 2009.

[30] M. K. Titsias, "Variational learning of inducing variables in sparse gaussian processes," *Artificial Intelligence and Statistics*, pp. 567–574, 2009.

[31] J. Hensman, N. Fusi, and N. D. Lawrence, "Gaussian processes for big data," *Uncertainty in Artificial Intelligence, AUAI Pres*, pp. 282–290, 2013.

[32] G. Wynne and V. Wild, "Variational gaussian processes: A functional analysis view," vol. 151, 2022, pp. 4955–4971. [Online]. Available: https://proceedings.mlr.press/v151/wynne22a.html

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

[34] D. Moore and S. J. Russell, "Gaussian process random fields," *In Advances in Neural Information Processing Systems*, pp. 3357—-3365, 2015.

[35] E. Snelson and Z. Ghahramani, "Local and global sparse gaussian process approximations," *In Proceedings of Artificial Intelligence and Statistics (AISTATS)*, pp. 524–531, 2007.

[36] R. Urtasun and T. Darrell, "Sparse probabilistic regression for activity-independent human pose inference," *IEEE Conference on Computer Vision and Pattern Recognition (CPVR)*, pp. 1–8, 2008.

[37] Y. Cao and D. J. Fleet, "Generalized product of experts for automatic and principled fusion of gaussian process predictions," *arXiv preprint arXiv:1410.7827*, 2014.

[38] 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.

[39] V. Tresp, "Mixtures of gaussian processes," *Advances in Neural Information Processing Systems*, pp. 654–660, 2001.

[40] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: A literature survey," *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.

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

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

[43] Y. K. Samo and S. J. Roberts, "String and membrane gaussian processes," *Journal of Machine Learning Research*, vol. 17, no. 1, p. 4485–4571, 2016.

[44] D. Rullière, N. Durrande, F. Bachoc, and C. Chevalier, "Nested kriging predictions for datasets with a large number of observations," *Statistics and Computing*, vol. 28, no. 4, pp. 849–867, 2018.

[45] B. Szabó and H. van Zanten, "An asymptotic analysis of distributed nonparametric methods," *Journal of Machine Learning Research*, vol. 20, no. 87, pp. 1—-30, 2019.

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

[47] C. Zhang and Y. Ma, *Ensemble machine learning: methods and applications*. Springer, 2012.

[48] 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.

[49] 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. 1253–1258, 2014.

[50] P. Donmez, G. Lebanon, and K. Balasubramanian, "Unsupervised supervised learning i: Estimating classification and regression errors with-out labels," *Journal of Machine Learning Research*, vol. 11, p. 1323–1351, 2010.

[51] E. Platanios, A. Blum, and T. Mitchell, "Estimating accuracy from unlabeled data," *In Uncertainty in Artificial Intelligence*, 2014.

[52] H. Rue and L. Held, *Gaussian Markov random fields: theory and applications*. CRC Press, 2005.

[53] C. Uhler, "Gaussian graphical models: an algebraic and geometric perspective," *arXiv preprint arXiv:1707.04345*, 2017.

[54] M. Drton and M. Maathuis, "Structure learning in graphical modeling," *Annual Review of Statistics and Its Application*, vol. 4, pp. 365–393, 2017.

[55] U. von Luxburg, *Statistical learning with similarity and dissimilarity functions*. Doctoral dissertation, Technische Universität Berlin, 2004.

[56] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC Press, 2015.

[57] N. Meinshausen and P. Bühlmann, "High-dimensional graphs and variable selection with the lasso," *The Annals of Statistics*, vol. 34, no. 3, p. 1436–1462, 2006.

[58] M. J. Wainwright, "Sharp thresholds for high-dimensional and noisy sparsity recovery using ℓ_1-constrained quadratic programming (lasso)," *IEEE transactions on information theory*, vol. 55, no. 5, pp. 2183–2202, 2009.

[59] J. Friedman, T. Hastie, and R. Tibshirani, "Sparse inverse covariance estimation with the graphical lasso," *Biostatistics*, vol. 9, no. 3, pp. 432–441, 2008.

[60] J. Friedman, T. Hastie, and R.Tibshirani, "Applications of the lasso and grouped lasso to the estimation of sparse graphical models," *Technical report, Stanford University*, pp. 1–22, 2010.

[61] D. M. Witten, J. H. Friedman, and N. Simon, "New insights and faster computations for the graphical lasso," *Journal of Computational and Graphical Statistics*, vol. 20, no. 4, pp. 892–900, 2011.

[62] D. Hallac, Y. Park, S. Boyd, and J. Leskovec, "Network inference via the time-varying graphical lasso," *In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 205–213, 2017.

[63] T. Wang, Z. R. Y. Ding, Z. Fang, Z. S. M. L. MacDonald, R. A. Sweet, J. Wang, and W. Chen, "Fastggm: An efficient algorithm for the inference of gaussian graphical model in biological networks," *PLOS Computational Biology*, vol. 12, no. 2, pp. 1–16, 2016.

[64] J. Zhang, M. Wang, Q. Li, S. Wang, X. Chang, and B. Wang, "Quadratic sparse gaussian graphical model estimation method for massive variables," *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, pp. 2964–2972, 2020.

[65] V. Chandrasekaran, P. Parrilo, and A. S. Willsky, "Latent variable graphical model selection via convex optimization," *The Annals of Statistics*, vol. 40, pp. 1935–1967, 2012.

[66] P. Xu, J. Ma, and Q. Gu, "Speeding up latent variable gaussian graphical model estimation via nonconvex optimizations," *Advances in Neural Information Processing Systems*, 2017.

[67] B. Frot, L. Jostins, and G. McVean, "Graphical model selection for gaussian conditional random fields in the presence of latent variables," *Journal of the American Statistical Association*, vol. 114, no. 526, pp. 723–734, 2019.

[68] H. Liu, J. Lafferty, and L. Wasserman, "nonparanormal: Semiparametric estimation of high dimensional undirected graphs," *The Journal of Machine Learning Research (JMLR)*, vol. 10, p. 2295–2328, 2009.

[69] J. Lafferty, H. Liu, and L. Wasserman, "Sparse nonparametric graphical models," *Statistical Science*, vol. 27, no. 4, p. 519–537, 2012.

[70] J. J. Mulgrave and S. Ghosal, "Bayesian inference in nonparanormal graphical models," *Bayesian Analysis*, vol. 15, no. 2, pp. 449–475, 2020.

[71] E. Candès, X. Li, y. MA, and j. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, pp. 1–37, 2011.

[72] M. Yuan, "Discussion: Latent variable graphical model selection via convex optimization," *The Annals of Statistics*, vol. 40, no. 4, pp. 1968–1972, 2012.

[73] B. Li and E. Solea, "A nonparametric graphical model for functional data with application to brain networks based on fmri," *Journal of the American Statistical Association*, vol. 113, no. 524, p. 1637–1655, 2018.

[74] E. Solea and H. Dette, "Nonparametric and high-dimensional functional graphical models," *arXiv preprint arXiv:2103.10568s*, 2021.

[75] G. J. McLachlan and T. Krishnan, *The EM Algorithm and Extensions*, 2nd ed. Wiley-Interscience, 2008.

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

[77] 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.

[78] B. Everitt, S. Landau, M. Leese, and D. Stahl., *Miscellaneous clustering methods*. SWiley New York, NY, USA, 2011, pp. 215–255.

[79] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[80] J. Kossen, N. Band, C. L. A. Gomez, T. Rainforth, and Y. Gal, "Self-attention between datapoints: Going beyond individual input-output pairs in deep learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 742–28 756, 2021.

[81] G. Camps-Valls, J. Verrelst, J. Munoz-Mari, V. Laparra, F. Mateo-Jiménez, and J. Gómez-Dans, "A survey on gaussian processes for earth-observation data analysis: A comprehensive investigation," *IEEE Geoscience and Remote Sensing Magazine*, vol. 4, no. 2, pp. 58–78, 2016.

[82] M. Seeger, C. Williams, and N. Lawrence, "Fast forward selection to speed up sparse gaussian process regression," *In Artificial Intelligence and Statistics*, 2003. [Online]. Available: http://infoscience.epfl.ch/record/161318

[83] T. Bertin-Mahieux, D. Ellis, B. Whitman, and P. Lamere, "The million song dataset," *International Society for Music Information Retrieval Conference, ISMIR*, pp. 1—-6, 2011.

[84] 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.

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

[86] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont, "Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data," *Journal of Machine learning research*, vol. 9, no. Mar, pp. 485–516, 2008.

[87] 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.

[88] A. Denzel and J. Kästner, "Gaussian process regression for geometry optimization," *he Journal of Chemical Physics*, vol. 148, no. 9, p. 094114, 2018.

[89] R. Gramacy, *Surrogates: Gaussian Process Modeling, Design, and Optimization for the Applied Sciences*. Chapman and Hall/CRC, 2020.

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

[91] 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.

[92] 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.

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

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

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

[96] N. S. Altman, "An introduction to kernel and nearest-neighbor non-parametric regression," *The American Statistician*, vol. 46, pp. 175–185, 1992.

[97] J. Read and F. Perez-Cruz, "Deep learning for multi-label classification," *arXiv preprint arXiv:1502.05988*, 2014.

[98] J. Du, Q. Chen, Y. Peng, Y. Xiang, C. Tao, and Z. Lu, "Ml-net: multi-label classification of biomedical texts with deep neural networks," *Journal of the American Medical Informatics Association*, vol. 26, no. 11, pp. 1279–1285, 2019.

[99] T. Nguyen and E. Bonilla, "Fast allocation of gaussian process experts," *Proceedings of the 31st International Conference on Machine Learning*, vol. 32, no. 1, pp. 145–153, 2014. [Online]. Available: https://proceedings.mlr.press/v32/nguyena14.html

Appendix A

Aggregating Dependent Gaussian Experts in Local Approximation

This chapter is based on the following article:

- Hamed Jalali and Gjergji Kasneci. Aggregating Dependent Gaussian Experts in Local Approximation. In 25th International Conference on Pattern Recognition (ICPR 2020), 2020. doi: 10.1109/ICPR48806.2021.9413079.

A.1 Abstract

Distributed Gaussian processes (DGPs) are prominent local approximation methods to scale Gaussian processes (GPs) to large datasets. Instead of a global estimation, they train local experts by dividing the training set into subsets, thus reducing the time complexity. This strategy is based on the conditional independence assumption, which basically means that there is a perfect diversity between the local experts. In practice, however, this assumption is often violated, and the aggregation of experts leads to sub-optimal and inconsistent solutions. In this paper, we propose a novel approach for aggregating the Gaussian experts by detecting strong violations of conditional independence. The dependency between experts is determined by using a Gaussian graphical model, which yields the precision matrix. The precision matrix encodes conditional dependencies between experts and is used to detect strongly dependent experts and construct an improved aggregation. Using both synthetic and real datasets, our experimental evaluations illustrate that our new method outperforms other state-of-the-art (SOTA) DGP approaches while being substantially more time-efficient than SOTA approaches, which build on independent experts.

A.2 Introduction

Gaussian processes (GPs) [10] are flexible, interpretable, and powerful non-parametric statistical methods which provide accurate prediction with a low amount of uncertainty. They apply Bayes' theorem for inference, which allows them to estimate complex linear and non-linear structures without the need for restrictive assumptions of the model. They have been extensively used in practical cases, e.g. optimization [11], data visualization, and manifold learning [12], reinforcement learning [15], multitask learning [17], online streaming models [19, 20], and time series analysis [21, 22]. The main bottleneck of using standard GPs is that they poorly scale with the size of the dataset. For a dataset of size N, the training complexity is $\mathcal{O}(N^3)$ because the inversion and determinant of the $N \times N$ kernel matrix are needed. The prediction over a test set and also storing the results suffers from an additional complexity of $\mathcal{O}(NlogN)$. This issue currently restricts GPs to relatively small training datasets, the size of which is typically in the order of $\mathcal{O}(10^4)$.

To deal with large datasets two different strategies are used. The first strategy is based on sampling a small subset of the full dataset. The methods that follow this strategy try to train a GP on the smaller subset and then generalize the results. A simple method in this case, is subset-of-data (SoD) [26] which only uses a subset of size m from the original dataset; its training complexity is $\mathcal{O}(m^3)$ where $m << N$. Since this approach ignores the remaining data, it has limited performance.

Another method which is called sparse kernel or compactly supported kernel [28, 29], ignores the observations that are not correlated or show a covariance that is smaller than a threshold. In radial-based kernels, if the distance between two different entries is larger than a determined value, their covariance are set to zero. Although the training complexity of this method is $\mathcal{O}(\alpha N^3)$, for certain interesting cases, it does not guarantee that the new modified kernel is positive semi-definite.

The most popular method in this area is the sparse approximation approach, which employs a subset of the data (called inducing points) and *Nyström* approximation to estimate the prior and posterior distributions [30, 31]. For m inducing points, the training complexity is $\mathcal{O}(Nm^2)$. Although the authors provide a full probabilistic model using the Bayesian framework, it is not conceivable to apply the method to large and high dimensional datasets because its capability is restricted by the number of the inducing points [33, 34].

The second strategy is to divide the full dataset into partitions, train the local GPs in each partition [35, 36], and then aggregate the local approximations [37, 1, 38]. Unlike sparse approximations, this local approach can model quick-varying systems and non-stationary data features. Since in this family, the training procedure is run in different subsets, the final prediction may be affected by the regions with the poor predictive performance or by discontinuous predictions in overlapping sub-regions.

The most popular local approximation methods are the mixture of experts (MoE) and the product of experts (PoE). The MoE works as a Gaussian mixture model (GMM). It combines the local experts with their hyper-parameters and improves the overall predictive power [39, 40]. The main drawback of this method is that a joint training is needed to learn the mixing probabilities and the individual experts. This joint training positively affects the predictive power and helps control the experts with poor performance, but - on the negative side - it increases the complexity [37].

The prominent product of experts (PoE) [41] and Bayesian committee machine (BCM) [42] provide a new framework for GPs. Independent experts are GPs that are learned separately. Both methods suffer from the discontinuity issue and the weak experts' problem [81, 38]. The generalized product of experts (GPoE) [37] and robust Bayesian committee machine (RBCM) [1] propose different aggregation criteria, which are robust to weak experts' predictions.

To cope with the consistency problem in the predictions, [44] suggested the nested pointwise aggregation of experts (NPAE), which provides consistent predictions but increases the time complexity. The authors of [38] proposed the generalized robust Bayesian committee machine (GRBCM) by considering one expert as a base expert, i.e., a global expert that is modified the RBCM to provide consistent predictions. The authors in [38] showed that this modified RBCM is capable of providing consistent predictions, especially for the disjoint data partitioning regime.

The idea behind the BCM and PoE families of methods is the conditional independence (CI) assumption between the local experts. These divide-and-conquer approaches can speed up the computation and provide a distributed learning framework. However, since the CI assumption is violated in practice, they return poor results in cases with dependent experts.

The key contribution of our work lies in considering the dependency between Gaussian experts and improve the prediction quality in an efficient way. To this end, we first develop an approach to detect the conditional correlation between Gaussian experts, and then we modify the aggregation using this knowledge. In the first step, a continuous form of a Markov random field is used to infer dependencies and then the expert set is divided into clusters of dependent experts. In the second step, we adopt GRBCM for this new scenario and present a new aggregation method that is accurate and efficient and leads to better predictive performance than other SOTA approaches, which use the CI assumption.

The structure of the paper is as follows. Section II introduces the GP regression problem and SOTA DGP approaches. In Section III the proposed model and the inference process are presented. Section IV shows the experimental results, and we conclude in Section V.

A.3 Problem Set-up

A.3.1 Background

Let us consider the regression problem $y = f(x) + \varepsilon$, where $x \in R^D$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and the Gaussian likelihood is $p(y|f) = \mathcal{N}(f, \sigma^2 I)$. The objective is to learn the latent function f from a training set $\mathcal{D} = \{X, y\}_{i=1}^{n}$. The Gaussian process regression is a collection of function variables any finite subset of which has a joint Gaussian distribution. The GP then describes a prior distribution over the latent functions as $f \sim GP\left(m(x), k(x, x^{'})\right)$, where $m(x)$ is a mean function and $k(x, x^{'})$ is the covariate function (kernel) with hyper-parameter ψ. The prior mean is often assumed as zero, and the kernel is the well-known squared exponential (SE) covariance function equipped with automatic relevance determination (ARD),

$$k(x, x^{'}) = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{D}\frac{(x_i - x_i^{'})^2}{\mathcal{L}_i}\right)$$

where σ_f^2 is a signal variance, and $\mathcal{L}_i$ is an input length-scale along the ith dimension, and $\psi = \{\sigma_f^2, \mathcal{L}_1, \ldots, \mathcal{L}_D\}$. To train the GP, the hyper-parameters $\theta = \{\sigma^2, \psi\}$ should be determined such that they maximise the log-marginal likelihood [10]

$$\log p(y|X) = -\frac{1}{2} y^T \mathcal{C}^{-1} y - \frac{1}{2}\log|\mathcal{C}| - \frac{n}{2}\log(2\pi) \tag{A.1}$$

where $\mathcal{C} = K + \sigma^2 I$. For a test set x^* of size n_t, the predictive distribution is also a Gaussian distribution $p(y^*|D, x^*) \sim \mathcal{N}(\mu^*, \Sigma^*)$, with mean and covariance respectively given by

$$\mu^* = k_*^T (K + \sigma^2 I)^{-1} y, \tag{A.2}$$

$$\Sigma^* = k_{**} - k_*^T (K + \sigma^2 I)^{-1} k_*, \tag{A.3}$$

where $K = k(X, X)$, $k_* = k(X, x^*)$, and $k_{**} = k(x^*, x^*)$.

According to (A.1), the training step scales as $\mathcal{O}(n^3)$ because it is affected by the inversion and determinant of $\mathcal{C}$, which is an $n \times n$ matrix. Therefore, for large datasets, training is a time-consuming task and imposes a limitation on the scalability of the GP.

A.3.2 Distributed Gaussian Process

To scale the GP to large datasets, the cost of the standard GP is reduced by distributing the training process. It involves dividing the full training dataset $\mathcal{D}$ into M partitions $\mathcal{D}_1, \ldots, \mathcal{D}_M$, (called experts) and training the standard GP on these partitions. The predictive distribution of the i'th expert $\mathcal{M}_i$ is $p_i(y^*|\mathcal{D}_i, x^*) \sim \mathcal{N}(\mu_i^*, \Sigma_i^*)$, where its mean and variance are calculated by using (A.2) and (A.3) respectively

$$\mu_i^* = k_{i*}^T (K_i + \sigma^2 I)^{-1} y_i, \tag{A.4}$$

$$\Sigma_i^* = k_{**} - k_{i*}^T (K_i + \sigma^2 I)^{-1} k_{i*}. \tag{A.5}$$

Aggregating these experts is based on the assumption that they are independent. The most prominent aggregation methods are PoE [41] and BCM [42]. GPoE [37] and RBCM [1] are new modified versions of PoE and BCM, which approach the discontinuity problem and overconfident predictions.

The term distributed Gaussian process was proposed by [1] to include PoE, BCM, and their derivatives, which are all based on the fact that the computations of the standard GP is distributed amongst individual computing units. Unlike sparse GPs, DGPs make use of the full dataset but divide it into individual partitions.

The predictive distribution of DGP is given as the product of multiple densities (i.e., the experts). If the experts $\{\mathcal{M}\}_{i=1}^M$ are independent, the predictive distribution of DGP for a test input x^* is

$$p(y^*|\mathcal{D}, x^*) \propto \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i, x^*). \tag{A.6}$$

The weights $\beta = \{\beta_1, \ldots, \beta_M\}$ describe the importance and influence of the experts. The typical choice of the weights is the difference in differential entropy between the prior $p(y^*|x^*)$ and the posterior $p(y^*|\mathcal{D}, x^*)$ [37]. With such weights however, the predictions of GPoE are too conservative and the predictions are not appropriate [38]. To address this issue, the simple uniform weights $\beta_i = \frac{1}{M}$ is used [1]. The predictive distribution of GPoE with normalized weights asymptotically converges to the full Gaussian process distribution but is too conservative [45].

A.3.3 Discussions of the Properties Existing Aggregations

Consistency To deal with the inconsistency issue, the nested pointwise aggregation of experts (NPAE) [44] considers the means of the local predictive distributions as random variables by assuming that y_i has not been observed and therefore allows the dependency between individual experts' predictions. Theoretically, it provides consistent prediction but its aggregation steps need much higher time complexity. For M individual partitions, NAPE needs to calculate the inverse of a $M \times M$ matrix in each test point that leads to a longer running time when a large training set is used or the number of partitions is large.

Another new model is the generalized robust Bayesian committee machine (GRBCM) [38] which introduces a base (global) expert and considers the covariance between the base and other local experts. For a global expert, M_b, in a base partition, D_b, the predictive distribution of GRBCM is

$$p(y^*|\mathcal{D},x^*) = \frac{\prod_{i=2}^{M} p_{bi}^{\beta_i}(y^*|\mathcal{D}_{bi},x^*)}{p_b^{\sum_{i=2}^{M}\beta_i - 1}(y^*|\mathcal{D}_b,x^*)}, \tag{A.7}$$

where $p_b(y^*|\mathcal{D}_b,x^*)$ is the predictive distribution of M_b, and $p_{bi}(y^*|\mathcal{D}_{bi},x^*)$ is the predictive distribution of an expert trained on the dataset $\mathcal{D}_{bi} = \{\mathcal{D}_b,\mathcal{D}_i\}$. It improves the prediction and consistency of the RBCM has time complexity $\mathcal{O}(\alpha nm_0^2) + \mathcal{O}(\beta n' nm_0)$, where m_0 in the number of assigned points to each expert, n' is the size of test set, $\alpha = (8M-7)/M$, and $\beta = (4M-3)/M$[38].

Conditional independence (CI) CI is a crucial assumption for many unsupervised ensemble learning methods. It has been used widely in regression and classification problems [48, 49]. The (R)BCM and (G)PoE methods are also based on CI assumption which reduces the computational costs of the training process, see Figure A.1a.

However, in practice, this assumption is often violated and their predictions are not accurate enough. Actually, the ensembles based on CI return sub-optimal solutions [2]. In this regard, few works have considered modelling dependencies between individual predictors. For classification, [50] used pairwise interaction between classifiers and [51] considered the agreement rates between subsets of experts. In another work [2], the authors suggested a model using clusters of binary classifiers in which the classifiers

in each cluster are conditionally dependent. The authors of [2] defined a specific score function based on covariance between classifiers to detect dependency.

In local approximation GPs, the only method that considered the dependency between experts is NPAE[44, 46]. It assumes the joint distribution of experts and y^* is a Gaussian distribution and uses the properties of conditional Gaussian distributions to define the meta-learner. Due to the high computational cost which cubically depends on the number of experts at each test point, this method does not provide an efficient solution for large real-world datasets.

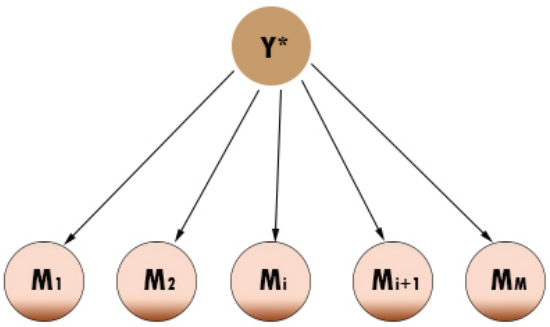
In the next section, we propose a new model that uses the dependency between experts and define a modified aggregation method based on GRBCM.

A.4 Distributed Gaussian Process with Dependent Experts

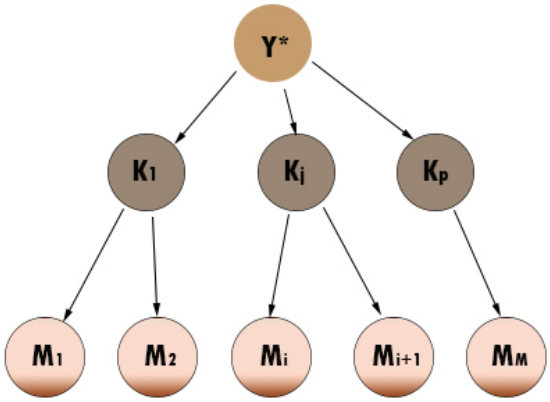
Assume the Gaussian experts $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ have been trained on different partitions and let $\mu^*_{\mathcal{M}} = [\mu^*_1, \ldots, \mu^*_M]^T$ be a $n_t \times M$ matrix that contains the local predictions of M experts at n_t test points. Our approach makes use of the experts' predictions, i.e. $\mu^*_{\mathcal{M}}$ in order to detect strong dependencies between experts. This step results clusters of correlated experts, $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_P\}$, $P \ll M$. By aggregating the experts at each cluster, it leads to a new layer of experts, $\mathcal{K} = \{\mathcal{K}_1, \ldots, \mathcal{K}_P\}$, which are conditionally independent given y^*. Figure A.1b depicts this model where the experts in cluster $\mathcal{C}_i$ are conditionally independent given $\mathcal{K}_i$, and each $\mathcal{K}_i$ is independent of $\mathcal{K}_j, i \neq j$ given y^*. The final prediction is done by using the $\mathcal{K}$ instead of $\mathcal{M}$.

Definition 2 (**Assignment function**). A function $\mathcal{H} : \mathcal{M} \to \mathcal{C}$ is the assignment function that represents the related cluster for each expert. $\mathcal{H}(\mathcal{M}_i) = \mathcal{C}_j$ means that the i'th expert belongs to j'th cluster of experts and it has a dependency with the experts in this cluster. Therefore, if $\mathcal{H}(\mathcal{M}_i) = \mathcal{H}(\mathcal{M}_j)$, the i'th and j'th experts are correlated to each other and belong to the same cluster.

In the following, we will show how to detect subsets of strongly dependent experts and present a new aggregation method for DGPs.



(a)



(b)

Figure A.1: **Computational graphs**: (a) DGP model with CI [1]; (b) DGP with clusters of dependent experts [2]

A.4.1 Dependency Detection with Gaussian Graphical Models

The key idea in an undirected graphical model (or pairwise Markov random field (MRF)) is to model the set of local estimators as a connected network such that each node represents a Gaussian expert and the edges are the interaction between them. This network model uses a matrix of parameters to encode the graph structure. In other words, it considers the edges as parameters, such that if there is a connection between two nodes, then there are non-zero parameters for the pair.

Gaussian graphical models (GGMs). Gaussian graphical models (GGMs) [52, 53, 54] are continuous forms of pairwise MRFs, i.e. the nodes are continuous. The basic assumption for GGMs is that the variables in the network follow a multivariate Gaussian distribution. The distribution for GGMs is

$$p(Z|\mu,\Sigma) = \frac{1}{(2\pi)^{Q/2}|\Sigma|^{1/2}} \exp\left\{-\frac{1}{2}(Z-\mu)^T\Sigma^{-1}(Z-\mu)\right\}, \tag{A.8}$$

where $Z = \{Z_1,\ldots,Z_Q\}$ are the variables (nodes), Q is the number of variables, and μ and Σ are the mean and covariance, respectively. The distribution in (A.8) can also be expressed using the precision matrix Ω:

$$\begin{aligned} p(Z|h,\Omega) &= \frac{|\Omega|^{1/2}}{(2\pi)^{Q/2}} \exp\left\{-\frac{1}{2}(Z-h)^T\Omega(Z-h)\right\} \\ &\propto \exp\left\{-\frac{1}{2}Z^T\Omega Z + h^T Z\right\}, \end{aligned} \tag{A.9}$$

where $\Omega = \Sigma^{-1}$ and $h = \Omega\mu$. The matrix Ω is also known as the potential or information matrix.

WLOG, let $\mu = 0$, then the distribution of a GGM shows the potentials defined on each node i as $exp\{-\Omega_{ii}(Z_i)^2\}$ and on each edge (i,j) as $exp\{-\Omega_{ij}Z_iZ_j\}$.

Unlike correlation networks, Equation (A.8), which encode the edge information in the network on the covariance matrix, a GGM is based on the precision matrix, Equation (A.9). In a correlation network, if $\Sigma_{ij} = 0$, then Z_i ard Z_j are assumed to be independent. While in a GGM, if $\Omega_{ij} = 0$, then Z_i ard Z_j are conditionally independent given all other variables, i.e. there is no edge between Z_i ard Z_j in the graph.

Network Learning. In the network, the locally trained experts are the nodes, and network learning results in the precision matrix Ω; the latter reveals the conditional dependencies between experts. GGMs use the common sparsity assumption, that is, there are only few edges in the network and thus the parameter matrix is sparse. This assumption usually makes sense in experts' networks because the interaction of one expert is limited to only a few other experts.

To this end, the Lasso regression [56] is used to perform neighborhood selection for the network. The Meinshausen-Bühlmann algorithm [57] is one of the first algorithms

in this area. [57] and [58] proved that with some assumptions, Lasso asymptotically recovers correct relevant subsets of edges. [59] proposed the efficient graphical Lasso which adopts a maximum likelihood approach subject to an L1 penalty on the coefficients of the precision matrix. The graphical Lasso has been improved in later works [60, 61, 62].

Let S be the sample covariance. Then, the Gaussian log-likelihood of the precision matrix Ω is equal to $log|\Omega| - trace(S\Omega)$. The Graphical Lasso (gLasso) maximizes this likelihood subject to an element-wise L_1 norm penalty on Ω. Precisely, the objective function is

$$\hat{\Omega} = \arg\max_{\Omega} \log|\Omega| - trace(S\Omega) - \lambda \|\Omega\|_1 , \tag{A.10}$$

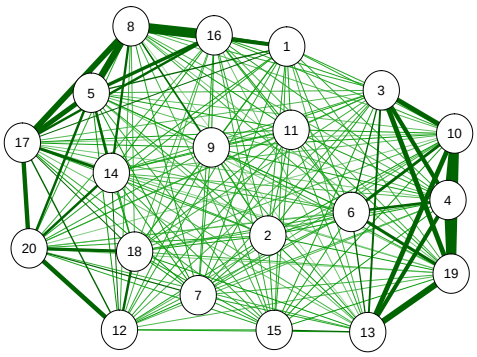
where the estimated neighborhood is then the non-zero elements of $\hat{\Omega}$. Since $\hat{\Omega}$ contains all information about the dependency between experts, we use it to construct the assignment function $\mathcal{C}$ and the clusters of experts $\mathcal{K}$.

A.4.2 Aggregation

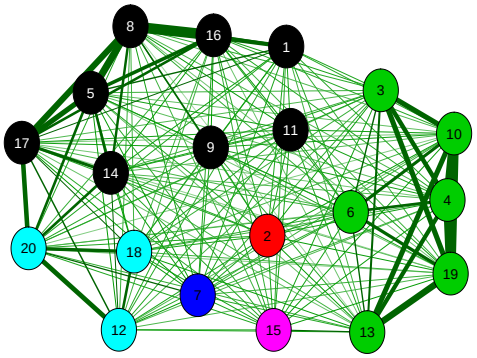
After determining the dependencies between the experts, we apply the following aggregation method. First, we define clusters of interdependent experts, i.e. that include experts with strong dependency. Then, by using the GRBCM method in ith cluster, we generate for each cluster a modified expert $\mathcal{K}_i$. The final prediction is done by aggregating the predictions of these modified experts.

Experts clustering After detecting the dependencies between experts, we use the precision matrix to find the assignment function. Performing a clustering approach on the precision matrix returns the clusters of experts $\mathcal{K}$; thus, each cluster $\mathcal{K}_i$ contains strongly dependent experts based on the precision matrix. To this end, we apply *spectral clustering (SC)* [55] which is more robust and works better in practice. Spectral clustering makes use of the relevant eigenvectors of the Laplacian matrix of the similarity matrix (here the precision matrix) alongside a standard clustering method. The Laplacian matrix is $L = D - \Omega$, where D is a diagonal matrix that includes the sum of the values in each row of Ω.

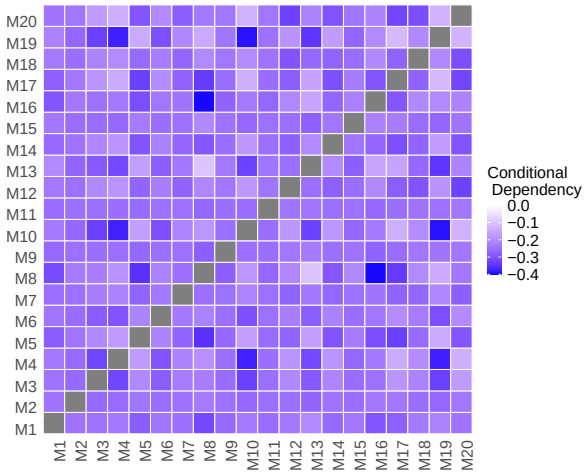
Figure A.2 depicts the GGM of a simulated dataset with 10^5 training points which



(a) $\lambda = 0.1$



(b) $\lambda = 0.1, P = 6$



(c) $\lambda = 0.1$

Figure A.2: **Gaussian graphical models**:(a) shows the interaction between experts in a GGM of 20 experts with a penalty term $\lambda = 0.1$, (b) reveals the GGM with 6 clusters of experts, and (c) depicts the *heat map* plot of the experts' precision matrix.

have been divided into 20 partitions (experts). This dataset is considered in Section A.5.1 in detail. Figure A.2a represents the sparse graph with a penalty term $\lambda = 0.1$ in graphical Lasso with the nodes (experts) and edges (interactions or dependencies). Even with this penalty term, the CI assumption is violated because all experts are connected to each other. Figure A.2b displays the graph after performing the spectral clustering on the precision matrix. The 6 clusters in the graph contain correlated experts, and clusters that are now represented by only one expert (e.g. the cluster with red color) contain the original experts that are not strongly dependent with the other experts. Figure A.2c

represents the *heat map* plot of the symmetric precision matrix and shows the conditional dependencies between experts. The main diagonal returns the experts internal potential while the other elements are conditional dependencies between experts. In fact this figure shows that the experts are conditionally dependent and the CI assumption is violated.

Final Aggregation We assume that the new experts $\{\mathcal{K}_1,\dots,\mathcal{K}_P\}$ are conditionally independent given y^* (see Figure A.1b), which is not a strong assumption due to the process by which they were generated. The task is to find the distribution of new experts $\{\mathcal{K}_1,\dots,\mathcal{K}_P\}$ and then find $p(y^*|\mathcal{D},x^*)$. The authors in [38] showed that GRBCM provides consistent predictions under some mild assumptions, i.e. it can recover the true posterior distribution of y^* when $n\to\infty$. Hence, we use the GRBCM aggregation method in each cluster by adding the global communication expert M_b to all clusters. For aggregating the new experts, we use either GPoE or GRBCM. Since the number of experts that are aggregated in each step is smaller than M, the computational cost of this scenario is smaller than the computational cost of GRBCM. Algorithm 1 depicts the aggregation process.

Algorithm 1 Aggregating Dependent Local Gaussian Experts

Input: μ^*_M, λ, P

1: Calculate sample covariance S of experts' predictions
2: Estimate $\hat{\Omega}$ using (A.10)
3: Estimate $\mathcal{H}$ by performing spectral clustering $SC(\hat{\Omega}, \mathrm{P})$
4: Obtain new experts $\{\mathcal{K}_1,\dots,\mathcal{K}_P\}$ using GRBCM (A.7)
5: Aggregate new experts using GPoE (A.6) or GRBCM (A.7)
6: **return** The estimated mean and variance of $p(y^*|\mathcal{D},x^*,\mathcal{K})$, i.e. $\mu^*_{\mathcal{K}}$ and $\Sigma^*_{\mathcal{K}}$.

The following proposition gives our predictive distribution and its asymptotic properties.

Proposition 2 (Predictive Distribution). Let X be a compact, nonempty subset of $\mathcal{R}^{n\times D}$, $\mu^*_M = [\mu^*_1,\dots,\mu^*_M]^T$ be the sub-models' predictions. We use $\{\mathcal{K}_1,\dots,\mathcal{K}_P\}$ as defined in Algorithm 1. We further assume that (i) $\lim_{n\to\infty} M = \infty$, (ii) $\lim_{n\to\infty} m_0 = \infty$, where m_0 is the partition size, and (iii) $\lim_{n\to\infty} |\mathcal{C}_i| = \infty$, $i = 1,\dots,P$, where $|\mathcal{C}_i|$ is the size of i'th cluster. The second condition implies that the original experts become more informative with increasing n, while the third condition means that the number of experts in each

cluster increases. In addition, the third condition implies that $P \ll M$, which describes the dependency between the experts. [1] Then the estimation based on Algorithm 1, $y^*_{\mathcal{K}}$ is consistent, i.e.

$$\begin{cases} \lim_{n\to\infty} \mu^*_{\mathcal{K}} = \mu^* \\ \lim_{n\to\infty} \Sigma^*_{\mathcal{K}} = \Sigma^*. \end{cases} \tag{A.11}$$

Proof: The proof is straightforward due to the consistency of GRBCM. According to assumptions (ii) and (iii), when $n \to \infty$, each cluster returns a consistent predictor because the aggregation inside the clusters is based on GRBCM. Combining the consistent new experts $\{\mathcal{K}_1, \ldots, \mathcal{K}_P\}$ in Step 5 of Algorithm 1 leads to a consistent prediction. We provide here the proof for the variance, when GPoE is used in Step 5, and note that the proof for the mean is analogous. Let $\Sigma^*_{\mathcal{K}_i}$ be the covariance matrix of $\mathcal{K}_i$ which is obtained in Step 4 of Algorithm 1, then the aggregated precision of GPoE (Step 5 of Algorithm 1) is equal to

$$\begin{aligned} \lim_{n\to\infty} (\Sigma^*_{\mathcal{K}})^{-1} &= \lim_{n\to\infty} \sum_{i=1}^{P} \frac{1}{P} (\Sigma^*_{\mathcal{K}_i})^{-1} = \sum_{i=1}^{P} \frac{1}{P} \lim_{n\to\infty} (\Sigma^*_{\mathcal{K}_i})^{-1} \\ &= \sum_{i=1}^{P} \frac{1}{P} (\Sigma^*)^{-1} = (\Sigma^*)^{-1}, \end{aligned}$$

where the first equality is based on the definition of GPoE with equal weights and the third one is due to the consistency of GRBCM.

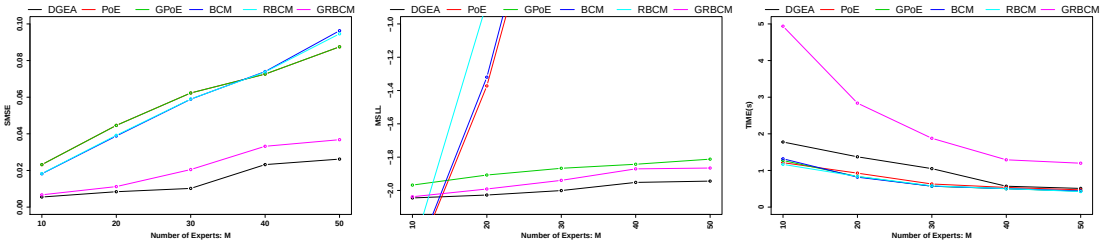
In the next section, we showcase the importance of taking local experts' dependencies into account and the competitive performance of our approach using both artificial and real-world datasets.

A.5 Experiments

The prediction quality of the proposed dependent Gaussian expert aggregation method (DGEA) is assessed in this Section. We showcase the importance of taking local experts' dependencies into account and the competitive performance of our approach using both

[1] If we assume perfect diversity between experts (i.e., CI), then $P \approx M$. In this case, the consistency still holds due to the consistency of GRBCM but it is not a realistic assumption.

artificial and real datasets. 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 prediction mean, while the MSLL evaluates the quality of predictive distribution [10]. The standard squared exponential kernel with automatic relevance determination and a Gaussian likelihood is used. The experiments have been done in MATLAB using the GPML package[2]. The random partitioning method on the training dataset has been used in all experiments.



(a) SMSE (b) MSLL (c) Time

Figure A.3: **Prediction quality** of different DGP methods with respect for different number of experts in the simulated data of the analytic function by (A.12).

A.5.1 Toy Example

The goal of our first experiment is to study the effect of dependency detection on the prediction quality and computation time. It is based on simulated data of a one-dimensional analytical function [38],

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

where $\varepsilon \sim \mathcal{N}(0, (0.2)^2)$. We generated $n = 10^4$ training points in $[0, 1]$, and $n_t = 0.1n$ test points in $[-0.2, 1.2]$. The data is normalized to zero mean and unit variance. We assigned 200, 250, 330, 500 and 1000 data points to each expert, which leads to 50, 40, 30, 20, and 10 experts respectively. Figure A.3 shows the sensitivity of different DGP methods with respect to the change in the number of experts.

[2] `http://www.gaussianprocess.org/gpml/code/matlab/doc/`

The DGEA prediction is based on using GPoE in Step 5 of Algorithm 1. Figure A.3a depicts the SMSE values of the different SOTA methods. Since PoE and GPoE have the same SMSE value, the line of PoE is hidden in the plot. By increasing the number of experts, the prediction error rises, because we assign a smaller amount of observations to each expert, and therefore the quality of each expert decreases. While (G)PoE and (R)BCM return poor predictions, the DGEA and GRBCM present better results and DGEA has the smallest prediction error for the different numbers of experts.

Figure A.3b reveals the quality of the predictive distribution. The overconfident methods (PoE, BCM, and RBCM) return smaller MSLL for 10 experts. However, their MSLL value dramatically increases with growing M. As the authors in [46] and [45] have shown, the GPoE is a conservative method and thus returns a higher quality. In Figure A.3b we can see that its predictive distribution has a higher quality compared to PoE and (R)BCM. However, the DGEA represents an even higher quality for the predictive distribution for the different values of M. Figure A.3c shows the computational costs of different methods. Comparing the running time of DGEA and GRBCM demonstrates that DGEA takes about half of the time of GRBCM, while its running time is almost indistinguishable from the running time of the most efficient methods, (G)PoE and (R)BCM.

A.5.2 Realistic Datasets

In this section, we use four realistic datasets, *Pumadyn*, *Kin40k*, *Sacros*, and *Song*. The *Pumadyn*[3] is a generated 32D dataset with 7168 training points and 1024 test points. The 8D *Kin40k* dataset [82] contains 10^4 training points and 3×10^4 test points. The *Sacros*[4] is a 21D realistic medium-scale dataset with 44484 training and 4449 test points. The *Song* dataset [5] [83] is a 91D dataset with 515,345 instances which is divided into 463,715 training examples and 51,630 test examples. We extract the first 10^5 songs from this dataset for training and keep the original set of 51,630 songs for testing. The random partitioning method has been used to divide the dataset into partitions and to generate the experts. The number of experts is 20 for *Pumadyn* and *Kin40k*, 72 for *Sacros*, and

[3] https://www.cs.toronto.edu/~delve/data/pumadyn/desc.html
[4] http://www.gaussianprocess.org/gpml/data/
[5] https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd

150 for *Song*. For the *Pumadyn* and *Kin40k* datasets, 5 clusters, and for *Sacros* and *Song*, 10 clusters are used.

Table A.1: **Prediction quality** for various methods on *Pumadyn*, *Kin40k*, *Sacros*, and *Song* data. For both quality measure, i.e. SMSE and MSLL, smaller values are better.

	Pumadyn		*Kin40k*		*Sacros*		*Song*	
Model	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
DGEA (Ours)	**0.0486**	**-1.5133**	**0.0538**	**-1.3025**	**0.0269**	**-1.823**	**0.8084**	**-0.122**
PoE	0.0505	4.8725	0.856	2.4153	0.0311	25.2807	0.8169	69.9464
GPoE	0.0505	-1.4936	0.0856	-1.2286	0.0311	-1.7756	0.8169	**-0.123**
BCM	0.0499	4.6688	0.0818	1.6974	0.0308	24.868	10.4291	44.1745
RBCM	0.0498	12.1101	0.0772	2.5256	0.0305	61.5392	5.4373	1.2089
GRBCM	0.0511	-1.488	0.0544	-1.2785	0.0305	-1.4308	0.8268	0.2073

Table A.1 depicts the prediction quality of different methods. In the *Pumadyn*, *Kin40k*, and *Sacros* dataset, DGEA clearly outperforms the other methods. BCM and RBCM show lower prediction error compared to (G)PoE on these datasets, but their negative log-likelihood (MSLL) is quite large. Since GPoE provides conservative predictions and its posterior distribution converges to the true predictive distribution, it shows a nice performance with respect to the MSLL value, even better than the performance of GRBCM on the *Pumadyn*, and *Sacros* datasets. The drawback of PoE and (R)BCM methods can be seen in their MSLL values, which shows that their predictive distribution does not have competitive quality and tends to produce overconfident and inconsistent predictions, which has also been discussed by [45] and [38]. With respect to the *Song* dataset, DGEA and GPoE return better predictions. While GPoE performs a little bit better than DGEA with respect to MSLL, DGEA has a lower prediction error.

The GRBCM method returns different prediction qualities for different detests. Since in this work a new ensemble method is proposed for the non-parametric regression problem, the random partitioning is used, because in this case all Gaussian experts can cover the full sample space and work as global predictors. The quality of GRBCM is higher with respect to the disjoint partitioning, which is consistent with the results presented in [38]. But overall, the prediction quality of DGEA outperforms the other methods, which shows the importance of taking the experts' dependencies into account.

A.6 Conclusion

In this work, we have proposed DGEA, a novel DGP approach which leverages the dependencies between experts to improve the prediction quality through local aggregation of experts' predictions. To combine correlated experts, comparable SOTA methods assume conditional independence between experts, which leads to poor prediction in practice. Our approach uses an undirected graphical model to detect strong dependencies between experts and defines clusters of interdependent experts. Theoretically, we showed that our new local approximation approach provides consistent results when $n \to \infty$. Through empirical analyses, we illustrated the superiority of DGEA over existing SOTA aggregation methods for scalable GPs.

For future work, we identify two directions for further research. First, for the aggregated posterior, we integrated Gaussian graphical models into the generalized robust Bayesian committee machine and generalized product of experts. Another aggregation approach can be the latent variable graphical model, assuming that the final predictor is a latent variable within the graph that may improve the prediction quality by using interdependencies between experts while reducing time complexity. Second, the GGM relies on the assumption that all experts are jointly Gaussian and cannot be used to explain complex models with the non-Gaussian distribution. Therefore, finding a flexible and capable substitute for the GGM to capture the properties of GP experts is left to future work.

Appendix B

Aggregating the Gaussian Experts' Predictions via Undirected Graphical Models

This chapter is based on the following article:

- Hamed Jalali and Gjergji Kasneci. Aggregating the Gaussian Experts' Predictions via Undirected Graphical Models. In IEEE International Conference on Big Data and Smart Computing (BigComp), 2022. doi: 10.1109/BigComp54360.2022.00014.

  The appendix section B.7 is based on this article:

- Hamed Jalali and Gjergji Kasneci. Gaussian Graphical Models as an Ensemble Method for Distributed Gaussian Processes. OPT2021: 13th Annual Workshop on Optimization for Machine Learning at 35th Conference on Neural Information Processing Systems (NeurIPS 2021), 2021. doi: 10.48550/arXiv.2202.03287.

B.1 Abstract

Distributed Gaussian process (DGP) is a popular approach to scale Gaussian processes to big data which divides the training data into some subsets, performs local inference for each partition, and aggregates the results to acquire global prediction. To combine the local predictions, the *conditional independence assumption* is used which basically means there is a perfect diversity between the subsets. Although it keeps the aggregation tractable, it is often violated in practice and generally yields poor results. In this paper, we propose a novel approach for aggregating the Gaussian experts' predictions by Gaussian graphical model (GGM) where the target aggregation is defined as an unobserved latent variable and the local predictions are the observed variables. We first estimate the joint distribution of latent and observed variables using the Expectation-Maximization (EM) algorithm. The interaction between experts can be encoded by the precision matrix of the joint distribution and the aggregated predictions are obtained based on the property of conditional Gaussian distribution. Using both synthetic and real datasets, our experimental evaluations illustrate that our new method outperforms other state-of-the-art DGP approaches.

B.2 Introduction

Gaussian processes (GPs) are powerful non-parametric statistical methods based on Bayes' theorem. Without the need for restrictive assumptions, they are capable to estimate complex models with a low amount of uncertainty. Despite many advantages, GPs suffer from their computational costs where they poorly scale with the size of the dataset.

The prominent distributed Gaussian processes (also called local approximation GPs) are based on the divide-and-conquer approach. It means the training data is divided into some partitions (called experts), the local inference is done for each partition separately, and at the end, these local estimations are combined using an ensemble method. All experts share the same hyper-parameters, which leads to automatic regularisation and the model tends to prevent the overfitting of individual experts.

In a DGP, the *conditional independence* assumption (CI) between partitions allows factorizing the global posterior distribution as a product of local distributions. Although

this assumption reduces the computational cost, it is often violated in practice. However, solutions that deal with the dependency problem (e.g. NPAE method [44]) suffer from extra computational costs and therefore, are impractical for large data sets.

The key contribution of our work lies in aggregating the local experts' predictions considering their dependencies. Unlike conventional DGPs, here the CI assumption is violated to improve the prediction quality. The conditional dependency is inferred as the interactions between nodes in a continuous form of a Markov random field (MRF). We consider the local and latent experts as nodes of an undirected graph. Then, the Gaussian graphical model (GGM) is used to construct the undirected graph between Gaussian experts and their interactions. Since the latent expert is unobserved, we use the latent variable Gaussian graphical model (LVGGM) to estimate the joint distribution of observed and latent experts. The final predictions are the mean of the conditional distribution of the latent expert given observed experts. Relative to the available baselines, our approach substantially provides competitive prediction performance than other state-of-the-art (SOTA) approaches, which use the CI assumption.

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

B.3 Background and Problem Set-up

B.3.1 Background

Let us consider the regression problem $y = f(x) + \varepsilon$, where $x \in R^d$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, and the Gaussian likelihood is $p(y|f) = \mathcal{N}(f, \sigma^2 I)$. The objective is to learn the latent function f from a training set $\mathcal{D} = \{X, y\}$ of size n. The Gaussian process regression is a collection of random variables of which any finite subset has a joint Gaussian distribution. The GP then describes a prior distribution over the latent functions as $f \sim GP(0, k(x, x'))$, where $k(x, x')$ is the covariancee function (kernel) with hyperparameters ψ. To train the GP, the hyperparameters $\theta = \{\sigma^2, \psi\}$ should be determined such that they maximise the

log-marginal likelihood,

$$\log p(y|X) = -\frac{1}{2}y^T\mathcal{C}^{-1}y - \frac{1}{2}\log|\mathcal{C}| - \frac{n}{2}\log 2\pi, \tag{B.1}$$

where $\mathcal{C} = K + \sigma^2 I$ and $K = k(X,X)$. According to (B.1), the training step scales as $\mathcal{O}(n^3)$ because it is affected by the inversion and determinant of the $n \times n$ matrix $\mathcal{C}$. Therefore, for large data sets, GP training is a time-consuming task and imposes limitations on the scalability of GPs.

B.3.2 Distributed Gaussian Process

The term distributed Gaussian process [1] uses the fact that the computations of the standard GP can be distributed among individual computing units. To do that, one divides the full training data set $\mathcal{D}$ into M partitions (called experts) and trains standard GPs on these partitions. Let $\mathcal{D}' = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ be the partitions, and X_i and y_i be the input and output of partition $\mathcal{D}_i$. All GP experts are trained jointly and share a single set of hyper-parameters $\theta = \{\sigma^2, \psi\}$. For a test set X^* of size n_t, the local prediction of the i-th GP expert $\mathcal{M}_i$ is:

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

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

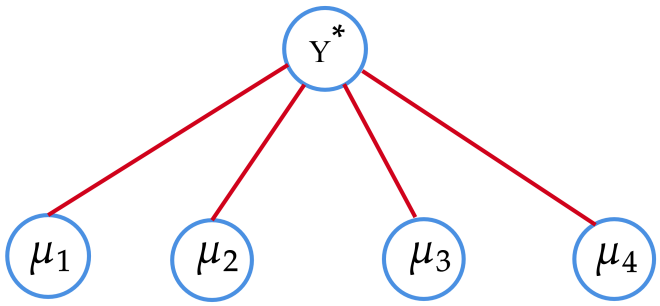
Aggregating the experts in DGP is based on the assumption that they are conditionally independent. If the experts $\{\mathcal{M}\}_{i=1}^M$ are independent, the posterior distribution of DGP is given as the product of multiple local densities. That is to say for a test input x^*

$$p(y^*|\mathcal{D}, x^*) \propto \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i, x^*), \tag{B.3}$$

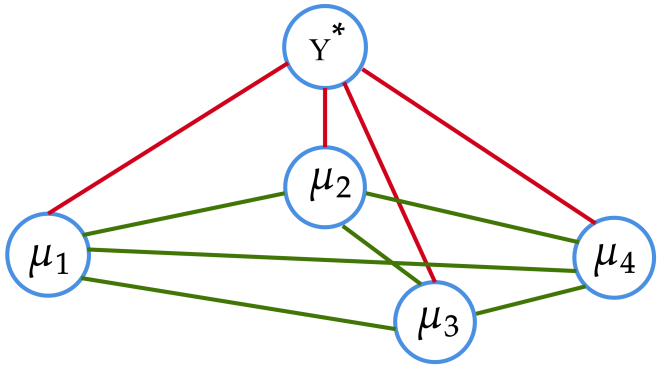
where the weights $\beta = \{\beta_1, \ldots, \beta_M\}$ describe the importance and influence of the experts. The most popular aggregation methods are generalised product of experts (GPoE), robust Bayesian committee machine (RBCM) and generalized robust Bayesian committee machine (GRBCM), see [38, 1].

B.3.3 Dependency

The CI assumption is used widely in ensemble methods for both regression and classification problems. The DGPs use CI to reduce the computational costs of the prediction process. However, their predictions are not accurate enough and CI-based aggregation generally returns sub-optimal solution [2, 3, 4]. In local approximation GPs, the dependency between experts has been discussed in few works. For instance, the nested pointwise aggregation of experts (NPAE) method [44] uses the internal correlation between local experts and the dependency between local experts and target variable y^*. However, this pointwise aggregation suffers from high time complexity which cubically depends on the number of experts at each test point, i.e. $\mathcal{O}(n_t M^3)$, and therefore, it is not an efficient solution for large datasets.



(a) Independent Experts



(b) Dependent Experts

Figure B.1: **Computational graphs** of (a) a Conditional-Independent based aggregation and (b) an aggregation based on the conditional dependency between local experts.

Figure B.1 shows the computational graphs of CI-based and dependency-based aggregation strategies. Figure B.1(a) reveals the aggregation based on conditional independence assumption between experts $\{\mu_1, \mu_2, \mu_3, \mu_4\}$. It means two local experts mu_i and mu_j are connected only via the target variable y^*, i.e. $mu_i \perp\!\!\!\perp mu_j \mid y^*$. However, this assumption is often violated in realistic conditions and the aggregation can lead to a sub-optimal solution. On the other hand, Figure B.1(b) represents an aggregation with dependent experts where the interactions between experts show the dependencies.

In the next section, we propose a new model that uses the dependency between experts and defines a modified aggregation method based on the latent variable Gaussian graphical model.

B.4 Aggregating Conditionally dependent Experts with an Undirected Graph

At the heart of our work is the following ingredient. First, we assume that y_i in (B.2) has not yet been observed, see [44]. Then the experts' predictions μ_i^* can be considered as a *random variable*. This allows us to leverage internal correlations between local experts, and also consider the correlation between the latent target variable y^* and local experts. Then, we exert the continuous form of Markov random field, called Gaussian graphical model (GGM), where the nodes of the graph are the experts (local and latent) and the edges are the interactions between them.

B.4.1 Aggregating Dependent Experts' Predictions

Assume the Gaussian experts $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ have been trained on separated subsets and let $\mu^* = [\mu_1^*, \ldots, \mu_M^*]^T$ be a $n_t \times M$ matrix that contains their centered predictions at n_t test points. As a consequence of the choice of the prior, the joint distribution of the local experts μ^* and target expert y^* is multivariate Gaussian distribution because any vector of linear combinations of observation is itself a Gaussian vector.

Let $\Sigma_{y^*\mu^*}$ encodes the correlation between latent expert y^* and local experts μ^*, and $\Sigma_{\mu^*\mu^*}$ depicts the correlation between local experts. Employing the properties of conditional Gaussian distributions for the centered random vector allows for the following

aggregation:

$$y_A^* = \Sigma_{y^*\mu^*}^T \Sigma_{\mu^*\mu^*}^{-1} \mu^*. \tag{B.4}$$

The linear estimator in (B.4) is the mean of conditional distribution of y^* given μ^*, i.e. $p(y^*|\mu^*)$ and is the best linear unbiased predictor of y^*, see [44].

In the next subsection, we show how the GGM can be adapted to the local approximation problem with a latent target variable and suggest a new method to compute the aggregated estimator y_A^*.

B.4.2 A Gaussian Graphical Models for Dependent Gaussian Experts

Gaussian graphical models [52, 53, 54] are continuous forms of pairwise MRFs that assume the variables in the network follow a multivariate Gaussian distribution. The distribution for a GGM is

$$p(\mu^*|\xi,\Omega) \propto \exp\left\{-\frac{1}{2}(\mu^* - \xi)^T \Omega(\mu^* - \xi)\right\}, \tag{B.5}$$

where $\mu^* = \{\mu_1^*, \ldots, \mu_M^*\}$ are the experts, and ξ and Ω are the mean and precision, respectively. The matrix Ω is also known as the potential or information matrix. In a GGM, if $\Omega_{ij} = 0$, then μ_i^* ard μ_j^* are conditionally independent given all other variables, i.e. there is no edge between μ_i^* ard μ_j^* in the graph.

GGMs use the common sparsity assumption, that is, there are only few edges in the network and thus the precision matrix is sparse. To this end, the graphical Lasso (GLasso) regression [59] is used to perform neighborhood selection for the network. It maximizes the log-likelihood subject to an element-wise $\mathcal{L}_1$ norm penalty on Ω. Precisely, the objective function is

$$\widehat{\Omega}_\lambda = \arg\min_{\Omega} \left(-\mathcal{L}(\Omega;S) + \lambda \|\Omega\|_1\right) \tag{B.6}$$

where $\mathcal{L}(\Omega;S) = \log|\Omega| - trace(S\,\Omega)$ is the Gaussian log-likelihood and S is the sample covariance.

B.4.3 GGM-Based Aggregation using EM Algorithm

The main input of the GLasso method is the sample covariance of our observations. Since the targeted expert y^* is unobserved, one row (column) in S, related to y^* is unknown. Let $S_{\mu^*\mu^*}$ is a known $M \times M$ matrix of the sample covariance of the observed variables μ^*, $S_{y^*\mu^*}$ is an unknown $1 \times M$ vector that shows the sample covariance between latent and observed expert, and $S_{y^*y^*}$ is the internal potential of a latent expert. To use the GLasso, it is needed to estimate unknown partitions of S, i.e. $S_{y^*\mu^*}$ and $S_{y^*y^*}$. Here, we explain how the expected-maximization algorithm can help us.

E-Step : The E-step Calculates $Q(\Omega \mid \Omega^{(t)})$, the expected value of the penalized negative log-likelihood function with respect to the conditional distribution of y^* given μ^* under the current estimate $\Omega^{(t)}$ of Ω:

$$\begin{aligned} Q(\Omega \mid \Omega^{(t)}) &= \mathrm{E}_{y^*|\mu^*,\Omega^{(t)}}\left[-\mathcal{L}(\Omega;S) + \lambda \|\Omega\|_1\right] \\ &= E_{y^*|\mu^*,\Omega^{(t)}}\{-\log|\Omega| + trace(S\,\Omega) + \lambda \|\Omega\|_1\} \\ &= -\log|\Omega| + trace\{E_{y^*|\mu^*,\Omega^{(t)}}(S)\Omega\} + \lambda \|\Omega\|_1 . \end{aligned}$$

Let $\Sigma^{(t)} = (\Omega^{(t)})^{-1}$, the conditional distribution of y^* given μ^* under the current estimate $\Omega^{(t)}$ follows

$$N\left(\Sigma^{(t)}_{y^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}\mu^*, \Sigma^{(t)}_{y^*} - \Sigma^{(t)}_{y^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}\Sigma^{(t)}_{\mu^*y^*}\right).$$

Therefore,

$$E_{y^*|\mu^*,\Omega^{(t)}}(S_{\mu^*y^*}) = S_{\mu^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}\Sigma^{(t)}_{\mu^*y^*}, \tag{B.7}$$

and

$$\begin{aligned} E_{y^*|\mu^*,\Omega^{(t)}}(S_{y^*y^*}) = \Sigma^{(t)}_{y^*y^*} - \Sigma^{(t)}_{y^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}\Sigma^{(t)}_{\mu^*y^*} + \\ \Sigma^{(t)}_{y^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}S_{\mu^*\mu^*}(\Sigma^{(t)}_{\mu^*\mu^*})^{-1}\Sigma^{(t)}_{\mu^*y^*}. \end{aligned} \tag{B.8}$$

The output of the E-step, $S^{(t)} = E_{y^*|\mu^*,\Omega^{(t)}}(S)$, can be used as an input for GLasso in M-step.

M-Step : This step returns the updated precision matrix $\Omega^{(t+1)}$ that maximize $Q(\Omega \mid \Omega^{(t)})$ over all $(M+1)\times(M+1)$ positive-definite matrices Ω. It is a *GLasso* problem and is equivalent to this minimization problem:

$$\arg\min_{\Omega}\left(-\log|\Omega| + trace\{S^{(t)}\Omega\} + \lambda \|\Omega\|_1\right). \tag{B.9}$$

The whole procedure of the proposed ensemble, EMGGM, is summarized in Algorithm 2.

Algorithm 2 GGM-Based Experts Aggregation (EMGGM)

Input: μ^*, λ, R (number of iterations)

1: Initialize y^*
2: Calculate sample covariance $S^{(0)}$ of (y^*, μ^*)
3: Estimate the initial parameter $\Omega^{(0)}$ using Equation (B.6)
4: $t \leftarrow 1$
5: **while** $t \leq R$ **do**
6: Estimate $E_{y^*|\mu^*,\Omega^{(t)}}(S_{\mu^* y^*})$ using Equation (B.7)
7: Estimate $E_{y^*|\mu^*,\Omega^{(t)}}(S_{y^* y^*})$ using Equation (B.8)
8: Update the sample covariance as $S^{(t)} = E_{y^*|\mu^*,\Omega^{(t)}}(S)$
9: Update the precision matrix $\Omega^{(t)}$ using Equation (B.9)
10: $\Sigma^{(t)} \leftarrow (\Omega^{(t)})^{-1}$
11: $t \leftarrow t+1$
12: **end while**
13: Estimate the aggregated prediction y_A^* using Equation (B.4)
14: **return** y_A^*

B.4.4 Discussion

The proposed ensemble is capable to aggregate local experts considering their dependencies and its computational and storage costs are much smaller than NPAE which uses dependent experts. Although the conventional GLasso [59] for network learning is almost a costly method, there are newer faster methods to learn a GGM that can be used instead of the GLasso in (B.6) and (B.9), see [63, 66, 64]. Besides, the normality assumption for joint distribution is not a restrictive assumption. In practice, we can relax this assumption and consider random variables without resorting to multi-dimensional

Gaussian distribution[69, 70, 74]. This gives the result that the proposed strategy can be considered as a general ensemble model, and not only for the local approximation GPs.

B.5 Experiments

In this section, we evaluate the quality of the aggregated estimator. We consider the prediction quality of our proposed method and other related SOTA models by using both simulated and real-world data sets. The quality of predictions is evaluated in two ways: we use the conventional mean absolute error (MAE) and the root mean squared error (RMSE). We use the standard squared exponential kernel with automatic relevance determination and a Gaussian likelihood. The K-means method is used for partitioning to divide the training data into some partitions.

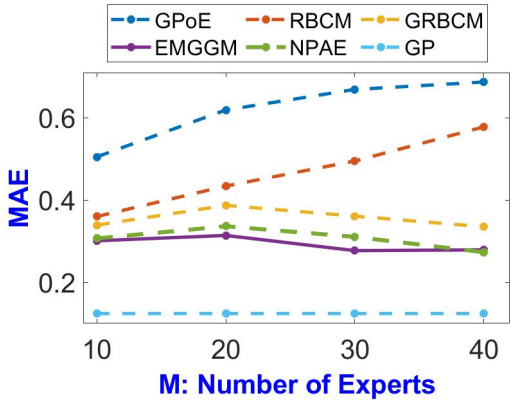
B.5.1 Synthetic Example

We use the simulated data of a one-dimensional analytical function [38, 3],

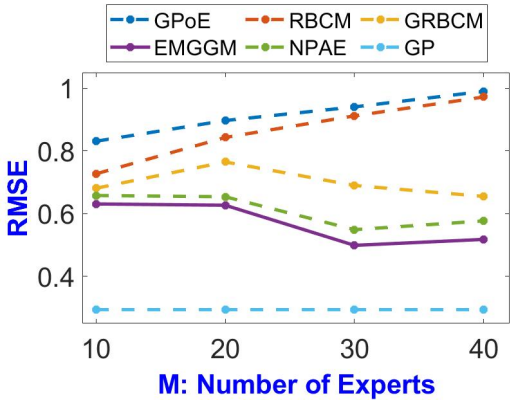
$$f(x) = 5x^2 sin(12x) + (x^3 - 0.5)sin(3x - 0.5) + 4cos(2x) + \varepsilon, \tag{B.10}$$

where $\varepsilon \sim \mathcal{N}(0, (0.2)^2)$. We generate $n = 10^4$ training points in $[0,1]$, and $n_t = 10^3$ test points in $[-0.2, 1.2]$. The data is normalized to zero mean and unit variance. We vary the number of experts, $M = \{10, 20, 30, 40\}$, to evaluate different partition sizes. The prediction quality of the proposed ensemble is compared with the other baselines: GPoE [37], RBCM [1], GRBCM [38], NPAE [44], and the full GP.

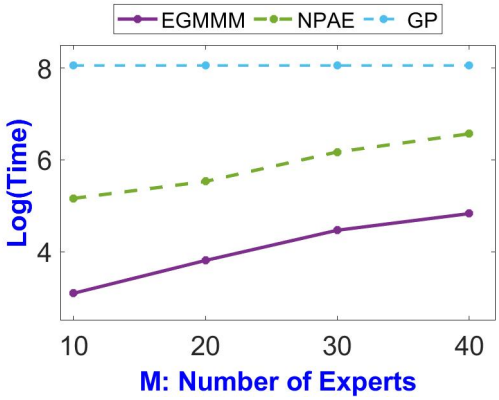
Figures B.2a and B.2b depict the prediction quality of different baselines. The ensemble methods that use dependency between experts, i.e. EMGGM and NPAE, outperform the CI-based baselines. However, the proposed method has slightly better predictions than NPAE. Figure B.2c presents the computation time of the ensembles that use dependency between experts. Remarkably, EMGGM provides predictions in just a fraction of NPAE's running time.



(a) Mean Absolute Error(MAE)



(b) Root Mean Squared Error (RMSE)



(c) Log of Prediction Time (in Second)

Figure B.2: **Prediction quality** of DGPs with respect for different number of experts in the simulated data.

B.5.2 Realistic Datasets

In this section, we use a multivariate realistic dataset, *Parkinson*. It is a 20D dataset with 5875 instances which is divided into 5000 training examples and 875 test examples. We consider the root mean squared error of the baselines' prediction for $M = 7, 10, 15$ experts. Because of the computational cost, we exclude the NPAE and the full GP methods, and instead, we use linear regression in this experiment. Table B.1 shows the final results and confirms the proposed method returns more accurate predictions than the other baselines.

Table B.1: Prediction quality measures of DGP methods on *Parkinson* dataset.

	GPoE	RBCM	GRBCM	Lin-Reg	**EMGMM**
M=7	0.0137	0.0132	0.0135	0.0168	**0.0131**
M=10	0.0152	0.0137	0.0139	0.0168	**0.0135**
M=15	0.0157	0.0140	0.0141	0.0168	**0.0136**

B.6 Conclusion

In this work, we have proposed a novel ensemble method, EMGGM, for distributed GPs which aggregate dependent local experts' predictions using GGMs. Our proposed approach uses undirected graphical models and an EM algorithm to estimate the final predictions. Through empirical analyses, we illustrated the superiority of EMGGM over existing SOTA aggregation methods. Finally, future works can develop this aggregation strategy to derive the full predictive distribution.

B.7 Appendix: Challenges and Further Discussions

B.7.1 Computational cost of EMGGM and NPAE

Both EMGGM and NPAE use dependent experts. However, there are two major differences between them. First, NPAE needs all training and test data points during aggregation. Let $\Gamma_i = k_{i*}^T (K_i + \sigma^2 I)^{-1}$. For a test point x^*, the pointwise covariance between experts i and j in NPAE, $K(x^*)_{ij}$, can be extended using (B.2) as

$$K(x^*)_{ij} = cov(\mu_i^*(x^*), \mu^*(x^*)_j) = Cov(\Gamma_i y_i, \Gamma_j y_j) = \Gamma_i Cov(y_i, y_j)\Gamma_j^T = \Gamma_i k(x_i, x_j)\Gamma_j^T .$$

Therefore, all auto-covariance $k(x_i, x_i)$ and cross-covariance $k(x_i, x_j)$ matrices are required for NPAE aggregation which raises the storage costs.

Second, both aggregation methods have a $\mathcal{O}(M^3)$ calculation in each iteration, the inverse of $M \times M$ matrix in NPAE and GLasso in the proposed method. NPAE should do this costly calculation at each test point and therefore it is not efficient for large data sets. However, the proposed model can converge after a small number of iterations. When $R \ll n_t$, the proposed method is much faster than NPAE. Although the conventional

GLasso for network learning is a costly method $\mathcal{O}(M^3)$, there are newer faster methods to learn a GGM that can be used instead of the GLasso, see [63, 66, 64]. For instance, the FST model [64] reduces the computational complexity of sparse Gaussian Graphical Model to a much lower order of magnitude $(\mathcal{O}(M^2))$.

B.7.2 Gaussian Assumption

The normality assumption for joint distribution is not a restrictive assumption. In practice, we can relax this assumption and consider random variables without resorting to multi-dimensional Gaussian distribution. As a semiparametric generalization for continuous variables, authors in [68, 69] introduced the nonparanormal graphical model where it is assumed that the variables follow a Gaussian graphical model only after some unknown smooth monotone transformations on each of them. [70] considered Bayesian inference in nonparanormal graphical models by putting priors on the unknown transformations through a random series based on B-splines.

On the other hand, nonparametric methods can be used for functional graphical models. Authors in [73] and [74] exerted additive conditional independence and functional principal components to learn a graphical model when observations on vertices are functions. This gives the result that the proposed strategy can be considered as a general ensemble model, and not only for the local approximation GPs.

B.7.3 Latent Variable GGMs

Latent variable GGMs (LVGGMs) are used to estimate the distribution of the observed variables with respect to some latent variables. GGMs with latent variables have been widely considered over the past decade. Authors in [65] proposed *Low-Rank Plus Sparse Decomposition* (LR+SD), a regularized maximum likelihood approach to estimate Ω via convex optimization. The precision matrix in LR+SD contains two terms: sparse structure Ω_{μ^*} and the low-rank terms $L^* = \Omega_{\mu^* y^*}\Omega^{-1}_{y^* y^*}\Omega_{y^* \mu^*}$. The precision matrix in this form is $\Omega = \Omega_{\mu^*\mu^*} - L^*$. The log-likelihood can be expressed in terms of the $S_{\mu^*\mu^*}$, $\Omega_{\mu^*\mu^*}$, and L^*:

$$\mathcal{L}(\Omega_{\mu^*\mu^*}, L^*; S_{\mu^*\mu^*}) = log|\left(\Omega_{\mu^*\mu^*} - L^*\right)| - trace\left(S_{\mu^*\mu^*}\left(\Omega_{\mu^*\mu^*} - L^*\right)\right). \tag{B.11}$$

Essentially, it is a misspecified optimization problem because the precision matrix is the sum of two matrices. However, if $\Omega_{\mu^*\mu^*}$ is sparse and there are few latent variables, it is possible to decompose the precision matrix into its summands [71, 65]:

$$\left(\hat{\Omega}_{\mu^*\mu^*}, \hat{L}^*\right) = \underset{\Omega_{\mu^*\mu^*}, L^* \in \mathcal{R}^{M\times M}}{\arg\min} - \mathcal{L}(\Omega_{\mu^*\mu^*}, L^*; S_{\mu^*\mu^*}) + \lambda\left(\gamma\left\|\Omega_{\mu^*\mu^*}\right\|_1 + \|L^*\|_*\right) \quad \text{(B.12)}$$

$$\text{such that} \quad \Omega_{\mu^*\mu^*} - L^* \succ 0,\ L^* \succeq 0 \quad \text{(B.13)}$$

such that $\Omega_{\mu^*} - L^* \succ 0$, $L^* \succeq 0$. Here, $\lambda > 0$ and $\gamma > 0$ are tuning parameters for sparsity and low rankness, and $\|L^*\|_*$ denotes the nuclear norm of L^* (i.e. the sum of its singular values). To speeding up the LR+SD model, [66] proposed a non-convex optimization model and showed that it is orders of magnitude faster than the convex relaxation-based methods. Author in [72] proposed a direct approach via Expectation-Maximization algorithm which converts LR+SD model to a conventional GGM. Here, we modified this approach and proposed EMGGM.

However, LR+SD model has been developed to estimate the marginal distribution of observed variables, i.e. $p(\mu^*) = \int p(\mu^*, y^*)dy^*$, while the desired predictive distribution is the conditional distribution of y^* given local experts' predictions $p(y^*|\mu^*)$. Hence, further work could consider the modified form of the log-likelihood in (B.11) and the convex optimization in (B.12) to estimate the aggregate estimator y_A^* in (B.4) via a convex or non-convex optimization problem.

Appendix C

Model Selection in Distributed Gaussian Processes: A Markov Random Fields Approach

This chapter includes the following publications:

- Hamed Jalali, Martin Pawelczyk and Gjergji Kasneci. Model Selection in Local Approximation Gaussian Processes: A Markov Random Fields Approach. In 2021 IEEE International Conference on Big Data (Big Data), 2021. doi: 10.1109/BigData52589.2021.9672077.

  The Appendix section C.7 is based on this article:

- Hamed Jalali, Martin Pawelczyk and Gjergji Kasneci. Gaussian experts selection using graphical models. Published on ArXiv, 2021. doi: 10.48550/arXiv.2102.01496.`https://arxiv.org/abs/2102.01496`

C.1 Abstract

Local approximations are popular methods to scale Gaussian processes (GPs) to big data. Local approximations reduce time complexity by dividing the original dataset into subsets and training a local expert on each subset. Aggregating the experts' prediction is done assuming either conditional dependence or independence between the experts. Imposing the *conditional independence assumption* (CI) between the experts renders the aggregation of different expert predictions time efficient at the cost of poor uncertainty quantification. On the other hand, modeling dependent experts can provide precise predictions and uncertainty quantification at the expense of impractically high computational costs. By eliminating weak experts via a theory-guided expert selection step, we substantially reduce the computational cost of aggregating dependent experts while ensuring calibrated uncertainty quantification. We leverage techniques from the literature on undirected graphical models, using sparse precision matrices that encode conditional dependencies between experts to select the most important experts. Moreover, our approach also provides a solution to the poor uncertainty quantification in CI-based models.

C.2 Introduction

Gaussian processes [10] are interpretable and powerful statistical methods to deal with uncertainty in prediction problems. These non-parametric methods apply Bayes' theorem to discover complex linear and non-linear structures without the need for restrictive assumptions on the model. Due to their capabilities, they are widely used in practical cases, e.g. optimization [11], data visualization, and manifold learning [12], reinforcement learning [15], multitask learning [17], online streaming models [19, 20] and time series analysis [21, 22].

The main limitation of these models is their computational cost on data sets with large numbers of observations. For a data set of size n, the time and space complexities during training are $\mathcal{O}(n^3)$ and $\mathcal{O}(n^2)$, respectively. These high complexities are due to the inversion of the $n \times n$ kernel matrix and determinant computation. On the test set, the predictions require additional time and space complexities of $\mathcal{O}(n \log n)$. This issue restricts GPs to relatively small training data sets of the order of $\mathcal{O}(10^4)$.

A popular method to scale GPs is the common divide-and-conquer approach. The idea is to partition the training data, perform local inference for each part separately, and combine the results to obtain a global posterior approximation [39, 41, 37, 1]. This distributed training approach generally imposes a *conditional independence assumption* (CIA) between partitions, which allows factorizing the global posterior distribution as a product of local distributions. Although this assumption reduces the computational cost of GPs, it leads to statistical inconsistency [46, 45]. That is, in the presence of infinite data, CIA based posterior approximations do not converge to the full GP posterior. Moreover, these approaches only provide pointwise variance and confidence intervals instead of a full GP predictive distribution. Solutions, that cope with the statistical consistency problem, suffer from extra computational costs [44, 38]. Hence, these solutions are impractical for large data sets.

Unlike the divide-and-conquer approach, which uses the full training set, another strategy tries to use a small part of the training data. For instance, the sparse approximation approach employs a subset of the data (called inducing points) and *Nyström* approximations to estimate the posterior distribution [30, 31, 84, 85]. Although this line of work provides a full probabilistic model, its capacity is restricted by the number of inducing points [33, 34], and therefore it is not appropriate for large and high dimensional data sets.

In this paper, we propose a new expert selection approach for distributed GPs which improves the prediction quality in both dependent and independent experts aggregation methods. The full experts set is divided into two subsets, *important* and *unimportant* experts. The importance of an expert is measured according to his interactions with other experts. Particularly, we use a Markov random field to construct a sparse undirected graph of experts that provides information about the overall interactions. Experts with more significant connections are treated as important experts and are used for the final aggregation. Furthermore, relative to consistent aggregation methods that use dependency information, our approach also yields consistent predictions and substantially speeds up the running time, while maintaining competitive prediction performance.

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

C.3 Background and Problem Set-up

C.3.1 Gaussian Process

Let us consider the regression problem $y = f(x) + \varepsilon$, where $x \in \mathbb{R}^D$ and $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ which corresponds to the Gaussian likelihood $p(y|f) = \mathcal{N}(f, \sigma^2 I)$. The objective is to learn the latent function f from a training set $\mathcal{D} = \{X, y\}$ of size n. The Gaussian process regression is a collection of random variables of which any finite subset has a joint Gaussian distribution. The GP then describes a prior distribution over the latent functions as $f \sim GP(m(x), k(x,x'))$, where $m(x)$ is a mean function and $k(x,x')$ is the covariance function (kernel) with hyperparameters ψ. The prior mean is often assumed to be zero, and the kernel is the well-known squared exponential (SE) covariance function equipped with automatic relevance determination (ARD),

$$k(x,x') = \sigma_f^2 \exp\left(-\frac{1}{2}\sum_{d=1}^{D}\frac{(x_d - x'_d)^2}{\mathcal{L}_d}\right),$$

where σ_f^2 is the signal variance, and $\mathcal{L}_d$ is an input length-scale along the d-th dimension, and $\psi = \{\sigma_f^2, \mathcal{L}_1, \ldots, \mathcal{L}_D\}$. To train the GP, the hyperparameters $\theta = \{\sigma^2, \psi\}$ should be determined such that they maximise the log-marginal likelihood,

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

where $\mathcal{C} = K + \sigma^2 I$ and $K = k(X,X)$. According to (C.1), the training step scales as $\mathcal{O}(n^3)$ because it is affected by the inversion and determinant of the $n \times n$ matrix $\mathcal{C}$. Therefore, for large data sets, GP training is a time-consuming task and imposes limitations on the scalability of GPs.

C.3.2 Local Gaussian Process Experts

The local approximation Gaussian process uses the fact that the computations of the standard GP can be distributed among individual computing units. To do that, one divides the full training data set $\mathcal{D}$ into M partitions (called experts) and trains standard GPs on these partitions. Let $\mathcal{D}' = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ be the partitions, and X_i and y_i be the input and

output of partition $\mathcal{D}_i$. For a test set X^*, the predictive distribution of the i-th expert $\mathcal{M}_i$ is $p_i(y^*|\mathcal{D}_i, X^*) \sim \mathcal{N}(\mu_i^*, \Sigma_i^*)$, where its mean and covariance are respectively:

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

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

where $K_i = k(X_i, X_i)$, $k_{i*} = k(X_i, X^*)$, and $k_{**} = k(X^*, X^*)$. Aggregation is based on the assumption that the experts are conditionally independent leading to predictive distribution of the form

$$p(y^*|\mathcal{D}, X^*) \propto \prod_{i=1}^{M} p_i^{\beta_i}(y^*|\mathcal{D}_i, X^*)$$

where $\beta = \{\beta_1, \ldots, \beta_M\}$ controls the expert importance.

The most popular aggregation methods are product of experts (PoE) [41] and Bayesian committee machine (BCM) [42]. Generalised product of experts (GPoE) [37] and robust Bayesian committee machine (RBCM) [1] are more recent modified versions of PoE and BCM, which add the experts' weight quantifying each expert's contribution to the predictive distribution. Generalized robust Bayesian committee machine (GRBCM) [38] is the most recent model, which uses a random subset $\mathcal{D}_b$ drawn from the entire training data and treats it as a global communication expert to augment each partition $\mathcal{D}_i$.

To choose the weights β_i several heuristics have been put forward. The authors of [37] suggested the difference in differential entropy between the prior and posterior distribution of each expert, i.e. $\beta_i = \frac{1}{2}(\log \Sigma^{**} - \log \Sigma_i^*)$ where the $(\Sigma^{**})^{-1}$ is the prior precision of $p(y^*)$. It has been used widely in GPoE, RBCM, and GRBCM. This varying weight may produce undesirable prediction error for GPoE. To fix this issue, [1] suggested to choose simple uniform weights $\beta_i = \frac{1}{M}$, which provides better predictions.

C.3.3 Dependencies Between Experts' Predictions

Assume the Gaussian experts $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ have been trained on different partitions and the first and second moments of their local posterior distributions have been defined in (C.2) and (C.3). Let $\mu^*(x^*) = [\mu_1^*(x^*), \ldots, \mu_M^*(x^*)]^T$ be an $M \times 1$ vector that contains the centered predictions of M experts for a given test point $x^* \in X^*$. Assuming that y_i in (C.2) has not yet been observed, the authors of [44] considered the prediction

mean $\mu_i^*(x^*)$ as a *random variable*. This allows us to consider correlations between the experts' predictions and latent variable y^*, $Cov(\mu_i^*, y^*)$, and also leverage internal correlations between experts, $Cov(\mu_i^*, \mu_j^*)$ where $i, j = 1, \ldots, M$. According to (C.2), the local experts are linear estimators: $\mu_i^* = \Gamma_i y_i$, where $\Gamma_i = k_{i*}^T (K_i + \sigma^2 I)^{-1}$. Using this result we can find the analytical expression for both covariances:

$$\begin{aligned} k_A(x^*)_i &= Cov(\mu_i^*(x^*), y^*(x^*)) = Cov(\Gamma_i y_i, y^*(x^*)) \\ &= \Gamma_i Cov(y_i, y^*(x^*)) = \Gamma_i k(x_i, x^*), \end{aligned} \tag{C.4}$$

$$\begin{aligned} K_A(x^*)_{ij} &= Cov(\mu_i^*(x^*), \mu_j^*(x^*)) = Cov(\Gamma_i y_i, \Gamma_j y_j) \\ &= \Gamma_i Cov(y_i, y_j) \Gamma_j^T = \Gamma_i k(x_i, x_j) \Gamma_j^T. \end{aligned} \tag{C.5}$$

C.3.4 Nested Point-wise Aggregation of Experts (NPAE)

Assume the means (predictions) of the local predictive distributions are random variables. Then $k_A(x^*) = Cov(\mu^*(x^*), y^*(x^*))$ and $K_A(x^*) = Cov(\mu^*(x^*), \mu^*(x^*))$ are the point-wise covariances. For *each* test point x^*, $k_A(x^*)$ is a $M \times 1$ vector and $K_A(x^*)$ is a $M \times M$ matrix and according to (C.4) and (C.5) their elements are defined as $k_A(x^*)_i = \Gamma_i k(X_i, x^*)$ and $K_A(x^*)_{ij} = \Gamma_i k(X_i, X_j) \Gamma_j^T$, where $\Gamma_i = k(X_i, x^*)^T (K_i + \sigma^2 I)^{-1}$ and $i, j = 1, \ldots, M$. The task is to aggregate variables $\mu_i^*(x^*), i = 1, \ldots, M$ into a unique predictor $y_A^*(x^*)$ of $y^*(x^*)$.

As a consequence from the choice of the prior, the joint distribution of random variables $(y^*, \mu_1^*, \ldots, \mu_M^*)$ is a multivariate normal distribution because any vector of linear combinations of observation is itself a Gaussian vector. This fact is used to define the aggregated predictor, but it also implies that the experts' predictions $(\mu_1^*, \ldots, \mu_M^*)$ follow a multivariate Gaussian. Employing properties of conditional Gaussian distributions for the centered random vector $(y^*(x^*), \mu^*(x^*))$ allows for the following aggregation:

Definition 3 (**Aggregated predictor**) [44])**.** For the test point x^* and sub-model predictions $\mu_1^*(x^*), \ldots, \mu_M^*(x^*)$, the aggregated predictor is defined as

$$y_A^*(x^*) = k_A(x^*)^T K_A(x^*)^{-1} \mu^*(x^*). \tag{C.6}$$

In [44, 46] the authors showed that this linear estimator is the *best linear unbiased*

predictor (BLUP) where the invertibility condition on $K_A(x^*)$ can be avoided by using matrices pseudo-inverses. This aggregated predictor has Gaussian distribution and its moments can easily be calculated using $k_A(x^*)$ and $K_A(x^*)$. This method is known as the nested pointwise aggregation of experts (NPAE) and provides consistent predictions.

C.3.5 Consistency

The conventional PoE, GPoE, BCM, and RBCM produce inconsistent predictions and generally do not converge to the predictive distribution of the standard GP when $n \to \infty$ [43, 46]. Using normalized equal weights, GPoE asymptotically converges to the full GP distribution, however, it is too conservative [45]. Integrating a global partition into RBCM, [38] showed that GRBCM can provide consistent results. However, it still uses the conditional independence assumption between non-global experts, which sometimes yields poor results.

To deal with the inconsistency issue, NPAE considers dependencies between individual experts' predictions and uses the property of conditional Gaussian distributions to find the predictive distribution of y^*. Although it theoretically provides consistent predictions, its aggregation step suffers from high time complexity because it needs to determine the inverse of a $M \times M$ covariance matrix between experts at *each* test point x^*, i.e. $\mathcal{O}(n_t M^3)$, where n_t is the number of available test points. This leads to impractically long running times for many partitions and large test sets.

To mitigate the drawbacks of the NPAE and CI-based aggregations, we will implicitly incorporate an expert selection step by using expert interaction strengths. The influences of the weak experts' prediction on prediction quality and computational cost can be eliminated by excluding them in both CI-based and NPAE methods. In the next section, we present our new approach. First, we explain the proposed pruning step which is based on undirected graphical models. Second, we propose a new aggregation method with this pruning approach and consider its asymptotic properties.

C.4 Expert Selection with Gaussian Graphical Models

At the heart of our work is the following ingredient. We assume that the local experts can be divided into 2 sets: important and unimportant experts. Aggregation based on important experts provides consistent and competitive predictions with significantly reduced computational time. Instead of expert selection, state-of-the-art (SOTA) methods use expert weighting, i.e. they assign weights to each expert, but they do not provide a systematic way to determine the most important experts. In NPAE, the point-wise weights of local expert predictions are defined as $k_A(x^*)^T K_A(x^*)^{-1}$. In section C.4.1, we propose a novel approach that estimates the set of important experts based on the experts' interactions in an undirected graph.

C.4.1 Gaussian Graphical Models for Correlated Gaussian Experts

Since the experts' mean predictors are random variables with joint Gaussian distribution, we can study them using the machinery of sparse Gaussian graphical models (GGM). In GGMs, the assumption is that there exist few interactions, i.e., edges, between variables; a penalty parameter controls the network's sparsity. It will allow us to obtain a set of important experts $\tilde{\mathcal{M}} = \{\tilde{\mathcal{M}}_1, \ldots, \tilde{\mathcal{M}}_p\}$, where $p < M$.

Definition 4 (Important and Unimportant Experts). Let $\mathcal{M} = \{\mathcal{M}_1, \ldots, \mathcal{M}_M\}$ be the set of GP experts and let $\mathcal{G}(\mathcal{V}, \mathcal{E})$ be their Gaussian graph, where $\mathcal{V} = \mathcal{M}$ and the edges $\mathcal{E}$ are the interactions between experts. Then, $\tilde{\mathcal{M}}_\alpha$ is the set of *important* experts, if it contains $\alpha \times 100$ of the most connected experts in $\mathcal{G}$, where $\alpha \in [0, 1]$. Its complement, $\tilde{\mathcal{M}}_\alpha^c$, contains the remaining *unimportant* experts and $\mathcal{M} = \tilde{\mathcal{M}}_\alpha \cup \tilde{\mathcal{M}}_\alpha^c$. The selection rate α controls how many experts should be assigned to $\tilde{\mathcal{M}}_\alpha$, i.e. $|\tilde{\mathcal{M}}_\alpha| = M_\alpha = \lceil \alpha M \rceil$, where $|\cdot|$ denotes the cardinality.

Gaussian Graphical Models. Undirected graphical models (known as pairwise Markov random fields (MRF)) provide a framework for encoding joint distributions over large numbers of interacting random variables. This framework uses a matrix of parameters to encode the graph structure. In other words, it encodes the edges as parameters: if there is a connection between two nodes, then there is a non-zero parameter indicating that the edge between nodes is present in the graph. In Gaussian graphical models [52, 53, 54]

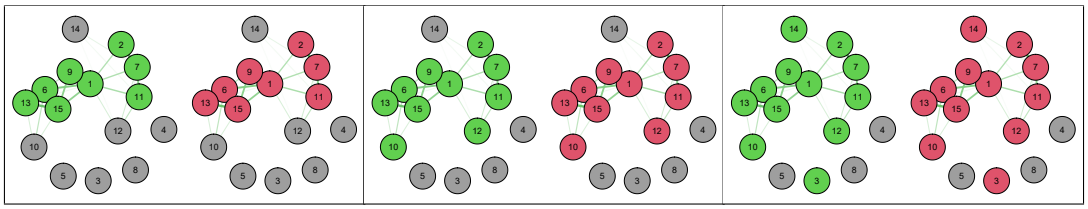
the nodes are continuous Gaussian random variables. The main assumption underlying GGMs is that all variables follow a multivariate Gaussian distribution,

$$p(\mu^*|\xi,\Psi) = \frac{1}{(2\pi)^{M/2}|\Psi|^{1/2}} \exp\left\{-\frac{1}{2}(\mu^* - \xi)^T \Psi^{-1}(\mu^* - \xi)\right\}, \tag{C.7}$$

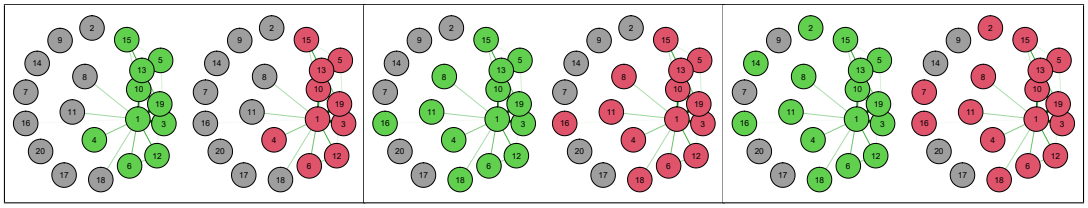
where $\mu^* = \{\mu_1^*, \ldots, \mu_M^*\}$ are the variables (nodes), M is the number of variables, and ξ and Ψ are the mean vector and the covariance matrix, respectively. We can rewrite (C.7) using the precision matrix Ω,

$$p(\mu^*|\eta,\Omega) = \frac{|\Omega|^{1/2}}{(2\pi)^{M/2}} \exp\left\{-\frac{1}{2}(\mu^* - \xi)^T \Omega(\mu^* - \xi)\right\} \propto \exp\left\{-\frac{1}{2}\mu^{*T}\Omega\mu^* + \eta^T\mu^*\right\}, \tag{C.8}$$

where $\Omega = \Psi^{-1}$ and $\eta = \Omega\xi$. The matrix Ω is also known as the potential or information matrix. Without loss of generality, let $\xi = 0$, then the distribution of a GGM shows the potentials defined on each node i as $\exp\{-\Omega_{ii}(\mu_i^*)^2\}$ and on each edge (i,j) as $\exp\{-\Omega_{ij}\mu_i^*\mu_j^*\}$. In the correlation network, expressed through (C.7), Ψ encodes in-



(a) $M = 15,\ \alpha = 50\%$ (b) $M = 15,\ \alpha = 70\%$ (c) $M = 15,\ \alpha = 80\%$



(d) $M = 20,\ \alpha = 50\%$ (e) $M = 20,\ \alpha = 70\%$ (f) $M = 20,\ \alpha = 80\%$

Figure C.1: **Ablation experiment 1.** Expert selection for synthetic data from (C.13) with varying expert set strength α, $M = \{15, 20\}$, and $\lambda = 0.1$. The green nodes reveal the $\alpha\%$ of the best experts w.r.t. their individual MSE errors while the red nodes are the most important experts according to Definition 5.

dependences: if $\Psi_{ij} = 0$, then μ_i^* and μ_j^* are assumed to be *independent*. On the other

hand, in the GGM in (C.8) we have: if $\Omega_{ij} = 0$, then μ_i^* are μ_j^* are *conditionally independent* given all other variables, i.e. there is no edge connecting μ_i^* and μ_j^* in the graph.[1]Besides, $\Omega_{ij} \approx 0$ means there is a weak interaction between μ_i^* and μ_j^*.

C.4.2 Network Learning for the Aggregated Posterior

We use the GGM as follows: the locally trained experts represent nodes and network learning involves computing the precision matrix Ω that provides the interactions (edges) between the experts. To keep inference in GGMs tractable, one imposes a *sparsity assumption*: there exist few edges in the network, and thus Ω is a sparse matrix. This assumption is empirically meaningful in an experts' network because the strong interaction of one expert can be typically limited to only a few other experts. In fact, since the local predictions of this expert are close to the local predictions of its adjacent experts, it has stronger interactions (or similarities) with them, see [55] for more information about similarity and dissimilarity.

The literature has suggested several inference algorithms to recover the edge set. Lasso regression [56, 57] can be used to perform neighborhood selection to recover the network. [57] and [58] proved that, under some regularity conditions, Lasso asymptotically recovers the correct and relevant subset of edges. The authors of [59] proposed a more efficient inference algorithm, the graphical Lasso (GLasso), which adopts a maximum likelihood approach subject to an l_1 penalty on the coefficients of the precision matrix. Its inference is fast and has been improved in subsequent works [86, 60, 61, 62]. The authors in [3] used the GLasso to detect dependencies between Gaussian experts and define clusters of strongly dependent experts.

The GLasso objective is formalized as follows. Let S be the sample covariance of local predictions μ^*. Then, the Gaussian log likelihood of the precision matrix Ω is equal to $\log|\Omega| - trace(S\Omega)$. The Graphical Lasso maximises this likelihood subject to an element-wise l_1-norm penalty on Ω. More precisely, the objective function is,

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

[1]We would like to emphasize that this does not hold in general, but here it does hold since we assumed that all variables are jointly Gaussian distributed.

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

Definition 5 (λ**-related Expert Importance**)**.** The λ-related importance of expert $\mathcal{M}_i$ based on its interactions is defined as $\mathcal{I}_i = \sum_{j=1, j\neq i}^{M} |\widehat{\Omega}_{\lambda,ij}|$, resulting in the sorted importance set $\mathcal{I} = \{\mathcal{I}_{i_1}, \mathcal{I}_{i_2}, \ldots, \mathcal{I}_{i_M}\}$.

Figure C.1 depicts graphs based on 5×10^3 training points from (C.13) divided among 15 experts (333 training points for each expert) and 20 experts (250 training points for each expert) and 500 test points. Figures (a), (b), and (c) present the graphs of 15 experts while (d), (e), and (f) show the related graphs of 20 experts. First, after training the local experts the prediction quality of each expert is measured based on their mean squared error (MSE). Then the experts that are among the $\alpha\%$ of the best experts with the lowest MSE values are highlighted in green. To evaluate the proposed expert selection approach, the most important experts – based on their interactions with other experts – are depicted as red nodes. For Figure C.1 we have used the network inference method based on GLasso approximation [61] and Definition 5. It can provide a selection approach which tends to choose the best experts as the important experts.

C.4.3 Point-wise Aggregation with Expert Selection

Assume $\tilde{\mathcal{M}}_\alpha$ and $\tilde{\mathcal{M}}_\alpha^c$ are important and unimportant experts' sets as defined by Definitions 4 and 5. Further, let $\tilde{\mu}_\alpha^*$ represent the local predictions of the experts in $\tilde{\mathcal{M}}_\alpha$. We use $\tilde{k}_\alpha$ and $\tilde{K}_\alpha$ to denote $Cov(\tilde{\mu}_\alpha^*, y^*)$ and $Cov(\tilde{\mu}_\alpha^*, \tilde{\mu}_\alpha^*)$, respectively. Using Definition 4, the following proposition gives the predictive distribution of **NPAE***, our new NPAE estimator with expert selection.

Proposition 3 (**Predictive Distribution**)**.** Let X be a compact, nonempty subset of $\mathcal{R}^{n\times D}$, $\mu^*(x^*) = [\mu_1^*(x^*), \ldots, \mu_M^*(x^*)]^T$ be the sub-models' predictions at a test point x^*. We further assume that (i) $\lim_{n\to\infty} M = \infty$ and (ii) $\lim_{n\to\infty} m_0 = \infty$, where m_0 is the partition size. The vector $\left(\mu_1^*, \ldots, \mu_M^*, y^*\right)$ is a multivariate Gaussian distribution and we obtain the following results:

(I) The experts' importance-based aggregated estimator of $y^*(x^*)$ given $\mu^*(x^*)$ is

Gaussian with mean and variance given by:

$$\begin{aligned}\mathbb{E}[y^*(x^*)|\tilde{\mu}^*_\alpha(x^*)] &= y^*_\alpha(x^*)\\ &= \tilde{k}_\alpha(x^*)^T\tilde{K}_\alpha(x^*)^{-1}\tilde{\mu}^*_\alpha(x^*) \qquad \text{(C.9)}\\ \mathbb{V}[y^*(x^*)|\mu^*(x^*)] &= k(x^*,x^*)\\ &\quad -\tilde{k}_\alpha(x^*)^T\tilde{K}_\alpha(x^*)^{-1}\tilde{k}_\alpha(x^*). \qquad \text{(C.10)}\end{aligned}$$

(II) $y^*_\alpha(x^*)$ is a consistent estimator for $y^*(x^*)$, i.e.

$$\lim_{n\to\infty}\sup_{x^*\in X^*}\mathbb{E}\left[y^*(x^*)-y^*_\alpha(x^*)\right]^2\to 0. \qquad \text{(C.11)}$$

(III) $y^*_\alpha(x^*)$ provides an appropriate approximation for $y^*_A(x^*)$ in (C.6), i.e.

$$\lim_{n\to\infty} y^*_\alpha(x^*)\to y^*_A(x^*). \qquad \text{(C.12)}$$

Proof. The proof of part (I) and (II) is obvious using the property of conditional Gaussian distribution and NPAE method. We here proof part (III).

Our proof is based on disjoint K-means partitioning, and the proof for random partitioning is similar. For the proof, we need to show that the importance-based estimator $y^*_\alpha(x^*)$ is almost equivalent to $y^*_A(x^*)$ in (C.9). To do that, we partition both $k_A(x^*)$ and $K_A(x^*)^{-1}$ with respect to $\tilde{\mathcal{M}}_\alpha$ and $\tilde{\mathcal{M}}^c_\alpha$. Hence, we obtain $k_A(x^*)=[k_\alpha(x^*),k_c(x^*)]$, $\mu^*(x^*)=[\tilde{\mu}^*_\alpha(x^*),\mu^*_c(x^*)]$, and

$$K_A(x^*)^{-1}=\begin{bmatrix}K_\alpha(x^*)^{-1} & K_{\alpha c}(x^*)^{-1}\\ K^t_{\alpha c}(x^*)^{-1} & K_c(x^*)^{-1}\end{bmatrix}.$$

Due to the definition of $\tilde{\mathcal{M}}^c_\alpha$, we have $K_{\alpha c}(x^*)^{-1}\approx 0$. Note that $K_c(x^*)^{-1}$ is approximately a diagonal matrix, where the diagonal elements are pointwise variances of experts' predictions corresponding to (C.3). For the sake of better readability, we omit x^*

from our notation. Now, we have:

$$
\begin{aligned}
y_A^* &= k_A^T K_A^{-1} \mu^* = [k_\alpha, k_c] \begin{bmatrix} K_\alpha^{-1} & K_{\alpha c}^{-1} \\ K_{\alpha c}^{-t} & K_c^{-1} \end{bmatrix} [\tilde{\mu}_\alpha^*, \mu_c^*] \\
&= k_\alpha K_\alpha^{-1} \tilde{\mu}_\alpha^* + k_c K_{\alpha c}^{-t} \tilde{\mu}_\alpha^* + k_\alpha K_{\alpha c}^{-1} \mu_c^* + k_c K_c^{-1} \mu_c^* \\
&\approx k_\alpha K_\alpha^{-1} \tilde{\mu}_\alpha^* + k_c K_c^{-1} \mu_c^* .
\end{aligned}
$$

What remains to be shown is that when $n \to \infty$, $k_c K_c^{-1} \mu_c^* \approx 0$. Let $\mathcal{M}_{c_i}$ be the i-th partition of $\tilde{\mathcal{M}}_\alpha^c$. Since $\mathcal{M}_{c_i}$ does not have strong interactions with other experts given x^*, the off-diagonal values of $K_A(x^*)^{-1}$ related to this partition are small. Since the interactions between experts are estimated using their local predictions, small interactions between this partition and the others means its prediction $\mu_{c_i}^*$ is not close to the other experts' predictions, see (C.5) and the definition of Γ_i. According to the disjoint partitioning method, we can conclude that x^* is away from the training partition $\mathcal{M}_{c_i}$ (If x^* is close to the training partition $\mathcal{M}_{c_i}$, then its local predictions should be approximately close to the local predictions of its neighbor partitions and therefore, it should have strong interactions with them but it not true due to the definition of $\tilde{\mathcal{M}}_\alpha^c$, see Section C.4.5 for more details.).

It is clear that the further x^* is away from a partition, the higher the relative distance $r_{c_i} = \|x^* - x\|_{x \in \mathcal{M}_{c_i}}$ grows. Using assumption (ii) stated in proposition 3, we have $r_{c_i} \to \infty$ since $n \to \infty$. It says by increasing the sample size, the distance between a test point and the partitions away from that increases. Thus we have $\lim_{r_{c_i} \to \infty} \Sigma_i^* = k_{**}$, i.e. $k(x^*, X_{c_i}) \to 0$. Without loss of generality, we can assume that δ is an upper bound for the values of μ_c^*. Then

$$
(k_c K_c^{-1} \mu_c^*)_i \leq k(x^*, X_{c_i}) \left[k(x*, x^*) - k(x^*, X_{c_i})^T (K_{c_i} + \sigma^2 I)^{-1} k(x^*, X_{c_i}) \right]^{-1} \delta \to 0.
$$

Therefore, $y_A^*(x^*) \approx y_\alpha^*(x^*)$ when $n \to \infty$.

The quality of this approximation has been discussed in Section C.5 by both synthetic and real-world data sets. As a direct consequence, $y_\alpha^*(x^*)$ inherits the asymptotic properties of $y_A^*(x^*)$. A detailed proof, showing that $y_A^*(x^*)$ is statistically consistent can be found in [46] (see their proof of Proposition 2), which employs a triangular array of observations. □

Proposition 3 reveals that the aggregation of the most important experts in NPAE* leads to consistent predictions and the estimator in (C.9) provides an appropriate approximation for the NPAE estimator in (C.6). This expert selection offers two interesting implications. First, it suggests that the unimportant experts are irrelevant to obtain consistent predictions and thus the prediction quality can suffer from these weak experts. Second, using only the most important experts indicates that computation time can be substantially reduced. In a nutshell, Proposition 3 suggests that the described expert selection strategy leads to a consistent and efficient estimator.

Figure C.2 presents the effect of Definition 5 on the prediction quality for 5×10^3 training points from (C.13) divided among 20 experts (250 training points for each expert) and 500 test points. The quality of predictions is evaluated in three ways: the mean absolute error (MAE), the 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 the predictive distribution [10]. We can see a significantly better quality of the aggregation based on the most important experts compared to the least important ones in the NPAE* method.

C.4.4 CI-Based Models with Experts Selection

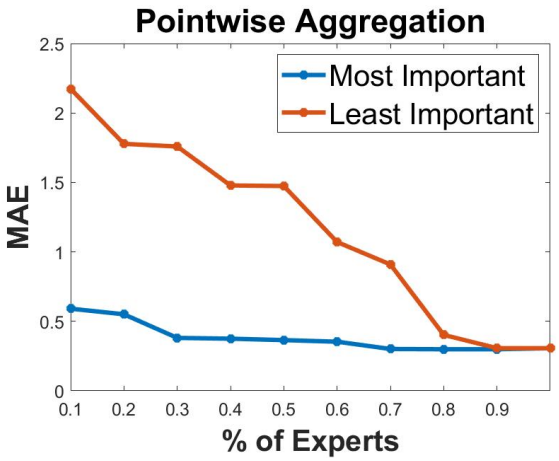
Our selection strategy can easily be extended to CI-based methods. Although it can not improve the asymptotic properties of CI-based models, it improves upon the prediction quality of the baseline models. Thus, the modifications consist of excluding the weak experts based on the procedure explained in Section C.4.1 which leads to (G)PoE* and (R)BCM* models.

Using the expert selection with GRBCM requires an extra step. The modification in the GRBCM model, say GRBCM*, consists of the following two tasks:

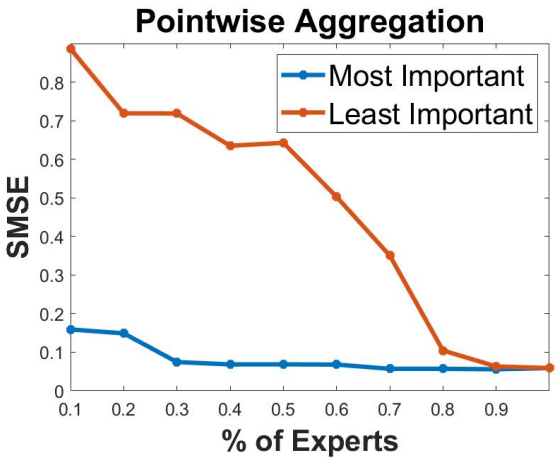
(i) excluding the weak experts based on the procedure explained in Section C.4.1

(ii) choosing $\mathcal{I}_{i_1}$ (i.e., the top-most expert) in the sorted importance set $\mathcal{I}$ as the global communication expert (see Definition 5).

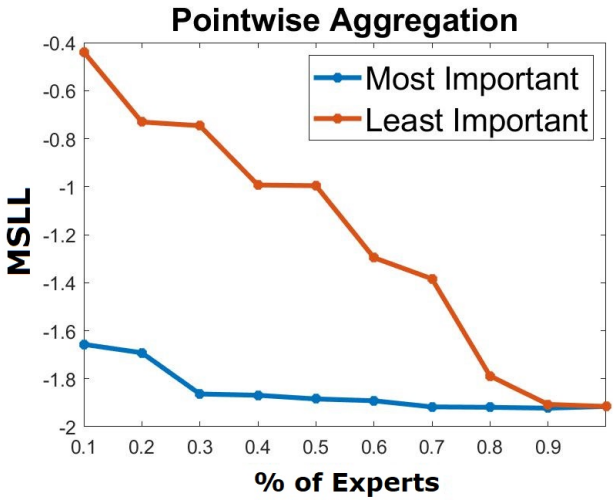
The global communication expert $\mathcal{D}_b$ in GRBCM* can be selected via sorted importance set $\mathcal{I}$ in Definition 5, because it contains expert interactions and the intensity



(a) MAE (NPAE*)



(b) SMSE (NPAE*)



(c) MSLL (NPAE*)

Figure C.2: **Ablation experiment 2.** Expert selection for synthetic data from (C.13) with varying expert set strength α and $\lambda = 0.1$. The blue line shows the prediction quality when $\alpha\%$ of most important experts are chosen using the GGM. The red line shows the prediction quality when the least important experts (according to Definition 5) are chosen.

of those interactions. The GRBCM* maintains the desired asymptotic properties of the original GRBCM, i.e. it leads to consistent estimation of the full Gaussian process at the presence of the assumption (i) and (ii) in Proposition 3 when $n \to \infty$. These modifications lead to a better predictive distribution at the end. While the first step eliminates

the negative effects of weak experts in the final aggregation, the second step chooses the most reliable (i.e., interconnected) expert in the related GGM as a global communicator. Note that the conventional GRBCM approach does not propose a strategy for selecting the global communication expert. Since this expert has a crucial impact on the final aggregation, the most interconnected expert is an appropriate choice for the global communication.

C.4.5 Relation between Experts Weights and expert selection

Here, we give a brief overview of how expert weighting is conducted for CI-based methods, and discuss the difference to our expert selection approach.

Assume a test point x^* is far away from a data partition $\mathcal{D}_i$. Using this partition, the prediction will generally be poor. Then, $k(x^*, X_i) \approx 0$, and $\Sigma_i^* \approx \Sigma^{**}$. Suppose the weights are defined as the difference in differential entropy. Then $\beta_i \approx 0$. Thus CIA based distributed GPs tend to assign a small weight to $\mathcal{D}_i$. On the other hand, in our expert selection approach, since $\mathcal{D}_i$ provides a poor prediction at x^*, its interactions with the other experts are weak, see (C.5) when $\Gamma_i \approx 0$. Hence, our approach recognizes this expert as an irrelevant expert. Consequently, expert selection eliminates this expert, while CIA based distributed GPs tend to keep this expert and assign a small weight to $\mathcal{D}_i$.

Finally, if equal weights are used for the CIA based methods, i.e. $\beta_i = 1/M$, weak and strong experts have an equal effect on the aggregated predictions. This may not be appropriate, particularly when there is a large number of candidate experts.

C.4.6 Computational Costs

The expert selection step significantly reduces the prediction cost of the NPAE method. The prediction cost of the original NPAE method is $\mathcal{O}(n_t M^3)$, where n_t is the number of available test points and M is the number of experts. Glasso has also cubic time complexity $\mathcal{O}(M^3)$ [2] . Therefore, the complexity of NPAE* is approximately $\mathcal{O}(n_t M_\alpha^3 + M^3) \approx \mathcal{O}((n_t \alpha^3) M^3)$ where $\alpha < 1$. It means the cost of Glasso can be ignored when n_t is

[2] There are newer faster methods to learn a GGM that reduce the computational complexity of sparse Gaussian Graphical Model to a much lower order of magnitude $\left(\mathcal{O}(M^2)\right)$, see [63, 64].

large and the aggregation complexity of NPAE* is much lower than NPAE. In CI-based methods, the complexities of the original and modified versions are of the same rate when the number of experts is large, see Section C.5.

C.5 Experiments

In this section, we evaluate the quality of the expert-importance-based estimator. We consider the prediction quality of our proposed model and other related SOTA models by using both simulated and real-world data sets. The quality of predictions is evaluated in two ways: we use the standardized mean squared error (SMSE) and the mean standardized log loss (MSLL). In addition, the conventional mean absolute error (MAE) is also used in Section C.5.1. We use the standard squared exponential kernel with automatic relevance determination and a Gaussian likelihood. Since the disjoint partitioning of training data captures the local features more accurately and outperforms random partitioning [38], it is mostly used in our experiments. All experiments have been conducted in MATLAB using the GPML package.[3]

C.5.1 Sensitivity Analysis using Synthetic Example

The goal of our first experiment is to study the effect of expert selection on prediction quality and computation time of the NPAE method. It is based on simulated data of a one-dimensional analytical function [38],

$$f(x) = 5x^2 sin(12x) + (x^3 - 0.5)sin(3x - 0.5) + 4cos(2x) + \varepsilon, \tag{C.13}$$

where $\varepsilon \sim \mathcal{N}(0, (0.2)^2)$. We generate $n = 5 \times 10^3$ training points in $[0,1]$, and $n_t = 0.1n$ test points in $[-0.2, 1.2]$. The data is normalized to zero mean and unit variance. We vary the number of experts, $M = \{10, 20, 30, 40, 50\}$, to consider different partition sizes. K-means method is used for the partitioning to compare the prediction quality of NPAE* with its original version NPAE, GPoE, RBCM, GRBCM and the full GP. Since the quality of PoE and BCM methods are low, we ignore them in this part. For α, we use two values, 0.5 and 0.8, which mean 50% and 20% of experts are excluded in the final aggregation.

[3] `http://www.gaussianprocess.org/gpml/code/matlab/doc/`

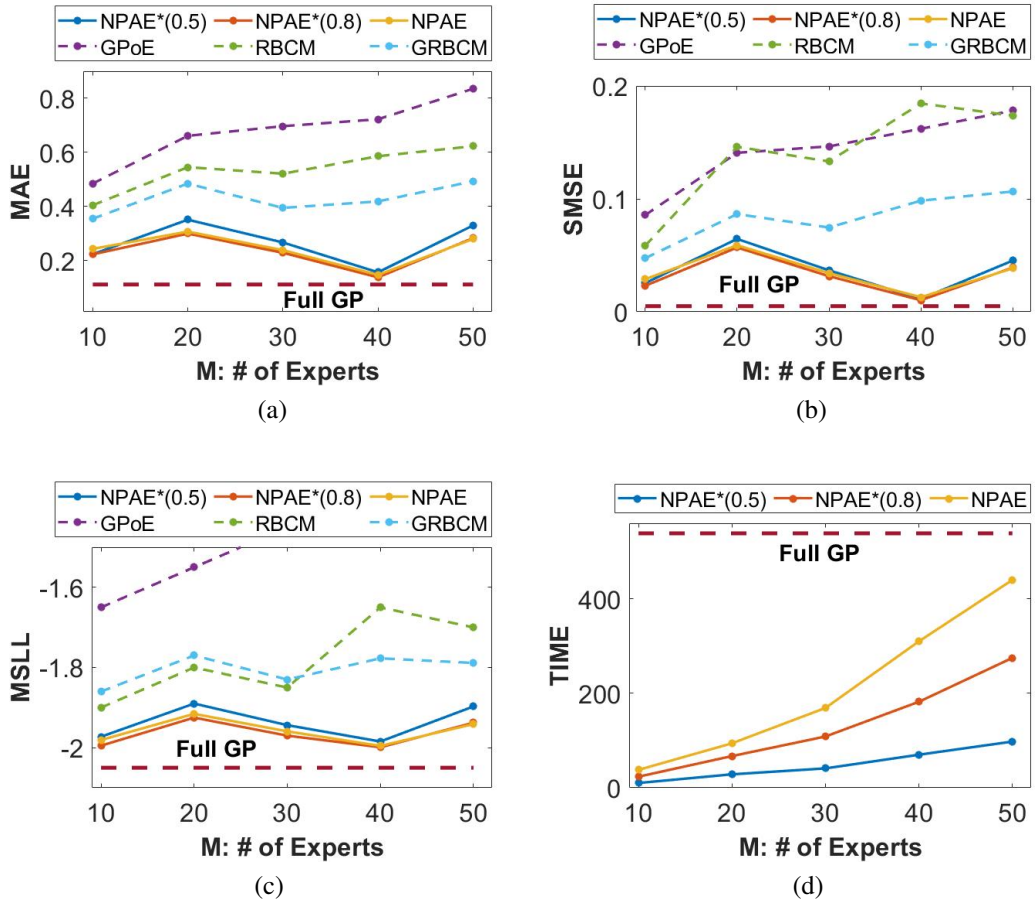
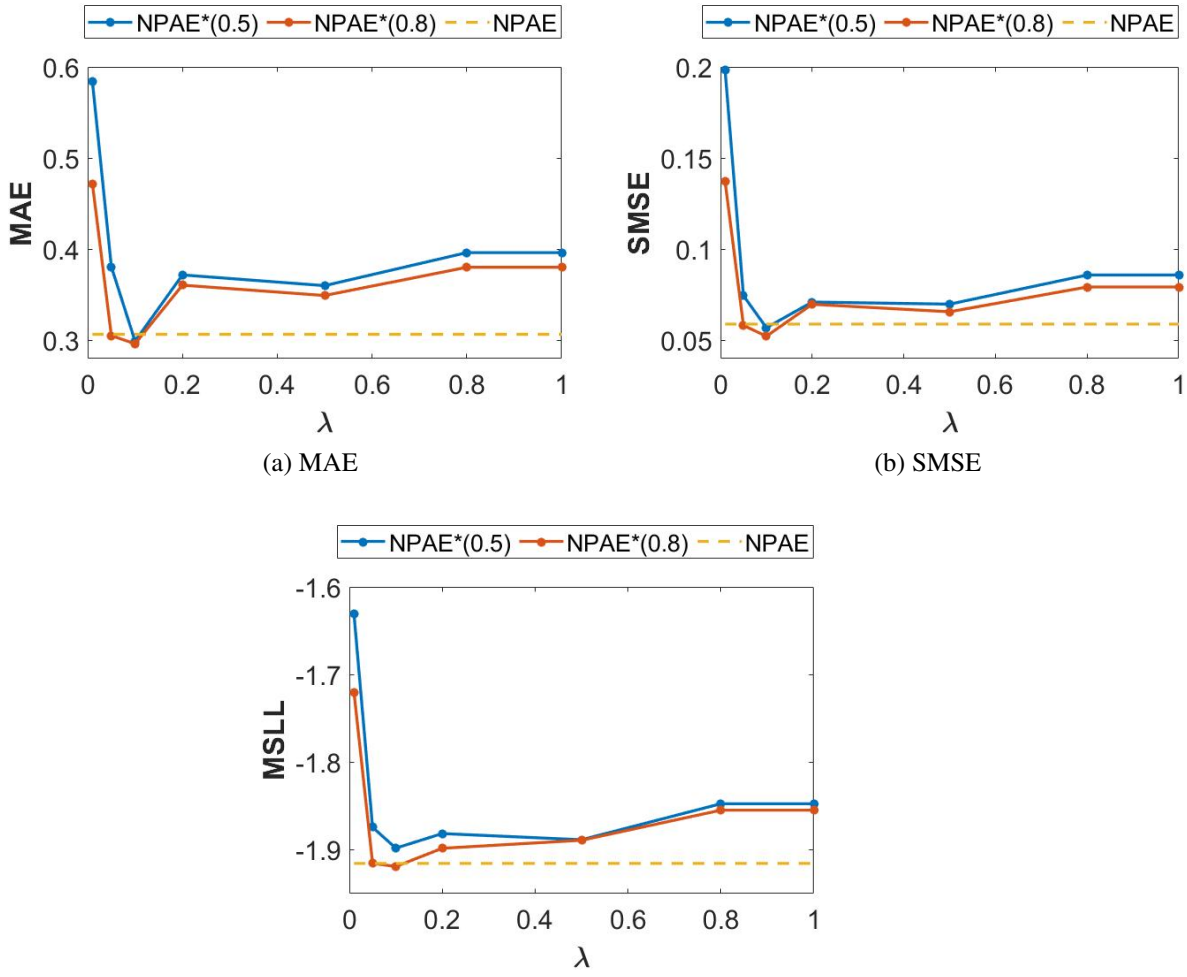


Figure C.3: **Prediction quality and computation time** as a function of experts. MAE, SMSE, MSLL, and running time of NPAE* with 50% and 80% of experts ,NPAE, GPoE, RBCM, GRBCM, and full GP for 5×10^3 training points and partition size $M = 10, 20, 30, 40, 50$ with K-means partitioning. The x-axis shows the number of partitions (experts) (M) used in training step for fixed $\lambda = 0.1$.

Figure C.3 depicts the prediction quality of NPAE* compared to other baselines and the full GP. For NPAE*, we vary the percentage of selected experts which leads to NPAE*(0.5) and NPAE*(0.8). These two methods reflect the prediction quality of the approximation method, proposed in Proposition 3 for pointwise NPAE method. The plots in C.3a, C.3b and C.3c indicate that even 50 percent of the highly connected experts, as selected by NPAE*(0.5), provide accurate approximation for NPAE while NPAE*(0.8) even improves the prediction quality of NPAE to some extent. When the number of experts increases, the difference in prediction error between NPAE family and CI-based

models increases, implying that with small partition size CI-based models are not capable to provide accurate predictions. Figure C.3d depicts the trade-off between the inclusion of more experts and computation time in NPAE method. Remarkably, NPAE*(0.5) and NPAE*(0.8) provide competitive prediction quality compared to NPAE in just a fraction of NPAE's running time.



(a) MAE

(b) SMSE

(c) MSLL

Figure C.4: MAE, SMSE and MSLL of NPAE* method with $\alpha = 0.5$ and $\alpha = 0.8$ for different values of the penalty term λ .

Finally, Figure C.4 presents the varying of the sparsity parameter λ to see how it affects the performance of NPAE*. It depicts the prediction quality measures for different λ values for 5×10^3 observations of the synthetic data set in (C.13) and $M = 20$. In this

experiments, NPAE*(0.5) and NPAE*(0.8) are used, which are related to $\alpha = 0.5$ and $\alpha = 0.8$, respectively. As the figure C.4 shows, small and large values (i.e. smaller than 0.05 or larger than 0.5) lead to slightly poor results but for values between 0.05 and 0.5, the network shows stable results. For the small λ value, the graph is dense with more edges while large λ leads to a very sparse network with only nodes and very few edges. Therefore, they do not provide appropriate results. The figures demonstrate $\lambda \approx 0.1$ is an appropriate choice to do the expert selection step.

A similar analysis can be found in Figure C.2 for α when $M = 20$. The blue lines show the prediction quality of NPAE* when the value of the α changes ($\alpha = 1$ shows the conventional NPAE method). We can see that when for $\alpha \geq 0.5$, the blue lines are almost horizontal, which means for values in this range, NPAE* returns a very close prediction to the standard NPAE. By increasing the α, the computational cost of NPAE* raises and approaches the NPAE's cost.

C.5.2 Real-World Data Sets

Partitioning Strategies in a Medium-Scale Data Set

In this section, we use a medium-scale real-world data set to assess the effect of the data assignment strategy on the prediction quality. The *Pumadyn*[4] is a generated data set with 32 dimensions and 7,168 training points and 1,024 test points. Both disjoint and random partitioning are used to divide the data set into 15 subsets. We consider the GPoE, RBCM, GRBCM, and NPAE with random and K-means partitioning. For the GPoE, RBCM, GRBCM and NPAE, we evaluate the proposed expert selection strategy with $\alpha = 0.8$ for GPoE, RBCM and GRBCM and $\alpha = 0.5$ and $\alpha = 0.8$ for NPAE. The penalty term λ is 0.1 in this experiment.

Table C.1 depicts the prediction quality of local approximation methods for this data set. The column Type shows the interactions between experts in the aggregation method, D for dependent experts, and CI for conditionally independent experts. The GPoE*, RBCM*, GRBCM*, and NPAE* are the modified versions of GPoE, RBCM, GRBCM, and NPAE, respectively. For CI-based methods, the modified methods outperform their original methods. For NPAE, the NPAE* with 50% of the experts returns an appropriate

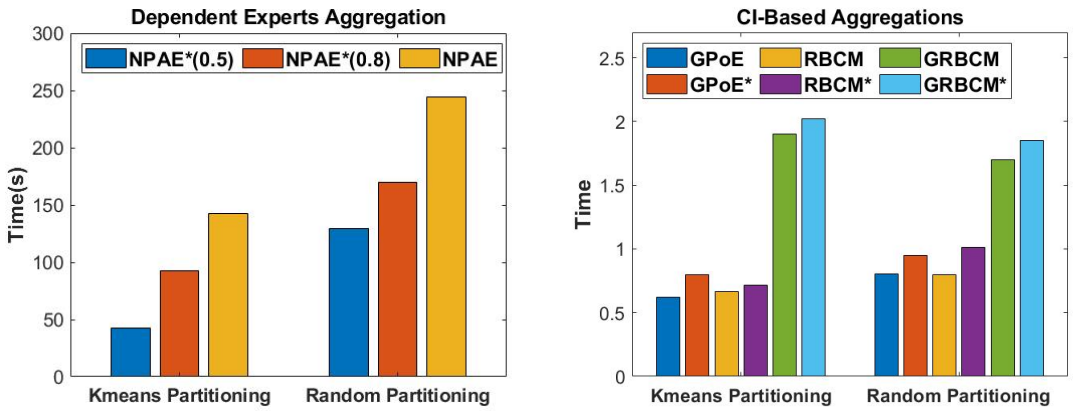
[4] `https://www.cs.toronto.edu/~delve/data/pumadyn/desc.html`

Table C.1: SMSE and MSLL of different baselines on the *Pumadyn* data set for different partitioning strategies. Both dependent (D) and conditionally independent (CI) aggregation methods are used.

		K-means		Random	
Model	Type	SMSE	MSLL	SMSE	MSLL
GPoE	CI	0.0483	-1.5166	0.0488	-1.5125
GPoE*	CI	0.0477	-1.5213	0.0485	-1.518
RBCM	CI	0.0478	1.1224	0.0485	2.9205
RBCM*	CI	0.0474	0.3949	0.0482	1.7881
GRBCM	CI	0.0499	-1.4949	0.0507	-1.502
GRBCM*	CI	0.0488	-1.508	0.0491	-1.510
NPAE*(0.5)	D	0.0470	-1.5285	0.0477	-1.5265
NPAE*(0.8)	D	**0.0468**	**-1.531**	**0.0474**	**-1.530**
NPAE	D	**0.0466**	**-1.536**	**0.0472**	**-1.534**

approximation for NPAE which has been discussed in Proposition 3. The NPAE* with 80% of the experts excludes only weak experts which leads to a significant improvement in prediction quality and offers results very close to the original NPAE. Besides, it has been widely accepted that the local features of the data can be captured more accurately in disjoint partitioning, see [38]. Table C.1 confirms this fact where the original and modified versions of aggregation methods reveal better prediction quality in K-means partitioning.

Figure C.5 depicts the prediction time of different baselines on the *Pumadyn* data set using both partitioning strategies. Figure C.5a compares the impact of our expert selection approach with NPAE and confirms that NPAE* is an appropriate fast approximation of NPAE. Its running time is up to one third that of NPAE while providing competitive results. On the other hand, Figure C.5b shows the computational costs of the CI-based methods and our suggested variants (GPOE*, RBCM*, and GRBCM*): the computational costs increase marginally while our modified versions improve upon prediction quality as shown in Table C.1.



(a) Dependent Experts

(b) Conditionally Independent Experts

Figure C.5: **Prediction Time** (seconds) of different aggregation for disjoint and random partitioning strategies in *Pumadyn* data set. In both partitioning strategies, the training data set is divided into 15 subsets and a fixed penalty term ($\lambda = 0.1$) is used.

Large Data sets

In this section we use three large-scale data sets, *Protein*[5], *Sacros*[6], and *Song*[7]. *Protein* has 9 dimensions and 45730 observations, 90% of which are used for training (i.e. 41,157 training and 4,573 test points). *Sacros* has 21 dimensions and 44,484 training and 4,449 test points. Finally, *Song* is a 91-dimensional data set with 515,345 instances, divided into 463,715 training and 51,630 test examples. We extract the first 10^5 songs from this data set for training and keep the original set of 51630 songs for testing. We used disjoint partitioning to divide the data sets into 70 (for *Protein*), 72 (for *Sacros*), and 150 (for *Song*) subsets.

Next, we compare SOTA baselines with NPAE*. We also consider GPoE*, RBCM*, and GRBCM*. For the model selection based methods, we set α to 0.8 which means 20% of weak experts are excluded. For the *Song* data set only, we set $\alpha = 0.1$ in NPAE*. Since NPAE is computationally burdensome, especially when M and n_t are large, it is not used in this part. Table C.2 reveals the prediction quality of local approximation baselines and shows NPAE* outperforms other SOTA methods on various data sets. In particular, NPAE* provides significantly better predictions on the *Song* data set using

[5] `https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure`

[6] `http://www.gaussianprocess.org/gpml/data/`

[7] `https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd`

Table C.2: **Prediction quality** for various methods on *Protein*, *Sacros*, and *Song* data. The table depicts SMSE and MSLL values for SOTA baselines and the modified versions of GPoE, RBCM, and NPAE, i.e. GPoE*, RBCM*, and NPAE* respectively.

		Protein		Sacros		Song	
Model	Type	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
PoE	CI	0.8553	33.2404	0.045	2.724	0.952	70.325
GPoE	CI	0.8553	-0.082	0.045	-1.183	0.952	-0.029
GPoE*	CI	0.8146	-0.1146	0.0038	-1.282	0.943	-0.034
BCM	CI	0.3315	-0.3329	0.007	-2.359	2.660	3.438
RBCM	CI	0.3457	-0.584	0.0045	-2.413	0.847	-0.006
RBCM*	CI	0.3411	-0.5953	0.0041	-2.520	0.842	-0.022
GRBCM	CI	0.3477	-0.613	0.0037	-2.642	0.836	-0.0926
GRBCM*	CI	0.3452	-0.623	0.0035	-2.731	0.819	-0.102
NPAE*	D	**0.3101**	**-0.6653**	**0.0028**	**-2.772**	**0.794**	**-0.110**

only 10 percent of mostly interacted experts. For CI-based methods, GPoE*, RBCM*, and GRBCM* improve the prediction quality of GPoE, RBCM, and GRBCM method, respectively. On the *Protein* data set, BCM and RBCM* have smaller SMSE values while the MSLL values of RBCM* and GRBCM are of the same rate. On the *Sacros* data set, the SMSE values of GPoE*, GRBCM, and GRBCM* are smaller than the other CI-based methods while RBCM*, GRBCM, and GRBCM* have smaller MSLL values. Although all CI-based methods return poor predictions on the *Song* data set, RBCM*, GRBCM, and GRBCM* provide smaller SMSE while the MSLL value in GRBCM and GRBCM* is smaller than other CI-based SOTA methods. Although the SMSE of RBCM and RBCM* are of the same rate, RBCM* significantly improves the MSLL values of the RBCM method.

C.6 Conclusion

In this work, we have proposed a novel expert selection approach for distributed GPs which leverages expert selection to aggregate dependent local experts' predictions. To

combine correlated experts, comparable SOTA methods use all experts and are affected by weak experts or leading to impractically high computational costs. Our proposed approach uses an undirected graphical model to find the most important experts for the final aggregation. Theoretically, we showed that our new local approximation approach provides consistent results when $n \to \infty$. Through empirical analyses, we illustrated the superiority of our approach which improves the prediction quality of existing SOTA aggregation methods, while being highly efficient.

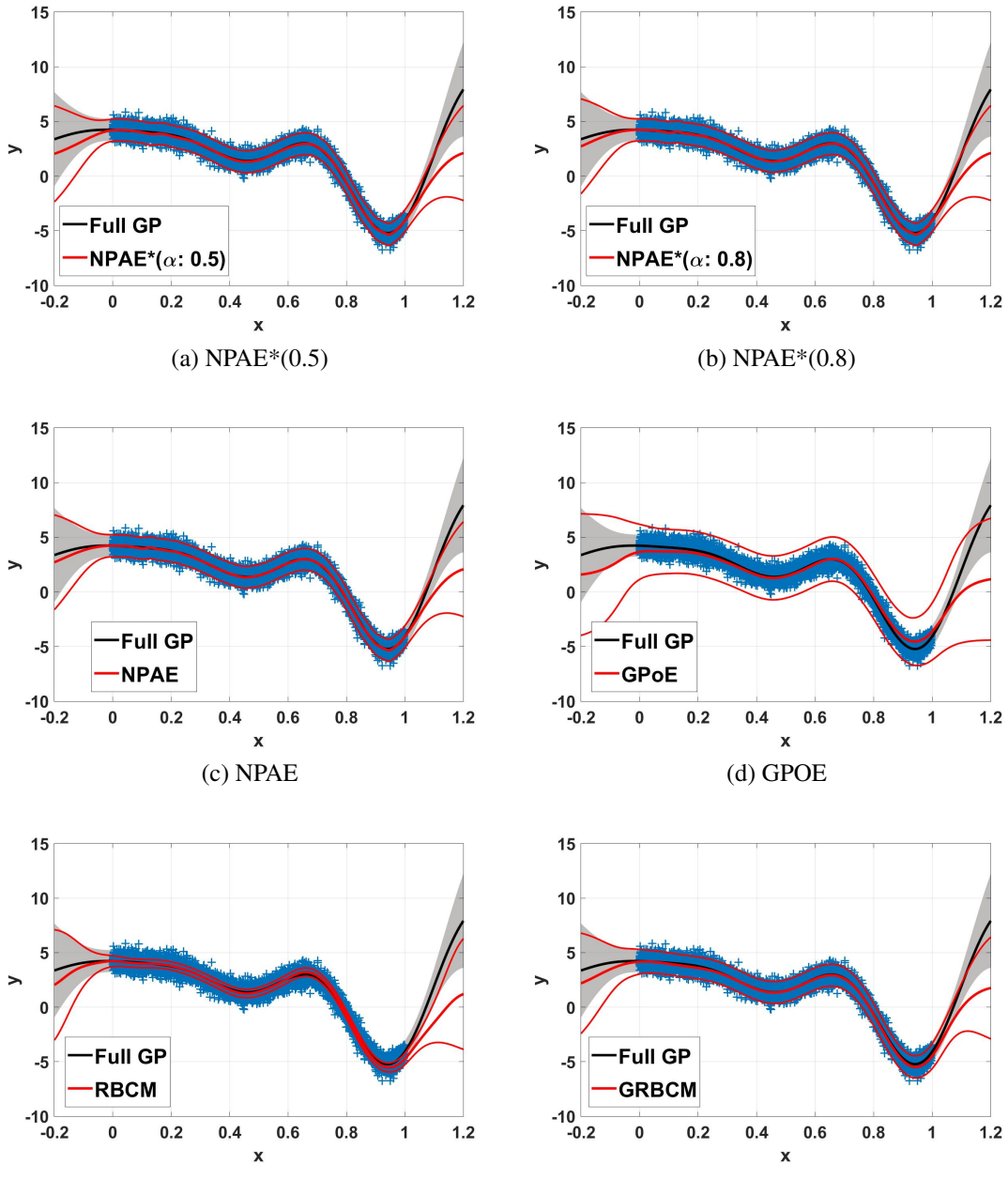
C.7 Appendix: Additional Experiments

C.7.1 Experimental Results: Synthetic Example

Figure C.6 depicts confidence intervals of different aggregation methods relative to the full GP from Section C.5.1 with $M = 20$. The NPAE, NPAE*(0.8), and NPAE*(0.5) produce reliable predictions when leaving the training data, i.e. for $x \in [-0.2, 0]$ and $x \in [1, 1.2]$. Their estimated mean values and confidence intervals are highly accurate, while the other methods show significant deviation from the full GP, e.g. see the mean vector of CI-based method in $x \in [-0.2, 0.2]$ and $x \in [0.8, 1.2]$ of figure C.6. Both NPAE*(0.5) and NPAE*(0.8) return impressive results which means their NPAE approximations are acceptable. The main drawback of GRBCM is its mean value, especially for $x \in [0.8, 1.2]$. However, its confidence intervals are much better than GPOE and RBCM methods. The results also fit the previous expectations about the GPOE and RBCM, RBCM provide overconfident results while GPOE is conservative.

C.7.2 Experimental Results: *Pumadyn* Data set

In this section we present the full experimental results for *Pumadyn* data set in Section C.5.2. We consider the GPoE, RBCM, GRBCM, and NPAE with random and K-means partitioning. For the GPoE, RBCM and NPAE, we evaluate the proposed expert selection strategy with $\alpha = 0.8$ for GPoE and RBCM and $\alpha = 0.5$ and $\alpha = 0.8$ for NPAE. We used both disjoint and random partitioning to divide the data set into 7, 15, and 20 subsets.



(a) NPAE*(0.5) (b) NPAE*(0.8)

(c) NPAE (d) GPOE

(e) RBCM (f) GRBCM

Figure C.6: **99% confidence interval** of different aggregation methods and full GP for 5×10^3 training points and partition size $m_0 = 250$ with K-means partitioning. The NPAE*(0.5) and NPAE*(0.8) results are based on $\alpha = 0.5$ and $\alpha = 0.8$, where $\lambda = 0.1$.

Table C.3: SMSE and MSLL of different baselines in *Pumadyn* data set for **K-means** partitioning strategy.

Number of Experts	M=7		M=15		M=20	
Model	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
GPoE	0.0475	-1.5213	0.0483	-1.5166	0.0499	-1.490
GPoE*	0.0473	-1.5299	0.0477	-1.5213	0.0497	-1.493
RBCM	0.0474	2.2199	0.0478	1.1224	0.0495	4.4283
RBCM*	0.0473	1.5514	0.0474	0.3949	0.0493	3.0198
GRBCM	0.0476	-1.5164	0.0499	-1.4949	0.0494	-1.5029
NPAE*(0.5)	0.0468	-1.5326	0.0470	-1.5285	0.0483	-1.5179
NPAE*(0.8)	0.0465	-1.5362	0.0468	-1.531	0.0480	-1.5202
NPAE	0.0464	-1.5367	0.0466	-1.536	0.0476	-1.5245

Table C.4: SMSE and MSLL of different baselines in *Pumadyn* data set for **Random** partitioning strategy.

Number of Experts	M=7		M=15		M=20	
Model	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
GPoE	0.0485	-1.511	0.0488	-1.5126	0.05	-1.4789
GPoE*	0.048	-1.52	0.0485	-1.518	0.0497	-1.4802
RBCM	0.0534	-0.0739	0.0485	2.9205	0.0495	3.4551
RBCM*	0.052	-0.3986	0.0482	1.7881	0.0492	2.2147
GRBCM	0.0498	-1.5056	0.0507	-1.502	0.0513	-1.4904
NPAE*(0.5)	0.0478	-1.525	0.0477	-1.5265	0.0488	-1.5039
NPAE*(0.8)	0.0469	-1.535	0.0474	-1.530	0.0480	-1.5103
NPAE	0.0473	-1.534	0.0472	-1.5344	0.0478	-1.5177

C.7.3 GRBCM Model with Expert Selection

The expert selection strategy can be easily extended to the GRBCM model. The modification in the GRBCM model, say GRBCM*, consists of the following two tasks:

(i) excluding the weak experts based on the procedure explained in Section C.4.1

(ii) choosing $\mathcal{I}_{i_1}$ (i.e., the top-most expert) in the sorted importance set $\mathcal{I}$ as the global communication expert (see Definition 5).

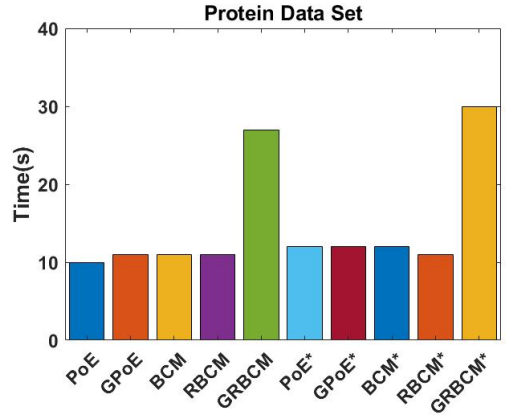
The global communication expert $\mathcal{D}_b$ in GRBCM* can be selected via sorted importance set $\mathcal{I}$ in Definition 5, because it contains expert interactions and the intensity of those interactions. The GRBCM* maintains the desired asymptotic properties of the original GRBCM, i.e. it leads to consistent estimation of the full Gaussian process at the presence of the assumption (i) and (ii) in Proposition 3 when $n \to \infty$. These modifications lead to a better predictive distribution at the end. While the first step eliminates the negative effects of weak experts in the final aggregation, the second step chooses the most reliable (i.e., interconnected) expert in the related GGM as a global communicator. Note that the conventional GRBCM approach does not propose a strategy for selecting the global communication expert. Since this expert has a crucial impact on the final aggregation, the most interconnected expert is an appropriate choice for the global communication.

C.7.4 Experimental Results: Large Scale Data sets

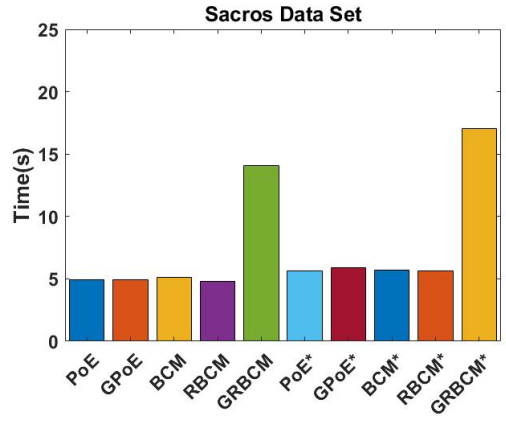
Table C.5 depicts the prediction quality of the CI-based aggregation method, i.e. (G)PoE, (R)BCM, and GRBCM. The consistent methods have been discussed in Table C.2. For a constant penalty term, $\lambda = 0.1$, the modified version of CI-based methods have been defined using $\alpha = 80\%$ of the most important experts. The (G)PoE*, (R)BCM*, and GRBCM* are the expert selection version of the original (G)PoE, (R)BCM, and GRBCM methods. Table C.5 shows a significant improvement after excluding the weak experts. Since the CI-based methods do not have appropriate asymptotic properties, they are not capable to outperform the NPAE which considers the experts' dependency. Figure C.7 presents the running time of the baselines on three large scale data sets in Table C.5. We can observe that the running time of both original and modified versions are of the same rate. The cost of the modified versions contain the cost of the GLasso (one time) and then the cost of aggregation on a smaller expert' set.

Table C.5: **Prediction quality** for various CI based baselines and their modified versions on *Protein*, *Sacros*, and *Song* data sets.The table depicts SMSE and MSLL values for (G)PoE, (R)BCM, GRBCM, and their modified version after excluding the weak experts, i.e. (G)PoE*, (R)BCM*, and GRBCM* respectively.

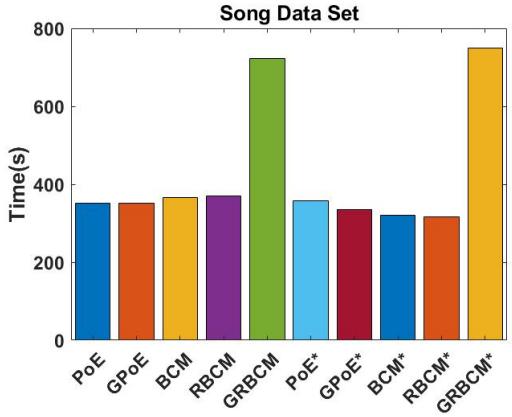
	Protein		Sacros		Song	
Model	SMSE	MSLL	SMSE	MSLL	SMSE	MSLL
PoE	0.8553	33.2404	0.045	2.724	0.952	70.325
GPoE	0.8553	-0.082	0.045	-1.183	0.952	-0.029
BCM	0.3315	-0.3329	0.007	-2.359	2.660	3.438
RBCM	0.3457	-0.584	0.0045	-2.413	0.847	-0.006
GRBCM	0.3477	-0.613	0.0037	-2.642	0.836	-0.0926
PoE*	0.8146	26.0748	0.038	1.139	0.943	55.476
GPoE*	0.8146	-0.1146	0.0038	-1.282	0.943	-0.034
BCM*	0.324	-0.3776	0.007	-2.500	2.161	2.349
RBCM*	0.3411	-0.5953	0.005	-2.520	0.842	-0.022
GRBCM*	0.3452	-0.623	0.0035	-2.731	0.819	-0.102



(a) *Protein* Data Set



(b) *Sacros* Data Set



(c) *Song* Data Set

Figure C.7: Running Time of CI-based aggregation methods on *Protein*, *Sacros*, and *Song* data sets.

Appendix D

Entry Dependent Expert Selection in Distributed Gaussian Processes Using Multilabel Classification

This chapter is based on the following articles:

- Hamed Jalali and Gjergji Kasneci. Entry Dependent Expert Selection in Distributed Gaussian Processes Using Multilabel Classification. Under review at the IEEE Transactions on Neural Network and Learning Systems (TNNLS), 2022.
- Hamed Jalali and Gjergji Kasneci. Expert Selection in Distributed Gaussian Processes: A Multi-label Classification Approach. GPSMDMS: NeurIPS Workshop onGaussian Processes, Spatiotemporal Modeling, and Decision-making Systems at 36th Conference on Neural Information Processing Systems (NeurIPS 2022).[1]

[1]This publication is a short version of the article submitted in TNNLS.

D.1 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.

D.2 Introduction

Gaussian processes (GPs) [10] 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 [87, 13, 88, 89, 20, 22], their cubic training and quadratic prediction costs [2] limit their application to big data use cases [1].

For GP regression, the major computational hurdle is the need to estimate the kernel

[2] I.e., in the size of the training set.

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 $\mathcal{O}(10^4)$.

To reduce computational expense, sparse approximation techniques use a subset of the training sets (called inducing points) and *Nyström* approximation to estimate posterior distributions [30, 31, 84, 85]. 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 [33, 34].

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, local inference is conducted for each partition separately, and then the local estimations are combined through ensemble learning [90, 91, 92]. A local GP that has expertise in a particular partition is called an expert. Experts share the same hyper-parameters, thus accounting for implicit regularisation and encountering overfitting [1, 93].

In a DGP, the *conditional independence* (CI) assumption between partitions (i.e., between experts given the target) allows 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 [45] caused by the partitioning of the data set, such that when $N \to \infty$, 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 [44, 46, 6]. 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 [4, 8], 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 [4], 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 [76] 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 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-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 [77, 93].

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 of 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 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 [94, 95].

The structure of the paper is as follows. Section D.3 introduces the problem formulation and related works. The proposed model and inference process are presented in Section D.4. Section D.5 discusses some associated details. Section D.6 shows the experimental results, and we conclude in Section D.7.

D.3 Problem Set-up

D.3.1 Gaussian Process

We start with the basic non-linear regression problem $y = f(x) + \varepsilon$, and the objective is to learn the latent function f from a training set $\mathcal{P} = \{X, y\}$. Assume the training set contains N observation, X is a d-dimensional variable, $x \in R^d$, and ε is a zero-mean Gaussian noise $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. 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 hyper-

parameters $\theta = \{\sigma^2, \tau\}$ such that they maximise the related log-marginal likelihood,

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

where $\mathcal{Z} = k(X,X) + \sigma^2 I$. Optimization task in (D.1) scales as $\mathcal{O}(N^3)$ because it requires to calculate 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 data sets.

D.3.2 Local Approximation Gaussian Process

The local approximation Gaussian process is a *divide-and-conquer* approach, which partitions the training data set into M subsets $\mathcal{P}' = \{\mathcal{P}_1, \ldots, \mathcal{P}_M\}$ and trains standard GPs on these subsets. It is also called 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 hyper-parameters $\theta = \{\sigma^2, \psi\}$ [1].

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:

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

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

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

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 [38, 4]. 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 that 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

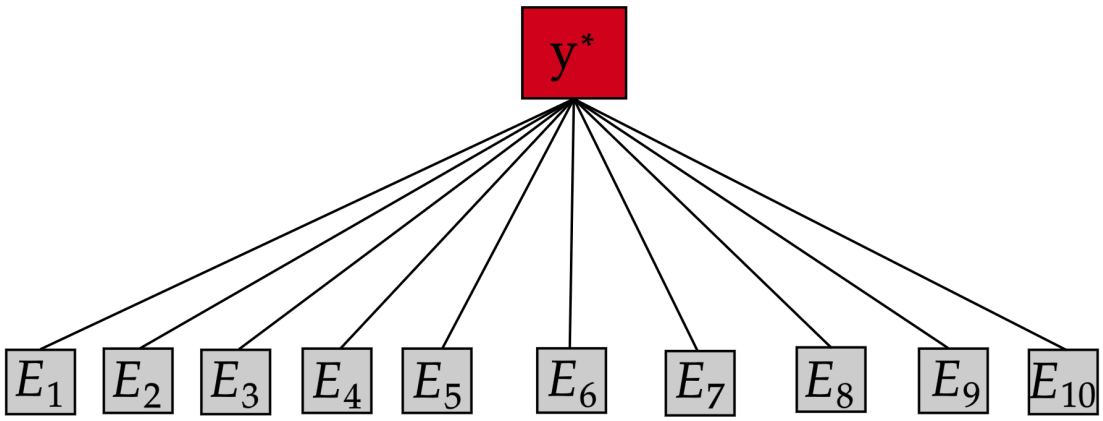


Figure D.1: **Computational graphs** of an aggregation based on conditional independence assumption between experts.

x^*

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

Equation (D.4) shows that the aggregated predictive distribution can be defined as the product of local densities. The $\beta = \{\beta_1,\dots,\beta_M\}$ describe the weights and influence of the experts. The most prevalent CI-based aggregation methods are product of experts (PoE) [41], generalised product of experts (GPoE) [37], Bayesian committee machine (BCM) [42], robust Bayesian committee machine (RBCM) [1] and generalized robust Bayesian committee machine (GRBCM) [38].

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

D.3.3 Beyond Conditional Independence Assumption

Ensemble methods extensively employ the conditional independence assumption for regression and classification problems [48, 49]. 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 [2]. In classification, modeling dependencies be-

tween classifiers has been considered in several works, for example, in [50, 51, 2]. However, few works have considered modeling expert dependencies in local approximation GPs. The nested pointwise aggregation of experts (NPAE) method [44, 46] 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 (D.2). Each of the local Gaussian experts $\mathcal{E}_i$ is a linear estimator because the related 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 [44] assumed that y_i in (D.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, $Cov(\mu_i^*, y^*)$, 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:

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

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

For a test point $x^* \in X^*$, the point-wise covariances are defined as

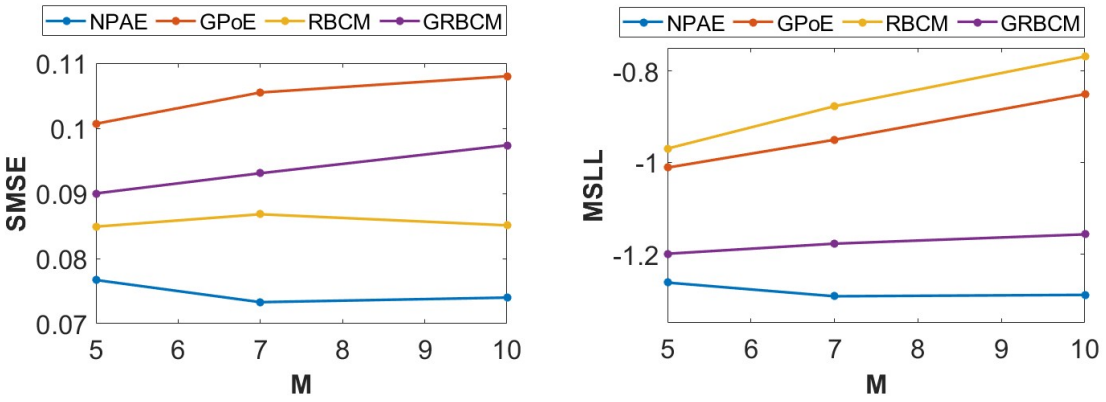
$$r(x^*) = Cov(\mu^*(x^*), y^*(x^*))$$

and

$$R(x^*) = Cov(\mu^*(x^*), \mu^*(x^*))$$

, where $r(x^*)$ is a $M \times 1$ 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 issue comes from here 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 6 (**Dependency-Based Aggregation**)**.** The aggregated predictor for the test



(a) SMSE (b) MSLL

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

point x^* and local predictions $\mu_1^*(x^*), \ldots, \mu_M^*(x^*)$ is defined as

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

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 [44].

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 [10].

Figure D.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'

[3] `https://archive.ics.uci.edu/ml/datasets/concrete+compressive+strength`

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.

D.3.4 Asymptotic Properties

Conventional DGP baselines suffer from *inconsistency*. Since the local experts are trained on separated partitions, the aggregation produces inconsistent predictions which 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 [1] conservatively converges to the full GP distribution when $N \to \infty$ [45, 46]. However, the authors in [38] showed that the GPoE produces consistent predictions using random partitioning under some mild assumptions.

The generalized robust Bayesian committee machine (GRBCM) [38] 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 (D.5) and (D.6), and the aggregated predictor in (D.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 $\mathcal{O}(N_t M^3)$, which is not an efficient solution for complex real-world data sets 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 experts set that modifies the aggregation estimator in (D.7).

D.4 Expert Selection in Local Approximation GPs

The current DGP baselines rely on experts' weighting with the goal to quantify the experts' importance. In [1], 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 (D.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.

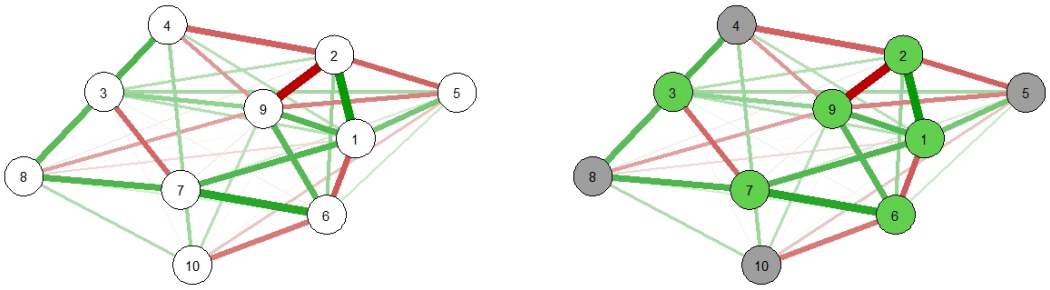
D.4.1 Expert Selection Using Graphical Models

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 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) [59, 61] 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,

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

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



(a) GGM (b) Selected Experts

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

Modeling the dependency in distributed learning by GGMs has been studied in [3, 6], where the precision matrix encodes the interactions between experts. The authors in [3] 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 [8, 4], 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 7 (**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 (D.7) [4].

Figure D.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'

Algorithm 3 Aggregating Dependent Experts Using GGM

Input: Local predictions of GP experts μ^*, sparsity hyperparameter λ, selection percentage α.

Output: Aggregated estimator y_A^*.

1: Calculate sample covariance S of experts' predictions.
2: Estimate $\hat{\Omega}$ using *GLasso* (D.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 7 .
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 (D.7).

predictions are used to quantify interactions between experts in this graph. Figure D.3a presents the original GGM related to this distributed learning case. Figure D.3b shows an expert selection scenario when only 60% of the experts are selected based on their importance. In [8, 4], 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 3 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 7, 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 $\mathcal{O}(M^3)$. 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.

D.4.2 Multi-label Classification for Flexible Expert Selection

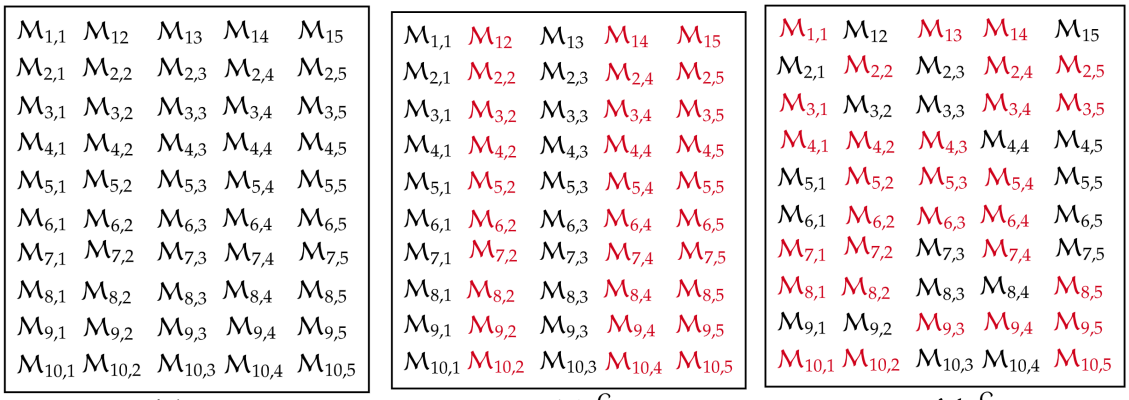
Assigning experts to new entry points in a distributed learning model can be seen as a classification problem. Let's assume that each expert is a class of estimators. The selection problem then for each test point x^* is defined as a multi-label classification task where each instance can be associated with some classes. The main advantage of this method is its flexibility because the selected experts depend on the given test 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 $\mathcal{M}^{\mathcal{C}}(x^*) = \{\mathcal{M}^{\mathcal{C}}{}_1(x^*), \ldots, \mathcal{M}^{\mathcal{C}}{}_K(x^*)\}$ that 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'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 D.4 describes the difference between static and dynamic expert selection models in an example with synthetic data points. Figure D.4a depicts the original aggregation where all experts are used, e.g., for the ensemble model in (D.7). The GGM-based expert selection in D.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 this 10 test points. The flexibility of the entry-based selection model is depicted in D.4c, where the model assigns different experts to each test point x^* and uses the ability of experts in a better way.

D.4.3 Adopted K-nearest neighbors (KNN)

The K-nearest neighbors algorithm [96] 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 learn-



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

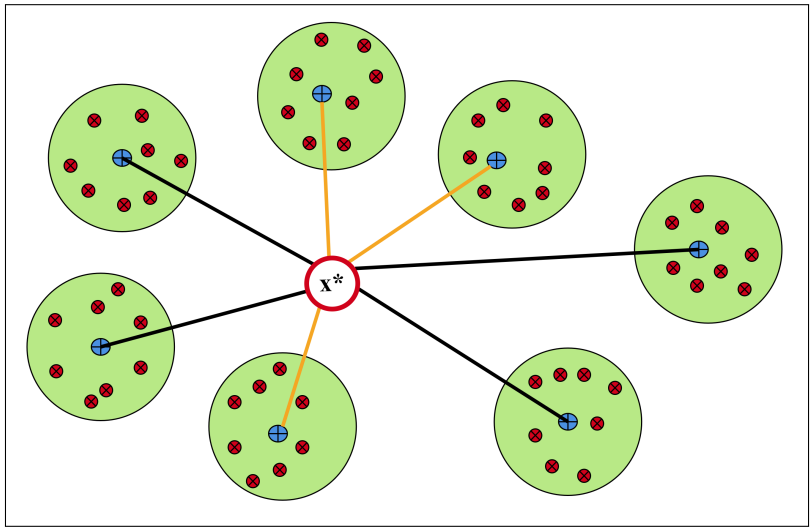
Figure D.4: **Expert Selection** scheme of both static and entry-dependent models for a setting of 5 experts with 10 test points. Both selection models assign 3 experts to each test points: (a) original set of experts $\mathcal{E}$, (b) static assignment of experts $\mathcal{E}^{\mathcal{G}}$, and (c)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

ing 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 $dist(x^*, \mathcal{C})$, in which the i'th 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 closest centroids 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. There are several distance measures that can be chosen, e.g. *Euclidean*, *Manhattan*, *Minkowski*, and *Hamming* distances. In this work we use the conventional *Euclidean* distance.



(a) KNN

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

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 [78].

Figure D.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.

Algorithm 4 summarizes the whole procedure of the KNN-based aggregation. The main advantage of the KNN method is 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 accepts new entry data at any time. Besides, instead of n training points, the modified KNN proposed in Algorithm 4 only uses the M centroids and scales the distance calculations in large data sets.

Generally, this selection method is defensible and justifiable because Gaussian pro-

Algorithm 4 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. $dist(x^*, \mathcal{C})$.
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 (D.2) and (D.3).
5: Aggregate local predictions from Step 4 using (D.7).

cesses predict better when a test point is close to them. Since the distance-based solution may have some drawbacks in high dimensional data sets, we propose another multi-label classification solution in the next section that can be effective in high dimensional cases.

D.4.4 Adapted Neural Networks for Classification

Conventional deep neural networks (DNNs) are widely used in machine learning problems, especially in classification tasks [97, 98]. 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 D.4.2. 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 data set. For each $x_i \in X$,

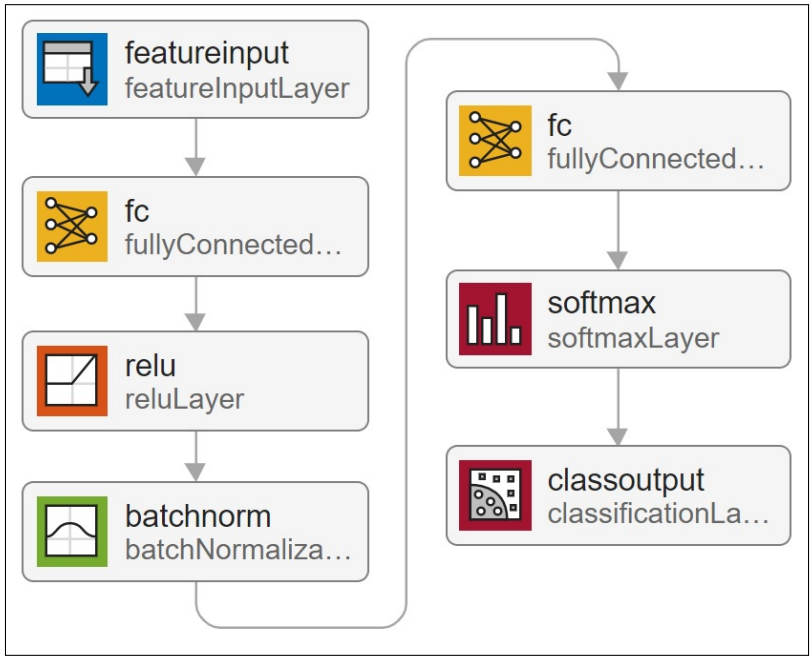


Figure D.6: **Deep Neural Network Architecture** for adopted multi-label classification in DGPs.

$i = 1, \ldots, N$, 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 D.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.

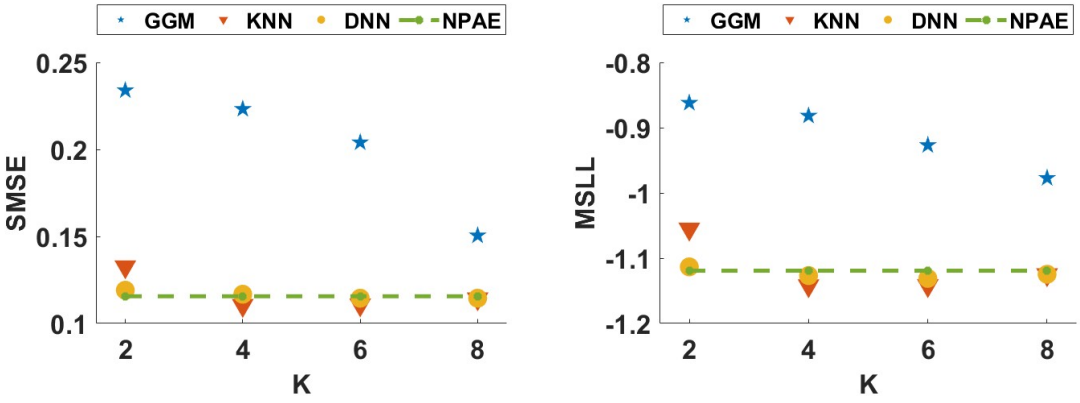
Figure D.7 depicts the prediction quality of expert selection methods on the *Concrete* data set with 10 partitions for different values of K. As it can be seen, multi-label-based expert selection models provide higher quality predictions with lower deviation from the NPAE model. The quality of the classification-based aggregations changes less as the selection parameter K increases. The case $K = 10$ leads to the original NPAE baseline

Algorithm 5 Aggregating Dependent Experts Using DNN

Input: Test point $x^* \in X^*$, training points X and their indices, index set $\mathcal{L}$, 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 $P^{\mathcal{L}}(x^*)$, 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 (D.2) and (D.3).
7: Aggregate local predictions from Step 6 using (D.7).



(a) SMSE (b) MSLL

Figure D.7: **Expert Selection** prediction qualities of different experts selection methods compared to original baseline, NPAE, from *Concrete* data set.

(the dashed green lines in Figures D.7a and D.7b), and both KNN and DNN return proper error values in both plots.

D.5 Discussion

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

D.5.1 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

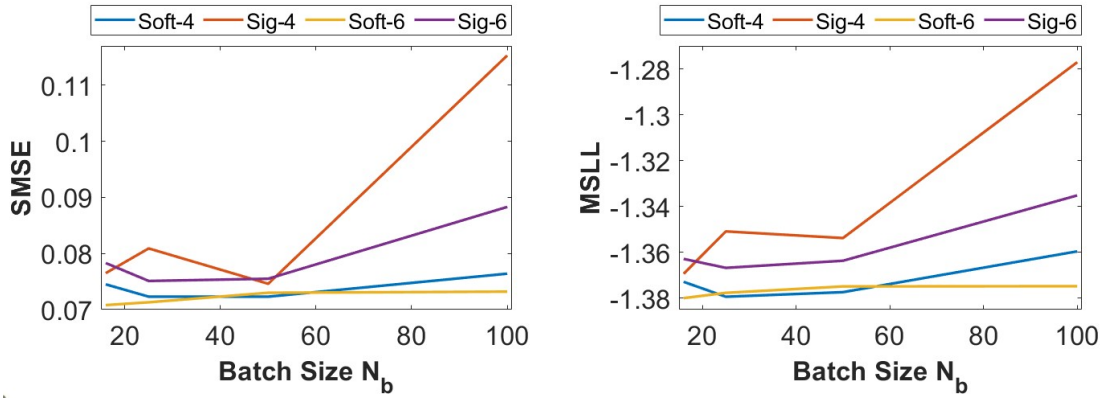
GLasso solves this optimization problem. This assumption is strong, and in some cases, it can be restrictive. Various solutions have been proposed to relax this assumption by considering the model as a nonparametric problem and solving after some smooth monotone transformation [69, 70, 73, 74] at the 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 [79, 80]. 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.

D.5.2 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 3 needs GLasso to estimate the precision matrix, and its computational cost is $\mathcal{O}(M^3)$, 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 λ leads to a dense graph with a more considerable computational cost.

The cost of the KNN approach 4 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 are usually negligible compared to the rest of the algorithm. However, we consider this aspect as well in the overall cost estimation. Algorithm 4 computes the distance between the new observation and each centroid point,



(a) SMSE (b) MSLL

Figure D.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* data set with 10 experts.

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 5 depends on the network structure, i.e., the number of layers L, the input dimension D, the output dimension M, 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 4 and 5 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 3.

D.5.3 Activation Functions in DNN

Using a *softmax* output layer for the DNN-based classification in Section D.4.4 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 D.8 explains how different activation functions can affect the prediction qual-

ity. 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.5.4 Expert Selection for CI-Based Baselines

In [4], 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 ((D.4)), only K experts are used, and the selection is based on the Algorithm 4 or Algorithm 5. Since these models are fast, the selection parameter K can be also set to relatively large values.

Table D.1: Expert Selection in CI-Based Baselines.

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

Table D.1 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.

D.6 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 data sets. 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 [38, 4], 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 have been conducted in MATLAB using the GPML package[4].

D.6.1 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 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.

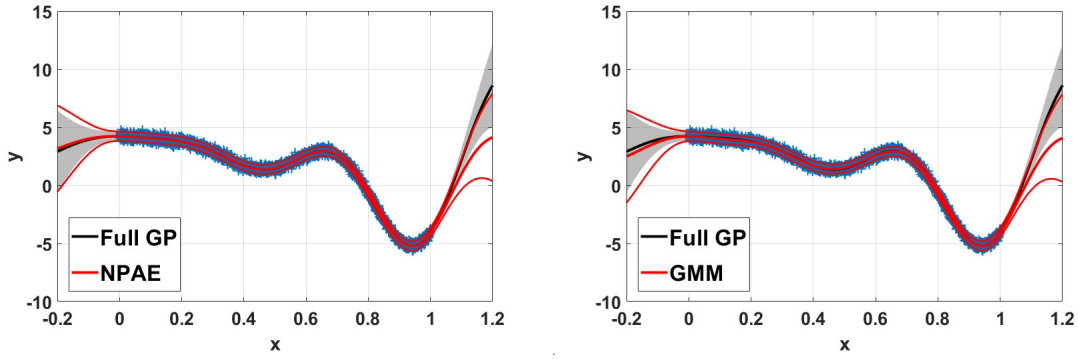
Synthetic Example

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 [38],

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

where $\varepsilon \sim \mathcal{N}(0, (0.2)^2)$. We generate n training points in $[0,1]$, and $N_t = 0.1n$ 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 in these experiments.

[4] `http://www.gaussianprocess.org/gpml/code/matlab/doc/`

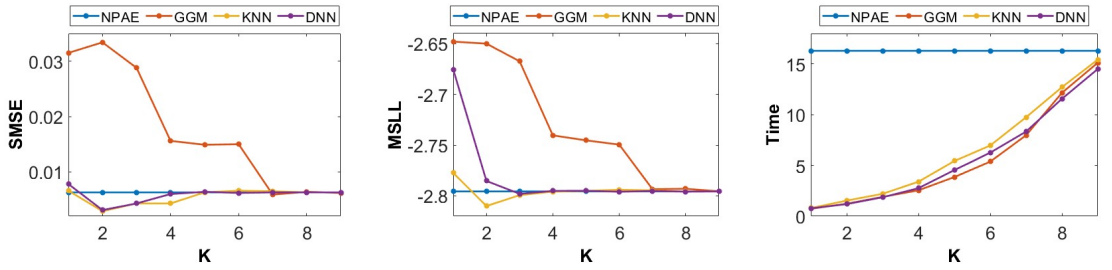


(a) NPAE (b) GGM

(c) KNN (d) DNN

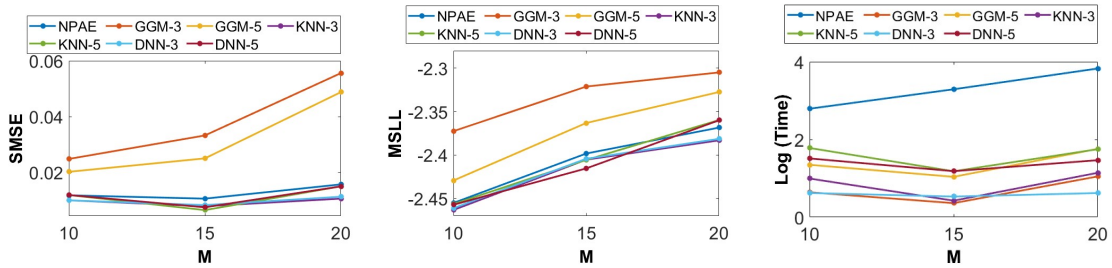
Figure D.9: **99% Confidence Interval** of NPAE, expert selection methods and full GP for $n = 3 \times 10^3$ training points from Equation ((D.9)) and $M = 10$ (partition size $m_0 = 300$) with K-means partitioning. The GGM, KNN, and DNN results are based on $K = 5$ selected experts. We see that the 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.

Figure D.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 (D.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

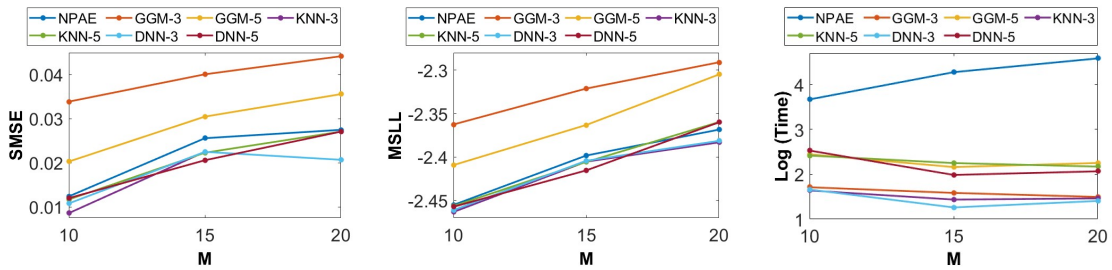


(a) SMSE and $n = 3000$ (b) MSLL and $n = 3000$ (c) Time and $n = 3000$

Figure D.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 $n = 3000$



(d) SMSE and $n = 5000$ (e) MSLL and $n = 3000$ (f) Time and $n = 3000$

Figure D.11: **Prediction quality** and running time of DGP baselines for 3000 and 5000 training points from Equation ((D.9)) with different numbers of partitions, $M = \{10, 15, 20\}$. For expert selection-based methods, $K = \{3, 5\}$ experts have been selected for the final aggregation.

GGM plot, which shows a significant deviation from the standard GP.

Figure D.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 D.10a and D.10b indicate that multi-label classification leads to a selection strategy with fast convergence to 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 D.10c indicates that the computational costs of the aggregations based on GGM, KNN, and DNN are at the same rate.

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 D.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 D.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.

Figures D.11c and D.11f show the running time of the baselines that consider the dependencies between experts. All expert selection-based aggregations have same prediction process (of $\mathcal{O}(N_t K^3)$), and their difference in only with respect to 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 D.5.2. In these experiments with smooth 1D data points, the running times of the GGM, the KNN, and the DNN are of the same rate.

Multi-Dimensional Real-World Data Set

The relative number of experts $K_M = \frac{K}{M}$ defined in Section D.6.1 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 data set and K_M to appraise the efficacy of the data assignment strategy on the prediction quality. *Pumadyn*[5] 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[1], GRBCM[38], NPAE[44], GGM-based aggregation [4], 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 D.2 depicts the prediction quality of local approximation methods for the *Pumadyn* data set. The column Type shows 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.

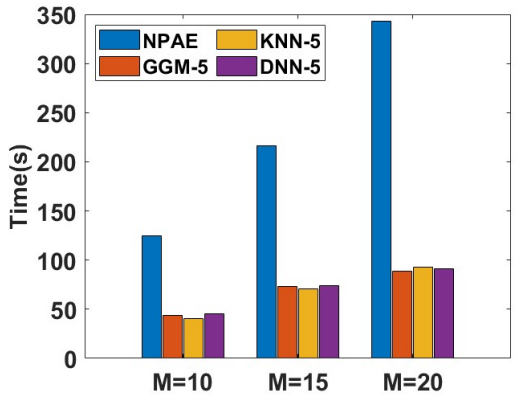
NPAE is a basis for comparison because it is the best linear unbiased predictor (BLUP). The multi-label classification methods provide accurate results, and their derivatives with $K_M = 0.5$ and $K_M = 0.7$ are close to the NPAE. This performance shows the fact that convergence occurs faster in these 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 D.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

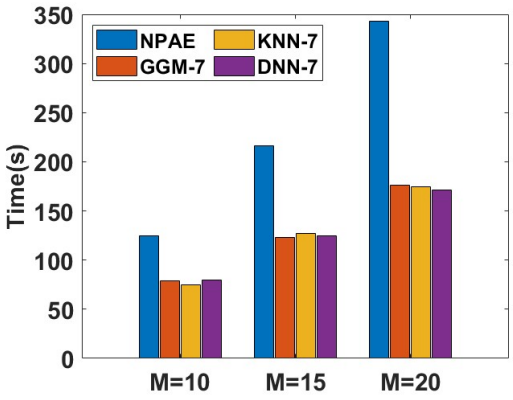
[5] `https://www.cs.toronto.edu/~delve/data/pumadyn/desc.html`

Table D.2: 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.

		$M = 10$		$M = 15$		$M = 20$	
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**



(a) Aggregation with 50% of Experts



(b) Aggregation with 70% of Experts

Figure D.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.

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 D.2 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 D.10 and D.11. The main reason for this issue is the low convergence rate of the GGM, which requires more experts.

D.6.2 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 them with available baselines. The baselines that we use here are GPoE with uniform weights[1], RBCM[1], GRBCM[38], NPAE[44], GGM-based aggregation[4], KNN and DNN based ensemble methods. Various real-world data sets with different sizes and dimensions are used here which have been explained in D.3.

Table D.3: Real-World Data Sets.

Data Set	(#) Observations	n	N_t	D
Airfoil[6]	1,503	1,203	300	5
Parkinson[7]	5,875	5,000	875	20
Pole Telecom[99]	15,000	10,000	5,000	26
Protein[8]	45,730	40,000	5,730	9
Sacros[9]	48,938	44,489	4,449	21
Song[10]	515,345	463,715	51,630	91

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.

[6] `https://archive.ics.uci.edu/ml/datasets/airfoil+self-noise`
[7] `https://archive.ics.uci.edu/ml/datasets/parkinsons+telemonitoring`
[8] `https://archive.ics.uci.edu/ml/datasets/Physicochemical+Properties+of+Protein+Tertiary+Structure`
[9] `http://www.gaussianprocess.org/gpml/data/`
[10] `https://archive.ics.uci.edu/ml/datasets/yearpredictionmsd`

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.

Table D.4: **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.

	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**

Tables D.4 and D.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 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 accurate prediction when $K_M = 0.5$. Our experiments with $K_M = 0.7$ [11] 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

[11] The results for $K_M = 0.7$ are not shown in the Tables D.4 and D.5

Table D.5: **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.

	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**

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[12]. 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 ((D.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.

D.7 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

[12] the MSLL values of the KNN and DNN methods for $K_M = 0.7$ are -0.8575 and -0.8587, respectively.

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.