Expert Selection in Distributed Gaussian Processes: A Multi-label Classification Approach

Hamed Jalali, Gjergji Kasneci
University of Tuebingen, Germany

EBERHARD KARLS UNIVERSITÄT TÜBINGEN

NEURAL INFORMATION PROCESSING SYSTEMS

Introduction

Distributed Gaussian process (GP) is a prominent method to scale Gaussian Processes (GPs) to big data based on the divide-and-conquer approach, 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 (CI), which means there is perfect diversity between the local experts. The CI assumption is often violated in practice, and experts' aggregation leads to sub-optimal and inconsistent solutions
- The proposed models that encode dependency between experts and cope with the consistency problem suffer from high computational costs, mainly when the number of experts is large
- The critical contribution of our work lies in selecting a dynamic and entry-dependent subset of local experts for each new data point using multi-label classification (MLC). The main advantage of this method is its flexibility because the selected experts depend on the given test point.
- We adopt two MLC algorithms to assign experts to data points: k-nearest neighbors (KNN) and deep neural networks (DNN).

Problem Set-Up

Consider the regression problem $y = f(x) + \epsilon$, where the objective is to learn the latent function f from a training set $\mathcal{D} = \{X, y\}_{i=1}^{n}$.

Distributed GP involves dividing the $\mathcal{D}$ into M partitions $\mathcal{D}_1, \ldots, \mathcal{D}_M$, and training the standard GP at each partition with predictive distribution $p_i(y^*|\mathcal{D}_i, x^*) \sim \mathcal{N}(\mu_i^*, \Sigma_i^*)$ for test point x^*.

The predictive distribution is given as the product of multiple densities. For independent experts the predictive distribution is

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

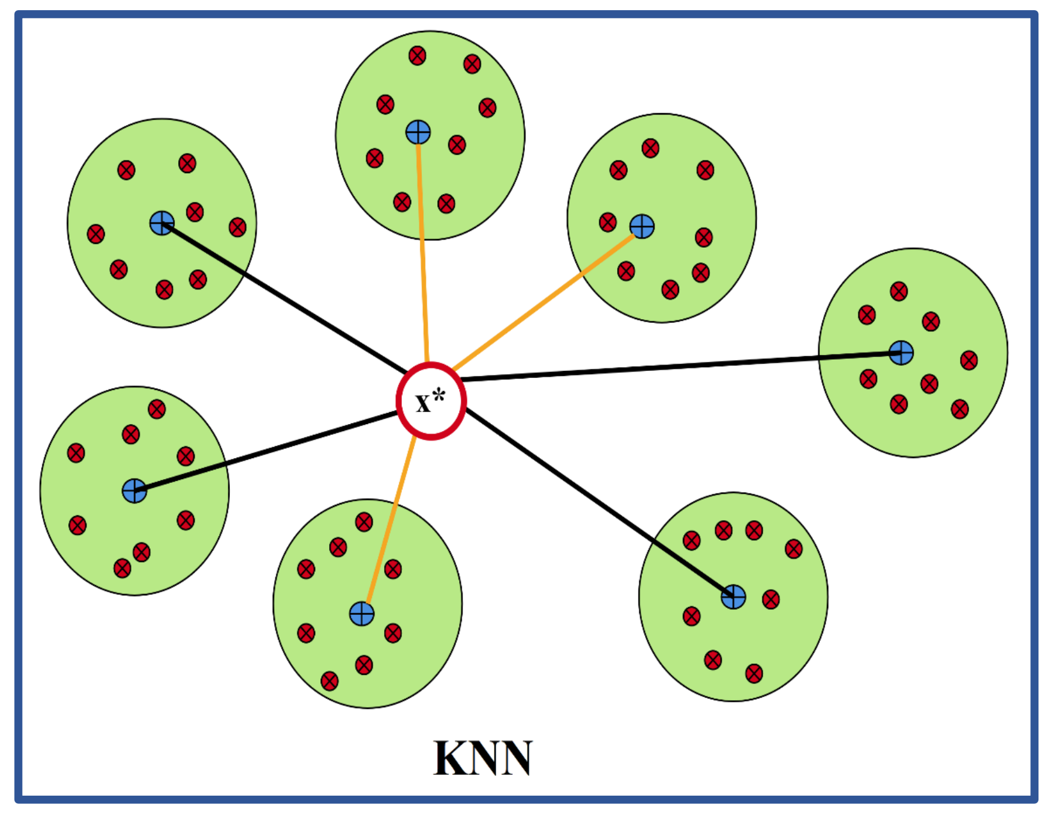
Methodology

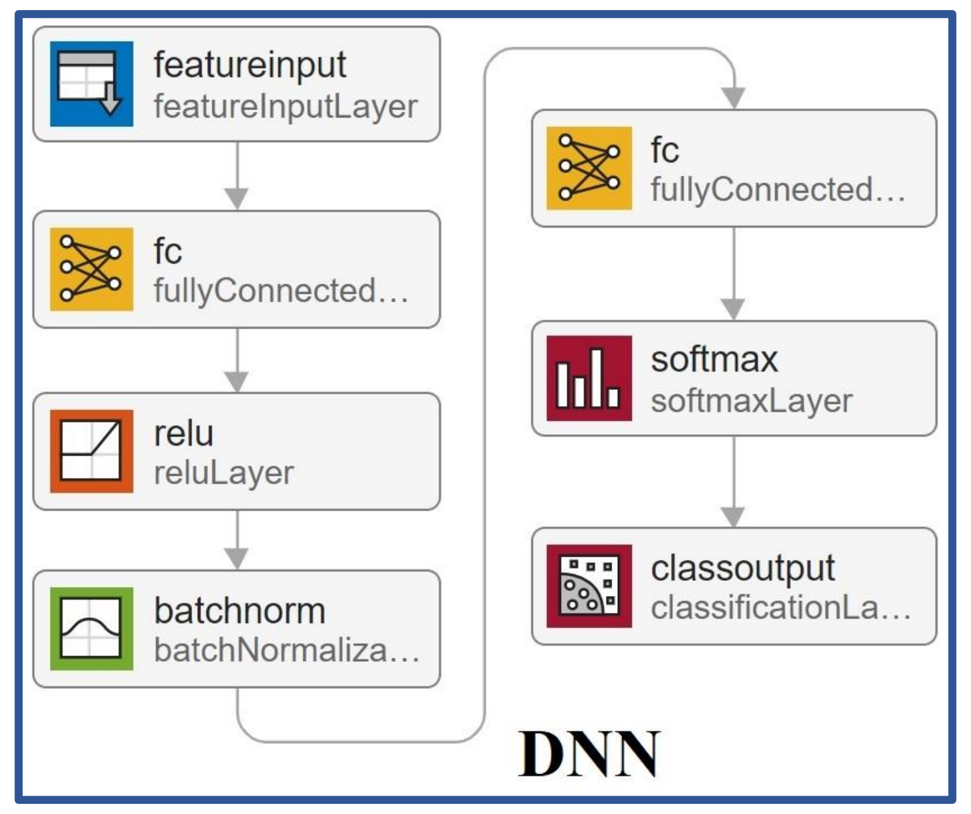
The selection problem for each test point is defined as a multi-label classification task where each instance can be associated with some classes (or experts).

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^*.

KNN: Let $\mathcal{D}' = \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ be the partitions and $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_M\}$ contains the related centroids of the clusters in $\mathcal{D}$. There is a $1 \times M$ vector $dist(x^*, \mathcal{C})$, in which the i'th element is the distance between x^* and $\mathcal{D}_i$, where $dist()$ is a distance metric. The adopted KNN selects K experts with the closest centroids to x^*.

DNN: For each $x_i \in X$, $i = 1, \ldots, N$, $l_i \in \mathcal{L}$ is the related partition label. The new set of points and labels $(X, \mathcal{L})$ is used for training the DNN. For each test point x^*, the network provides 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^*.





Algorithms

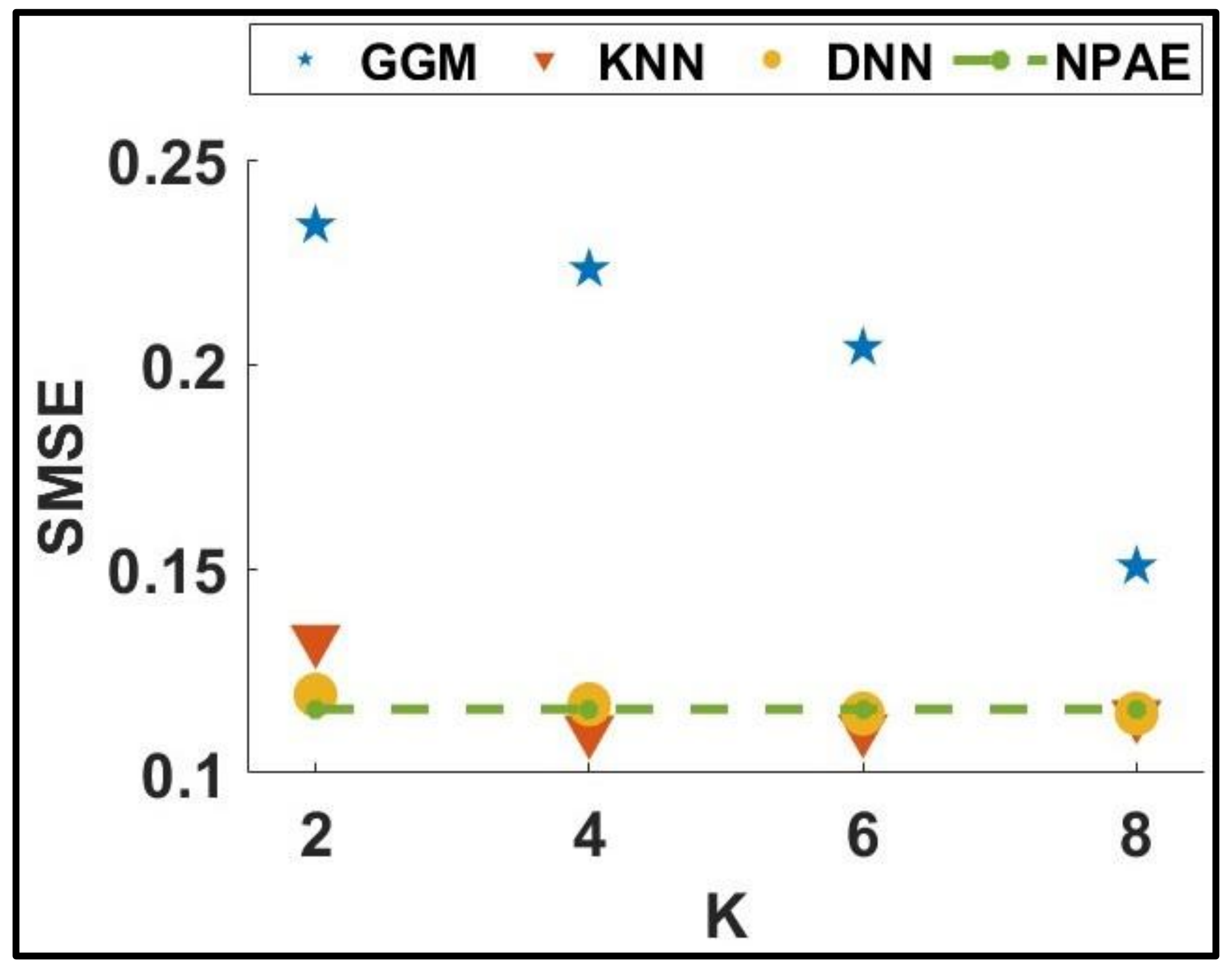
Algorithm Aggregating Dependent Experts Using KNN

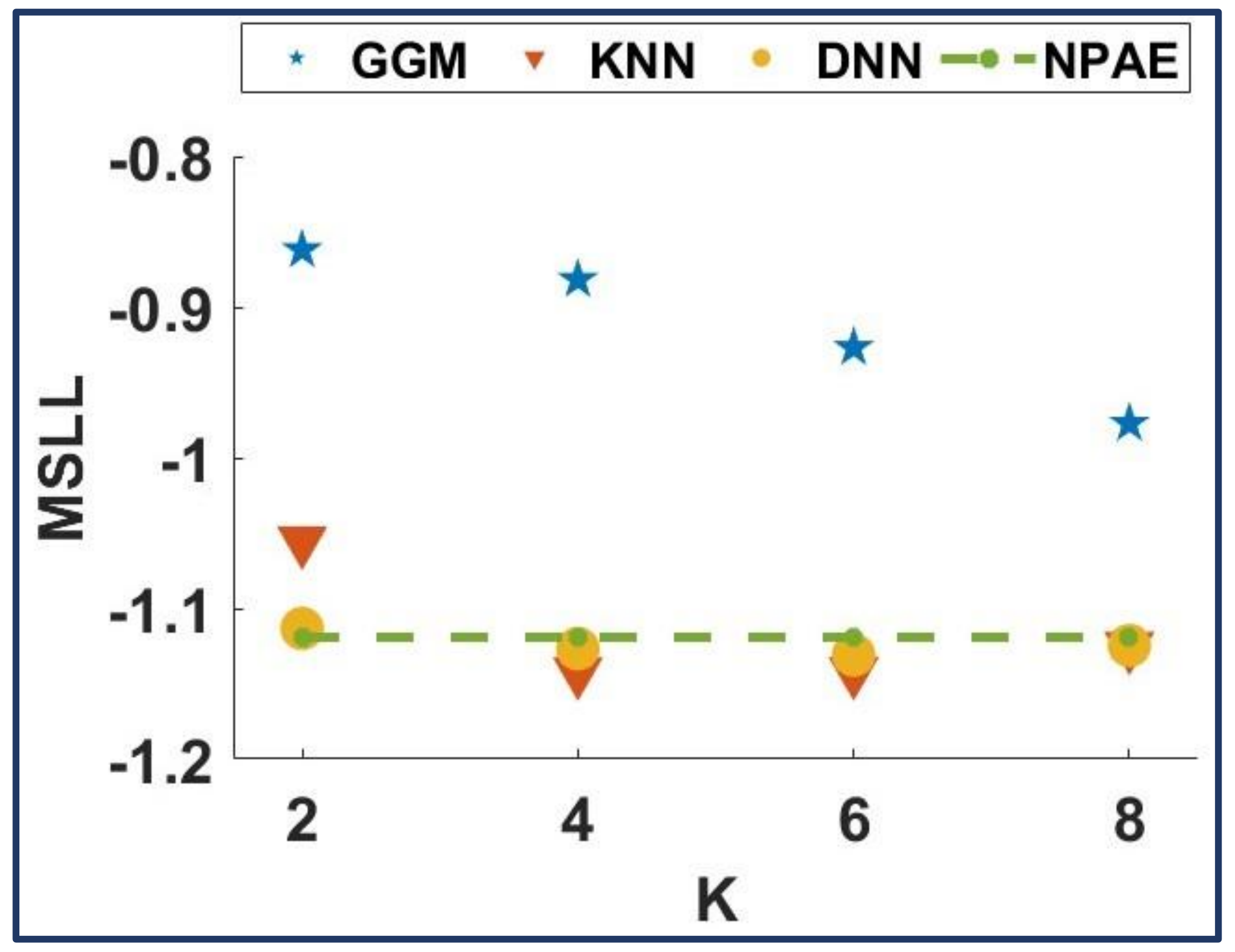
```
Input: Test point $x^* \in X^*$, centroids set $\mathcal{C}$, hyperparameter $K$, Local GPs moments, distance metric.
Output: Aggregated estimator $y_A^*(x^*)$
1: Calculate distance vector for $x^*$, i.e. $dist(x^*, \mathcal{C})$.
2: Sort the elements of $dist(x^*, \mathcal{C})$ ascendingly.
3: Select 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 $\mathcal{M}^{\mathcal{C}}(x^*)$.
5: Aggregate local predictions from Step 4.
```

Algorithm 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 $\mathcal{M}^{\mathcal{C}}(x^*)$ .
7: Aggregate local predictions from Step 6.
```

Experiments





Conclusion:

We proposed a novel approach that converts the GP expert selection into a multi-label classification problem. It

- uses KNN and DNN to assign the proper experts to new data points
- provides a flexible selection method to capture the specific behavior of the new entries
- can be used for CI-based and dependency-based ensemble approaches.