IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 11, 2020, accepted September 20, 2020, date of publication September 24, 2020,
date of current version October 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3026337

Gaussian Process Regression Ensemble Model for Network Traffic Prediction

ABDOLKHALEGH BAYATI, (Member, IEEE), KIM-KHOA NGUYEN, (Member, IEEE), AND MOHAMED CHERIET, (Senior Member, IEEE)
Synchromedia Laboratory, École de Technologie Supérieure, University of Quebec, Montreal, QC H3C 1K3, Canada

Corresponding author: Abdolkhalegh Bayati (abdolkhalegh.bayati.1@ens.etsmlt.ca)

This work was supported in part by Natural Sciences and Engineering Research Council of Canada (NSERC), and in part by Ciena under Project CRDPJ 461084. The work of Mohamed Cheriet was supported by the Canada Research Chair, Tier 1.

ABSTRACT Network traffic prediction is substantial for network optimization and resource management. However, designing an efficient predictive model considering different traffic characteristics, including periodicity, nonlinearity, and nonstationarity, is challenging. Recently, ensemble learning is attracting much attention from researchers in the machine learning community. Although ensemble learning has proven exceptional performance in modelling the intricate problems, it may not be able to handle varying patterns and chaotic behaviour, which are typical properties of traffic data (and many other time-series problems). For this reason, ensemble methods show a limited prediction accuracy in network traffic modelling. We address this issue by proposing an ensemble of learners for time-series prediction that considers the accuracy of individual learners as well as diversity among their outcomes. Each learner contributes to the optimization process by finding the optimal accuracy-diversity balance in a segment of feature space. This divide-and-conquer approach avoids complicated objective functions with many local optimums while fitting the ensemble model on large datasets. Experimental results on the real traffic traces show our proposed method outperforms other state-of-the-art predictors with 12% on average in prediction accuracy for different datasets.

INDEX TERMS Network traffic prediction, machine learning, ensemble learning, Gaussian process regression (GPR), Dirichlet process (DP) clustering.

I. INTRODUCTION

Network traffic prediction is an efficient tool for improving proactive resource scheduling and traffic engineering. It has been utilized to solve various problems such as resource management, mobile data offloading, data-center traffic management, etc. Traffic prediction is challenging because the behaviour of network traffic is affected by many factors including users' behaviour, network protocols, topology, and management policies. Many predictive models have been proposed based on different algorithms including neural networks [1], kernel-based methods [2], time-series models [3], etc. They rely mainly on a single learner to train and forecast the traffic data. Although those models are efficient for specific types of network traffic, they lack flexibility and generalization to capture the complex and varying behaviour exhibited in traffic time-series. This work investigates a new approach based on ensemble learning to fulfill this shortcoming.

Ensemble learning [4] is a recent direction of research in machine learning algorithms in which a group of learners are trained and combined to increase the accuracy and generalization of the prediction. The base learners in an ensemble model can be either the same type of machine learning algorithm or different types of algorithms. The learners are trained on the same training set or different subsets of the training set. The outcome of the ensemble model for an input sample is calculated by combining the prediction results of individual learners on this sample. Combining techniques include voting (for classification), weighted sum (for regression), etc. Many ensemble models have been proposed for classification [5], regression [6], online learning [7], and learning concept drift [8]. Nevertheless, there are a limited number of ensemble models for time-series prediction [9]. In this work, we propose a new ensemble model for time-series prediction and apply it on the complex time-series of network traffic.

The associate editor coordinating the review of this manuscript and approving it for publication was Pengcheng Liu.

An ensemble of learners outperforms any of its members if the individual learners are both diverse and accurate [4]. The *accuracy* of a learner is measured based on the difference between the learner's prediction and the actual target value. Accuracy is a necessary condition to avoid poor learners to obtain the majority of votes. *Diversity* measures the discrepancy among the outputs of the learners. It is required to ensure the learners make uncorrelated errors and achieve better generalization performance [4]. In a diverse ensemble, the learners rectify the prediction errors of each other. Maintaining accuracy and diversity is the cornerstone of ensemble learning. Existing ensemble models employ various techniques to increase these two factors. For example, accuracy has been increased by exploiting effective features, finding appropriate lag (in time-series prediction) [10], and optimal parameters for base learners [9]. Diversity has been increased using various methods such as bagging, boosting, and employing different base learners [4].

Accuracy and diversity are two conflicting objectives [11]. The increase in diversity among learners can lead to a decrease in their accuracy. Finding the optimal balance between accuracy and diversity improves the ensemble prediction performance. Despite this relationship, many ensemble algorithms employ separate techniques to enhance both objectives *independently*. They do not consider the relationship between accuracy and diversity as two conflicting objectives. Therefore, they cannot guarantee the optimal balance between accuracy and diversity to minimize the ensemble prediction error and maximize its generalization. To address this shortcoming, the trade-off between accuracy and diversity must be reflected in the training phase of the ensemble model. This idea forms the basis of our approach to ensemble modelling.

In this work, we propose an ensemble model for traffic time-series prediction which optimizes both the accuracy of learners and diversity between them. During the training phase of the proposed model, the accuracy of learners and diversity between their outputs are adjusted as two parameters to find the global optimum point which minimizes the ensemble prediction error. The control variables that optimize the balance between accuracy and diversity are (i) the number of base learners, (ii) their parameters, and (iii) their training sets. The training phase determines the number of required learners in the ensemble, values of their parameters, and the samples that must be assigned to the training set of each learner.

We designed a divide-and-conquer approach to distribute the process of finding the optimal accuracy-diversity balance between the base learners. It reduces the computational requirements of the training phase and improves the training performance for large datasets. In this approach, each base learner provides a trade-off between accuracy and diversity in a small region of feature space. The feature space is firstly segmented into multiple subspaces, each is assigned to a base learner. Then, each base learner provides the balance between accuracy and diversity in a local area of the feature space. In other words, each base learner maintains the balance between two terms: (i) accuracy on its corresponding segment, and (ii) diversity on the adjacent segments.

We employ Gaussian Process Regression (GPR) as the base learner in our ensemble model. GPR is a kernel-based Bayesian method and a powerful tool for regression and function approximation [12]. In prior work, we showed GPR can handle different traffic characteristics including long/short range dependencies (LRD/SRD), self-similarity, periodicity [13], and multiscale behaviour [14]. The standard GPR is not appropriate to be used in the proposed ensemble model because the Gaussian likelihood in GPR considers only the prediction accuracy. We changed the training phase of GPR to make it compatible with the requirements of our ensemble model by introducing a new likelihood function which considers both accuracy and diversity.

The proposed model has two advantages over the existing time-series ensemble models. First, it optimizes the accuracy-diversity trade-off to improve the prediction performance. Second, it is appropriate for large datasets. The main contributions of the proposed model are as follows.

- Our model considers both accuracy and diversity as two conflicting objectives and finds the optimal balance between them to minimize the ensemble prediction error. This is the main focus of this work.
- We proposed a divide-and-conquer approach to optimizing the balance between accuracy and diversity during the training phase. In this approach, each learner considers a small portion of the training samples in a region of the feature space during the accuracy-diversity optimization. Comparing to the approaches when each learner enhances the accuracy-diversity balance considering the whole training set, our approach leads to an unsophisticated objective function for optimizing the parameters of the base learners. It resolves the issues related to complexity of the problem and allows the model to achieve accuracy-diversity balance for large datasets in a reasonable time.
- We proposed a new *GPR ensemble likelihood function* which improves the the standard GPR to satisfy the requirements of our ensemble model.
- We compared our ensemble model with well-known time-series prediction algorithm such as Long Short-Term Memory (LSTM), autoregressive integrated moving average (ARIMA), fractional autoregressive integrated moving average (FARIMA), Support Vector Regression (SVR), Least absolute shrinkage and selection operator (LASSO), Gradient Tree Boosting (GTB), Random Forest (RF), and Extremely Randomized Trees (ERT) using real traffic datasets.

The remainder of the paper is organized as follows. Section II reviews prior research related to this work. Three main components used in the proposed model (i.e., the concept of ensemble learning, GPR, and clustering algorithm) are explained in Section III. The phases of the proposed GPR ensemble model are detailed in Section IV.

Section V presents our experimental results. Finally, the conclusion is drawn, and we outline the future work.

II. RELATED WORK

Network traffic prediction has been utilized to solve various problems. For example, it has been used in proactive resource management [15], mobile data offloading [16], data-center traffic management [17], optimizing inter-data-center traffic flows [18], and forecasting big-data applications demands [19]. State-of-the-art traffic predictors are based on different machine learning algorithms including neural networks [1], Wavelet transform [18], kernel-based methods [2], time-series analysis [3], and LASSO [20]. ARIMA is a class of statistical models for analyzing and forecasting time-series data that has been used for SRD traffic modelling and prediction [21]. FARIMA is the generalization of the ARIMA model in which non-integer values for the differencing parameter is allowed so that it can capture LRD traffic [22]. LSTM is the result of recent advances in deep learning algorithms. It is a type of Recurrent Neural Networks (RNN) [23] and is famous for its performance in time-series prediction. To the best of our knowledge, all existing traffic models and predictors are based on an individual learner.

In [13], we proposed a GPR-based traffic predictor by designing a covariance function to handle different traffic characteristics such as LRD/SRD, self-similarity, and periodicity. It achieved remarkable results compared to other time-series forecasting methods. However, the standard GPR has two main limitations. First, computational requirements of GPR scale cubically with the number of training samples [24] which is the consequence of the inversion of the covariance matrix. Second, using a stationary covariance function, GPR with lacks the flexibility for modeling the nonstationary data [25]. Both issues are fixed in this work. The problem of cubic time complexity is fixed because each GPR expert is trained over a limited subset of the samples. Thus, the inversion of a large covariance matrix in the standard GPR is substituted with the multiple inversion of small matrices. Also, using multiple GPR experts can easily handle the nonstationarity [25]. These improvements are the result of using a group of GPR learners in an ensemble model instead of an individual GPR.

In this work, we proposed a divide-and-conquer approach to improve GPR's time-complexity and optimize the accuracy-diversity balance. A great example of the divide-and-conquer approach for reducing GPR time complexity is presented in [26]. They studied efficient global optimization (EGO), which employs GPR (also known as Kriging) as a surrogate model to approximate the objective function. Since GPR's time complexity is a real bottleneck in EGO, they proposed to use a GPR approximation called Cluster Kriging that splits a huge data set into several small clusters and improves the GPR time complexity. Another example is a gradient boosting algorithm for approximating a Gaussian process regression (VAGR) proposed in [27]. VAGR sequentially creates random training subsets and approximates the full Gaussian process regression model using the residuals computed from variance-adjusted predictions. VAGR has a time complexity of $O(nm^3)$ for a training dataset of size n and the chosen batch size m.

Recently, ensemble learning has drawn attention of researchers thanks to its promising ability to resolve complex problems. Existing ensemble models employed different base learners including neural network [28], support vector machine (SVM) [29], nearest neighbors classifier [5], and also hybrid of various classifiers [30]. There are well-known ensemble models based on decision trees including Gradient Tree Boosting (GTB), Random Forest (RF), and Extremely Randomized Trees (ERT). GTB is an ensemble of decision trees based on boosting approach [31]. RF algorithm is an ensemble of decision trees, and achieves diversity using random split of the dataset [32]. ERT is an extension of RF in which the randomness of the subsets has been improved [33]. Ensemble models combine a set of learners to solve different problems such as classification [5], regression [6], online learning [7], and learning concept drift [8]. A small portion of studies on ensemble learning focused on the time-series prediction. In [34], a large number of RNN have been generated by training on different sets of examples. An ensemble of support vector regression (SVR) has been designed for prognostics of time-series data in [6].

Accuracy and diversity are two essential criteria to decide the performance of ensemble learning. Unfortunately, the majority of existing time-series ensemble models consider either accuracy or diversity only [9]. Few models consider both criteria. Recently, a layered ensemble architecture (LEA) has been proposed in [9] to address this problem. In LEA, accuracy is obtained by finding an appropriate time lag, and diversity is mainly achieved using different training sets for learners. Unfortunately, LEA does not model a trade-off between accuracy and diversity as two conflicting objectives. Instead, it uses separate techniques to increase both factors independently. This approach cannot guarantee the optimal trade-off between accuracy and diversity of individual learners.

In [11], accuracy and diversity have been formulated to create the multiobjective deep belief networks ensemble (MODBNE). Each learner in MODBNE refines the error of other learners in the cost of reducing its accuracy. In MODBNE, the objective is a function of the output of all the learners for all the training samples. MODBNE is appropriate for small training sets because its complexity increases significantly by the rise in the number of samples and number of learners. The objective function in MODBNE measures the accuracy and diversity considering all the training samples and base learners. For large data sets, it leads to a complicated function with too many local optimums and prevents the ensemble from finding the optimal balance. In our time-series ensemble model, the accuracy-diversity trade-off is calculated locally for each learner which allows avoiding the complexity issues. Each learner considers the samples in a small region of the feature space during accuracy-diversity

TABLE 1. Model notation.

Variable	Description
x_i	feature vector of i-th sample
y_i	label (or target value) of i-th sample
X	the set of feature vectors
Y	the set of labels
$\mathcal{D}$	dataset of samples
d	length of feature vector (i.e., number of features in x)
$k(x_i, x_j, \theta)$	covariance function with hyperparameters θ
K	covariance matrix
o	occupation number in DP clustering algorithm
α	innovation parameter in DP clustering algorithm
$f_m(x_i)$	prediction of learner m for sample x_i
$\hat{f}_{ens}(x_i)$	the final prediction of GPR ensemble model for x_i
$\mathcal{P}$	standard Gaussian likelihood in GPR
$\mathcal{L}$	the proposed GPR ensemble likelihood function
$\mathcal{V}$	the measure of accuracy-diversity balance in $\mathcal{L}$
$\mathcal{E}$	the measure of accuracy in $\mathcal{V}$
$v_m(x_i)$	effect of expert m on the accuracy-diversity balance for x_i
$\mathcal{I}$	the measure of diversity in $\mathcal{V}$
M	total number of experts in the GPR ensemble model

optimization. Since the complexity of objective function (for training each learner) is not affected by the size of the training set, our approach is appropriate for large datasets.

III. BACKGROUND

This section describes the main building blocks of the model including ensemble learning, Gaussian process regression (GPR), and Dirichlet process (DP) clustering.

A. ENSEMBLE LEARNING

An ensemble learning system consists of a set of individual learners where each learner provides an estimate of the target variable. It predicts the target variable by combining the results of all the individual learners. There are three main steps in building an ensemble learning system: (i) ensemble generation, (ii) training each model, and (iii) combining the results [35]. The learners can be the same or different types of machine learning algorithms. The success of the ensemble learning system depends on the accuracy and diversity among the results of the learners [8].

Accuracy is measured based on different metrics such as mean squared error (MSE), normalized mean squared error (NMSE) [13], etc. In an ensemble model with M learners (f_i), diversity over N data points (x_i) can be measured as:

$$\mathcal{I} = \sum_{m=1}^{M} \sum_{i=1}^{N} \left(f_m(x_i) - \bar{f}_{\neg m}(x_i)\right)^2, \tag{1}$$

$$\bar{f}_{\neg m}(x) = \frac{1}{M-1} \sum_{\substack{j=1 \\ j \neq m}}^{M} f_j(x). \tag{2}$$

Equation (1) is the basis for *negatively correlated ensemble learning* [35]. It measures the difference between the prediction result of learner f_m and the average of outcomes of other experts in the ensemble.

B. GAUSSIAN PROCESS REGRESSION (GPR)

We used GPR as the base learner in our ensemble model. According to the definition, a Gaussian process (GP) is a set of random variables that any subset of them has a joint Gaussian distribution. Gaussian Process Regression (GPR) [12] is a supervised machine learning algorithm that provides the mapping function between the input $X = \{x_i\}$ and (continuous) output $Y = \{y_i\}$. Consider n pairs of input and noisy output observations, $\mathcal{D} = \{(x_i, y_i) | i = 0, 1, 2, \ldots, n-1\}$, and the unknown mapping function $f(x_i)$:

$$y_i = f(x_i) + \varepsilon_i, \tag{3}$$

where $\varepsilon_i \sim \mathcal{N}(0, \sigma^2)$ is the independent Gaussian noise, and $f(X) \sim \mathcal{GP}(\bar{m}(X), k(x_i, x_j; \theta))$. $\bar{m}(X)$ denotes the mean function and $k(x_i, x_j; \theta)$ is the arbitrary covariance function with hyperparameters θ. We employ the *Rational Quadratic* (RQ) covariance function defined as [13]:

$$k_{RQ}(x_i, x_j; l, \alpha_c) = s^2 . \left(1 + \frac{(x_i - x_j)^2}{2\alpha_c l^2}\right)^{-\alpha_c}, \ \alpha_c > 0, \ \ l > 0, \tag{4}$$

with the set of hyperparameters $\theta = \{\alpha_c, l, s\}$. In [13], we showed that RQ covariance function can handle traffic characteristics such as LRD/SRD and periodicity. Periodicity is a traffic pattern that is consistently observable at different time scales of traffic data. Thus, using periodic covariance functions to model such behaviour is a well-known approach. For example, in [13], we proposed a *semi-periodic self-similar (SPSS)* covariance function for GPR, capturing LRD/SRD and patterns of periodicity in traffic data. In this work, however, the periodic pattern may not be perceptible by individual learners because the data samples are clustered into smaller subsets. The samples in a cluster are expected to have the same pattern, but they do not necessarily illustrate periodic behaviour. Therefore, we used the RQ covariance function for modelling.

The goal is to predict y_* for previously unobserved sample x_* which does not belong to $\mathcal{D}$. The conditional distribution of $\bar{f}_* = f(x_*)$ is:

$$\bar{f}_* | D, x_*, \theta \sim \mathcal{N}\left(\hat{f}_*, \ V_*^2\right), \tag{5}$$

$$\hat{f}_* = K_*^\top (K + \sigma^2 I)^{-1} Y, \tag{6}$$

$$V_*^2 = k(x_*, x_*; \theta) - K_*^\top (K + \sigma^2 I)^{-1} K_*. \tag{7}$$

Equation (6) provides the predicted value. K is the $n \times n$ *covariance matrix* of X, i.e., $[K]_{ij} = k(x_i, x_j; \ \theta)$. K_* is a $n \times 1$ vector where $[K_*]_i = k(x_i, x_*; \theta)$.

Since the values of hyperparameters have a significant impact on the prediction accuracy, they have to be selected carefully. We used Bayesian inference to estimate the values of hyperparameters. This approach is based on the maximization of the Gaussian likelihood function [12]:

$$p(Y|X, \theta) = \frac{1}{(2\pi)^{n/2} |K|^{1/2}} exp\left(-\tfrac{1}{2} Y^T K^{-1} Y\right). \tag{8}$$

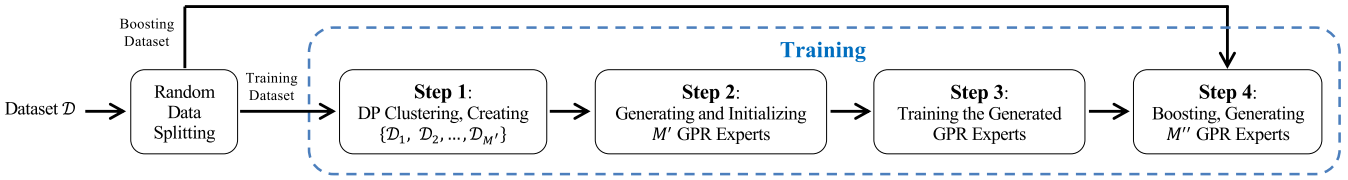


FIGURE 1. Creating and training the GPR ensemble model.

This likelihood function simply obtained by considering $Y \sim \mathcal{N}(0, K + \sigma^2 I)$. Equivalently, we can maximize the *log-likelihood function* [13]:

$$log\left(p(Y|X,\theta)\right) = -\frac{1}{2}Y^\top K^{-1}Y - \frac{1}{2}log(|K|) - \frac{n}{2}log(2\pi) \tag{9}$$

This log-likelihood function considers only the prediction accuracy. We designed a new likelihood function for GPR to satisfy the requirements of our ensemble model regarding the accuracy-diversity optimization.

C. DIRICHLET PROCESS CLUSTERING

A clustering algorithm is required to partition the feature space into multiple subspaces. Any clustering algorithm that satisfies both following requirements can be used. First, the number of subspaces is not known as prior knowledge and depends on the training samples. The clustering algorithm has to determine the number of subspaces automatically. Second, the probability that a new data sample belongs to an existing cluster is required in our ensemble model (in the training and prediction) and the clustering algorithm must be able to estimate such a probability. Dirichlet process (DP) clustering is a natural choice to provide both requirements.

DP has been used in nonparametric Bayesian models [25]. It is a distribution over distributions, i.e., each draw from a Dirichlet process is itself a distribution while the marginal distributions are Dirichlet distributions. A $DP(\alpha, G_0)$ is described by a base distribution G_0 and a positive scalar α, usually indicated as the *innovation parameter*. Consider a sample distribution G drawn randomly from a DP, and random variables c_i sampled from G:

$$G|\alpha, G_0 \sim DP(\alpha, G_0), \tag{10}$$

$$c_i|G \sim G, i = 1 \ldots n. \tag{11}$$

Random variable c_i indicates the cluster of sample x_i.

Assume the indicators (c_i) take M unique values. It can be shown the conditional probability of a single indicator c_i given other indicators when integrating out G and letting M tends to infinity exhibits a clustering effect [36]:

$$p(c_i = m|c_{\neg i}, \alpha) = \frac{o_{\neg i,m}}{M - 1 + \alpha}, \tag{12}$$

$$p(c_i \neq c_j, \quad \forall j \neq i|c_{\neg i}, \alpha) = \frac{\alpha}{M - 1 + \alpha}, \tag{13}$$

where $c_{\neg i}$ is the set of all indicators excluding the ith indicator c_i, and $o_{\neg i,m}$ is the *occupation number* of the cluster m ignoring the value of c_i. In the standard DP, the occupation number is calculated as:

$$o_{\neg i,m} = \sum_{j \neq i} \delta(c_j, m). \tag{14}$$

According to the equations (12) and (13), the indicator c_i belongs to a new cluster (with probability $\frac{\alpha}{M-1+\alpha}$) or the existing cluster m (with probability $\frac{o_{\neg i,m}}{M-1+\alpha}$). The number of clusters is not predetermined, and it depends on the value of α and the number of sample points. A larger value of α induces a higher tendency of creating more clusters, and a smaller value of α leads to a lower number of clusters.

IV. GPR ENSEMBLE MODEL FOR TRAFFIC PREDICTION

This section presents the proposed GPR ensemble model. The algorithm is explained in three phases: preprocessing, training, and prediction. The proposed model requires three steps for being used in real applications. First, the input data (i.e., traffic time-series) must be processed and formatted to create a dataset. Second, the model is trained using the dataset. Finally, the trained model is employed for prediction. For example, the proposed model may be used for proactive resource allocation in data centers. For such an application, we can employ an instance of the trained model in the network controller, where (i) it can access the traffic data on the links, and (ii) its outcome is accessible to the network controller for resource allocation. In preprocessing phase, traffic samples are prepared and structured in a dataset to be used in the training and prediction phases. It normalizes the time-series and eliminates their trends. In the training phase, the GPR learners are generated and trained considering learners' accuracy and their diversity. In the prediction phase, the outputs of base learners for a new sample are combined to create the ensemble prediction. The ensemble prediction is a weighted sum of individual outputs. The input of this model is the time-series of traffic bit-rates (presented as t_i in this section), and the output is the predicted traffic bit-rates (presented as $f_{ens}(x_i)$). The type and source of traffic bit-rates depend on the application. For example, it can be traffic bit-rates on the links (for predicting the link utilization) or traffic load between each pair of points in the network (for predicting the traffic matrix). In our experiments, both traffic types are considered (i.e., CAIDA and Waikato datasets consist of link utilization, but Abilene datasets consist of traffic matrix).

A. PREPROCESSING

In the preprocessing phase, two transformations are applied to the traffic time-series, and then the dataset is created using the transformed samples. The input data of this step includes time-series values shown as t_i. This step's output is a dataset (called $\mathcal{D}$) consisting of a feature vector (presented as x_i) and a target value (shown as y_i) for each data point. The first transformation is known as the *difference* operator which eliminates the nonstationarity and trend in the time-series [37]:

$$r_i = (1 - B)t_i = t_i - t_{i-1}, \tag{15}$$

where t_i is the original traffic sample, and B is the backshift operator. The second transformation is the normalization of input using a sigmoid function:

$$q_i = \frac{2}{1 + e^{-a_n r_i / l_n}} - 1, \tag{16}$$

where l_n is the link capacity, and a_n is a coefficient value that alters the shape of the normalization function. The function (16) maps the input value into the range of $[-1, 1]$.

The transformed traffic data, q_i, is used to create the training dataset $\mathcal{D} = \{(x_i, y_i) | i = 0, 1, 2, \ldots, n-1\}$ in which feature vector x_i (with length d) and label y_i are:

$$x_i = [q_{i-1}, q_{i-2}, \ldots q_{i-d}], \tag{17}$$

$$y_i = q_i. \tag{18}$$

It means for data point i, the target value is q_i, and the feature vector is the d values observed before q_i (i.e., $q_{i-1}, q_{i-2}, \ldots q_{i-d}$) where d is the size of the feature vector.

B. TRAINING

The base learners have to be trained in such a manner that the whole ensemble achieves the optimal balance between the accuracy of base learners and the diversity of their outputs to minimize the prediction error. The ensemble training process must find the optimal balance by tuning two parameters at the same time: accuracy and diversity. In the proposed model, this optimization process is done using a divide-and-conquer approach. Each base learner finds the optimal balance between accuracy and diversity in a small region of the feature space. Hence, the feature space is segmented into smaller subspaces, and each subspace is used as the training set of a base learner. Then, each base learner is trained to provide a balance between the following targets: (i) accuracy on its corresponding segment (or training set), and (ii) diversity on the adjacent segments.

Database $\mathcal{D}$ is employed to train the ensemble model. The samples in $\mathcal{D}$ are divided (randomly) into two groups: *the training dataset*, and *the boosting dataset*. The training dataset is denoted as $\mathcal{D}'$ and includes 80% of samples in $\mathcal{D}$. The training phase is done in four steps as shown in Fig. 1. In Step 1, the training dataset is segmented into M' segments using DP clustering algorithm while the value of M' is determined automatically in DP. In Step 2, M' base learners are generated and their hyperparameters are initialized by maximizing the standard Gaussian likelihood. In Step 3, the base learners are trained to provide the optimal accuracy-diversity balance. Instead of using the standard Gaussian likelihood function, we propose a likelihood function which considers both criteria (accuracy and diversity). In Step 4, M'' base learners are generated and added to the ensemble using the boosting dataset as input. The total number of generated base learners during the four steps of the training phase is $M = M' + M''$. The steps are detailed in the remainder of this section.

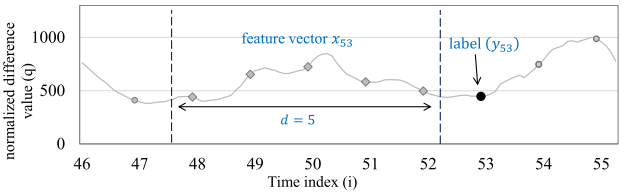


FIGURE 2. An example of creating the training sample $i = 53$ from processed time-series values (q_i). Sample $i = 53$ includes feature vector x_{53} (with length $d = 5$) and target value y_{53}.

1) STEP 1

DP clustering algorithm (explained in Section III-C) is used to partition the training dataset. DP algorithm generates M' clusters $\mathcal{D}_m$ ($m = 1, \ldots, M'$) while each cluster represents a subspace in the feature space. The number of generated clusters is determined automatically during the clustering and can be tuned by changing the innovation parameter α. The goal is to place similar samples (i.e., with similar feature vector) in the same segment. The probability that a new sample belongs to a cluster depends on the similarity of the new sample to the members of the cluster. It is calculated based on the occupation number. Since the occupation number in Equation (14) is not data-dependent [36] we replace it with the following kernel-based function:

$$o_{\neg i,m} = (M - 1)\frac{\sum_{j \neq i} k(x_i, x_j; \theta)\delta(c_j, m)}{\sum_{j \neq i} k(x_i, x_j; \theta)}, \tag{19}$$

$$\delta(c_j, m) = \begin{cases} 1 & \text{if } c_j = m \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

The function $k(x_i, x_j; \theta)$ is an arbitrary covariance function. We used the *dot-product covariance function*:

$$k(x_i, x_j; l) = exp\left(l^{-1} \sum_{k=1}^{d} [x_i]_k \cdot [x_j]_k\right), \tag{21}$$

in which l is the hyperparameter of the covariance function. The element $[x_i]_k$ is the k-th component of the feature vector x_i. It has been shown that dot-product covariance function is effective in the classification problems where the feature values are in the range of $[-1, 1]$ on each dimension [12].

The steps in DP clustering are shown in Algorithm 1. Function $pop(\mathcal{D}')$ in Line 1 removes one sample from $\mathcal{D}'$ and returns the sample. Function $size(\mathcal{D}')$ in Line 2 returns the

Algorithm 1 Step 1 of Training Phase (DP Clustering)

Input: $\mathcal{D}'$, α ▷ Training Dataset
Output: $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_{M'}$ ▷ M' Clusters

1: $M' = 1$, $\mathcal{D}_1 \leftarrow pop(\mathcal{D}')$
2: **while** $size(\mathcal{D}') > 0$ **do**
3: $x_i \leftarrow pop(\mathcal{D}')$
4: $\hat{m} = 1$, $P_{max} = 0$
5: **for** $m = 1$ *to* M' **do**
6: $O_m = occupationNumber(m, x_i)$ ▷ Eq. (19)
7: $P_m = clusteringProbability(O_m, \alpha)$ ▷ Eq. (12)
8: **if** $P_m > P_{max}$ **then**
9: $P_{max} = P_m$, $\hat{m} = m$
10: $P_{new} = newClusterProbability(\alpha)$ ▷ Eq. (13)
11: **if** $P_{max} > P_{new}$ **then**
12: $\mathcal{D}_{\hat{m}} \leftarrow x_i$ ▷ Assign to an existing cluster
13: **else**
14: $M' = M' + 1$
15: $\mathcal{D}_{M'} \leftarrow x_i$ ▷ Create a new cluster

number of sample in the set $\mathcal{D}'$. The occupation number is calculated using Function *occupationNumber* in Line 5 based on Equation (19). The probability of assigning x_i to the existing clusters is calculated in Line 7 using Function *clusteringProbability* according to Equation (12). Function *newClusterProbability* computes the probability of creating a new cluster and assigning x_i to the new cluster based on Equation (13). The output of Algorithm 1 is M' clusters of samples while M' is determined automatically.

2) STEP 2

Base learners f_m ($m = 1, \ldots, M'$) are generated in this step. Cluster $\mathcal{D}_m$ is assigned as the training set of learner f_m. All the learners utilize the RQ covariance function defined in Equation (4) while the hyperparameters of f_m are denoted as θ_m. The hyperparameters of the learners are initialized using standard GPR training process. In standard GPR, the values of hyperparameters θ are determined by maximizing the Gaussian likelihood function $p(Y|X, \theta)$ defined in (8). The learners achieve the best fit on their training set. However, they cannot guarantee the accuracy-diversity balance.

3) STEP 3

In this step, the ensemble is trained to provide the balance between accuracy and diversity. Each base learner optimizes the accuracy-diversity balance in a small region of the feature space. For each base learner, the goal is to compromise between the accuracy on its corresponding subspace and diversity of predictions on its neighbor subspaces. This goal has two sides. On the one hand, it enhances the learner's accuracy which is independent of the samples in other segments and the results of other learners. On the other hand, it improves the diversity which depends on the outputs of other learners. To achieve this goal, a new likelihood function is introduced which considers the accuracy (on the learner's corresponding subspace) and diversity (on the learner's adjacent subspaces). The values of θ_m is selected considering the accuracy of f_m on $\mathcal{D}_m$ and diversity of the learners on the subspaces that are adjacent to $\mathcal{D}_m$.

The values of hyperparameters have been initialized in Step 2. In this step, they are evolved to maximize the proposed *GPR ensemble likelihood function* which considers accuracy and diversity as follows:

$$\mathcal{L}_m = \mathcal{P}_m + \mathcal{V}_m, \tag{22}$$

where $\mathcal{P}_m = log\left(p(Y_{\mathcal{D}_m}|X_{\mathcal{D}_m}, \theta_m)\right)$ is the Gaussian log-likelihood defined in Equation (9), and $\mathcal{V}_m$ measures the diversity:

$$\mathcal{V}_m = \frac{1}{|D_{\neg m}|} \sum_{x_i \in \mathcal{D}_{\neg m}} p_m(x_i) \cdot v_m(x_i), \tag{23}$$

in which $\mathcal{D}_{\neg m}$ is the set of samples x_i in the clusters adjacent to $\mathcal{D}_m$ (excluding $\mathcal{D}_m$) for which the $p_m(x_i) \geq \epsilon$, and $|D_{\neg m}|$ is the number of samples in $D_{\neg m}$. Function $v_m(x_i)$ is calculated as:

$$v_m(x_i) = \frac{1}{2} \cdot \frac{\mathcal{I}_m(x_i) - \mathcal{E}_m(x_i)}{\mathcal{I}_m(x_i) + \mathcal{E}_m(x_i)} + \frac{1}{2}, \tag{24}$$

while $\mathcal{I}_m(x_i)$ is defined based on Equation (1) and measures the distance between estimation of expert m for x_i and $\bar{f}_{\neg m}(x_i)$:

$$\mathcal{I}_m(x_i) = \left(f_{m|\theta_m}(x_i) - \bar{f}_{\neg m}(x_i)\right)^2, \tag{25}$$

where $\bar{f}_{\neg m}(x_i)$ (the average of predicted values of other learners on x_i) is defined based on Equation (2):

$$\bar{f}_{\neg m}(x) = \frac{1}{M-1} \sum_{\substack{j=1 \\ j \neq m}}^{M} p_j(x) f_j(x). \tag{26}$$

Also, $\mathcal{E}_m(x_i)$ is the estimation error of expert m on x_i:

$$\mathcal{E}_m(x_i) = \left(f_{m|\theta_m}(x_i) - y_i\right)^2. \tag{27}$$

$p_m(x_i)$ is the probability of x_i belongs to $\mathcal{D}_m$ based on Equations (12) and (19). In other words, $p_m(x_i)$ measures the similarity of between x_i and $\mathcal{D}_m$ and is defined as:

$$p_m(x_i) = \frac{p(c_i = m|c_{\neg i}, \alpha)}{\sum_{j=1}^{M} p(c_i = j|c_{\neg i}, \alpha)}. \tag{28}$$

Function $p_m(x_i)$ can be assumed as a similarity score between x_i and samples in D_m. When $p_m(x_i)$ is less than a threshold (which means x_i is not similar to cluster m), x_i is excluded from the calculation of $\mathcal{V}_m$. This allows learner m to focus on its local neighbourhood instead of the whole training dataset. The value of threshold is determined based on the minimum value of $p(x)$ for x in cluster m.

The likelihood function $\mathcal{L}_m$ is the sum of two terms. The first term ($\mathcal{P}_m$) is calculated using the samples of subspace m (i.e., $\mathcal{D}_m$) and shows the accuracy of f_m on $\mathcal{D}_m$. The second term ($\mathcal{V}_m$) is computed using the samples of surrounding

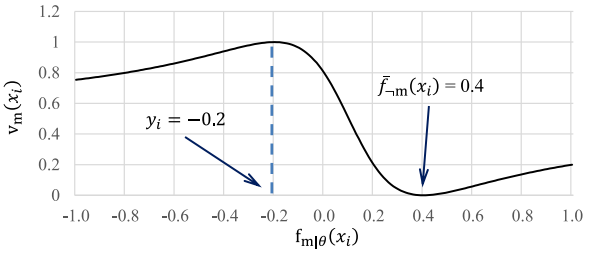
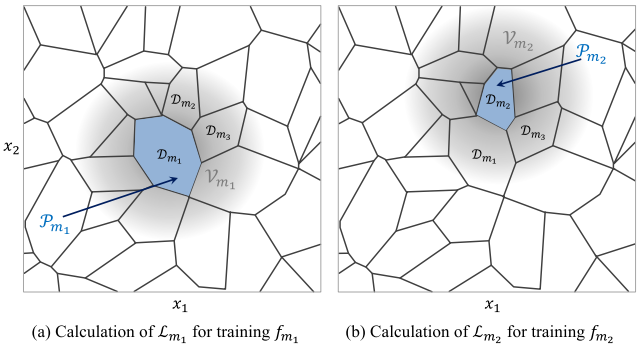


FIGURE 3. An example for function $v_m(x_i)$ for $x_i \notin D_m$ where $y_i = -0.2$ and $\bar{f}_{\neg m}(x_i) = 0.4$. The minimum value of $v_m(x_i)$ happens when $f_{m|\theta}(x_i) = \bar{f}_{\neg m}(x_i)$, and its maximum value occurs when $f_{m|\theta}(x_i) = y_i$.



(a) Calculation of $\mathcal{L}_{m_1}$ for training f_{m_1} (b) Calculation of $\mathcal{L}_{m_2}$ for training f_{m_2}

FIGURE 4. The process of calculating the likelihood function. Likelihood function $\mathcal{L}_m$ is sum of $\mathcal{P}_m$ and $\mathcal{V}_m$ while $\mathcal{P}_m$ is calculated using the samples in $\mathcal{D}_m$, and $\mathcal{V}_m$ is computed using the prediction results of other learners on the adjacent subspaces $\mathcal{D}_{\neg m}$ (the grey area).

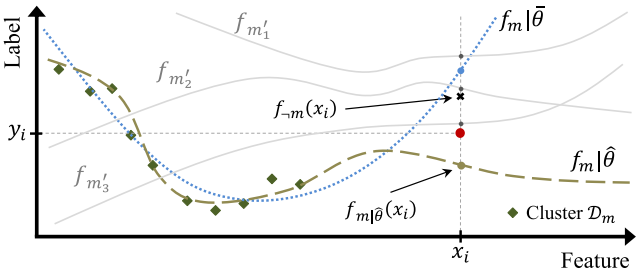


FIGURE 5. Example of hyperparameter optimization for function f_m. The prediction results of other learners (e.g., $f_{m'_1}$, $f_{m'_2}$, $f_{m'_3}$) on the surrounding samples (e.g., x_i) must be considered in the hyperparameter optimization of learner f_m.

subspaces ($\mathcal{D}_{m'}, m' \neq m$). $\mathcal{V}_m$ measures the contribution of $f_m|\theta_m$ to the prediction diversity on other subspaces.

Function $v_m(x_i)$ is in the range of [0, 1]. It is minimized when $\mathcal{I}_m(x_i) = 0$, and maximized when $\mathcal{E}_m(x_i) = 0$. The outcome $\mathcal{E}_m(x_i) = 0$ occurs when there is no difference between the prediction of learner m for x_i and target value y_i (i.e., $f_{m|\theta}(x_i) = y_i$), which means the prediction error of learner m for x_i is zero. The term $\mathcal{I}_m(x_i) = 0$ happens when the prediction of learner m for x_i is equal to the average prediction of other learners (i.e., $f_{m|\theta}(x_i) = \bar{f}_{\neg m}(x_i)$). This is the worst case for the diversity of the learner m when it generates the same outcome as other learners.

We can analyze $v_m(x_i)$ using the following example. Consider x_i as a sample in the neighbourhood of cluster m (but it is not in cluster m) with target value $y_i = -0.2$. The average prediction of other learners for x_i is $\bar{f}_{\neg m}(x_i) = 0.4$. Based on this assumptions, function $v_m(x_i)$ is calculated as:

$$v_m(x_i) = \frac{1}{2}\frac{\left(f_{m|\theta}(x_i) - 0.4\right)^2 - \left(f_{m|\theta}(x_i) + 0.2\right)^2}{\left(f_{m|\theta}(x_i) - 0.4\right)^2 + \left(f_{m|\theta}(x_i) + 0.2\right)^2} + \frac{1}{2}. \quad (29)$$

which is illustrated in Fig. 3. As shown, v_m has a minimum value at $f_{m|\theta}(x_i) = \bar{f}_{\neg m}(x_i)$, and it has a maximum value at $f_{m|\theta}(x_i) = y_i$.

Probability $p_m(x_i)$ limits the calculation of $\mathcal{V}_m$ to the subspaces that are adjacent to $\mathcal{D}_m$. When the distance between x_i and $\mathcal{D}_m$ is high (i.e., x_i is not similar to samples in $\mathcal{D}_m$ which means x_i is not in the surrounding subspaces), $f_{m|\theta_m}(x_i)$ does not affect $\mathcal{V}_m$. Since $p_m(x_i)$ takes small values (close to zero) for samples that are far from $\mathcal{D}_m$, it limits the number of samples that affect the likelihood function of f_m. Since f_m optimizes the accuracy-diversity trade-off in its local neighborhood, the complexity of objective function $\mathcal{V}_m$ (and $\mathcal{L}$) is decreased significantly (compared to an objective function which considers the whole dataset). Also, the complexity of training process for f_m does not depend on the size of the dataset $\mathcal{D}$. Fig. 4 shows this property of our ensemble training. During the training of f_m, $\mathcal{P}_m$ is calculated using the samples in the subspace $\mathcal{D}_m$, and $\mathcal{V}_m$ is calculated based on samples in adjacent subspaces $\mathcal{D}_{\neg m}$. The samples that are not in the adjacent subspaces and the predictions of experts for those samples do not affect $\mathcal{L}_m$.

Fig. 5 gives an intuitive illustration of the accuracy-diversity balance. The samples of one cluster ($\mathcal{D}_m$) are given in a one-dimensional feature space x. The goal is to train f_m (and find the optimal θ_m) in a manner that it provides an optimal accuracy-diversity balance in the regions around $\mathcal{D}_m$. In this example, (x_i, y_i) is a data point in a subspace adjacent to $\mathcal{D}_m$ ($x_i \notin \mathcal{D}_m$). GPR expert f_m has to diminish the prediction error of other learners on x_i by providing diversity. There are three learners ($f_{m'_1}$, $f_{m'_2}$, and $f_{m'_3}$) that have been trained on their subspaces ($\mathcal{D}_{m'_1}$, $\mathcal{D}_{m'_2}$, and $\mathcal{D}_{m'_3}$). So, the values of their hyperparameters and their predicted values for x_i are known. Consider two possible sets of hyperparameters $\bar{\theta}$ and $\hat{\theta}$ for f_m where the outcome of standard Gaussian likelihood function on $\mathcal{D}_m$ is almost the same and maximized for both sets (e.g., they are two local optimums of Equation (8), and $p(Y_{\mathcal{D}_m}|X_{\mathcal{D}_m}, \bar{\theta}) \simeq p(Y_{\mathcal{D}_m}|X_{\mathcal{D}_m}, \hat{\theta})$). It means they result in the same level of accuracy for f_m, so both $\bar{\theta}$ and $\hat{\theta}$ have equal chances to be selected in the standard GPR training. However, $\hat{\theta}$ is more likely to be selected when we employ our GPR ensemble likelihood function. Consider the prediction results of $f_m, f_{m'_1}, f_{m'_2}$, and $f_{m'_3}$ for x_i. It can be shown $f_{m|\hat{\theta}}(x_i)$ leads to higher diversity compared to $f_{m|\bar{\theta}}(x_i)$ according to Equation (1). So, $\hat{\theta}$ is preferred because it provides a higher diversity on x_i. In this example, we considered only the outcome of three learners on sample x_i. In a real scenario, we need to consider all the training samples and base learners in the neighbor segments.

The accuracy-diversity balance is improved after one round of optimizing the experts' hyperparameters within Step 3.

It is possible to enhance the balance by multiple iteration of Step 3.

4) STEP 4

In this step, the samples in the boosting dataset are used to enhance the ensemble performance. First, the generated (and trained) learners are employed to predict the samples in the boosting dataset. The prediction algorithm is explained in Section IV-C. The goal is to select the samples in the boosting dataset that their corresponding prediction error are significant. In other words, the samples that cannot be predicted accurately by the existing learners are selected to form a subset called $\mathcal{D}''$. The prediction error is measured using NMSE [13]:

$$NMSE = \frac{1}{\sigma^2 \, |\mathcal{D}''|} \sum_{(x_i, y_i) \in \mathcal{D}''} (y_i - \hat{f}_{ens}(x_i))^2, \tag{30}$$

where (x_i, y_i) is a sample in $\mathcal{D}''$, $|\mathcal{D}''|$ is the number of samples in $\mathcal{D}''$, σ^2 is the variance of $\{y_i\}$, and $\hat{f}_{ens}(x_i)$ is the voted prediction for x_i (defined in IV-C). The smaller values of NMSE are preferred, and $NMSE = 0$ corresponds to a perfect predictor with no error. $(x_i, y_i) \in \mathcal{D}''$ is added to the boosting dataset if its corresponding prediction error is greater than threshold e_{thr} (i.e., $\frac{1}{\sigma^2}(y_i - \hat{f}_{ens}(x_i))^2 \geq e_{thr}$). The value of e_{thr} is set to 50 percentile of the NMSE of the boosting dataset. Therefore, 50% of the boosting samples (with significant prediction error) will be selected.

The prediction error of generated experts is high for the samples in $\mathcal{D}''$. It will be used to generate and train M'' new learners which will be added to the ensemble to improve the overall prediction accuracy. The new learners are generated as explained in Step 1, Step 2, and Step 3. First, the $\mathcal{D}''$ is clustered using DP algorithm to create M'' clusters (while the value of M'' is determined in DP). Then, M'' learners are generated and initialized using the new clusters. Finally, the learners are trained to optimize the accuracy-diversity by maximizing the likelihood function.

Algorithm 2 implements steps 2, 3, and 4. In Step 2, the hyperparameters of GPR experts are initialized using function *fitStandardGPR* (line 2). It takes the samples in one subspace and optimizes the standard Gaussian likelihood in Equation (8). In Step 3, the accuracy-diversity balance is optimized using function *fitAccuracyDiversity* (line 4). In Step 4, the ensemble model is used to predict the samples in boosting dataset and create $\mathcal{D}''$ (lines 5 to 11). Then, M'' clusters are created by applying DP clustering to $\mathcal{D}''$ (line 12). Each cluster is employed to initialize and train a new GPR expert (lines 13 to 16). Note the difference between inputs of *fitAccuracyDiversity* in lines 4 and 16. In line 4, the input consists of training dataset $\mathcal{D}'$. In line 16, both $\mathcal{D}'$ and $\mathcal{D}''$ are required for the accuracy-diversity balance because the new experts must consider the results of M' previously generated experts on the whole dataset.

Algorithm 2 Steps 2, 3 and 4 of Training Phase

Input: $\mathcal{D}_1, \ldots, \mathcal{D}_{M'}$, boosting data ▷ M'Clusters, Boosting Dataset
Output: $f_1, f_2, \ldots, f_M$ ▷ M GPR Experts

1: **for** $m = 1$ to M' **do** ▷ **Step 2**
2: $f_m = fitStandardGPR(\mathcal{D}_m)$ ▷ Maximize Eq. (8)
3: **for** $m = 1$ to M' **do** ▷ **Step 3**
4: $f_m = fitAccuracyDiversity(f_m, \mathcal{D}')$ ▷ Maximize Eq. (22)
5: **for** $x_j \in$ boosting **do** ▷ **Step 4**
6: $e_j = NMSE(\hat{f}_{ens}(x_j), \ y_j)$ ▷ Prediction error in Eq. (30)
7: $threshold = percentile(e, 70)$
8: $\mathcal{D}'' = \{\}$
9: **for** $x_j \in$ boosting **do**
10: **if** $e_j \geq threshold$ **then**
11: $\mathcal{D}'' \leftarrow x_j$
12: $\{\mathcal{D}''_1, \ldots, \mathcal{D}''_{M''}\} \leftarrow DP(\mathcal{D}'')$ ▷ DP clustering for $\mathcal{D}''$
13: **for** $m = 1$ to M'' **do**
14: $f_{m+M'} = fitStandardGPR(\mathcal{D}''_m)$
15: **for** $m = 1$ to M'' **do**
16: $f_{m+M'} = fitAccuracyDiversity(f_{m+M'}, \mathcal{D}' \cup \mathcal{D}'')$

C. PREDICTION

The final voted prediction for a new sample x_i is the weighted sum of predictions of the GPR components:

$$\hat{f}_{ens}(x_i) = \sum_{m=1}^{M} p_m(x_i) f_m(x_i), \tag{31}$$

where M is the number of generated experts, and $p_m(x_i)$ is defined in Equation (28). The contribution of expert m in the final voted predicted value, $\hat{f}_{ens}(x_i)$, is proportional to $p_m(x_i)$ (i.e., the similarity of x_i to the the samples in the cluster m). The term $p_m(x_i)$ gives strong weights to the experts that have been trained on samples similar to x_i. It reduces the effects of learners that have been trained on clusters which are far from x_i in the feature space. The ensemble prediction is shown in Fig. 6. As shown, the outcome of each learner is multiplied by the probability of sample belongs to the training set of the learner which is calculated using DP clustering algorithm.

D. COMPUTATIONAL COMPLEXITY

In standard GPR, the inverse of covariance matrix K is required to optimize the likelihood function in Equation (8). The time complexity of matrix inversion is $O(N^3)$ where N is the number of training samples. Thus, the standard GPR has a cubic computational requirement.

In our algorithm, two parts must be considered for time complexity analysis (i) DP clustering (in Step 1 of training),

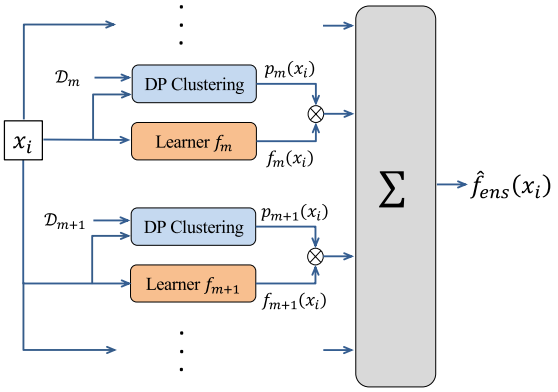


FIGURE 6. The final predicted value is the weighted sum of the individual prediction. The weight of each learner is proportional to the similarity of x_i to the corresponding cluster of the learner.

and (ii) hyperparameter optimization (in Step 2 and Step 3). In DP clustering, a covariance matrix of size $N \times N$ is required for calculation of the occupation number in Equation (19). Each element of the covariance matrix is the result of Equation (21). The time complexity for calculation of such a matrix in DP clustering is in the order of $O(N^2)$.

In steps 2, 3 and 4, the hyperparameters of M learners must be optimized. For expert m, the inverse of covariance matrix of $\mathcal{D}_m$ is needed. The size of $\mathcal{D}_m$ is not predetermined and depends on the DP clustering, training set, and length of feature vector. Assuming $|\mathcal{D}_m| \sim \frac{N}{M}$ (on average), the time complexity of hyperparameter optimization in our algorithm is $O(\frac{N^3}{M^3})$. For small values of M, the time complexity of algorithm is close to the standard GPR. This can be avoided by choosing appropriate values for α. As the number of clusters increases, the average size of clusters decreases. In Section V, the number and size of clusters for different values of α are investigated. It shows that for a wide range of values for α, the time complexity of GPR ensemble model is significantly smaller than standard GPR.

E. GRADIENT DESCENT OPTIMIZATION

The proposed likelihood function is optimized using the gradient descent method, which requires the likelihood function derivative. In this section, we provide the derivative of the likelihood function in Equation (22) with respect to hyperparameters θ:

$$\frac{\partial \mathcal{L}_m}{\partial \theta_m} = \frac{\partial \mathcal{P}_m}{\partial \theta_m} + \frac{\partial \mathcal{V}_m}{\partial \theta_m}. \tag{32}$$

The first part (i.e., $\frac{\partial \mathcal{P}_m}{\partial \theta_m}$) is the derivative of the standard GPR log-likelihood function, and has been investigated in existing GPR models (e.g., [13]):

$$\frac{\partial \mathcal{P}_m}{\partial \theta_m} = \frac{1}{2} Y^\top K^{-1} \frac{\partial K}{\partial \theta_m} K^{-1} Y - \frac{1}{2} Tr\left(K^{-1} \frac{\partial K}{\partial \theta_m}\right), \tag{33}$$

where Tr is the trace of the matrix, and $\frac{\partial K}{\partial \theta_m}$ is the covariance matrix derivative:

$$\left[\frac{\partial K}{\partial \theta_m}\right]_{l,k} = \frac{\partial k(x_l, x_j, \theta_m)}{\theta_m}. \tag{34}$$

The second part of the derivative (i.e., $\frac{\partial \mathcal{V}_m}{\partial \theta_m}$) depends on $f_{m|\theta_m}$, $\mathcal{I}_m$, and $\mathcal{E}_m$. Function $f_{m|\theta_m}(x_*)$ is the prediction of learner m for input x_*, and it is defined in Equation (6):

$$f_{m|\theta_m}(x_*) = \hat{f}_* = K_*^\top (K + \sigma^2 I)^{-1} Y. \tag{35}$$

Its gradient is calculated as:

$$\frac{\partial f_{m|\theta_m}(x_*)}{\partial \theta_m} = \frac{\partial K_*^\top}{\partial \theta_m}(K + \sigma^2 I)^{-1} Y + K_*^\top (K + \sigma^2 I)^{-1} \frac{\partial K^\top}{\partial \theta_m}(K + \sigma^2 I)^{-1} Y. \tag{36}$$

Accordingly, the derivative of $\mathcal{I}_m(x_i)$ and $\mathcal{E}_m(x_i)$ are:

$$\frac{\partial \mathcal{I}_m(x_i)}{\theta_m} = \frac{\partial \left(f_{m|\theta_m}(x_i) - \bar{f}_{\neg m}(x_i)\right)^2}{\theta_m} = 2\left(f_{m|\theta_m}(x_i) - \bar{f}_{\neg m}(x_i)\right) \frac{\partial f_{m|\theta_m}}{\partial \theta_m}, \tag{37}$$

$$\frac{\partial \mathcal{E}_m(x_i)}{\theta_m} = \frac{\partial \left(f_{m|\theta_m}(x_i) - y_i\right)^2}{\theta_m} = 2\left(f_{m|\theta_m}(x_i) - y_i\right) \frac{\partial f_{m|\theta_m}}{\partial \theta_m}, \tag{38}$$

where $\frac{\partial f_{m|\theta_m}}{\partial \theta_m} = \frac{\partial f_{m|\theta_m}(x_i)}{\partial \theta_m}$. The derivative of $\frac{\partial v_m(x_i)}{\theta_m}$ is:

$$\frac{\partial v_m(x_i)}{\theta_m} = 2\frac{\mathcal{E}_m(x_i)\left(f_{m|\theta_m}(x_i) - \bar{f}_{\neg m}(x_i)\right)}{\left(\mathcal{I}_m(x_i) + \mathcal{E}_m(x_i)\right)^2} \frac{\partial f_{m|\theta_m}}{\partial \theta_m} - 2\frac{\mathcal{I}_m(x_i)\left(f_{m|\theta_m}(x_i) - y_i\right)}{\left(\mathcal{I}_m(x_i) + \mathcal{E}_m(x_i)\right)^2} \frac{\partial f_{m|\theta_m}}{\partial \theta_m}. \tag{39}$$

Finally, we can calculate the derivative of $\frac{\partial \mathcal{V}_m}{\theta_m}$:

$$\frac{\partial \mathcal{V}_m}{\theta_m} = \frac{1}{|D_{\neg m}|} \sum_{x_i \in \mathcal{D}_{\neg m}} p_m(x_i) \frac{\partial v_m(x_i)}{\theta_m}. \tag{40}$$

In calculating the derivative, those terms that do not depend on θ are considered constant (e.g., $p_m(x_i)$, $\mathcal{D}_{\neg m}$, $\bar{f}_{\neg m}(x_i)$, and Y).

V. EXPERIMENTAL RESULTS

A. SETUP

We carried out experiments on six different datasets which have been created using traffic samples from three well-known sources of traffic data. The list of datasets and data sources are shown in Table 2. The first two datasets (CAIDA-01, and CAIDA-02) have been created from the traffic data monitored on 10GigE links from 2008 to 2015 provided by the Center for Applied Internet Data Analysis (CAIDA) [38]. Abilene Internet2 Network traffic data [39] (from 2007-01-01 to 2007-10-14) has been used to build the datasets Abilene-01 and Abilene-02. Abilene Internet2 Network is a high-performance backbone network for research

TABLE 2. Datasets of traffic time-series created from different sources at different time-scale.

Dataset	Source of data	Time-scale (minute)
CAIDA-01	CAIDA Anonymized Internet Traces	0.5
CAIDA-02	CAIDA Anonymized Internet Traces	1
Abilene-01	Internet2 Abilene Network	5
Abilene-02	Internet2 Abilene Network	10
Waikato-01	Waikato VIII	15
Waikato-02	Waikato VIII	30

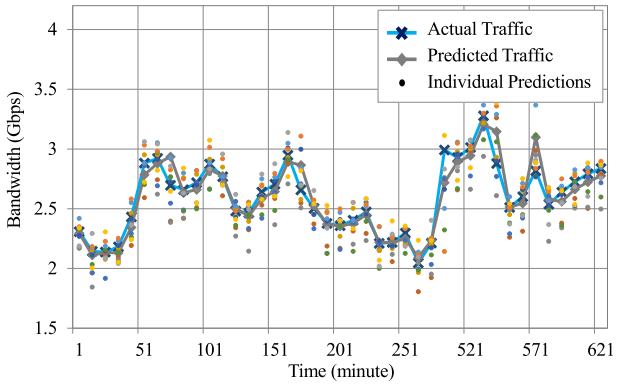


FIGURE 7. Prediction results of the proposed ensemble model (including the individual prediction results and the final predicted values) for traffic time-series at time-scale of 10 minutes (Abilene-02).

and education institutes in the United States. Waikato traffic traces [40] collected from 2011 to 2013 have been exploited to form datasets Waikato-01 and Waikato-02. The time-scales of the traffic samples in the datasets varies from 30 seconds (in CAIDA-01) to 30 minutes (in Waikato-02) as determined in Table 2. These datasets are created based on the traffic bit-rate time-series (in Mbps). Abilene datasets consist of traffic bit-rates between pairs of points (i.e., traffic matrix), but Waikato and CAIDA datasets include traffic loads on the links in the network (i.e., link utilization). The process of preparing the dataset is the same for all these datasets, as explained in Section IV-A. The main difference between these datasets is the time-scale of traffic samples, as illustrated in Table 2. Therefore, the prediction task for these datasets is to estimate target values (i.e., y_i) according to the feature vector (i.e., x_i).

We compared our model with different algorithm including traditional time-series algorithms (i.e., ARIMA [21], FARIMA [22]), supervised regression methods (i.e., standard GPR [13], SVR [6], LASSO [20]), ensemble learning methods (i.e., GTB [31], RF [32], ERT [33]), and a deep learning time-series predictor (i.e., LSTM [23]). These models have been discussed in Section II. We performed separate experiments on the datasets in Table 2. In the experiments, each dataset is divided into two non-overlapped subsets. The first subset is used for the model selection (i.e., selecting the optimal size of the training set, the optimal number of features d, and the optimal values for parameters) for each algorithm using a cross-validation process. The second subset is divided into 100 portions, and each portion is employed to create a pair of the non-overlapped train and test sets (random train-test split). For each pair, the models are fitted on the train set (using the training process explained in Section IV-B) and then, evaluated on the test set. The reported results are the average of 100 prediction error measurements which have been achieved from this process. The prediction error is evaluated using NMSE in Equation (30).

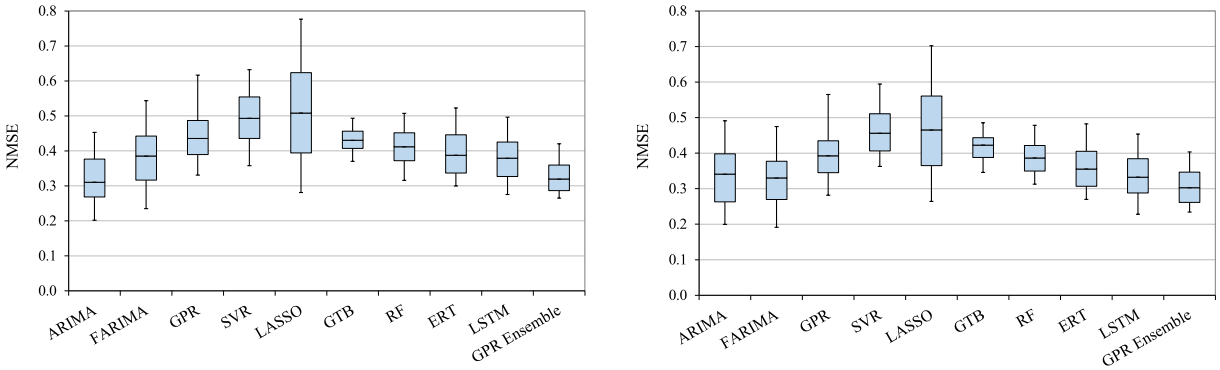
B. RESULTS

In this section, we present the results of our experiments carried out on the datasets. In Fig. 7, an example of the outcome of the GPR ensemble model for predicting the traffic samples in Abilene-02 is presented. Each point is the outcome of one GPR expert. Also, the final predicted values (weighted sum of individual predictions) are illustrated. The individual experts have different errors (diversity), so the error of each expert is corrected by other experts. Thus, the GPR ensemble model was able to predict most of the samples presented in Fig. 7 accurately. For few samples such as $t = 581$, the individual errors are not diverse around the actual value. Therefore, the final prediction error is high compared to other samples. It shows that diversity of individual predictions has an important role in reducing the final prediction error in ensemble model.

The average prediction error of different algorithms are illustrated in Table 3. As shown, the GPR ensemble model outperforms other models in the experiments on CAIDA-02, Abilene-01, Abilene-02, and Waikato-02. In two cases (CAIDA-01 and Waikato-1) ARIMA and LSTM have the smallest prediction error respectively. Nevertheless, GPR ensemble error is close to the winner algorithms in these two cases. The details of results are illustrated in Fig. 8, 9, and 10. The prediction error of different models for datasets CAIDA-01 and CAIDA-02 are shown in Fig. 8. On average, ARIMA has the lowest prediction error for CAIDA-01 compared to other algorithms. GPR ensemble model has the second smallest prediction error. Also, its variance is less than ARIMA. For dataset CAIDA-02, GPR ensemble model has the best prediction results (in term of average prediction error). The performance of FARIMA is close to GPR ensemble model. In both cases, the GPR ensemble model has the minimum variance in prediction which shows it is more stable compared to other models. The average of prediction error of the algorithms for CAIDA datasets is higher than other datasets (see Table 3). It is because of the traffic behaviour at small time-scales. The time-scale of samples in CAIDA-01 and CAIDA-02 are 30 seconds and 1 minute respectively. It has been shown that traffic exhibit short-range dependency (SRD) at small time-scales [41]. The traffic fluctuations are notable and temporal changes in traffic patterns are more frequent at small time-scales. Therefore,

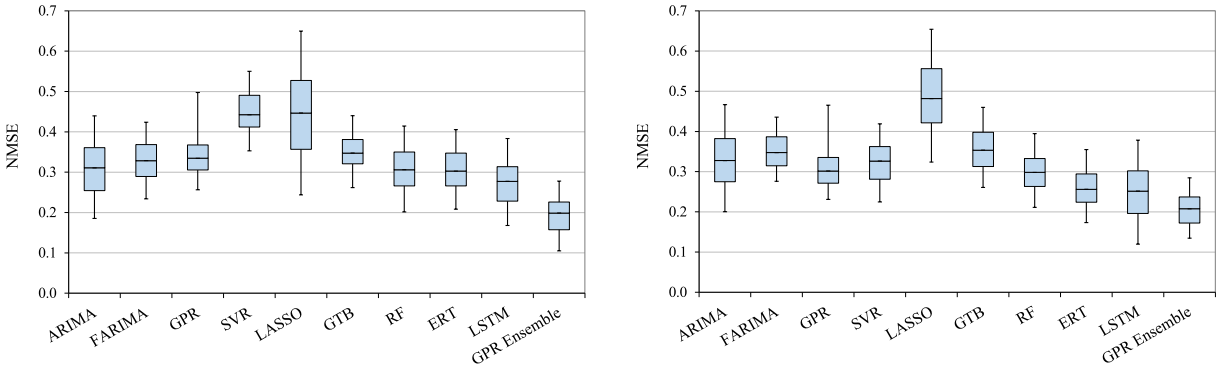
TABLE 3. Average of prediction error of different models on traffic datasets.

	ARIMA	FARIMA	GPR	SVR	LASSO	GTB	RF	ERT	LSTM	GPR Ensemble
CAIDA-01	**0.31**	0.37	0.43	0.49	0.51	0.43	0.41	0.39	0.38	0.33
CAIDA-02	0.33	0.31	0.39	0.47	0.47	0.41	0.39	0.35	0.33	**0.30**
Abilene-01	0.30	0.32	0.33	0.44	0.42	0.35	0.31	0.30	0.26	**0.19**
Abilene-02	0.32	0.35	0.30	0.33	0.48	0.35	0.30	0.26	0.24	**0.21**
Waikato-01	0.28	0.28	0.31	0.25	0.22	0.26	0.18	0.30	**0.16**	0.18
Waikato-02	0.32	0.31	0.29	0.34	0.35	0.31	0.26	0.25	0.15	**0.14**



a) CAIDA-01 b) CAIDA-02

FIGURE 8. Prediction results for CAIDA datasets: (a) CAIDA-01 at time-scale of 30 seconds, (b) CAIDA-02 at time-scale of 1 minute.



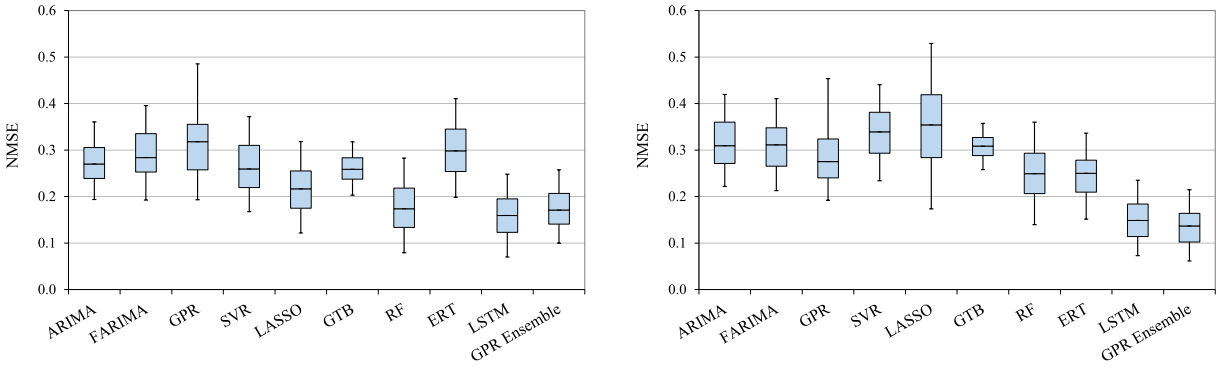
a) Abilene-01 b) Abilene-02

FIGURE 9. Prediction results for Abilene datasets: (a) Abilene-01 at time-scale of 5 minutes, (b) Abilene-02 at time-scale of 10 minutes.

the average of the errors on these datasets is higher for all the algorithms.

The prediction results for datasets Abilene-01, and Abilene-02 are presented in Fig. 9. The time-scales of traffic samples are 5 and 10 minutes for these datasets respectively. In both cases, the GPR ensemble model outperforms other models with average NMSE of 0.19 for Abilene-01, and 0.20 for Abilene-02. Also, the variance of its prediction results is lower than other models. The second smallest prediction error belongs to LSTM with average NMSE of 0.26 and 0.24 for these datasets. Generally, the average of prediction error of all algorithms is smaller for Abilene datasets compared to the results for CAIDA datasets. The results of predictions for Waikato datasets are demonstrated in Fig. 10. In the case of Waikato-01, the average NMSE for LSTM is 0.16 (which is less than other competitors), and the average prediction errors of GPR ensemble model and RF are equal to 0.18. For Waikato-02, the GPR ensemble model has the best prediction result with average NMSE equal to 0.14.

The results of our experiments show the GPR ensemble model has a higher prediction accuracy compared to other ensemble models for many traffic datasets. Also, our model experiences a small variance in its prediction results which shows it is able to handle traffic characteristics and varying patterns. Unlike other models which have different results at various traffic time-scales, GPR ensemble model has a



a) Waikato-01 b) Waikato-02

FIGURE 10. Prediction results for Waikato datasets: (a) Waikato-01 at time-scale of 15 minutes, (b) Waikato-02 at time-scale of 30 minutes.

satisfactory and stable performance at all the traffic time-scales. The high prediction accuracy in GPR ensemble model is achieved by optimizing the balance between accuracy and diversity between prediction results of individual experts during the training phase.

C. LIKELIHOOD OPTIMIZATION

We monitored and analyzed the results for likelihood function convergence during the hyperparameter optimization. An example learner is selected, and its likelihood values (for $\mathcal{P}_m$ and $\mathcal{V}_m$) are tracked. This learner is assigned to a cluster with 166 samples during the training process, and it has 237 samples in its neighbourhood (i.e., samples that $p_m(x)$ is greater than the threshold for them).

The changes in $\mathcal{P}_m$ and $\mathcal{V}_m$ as two terms of the likelihood function are shown in Fig. 12. As shown, the value of $\mathcal{V}_m$ decreased during the first iterations (before iteration 41), while there is a significant improvement for $\mathcal{P}_m$. In the middle of optimization, there is a period that both terms are increased together (between iterations 41 to 51). After iteration 51, it seems $\mathcal{P}_m$ reaches its optimum point and stays stable, while there is a notable gain for $\mathcal{V}_m$. We noted the increase in $\mathcal{V}_m$ (in the final iterations) causes a minor decrease in $\mathcal{P}_m$, but the total likelihood is improving. In general, we observed similar behaviour in other learners. In the first optimization iterations, the hyperparameters are changed to improve $\mathcal{P}_m$ (i.e., improve accuracy on the assigned cluster), and in the final iterations, the hyperparameters are tuned to gain $\mathcal{V}_m$ (i.e., to improve diversity on adjacent samples).

D. EXECUTION TIME

We compare the execution time of different algorithms. As mentioned in Section IV-D, the computational complexity of the proposed algorithm can be affected by the number and size of the clusters. In the worst case, the time complexity of the GPR ensemble model is comparable to the standard GPR model if a cluster's size is close to N (size of the training set). On the other hand, the size and number of clusters depend on the innovation parameter α in DP clustering. Therefore,

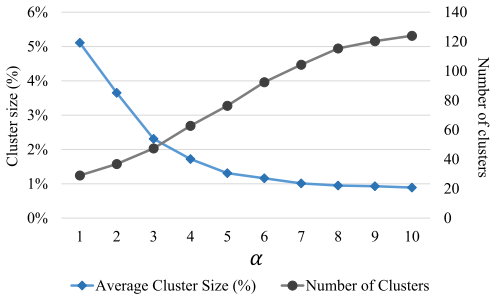


FIGURE 11. Average cluster size and number of clusters in DP clustering algorithm for different values of α when training set of size $N = 10000$.

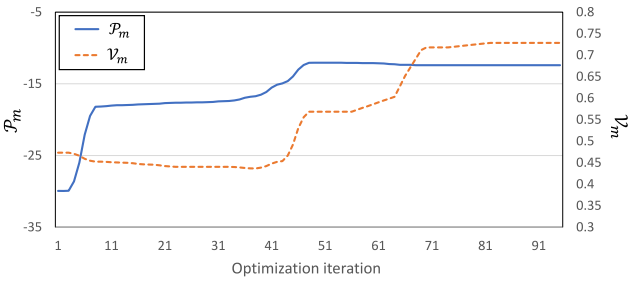


FIGURE 12. The optimization of likelihood function for an example learner that illustrates the changes in $\mathcal{P}_m$ and $\mathcal{V}_m$ during hyperparameter tuning.

we need to investigate the size and number of clusters for different values of α. Fig. 11 shows the average size and number of clusters created by applying the DP algorithm on dataset of size $N = 10000$. DP clustering algorithm has been executed 100 times for each dataset in Table 2 while 10000 samples were selected randomly from the dataset in each run. As shown, the number of clusters is small when the value of α is close to zero, and it increases as α increases. At the same time, the average cluster size decreases as α increases. Even for small values of α, the cluster size is much smaller than the whole training set. For example, the average cluster size is around 5% of the training set (i.e., $\sim$ 500 samples) for $\alpha = 0.5$.

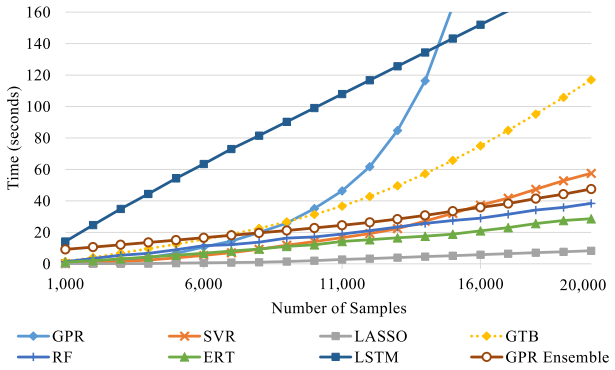


FIGURE 13. Training time of different models with varying training size.

Fig. 13 shows the execution time of different algorithms on varying size training sets. All the experiments are conducted in Python 3.7 on a workstation (Ubuntu Server 18.04.2 LTS, 8 Intel 2.40 GHz processors, 32 GB of RAM). As shown, the training time of standard GPR increases very fast (i.e., cubically) as the training size increases. For dataset of size 2000, it takes 981 seconds to train standard GPR. The training time of LSTM grows very fast (but linearly). LASSO has the shortest training time compared to other models as it is a low performance linear regression model (with high prediction error). The effect of training size on the execution time of the GPR ensemble model is significantly smaller than standard GPR model (and also LSTM and GPT). Since each expert in GPR ensemble model is trained on a segment of feature space, the size of training set has a small effect on the training time of the model. This is the result of local optimization of accuracy-diversity balance in our ensemble training.

VI. CONCLUSION

While network traffic prediction is a powerful means to improve the network performance, it is a very challenging task. The traffic prediction model must handle different traffic patterns at various time-scales. In this article, we presented an algorithm for traffic prediction based on the ensemble of GPR experts, and proposed an optimization framework to optimize the balance between accuracy and diversity of these experts. The proposed framework reduces the complexity through a divide-and-conquer approach where each learner optimizes accuracy-diversity balance in a segment of the feature space, and each segment is used to train a base learner. The proposed GPR ensemble model has been applied on real traffic traces. The experimental results show our GPR ensemble model outperforms other models in traffic prediction. In the future work, we will investigate the performance of our model regarding various traffic characteristics such as burstiness, and LRD/SRD. We intend to integrate the model into existing network controllers to improve the resource and traffic management in networks. We will also compare our model with new deep learning algorithms and graph-based neural networks to show its efficiency.

REFERENCES

[1] F. Morales, M. Ruiz, L. Gifre, L. M. Contreras, V. López, and L. Velasco, "Virtual network topology adaptability based on data analytics for traffic prediction," *J. Opt. Commun. Netw.*, vol. 9, no. 1, p. A35, 2017.

[2] N. Haghighat, M. Nouri, M. G. Shayesteh, and H. Kalbkhani, "Variable bit rate video traffic prediction based on kernel least mean square method," *IET Image Process.*, vol. 9, no. 9, pp. 777–794, Sep. 2015.

[3] S. Kang, S. Lee, Y. Won, and B. Seong, "On-line prediction of nonstationary variable-bit-rate video traffic," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1219–1237, Mar. 2010.

[4] R. Polikar, *Ensemble Machine Learning: Methods and Applications*. Boston, MA, USA: Springer, 2012, ch. 1, pp. 1–34.

[5] S. S. Khan and A. Ahmad, "Relationship between variants of one-class nearest neighbors and creating their accurate ensembles," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1796–1809, Sep. 2018.

[6] J. Liu, V. Vitelli, E. Zio, and R. Seraoui, "A novel dynamic-weighted probabilistic support vector regression-based ensemble for prognostics of time series data," *IEEE Trans. Rel.*, vol. 64, no. 4, pp. 1203–1213, Dec. 2015.

[7] Y. Sun, K. Tang, L. L. Minku, S. Wang, and X. Yao, "Online ensemble learning of data streams with gradually evolved classes," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 6, pp. 1532–1545, Jun. 2016.

[8] L. L. Minku, A. P. White, and X. Yao, "The impact of diversity on online ensemble learning in the presence of concept drift," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 5, pp. 730–742, May 2010.

[9] M. M. Rahman, M. M. Islam, K. Murase, and X. Yao, "Layered ensemble architecture for time series forecasting," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 270–283, Jan. 2016.

[10] W. Yan, "Toward automatic time-series forecasting using neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 7, pp. 1028–1039, Jul. 2012.

[11] C. Zhang, P. Lim, A. K. Qin, and K. C. Tan, "Multiobjective deep belief networks ensemble for remaining useful life estimation in prognostics," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2306–2318, Oct. 2017.

[12] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[13] A. Bayati, V. Asghari, K. Nguyen, and M. Cheriet, "Gaussian process regression based traffic modeling and prediction in high-speed networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–7.

[14] A. Bayati, K. Khoa Nguyen, and M. Cheriet, "Multiple-Step-Ahead traffic prediction in high-speed networks," *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2447–2450, Dec. 2018.

[15] U. Challita, L. Dong, and W. Saad, "Proactive resource management for LTE in unlicensed spectrum: A deep learning perspective," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4674–4689, Jul. 2018.

[16] D. Liu, L. Khoukhi, and A. Hafid, "Prediction-based mobile data offloading in mobile cloud computing," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4660–4673, Jul. 2018.

[17] Y. Zhan, S. M. Ghamkhari, H. Akhavan-Hejazi, D. Xu, and H. Mohsenian-Rad, "Cost-aware traffic management under demand uncertainty from a colocation data center user's perspective," *IEEE Trans. Services Comput.*, early access, Jan. 23, 2018, doi: 10.1109/TSC.2018.2796095.

[18] Y. Li, H. Liu, W. Yang, D. Hu, X. Wang, and W. Xu, "Predicting inter-data-center network traffic using elephant flow and sublink information," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 4, pp. 782–792, Dec. 2016.

[19] Y. Peng, K. Chen, G. Wang, W. Bai, Y. Zhao, H. Wang, Y. Geng, Z. Ma, and L. Gu, "Towards comprehensive traffic forecasting in cloud computing: Design and application," *IEEE/ACM Trans. Netw.*, vol. 24, no. 4, pp. 2210–2222, Aug. 2016.

[20] G. Choudhury, D. Lynch, G. Thakur, and S. Tse, "Two use cases of machine learning for SDN-enabled IP/Optical networks: Traffic matrix prediction and optical path performance prediction [invited]," *J. Opt. Commun. Netw.*, vol. 10, no. 10, p. D52, 2018.

[21] A. Adas, "Traffic models in broadband networks," *IEEE Commun. Mag.*, vol. 35, no. 7, pp. 82–89, Jul. 1997.

[22] Y. Shu, Z. Jin, L. Zhang, L. Wang, and O. W. W. Yang, "Traffic prediction using FARIMA models," in *Proc. IEEE Int. Conf. Commun.*, Jun. 1999, pp. 891–895.

[23] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-C. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.
[24] P. Li and S. Chen, "A review on Gaussian process latent variable models," *CAAI Trans. Intell. Technol.*, vol. 1, no. 4, pp. 366–376, Oct. 2016.
[25] S. Sun and X. Xu, "Variational inference for infinite mixtures of Gaussian processes with applications to traffic flow prediction," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 466–475, Jun. 2011.
[26] H. Wang, B. van Stein, M. Emmerich, and T. Bäck, "Time complexity reduction in efficient global optimization using cluster kriging," in *Proc. Genetic Evol. Comput. Conf.*, Jul. 2017, pp. 889–896.
[27] H.-M. Lu, J.-S. Chen, and W.-C. Liao, "Nonparametric regression via variance-adjusted gradient boosting Gaussian process regression," *IEEE Trans. Knowl. Data Eng.*, early access, Nov. 15, 2019, doi: 10.1109/TKDE.2019.2953728.
[28] J. Yang, X. Zeng, S. Zhong, and S. Wu, "Effective neural network ensemble approach for improving generalization performance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 24, no. 6, pp. 878–887, Jun. 2013.
[29] P. Cheng, S. Wang, J. Ma, J. Sun, and H. Xiong, "Learning to recommend accurate and diverse items," in *Proc. 26th Int. Conf. World Wide Web*, Apr. 2017, pp. 183–192.
[30] Z. Lu, X. Wu, and J. C. Bongard, "Active learning through adaptive heterogeneous ensembling," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 2, pp. 368–381, Feb. 2015.
[31] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *Ann. Statist.*, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.
[32] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, Aug. 1998.
[33] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, Apr. 2006.
[34] M. Assaad, R. Boné, and H. Cardot, "A new boosting algorithm for improved time-series forecasting with recurrent neural networks," *Inf. Fusion*, vol. 9, no. 1, pp. 41–55, Jan. 2008.
[35] A. J. Ferreira and M. A. T. Figueiredo, *Boosting Algorithms: A Review of Methods, Theory, and Applications*. Boston, MA, USA: Springer, 2012, pp. 35–85.
[36] C. E. Rasmussen and Z. Ghahramani, "Infinite mixtures of Gaussian process experts," in *Proc. 14th Int. Conf. Neural Inf. Process. Syst., Natural Synth. NIPS*. Cambridge, MA, USA: MIT Press, 2001, pp. 881–888.
[37] D. Montgomery, C. Jennings, and M. Kulahci, "Introduction to time series analysis and forecasting," in *Wiley Series in Probability and Statistics*. Hoboken, NJ, USA: Wiley, 2015.
[38] *The CAIDA UCSD Anonymized Internet Traces 2010-2015*. Accessed: Jul. 14, 2020. [Online]. Available: http://www.caida.org/data/passive/
[39] *Abilene Internet2 Network*. [Online]. Available: http://noc.net.internet2.edu/i2network/live-network-status/historical-abilene-data.html
[40] (2011). *Waikato VIII*. [Online]. Available: http://wand.net.nz/wits/waikato/8/
[41] W. Willinger, V. Paxson, R. H. Riedi, and M. S. Taqqu, "Chapter: Long-range dependence and data network traffic," in *Book, Theory and Applications of Long-Range Dependence*. Boston, MA, USA: Springer, 2003, pp. 373–407.

ABDOLKHALEGH BAYATI (Member, IEEE) received the bachelor's degree in software engineering from the Shahid Chamran University of Ahvaz, in 2009, and the master's degree in information technology from the Sharif University of Technology, Tehran, in 2012. He is currently pursuing the Ph.D. degree with the Synchromedia Laboratory, Department of System Engineering, University of Quebec, under the supervision of Prof. Mohamed Cheriet. His Ph.D. is focused on the applications of machine learning algorithms in network optimization. He developed predictive models for network traffic and incorporated them into strategies for network optimization. He is also a Senior Data Scientist at Aviva Canada.

KIM-KHOA NGUYEN (Member, IEEE) received the Ph.D. degree in electrical and computer engineering from Concordia University. He served as the CTO at Inocybe Technologies, a leading company in software-defined networking solutions. He was an Architect at Canarie's GreenStar Network and also involved in establishing CSA/IEEE standards for green ICT. He has led research and development in large-scale projects with Ericsson, Ciena, Telus, and InterDigital. He is currently an Associate Professor with the Department of Electrical Engineering, École de technologie supérieure, University of Quebec, Montreal, QC, Canada. He has authored 60 publications and holds several industrial patents. His research interests include network optimization, cloud computing, the Internet of Things (IoT), big data, machine learning, smart city, high-speed networks, and green ICT. He was a recipient of the Microsoft Azure Global IoT Contest Award, in 2017, and the Ciena's Aspirational Prize, in 2018.

MOHAMED CHERIET (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from the University of Pierre and Marie Curie (Paris VI), Paris, France, in 1985 and 1988, respectively. Since 1992, he has been a Professor with the Department of Automation Engineering, École de Technologie Supérieure (ÉTS), University of Quebec, Montreal, QC, Canada, where he was a Full Professor, in 1998. Since 1998, he has been the Founder and the Director of the Synchromedia Laboratory for Multimedia Communication in Telepresence, ÉTS, which targets multimedia communication in telepresence applications. He also co-founded the Laboratory for Imagery, Vision and Artificial Intelligence, ÉTS, where he was the Director, from 2000 to 2006. He is currently an Expert in computational intelligence, pattern recognition, mathematical modeling for image processing, cognitive learning, and machine learning approaches and perception. He has coauthored a book *Character Recognition Systems: A Guide for Students and Practitioners* (Wiley, Spring 2007). His research interests include cloud computing and network virtualization. He has authored or coauthored more than 450 technical articles in this field. He is a Fellow of the International Association for Pattern Recognition, the Canadian Academy of Engineering, and the Engineering Institute of Canada. He was a recipient of the 2016 IEEE J. M. Ham Outstanding Engineering Educator Award and the 2012 Queen Elizabeth II Diamond Jubilee Medal. He is also the Steering Committee Member of the IEEE Green ICT Initiative, the Founder and the Former Chair of the IEEE Montreal Chapter of Computational Intelligent Systems, and the Chair of the IEEE ICT Emissions Working Group. He serves on the editorial boards for several renowned journals and international conferences.