Mining the Situation: Spatiotemporal Traffic Prediction with Big Data

Jie Xu, Dingxiong Deng, Ugur Demiryurek, Cyrus Shahabi, Mihaela van der Schaar

***Abstract*—With the vast availability of traffic sensors from which traffic information can be derived, a lot of research effort has been devoted to developing traffic prediction techniques, which in turn improve route navigation, traffic regulation, urban area planning etc. One key challenge in traffic prediction is how much to rely on prediction models that are constructed using historical data in real-time traffic situations, which may differ from that of the historical data and change over time. In this paper, we propose a novel online framework that could learn from the current traffic situation (or context) in real-time and predict the future traffic by matching the current situation to the most effective prediction model trained using historical data. As real-time traffic arrives, the traffic context space is adaptively partitioned in order to efficiently estimate the effectiveness of each base predictor in different situations. We obtain and prove both short-term and long-term performance guarantees (bounds) for our online algorithm. The proposed algorithm also works effectively in scenarios where the true labels (i.e. realized traffic) are missing or become available with delay. Using the proposed framework, the context dimension that is the most relevant to traffic prediction can also be revealed, which can further reduce the implementation complexity as well as inform traffic policy making. Our experiments with real-world data in real-life conditions show that the proposed approach significantly outperforms existing solutions.**

I. Introduction

Traffic congestion causes tremendous loss in terms of both time and energy wasted. According to a recent report from the Texas Transportation Institute [?], in 2007, 439 metropolitan areas experienced 4.2 billion vehicle-hours of delay, which is equivalent to 2.8 billion gallons in wasted fuel and $87.2 billion in lost productivity, or about 0.7% of the nation's GDP. Traffic congestion is caused when the traffic demand approaches or exceeds the available capacity of the traffic system. In the United States, the Federal Highway Administration [?] [?] has observed that the number of miles of vehicle travel increased by 76 percent from 1980 to 1999, while the total miles of highway increased merely by 1.5 percent, which hardly accommodates growth in travel. It is now generally conceded that it is impossible to build our way out of congestion, mainly because increased capacity results in increased demand. These factors motivate an information-based approach to address these problems.

Fortunately, due to thorough sensor instrumentations of road networks in major cities as well as the vast availability of auxiliary commodity sensors from which traffic information can be derived (e.g., CCTV cameras, GPS devices), a large volume of real-time and historical traffic data at very high spatial and temporal resolutions have become available. Several companies, such as Inrix, now sell both types and at our research center we have had access to both datasets from Los Angeles County for the past three years. As shown by many studies [?] [?] [?] [?], these traffic datasets can be used to predict traffic congestion, which in turn enables drivers to avoid congested areas (e.g., through intelligent navigation systems), policy makers to decide about changes to traffic regulations (e.g., replace a carpool lane with a toll lane), urban planners to design better pathways (e.g., adding an extra lane) and civil engineers to plan better for construction zones (e.g., how a short-term construction would impact traffic).

One major challenge in predicting traffic is how much to rely on the prediction model constructed using historical data in the real-time traffic situation, which may differ from that of the historical data due to the fact that traffic situations are numerous and changing over time. Previous studies showed that depending on the traffic situation one prediction model may be more useful than the other. For example, in [?] it is shown that a hybrid forecasting model that selects in real-time depending on the current situation between an Auto-Regressive Integrated Moving Average (ARIMA) model and a Historical Average Model (HAM) model yields significant better prediction accuracy. It is shown that the ARIMA prediction model is more effective in predicting the speed in normal conditions but at the edges of the rush-hour time (i.e., the beginning and the end of rush hour), the HAM model is more useful. This becomes even more challenging when considering different causes for congestion, e.g., recurring (e.g., daily rush hours), occasional (e.g., weather conditions), unpredictable (e.g., accidents), and temporarily – for short term (e.g., a basketball game) or long term (e.g., road construction) congestions. However there is no holistic approach on when and in which situations to switch from one prediction model to the other for a more effective prediction. The exhaustive method that trains for each traffic situation a prediction model is obviously impractical since it would induce extremely high complexity due to the numerous possible traffic situations.

Our main thesis in this paper is that we try to learn from the current traffic situation in real-time and predict the future traffic by matching the current situation to the most effective

J. Xu and M. van der Schaar are with the Department of Electrical Engineering, University of California, Los Angeles. Emails: jiexu@ucla.edu, mihaela@ee.ucla.edu. J. Xu and M. van der Schaar's research is partially supported by the AFOSR DDDAS grant.

D. Deng, U. Demiryurek and C. Shahabi are with the Department of Computer Science, University of Southern California. Emails: dingxiod@usc.edu, demiryur@usc.edu, shahabi@usc.edu.

prediction model that we constructed using historical data. First, a finite (possibly small) number of traffic predictors are constructed for the same number of representative traffic conditions using historical data. Using a small set of base predictors reduces the training and maintenance costs. Given this set of base predictors, we learn to select the most effective predictor that best suits the current traffic situation in real-time. For instance, suppose we have two traffic predictors trained on historical datasets in different weather conditions, sunny and rainy. We will learn online which predictor to use for prediction in cloudy weather which does not have a predictor trained for it. The basic idea to learn and select the most effective predictor is based on estimating the reward of using a predictor in different situations. The reward estimate is calculated based on how accurate each predictor has been in predicting, say, speed value, given the actual speed values we have observed in the recent past via the real-time data. However, significant challenges still remain as we will explain shortly.

Many features can be used to identify a traffic "situation", which henceforth are called *context*. Example features include: location, time of day, weather condition, number of lanes, area type (e.g., business district, residential) etc. Therefore, the context space is a multidimensional space with D dimensions, where D is the number of features. Since the context space can be very large, learning the most effective predictor in each individual context (i.e. a D-dimensional point in the context space) using reward estimates for this individual context can be extremely slow. For example, there are numerous possible weather conditions (characterized by temperature, humidity, wind speed etc.) but each specific weather condition only appears occasionally in real-time. Thus, we may initially group weather conditions into rough categories such as sunny, rainy, cloudy etc. and then refine each category to improve prediction. However, how to adaptively group contexts and partition the context space poses a significant challenge for fast learning of the best predictor for different traffic contexts. Moreover a rigorous performance characterization of such a method is missing. These are the problems that we are going to solve in this paper.

To evaluate our approach, we obtain and prove both short-term and long-term performance guarantees (bounds) for our online algorithm. This provides not only the assurance that our algorithm will converge over time to the optimal predictor for each possible traffic situation (i.e., there is no loss in terms of the average reward) but also provides a bound for the speed of convergence of our algorithm to the optimal predictor (i.e., our algorithm is fast to converge to the optimal performance). In addition, we conducted a number of experiments to verify our approach with real-world data in real-life conditions. The results show our approach significantly outperforms existing approaches that do not adapt to the varying traffic situations.

The remainder of the paper is organized as follows. Section II reviews the related work and highlights the distinctions of our approach. Section III formulates the traffic prediction problem and defines the performance metric. Section IV describes our context-aware adaptive traffic prediction algorithm. Section V discusses several ways to optimize our algorithm. Section VI reports our experimental results with real world traffic datasets. Section VII concludes the paper.

II. Related work

In this related work section, we first compare our scheme against other existing traffic prediction work (i.e. application-related work) and afterwards we compare our work against various classes of online learning techniques (i.e. algorithm and theory related work).

A. Traffic prediction

Several traffic prediction techniques have been studied in the past. The majority of these techniques focus on predicting traffic in typical conditions (e.g., morning rush hours) [?] [?] [?] [?], and more recently in the presence of accidents, e.g., [?] [?]. Both qualitative [?] and quantitative [?] approaches have been used to measure the impact of an accident on road networks and various machine learning techniques have been applied to predict the typical traffic conditions and the impact of accidents, including Naive Bayesian classifier [?], Decision Tree classifier [?], and Nearest Neighbor classifier [?]. The main differences between our work and the existing studies on traffic prediction are: 1) All existing approaches for traffic prediction aim at predicting traffic in specific traffic situations, e.g. either typical conditions or when accidents occur. Instead, our scheme is applicable to all traffic situations and learns to match the current traffic situation to the best traffic prediction model, by exploiting spatiotemporal and other context similarity information. 2) All existing approaches used for traffic prediction deploy models learned offline (i.e. they rely on a priori training sessions) or they are retrained after long periods and thus, they cannot adapt to (learn from) dynamically changing traffic situations. Instead, our scheme is able to dynamically adapt to the changing traffic situations on the fly and improve the traffic prediction over time as additional traffic data is received. 3) Most previous work is based on empirical studies and does not offer rigorous performance guarantees for traffic prediction. Instead, our scheme is able to provide both short-term and long-term performance bounds.

B. Ensemble learning

Our framework builds a hybrid traffic predictor on top of a set of base predictors and thus, it appertains to the class of ensemble learning techniques. Traditional ensemble schemes [?] [?] for data analysis are mostly focused on analyzing offline datasets; examples of these techniques include bagging [?] and boosting [?]. In the past decade much work has been done to develop online versions of such ensemble techniques. For example, an online version of Adaboost is described in [?]. Another strand of literature on online ensemble learning is represented by prediction with expert advice and the weight update schemes [?] [?] [?] [?] [?]. These algorithms assign weights to experts and make a final prediction by combining the experts' predictions according to the weights. The weights are updated in a manner that may enable regret bounds to be derived. Most of these schemes develop multiplicative update rules [?] [?] [?]. For example, the weighted majority algorithm in [?] decreases the weights of the experts in the pool that disagree with the true label whenever the ensemble makes a mistake. Additive weight update is adopted in [?] where the weights of the experts that predict correctly are increased by a certain amount. In [?], weights of the experts are updated based on stochastic gradient descent. However, none of this work considers the context information when making the prediction (or equivalently, they consider that the context is the same in all time slots). We do consider context information and hence, our benchmark for regret analysis is much tougher. Specifically, in the existing work, the regret is defined with respect to the *context-free* benchmark in which the predictions are all made by the single best predictor ignoring context information. In our paper, the regret is defined with respect to the *context-dependent* benchmark in which the predictions are made by the best predictor conditional on each context. Given any context arrival process, the sum reward obtained by the context-dependent benchmark is greater than that by the context-free benchmark. Thus, even though existing weighted majority type algorithms can achieve a good (e.g. sublinear in time) regret bound against the context-free benchmark, the regret bound will not be sublinear in time when compared against the context-dependent benchmark. In contrast, our algorithm achieves a regret bound that is sublinear in time compared against the context-dependent benchmark, thereby providing both short-term and long-term performance guarantees. When there are several contexts, previous work provides regret bounds on average over all contexts while our work provides regret bounds on each context separately.

C. *Contextual multi-armed bandits*

When establishing the regret bound of the proposed algorithm, we adapted techniques from multi-armed bandit (MAB) problems [?] [?] [?] [?] [?] [?] since techniques used for ensemble learning problems, such as weighted majority type algorithms, lead to weak regret bounds for the considered contextual learning scenario. In our setting the prediction action does not have an explicit impact on reward realization and the learner can observe the realized rewards of all predictors. Hence, the considered problem is not an MAB problem and our algorithm is not an MAB algorithm. In our proposed algorithm, all time slots are equal in terms of the algorithm implementation and operation and there are no exploration or exploitation slots. In principle we could analyze all the time slots in the same way. However, this would lead to weak regret bounds. To get our strong regret bounds, we exploit the fact that we have stronger confidence bounds of the reward estimates in some slots than in others and hence, we use different ways to bound the learning loss in different slots. We divide slots in two types: type-1 slots represent slots for which we can have stronger confidence bounds of the reward estimates while type-2 slots represent slots for which we do not have such strong confidence bounds. Note that this differentiation of slots is very different from the differentiation between exploration and exploitation slots in the MAB literature, which we highlight in Table **??**.

III. Problem Formulation

A. *Problem setting*

Figure **??** illustrates the system model under consideration. We consider a set of locations $\mathcal{L}$ where traffic sensors are deployed. These locations can be either on the highways or arterial streets. We consider an infinite horizon discrete time system $t = 1, 2, ...$ where in each slot t a traffic prediction request from one of the locations $l_o \in \mathcal{L}$ arrives to the system in sequence. Given the current traffic speed $\boldsymbol{x}^t$ at this location, the goal is to predict the traffic speed $\hat{y}^t$ in some predetermined future time, e.g. in the next 15 minutes or in the next 2 hours. Note that the notation t is only used to order the requests according to their relative arrival time. Each request can come from any location in $\mathcal{L}$ at any time in a day, thereby posing a spatiotemporal prediction problem.

Each request is associated with a set of traffic context information which is provided by the road sensors. The context information can include but is not limited to:

- The location context, e.g. the longitude and latitude of the requested location l_o, the location type (highway, arterial way), the area type (business district, residential).
- The time context, e.g. whether on weekday or weekend, at daytime or night, in the rush hour or not, etc.
- The incident context, e.g. whether there is a traffic incident occurred nearby and how far away from l_o, the type of the incident, the number of affected lanes etc.
- Other contexts such as weather (temperature, humidity, wind speed etc.), temporary events etc.

We use the notation $\theta^t \in \Theta$ to denote the context information associated with the t-th request where Θ is a D-dimensional space and D is the number of types of contexts used. Without loss of generality, we normalize the context space Θ to be $[0, 1]^D$. For example, time of day can be normalized with respect to 24 hours.

The system maintains a set of K base predictors $f \in \mathcal{F}$ that can take input of the current speed $\boldsymbol{x}^t$, sent by the road sensors, and output the predicted speed $f(\boldsymbol{x}^t)$ in the predetermined future at location l_o. These base predictors are trained and constructed using historical data for K representative traffic situations before the system operates. However, their performance is unknown for the other traffic situations which are changing over time. We aim to build a hybrid predictor that selects the most effective predictor for the real-time traffic situation by exploiting the traffic context information. Thus, for each request, the system selects the prediction result of one of the base predictors as the final traffic prediction result, denoted by y^t. The prediction result can be consumed by third-party applications such as navigation.

Eventually, the real traffic at the predetermined future for the t-th request, denoted by $\hat{y}^t$, is revealed. We also call $\hat{y}^t$ the ground-truth label for the t-th request. For now we assume that the label is revealed for each request at the end of each prediction. In reality, the label can arrive with delay or even be missing. We will consider these scenarios in Section V. By comparing the system predicted traffic y^t and the true traffic $\hat{y}^t$, a reward r^t is obtained according to a general reward function $r^t = R(y^t, \hat{y}^t)$. For example, a simple

	Existing work (multi-armed bandit)		This work	
	Exploration slot	Exploitation slot	Type-1 slot	Type-2 slot
Algorithm Implementation	(1) Select an under-explored predictor (2) Counters for different predictors are *different*	(1) Select the predictor with the highest reward estimate (2) Counters for different predictors are *different*	(1) Select the predictor with the highest reward estimate (2) Counters for different predictors are the *same*	
Regret Analysis	Weaker confidence on the reward estimate of the *best* predictor	Stronger confidence on the reward estimate of the *best* predictor	Weaker confidence on the reward estimates of *all* predictors	Stronger confidence on the reward estimates of *all* predictors

TABLE I
COMPARISON OF SLOT TYPES WITH MAB ALGORITHMS.

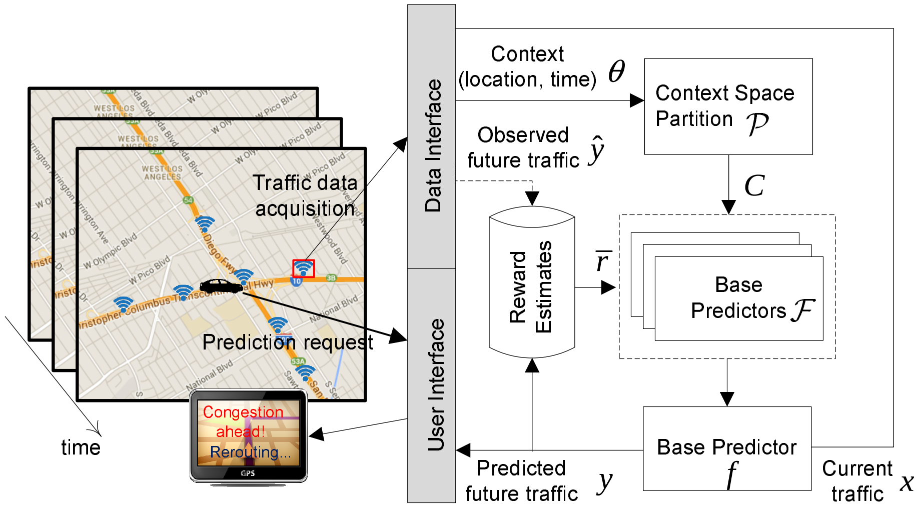
Traffic data acquisition
Prediction request
time
Congestion ahead! Rerouting...
Data Interface
User Interface
Context (location, time) θ
Context Space Partition $\mathcal{P}$
Observed future traffic $\hat{y}$
C
Reward Estimates
$\bar{r}$
Base Predictors $\mathcal{F}$
Base Predictor f
Predicted future traffic y
Current traffic x

Fig. 1. System Diagram.

reward function indicates the accuracy of the prediction, i.e. $R(y^t, \hat{y}^t) = I(y^t = \hat{y}^t)$ where $I(\cdot)$ is the indicator function. The system obtains a reward 1 only if the prediction is correct and 0 otherwise. Other reward functions that depend on how close the prediction is to the true label can also be adopted.

As mentioned, each base predictor is a function of the current traffic $\boldsymbol{x}^t$ which outputs the future traffic prediction y^t. Since for a given $\boldsymbol{x}^t$ the true future traffic $\hat{y}^t$ is a random variable, the reward by selecting a predictor f, i.e. $R(f(\boldsymbol{x}^t), \hat{y}^t)$, is also a random variable at each t. The effectiveness of a base predictor is measured by its expected reward, which depends on the underlying unknown joint distribution of $\boldsymbol{x}^t$ and $\hat{y}^t$. The effectiveness of a base predictor in a traffic context θ is thus its expected reward conditional on θ and is determined by the underlying unknown joint distribution of $\boldsymbol{x}^t$ and $\hat{y}^t$ conditional on the situation θ. Let $\mu_f(\theta) = E\{R(f(\boldsymbol{x}), \hat{y})|\theta\}$ be the expected reward of a predictor f in context θ. However, since the base predictors are constructed using historical data, their expected rewards are unknown a priori for real-time situations which may vary over time. Therefore, the system will continuously revise its selection of base predictors as it learns better and better the base predictors' expected rewards in the current context.

B. Spatiotemporal prediction and multi-predictor diversity gain

By taking into consideration the traffic context information when making traffic prediction, we are exploiting the multi-predictor diversity to improve the prediction performance. To get a sense of where the multi-predictor diversity gain comes from, consider the simple example in Figure **??**, which shows the expected rewards of various base predictors. Since the traffic prediction is a spatiotemporal problem, we use both time of day and location of the traffic as the context information. Given a location 5 miles from the reference location, we have three predictors constructed for three representative traffic situations - morning around 6am, afternoon around 2pm and evening around 7pm. These predictors work effectively in their corresponding situations but may not work well in other time of day contexts due to the different traffic conditions in different times of the day. If we use the same predictor for the entire day, then the average prediction performance can be very bad. Instead, if we use the predictor for traffic situations that are similar to its representative situation, then much better prediction performance can be obtained. However, the challenge is when to use which predictor for prediction since the effectiveness of the base predictors is unknown for every traffic context. For example, the three base predictors (constructed for locations 0 mile, 5 miles and 10 miles from the reference location, respectively, around time 12pm) have complex expected reward curves which need to be learned over time to determine which predictor is the best at different locations.

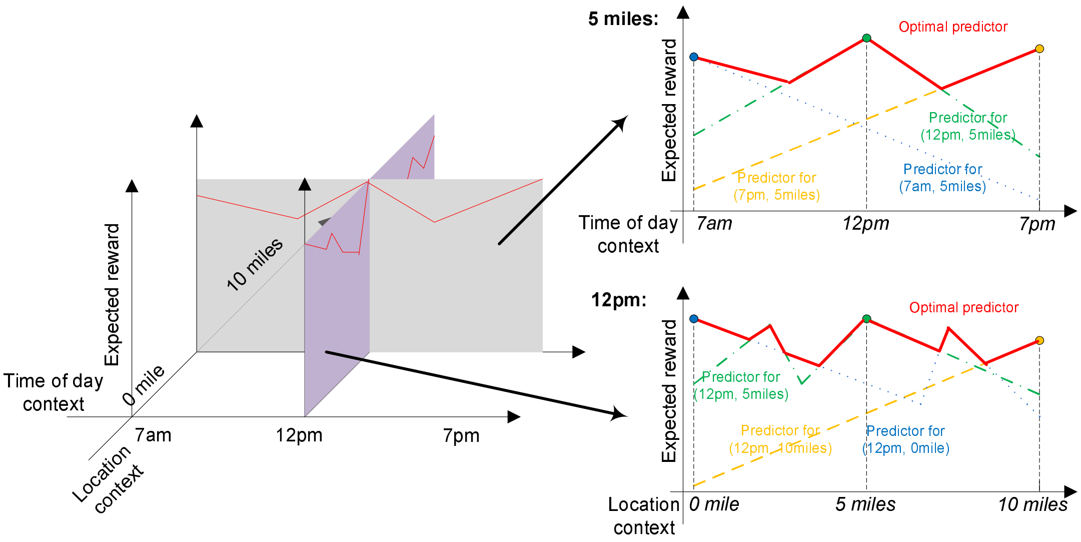

Fig. 2. Spatiotemporal prediction and multi-predictor diversity gain.

C. Performance metric for our algorithm

The goal of our system is to learn the optimal hybrid predictor which selects the most effective base predictor for each traffic situation. Since we do not have the complete knowledge of the performance of all base predictors for all contexts in the online environment, we will develop online learning algorithms that learn to select the best predictors for different traffic contexts over time. The benchmark when evaluating the performance of our learning algorithm is the optimal hybrid predictor that is constructed by an oracle that has the complete information of the expected rewards of all base predictors in all situations. For a traffic context θ, the optimal base predictor selected in the oracle benchmark is

$$f^*(\theta) := \arg\max_{f\in\mathcal{F}} \mu_f(\theta), \forall\theta \in \Theta \qquad (1)$$

Let σ be a learning algorithm and $f^{\sigma(t)}$ be the predictor selected by σ at time t, then the regret of learning by time T is defined as the aggregate reward difference between our learning algorithm and the oracle solution up to T, i.e.

$$\mathrm{Reg}(T) := E\left[\sum_{t=1}^{T} \mu_{f^*(\theta^t)}(\theta^t) - \sum_{t=1}^{T} R(f^{\sigma(t)}(\boldsymbol{x}^t), \hat{y}^t)\right] \qquad (2)$$

where the expectation is taken with respect to the randomness of the prediction, true traffic realization and predictors selected. The regret characterizes the loss incurred due to the unknown transportation system dynamics and gives the convergence rate of the total expected reward of the learning algorithm to the value of the optimal hybrid predictor in (**??**). The regret is non-decreasing in the total number of requests T but we want it to increase as slow as possible. Any algorithm whose regret is sublinear in T, i.e. $\mathrm{Reg}(T) = O(T^q)$ such that $q < 1$, will converge to the optimal solution in terms of the average reward, i.e. $\lim_{T\to\infty} \frac{\mathrm{Reg}(T)}{T} = 0$. The regret of learning also gives a measure for the rate of learning. A smaller q will result in a faster convergence to the optimal average reward and thus, learning the optimal hybrid predictor is faster if q is smaller.

IV. Context-Aware Adaptive Traffic Prediction

A natural way to learn a base predictor's performance in a non-representative traffic context is to record and update its sample mean reward as additional data (i.e. traffic requests and the realized traffic) in the same context arrives. Using such a sample mean-based approach to construct a hybrid predictor is the basic idea of our learning algorithm; however, significant challenges still remain.

On the one hand, exploiting the context information can potentially boost the prediction performance as it provides ways to construct a strong hybrid predictor as suggested in Section III(B). Without the context information, we would only learn the average performance of each predictor over all contexts and thus, a single base predictor would always be selected even though on average it does not perform well. On the other hand, building the optimal hybrid predictor can be very difficult since the context space Θ can be very large and the value space can be continuous. Thus, the sample mean reward approach would fail to work efficiently due to the small number of samples for each individual context θ.

Our method to overcome this difficulty is to dynamically partition the entire context space into multiple smaller context subspaces and maintain and update the sample mean reward estimates for each subspace. This is due to the fact that the expected rewards of a predictor are likely to be similar for similar contexts. For instance, similar weather conditions would have similar impacts on the traffic on close locations. Next, we will propose an online prediction algorithm that adaptively partitions the context space according to the traffic prediction request arrivals on the fly and guarantees sublinear learning regret.

A. Algorithm description

In this subsection, we describe the proposed online context-aware traffic prediction algorithm (CA-Traffic). First we introduce several useful concepts for describing the proposed algorithm.

- **Context subspace**. A context subspace C is a subspace of the entire context space Θ, i.e. $C \subseteq \Theta$. In this paper, we will consider only context subspaces that are created by uniformly partitioning the context space on each dimension, which are enough to guarantee sublinear learning regrets. Thus, each context subspace is a D-dimensional hypercube with side length being 2^{-l} for some l. We call such a hypercube a level-l subspace. For example, when the entire context space is $[0,1]$, namely the context dimension is $D=1$, the entire context space $[0,1]$ is a level-0 subspace, $[0,1/2)$ and $[1/2,1]$ are two level-1 subspaces, $[0,1/4)$, $[1/4,1/2)$, $[1/2,3/4)$, $[3/4,1]$ are four level-2 subspaces etc.
- **Context space partition**. A context space partition $\mathcal{P}$ is a set of non-overlapping context subspaces that cover the entire context space. For example, when $D=1$, $\{[0,1]\}$, $\{[0,1/2),[1/2,3/4),[3/4,1]\}$ are two context space partitions. Since our algorithm will adaptively partition the context space by adaptively removing subspaces from the partition and adding new subspaces into the partition, the context space partition is time-varying depending on the context arrival process of the traffic requests. Initially, the context space partition includes only the entire context space, i.e. $\mathcal{P}^0 = \{\Theta\}$.
- **Active context subspace**. A context subspace C is active if it is in the current context space partition $\mathcal{P}^t$, at time t. For each active context subspace $C \in \mathcal{P}^t$, the algorithm maintains the sample mean reward estimates $\bar{r}_f^t(C)$ for each predictor for the context arrivals to this subspace from time 1 to time t. For each active subspace $C \in \mathcal{P}^t$, the algorithm also maintains a counter M_C^t that records the number of context arrivals to C from time 1 to time t[1].

The algorithm works as follows (see also a formal description in Algorithm). We will describe the algorithm in two parts. The first part (line 3 - 9) is the predictor selection and reward estimates update. When a traffic prediction request comes, the traffic speed vector $\boldsymbol{x}^t$ along with the traffic context information θ^t are sent to the system. The algorithm first checks which active subspace $C \in \mathcal{P}^t$ in the current partition $\mathcal{P}^t$ the context θ^t belongs to (line 3) and the level l of this subspace (line 4). Next, the algorithm activates all predictors and obtains their predictions $f(\boldsymbol{x}^t), \forall f \in \mathcal{F}$ given the input $\boldsymbol{x}^t$ (line 5). However, it selects only one of the predictions as the final prediction y^t, according to the selection as follows (line 6)

$$y^t = \tilde{f}(\boldsymbol{x}^t) \quad \text{where} \quad \tilde{f} = \arg\max_f \bar{r}_f^t(C) \tag{3}$$

In words, the selected base predictor has the highest reward estimate for the context subspace C among all predictors. This is an intuitive selection based on the sample mean rewards. When the true traffic label $\hat{y}^t$ is revealed (line 7), the sample mean reward estimates for all predictors are then updated (line 8) and the counter steps by 1 (line 9).

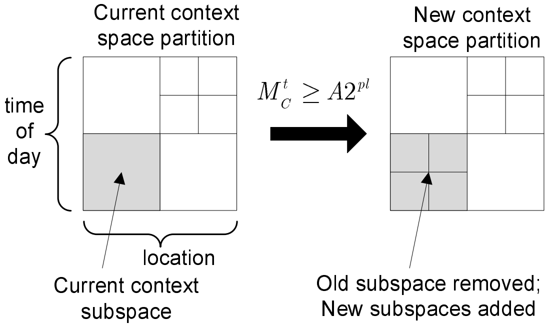
Current context space partition
New context space partition
$M_C^t \geq A2^{pl}$
time of day
location
Current context subspace
Old subspace removed; New subspaces added

Fig. 3. An illustration of the context space partitioning in a 2-dimensional space: the lower left subspace is further partitioned into 4 smaller subspaces because the partition condition is satisfied.

The second part of the algorithm, namely the adaptive context space partitioning, is the key to our algorithm (line 10 - 12). At the end of each slot t, the algorithm decides whether to further partition the current subspace C, depending on whether we have seen sufficiently many request arrivals in C. More specifically, if $M_C^t \geq A2^{pl}$, then C will be further partitioned (line 10), where l is the subspace level of C, $A > 0$ and $p > 0$ are two design parameters. When partitioning is needed, C is uniformly partitioned into 2^D smaller hypercubes (each hypercube is a level-$(l+1)$ subspace with side-length half of that of C). Then C is removed from the active context subspace set $\mathcal{P}^t$ and the new subspaces are added into $\mathcal{P}^t$ (line 11). In this way, $\mathcal{P}^t$ is still a partition whose subspaces are non-overlapping and cover the entire context space. Figure **??** provides an illustrative example of the context space partitioning for a 2-dimensional context space. The current context space partition $\mathcal{P}^t$ is shown in the left plot and the current subspace C is the shaded bottom-left square. When the partitioning condition is satisfied, C is further split into four smaller squares. The context space partitioning process helps refine the learning in smaller subspaces. In the next subsection, we will show that by carefully choosing the design parameters A and p, we can achieve a regret upper bound that is sublinear in time, which implies that the optimal time-average prediction performance can be achieved.

B. Learning regret analysis

In this subsection, we analyze the regret of the proposed traffic prediction algorithm. To enable this analysis, we make a technical assumption that each base predictor achieves similar expected rewards (accuracy) for similar contexts; this is formalized in terms of a Hölder condition.

Assumption. *For each $f \in \mathcal{F}$, there exists $L > 0$, $\alpha > 0$ such that for all $\theta, \theta' \in \Theta$, we have*

$$|\mu_f(\theta) - \mu_f(\theta')| \leq L\|\theta - \theta'\|^\alpha \tag{4}$$

This is a natural and reasonable assumption in traffic prediction problems since similar contexts would lead to

[1] This method requires keeping all request history in memory which can be a concern for some systems. Alternatively, the counters can be restarted after every partition. Our bounds will still hold but the performance of the algorithm will be worse in practice since it does not capitalize on the prior knowledge.

Algorithm Context-aware Traffic Prediction (CA-Traffic)

```
Initialize $\mathcal{P}^0 = \{\Theta\}$, $\bar{r}_f(\Theta) = 0, \forall f \in \mathcal{F}$, $M_\Theta^0 = 0$.
for each traffic prediction request (time slot $t$) do
    Determine $C \in \mathcal{P}^t$ such that $\theta^t \in C$.
    Determine the level $l$ of $C$.
    Generate the predictions results for all predictors $f(\boldsymbol{x}^t), \forall f$.
    Select the final prediction $y^t = \tilde{f}(\boldsymbol{x}^t)$ according to (??).
    The true traffic pattern $\hat{y}^t$ is revealed.
    Update the sample mean reward $\bar{r}_f(C), \forall f$.
    $M_C^t = M_C^t + 1$.
    if $M_C^t \geq A2^{pl}$ then
        $C$ is further partitioned.
    end if
end for
```

similar impact on the prediction outcomes. Note that L is not required to be known and that an unknown α can be estimated online using the sample mean estimates of rewards for similar contexts, and our proposed algorithm can be modified to include the estimation of α.

To obtain sharp bounds on the prediction regret, we divide the time slots into two different types depending on a deterministic control function $\zeta(t) = 2^{2\alpha l}\ln(t)$ where l is the level of the context subspace C that the time-t context belongs to: if $M_C^t \leq \zeta(t)$, then slot t is a *type-1* slot; if $M_C^t > \zeta(t)$, then slot t is a *type-2* slot. The important difference between these two types of slots is that for the type-2 slot, we can have a stronger confidence bound on the estimated rewards of the various predictors for the current context subspace C because we have sufficiently many samples according to the deterministic function. This will help us to derive the regret bound. However, this differentiation of slots is used only in our regret analysis; all slots are equal in terms of the implementation and operation of our algorithm.

Because any time slot is either a type-1 slot or a type-2 slot, the prediction regret therefore can be divided into two parts:

$$\text{Reg}(T) = \text{Reg}_1(T) + \text{Reg}_2(T) \tag{5}$$

where $\text{Reg}_1(T)$, $\text{Reg}_2(T)$ are the regret due to choosing non-optimal predictors in type-1 slots and type-2 slots, respectively. We will bound these two parts separately to get the total regret bound. To do this, we will first investigate the regret incurred for a level-l context subspace and then sum up the regret incurred in context subspaces of all levels. Without loss of generality, we assume that, for any context, the reward difference between the optimal predictor and any non-optimal predictor is bounded by 1.

In Lemma 1, we bound, for any level-l subspace, the regret due to choosing non-optimal predictors in type-1 slots.

Lemma 1. *For every level-l context subspace C, the regret due to choosing non-optimal predictors in type-1 slots is bounded by $2^{2\alpha l}\ln(t)$.*

Proof. Due to the definition of type-1 slot, at any time t, there are no more than $\zeta(t) = 2^{2\alpha l}\ln(t)$ type-1 slots for level-l context subspace. Hence, the regret is bounded above by $2^{2\alpha l}\ln(t)$. □

Next, we bound, for any level-l subspace, the regret due to choosing non-optimal predictors in type-2 slots. To do this, we need to introduce some additional notations. For each subspace C, let f_C^* be the predictor which is optimal for the *center context* in that subspace. Let $\bar{\mu}_{f,C} := \sup_{\theta \in C}\mu_f(\theta)$ and $\underline{\mu}_{f,C} := \inf_{\theta \in C}\mu_f(\theta)$ for any predictor f. For a level-l subspace C, we define the set of *sub-optimal* predictors as

$$\mathcal{S}_{C,l,B} = \{f : \underline{\mu}_{f_C^*,C} - \bar{\mu}_{f,C} > B2^{-\alpha l}\} \tag{6}$$

where $B > 0$ is a constant (which will be determined later) and α is the Hölder condition parameter. For those non-optimal predictors that do not belong to the sub-optimal set, we call them as the *near-optimal* predictors.

Lemma 2. *Assume $2L(\sqrt{D})^\alpha + (2 - B) \leq 0$. Then for every level-$l$ context subspace C, the regret due to choosing non-optimal predictors in type-2 slots is bounded by $K\frac{\pi^2}{3} + AB2^{(p-\alpha)l}$.*

Proof. To bound the regret in type-2 slots, we consider the regret due to choosing sub-optimal and near-optimal predictors separately.

(1) We first bound the regret due to choosing sub-optimal predictors for subspace C, denoted by $\text{Reg}_{s,C}(T)$. Because the maximum loss due to choosing a non-optimal predictor is at most 1 due to normalization, we can bound the probability of choosing sub-optimal predictors instead. Let $\lambda_{f,C}^t$ be the event that a sub-optimal predictor $f \in \mathcal{S}_{C,l,B}$ is selected at time t. Let γ_C^t be the event that the context arrival belongs to C at time t and time slot t is a type-2 slot. The regret due to choosing sub-optimal predictors in type-2 slots when the context belongs to C up to T is bounded above as follows:

$$\begin{aligned} \text{Reg}_{s,C}(T) \quad &\leq \sum_{t=1}^{T} \sum_{f \in \mathcal{S}_{C,l,B}} \Pr(\lambda_{f,C}^t \cap \gamma_C^t) \\ &\leq \sum_{t=1}^{T} \sum_{f \in \mathcal{S}_{C,l,B}} \Pr(\{\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)\} \cap \gamma_C^t) \end{aligned} \tag{7}$$

The second inequality is because, for $\lambda_{f,C}^t$ to occur, it is necessary that $\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)$ occurs. Furthermore, $\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)$ holds only if for any given positive value (denoted by $H_t > 0$), the following joint event occurs,

$$\begin{aligned} &\{\bar{r}_f^t(C) \geq \bar{\mu}_f^t(C) + H_t\} \cup \{\bar{r}_{f_C^*}^t(C) \leq \underline{\mu}_{f_C^*}^t - H_t\} \\ \cup \quad &\{\{\bar{r}_f^t(C) \geq \bar{r}_{f^*}^t(C)\} \cap \{\bar{r}_f^t(C) < \bar{\mu}_f^t(C) + H_t\} \\ &\quad \cap \{\bar{r}_{f_C^*}^t(C) > \underline{\mu}_{f_C^*}^t(C) - H_t\}\} \end{aligned} \tag{8}$$

Therefore, we have

$$\begin{aligned} &\Pr(\{\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)\} \cap \gamma_C^t) \\ \leq \quad &\Pr(\{\bar{r}_f^t(C) \geq \bar{\mu}_f^t(C) + H_t\} \cap \gamma_C^t) \\ &+\Pr(\{\bar{r}_{f_C^*}^t(C) \leq \underline{\mu}_{f_C^*}^t - H_t\} \cap \gamma_C^t) \\ &+\Pr(\{\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)\} \\ &\quad \cap \{\bar{r}_f^t(C) < \bar{\mu}_f^t(C) + H_t\} \\ &\quad \cap \{\bar{r}_{f_C^*}^t(C) > \underline{\mu}_{f_C^*}^t(C) - H_t\} \cap \gamma_C^t) \end{aligned} \tag{9}$$

There are three terms on the right-hand side of the above equation. We want to find conditions on H_t such that the third term equals zero. In this way, the regret can be bounded using only the first two terms. Let $\bar{r}_f^{t,\text{best}}(C)$ denote the reward estimate for a sub-optimal predictor f in the best case (over the subspace C) and the $\bar{r}_{f_C^*}^{t,\text{worst}}(C)$ denote the reward estimate for f_C^* in the worst case (over the subspace C). We define $\bar{r}_f^{t,\text{best}}(C)$ (and similarly $\bar{r}_{f_C^*}^{t,\text{worst}}(C)$) as follows. The reward estimate $\bar{r}_f^t(C)$ can be expressed as

$$\bar{r}_f^t(C) = \sum_{\tau \in \mathcal{E}_f^t(C)} (\mu_f^\tau(\theta^\tau) + \epsilon^\tau) \tag{10}$$

where $\mathcal{E}_f^t(C)$ is the set of slots when the predictor f is selected by time t for contexts in subspace C, $\mu_f^\tau(\theta^\tau)$ is the expected reward by selecting f for the actual context arrived in slot τ and ϵ^τ is the noise in the observed reward in slot τ. Then $\bar{r}_f^{t,\text{best}}(C)$ is defined as

$$\bar{r}_f^{t,\text{best}}(C) = \sum_{\tau \in \mathcal{E}_f^t(C)} (\mu_f^\tau(\theta_f^*(C)) + \epsilon^\tau) \tag{11}$$

where $\theta_f^*(C) = \arg\max_{\theta \in C} \mu_f(\theta)$. $\bar{r}_{f_C^*}^{t,\text{worst}}(C)$ is defined in a similar way. It is then easy to see $\bar{r}_f^{t,\text{best}}(C) - \bar{r}_f^t(C) \leq L(\sqrt{D}/2^l)^\alpha$ and $\bar{r}_{f_C^*}^t(C) - \bar{r}_{f_C^*}^{t,\text{worst}}(C) \leq L(\sqrt{D}/2^l)^\alpha$ by applying the Hölder condition.

The third term can be bounded above as follows:

$$\begin{aligned} & \Pr(\{\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)\} \\ & \quad \cap \{\bar{r}_f^t(C) < \bar{\mu}_f^t(C) + H_t\} \\ & \quad \cap \{\bar{r}_{f_C^*}^t(C) > \underline{\mu}_{f_C^*}^t(C) - H_t\} \cap \gamma_C^t) \\ \leq \; & \Pr(\{\bar{r}_f^{t,\text{best}}(C) \geq \bar{r}_{f_C^*}^{t,\text{worst}}(C)\} \\ & \quad \cap \{\bar{r}_f^{t,\text{best}}(C) < \bar{\mu}_f^t(C) + L(\sqrt{D}/2^l)^\alpha + H_t\} \\ & \quad \cap \{\bar{r}_{f_C^*}^{t,\text{worst}}(C) > \underline{\mu}_{f_C^*}^t(C) - L(\sqrt{D}/2^l)^\alpha - H_t\} \cap \gamma_C^t) \end{aligned} \tag{12}$$

where $L(\sqrt{D}/2^l)^\alpha$ is the possible maximum reward difference between the center context and a border context for a given predictor according to the Hölder condition. The reason is because 1) for $\bar{r}_f^t(C) \geq \bar{r}_{f_C^*}^t(C)$ to occur, it is necessary for $\bar{r}_f^{t,\text{best}}(C) \geq \bar{r}_{f_C^*}^{t,\text{worst}}(C)$ to be true; 2) for $\bar{r}_f^t(C) < \bar{\mu}_f^t(C) + H_t$ to occur, it is necessary for $\bar{r}_f^{t,\text{best}}(C) < \bar{\mu}_f^t(C) + L(\sqrt{D}/2^l)^\alpha + H_t$ to be true due to $\bar{r}_f^{t,\text{best}}(C) - \bar{r}_f^t(C) \leq L(\sqrt{D}/2^l)^\alpha$; 3) for $\bar{r}_{f_C^*}^t(C) > \underline{\mu}_{f_C^*}^t(C) - H_t$ to occur, it is necessary for $\bar{r}_{f_C^*}^{t,\text{worst}}(C) > \underline{\mu}_{f_C^*}^t(C) - L(\sqrt{D}/2^l)^\alpha - H_t$ to be true due to $\bar{r}_{f_C^*}^t(C) - \bar{r}_{f_C^*}^{t,\text{worst}}(C) \leq L(\sqrt{D}/2^l)^\alpha$.

Our objective is to show that right-hand side of (**??**) is zero, thereby implying that the left-hand side is also zero. To show that the right-hand side is zero, we will find a condition under which the following three events,

$$\bar{r}_f^{t,\text{best}}(C) \geq \bar{r}_{f_C^*}^{t,\text{worst}}(C) \tag{13}$$

$$\bar{r}_f^{t,\text{best}}(C) < \bar{\mu}_f^t(C) + L(\sqrt{D}/2^l)^\alpha + H_t \tag{14}$$

$$\bar{r}_{f_C^*}^{t,\text{worst}}(C) > \underline{\mu}_{f_C^*}^t(C) - L(\sqrt{D}/2^l)^\alpha - H_t \tag{15}$$

cannot occur at the same time. Observe the second and third events- if

$$\underline{\mu}_{f_C^*}^t(C) - L(\sqrt{D}/2^l)^\alpha - H_t \geq \bar{\mu}_f^t(C) + L(\sqrt{D}/2^l)^\alpha + H_t, \tag{16}$$

then we must have $\bar{r}_f^{t,\text{best}}(C) < \bar{r}_{f_C^*}^{t,\text{worst}}(C)$. This contradicts $\bar{r}_f^{t,\text{best}}(C) \geq \bar{r}_{f_C^*}^{t,\text{worst}}(C)$. Thus, we next find H_t such that (**??**) holds. Since f is a sub-optimal predictor, we have $\underline{r}_{f_C^*}^t(C) - \bar{\mu}_f^t(C) > B2^{-\alpha l}$. Therefore, (**??**) holds if

$$2L(\sqrt{D}/2^l)^\alpha + 2H_t - B2^{-\alpha l} \leq 0 \tag{17}$$

Let $H_t = 2^{-\alpha l}$ and $B = 2L(\sqrt{D})^\alpha + 2$, the above inequality holds. Therefore, we have found a condition the left-hand side of (**??**) is zero.

Next, we bound the first two terms on the right-hand side of (**??**) by using the Chernoff-Hoeffding bound. Since on the event γ_C^t, the number of samples is greater than $2^{2\alpha l}\ln(t)$, the first term can be bounded as

$$\begin{aligned} & \Pr(\{\bar{r}_f^t(C) \geq \bar{\mu}_f^t(C) + H_t\} \cap \gamma_C^t) \\ \leq \; & e^{-2(H_t)^2 2^{2\alpha l}\ln(t)} = t^{-2} \end{aligned} \tag{18}$$

The last equality is by substituting $H_t = 2^{-\alpha l}$ into the equation. Similarly, for the second term on the right-hand side of (**??**), we can also have

$$\begin{aligned} & \Pr(\{\bar{r}_{f_C^*}^t(C) \leq \underline{\mu}_{f_C^*}^t - H_t\} \cap \gamma_C^t) \\ \leq \; & e^{-2(H_t)^2 2^{2\alpha l}\ln(t)} = t^{-2} \end{aligned} \tag{19}$$

Summing over time and all sub-optimal predictors, we have the

$$\text{Reg}_{s,C}(T) \leq \sum_{t=1}^{T} \sum_{f \in \mathcal{S}_{C,l,B}} 2t^{-2} \leq K\frac{\pi^2}{3} \tag{20}$$

(2) Next we bound the regret due to choosing near-optimal predictors in type-2 slots. Due to the definition of near-optimal predictors, regret due to selecting a near-optimal predictor is at most $B2^{-\alpha l}$. Because there could be at most $A2^{pl}$ slots for a level-l subspace C according to the partitioning rule, the regret of this part is at most $AB2^{(p-\alpha)l}$.

Combining (1) and (2), the regret due to choosing non-optimal predictors in type-2 slots is bounded by $K\frac{\pi^2}{3} + AB2^{l(p-\alpha)}$.

□

Now, we combine the results in Lemma 1 and Lemma 2 to obtain the complete regret bound. The regret depends on the context arrival process and hence, we let $W_l(T)$ denote the number of level-l subspaces that have been activated by time T. Before we derive Theorem 1, we provide a bound on the highest level of active subspace by time T.

Lemma 3. *Given a time T, the highest level of active subspace is at most $\lceil \log_2(T/A)/p \rceil + 1$.*

Proof. It is easy to see that the highest possible level of active subspace is achieved when all requests by time T have the same context. This requires $A2^{pl_{max}} < T$. Therefore, $l_{max} = \lceil \log_2(T/A)/p \rceil + 1$. □

Theorem 1 establishes the regret bound.

Theorem 1. *Assume $p = 3\alpha$ and $B = 2L(\sqrt{D})^{\alpha} + 2$. The regret is upper bounded by*

$$Reg(T) \leq \sum_{l=1}^{\lceil \frac{\log_2(T/A)}{3\alpha} \rceil + 1} W_l(T)(2^{2\alpha l}(\ln(T) + AB) + K\frac{\pi^2}{3}) \tag{21}$$

Proof. Combining the result of Lemma 1 and Lemma 2, it is easy to see that the regret is upper bounded by

$$\begin{aligned} & \text{Reg}(T) \\ \leq & \sum_{l=1}^{\lceil \frac{\log_2(T/A)}{p} \rceil + 1} W_l(T)(2^{2\alpha l}\ln(T) + K\frac{\pi^2}{3} + AB2^{(p-\alpha)l}) \end{aligned} \tag{22}$$

In order to balance the time order of different terms on the right-hand side, we let $p = 3\alpha$. Although choosing p smaller than 3α will not make the regret of a subspace larger, it will increase the number of subspaces activated by time T, causing an increase in the regret. Since we sum over all activated subspaces, it is best to choose p as large as possible. □

The following corollary establishes the regret bound when the context arrivals are uniformly distributed over the entire context space. For example, if the context is the location, then the requests come uniformly from the area $\mathcal{L}$. This is the worst-case scenario because the algorithm has to learn over the entire context space.

Corollary 1. *If the context arrival by time T is uniformly distributed over the context space, we have*

$$\begin{aligned} & Reg(T) \\ \leq & (T/A)^{\frac{D+2\alpha}{D+3\alpha}} 2^{D+2\alpha}(\ln(T) + AB) + (T/A)^{\frac{D}{D+3\alpha}} 2^D K\frac{\pi^2}{3} \end{aligned} \tag{23}$$

Proof. First we calculate the highest level of subspace when context arrivals are uniform. In the worst case, all level l subspaces will stay active and then, they are deactivated until all level $l+1$ subspaces become active and so on. Let l_{max} be the maximum level subspace under this scenario. Because there must be some time $T' < T$ when all subspaces are level l subspaces, we have

$$2^{Dl} A 2^{3\alpha l} < T \tag{24}$$

where 2^{Dl} is the maximum number of level l subspaces and $2^{3\alpha l}$ is the maximum number of time slots that belong to a level l subspace. Thus, we have $l_{max} < \frac{\log_2(T/A)}{D+3\alpha} + 1$. After substituting it into the regret bound in Theorem 1, we get

$$\begin{aligned} & \text{Reg}(T) \\ \leq & \sum_{l=1}^{\frac{\log_2(T/A)}{D+3\alpha}+1} 2^{Dl}(2^{2\alpha l}\ln(T) + K\frac{\pi^2}{3} + AB2^{l(p-\alpha)}) \\ \leq & (T/A)^{\frac{D+2\alpha}{D+3\alpha}} 2^{D+2\alpha}(\ln(T) + AB) + (T/A)^{\frac{D}{D+3\alpha}} 2^D K\frac{\pi^2}{3} \end{aligned} \tag{25}$$

□

We have shown that the regret upper bound is sublinear in time, implying that the average traffic prediction reward (e.g. accuracy) achieves the optimal reward as time goes to infinity. Moreover, it also provides performance bounds for any finite time T rather than the asymptotic result. Ensuring a fast convergence rate is important for the algorithm to quickly adapt to the dynamically changing environment.

V. Extensions

A. Dimension reduction

In the previous section, the context space partitioning is performed on all context dimensions simultaneously. In particular, each context subspace C has a dimension D and each time it is further partitioned, 2^D new subspaces are added into the context space partition $\mathcal{P}$. Thus, learning can be very slow when D is large since many traffic requests are required to learn the best predictors for all these subspaces. One way to reduce the number of new subspaces created during the partitioning process is to maintain the context partition and subspaces and perform the partitioning for each dimension separately. In this way, each time a partitioning is needed for one dimension, only two new subspaces will be created for this dimension. Therefore, at most $2D$ more subspaces will be created for each request arrival.

The modified algorithm works as follows. For each context dimension (e.g. time of day, type and distance), we maintain a similar context space and partition structure as in Section III (in other words the context space dimension is 1 but we have D such spaces). Denote $\mathcal{P}_d^t$ as the context space partition for dimension d and C_d^t as the current context subspace for dimension d, at time t. Note now that since we consider only one dimension, C_d^t is a one-dimensional subspace for each d. Each time a traffic prediction request $\boldsymbol{x}^t$ with context θ^t arrives, we obtain the prediction results of all base predictors given $\boldsymbol{x}^t$. The final prediction y^t is selected according to a different rule than (**??**) as follows

$$y^t = \tilde{f}(\boldsymbol{x}^t) \quad \text{where} \quad \tilde{f} = \arg\max_f \{\max_d \bar{r}_f^t(C_d^t)\} \tag{26}$$

In words, the algorithm selects the predictor that has the highest reward estimate for all current subspaces among all context dimensions. Figure **??** shows an illustrative example for the predictor selection when we only use the time of day and location as the contexts. In this example, the time of day context (10:05am) falls into the subspace at the most left quarter (7am - 11pm) and the location context (3.7 miles away from a reference location) falls into the right half subspace (2.5 - 5 miles). According to the time of day context dimension, the predictor with the highest reward estimate is Predictor 1 while according to the location context dimension, the predictor with the highest reward estimate is Predictor 2. Overall, the best estimated predictor is Predictor 2, which is selected by the algorithm.

After the true traffic $\hat{y}^t$ is observed, the reward estimates for all predictors in all D one-dimensional context subspaces $C_d^t, \forall d$ are updated. The D partitions $\mathcal{P}_d^t, \forall d$ are also updated in a similar way as before depending on whether there have been sufficiently many traffic requests with contexts in the current subspaces. Figure **??** illustrates the context space partition for each individual dimension. In this example, only

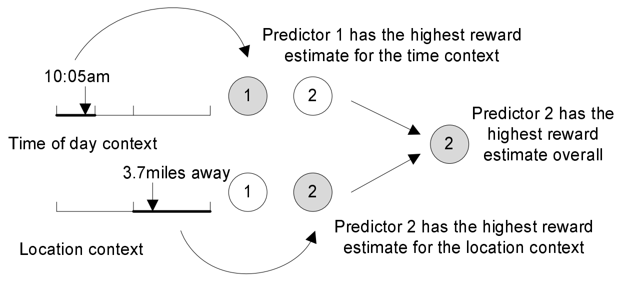
Predictor 1 has the highest reward estimate for the time context
10:05am
1
2
Time of day context
Predictor 2 has the highest reward estimate overall
2
3.7miles away
1
2
Location context
Predictor 2 has the highest reward estimate for the location context

Fig. 4. An illustrative example for predictor selection with separately maintained context partition: a request with context (10:05am and 3.7 miles away from reference location) arrives; Predictor 1 is the best for the time of day context and Predictor 2 is the best for the location context; Predictor 2 is the finally selected predictor

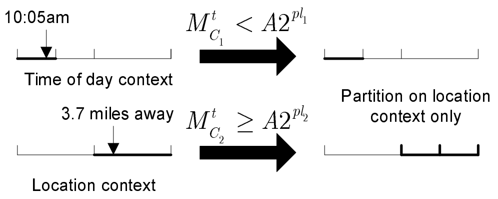

Fig. 5. An illustrative example for context space partition with relevant context: partitioning only occurs on the location context since the partitioning condition is satisfied.

the location context satisfies the partitioning condition and hence its right half subspace is further partitioned.

B. Relevant context dimension

While using all context dimensions will provide the most refined information and thus lead to the best performance, it is equally important to investigate which dimension or set of dimensions is the most informative for a specific traffic situation. The benefits of revealing the most relevant context dimension (or set of dimensions) are manifold, including reduced cost due to context information retrieval and transmission, reduced algorithmic and computation complexity and targeted active traffic control. In the extreme case, a context dimension (e.g. time of day) is not informative at all if for all values of the context along this dimension, the best traffic predictor is the same. Hence, having this context dimension does not add benefits for the traffic prediction but only incurs additional cost.

For expositional clarity, in the following we will focus only on the most relevant context. The extension to the k most relevant context dimensions ($\forall k < D$) is straightforward. Let $\mu_f(\theta_d)$ be the expected prediction reward of predictor $f \in \mathcal{F}$ when the context along the d-th dimension is θ_d and $f^*(\theta_d) = \arg\max_f \mu_f(\theta_d)$ be the predictor with the highest expected reward given θ_d. Then the expected reward if we only use the d-th dimension context information is $R_d = E_{\theta_d}\{\mu_{f^*(\theta_d)}(\theta_d)\}$ where the expectation is taken over the distribution of the d-th dimension context. The most relevant context dimension is defined to be $d^* = \arg\max_d R_d$.

Our framework can be easily extended to determine the most relevant context dimension. For each dimension, we maintain the similar partition and subspace structure as in Section III (with $D = 1$). In addition, we maintain the time-average prediction reward $\bar{R}^t_d$ for each dimension d. The estimated most relevant dimension at time t is thus $(d^*)^t = \arg\max_d \bar{R}^t_d$.

Theorem 2. *The estimated most relevant dimension converges to the true most relevant dimension, i.e.* $\lim_{t\to\infty} (d^*)^t = d^*$.

Proof. Since for each dimension d, the time-average regret tends to 0 as $t \to \infty$, the time-average reward also $\bar{R}^t_d \to R_d$ as $t \to \infty$. Therefore, the most relevant dimension can also be revealed when $t \to \infty$. □

C. Missing and delayed feedback

The proposed algorithm requires the knowledge of the true label $\hat{y}^t$ on the predicted traffic to update reward estimates of different predictors so that their true performance can be learned. In practice, the feedback about true traffic label $\hat{y}^t$ can be missing or delayed due to, for example, delayed traffic reports and sensors being down temporarily. In this subsection, we can make small modifications to the proposed algorithm to deal with such scenarios.

Consider the case when the feedback is missing with probability p_m. The algorithm can be modified so that it updates the sample mean reward and performs context space partitioning only for requests in which the true label is revealed. Let $\text{Reg}^m(T)$ denote the regret of the modified algorithm with missing feedback, we have the following result.

Proposition 1. *Suppose the feedback about the true label is missing with probability* p_m*, we have*

$$\begin{aligned} & \mathit{Reg}^m(T) \\ \leq & \sum_{l=1}^{\lceil \frac{\log_2(T/A)}{3\alpha} \rceil + 1} W_l(T)(2^{2\alpha l}(\tfrac{1}{1-p_m}\ln(T) + AB) + K\tfrac{\pi^2}{3}) \end{aligned} \tag{27}$$

Proof. Missing labels cause more type-1 slots to learn the performance of base predictors accurately enough. In expectation, $\frac{1}{1-p_m} - 1$ more type-1 slots are required in ratio. Hence, the regret incurred in type-1 slots increases to $\frac{1}{1-p_m}$ of before. The regret incurred in type-2 slots is not affected since the control function $\zeta(t)$ ensures that the reward estimates are accurate enough. Using the original regret bound and taking into account the increased regret incurred in type-1 slots, we obtain the new regret bound. □

Consider the case when the feedback is delayed. We assume that the true label of the request at t is observed at most L_{max} slots later. The algorithm is modified so that it keeps in its memory the last L_{max} labels and the reward estimates are updated whenever the corresponding true label is revealed. Let $\text{Reg}^d(T)$ denote the regret of the modified algorithm with delayed feedback. We then have the following result

Proposition 2. *Suppose the feedback about the true label is delayed by at most* L_{max} *slots, then we have*

$$\mathit{Reg}^d(T) \leq L_{max} + \mathit{Reg}(T) \tag{28}$$

Proof. A new sample is added to sample mean accuracy whenever the true label of a precious prediction arrives. The worst case is when all labels are delayed by L_{max} time slots. This is equivalent to starting the algorithm with an L_{max} delay. □

The above two propositions show that the missing and delayed labels reduce the learning speed. However, since the regret bounds are still sublinear in time T, the time average reward converges to the optimal reward as $T \to \infty$. This shows that our algorithm is robust to errors caused by uncertain traffic conditions.

VI. Experiments

A. *Experimental setup*

1) Dataset: Our experiment utilizes a very large real-world traffic dataset, which includes both real-time and historically archived data since 2010. The dataset consists of two parts: (i) Traffic sensor data from 9300 traffic loop-detectors located on the highways and arterial streets of Los Angeles County (covering 5400 miles cumulatively). Several main traffic parameters such as occupancy, volume and speed are collected in this dataset at the rate of 1 reading per sensor per minute; (ii) Traffic incidents data. This dataset contains the traffic incident information in the same area as in the traffic sensor dataset. On average, 400 incidents occur per day and the dataset includes detailed information of each incident, including the severity and location information of the incident as well as the incident type etc.

2) Evaluation Method: The proposed method is suitable for any spatiotemporal traffic prediction problem. In our experiments, the prediction requests come from a freeway segment of 3.4 miles on interstate freeway 405 (I-405) during daytime 8am to 5pm. Figure **??** shows the freeway segment used in the experiment. Locations will be referred using the distance from the reference location A. For each request from location l_o, the system aims to predict whether the traffic will be congested at l_o in the next 15 minutes using the current traffic speed data. If the traffic speed drops below a threshold λ, then the location is labeled as congested, denoted by $\hat{y} = 1$; otherwise, the location is labeled as not congested, denoted by $\hat{y} = -1$. We will show the results for different values of λ. We use the simple binary reward function for evaluation. That is, the system obtains a reward of 1 if the prediction is correct and 0 otherwise. Therefore, the reward represents the prediction accuracy. The context information that we use in the experiments include the time of day when the prediction request is made and the location where the request comes from. These contexts capture the spatiotemporal feature of the considered problem. Nevertheless, other context information mentioned in Section III(A) can also be adopted in our algorithm.

Using historical data, we construct 6 base predictors (Naive Bayes) for 6 representative situations with context information from the set $[8\text{am}, 12\text{pm}, 4\text{pm}] \times [0 \text{ mile}, 3.4 \text{ miles}]$. These are representative traffic situations since 8am represents the morning rush hour, 12pm represents non-rush hour, 4pm represents the afternoon rush hour, "0 mile" is at the freeway intersection and "3.4 miles" is the farthest location away from the intersection in considered freeway segment.

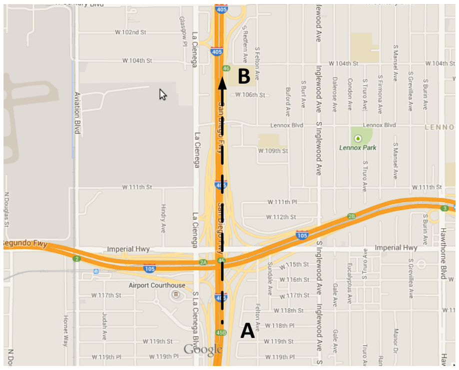

Fig. 6. Freeway segment used in the experiment.

	CA-Traffic	MU	AU	GDU
$\lambda = 50$ mph	0.94	0.83	0.83	0.82
$\lambda = 30$ mph	0.91	0.82	0.80	0.78

TABLE II
Overall prediction accuracy.

3) Baseline Approaches: Since our scheme appertains to the class of online ensemble learning techniques, we will compare our scheme against several such approaches. These baseline solutions assign weights to base predictors but use different rules to update the weights. Denote the weight for base predictor f by w_f. The final traffic prediction depends on the weighted combination of the predictions of the base predictors:

$$y = \begin{cases} +1, & \text{if } \sum_{f \in \mathcal{F}} w_f y_f \geq 0 \\ -1, & \text{otherwise} \end{cases} \tag{29}$$

Three approaches are used to update the weights:

- Multiplicative Update (MU) [**?**] [**?**]: If the prediction is correct for predictor f, i.e. $y_f = \hat{y}$, then $w_f \leftarrow \alpha w_f$ where $\alpha > 1$ is a constant; otherwise, $w_f \leftarrow w_f/\alpha$. In our experiments, $\alpha = 1.05$ since α is usually chosen to be close to 1 for convergence purpose.
- Additive Update (AU) [**?**]: If the prediction is correct for predictor f, i.e. $y_f = \hat{y}$, then $w_f \leftarrow w_f + 1$; otherwise, $w_f \leftarrow w_f$.
- Gradient Descent Update (GDU) [**?**]: The weight of predictor f is update as $w_f \leftarrow (1-\beta)w_f - 2\beta(w_f y_f - \hat{y})\hat{y}$ where $\beta \in (0, 1)$ is a constant. In our experiments, we use a small β, namely $\beta = 0.2$ for convergence purpose.

B. *Prediction accuracy*

In Table **??**, we report the prediction accuracy of our proposed algorithm (CA-Traffic) and the baseline solutions for $\lambda = 50$ mph and $\lambda = 30$ mph. Our algorithm outperforms the baseline solutions by more than 10% in terms of prediction accuracy.

Table **??** and **??** further report the prediction accuracy in different traffic situations. In Table **??**, the location context is fixed at 0.8 miles from the reference location and the accuracy for various time of day contexts (i.e. 10am, 2pm and 5pm) are presented for our proposed algorithm and the

($\lambda = 50$mph)	**CA-Traffic**	MU	AU	GDU
10am	0.93	0.81	0.85	0.83
2pm	0.93	0.86	0.81	0.80
5pm	0.99	0.86	0.87	0.88
($\lambda = 30$mph)	**CA-Traffic**	MU	AU	GDU
10am	0.93	0.83	0.83	0.81
2pm	0.91	0.80	0.83	0.82
5pm	0.99	0.81	0.85	0.83

TABLE III
TRAFFIC PREDICTION ACCURACY AT 0.8 MILES.

($\lambda = 50$ mph)	**CA-Traffic**	MU	AU	GDU
0.8 mile	0.92	0.84	0.83	0.82
2.1 mile	0.96	0.81	0.85	0.85
3.1 mile	0.93	0.85	0.83	0.81
($\lambda = 30$ mph)	**CA-Traffic**	MU	AU	GDU
0.8 mile	0.92	0.81	0.83	0.82
2.1 mile	0.93	0.83	0.82	0.81
3.1 mile	0.94	0.81	0.82	0.82

TABLE IV
TRAFFIC PREDICTION ACCURACY AT 10AM

	CA-Traffic	MU	AU	GDU
10 am, 0.8 miles	54.3	68.6	64.3	66.0
2 pm, 0.8 miles	59.8	70.2	74.3	76.9
5 pm, 0.8 miles	8.9	12.7	12.1	13.4
	CA-Traffic	MU	AU	GDU
0.8 miles, 10 am	47.8	55.9	54.9	53.0
2.1 miles, 10 am	33.8	43.8	47.9	46.7
3.1 miles, 10 am	77.1	93.6	93.5	97.9

TABLE V
MEAN SQUARE ERROR OF TRAFFIC SPEED PREDICTION (MPH^2).

benchmarks. In Table **??**, the time of day context is fixed at 10am and the accuracy for various location contexts (i.e. 0.8 miles, 2.1 miles, 3.1 miles) are reported. In all traffic situations, the proposed algorithm significantly outperforms the baseline solutions since it is able to match specific traffic situations to the best predictors.

Our proposed algorithm not only can predict traffic congestion but also can be used to predict the actual traffic speed. In Table **??**, we report the mean square errors of the traffic speed prediction by using different algorithms. As we can see, our algorithm achieves much smaller mean square errors than the baseline approaches.

C. Convergence of learning

Since our algorithm is an online algorithm, it is also important to investigate its convergence rate. Figure **??** and **??** illustrate the prediction accuracy of our proposed algorithm over time, where the horizontal axis is the number of requests. As we can see, the proposed algorithm converges fast, requiring only a couple of hundreds of traffic prediction requests.

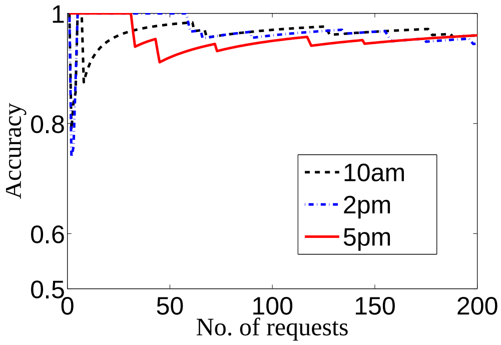

Fig. 7. Accuracy over time with different time of day contexts at 0.8 miles. ($\lambda = 50mph$)

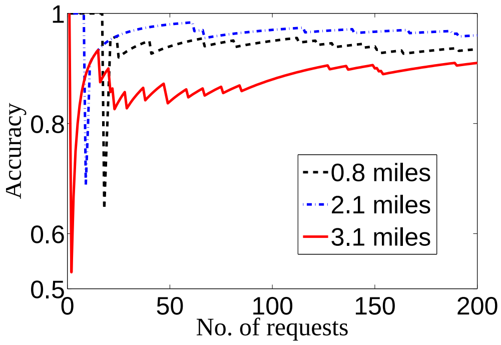

Fig. 8. Accuracy over time with different location contexts at 10am. ($\lambda = 50mph$

($\lambda = 50$ mph)	**CA-Traffic**	**CA-Traffic(R)**	MU	AU	GDU
time of day&distance	0.94	0.89	0.83	0.83	0.82
time of day	0.78	0.92	0.76	0.76	0.78
distance	0.72	0.88	0.79	0.77	0.78
($\lambda = 30$ mph)	**CA-Traffic**	**CA-Traffic(R)**	MU	AU	GDU
time of day&distance	0.91	0.80	0.81	0.83	0.78
time of day	0.76	0.86	0.70	0.72	0.75
distance	0.75	0.89	0.72	0.71	0.74

TABLE VI
TRAFFIC PREDICTION ACCURACY WITH INCOMPLETE CONTEXT INFORMATION.

D. Missing context information

The context information associated with the requests may be missing occasionally due to, for example, missing reports and record mistakes. However, our modified algorithm (described in Section V(A)), denoted by CA-Traffic(R), can easily handle these scenarios. In this set of experiments, we show the performance of the modified algorithm for the extreme cases in which one type of context information is always missing. Table **??** reports the accuracy of our algorithms (CA-Traffic and CA-Traffic(R)) as well as the baseline approaches. Although CA-Traffic(R) performs slightly worse than CA-Traffic when there is no missing context, it performs much better than CA-Traffic and the benchmark solutions when context can be missing because it maintains the context partition separately for each context type and hence, it is robust to missing context information.

E. Relevant context

In this set of experiments, we unravel the most relevant context that leads to the best prediction performance. To do so, we run the algorithm using only a single context

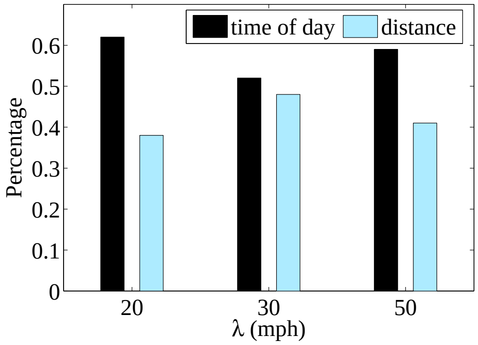

Fig. 9. Relative importance of contexts.

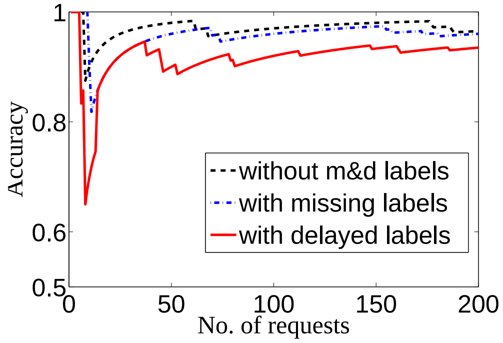

Fig. 10. Prediction accuracy with missing and delayed labels. $\lambda = 50mph$.

(i.e. either time of day or location) and record the average reward. The most relevant context is the one leading to the highest average reward. Figure **??** shows the relative importance (e.g. $Reward(\text{time of day})/(Reward(\text{time of day}) + Reward(\text{location}))$) of each context for different congestion thresholds $\lambda = 20\text{mph}, 30\text{mph}, 50\text{mph}$. The figure shows that the time of the day represents a more relevant context for the traffic prediction problem in our experiment.

F. Missing and delayed labels

Finally, we investigate the impact of missing and delayed labels on the prediction accuracy, as shown in Figure **??** and **??**. In the missing label case, the system observes the true traffic label with probability 0.8. In the delayed label case, the true label of the traffic comes at most five prediction requests later. In both cases, the prediction accuracy is lower than that without missing or delayed labels. However, the proposed algorithm is still able to achieve very high accuracy which exceeds 90%.

VII. Conclusions

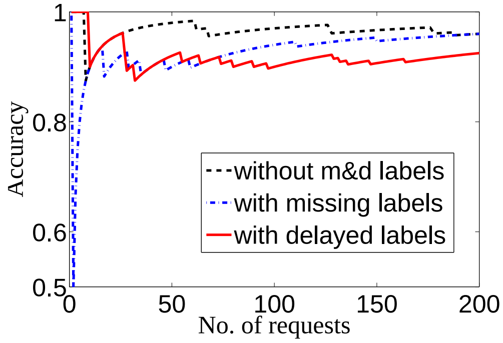

Fig. 11. Prediction accuracy with missing and delayed labels. $\lambda = 30mph$.

In this paper, we proposed a framework for online traffic prediction, which discovers online the contextual specialization of predictors to create a strong hybrid predictor from several weak predictors. The proposed framework matches the real-time traffic situation to the most effective predictor constructed using historical data, thereby self-adapting to the dynamically changing traffic situations. We systematically proved both short-term and long-term performance guarantees for our algorithm, which provide not only the assurance that our algorithm will converge over time to the optimal hybrid predictor for each possible traffic situation but also provide a bound for the speed of convergence to the optimal predictor. Our experiments on real-world dataset verified the efficacy of the proposed scheme and showed that it significantly outperforms existing online learning approaches for traffic prediction. For future work, we plan to extend the current framework to distributed scenarios where traffic data is gathered by distributed entities and thus, coordination among distributed entities are required to achieve a global traffic prediction goal.

References

[1] "What congestion means to your town, 2007 urban area totals," *Texas Transportation Institute*. [Online]. Available: http://mobility.tamu.edu/ums/congestion_data/tables/national/table_2.pdf

[2] "Annual vehicle miles of travel," *Federal Highway Administration*, 2003-02-14. [Online]. Available: http://www.fhwa.dot.gov/ohim/onh00/graph1.htm

[3] "The highway system," *FHWA, Office of Highway Policy Information*. [Online]. Available: http://www.fhwa.dot.gov/ohim/onh00/onh2p5.htm

[4] W. Liu, Y. Zheng, S. Chawla, J. Yuan, and X. Xing, "Discovering spatio-temporal causal interactions in traffic data streams," in *KDD*, 2011, pp. 1010–1018.

[5] M. Miller and C. Gupta, "Mining traffic incidents to forecast impact," ser. UrbComp '12. New York, NY, USA: ACM, 2012, pp. 33–40.

[6] B. Pan, U. Demiryurek, and C. Shahabi, "Utilizing real-world transportation data for accurate traffic prediction," ser. ICDM '12. Washington, DC, USA: IEEE Computer Society, 2012, pp. 595–604.

[7] B. Pan, U. Demiryurek, C. Shahabi, and C. Gupta, "Forecasting spatiotemporal impact of traffic incidents on road networks," in *ICDM*, 2013, pp. 587–596.

[8] S. Lee and D. B. Fambro, "Application of subset autoregressive integrated moving average model for short-term freeway traf?c volume forecasting," in *TRR98*, 1998.

[9] X. Li, Z. Li, J. Han, and J.-G. Lee, "Temporal outlier detection in vehicle traffic data," in *Data Engineering, 2009. ICDE '09. IEEE 25th International Conference on*, 2009, pp. 1319–1322.

[10] G. Giuliano, "Incident characteristics, frequency, and duration on a high volume urban freeway," *Transportation Research Part A: General*, vol. 23, no. 5, 1989.

[11] Y. Chung and W. Recker, "A methodological approach for estimating temporal and spatial extent of delays caused by freeway accidents," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 13, no. 3, pp. 1454–1461, Sept 2012.

[12] B. Stephen, F. David, and W. S. Travis, "Naive bayesian classifier for incident duration prediction," in *Transportation Research Board*, 2007.

[13] W. Kim, S. Natarajan, and G.-L. Chang, "Empirical analysis and modeling of freeway incident duration," in *Intelligent Transportation Systems, 2008. ITSC 2008*, Oct 2008, pp. 453–457.

[14] J. Kwon, M. Mauch, and P. Varaiya, "The components of congestion: delay from incidents, special events, lane closures, weather, potential ramp metering gain, and demand," in *in Proceedings of the 85th annual meeting of the Transportation Research*, 2006.

[15] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[16] Y. Freund, H. S. Seung, E. Shamir, and N. Tishby, "Selective sampling using the query by committee algorithm," *Machine learning*, vol. 28, no. 2-3, pp. 133–168, 1997.

[17] W. Fan, S. J. Stolfo, and J. Zhang, "The application of adaboost for distributed, scalable and on-line learning," in *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1999, pp. 362–366.

[18] N. Littlestone and M. K. Warmuth, "The weighted majority algorithm," *Information and computation*, vol. 108, no. 2, pp. 212–261, 1994.

[19] A. Blum, "Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain," *Machine Learning*, vol. 26, no. 1, pp. 5–23, 1997.

[20] M. Herbster and M. K. Warmuth, "Tracking the best expert," *Machine Learning*, vol. 32, no. 2, pp. 151–178, 1998.

[21] A. Fern and R. Givan, "Online ensemble learning: An empirical study," *Machine Learning*, vol. 53, no. 1-2, pp. 71–109, 2003.

[22] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for svm," *Mathematical programming*, vol. 127, no. 1, pp. 3–30, 2011.

[23] A. Slivkins, "Contextual bandits with similarity information," *arXiv preprint arXiv:0907.3986*, 2009.

[24] C. Tekin and M. van der Schaar, "Distributed online big data classification using context information," in *Proceedings of 51st annual allerton conference on communication, control, and computing*, 2013.

[25] C. Tekin, S. Zhang, and M. van der Schaar, "Distributed online learning in social recommender systems," *Selected Topics in Signal Processing, IEEE Journal of*, vol. 8, no. 4, pp. 638–652, 2014.

[26] J. Xu, C. Tekin, and M. van der Schaar, "Learning optimal classifier chains for real-time big data mining," in *Proceedings of 51st annual allerton conference on communication, control, and computing*, 2013.

[27] J. Xu, M. van der Schaar, J. Liu, and H. Li, "Timely popularity forecasting based on social networks," in *IEEE Infocom*, 2015.

[28] C. Tekin and M. van der Schaar, "Discover the expert: Context-adaptive expert selection for medical diagnosis," *to appear in Emerging Topics in Computing, IEEE Journal on*, 2015.