Spatiotemporal Summarization of Traffic Data Streams

Bei Pan, Ugur Demiryurek, Farnoush Banaei-Kashani and Cyrus Shahabi
Department of Computer Science
University of Southern California
Los Angeles, CA 90089
{beipan,demiryur,banaeika,shahabi}@usc.edu

ABSTRACT

With resource-efficient summarization and accurate reconstruction of the historic traffic sensor data, one can effectively manage and optimize transportation systems (e.g., road networks) to become smarter (better mobility, less congestion, less travel time, and less travel cost) and greener (less waste of fuel and less greenhouse gas production). The existing data summarization (and archival) techniques are generic and are not designed to leverage the unique characteristics of the traffic data for effective data reduction. In this paper, we propose and explore a family of data summaries that take advantage of the high temporal and spatial redundancy/correlation among sensor readings from individual sensors and sensor groups, respectively, for effective data reduction. In particular, with these summaries we derive and maintain a "signature" as well as a series of "outliers" for the readings received from each individual sensor or group of co-located sensors. While signatures capture the typical readings that estimate the actual readings with bounded error, the outliers represent the actual readings where the error-bound is violated. With the combination of signatures and outliers, our proposed data summaries can effectively represent the actual data with much smaller storage footprint, while allowing for efficient querying of the sensor data with bounded error. Our experiments with a real traffic sensor dataset shows that our proposed data summaries use only 23% of the storage space otherwise required for storing the actual data, while allowing for highly accurate query results with guaranteed precision.

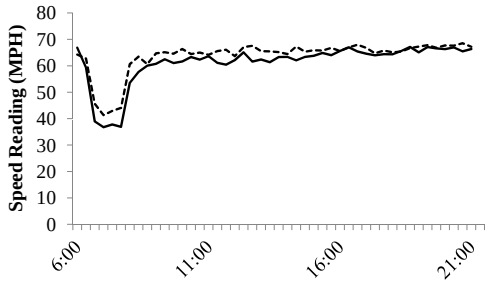
1. INTRODUCTION

The vast amounts of traffic data collected from the traffic sensors are extremely valuable for real-time decision-making, planning and management of intelligent transportation systems (ITS). Traffic sensors collect various readings such as traffic speed, volume, and occupancy data. In many cases, the traffic data remains useful for *historical analysis* long after it is collected. For example, the traffic data collected from Los Angeles County road networks can be aggregated over time to estimate the effects of the newly added traffic lights, or it could be combined geographically with data from other cities to derive a broader picture of spatiotemporal traffic flow. Even deeper insight might be gained by integrating historical traffic data with historical demographics data. However, the majority of ITS deployments focus on placing the sensors in the field to collect data and consume it immediately rather than implementing historical data storage that enables analysis and mining of the traffic data. Considering the huge number of sensors located on the road networks and their 24/7 continuous operation with frequent sampling, implementation of a scalable data storage and querying system that facilitates the analysis and management of the traffic sensor data is an intrinsically challenging data management task.

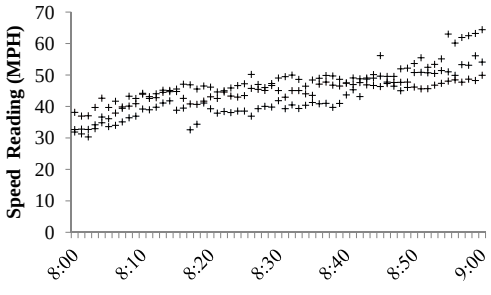
A naive approach for archiving historical traffic sensor data is to maintain the entire data by appending every new sensor reading to the historic dataset as the reading arrives at the data storage systems. This approach can be implemented by using flat files or database management systems (DBMS) such as Oracle and Microsoft SQL Server which are continuously updated with the streaming datasets. However, there are two major problems with this approach. First, a comprehensive data collection strategy is infeasible since the sensor data is *unbounded*. Second, the computational overhead of historical querying and statistical analysis of such vast amount of data is prohibitively high. An alternative data archival approach is to employ lossy data reduction techniques (e.g., Wavelet Decomposition [16] or SVD [17]). The main idea behind these data reduction techniques is to leverage the redundancy in data and compactly store the main patterns in the data (i.e., data sketch) in such a way that once needed, the dataset can be reconstructed in its entirety with a minimal loss of accuracy. However, such data reduction techniques also have serious shortcomings. Even though these techniques offer very good data reduction rates on certain datasets, the reduction efficacy is highly data-dependent. For example, high variations in the datasets can cause these techniques to store large amount of data (hence less data reduction) for acceptable accuracy. In addition, query processing with these methods requires developing complex routines to refactor the sketches and rewrite the queries. Finally, the maintenance of the sketches is not straightforward as they are frequently invalidated with the streaming sensor data. Although there are several incremental algorithms that update the sketches, they cannot handle the frequent updates of the sensor data.

In this paper, we propose a data summarization technique that significantly reduces the storage requirement of the traffic sensor data and enables efficient query processing on the historical datasets. Our proposed approach builds on the observation that there is a strong *correlation* (both temporally and spatially) and *redundancy* present among the measurements of the single and multiple traffic sensor(s). For example, Figure 1(a) plots the average speed measurement from a single sensor located on I-10 East for two consec-

IWGS November 2, 2010, San Jose, CA, USA



(a) Average Speed of a sensor for two consecutive Mondays



(b) Average Speed of multiple sensors on Wednesday

Figure 1: Examples of sensor readings

utive Mondays from 6 AM to 9 PM. As shown, both signals follow almost the same trend, and hence it is obvious that maintainng the two sets of measurements in their entirety is redundant. As another example, Figure 1(b) depicts a scatter plot of speed measurements (for Wednesday from 8 AM to 9 AM) from four different sensors which are spatially close to each other on a segment of I-10 East. Similarly, there exists a strong correlation among the speed measurements of multiple sensors in spatial proximity. Given these observations, with our data summarization technique, we derive and maintain data *signatures* which represent typical patterns of the sensor readings which approximates the actual readings with bounded error. These data signatures, first enable us to store the streaming sensor data more efficiently by discarding the redundant sensor readings, and hence, provide cost effective data growth. Second, we can evaluate the spatiotemporal queries based on a small but informative summary of the sensor readings with sufficiently accurate results, rather than having to scan the entire datasets which yields unacceptable response times. Specifically, with signature based approach, we only store the streaming data which falls outside of the signature (i.e., *outlier*) within a given error-bound; otherwise we discard the data since it is already represented by the signature. With our study, we use a large-scale traffic sensor dataset collected from the entire Los Angeles County highways for the past two years. Based on our experiments on this real dataset, we observe that our proposed approach can reduce the storage requirement up to 77% while maintaining high accuracy (with bounded-errors) on the query results.

The remainder of this paper is organized as follows. In Section 2, we formally define the problem of sensor data summarization and approximate querying with bounded-error rates. In Section 3, we provide the overview of our proposed data summarization approach. In Section 4, we establish the theoretical foundation of our proposed data summaries, followed by the corresponding processing techniques in Section 5. In Section 6, we present experimental results with real-world traffic sensor data. In Section 7, we review the related work on data reduction techniques as well as sensor data systems. In Section 8, we conclude and discuss our future work.

2. PROBLEM DEFINITION

In our study, we consider the readings collected from each sensor as a time-series with each reading observed by a sensor node at time t. Each sensor node is located on a road network segment. Each sensor reading contains multiple attributes(i.e., speed, volume and occupancy) describing the traffic behavior. In the rest of this paper, we only use the speed reading to formalize our problem. In our historical traffic sensor dataset, each sensor reading is represented as a combination of sensor_id, speed value, date, and time, denoted by $<i, v, d, t>$. The speed value denotes the average speed during its sensor sampling time unit. We denote the entire dataset by D that contains all sensor readings during the time interval $[T_s, T_e]$ where T_s and T_e represent the beginning and ending timestamp of the data collection period. Our goal is to provide approximate results (with a bounded-error) to the queries that ask for the speed reading for a single sensor during time interval $[t_s, t_e]$, where $[t_s, t_e] \subseteq [T_s, T_e]$. Note that other queries for speed readings (e.g., average query, aggregate query) can be answered on top of this query defined here. We refer to such query as *spatiotemporal query*.

Since we are interested in answering the spatiotemporal queries within a specified error-bound, we define *precision constraint* which incorporates user specified precision parameters and enforces an approximate result to deviate from the exact result by (at most) $\pm$ error-bound ϵ with probability δ.

DEFINITION 1. **Precision Constraint.** *Let ϵ and δ denote relative error and probabilistic guarantee specified by users, respectively. The approximate result Y to a query should satisfy the following precision constraint:*

$$P[|Y - A| \leq \epsilon \cdot A] \geq 1 - \delta \quad (1)$$

where A represents the exact result of the query.

Precision Constraint states that the approximate result Y should hold a relative error of at most ϵ of the exact result A with probability of at least $1 - \delta$. For example, if a user specifies $\epsilon = 0.1$ and $1 - \delta = 0.9$, Y should satisfy $P[|Y - A| \leq 0.1 \cdot A] \geq 0.9$, i.e., with 90 percent probability the approximate answer is within the 10 percent of the exact result.

3. OVERVIEW OF APPROACH

One way of answering spatiotemporal queries approximately is to capture the underlying patterns from the sensor readings and use them instead of the exact readings. Towards this end, we create a concise but reasonably accurate pattern of a sensor or a group of sensors called $signature$ by averaging the sensor readings. Hence, given a spatiotemporal historical query, we can use the signatures to represent the results rather than scanning the entire historical data. However, when the signatures are not sufficient to represent the exact sensor readings within precision constraint, we store the sensor readings that violate the constraint as $outlier$. We consider both signatures and outliers as *data summaries* to answer the spatiotemporal queries.

To further improve the storage efficiency, we explore temporal and spatial correlation in constructing data summaries based on the following observations. Like in most environmental monitoring sensor network deployments (e.g., pollution, temperature), the data generated by the traffic sensor nodes is highly auto correlated both in time and space. For example, the weekday readings from a

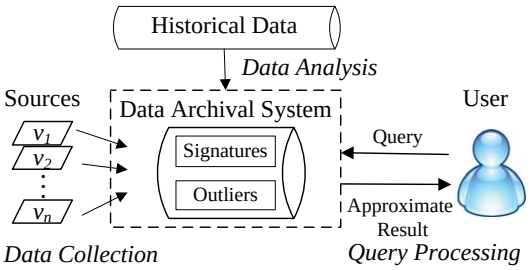


Figure 2: Architecture of Our Data Archival System

specific traffic sensor usually follow a similar pattern, i.e., the sensor reports 50-60 MPH (i.e., Miles Per Hour) on average from 6AM to 8AM and 20-35 MPH from 8AM to 9AM and so on. Similarly, the readings from multiple sensors located within a spatial proximity may also be strongly correlated, since the traffic flow hardly diverses or accumulates within a small spatial region, especially for the road segments with no exits/entries. These correlations can be captured accurately by constructing different types of data summaries from the historical data traces.

We now explain the construction and maintenance steps of the data summaries shown in Figure 2. Our approach involves three phases. At the data analysis and query processing phase, we use the historical data to precompute the signatures and their corresponding outliers to support historical spatiotemporal queries submitted by users. At the data collection phase, we compare the incoming sensor readings to corresponding signatures to identify whether they violate the precision constraint. If precision constraint is violated, to avoid storing all such sensor readings, we conduct sampling among them with rate $1-\delta$ and only store the samples, otherwise, we discard the reading because we can use its signature value to represent it in the query processing.

4. CONSTRUCTION OF SPATIOTEMPORAL DATA SUMMARIES

As mentioned, the storage requirement of our data archived system includes two main components, namely signatures and outliers. Before formally define these two components, let us first introduce an important parameter T for sensor readings sampling time unit. Thereby, every T minute(s), the sensor readings are sampled once. With the help of T, signature is defined as follows:

DEFINITION 2. **Signature.** *A signature S for a sensor node (or a set of sensor nodes) is defined by a sequence of sensor readings during time interval $[t_s, t_e]$. The length of the sequence equals to the number of samples taken during interval $[t_s, t_e]$, namely, the number of time units T covered by $[t_s, t_e]$. For example, given sampling time unit T=1 minute, and t_s=6:00AM , t_e=21:00PM, the signature S of a sensor have a sequence of (21-6)*60 + 1 = 901 average sensor readings with each one representing the average speed for 1 minute. The solid line in Figure 3 shows a sample signature of a sensor node.*

In general, we can adopt signatures to answer a spatiotemporal query within precision constraint. However, certain traffic conditions (e.g., lane closures, accidents, sports games) may cause the sensor readings fluctuate significantly from the signatures. Clearly, when users conduct queries corresponding to the time intervals of such conditions, the signatures are not sufficient to satisfy the precision constraint. Therefore, to ensure the precision constraint, in addition to signatures, we store the outliers, i.e., the sensor readings that fall outside of the relative error (ϵ) range as outliers. To derive the outliers from the precision constraint, we rearrange the Equation (1) by removing the absolute sign and disregarding the probability guarantee. Thus, we can obtain the following constraint for the exact result A:

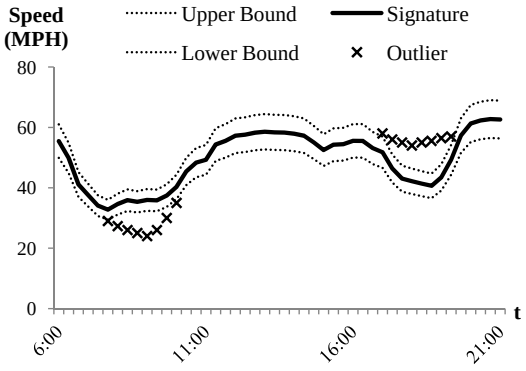


Figure 3: An example of data summary

$$A \geq \frac{Y}{1+\epsilon} \quad and \quad A \leq \frac{Y}{1-\epsilon} \tag{2}$$

The definition of outliers is as follows.

DEFINITION 3. **Outlier.** *If a sensor reading v during a sensor sampling interval (i.e.,$[t_j, t_j + T]$) satisfies one of the following inequalities with its corresponding signature S_j:*

$$v < \frac{S_j}{1+\epsilon} \quad or \quad v > \frac{S_j}{1-\epsilon}, \tag{3}$$

v is identified as an outlier.

Note that in the definition of an outlier, approximate value Y and exact value A are replaced by signature value S_j and sensor reading v, respectively. In Figure 3, the dash lines indicate the error bounds of a sample signature and the crosses represent the outliers that are outside of the error-bounds. When sensor readings are identified as outliers, we only store a subset of them by sampling with probability $1-\delta$. This enables us to avoid maintaining all the outliers.

Accordingly, given a query, we not only utilize the signatures to provide approximate answers, but also incorporate the outliers when the signatures are not sufficient in satisfying the precision constraints. We argue that the combination of signatures and outliers can satisfy the precision constraints. The justification of our argument is as follows. For the query results (or part of it) from the outliers, they are 100 percent accurate with no error, because we store the exact sensor reading as outlier. In this case, we have $P[|Y - A| = 0 \leq \epsilon \cdot A] = 1$. On the other hand, if the results are from signatures, based on the way we sample the outliers, there is 1-δ probability that the exact result is within the error range ϵ, so we have $P[|Y - A| \leq \epsilon \cdot A] = 1 - \delta$. Combining the two inequality in these two cases, we have $P[|Y - A| \leq \epsilon \cdot A] \geq 1 - \delta$. Hence, the combination of the results from signatures and outliers can guarantee the precision constraint.

We explain our sensor data summarization methods in the following subsections.

4.1 Basic Summarization

So far, we have formally defined the two components of data summaries: signatures and outliers. In this section, we explain our

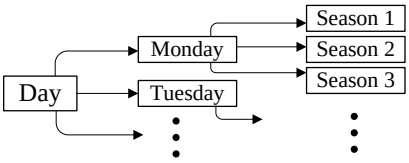


Figure 4: Different Levels of Temporal Summaries

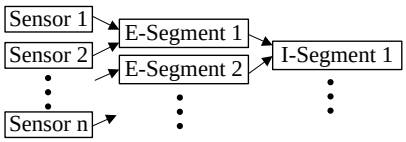


Figure 5: Different Aggregation Level of Spatial Summaries

basic summarization technique. In this technique, we compute the daily signature of a sensor by averaging all historical sensor readings from that sensor, and each signature is indexed by sensor ID i (i.e., <$\underline{i}$, S>). We repeat this process for all sensors. For each sensor, the outliers are identified by comparing the sensor reading to the corresponding value in its signature S. Specifically, since each sensor reading is represented as the combination of sensor id, speed value, date, and time (i.e., <i, v, d, t>), we use the time attribute t to find its corresponding value of signature S in time unit $[t, t+T]$, denoted as S_j. Then, we examine v and S_j in the context of outlier definition to determine whether the reading is an outlier. Each outlier is indexed by sensor id, date and time (i.e.,<$\underline{i}, v, \underline{d}, \underline{t}$>).

Since each signature is generated by averaging the entire historical data of the corresponding sensor, the storage need of signatures is negligible. However, if a signature is not representative enough (i.e., does not capture the typical patterns of its corresponding sensor), the storage needed to maintain the outliers can be high. We address this problem by maintaining several signatures for one sensor at different temporal and spatial scales. Meanwhile, we also aim to strike a compromise between the storage of signatures and outliers, and minimize the overall storage requirement of data summaries. Towards these ends, we propose two different data summarization techniques that exploit temporal and spatial correlations of the sensors. We elaborate on these techniques as follows.

4.2 Temporal Summary

In real-world road networks, the traffic patterns may show variations among different days within a week or even different seasons. Take traffic behaviors in weekday and weekends as an example. In weekdays, traffic is always congested in the morning and afternoon rush hours. However, in weekends, the traffic follows a totally different pattern. We can characterize such diversities with more than one signatures corresponding to different temporal scales. Trivially, increasing the number of signatures reduces the amount of storage needed by outliers.

We explain our temporal summary technique using the example in Figure 4 where we focus on three levels of temporal summaries. The leftmost level indicates the method using single signature for each sensor as discussed in the basic summarization. At the second level, we increase the granularity of temporal summaries by providing seven signatures with each one representing a unique day in the week. Each signature at this level is computed by averaging all the sensor readings collected on the corresponding day. For example, $Wednesday$ signature of a sensor is the average of its sensor readings collected on Wednesdays in the historical dataset. In this level, each signature is indexed by sensor_id i with the weekday category w (i.e.,<$\underline{i}, \underline{w}, S$>), where $w \in \{Mon, Tue, ..., Sun\}$. At the third level, we increase the temporal granularity by introducing seasonal information. Based on each signature generated in the previous level, we derive three signatures with each one representing the sensor readings within a particular season. Similarly, we use sensor ID i, weekday category w, and season category z as an index of each signature (i.e., <$\underline{i}, \underline{w}, \underline{z}, S$>). In this level, we create a total of 21 signatures for each sensor characterizing the sensor readings at different temporal scales.

To identify whether a sensor reading (i.e.,<i, v, d, t>) is an outlier, besides the time information t, we also need to use the date information d to find its weekday value w and its season value z to identify the category of the corresponding signature S. Once we have identified S, we use similar strategy with the basic summarization in examining and storing the outliers.

4.3 Spatial Summary

Besides the temporal correlation of the traffic sensor data, we can use the spatial correlation of sensor locations to generate different signatures. Specifically, we exploit the fact that the traffic sensors co-located within a spatial proximity report similar readings and behaviors. The main challenge here is to identify the sensors that are within a close spatial proximity. Based on our observation of the real-world traffic dataset, the traffic flow changes only at entries/exits or intersections. Therefore, the sensor readings between two adjacent entries/exits, or between two adjacent intersections may show similar values. Therefore, instead of maintaining one signature per sensor, we can compute one common signature for a group of sensors. Subsequently, with spatial summarization, we aim at eliminating the redundant signatures.

We define two types of segments for spatial summaries that include groups of sensors which have similar patterns: the segment between two adjacent exits/entries *E-Segment* and the segment between two adjacent intersections *I-Segment*. As illustrated in Figure 5, in a typical road network, each E-Segment includes a small number of sensors, and each I-Segment includes several E-Segments, corresponding to a larger number of sensors. To compute the signatures for each segment, we first identify the set of sensors located in that segment by maintaining a mapping between group ID g and sensor ID i, (i.e., <$\underline{g}, \underline{i}$>). Next, we calculate the average of the sensor readings from all sensors in this group and use it as the common signature for each individual sensor located in that group. In this case, we index each signature by group ID (g) (i.e., <$\underline{g}, S$>).

To identify outliers, we first use the mapping between group_id and sensor_id to identify to which group the sensor belongs. Next, we employ the group signature S as the corresponding signature for the sensor. Finally, we follow the similar process discussed in the basic summarization to identify and store outliers .

5. QUERY PROCESSING

So far, we have discussed three different strategies for constructing data summaries. In this section, we introduce our proposed approach to answer spatiotemporal queries based on each type of data summaries. Algorithm 1 depicts the query processing for the basic summarization technique. With this technique, given a query, asking for sensor readings during a particular time interval $[t_s, t_e]$ for a particular sensor i, we perform the following four steps to generate the answer.

- First, we partition the time interval $[t_s, t_e]$ into individual sensor sampling intervals (i.e., $[t_j, t_j+T]$), and initialize the

Algorithm 1 Spatiotemporal Query(Sensor i, Time Interval$[t_s, t_e]$)

```
Let R be the result array for the query, initialize with empty
for each time interval [t_j, t_j + T] in [t_s, t_e] do
    Abstract date information from t_j, denoted as d'
    Abstract time information from t_j, denoted as t'
    Use its i, d', t' to search outlier table.
    if any outlier O found then
        Add O to R_j for interval [t_j, t_j + T]
    else
        Employ its sensor ID i to find its signature S
        Use t' to identify the position(k) of [t_j, t_j + T] in the
        sequence of S
        Add S_k to the R_j for interval [t_j, t_j + T].
    end if
end for
Return R
```

empty result array to carry the result value for each sampling interval.

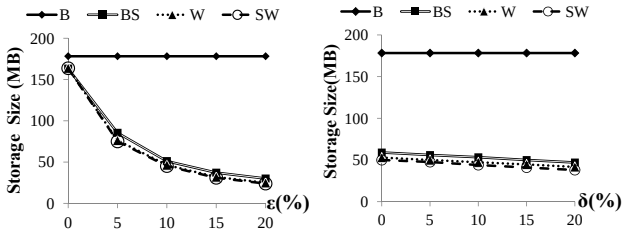
- Second, for each interval $[t_j, t_j + T]$, we extract date and time information from interval $[t_j, t_j + T]$, denoted as d' and t'. We use sensor_id i, d' and t' to search the database to check if any corresponding outlier stored. If we find such an outlier, we insert its value to the result array for interval $[t_j, t_j + T]$ and skip the third step, otherwise, we continue with the third step.
- Third, we employ the sensor id i to search for its corresponding signature S. Then, we utilize t' to find the corresponding position (k) of S representing the interval $[t_j, t_j + T]$ and insert S_k to the result set for the interval $[t_j, t_j + T]$.
- Finally, once we have gone through all the individual sensor sampling intervals in $[t_s, t_e]$, we return the result array as the approximate answer to the query.

For the queries based on temporal or spatial summaries, we use the similar framework with a few changes in the third step. For temporal summaries, besides sensor id i, we add date information d' from t_j to search for the corresponding signature S. Since we maintain several signatures instead of one per sensor, we need to identify the particular one for t_j by using d'. For spatial summaries, before searching for the signature, we identify the group ID of sensor i. Next, instead of sensor ID, we use group ID to find the group signature as the signature for sensor i in the third step.

6. EXPERIMENTS

6.1 Methodology

In our experiments, we use a large-scale and high resolution (both spatially and temporally) traffic sensor (i.e., loop detector) dataset collected from entire Los Angeles County highways. This dataset includes both inventory and real-time data for around 1800 traffic sensors covering approximately 3000 miles. The sampling rate of the streaming data is 1 reading/sensor/min. The format of the data is <$sensor\_id$, $reading$, $date$, $time$>, e.g., <717534, 67, 2009/06/23,13:40>. To evaluate the storage efficiency, we compare our summarization techniques with a baseline solution which stores entire historical sensor readings. In the first two sets of experiments, we vary two precision constraint parameters: error range ϵ and probabilistic guarantee rate δ by comparing the storage requirement of the baseline approach, with our techniques using different summary strategies (i.e., temporal and spatial summaries).



(a) Effect of ϵ (b) Effect of δ

Figure 6: The overall storage size for temporal summaries

When varying one parameter, we set the value of the other one to 10%. In the third set of experiments, we fix the precision constraint (i.e., both ϵ and δ are set to 10 %), and compare the different combinations of our two summarization techniques in storage efficiency and signature size. The performance is measured as the storage requirement of each technique. For all the experiments, we use a PC running Windows with Intel 6420 Dual CPU 2.13G and 3.0 GB RAM.

6.2 Results

6.2.1 Performance of Temporal Summaries

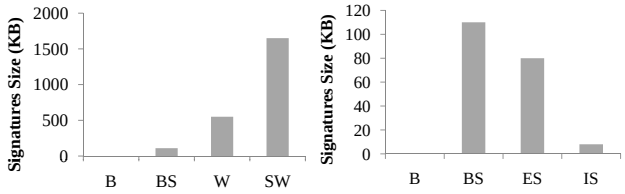
First, we compare the baseline approach (B), the basic summarization (BS) technique and two temporal summarization techniques based on two temporal scales: Weekday (W), Seasonally Weekday (SW). For SW, we define three seasons: spring, summer, fall with each one covering four months of the year, Jan-Apr, May-Aug, Sep-Dec, respectively.

Figure 6 shows that as ϵ and δ increases the overall storage requirement of our summarization techniques based on different temporal correlations decrease sharply as compared to the baseline approach. For the effect of ϵ and δ, we observe that as ϵ increase, we holds an increasing reduction rate on storage requirement. Specifically, when ϵ increases to 10, the storage requirement of our system is reduced by nearly 75% as compared to the baseline approach. As ϵ increase to 20, the reduction reaches 80% to 85% percent. This indicates that about 75% to 85% of the entire sensor readings are distributed within a small error range of the signatures. For δ, the storage size decreases linearly as δ increases. The reason is that we sample the outliers to store, so the storage size is proportional to the sampling rate. In general, the higher the temporal scales, the more signatures are stored, hence resulting in less number of outliers as shown in Figure 6. From approach BS to W, the decrease in storage requirement is noticeable, but from W to SW, the two lines nearly overlap with each other, which indicates the amount of storage saving is negligible. One reason is that the traffic sensor readings hardly changes across different seasons in Los Angeles. Figure 7(a) shows the size of signatures for SW is two times higher than that of W. In conclusion, to make a trade-off between the number of signatures and the storage efficiency, the temporal summary by weekday signatures is the proper temporal summarization approach to choose in Los Angeles.

6.2.2 Performance of Spatial Summaries

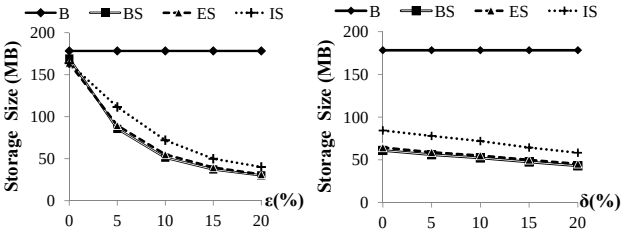
Next, we study the impacts of different spatial summarization techniques on the storage size. We compare the baseline approach (B) and basic summarization (BS) with two spatial summaries designed for sensors within an E-Segment (ES), and within an I-Segment (IS).

As shown in Figure 8, the impacts of ϵ and δ are similar to those



(a) Temporal Summaries (b) Spatial Summaries

Figure 7: The signatures size for different summaries



(a) Effect of ϵ (b) Effect of δ

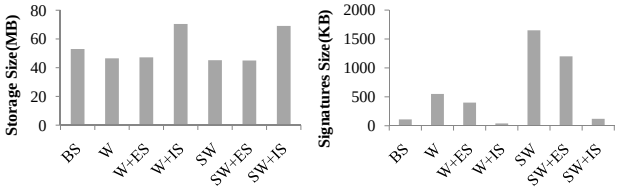
Figure 8: The overall storage size for spatial summaries

previous experiments. When we increase the spatial aggregation level by grouping sensors according to different size of segments (in our dataset, I-Segments are longer than E-Segments), the overall representation capability of each group signature for individual sensor is reduced, therefore the number of outliers increases. In particular, comparing IS with BS in both Figure 8 and Figure 7(b), although the signature size of IS is reduced significantly as compared that of BS, IS shows a sharp increase of outliers. One possible reason for that is the sensor readings generally fluctuate a lot within two adjacent intersections. We observe that both BS and ES require similar storage capacity in most cases, which means that sensors located between two exits mostly maintain similar speed readings hence the amount of outliers does not increase significantly. Moreover, the signature size of ES is smaller than that of BS. Hence, the ES based spatial summarization technique was the best one in this set of experiments.

6.2.3 Performance of Spatial & Temporal Summaries

With our previous two experiments, we fixed one type of summarization technique at one time to examine the effect of others. In our third set of experiments, we vary both the spatial and temporal summarization technique simultaneously to identify the optimal combination for our system. With this set of experiments, the same notation as in the last two experiments are used, (e.g., $W+ES$ indicates the combination of week day summarization and E-Segment summarization). We fix both ϵ and δ to 10%.

Figure 9 shows the comparison of the seven combinations for overall storage size and signature size. As shown, although IS methods is useful in decreasing the amount of signatures significantly, it sacrifices the overall storage size because of the increasing number of outliers, so it cannot be considered as a part of the optimal choices. When comparing the performance of techniques including W and the ones including SW, the storage size does not change much, but the signature sizes of the ones with SW are much larger than that of the ones with W. Therefore, we should also exclude SW technique from our optimal choices. Now, let us compare the remaining three choices: BS, W and $W + ES$. As shown, W and $W+ES$ requires similar storage requirement that is



(a) Overall Storage Size (b) signature Size Only

Figure 9: The performance of temporal & spatial summaries

much less than that of BS. Hence, we eliminate BS. But $W + ES$ performs better by maintaining less signatures as compared to W. Hence, $W + ES$ is the optimal solution for our system. By comparing the overall storage size of $W + ES$ with that of the baseline approach shown in previous experiments, we observe that the storage requirement of our system as decreased by 77% percent of the baseline approach with ϵ=10% and δ=10%.

7. RELATED WORK

In the past, different types of data reduction techniques have been widely used to reduce the size of the large sensor datasets. The prominent data reduction techniques are Wavelets [16], Single Value Decompositions (SVD) [17] and Principal Component Analysis (PCA) [12]. The main idea behind these techniques is to compactly store the main patterns in the data (i.e., sketches) in such a way that the dataset can be reconstructed back in its entirety from those patterns, with minimal loss of accuracy. Wavelets - a widely used technique in signal processing and image compression - compress large datasets by hierarchically decomposing the raw data and storing a small number of wavelet basis functions (i.e., wavelet coefficients) which best describe the data. Wavelets have been applied successfully in answering range-sum aggregate queries over data cubes [16], in selectivity estimation [13] and in approximate query processing [4, 11, 15]. Likewise, SVD and PCA represent a multivariate dataset using the smallest possible number of new variables (i.e., principal components) that are selected based on the statistical characteristics of the dataset. PCA reduce the size of the datasets by maintaining a sketch of archived historical data (i.e., small number of principal components and a transformed dataset). However, the main difference is that although these compression techniques enable approximate query processing on the set of sketches, most of them do not guarantee any error bounds on the query results. Specifically, depending on the spatial and temporal extent of the query, the variation between the actual result and the approximated result can be unacceptable. In contrast, our approach ensure both an error bound and probabilistic guarantee on the results to the spatiotemporal queries.

Another line of related work is data stream processing. In many streaming techniques, the structures similar with signatures(e.g., synopsis) are built online for real-time approximate query purposes, examples include equi-depth histograms and Haar wavelets [9, 13], maintaining samples and simple statistics over sliding windows [6], data clustering and decision tree construction [10, 7]. But most these research efforts focus on the application of online data monitoring rather than queries over historical dataset. Similarly, in some large streaming projects, queries over historical data streams do not receive much attention. In the area of sensor networks, such systems includes Aurora[1] which is designed for the purpose of managing data streams for monitoring applications and Telegraph [5] from UC Berkeley which focuses on creating adaptive engine

over querying streaming data from sensors. However, for other streaming projects, such as Coguar [8] from Cornell University which considers sensor network as a distributed database system, STREAM [2] which serves as a general-purpose data stream management system as well as Niagara[14] which is designed for internet XML query processing, do concern the historical queries, but the type of their queries do not include spatial and temporal filters as the spatiotemporal query defined in this paper.

For the data reduction in spatiotemporal domain, there have been several studies customized for specific types of spatiotemporal dataset. One of them is the work done by Cao et al. [3] on data reduction over trajectories of moving objects. They adopted line simplification approach from graphics field to reduce the storage size of trajectories. Unlike the traditional reduction technique, the line simplification based approach can guarantee a deterministic error bound, which is very similar with our error bounds structure. However, the approach of line-simplification aims at geographically simplify the presentation of individual moving objects trajectory, therefore heavily relies on the structure of trajectory data and not applicable on the traffic sensor data described in this paper.

8. CONCLUSION & FUTURE WORK

In this paper, we proposed a family of data summarization techniques that significantly reduce the storage requirement of the traffic sensor data while enabling efficient query processing on historical traffic datasets. Unlike standard data summarization techniques, our proposed approach builds on insights about the nature of the traffic sensor dataset. In particular, we observed that there is a strong correlation (both temporally and spatially) and redundancy among the measurements of individual as well as groups of traffic sensor(s). Driven by these observations, we introduced a family of summarization techniques that capture data signatures and outliers to approximates the actual readings with bounded error and probabilistic guarantee. Our experiments with a real traffic sensor dataset showed that our proposed data summaries use only 23for storing the actual data, while providing highly accurate query results with guaranteed precision.

We intend to pursue this study in three different directions. First, we plan to extend our experiments to compare the summarization efficiency of our approach with standard data summarization techniques such as wavelets and SVD with more complex spatiotemporal queries (e.g., average, aggregate) over traffic sensor dataset. Second, we intend to investigate efficient ways for maintenance/ update of the signatures. Finally, to enhance the summarization ratio with our proposed techniques, we plan to study these techniques under various spatiotemporal data granularities with different correlations. Moreover, we study patterns in outliers for the purpose of minimizing the outlier storage.

9. ACKNOWLEDGMENTS

This research has been funded in part by NSF grant CNS- 0831505 (CyberTrust), the NSF Integrated Media Systems Center (IMSC), unrestricted cash-gift and equipment gift from Google, and in part from the METRANS Transportation Center, under grants from USDOT and Caltrans. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

10. REFERENCES

[1] D. J. Abadi, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, S. Lee, M. Stonebraker, N. Tatbul, and S. Zdonik. Aurora: a new model and architecture for data stream management. In *VLDB Journal*, August 2003.

[2] A. Arasu, B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein, and J. Widom. Stream: The stanford stream data manager. In *IEEE Data engeneering Bulletin*, volume 26, 2003.

[3] H. Cao, O. Wolfson, and G. Trajcevski. Spatio-temporal data reduction with deterministic error bounds. In *The VLDB Journal - The International Journal on Very Large Data Bases*, volume 23, pages 211–228. Springer-Verlag New York, Inc, 2006.

[4] K. Chakrabarti, M. Garofalakis, R. Rastogi, and K. Shim. Approximate query processing using wavelets. In *Proceedings of the 26th VLDB Conference*, 2000.

[5] S. Chandrasekaran, O. Cooper, A. Deshpande, M. J. Franklin, J. M. Hellerstein, W. Hong, S. Krishnamurthy, S. R. Madden, F. Reiss, and M. A. Shah. Telegraphcq: continuous dataflow processing. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 668–668, San Diego, California, 2003.

[6] Y. Datar, A. Gionis, P. Indyk, and R. Motwani. Maintaining stream statistics over sliding windows. In *Proc. of the 13th Annual ACM-SIAM Symp. on Discrete Algorithms*, January 2002.

[7] P. Domingos and G. Hulten. Mining high-speed data streams. In *Proc. of the Sixth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, August 2000.

[8] W. F. Fung. Cougar: The network is the database. In *Proc. of SIGMOD Conference*, pages 621–621. ACM Press, 2002.

[9] P. B. Gibbons, Y. Matias, and V. Poosala. Fast incremental maintenance of approximate histograms. In *Proc. of the 23rd Intl.Conf. on Very Large Data Bases (VLDB)*, August 1997.

[10] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. OŠCallaghan. Clustering data streams. In *Proc. of the 2000 Annual Symp. on Foundations of Computer Science (FOCS)*, November 2000.

[11] M. Jahangiri, D. Sacharidis, and C. Shahabi. Shift-split: I/o efficient maintenance of wavelet-transformed multidimensional data. In *24th ACM SIGMOD International Conference on Management of Data*, Baltimore, Maryland, USA, June 2005.

[12] I. T. Jolliffe. Principal component analysis. In *Springer-Verlag*, 1986.

[13] Y. Matias, J. S. Vitter, and M. Wang. Dynamic maintenance of wavelet-based histograms. In *Proc. of the 26rd Intl.Conf. on Very Large Data Bases (VLDB)*, September 2000.

[14] J. Naughton and et al. The niagara internet query system. In *Proceedings of the 26th VLDB Conference*, 2000.

[15] R. R. Schmidt and C. Shahabi. Propolyne: A fast wavelet-based algorithm for progressive evaluation of polynomial range-sum queries(extended version). In *VIII. Conference on Extending Database Technology*, Prague, March 2002.

[16] J. S. Vitter and M. Wang. Approximate computation of multidimensional aggregates of sparse data using wavelets. In *Proc. of the 1999 ACM SIGMOD Conf.*, pages 193–204, Philadelphia, Pennsyl-vania, 1999.

[17] M. E. Wall, A. Rechtsteiner, and L. M. Rocha. Singular value decomposition and principal component analysis. 2003.