A Confabulation Model for Abnormal Vehicle Events Detection in Wide-Area Traffic Monitoring

Qiuwen Chen[1], Qinru Qiu[1], Qing Wu[2], Morgan Bishop[2], and Mark Barnell[2]

[1]Dept. of Electrical Engineering & Computer Science
Syracuse University
Syracuse, NY 13244, USA

[2]Air Force Research Laboratory
Information Directorate, RITC
525 Brooks Road, Rome, NY, 13441, USA

***Abstract* — The advanced sensing and imaging technologies of today's digital camera systems provide the capability of monitoring traffic flows in a very large area. In order to provide continuous monitoring and prompt anomaly detection, an abstract-level autonomous anomaly detection model is developed that is able to detect various categories of abnormal vehicle events with unsupervised learning. The method is based on the cogent confabulation model, which performs statistical inference functions in a neuromorphic formulation. The proposed approach covers the partitioning of a large region, training of the vehicle behavior knowledge base and the detection of anomalies according to the likelihood-ratio test. A software version of the system is implemented to verify the proposed model. The experimental results demonstrate the functionality of the detection model and compare the system performance under different configurations.**

***Index Terms* — anomaly detection, cogent confabulation, intelligent transportation, unsupervised learning**

I. Introduction

Anomaly detection, which refers to the techniques of identifying patterns that do not conform to the regular observations in a given data set, is of considerable importance. The problems of anomaly recognition and detection frequently arise from different domains, such as medical diagnosis and network intrusion detection. This paper focuses on developing an abstract-level autonomous anomaly detection model that provides continuous monitoring of vehicle behaviors over a very large area using unsupervised machine learning. Taking advantage of the advanced sensing and imaging capability of today's digital camera systems, our model may enable anomalous traffic situation detection for wide area traffic monitoring which is not achievable solely by human observers.

Several factors make building such a system challenging. Firstly, it is very hard to define a perfect normal region from which the abnormal vehicles can be differentiated. Secondly, normal traffic situations may evolve over time, so it is critical for the system to be adaptive to such changes. Thirdly, modeling and learning algorithm design can be difficult since there may be thousands of vehicles appearing simultaneous in the area. Last but not least, the requirement to handle such large volume of inputs and provide real-time monitoring, imposes stringent requirements in computation performance.

Many techniques have been studied for anomaly detection [1]. In [2], outliers are detected by a kernelized one-class SVM classifier. Reference [3] uses replicator neural networks to provide a measure of outlierness. Nearest neighbor approaches assume that normal data occur in dense neighborhood while anomalies lie far from their neighbors [4]. Reference [5] improves the method by using a rank based approach. Bayesian network based methods such as [6] have also been well studied. More specifically, traffic anomalies are detected from camera images by trajectory clustering [7] or using Gaussian mixture model [8]. None of these approaches systematically scale to very large areas, nor provide explicit reasoning of the causes of the anomaly.

In this paper, we present an autonomous traffic anomaly recognition and detection (AnRAD) model based on cogent confabulation [9], which is a computation model that mimics human information processing. It extracts the conditional probability among symbolic representations of features in an unsupervised environment. Thus the large area can be partitioned into smaller zones with independent models. Then, a knowledge base (KB) is built for each zone by feeding traffic records into properly modeled knowledge networks. When new traffic information is received, anomaly scores will be calculated by means of likelihood-ratio test for the observed events. Events with high anomaly scores will be marked as potential anomalies and the reasoning is achieved by the identification of key lexicons. The unique features of this platform can be summarized as follows:

1. The model is able to handle a large volume of vehicle traces over a big area, which is not considered in [7][8]. The surveillance zones are estimated and partitioned to balance the computation load and the statistical model is more locally accurate.
2. The confabulation based model has low complexity for both training and recall. Therefore, the system can be trained even during operating time, and this enables continuous improvements to the KB quality.
3. By proper modeling, the system is capable of capturing the contextual information among vehicles and their neighbors. Thus, abnormal events caused by interaction

978-1-4799-3564-2/14/$31.00 ©2014IEEE

among vehicles, such as tailgating, can be detected as well. Such events are not considered in previous research.

4. The overall system has a hierarchical architecture and the work load in each level of the hierarchy is inherently parallel. This can be further exploited in future work to provide real-time computing capabilities.

The rest of the paper is structured as the follows. Section II provides the background concepts of cogent confabulation. Section III elaborates the designs of the model. Section IV presents experimental results and Section V concludes the current progress and discusses the potentials of future research.

II. BACKGROUND

Cogent confabulation is a cognitive computing model that mimics the learning, information storage and recall process of the human brain. It is proposed in [10] that aspects of learning and cognition can be formed from four major components: a universal symbolic system to represent objects, knowledge links between symbols, confabulation and action command launching. We will briefly describe each of these four components.

The confabulation model represents the observation using a set of features. These features construct the basic dimensions that describe the world of applications, e.g. vehicle speed and coordinates. Different observed attributes of a feature are referred as *symbols*. The set of symbols used to describe the same feature forms a *lexicon* and the symbols in a lexicon are exclusive to each other. *Knowledge links* (*KL*) are established among lexicons. They are directed edges from the source lexicons to target lexicons. Each knowledge link is associated with a matrix. The *ij*th entry of the matrix gives the conditional probability $\log[p(s_i|t_j)]$ between the symbols s_i in the source lexicon and t_j in the target lexicon. The knowledge matrix is constructed during training by extracting and associating features from the inputs.

The cogent confabulation model has close resemblance to a neural system. The symbols are analogous to neurons and knowledge links between symbols are analogous to synapses between neurons. Whenever an attribute is observed, the corresponding symbol (i.e. neuron) is activated, and an excitation is passed to other symbols (i.e. neurons) through knowledge links (i.e. synapses).

The excitation of a symbol t in lexicon l is calculated by summing up all incoming knowledge links:

$$el(t) = \sum_{k \in F_l} \left(\sum_{s \in S_k} I(s) \ln \left(\frac{p(s|t)}{p_0} \right) + B \right) \quad (1)$$

where F_l denotes the set of lexicons that have connections to l, and S_k is a set that consists the collections of symbols in lexicon k; $I(s)$ is the firing strength of source symbol s, and it is set to 1 if s is observed without ambiguity; p_0 is the minimum probability that is considered informative. Parameter B is a constant called *band gap*, it is 0 if none of the active source symbols in S_k has knowledge links go into t. The band gap ensures that symbols with more KLs receive higher excitation over those with fewer KLs.

As we can see, the excitation level of a symbol is actually its log-likelihood given the observed attributes in other lexicons. In [11], the excitation levels are used to resolve the ambiguity in observation via maximum likelihood inference. In this paper, the excitation level enables us to detect anomaly using the *likelihood ratio* method.

When compared to other schemes, the training and recall process in the confabulation model are simple however massively parallel. Since the model is highly configurable, the system can be easily modified to fit diversified applications, or be optimized for better performance. Finally, because training and recall share the same knowledge data structures, the model offers unsupervised learning and online updating capabilities.

A previous research in [11], which performs confabulation-based text recognition on a high performance computing (HPC) cluster, demonstrates the model's ability to handle incomplete data and to capture the causality between observations.

III. MODEL DESIGN

In this section, the detailed design aspects will be discussed. This covers the partition of the monitoring area into detection zones, the feature set, and the topology of knowledge graph that consists of knowledge links and lexicons, the training and detection algorithms, and the incremental learning of the knowledge base.

The input to the system for both training and recall are traffic data (extracted from motion video) formatted as series of vehicle records ordered by time. Each record consists of a timestamp, vehicle type, the location and speed of the vehicle. We assume that there is no tracking information available that correlates records in different time instances.

A. Surveillance Zone Partition

The whole monitoring area is first divided into multiple smaller detection zones in the aid of training data. The benefits of this step are manifold:

1. It is possible that thousands of vehicles exist at the same time in the area. The complexity of finding the nearest neighbors is a quadratic function of the number of the vehicles in the detection zone. This is computationally expensive as the monitoring area gets bigger and the number of vehicles in the area increases. Partitioning the monitoring area into smaller zones and processing each zone independently can effectively reduce the complexity.
2. Each detection zone will be trained and operated as an independent unit, thus the workload can easily be parallelized and the knowledge base can be stored in a distributed manner.
3. The number of possible attributes of certain features, e.g. vehicle coordinates, is directly proportional to the area of the zones. Therefore the partitioning method effectively reduces the number of symbols in a lexicon, and reduces the complexity of the confabulation model.
4. The traffic situation varies from location to location. Therefore it is not appropriate to describe the whole area by a single model, since vehicles may exhibit different behaviors in different zones. Zone partitioning helps to improve the accuracy of the model.

Because the number of vehicles directly determines the

computation workload of the training and recall processes, it is used as the criteria for zone partitioning. The partition algorithm is described in Figure 1. In general the algorithm divides the zones based on the average traffic density and ensures that none of the monitoring zone has more than the Partition Constraint N of vehicles appearing at the same time slot in the training set. The resulting zones are organized in a sibling tree structure. Each parent node is associated with four child zones, which are derived by directly splitting the parent zone into four equally-sized patches.

```
1  Find the MAX and MIN values of vehicle coordinates
2  Define the root zone and set Z = {root}
3  For each time slot t in the training set
4     reset vehicle_count of each zone in Z
5     For each record x at time slot t
6        Find zone z that x.location falls into
7        if z.vehicle_count exceed limit N
8           Split z into 4 child zones z[1-4]
9           Update vehicle_count of z[1-4]
10          Update set Z
11       Else
12          z.vehicle_count++
13       End if
14    End for
15 End for
```

Figure 1. Zone partitioning algorithm.

An example of the resulting zone partitioning is shown in Figure 2. The grids with yellow borders are detection zones.

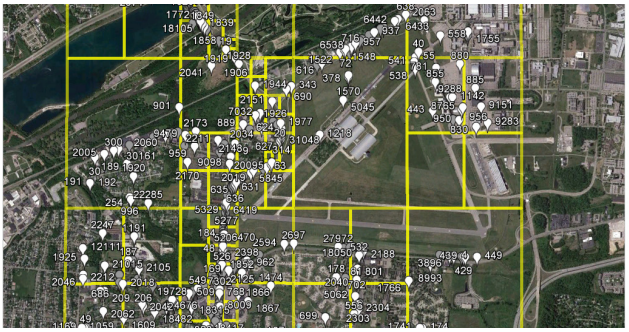

Figure 2. A zone partitioning example.

Furthermore, each zone is designed to have a "buffer area" which is a margin overlapping with the adjacent zones. Vehicles in these margin areas will not be tested for anomaly detection, but be used to generate supporting information, such as neighbors to the target vehicles, or previous records of the current target vehicles. This allows the system to consider the vehicles about to enter or already exited the zone.

B. Lexicon Definitions and Feature Extraction

After the zone partition, the confabulation model can be built for each detection zone. The first task is to define the lexicons, namely features that are used to describe the vehicle behavior in traffic. Because the confabulation model was first introduced for applications in natural language processing [11], in this work we continue the tradition and refer the set of lexicons describing an observation as a "*sentence*".

In the anomaly detection problem, we consider the behavior of a vehicle within the context of its neighbors during the current and previous observations. If we define all observations made in the same time slot as a *frame*, the detection method involves three consecutive frames. Four classes of objects are defined: *target*, *neighbor*, *auxiliary center*, and *supporter*. Each vehicle appearing in a detection zone at frame t is considered as a *target*. The ten vehicles closest to the target in the same frame are defined as *neighbors*. Based on the current location and speed of target, we can estimate its location in the previous frames. The estimated targets in frames t-1 and t-2 are referred as the *auxiliary centers*. The nearest ten neighbors of the auxiliary center in the corresponding frame are called the *supporters*. Figure 3 shows an example of the four types of vehicle records. A sentence (i.e. the set of lexicons for the detection problem) is generated based on the observations of each target within the context neighbors, auxiliary centers and supporters.

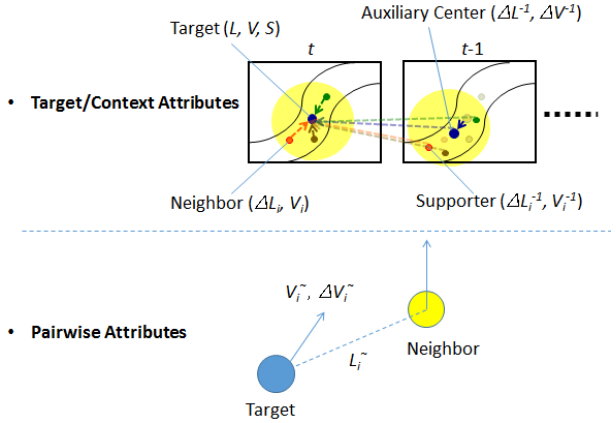

Figure 3. Classification of vehicle records and correspondent lexicons.

Three lexicons are used to describe the basic attributes of a target vehicle, target location (L), target velocity (V), and target size (S). The target location is expressed in geographic latitude and longitude and is discretized to levels of approximately ten meters. The target speed is expressed as the combination of ground speed and direction. The target size is discretized to reflect the five different vehicle categories: sedan, truck, SUV, moving truck and 18-wheeler.

Two lexicons are associated with each auxiliary center, center displacement (ΔL^{-t}) and center acceleration (ΔV^{-t}), t = 1, 2. The center displacement is the distance between the target and an auxiliary center represented by the displacement in latitude and longitude. It describes how far the target moved in the last 1 and 2 frames. The center acceleration specifies the change of velocity (speed and direction) of the target during the last 1 and 2 frames.

Two lexicons are associated with each neighbor or supporter. The relative location lexicon (denoted as ΔL_i for the ith neighbor and ΔL_i^{-t} for the ith supporter in frame -t) gives the relative position of the neighbor (or supporter) with the respect to the target. The velocity lexicon (denoted as V_i, for the ith neighbor and V_i^{-t} for the ith supporter in frame -t) specifies the

velocity of the neighbor (or supporter) as a combination of the speed and direction.

Three lexicons are used for pairwise attributes that describes the relation between the target and each of its neighbors. Pairwise location lexicon ($L_i^{\sim}$) specifies the distance (in meters) and direction (in degrees, relative to neighbor's moving direction) between the target and the *i*th neighbor. Pairwise speed lexicon ($V_i^{\sim}$) specifies the target's absolute speed (in m/s) and relative direction (in degrees) with respect to the neighbor's direction. Pairwise speed changes lexicon ($\Delta V_i^{\sim}$) captures the target relative speed (in m/s) and relative direction (in degrees) with respect to the *i*th neighbor.

In total, 97 lexicons (features) are used to describe the status and context of a target vehicle. The set of observed attributes of these features form a sentence, which is the basic input for the confabulation training and recall processes. Every vehicle in the detection zone is treated as a target; therefore there is a sentence for each one of them.

C. Overall confabulation model

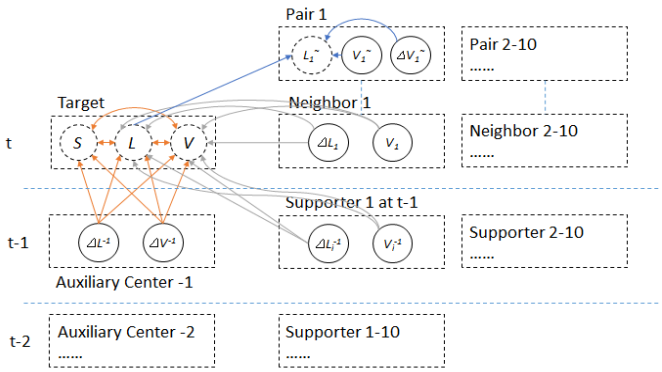

Figure 4. Illustration of knowledge links among lexicons.

Figure 4 shows the overall confabulation model with lexicons and the knowledge links among them. Lexicons *S*, *L*, *V* and $L_i^{\sim}$, $1 \le i \le 10$, are represented using dashed circles. Each one of them corresponds to a general category of abnormal behavior of the target vehicle, such as abnormal location or speeding, inconsistency between vehicle size and its status, and abnormal interactions with neighbors. We refer these lexicons as the *key lexicons* and others as the *regular lexicons*. Only the excitation levels of the key lexicons need to be evaluated. All other lexicons only provide supporting context for them. A key lexicon can have incoming knowledge links from all other lexicons while a regular lexicon can only have outgoing knowledge links.

Comparing to the original confabulation model [10][11], this work adopts a new feature called "shared links". It is motivated by the observation that for most of the time, a target vehicle may only have a small number of neighbors (usually less than five). Therefore the neighbor lexicons with large indices may have few training data. Figure 5 shows how the current training data reduce as the neighbor lexicon index increases. The amount of available training data directly affects the accuracy of the knowledge link derived. Meanwhile neighboring vehicles with similar behavior may be associated more often during multiple times of observation, because the neighbors are mapped to lexicon indices based on their relative distance to the target. Our model overcomes these problems by letting the knowledge links initiated from neighbor lexicons share the same knowledge matrix. In this way the training data for neighbors can all contribute to the same knowledge base and hence reduces the false alarm rate. Moreover, shared knowledge helps to reduce the size of KB since only one matrix is maintained for multiple knowledge links.

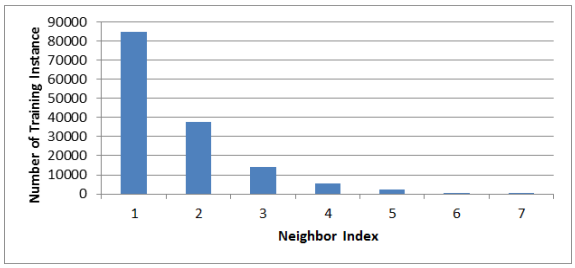

Figure 5. The number of available training instances decreases when neighbor lexicon index increases.

The detection process is divided into three stages: self-check, cross check and pairwise check. For self-check, the target status, namely the location, speed and type are checked with its own attributes: the *S*, *L* and *V* lexicons are connected to each other; the ΔL^{-t} and ΔV^{t} lexicons are also connected to the current status and served as the triggering sources. The vehicle status should also pass cross check to be justified as normal. Thus *S*, *L* and *V* lexicons are also connected with neighbor and support lexicons, so that the "supporters" agree with the target vehicle's behavior. Finally, the pairwise location lexicon $L_i^{\sim}$ is used as the target lexicon to detect abnormal interactions among vehicles. It is tested against the target location *L* and pairwise speed information $V_i^{\sim}$ and $\Delta V_i^{\sim}$.

D. Confabulation Training and Recall

The confabulation algorithms consist of two procedures: unsupervised learning (*training*) and anomaly detection (*recall*). Both procedures are based on the same knowledge base data structure.

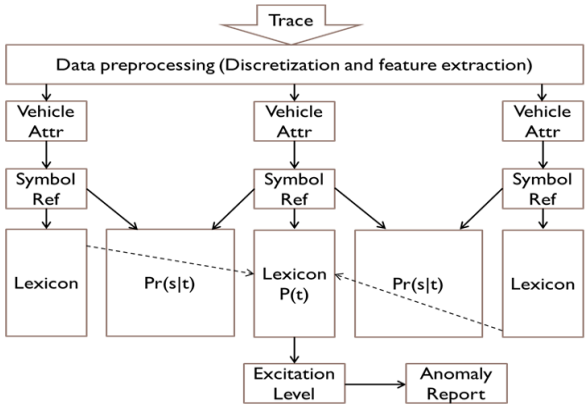

Figure 6. Training and recall using confabulation model.

Figure 6 shows the task flow of the training and recall procedures. The observed traffic data are first preprocessed for

zone partitioning and feature extraction. The observed attribute of the lexicons are collected and assembled into a sentence. Each observed attribute is mapped into a globally unique reference number called a *symbol* using the two-level hash function. For a given lexicon, all attributes observed during the training process form its candidate set. After preprocess, for each knowledge link, the co-occurrence of the source and target symbols are counted and the logarithm conditional probability is calculated. Finally, the knowledge matrices are stored into a knowledge base. The steps are summarized in Figure 7.

```
1  Reset all KLs and lexicons
2  For each input sentence T
3     Map elements in T into reference numbers
4     For each symbol o_i in T
5        Add o_i to the candidate set of lexicon[i]
6        For each symbol o_j in T
7           If (i != j), KL[o_i, o_j].count++
8        End for
9     End for
10 End for
11 Finalize value of each KL and store into KB
```

Figure 7. AnRAD training procedure.

The same preprocessing and feature extraction procedures are performed in the recall procedure. The excitation levels $el(t)$ of the key lexicons are calculated, which give the likelihood values of the observations given the context of the target and neighbors. For each key lexicon, the excitation levels of other symbols in its candidate set are also calculated and the symbol t_{best} with the highest excitation level is obtained. An *anomaly score* $as(l,t)$ is defined based on the *likelihood ratio test* as shown in Equation (2).

$$as(l,t) = \frac{el(t_{best}) - el(t)}{el(t_{best})} \quad (2)$$

A high anomaly score for observed symbol t indicates that the likelihood of t is much lower than the likelihood of the typical attributes in the current traffic context.

Since an abnormal event usually lasts for multiple time slots, a vehicle is reported as "abnormal" only when the anomaly score of one of its key lexicons exceeds a threshold for three continuous frames. This constraint is specified by Equation (3).

$$\min_{j=0,1,2} \{as^{-j}(l,t)\} > \theta_l \quad (3)$$

The overall procedure of the anomaly detection is summarized in Figure 8.

```
1  For each input sentence T
2     Map elements in T into reference numbers
3     For each target key lexicon l and observation t
4        Calculate el(t) by Eq. (1)
5        For each candidates t_j in lexicon[l], find el(t_best)
6        Calculate as(l, t) by Eq. (2)
7        If Eq. (3) satisfied, flag anomaly on the target
9     End for
10 End for
11 Output anomaly reports
```

Figure 8. AnRAD recall procedure.

E. *Incremental Learning*

One aspect of the traffic anomaly detection problem is that it is hard to obtain an accurate model without a sufficient volume of training data. Therefore it is important to keep learning even during operation time. Since AnRAD is probability-based and the data structures for training and detection procedures are consistent, this allows incremental learning to refine the model during the detection operation. To implement this feature, trained KBs are first loaded into memory and organized in the structure as shown in Figure 4. When new data are received, the system updates the event counter for each knowledge link. The probability calculations are performed periodically based on the updated event counter. As a result the KLs are updated on-the-fly and used for future anomaly detection.

Efficient storage of the knowledge matrices and searching in the feature space are computationally challenging due to the large size of the feature space. For example, in order to provide sufficient resolution for a typical zone of 500×500 (meter^2), more than 2500 location symbols are needed. Also, about 400 symbols are needed to describe the relative distance between a target and one of its neighbors within a 100-meter radius. Each KL matrix has about 10^6 entries, and there are 30 knowledge links in each detection zone.

Fortunately, the knowledge links are sparse matrices. For example, not all locations in the detection zone can be target location since most vehicles only appear on roads. In average, only 1% of the entries in the KL matrices are actually visited in a 90-minute training data. This number does not increase much when the training data get longer. For a 240-minute training sequence, the nonzero entries in KL matrices are about 2%.

In this work, the knowledge matrix is organized using a 2-level hash table. The first level maps each application specific symbol into a globally unique reference number. It isolates the confabulation training and recall from application specific data types, thus increases the adaptability of the approach. After the symbol mapping, the pairs of reference numbers between connected lexicons are used as keys to locate the entries of the KL in the second-level hash tables. The second-level hash function exploits the sparse characteristic of the KLs and improves the search performance. The average collision rate of the second-level level hash table is 1.4. Figure 9 shows that the first-level hashing reduces the pattern combinations of ΔL_i and L from 1,000,000 to about 120,000 (blue line); then the second-level hashing further reduces the storage requirement by another 90% (red line), which is only 1% of the original size.

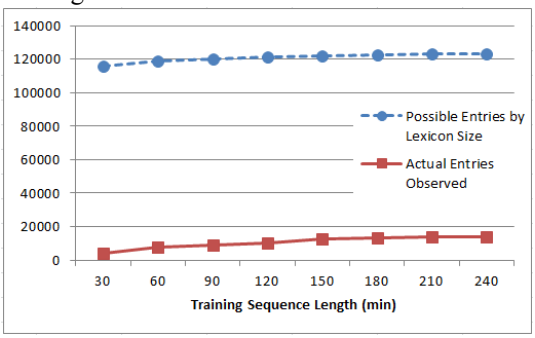

Figure 9. Storage reduction by the two-level hashing method.

IV. Experimental Results

Experiments are conducted to evaluate the performance of AnRAD. The monitoring data cover a 10×10 (mile^2) area.

A. Single Detection Zone Tests

In this test, one detection zone of 500×500 (meter^2) with moderate traffic density is randomly selected from the monitoring area. The training data has 240 minutes of normal traffic. The testing data include 10 minutes of normal traffic data with manually inserted abnormal events representing typical hazardous vehicle activities. The abnormal events include cars deviating from the road, speeding, tailgating, 18-wheelers running at abnormal speed, and cars unexpectedly stop-and-go in the middle of the road.

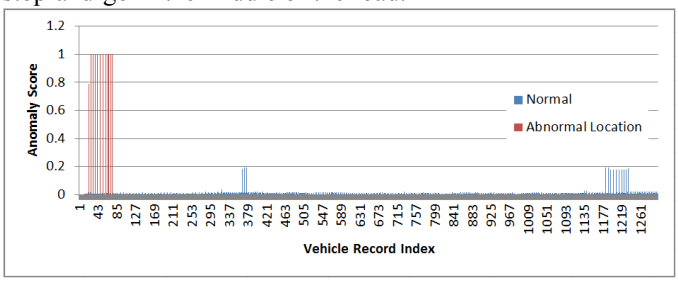

Figure 10. Anomaly score of location key lexicon.

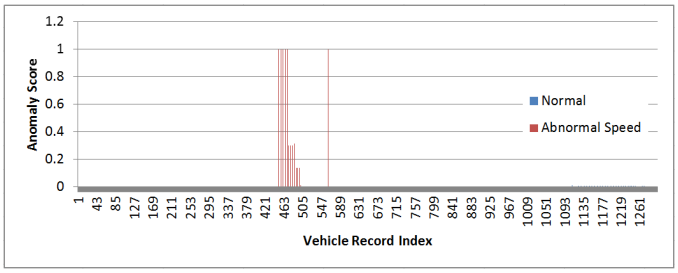

Figure 11. Anomaly score of speed key lexicon.

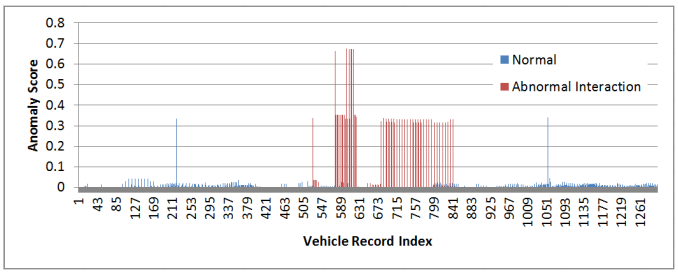

Figure 12. Anomaly score of the first neighbor pair lexicon.

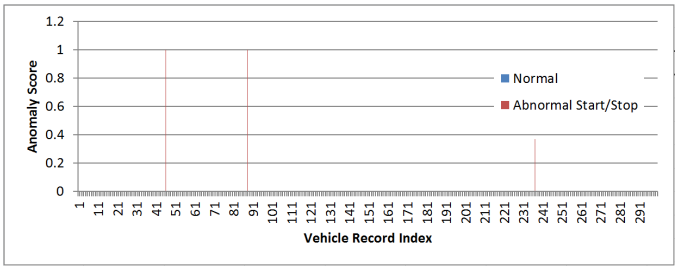

Figure 13. Anomaly score of speed lexicon for stop-and-go events.

Figures 10, 11, 12 and 13 show the anomaly scores of the key lexicons for all vehicles in the testing data when abnormal events appeared. The X-axis of all the plots gives the indices of vehicles. The Y-axis gives the magnitude of the anomaly scores. Each figure corresponds to a type of abnormal activities. The anomaly scores of the manually-inserted abnormal targets are highlighted in red in each figure. As we can see, the anomaly scores in red are significantly higher than the normal ones, indicating anomalies can be detected by a decision threshold. Furthermore, the anomaly scores demonstrate obvious temporal continuity for most categories of abnormal events, except that of abnormal stop-and-go of vehicles, which give short spikes only when the moving status changes.

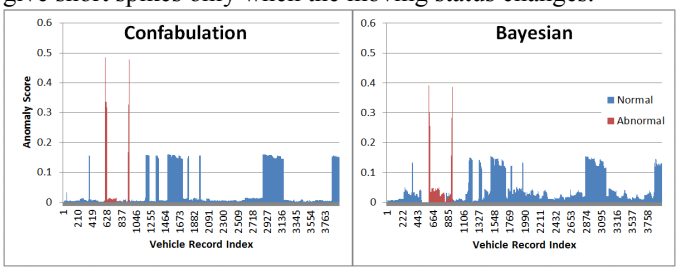

Figure 14. Comparison of anomaly score between excitation level and Bayesian probability.

Different from the calculation of the maximum *a posteriori* probability in a traditional Bayesian model [6], the prior probability of an observed attribute weights less in the confabulation model. The proposed method is more effective in detecting outliers that does not appear abnormal by itself but only becomes abnormal in a specific context. For example, an 18-wheeler by itself is quite common in the detection zone. However this vehicle type becomes abnormal when it drives at a speed that is normally observed on sedans. As we can see from Figure 14, confabulation based anomaly scores for such events are 20% higher than those calculated with Bayesian models, therefore providing a better detection chance.

B. Computation Costs and Partition Constraints

In the second test, synthetic traces are generated over the same big area to compare the computational workload. Since the current testing system has not been optimized for multi-threading or parallel processing, the results are used for general trend comparison only.

In Figure 15, the X axis is the total number of vehicles appeared in the entire monitoring area, and the Y axis is the average computation time for each frame over the whole area. Different partition constraints (values of N in Figure 1) are tested. It is observed the smaller N is, or in other words the finer the zones are divided, the smaller the computation time per frame is. This is mainly because smaller zones generate fewer candidates for the lexicons, therefore reducing the computation needed for searching knowledge entries and comparing candidate excitation levels. The only exception is that computing times for both training and recall with N=10 are slightly larger than the ones with N=20. This is due to the fact that with N=10, some of the zones become smaller than the overlapping margins between adjacent zones. This actually increases the total area of zones significantly and adds redundant calculation on the vehicles in the overlapping areas. Therefore N=20, which limits the maximum number of vehicles in each zone to 20, is chosen for the partitioning algorithm.

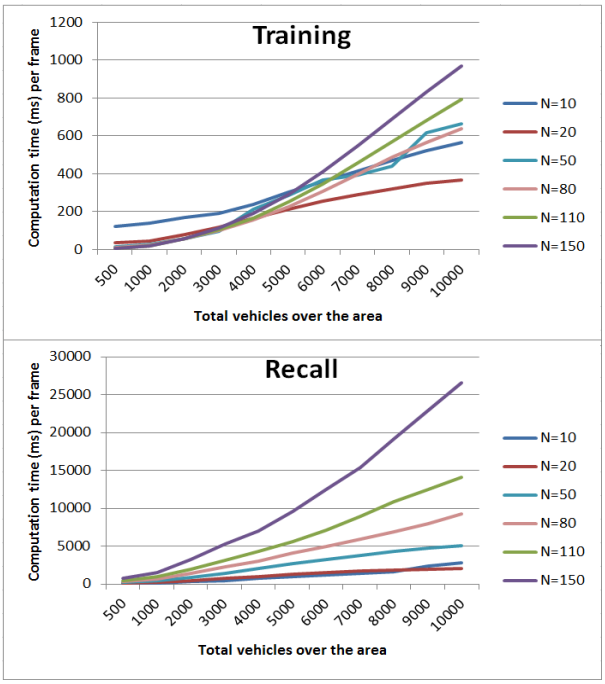

Figure 15. Computation time costs for the whole area with different zone partition constraints.

C. *Incremental Learning*

In this final experiment, training sequences are streaming into the system in an incremental fashion to verify that the proposed model can evolve and improve as more and more training data become available. In this test, one detection zone is selected and the training data are presented to the system gradually in 10-minute-long segments. After every incremental training step, a 10-minute-long normal traffic sequence (which is different from any of the training segment) is used for testing, and the number of detected anomalies in any key lexicons is reported. Since the testing sequence consists of normal vehicles, these anomaly counts are false alarms and expected to decrease with better training.

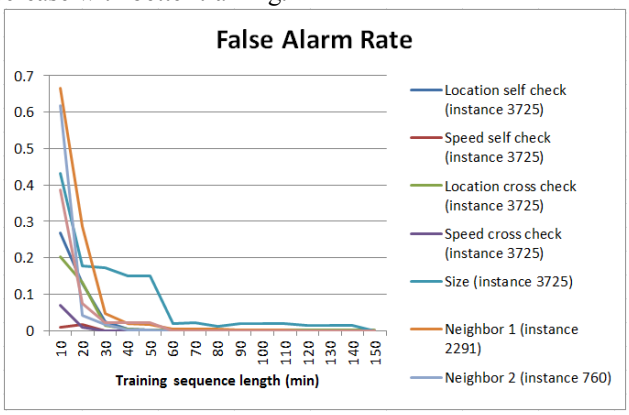

Figure 16. False alarm rate against volume of training set

Figure 16 shows that the false alarm rate reduces as the training goes on. The false alarm rate is calculated as the ratio of reported anomalies over the total number of checked instance. Each line represents a false alarm rate of one category of anomaly category (i.e., a key lexicon). With insufficient training, e.g. 10 minutes of training sequence, the false alarm rate can be as high as 60%. Meanwhile with more training, the false alarm rate drops quickly, and the system gives almost no false alarms after trained with 150 minutes of data. This indicates that the model has been updated incrementally and become more accurate when trained with more data.

V. Conclusions & Future Work

This paper presents an anomaly detection system based on the cogent confabulation model and its application in abnormal vehicle behavior detection. The model construction, data preprocessing, training and recall algorithms are presented. The experimental results show that synthetic anomalies can be easily detected in this model. Although the AnRAD system has shown promising potential in anomaly detection, many improvements are still needed. For example the parameters such as anomaly score thresholds need to be tuned carefully in the current system. Future work is needed to find a more general decision criterion.

VI. Acknowledgement and Disclaimer

This work was partially supported by the National Science Foundation under Grants CCF-1337198 and CCF-1337300, and the Air Force Research Laboratory, under contract FA8750-12-1-0251.

Received and approved for public release by AFRL on 11/07/2013, case number 88ABW-2013-4667. Any Opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of AFRL or its contractors.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly Detection: A Survey," ACM Computing Surveys (CSUR), Volume 41 Issue 3, July 2009.

[2] V. Roth, "Outlier Detection with One-class Kernel Fisher Discriminants," Proc. of the Conference on Advances in Neural Information Processing Systems, 2004.

[3] S. Hawkins, H. He, G. J. Williams, R. A. Baxter, "Outlier Detection using Replicator Neural Networks," Proc. of International Conference on Data Warehousing and Knowledge Discovery, 2002.

[4] J. X. Yu, W. Qian, H. Lu, and A. Zhou, "Finding Centric Local Outliers in Categorical/Numerical Spaces," Knowledge and Information Systems, March, 2006.

[5] H. Huang, K. Mehrotra, and C. K. Mohan, "Rank-based outlier detection," Journal of Statistical Computation and Simulation, vol. 82, pp. 1–14, 2011.

[6] A. A. Sebyala, T. Olukemi, and L. Sacks, "Active platform security through intrusion detection using naive Bayesian network for anomaly detection," Proc. of the London Communications Symposium, 2002.

[7] Z. Fu, W. Hu, T. Tan, "Similarity based vehicle trajectory clustering and anomaly detection," Proc. of International Conference on Image Processing, 2005.

[8] H. Sheng, C. Li, Q. Wei and Z. Xiong, "Real-time Detection of Abnormal Vehicle Events with Multi-feature over Highway Surveillance Video," Proc. of International Conference on Intelligent Transportation Systems, 2008.

[9] R. Hecht-Nielsen, "Confabulation Theory: The Mechanism of Thought," Springer, August 2007.

[10] R. Hecht-Nielsen, "The Mechanism of Thought," Proc. of International Joint Conference on Neural Networks, 2006.

[11] Qinru Qiu, Q. Wu, M. Bishop, R. Pino, and R. W. Linderman, "A Parallel Neuromorphic Text Recognition System and Its Implementation on a Heterogeneous High Performance Computing Cluster," IEEE Transactions on Computers, 2013.