World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:15, No:6, 2021

A Real-Time Bayesian Decision-Support System for Predicting Suspect Vehicle's Intended Target Using a Sparse Camera Network

Payam Mousavi, Andrew L. Stewart, Huiwen You, Aryeh F. G. Fayerman

Abstract**—**We present a decision-support tool to assist an operator in the detection and tracking of a suspect vehicle traveling to an unknown target destination. Multiple data sources, such as traffic cameras, traffic information, weather, etc., are integrated and processed in real-time to infer a suspect's intended destination chosen from a list of pre-determined high-value targets. Previously, we presented our work in the detection and tracking of vehicles using traffic and airborne cameras. Here, we focus on the fusion and processing of that information to predict a suspect's behavior. The network of cameras is represented by a directional graph, where the edges correspond to direct road connections between the nodes and the edge weights are proportional to the average time it takes to travel from one node to another. For our experiments, we construct our graph based on the greater Los Angeles subset of the Caltrans's "Performance Measurement System" (PeMS) dataset. We propose a Bayesian approach where a posterior probability for each target is continuously updated based on detections of the suspect in the live video feeds. Additionally, we introduce the concept of 'soft interventions', inspired by the field of Causal Inference. Soft interventions are herein defined as interventions that do not immediately interfere with the suspect's movements; rather, a soft intervention may induce the suspect into making a new decision, ultimately making their intent more transparent. For example, a soft intervention could be temporarily closing a road a few blocks from the suspect's current location, which may require the suspect to change their current course. The objective of these interventions is to gain the maximum amount of information about the suspect's intent in the shortest possible time. Our system currently operates in a human-on-the-loop mode where at each step, a set of recommendations are presented to the operator to aid in decision-making. In principle, the system could operate autonomously, only prompting the operator for critical decisions, allowing the system to significantly scale up to larger areas and multiple suspects. Once the intended target is identified with sufficient confidence, the vehicle is reported to the authorities to take further action. Other recommendations include a selection of road closures, i.e., soft interventions, or to continue monitoring. We evaluate the performance of the proposed system using simulated scenarios where the suspect, starting at random locations, takes a noisy shortest path to their intended target. In all scenarios, the suspect's intended target is unknown to our system. The decision thresholds are selected to maximize the chances of determining the suspect's intended target in the minimum amount of time and with the smallest number of interventions. We conclude by discussing the limitations of our current approach to motivate a machine learning approach, based on reinforcement learning in order to relax some of the current limiting assumptions.

P. Mousavi*, A. L. Stewart*, H. You, and A. F. G. Fayerman are with MDA, Richmond, BC Canada (e-mail: payam.mousavi@mda.space, andrew.stewart @mda.space, huiwen.you@mda.space, aryeh.fayerman@mda.space). * These authors contributed equally to this work.

Keywords**—**Autonomous surveillance, Bayesian reasoning, decision-support, interventions, patterns-of-life, predictive analytics, predictive insights.

I. Introduction

REAL-time collection, fusion, and processing of multiple data sources are essential ingredients for decision-support in the future surveillance and defense systems. The ever-increasing quantities of multi-modal data collected by a large array of sensors and platforms is an opportunity to leverage the latest advances in Artificial Intelligence (AI) and Machine Learning (ML) in particular, to gain actionable intelligence in a timely manner and with minimal supervision. Recent revival of interest in AI/ML (and in particular deep learning) for defense applications is mostly the result of the exponential increase in the available compute as well as the improved infrastructure and technologies for sensing, storing, and accessing the available data sources that are essential for the success of such approaches. Arguably, some of the most remarkable advances have come in perception (e.g., detection and classification of objects in images) where deep neural networks trained with millions of labeled images can achieve super-human performance. Unfortunately, similar advances in reasoning and decision-making have yet to occur, as the current technologies cannot seem to address them effectively.

In this work, we tackle the task of predicting the intended target of a pre-identified suspect vehicle as it travels across a network of roads where several sensors, i.e., traffic cameras, are used to monitor the environment. Fig. 1 is an illustration of our problem based on the Los Angeles road network. Inferring the 'true intention' given a trajectory is generally an ill-posed problem if no further assumptions are made [1]. To make progress, we make several simplifying assumptions, some of which are to be removed in future work. First, we assume that there is a finite number of known high-value targets and that the suspect intends to travel to one of them. Second, we assume that our sensor network is sufficiently dense so that the suspect cannot evade detection as it travels towards the target. We will address this shortcoming in future work by allowing the operator to task a UAV to monitor any location not currently covered by our camera network.

We build on prior work by our team [2] that took full

advantage of the power of deep neural networks to detect and track objects in Full-Motion Video (FMV). Here, we propose a principled Bayesian method [3] that takes as input a time series of multiple sightings of a suspect vehicle to estimate continuously the posterior probability for each target location.

Fig. 1 Video streams are processed to identify and track a suspect vehicle in Los Angeles

The proposed approach has several advantages: It allows for the integration of an arbitrary number of data streams as they become available in the future (e.g., traffic information, weather, etc.). Contrary to approaches based on deep learning, the models are more transparent, and the updates in beliefs could be traced back and verified. Moreover, the decision thresholds can be adjusted to obtain a balance between precision and recall as the situation warrants. Finally, our system supports both an operator-on-the-loop mode, whereby the operator is in full control, only taking course-of-action (COA) recommendations from the system, as well as an autonomous mode that only prompts the operator for decisions in critical situations. A robust autonomous operation is critical to success of such systems as their scale grows. An additional novel idea we experiment with here is the use of 'Soft Interventions' (SI). Here SI refers to a road or intersection blockage, ultimately decided upon by the operator, which forces the suspect to re-route, thereby sometimes revealing their true intentions more quickly. This idea is inspired by a sub-field of statistics, namely Causal Inference (CI) [4]. Historically, CI has focused on determining cause-and-effect relationships in the field of epidemiology (among others), and has recently gained popularity in other sub-fields of ML [5]. The central idea is the observation that by actively intervening in an environment, we can distinguish between causes and spurious correlations, an essential component of effective prediction and decision-making. Here, we focus on simple interventions, e.g., temporary blockage of roads or intersections. More sophisticated interventions will be explored in future work.

To build intuition, we begin by presenting the details of our algorithm, demonstrating the main ideas and evaluating its performance on a smaller map generated using the road network of a neighborhood in Vancouver, British Columbia. We then scale the algorithm to a much larger graph based on a part of a publicly available PeMS dataset [6]. We present a high-level overview of our software architecture, enabling the real-time collection, aggregation, and processing of the streaming data. Finally, we will summarize the results and conclude with proposing a set of future directions.

II. Approach and Algorithms

We represent the street map as a directed bi-directional graph, as shown in Figs. 2 (A) and (C), with nodes corresponding to the hypothetical camera locations. The edges represent direct connections between nodes with their corresponding weights equal to the time it takes to travel between those nodes. Larger weight values correspond to longer travel times between those nodes. Two nodes are connected if and only if a vehicle can travel from one node to the other *and* the corresponding shortest route between those two nodes does not contain any other node in the set. Fig. 2 (A) is a toy example that covers a few blocks of Vancouver, British Columbia. We will use this example to describe our algorithm before proceeding to a much larger graph created using real traffic data. In Fig. 2 (A), the camera nodes are represented using blue circles, while red triangles represent the three potential target locations, Node 15, Node 16, and Node 17. The graph data structure representing our assumed connections between the nodes is shown in Fig. 2 (C). This example computes weight between nodes using the Manhattan distance [7].

To generate scenarios, we assume that the suspect will travel from an initial random node to a target node also selected randomly from a set of target nodes. A top-k (here with $k = 3$), set of shortest routes between those two nodes is computed using Djikstra's algorithm [8]. The suspect's selected route is randomly sampled from this set using an inverse weighting scheme that results in the shortest route being more probable. This route is dependent on the graph structure as well as the weights that could be continuously updated depending on traffic conditions, weather, or any interventions.

We use a Bayesian methodology to estimate $P(T_j|X_i)$, the probability of target j being the intended target, given the current node location i:

$$P(T_j|X_i) \propto P(X_i|T_j) \cdot P(T_j) \tag{1}$$

The probability $P(T_j)$ is *the prior*, our belief about the intended target location prior to the current observation. It is assumed initially to be $1/M$, with M being the number of targets. The term $P(X_i|T_j)$ is *the likelihood*, a measure of the probability of traversing node X_i given target T_j was the intended target. Note that, the normalization factor is in general intractable as it is not possible to enumerate an indefinite number of behaviors or targets. Here, the assumption that one of the finite number of targets is the correct one leads to a simple normalization: $\sum_{i=1}^{M} P(T_j) = 1$, marginalizing $P(X_i)$ over T_j. We estimate the likelihoods as the average frequencies of each node being traversed given an initial point and a target

calculated through many randomly initialized simulations. In principle, the likelihoods could incorporate more sophisticated models that could be either hand-crafted or learned, e.g., via reinforcement learning [9]. The likelihoods are pre-computed for a given graph and could potentially be pre-computed for different combinations of traffic, time, weather, etc. During operations, at each time step, the posterior probability of each target is calculated by multiplying the likelihood for that node by the current prior probabilities for each potential target. Each step then leads to an updated probability accounting for the full trajectory traversed by the suspect. Fig. 2 (A) shows an example of a trajectory, represented by green circles and lines. The suspect vehicle was first identified at Node 0, with the intended target (only known to the suspect) being Node 15. The real-time posterior estimates are shown in Fig. 2 (B) using solid lines. To implement SI, the idea is to impose minimal constraints on the network in order to gain maximum amount of information regarding the suspect's true intention. Fig. 2 (D) is a simplified schematic to give an intuitive understanding of the power of such interventions. If the suspect is currently at node X_i, the green circle in the figure, it is unclear which of the two targets, red circles, is the final destination. Moreover, if the suspect takes the edge e_2, no new information is revealed concerning the final destination. Removing this edge, i.e., blocking this street segment, forces the suspect to reveal its intentions by making a choice between the two remaining edges, e_1 and e_2.

More formally, at each step, if we are not sufficiently confident about the final destination, we consider an intervention (from a set of pre-selected road segments) that removes the edge with the maximum entropy [10]. This ensures that we have gained maximum information while updating the posteriors. The entropy, here measured in bits, for each outgoing edge is defined as, $S = -\sum_{i=1}^{M} P_i \log_2 P_i$, where P_i's correspond to the posterior probabilities of target i. The decision on whether or not to intervene is made by balancing the cost of taking an intervention, i.e., the inconvenience caused to others, with the benefit of gaining maximum information about the suspect. This preference is set by varying a threshold which will be discussed later. A concrete example, demonstrating the benefits of interventions is shown in Figs. 2 (A) and (B). Two trajectories starting from Node 0 travelling to Node 15 are shown in green and purple. The green trajectory represents a scenario without interventions. The corresponding target posteriors are plotted in Fig. 2 (B) with solid lines. At time t_3, we observe that target Node 16 is incorrectly determined to be the most likely target. The purple trajectory corresponds to the same scenario, except the operator intervened to close the road segment connecting Node 4 and Node 7. This forced the suspect to re-route by travelling to Node 9 on their way to the intended target at Node 15. The new post-intervention posteriors are shown as dotted lines in Fig. 2 (B). We see that the adjusted posteriors correctly identify Node 15 as the most likely target.

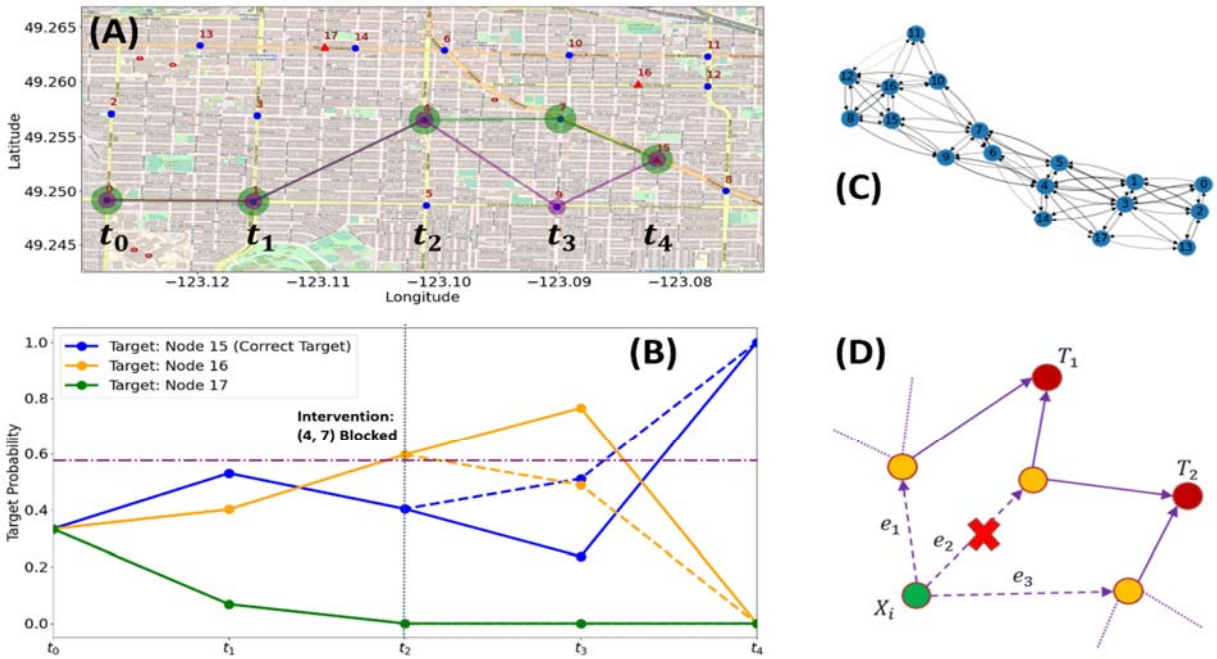

Fig. 2 A simple example to demonstrate our approach, (A) A neighborhood in Vancouver, British Columbia, represented using 18 nodes and 89 directed edges. The blue circles correspond to camera locations, while the three red triangles are arbitrarily chosen target locations. Two paths are shown: The green path corresponds to a scenario without interventions, while the purple path is the result of blocking the road from Node 4 to Node 7, (B) Time-dependent target probabilities for the case without intervention, solid lines, and with interventions, dotted lines. The purple horizontal line corresponds to the threshold used for decision-making. (C) Graph data structure corresponding to the map. (D) A simple diagram to demonstrate the utility of interventions. Intervening on $e_2$results in the largest gain in information concerning the suspect's intention

We will next discuss the dataset used to create a much larger graph based on Los Angeles, followed by the tuning and evaluation of both graphs in subsequent sections.

III. Datasets

The Caltrans PeMS [6] is a publicly available traffic dataset provided by the California Department of Transportation. The PeMS dataset is collected from nearly 40,000 individual sensors spread across all major metropolitan areas of California. These loop sensors measure vehicle count and are sampled every 30 seconds. Caltrans subsequently processes the data to return average velocity and flow rate for each sensor [11]. Due to the large data volume, liberal usage policy, and quality of data, the PeMS dataset is frequently used in the literature as a training source for traffic prediction [12]-[14].

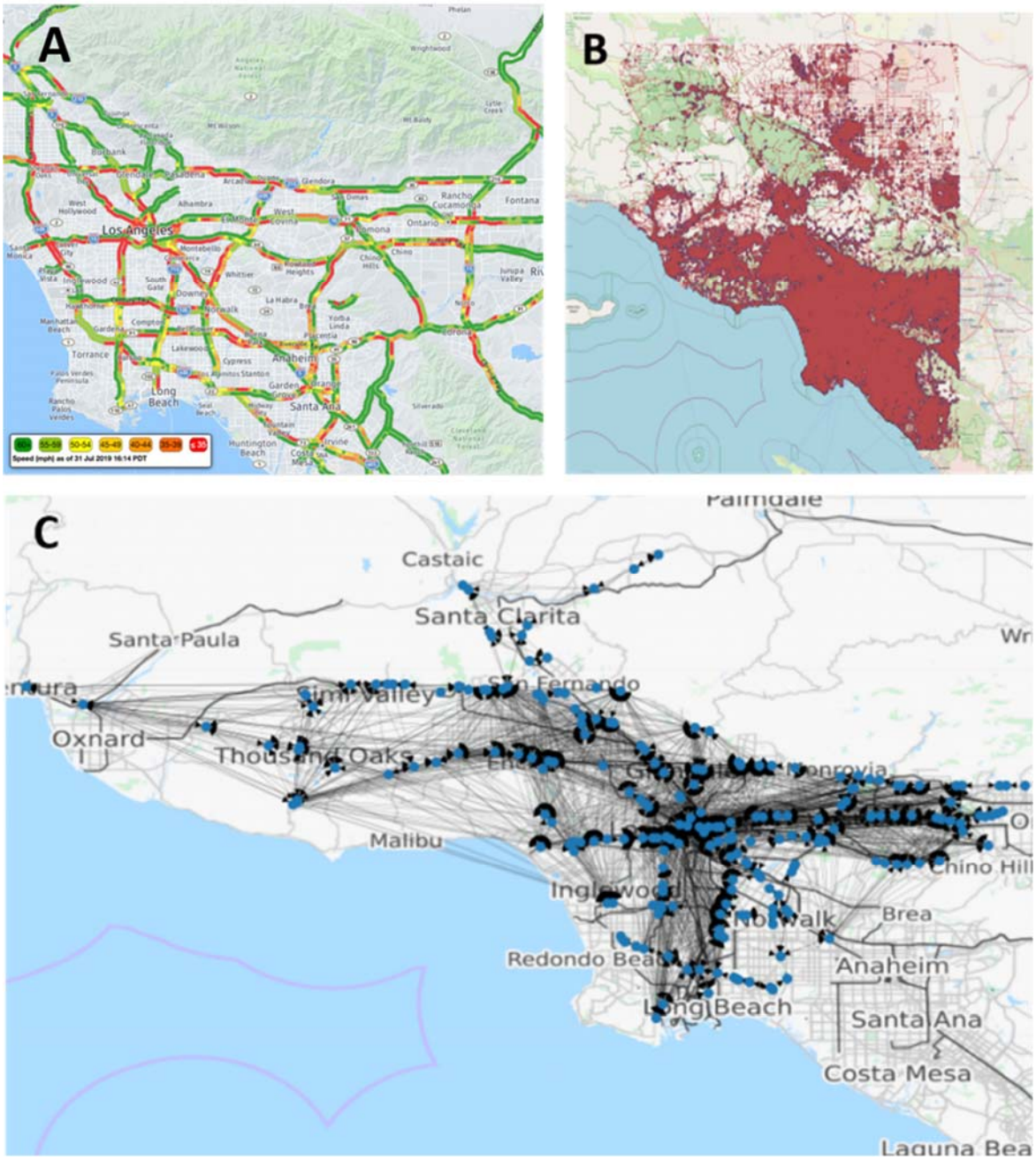

Fig. 3 (A) Visualization of the PeMS data for District 7 provided by Caltrans [6], (B) OpenStreetMaps node and edges, in red, encompassing the greater Los Angeles region, (C) Greater Los Angeles region overlaid with a subsample of the PeMS nodes

The specific area of interest selected was the PeMS District 7 region that encompasses the greater Los Angeles area, outlined in Fig. 3 (A). District 7 data are collected from 2055 sensors and are updated at 5-minute intervals. One key element that is not provided in the dataset is an underlying structure to the road network. Yu et al. [12] construct a weighted adjacency matrix by computing the distance between all pairs of nodes with a threshold value to disconnect nodes sufficiently far apart. This strategy constructs a graph where edges do not necessarily correspond to any realistic route between nodes. For example,

a node on a northbound lane of a highway may be physically close to a node on the adjoining southbound lane resulting in a low edge weight. However, the travel distance of a vehicle on the road network may be significantly larger. To better inform the graph construction, OpenStreetMap (OSM) [15] provides a comprehensive mapping dataset including road connections. The raw XML files, provided by OSM, were converted into a graph network using SUMO's *netconvert utility*. SUMO is a traffic simulator provided by the Eclipse foundation [16]. The graph that encompassed all of the nodes in the PeMS District 7 dataset contained approximately 400k nodes and 1M edges, Fig. 3 (B). This is a directed graph meaning physical traffic direction along edges was maintained. This graph was far too large for rapid, iterative, development; hence, maintaining a graph constructed by overlaying the PeMS District 7 nodes on top of the OSM graph was impractical. Rather, this large combined graph was created a single time and Dijkstra's shortest path algorithm [8] was repeatedly called to compute the path between each of the PeMSD7 nodes using the nearest edges of the OSM graph. This reduced size network was comprised of PeMSD7 nodes and edges weighted by the physical travel time required to traverse from one node to another. This road network-informed graph, Fig. 3 (C), was further downsampled to 231 nodes using a similar strategy to further decrease the iteration time required during development.

Five targets were selected from the 231 nodes. Four of these targets were selected along the periphery of the graph with one target near the graph's centroid. Given the large number of edges, greater than 2500, the SI strategy described above was altered such that a node was removed from the graph rather than a single edge (i.e., blocking traffic flow in an intersection). Four nodes with a high degree (relative to other nodes) were selected as the intervention set. Nodes with high degree will likely impact more routes which should yield more information when the agent is forced to re-route.

IV. Metrics and Results

The proposed system is designed to evaluate the probabilities for each target and recommend a COA to the operator. The possible COAs are to keep monitoring, report the suspect to the authorities, or to make a soft intervention. Note that in our experimental scenario, reporting a wrong target location is considered a failure equal to not intercepting the suspect before it reaches its destination.

To make concrete recommendations, two thresholds are required. First, a posterior probability threshold that quantifies the minimum probability estimate for a target before it is reported to the authorities. Second, an entropy threshold is used to decide whether or not we should intervene. Recall, that we associate a cost with making interventions and therefore, we attempt to avoid making them unless the benefit, i.e., the increased confidence in the target, outweighs the cost. At runtime, these thresholds could be tuned to reflect the preferences of the system operators.

To help determine the 'optimum' threshold values as well as evaluate the performance of our system, we use several common metrics as well a custom score function to quantify our preferences:

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F1 = 2.\frac{Precision.Recall}{Precision+Recall} \tag{4}$$

where TP: True Positives – Target determined correctly; FP: False Positives – Target incorrectly determined; TN: True Negatives – There are no true negatives given our assumptions; FN: False Negatives – No recommendation was made and the target reached its destination.

All proposed metrics are bounded between 0 and 1, with 1 corresponding to perfect performance. Intuitively, precision is diminished as the number of false positives increases, i.e., system is more 'sensitive', while recall is degraded if there are more false negatives, i.e., system is less 'sensitive'. The F1-score is the harmonic mean of precision and recall and is often used as a good metric to summarize the overall performance of the system. We also define a custom scoring function to incorporate the cost of intervention and reward the quicker correct determination of the target:

$$Score = \frac{1}{n}.\sum_{i=1}^{n}\left(A.{G_t}^{(i)} - B.I^{(i)} + C.\left({G_t}^{(i)}+1\right)\left(\frac{N^{(i)}-1}{2L^{(i)}}\right)\right) \tag{5}$$

where A, B, and C are constants that capture the operator's preferences defined in Table I, $G_t \in [-1, 1]$ (incorrect/correct determination), $I \in [0, 1]$ (intervened or not), N is the number of nodes suspect has to traverse to reach the target (given the initial starting point), and L is the number of nodes traversed before the target determination was made. This function encodes our preference for fewer interventions by associating a cost, scaled by B. The last term is used to reward the system for faster correct recommendations. Finally, the function is normalized by total number of simulated scenarios n. To estimate the optimum thresholds, we performed an exhaustive grid search over the 2-dimensional parameter space and selected the thresholds yielding optimum performance, Table I. These parameters are then fixed. Note that the custom scores of the two graphs are not comparable as they depend on specific graph properties, e.g., number of nodes.

TABLE I
Optimum Thresholds and Constant Parameters Used in Evaluation of Scenarios in the Vancouver and Los Angeles Graphs

Parameter	Vancouver	Los Angeles
Posterior Threshold	0.575	0.484
Entropy Threshold	1.263	0.227
A	10	10
B	3	3
C	2	2

Fixing the thresholds at their optimum values, $n = 3{,}000$ scenarios were simulated to obtain the following results:

TABLE II
METRICS CALCULATED FOR VANCOUVER AND LOS ANGELES GRAPHS

Metrics	Vancouver		Los Angeles	
	Interventions	No Interventions	Interventions	No Interventions
Precision	0.706	0.700	0.760	0.727
Recall	0.760	0.756	0.794	0.774
F1	0.732	0.727	0.777	0.750
Score	2.71	2.56	5.50	3.59

For both graphs, we see a slight improvement in F1 score if interventions are allowed. This comes at the cost of lower overall score arising from the additional penalty from making interventions.

Note that in the case of Vancouver, due to the very small size of the graph the interventions have a small effect. As expected, we get a larger performance boost in the larger graph where the targets are sufficiently far from the initial starting points to allow more time for decision-making. Also note that the larger *Score* value of the intervened simulations supports the fact that the performance gain due to these interventions did in fact outweigh the costs, as specified by the constants in Table I.

A similar scheme was employed on the larger graph constructed from PeMS District 7 data, Figs. 4 (A) and (B). This larger graph is composed of an order of magnitude more nodes than the toy example; however, the degree of separation between any two nodes does not scale with the absolute number of nodes. This is highlighted in Fig. 4 (A) example where a suspect can traverse half of the graph's geographic expanse, initially starting near Santa Monica, t_0, travelling to the target in Pomona, t_7, in only seven steps. Even with this limited number of suspect acquisitions, the correct target can be determined from the Bayesian approach in four to five time steps, Fig. 4 (C). The performance of the Bayesian trajectory approach far exceeds a naïve approach of only using the distance between the suspect to each target.

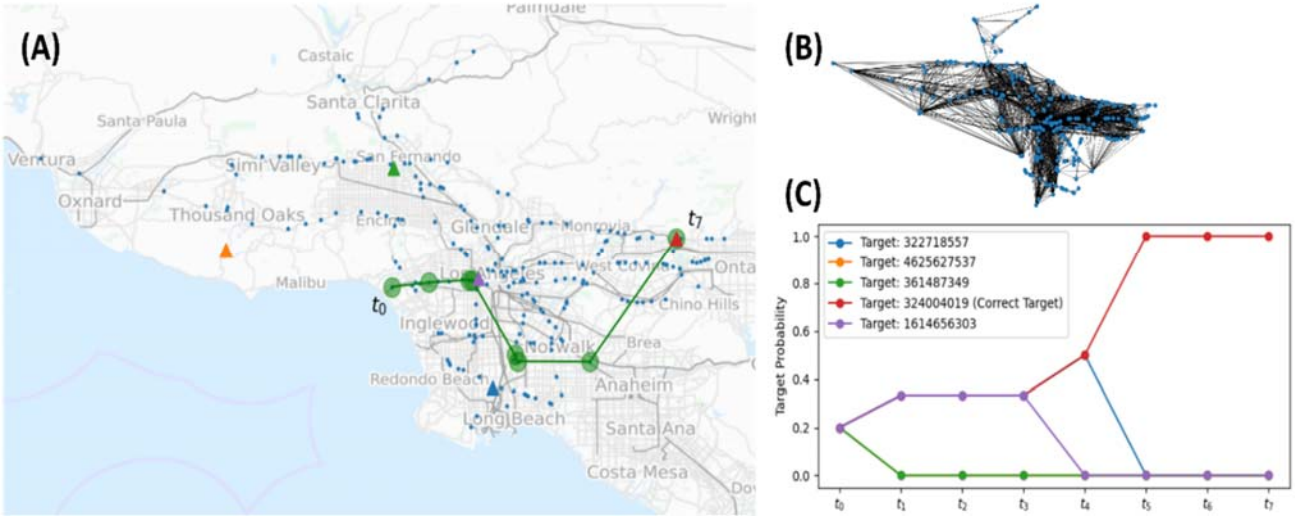

Fig. 4 (A) A Sample trajectory in the LA graph. Five target locations are arbitrarily selected (and marked on the map by triangles). The trajectory is shown in green, (B) Graph data structure for LA map, (C) Time-dependent target probabilities. The target (shown in red) was correctly identified (without any interventions)

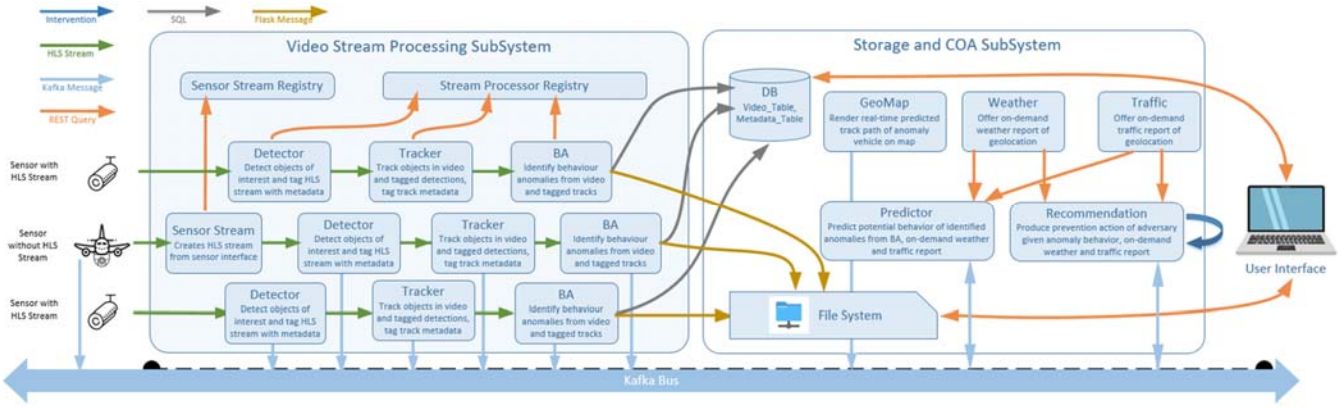

Fig. 5 A high-level schematic of our system architecture to support live streaming, video processing, and decision support

V. SYSTEM ARCHITECTURE

To demonstrate and explore various operational scenarios, we implemented an application that supports both video stream analytics and real-time decision support. We build on the real-time video streaming system we developed in [2] that is capable

of receiving and processing real-time video data streams from traffic or airborne cameras. *Detector* and *Tracker* algorithms are then used to further process the data before serving the results across the system through HTTP Live Streaming (HLS) protocol and Kafka message bus.

The decision-support sub-system is responsible for serving the *Predictor* and *Recommender* algorithms as services. The prediction algorithm is wrapped in the predictor service and takes as input the real-time video for each node, weather conditions, traffic report as well as the operator intervention decisions. The estimated target probabilities as well as a set of recommendations (served by the recommender service) are provided to the operator to make the final decision. We also utilize an integrated storage capability for more efficient aggregation, processing, and access to historical data.

The architecture is designed to be modular, wrapping individual algorithms and features, into a stand-alone service. This results in a decoupled, scalable and maintainable application to support future developments. All services are containerized to enable consistent and reliable services allowing modular deployment as needed. A high-level schematic of our software architecture is shown in Fig. 5.

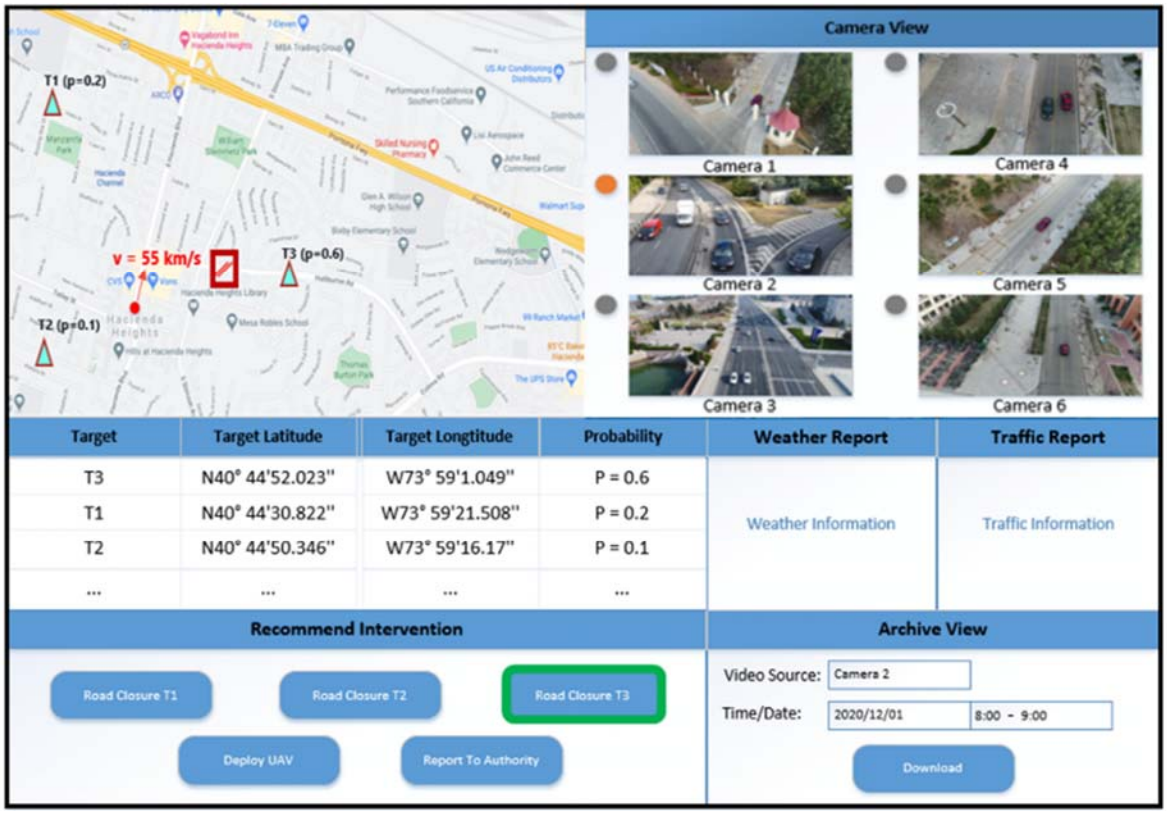

Fig. 6 Mock User Interface displaying the geo-map, multiple live camera feeds, target probabilities, recommendations, as well as access to live weather/traffic information and historical footage

On the landing page, the web interface Fig. 6, shows the nearest six cameras' live streaming videos to the suspect's estimated location. Operators can select specific cameras for manual monitoring. The detector/tracker services run in the background and suspect's location is fed into the *Predictor* and *Recommender* services for further processing. Right below the geo-map view of the interface, the list of predicted targets and their associated probabilities are displayed. Operators see the recommended intervention options proposed by the algorithms. When an operator chooses an intervention from the list, the web interface re-renders and displays updated prediction results and recommendation options, along with corresponding visual effect on the map view, for example, placing a marker to indicate a road closure of the road leading to target T3 in Fig. 6. The real-time weather and traffic conditions are also displayed. The system allows the operators to access (and potentially download) any video footage by specifying the time range and the camera.

The system is designed with incremental implementation in mind. As the algorithms are matured (and new ones are added), our design allows for fast and easy integration and serving of these algorithms.

VI. Conclusion

In this work, we have taken initial steps in integrating deep learning-based perception models capable of detecting and tracking vehicles in FMV with a robust and flexible Bayesian framework. Our preliminary results show promise in delivering automated recommendations to operators for traffic surveillance applications, paving the way for semi-autonomous/autonomous modes of operation in the future. Increasingly, we live in a hyper-connected world, and the

attention of the limited numbers of human operators is scarce and extremely valuable. Systems such as ours will enable more efficient processing and analysis of large and diverse data streams, only prompting the operator for critical decisions.

While we focused on the traffic scenario because of convenience due to availability of public data, our approach is general and could be adapted to other decision-support tools in different domains.

We are currently working on multiple extensions of this work. Reinforcement Learning (e.g., Q-learning) [9] could be used to model the scenario as a Markov Decision Process (MDP), with states corresponding to the current location of the suspect, and the actions are the set of recommendations available. This is one way to directly estimate the likelihood function presented here. Moreover, we are actively working to go beyond some of the simplifying assumptions made in this work, allowing for more sophisticated behavior by the suspect as well as leveraging additional information sources for decision-making, e.g., type of vehicle, speed, etc. Ultimately, we work towards setting up an adversarial training loop, relying on *self-play* [17], training both the *Suspect* and the *Recommender*, further improving the performance of our decision-support tool. It is then possible to close the loop by optimizing the number and the placement of the sensors to ensure maximum performance while balancing the costs.

ACKNOWLEDGMENT

We would like to thank *Defence Research and Development Canada (DRDC)*'s *Innovation for Defence Excellence and Security (IDEaS)* program for supporting this work.

REFERENCES

[1] B. D. Ziebart, A. Maas, J. A. Bagnell, and A. K. Dey, "Maximum Entropy Inverse Reinforcement Learning," in *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence*, 2008, p. 6.

[2] A. J. Macdonald *et al.*, "Unsupervised behaviour anomaly detection from fixed camera full motion video," in *Artificial Intelligence and Machine Learning in Defense Applications II*, Sep. 2020, vol. 11543, p. 115430M, doi: 10.1117/12.2572580.

[3] D. Sivia and J. Skilling, *Data Analysis: A Bayesian Tutorial*. Oxford University Press, 2006.

[4] J. Pearl, *Causality*. Cambridge University Press, 2009.

[5] J. Peters, D. Janzing, and B. Schölkopf, *Elements of Causal Inference: Foundations and Learning Algorithms*. MIT Press, 2017.

[6] "PeMS Data Source | Caltrans." https://dot.ca.gov/programs/traffic-operations/mpr/pems-source (accessed Apr. 11, 2021).

[7] "Dictionary of Algorithms and Data Structures." https://xlinux.nist.gov/dads/ (accessed May 24, 2021).

[8] T. H. Cormen, T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction To Algorithms*. MIT Press, 2001.

[9] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018.

[10] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. John Wiley & Sons, 2012.

[11] C. Chen, "Freeway Performance Measurement System (PeMS)," Jul. 2003, Accessed: Apr. 12, 2021. (Online). Available: https://escholarship.org/uc/item/6j93p90t.

[12] B. Yu, H. Yin, and Z. Zhu, "Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, Stockholm, Sweden, Jul. 2018, pp. 3634–3640, doi: 10.24963/ijcai.2018/505.

[13] C. Snyder and M. N. Do, "STREETS: A novel camera network dataset for traffic flow," in *Advances in Neural Information Processing Systems*, 2019, vol. 32, Accessed: Apr. 12, 2021. (Online). Available: https://experts.illinois.edu/en/publications/streets-a-novel-camera-network-dataset-for-traffic-flow.

[14] L. Bai, L. Yao, C. Li, X. Wang, and C. Wang, "Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting," *arXiv:2007.02842 (cs, stat)*, Oct. 2020, Accessed: Apr. 12, 2021. (Online). Available: http://arxiv.org/abs/2007.02842.

[15] "OpenStreetMap," *OpenStreetMap*. https://www.openstreetmap.org/ (accessed Apr. 12, 2021).

[16] "Eclipse SUMO - Simulation of Urban MObility," *Eclipse SUMO - Simulation of Urban MObility*. https://www.eclipse.org/sumo/ (accessed Apr. 12, 2021).

[17] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust Adversarial Reinforcement Learning," in *International Conference on Machine Learning*, Jul. 2017, pp. 2817–2826, Accessed: Apr. 14, 2021. (Online). Available: http://proceedings.mlr.press/v70/pinto17a.html.