Adapting Complex Event Detection to Perceptual Domain Shift

Brian Wang
U. California, Los Angeles

Julian de Gortari Briseno
U. California, Los Angeles

Liying Han
U. California, Los Angeles

Henry Phillips
Soar Technology, LLC

Jeffrey Craighead
Soar Technology, LLC

Ben Purman
Soar Technology, LLC

Lance Kaplan
DEVCOM ARL

Mani Srivastava
U. California, Los Angeles

Abstract*—Many autonomous systems today, such as robots, drones, and actuated networked sensors, have deployed deep learning models for detecting events from unstructured sensory data. However, the strong performance of these models is restricted to events with short intervals of time and space due to the limited context memory of their architectures. Thus, detecting events that transpire over long periods of time with multiple spatially distant sensor sources (known as *complex events*) remains challenging for these purely neural-based methods, particularly as environmental conditions and object appearances change. In recent years, neurosymbolic approaches have been proposed that use both neural-based perception and symbolic reasoning for capturing complex events. Yet, these approaches still inherit the vulnerability of neural models to perceptual domain shift arising from changing environmental conditions and object appearances, leading to a reduction in the performance of complex event detection. At the same time, standard domain adaptation mechanisms for neural models do not scale well to complex events involving spatiotemporal reasoning over simpler events. We address these problems in the context of a prototype neurosymbolic system called DANCER, which performs Domain Adaptation and Neurosymbolic inference in Complex Event Reasoning. DANCER aims to provide domain adaptation in a post-deployment setting while minimizing runtime user burden for annotation. To enable training and evaluation of DANCER, we also provide a physics-based synthetic sensor data generator to create videos given complex scenario specifications. We evaluate DANCER on a dataset of generated synthetic data. We show that DANCER yields a 48% increase in accuracy of complex event detection using domain adaptation while significantly reducing the annotation time of our synthetic complex events by up to 2.7x, demonstrating DANCER's ability to effectively detect complex events under perceptual domain shift.

I. Introduction

Autonomous systems today are equipped with rich sensing capabilities transmitting high dimensional unstructured data, and powered by an ever-growing list of AI/ML solutions that process and analyze such data on-device. In the future, these systems must be capable of detecting and reacting to various events of interest occurring in their environment. Typical applications in autonomous systems involve spatiotemporal reasoning between events observed by different sensors in order to coordinate actions by robots, actuated sensors, and other autonomous systems.

Examples of such applications include urban scenarios such as detecting street takeovers [1], flash mob robberies [2], suspicious package placements [3], and coordinated terrorism attacks [4]. These scenarios may be captured by traffic cameras, mobile robots, and drones. Other examples include military scenarios with aerial drones for surveillance or rescue missions [5], and detecting complex actions in sporting events using pan-tilt-zoom cameras [6].

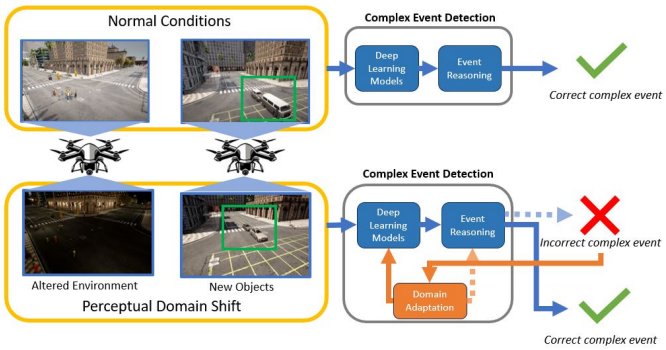

Fig. 1: We propose a complex event detection system that adapts to different cases of perceptual domain shift using high-level complex event labels.

Building the necessary detection pipelines for these autonomous system applications requires reasoning and analysis over events that are distributed across multiple sensors, and involve varying spatial and temporal relationships of diverse objects and the environment. Detecting complex events in these settings has been historically examined via purely symbolic approaches, such as logic programs [7]–[12] or finite state machines [13]. However, these approaches do not adapt well to unstructured data due to the brittleness of logical rules and human-engineered finite state machines. On the other hand, while purely neural approaches are capable of adapting to different types of unstructured data and are well-suited to capturing simpler events (e.g. object detection from video), they struggle with reasoning about multisensor data over longer periods of time. To address this gap, various neurosymbolic architectures [14]–[18] have been proposed in recent years. These architectures are capable of reasoning over *complex events from unstructured data*. However, as the appearances of objects and environmental conditions change over time, both neural and neurosymbolic architectures suffer

from potential perceptual domain shifts (shown in Figure 1), reducing the performance of complex event detection.[1] This necessitates a method of domain adaptation, which fine-tunes deep learning models to the new perceptual domain. However, prior methods on domain adaptation for neural models do not enable learning from high-level complex event labels, while other adaptation approaches in neurosymbolic methods do not consider neural models with high dimensional outputs (such as object detectors).

We introduce *DANCER*, a system for detecting complex events over unstructured data while adapting to *perceptual* domain shift. These events are commonly found in both urban and tactical scenarios with multiple objects comprising each event, spread across multiple non-overlapping cameras with varied vantage points and fields of view. As cameras are increasingly being equipped with onboard GPUs [19], [20], object detection and other functionality can be performed closer to the source of the events. At the same time, it is necessary to aggregate information from multiple individual cameras in order to make sense of complex events that are distributed in space and time.

We provide 3 contributions. Our first contribution is an approach for domain adaptation in complex event scenarios that leverages temporal dependencies between events in a multistage annotation process which iteratively increases user involvement as necessary. This approach aims to minimize user annotation effort post-deployment of the complex event detection system. Our second contribution is a set of tools for simulating complex events across varying weather conditions and object appearances while generating both street-level and aerial video footage. Our third contribution is a language for specifying complex events involving multiple objects and regions of interest across spatially distributed cameras. We have publicly released our code at https://github.com/nesl/DANCER, which includes the associated software program and interfaces for domain adaptation, simulation tools, datasets, language, and the complex event detection system.

II. Related Work

Complex Event Processing Over Structured Data

Several works rely on a SQL-like language in order to match patterns of structured data, such as RFID readings [7], [8]. These languages involve various operations for filtering based on simple predicates, aggregating multiple event streams together, and publishing new types of streams based on previous operations. We focus on capturing events from unstructured data (i.e. video), and aim to overcome the lack of spatial constraints in these languages.

[1]In addition to perceptual domain shift, *symbolic domain shift* may also occur, thus requiring domain adaptation to also update its event reasoning abilities (shown via the dotted orange line in Figure 1). Though we do not consider cases of symbolic domain shift in this work, symbolic domain shifts mainly focus on changes in complex event definitions (e.g. a new variant of a crime may occur, which involves different objects to be observed).

A variety of spatial temporal logic languages have also emerged in the past decade. These languages describe a set of formal rules over systems, which can then be implemented in a programming language to enforce certain constraints, such as safety (i.e. rules must never be violated). A number of spatial logic languages [9]–[12] provide a set of primitives for both time and space, enabling selection, aggregation, and ordering of different complex events. While many of these spatial temporal logic languages introduce interesting spatial constraints, they do not present systems that are capable of matching events over unstructured sensory data and thus are unsuitable for domains involving data such as audio or video.

Complex Event Processing Over Unstructured Data

Several systems combine neural models and symbolic programs (known as neurosymbolic architectures [14]) to process complex events from unstructured sensory data [15]–[17], [21]. All these systems employ a two-stage pipeline where the first stage utilizes neural modules to process raw sensor data to obtain primitive events, and the other stage consists of a symbolic or symbolic-aided reasoner that specifies complex events based on the patterns of primitive events for detection. In particular, [17], [21] have proven to work well in the realm of computer vision. Our work also seeks to perform complex event detection via a neurosymbolic pipeline while also supporting domain shift.

There is also a significant body of work on distributed video analytics for event processing. These works aim to perform lightweight scalable video processing and neural inference in distributed settings [22]–[24], with some including query languages for applying symbolic operators over detected objects [25], or enable question answering over the video stream [26]. However, unlike prior work, we address complex events involving multiple objects and their composition, while also considering the challenge of domain shift of neural modules that have high dimensional output (e.g. object detectors).

Distribution Of Tasks Across Devices

Sensor network programming [27]–[31] provides a method of issuing queries across various sensor networks, where the number of nodes range from 10s to hundreds. Devices are tasked with some sensing objective by a command which is broadcast across all nodes - results of the sensing objective are then aggregated or routed to an origin query node. However, these systems still lack the ability to represent more complex spatial and temporal relations between tasks which are present in spatial temporal logic languages.

Domain Adaptation Under Perceptual Domain Shift

Unsupervised domain adaptation has been shown to be possible for object detection models under different perceptual shifts (e.g. weather, new objects). However, these methods typically require knowledge about perceptual domain shift characteristics ahead of time [32]–[34] which may not be readily available in complex event scenarios. On the other hand, semi-supervised domain adaptation for object detection

has also been studied [35], [36] which aim to minimize user annotation effort by introducing additional neural models to automate parts of the annotation process. However, these works still rely on users to annotate at the level of an image, which is burdensome in complex event scenarios involving multiple video streams over long periods of time. While some prior neurosymbolic approaches consider perceptual domain shift [15] via a differentiable symbolic stage, their neural models focused on classification. In the case of more complex models with high dimensional output, such as object detectors, it becomes far more difficult to propagate information about object locations from high-level complex event labels. Our system provides semi-supervised domain adaptation of neurosymbolic pipelines with object detectors.

III. Background Concepts

Complex Event Hierarchy

When converting unstructured sensor data into high-level complex events, we consider an event hierarchy that evolves low-level data into higher order inferences. This hierarchy is shown in Figure 2. At the lowest level, we generate *object events* from raw video frames. Object events are mainly frame-by-frame object detection results that involve bounding box information and object class. Object events are then aggregated over a short time window in order to derive *vicinal events*. Vicinal events allow us to determine if a unique object has entered a particular area, and assigns unique track IDs to each individual object (thus requiring multiple frames of object detections).

Note that these vicinal events are not specific to cameras, but also applicable to other forms of sensing such as LIDAR or acoustic sensing. Vicinal events are already present in today's systems. For example, cameras operating in a home setting typically already monitor vicinal events, such as Motion Zones in Amazon Ring [37], or known in the military as watchboxes [38]. These systems aim to prioritize events of interest, such as when objects approach one's front door rather than moving on a distant street, so false alarms are not generated.

We use vicinal events to generate a set of *watchbox states* that monitor spatial regions of interest (an example of a watchbox is shown in Figure 2 in the bottom right, highlighted in blue). Watchbox states record multiple vicinal events and object locations. *Atomic events* execute over multiple watchboxes connected by relational operators and logical operators, while *complex events* allow for different temporal constraints, pattern constraints, and logical relationships between atomic events. These events are further described in Section IV.

Domain Adaptation

The problem of *domain shift* or *distribution shift* [39] is well known in machine learning, where models trained on one particular dataset (known as a source dataset) fail to generalize well to a new dataset (known as a target dataset). For instance, a detection model trained on a dataset collected in Los Angeles may not perform as well on images from London, due to changes in weather, scenery, and object appearances (such as new vehicle models).

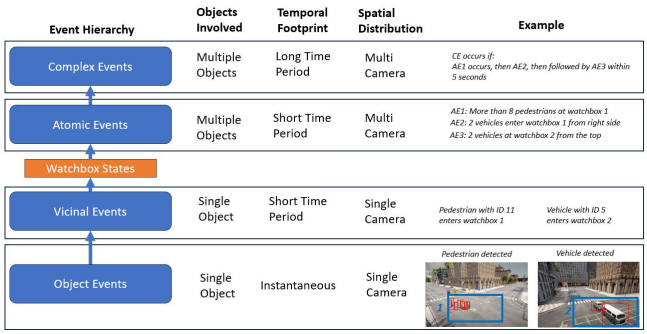

Fig. 2: Different types of events in our event hierarchy.

While a simple approach to handling domain shift requires ground truth annotations of data from the target domain, this is challenging for two reasons, particularly when considering complex event settings. The first is the high cost of manual labor in order to provide annotations. The second is that in many complex event scenarios, the annotations provided are too coarse to meaningfully adapt to domain shift. For example, a complex event label of "pedestrian stealing package" is far too underspecified to fine-tune an object detector with annotated objects - there is simply not enough information about where objects might be located in an image, nor the relationship between them.

IV. System Architecture and Methods

The design of *DANCER* is comprised of an edge computing component, known as the *DANCER client* and the *DANCER coordinator* which involves the simulation of complex events, detection and reasoning, as well as domain adaptation. Figure 3 shows the high level architecture of *DANCER*. The *DANCER* client executes on a GPU-enabled edge device, such as an aerial drone or traffic camera. First, the client performs object detection, which processes each frame of a video, extracts a bounding box for all objects and classifies each one. We utilize YOLOv5s [40] to perform both fast and lightweight object detection capable of running on mobile devices. Once objects are detected, we assign unique IDs to each object in order to track them across video frames. We use a state-of-the-art tracking algorithm known as ByteTrack [41] that also executes on the client. This algorithm takes in bounding boxes of the detected objects and uses both a Kalman filter and intersection-over-union of successive frames to assign unique IDs. Finally, the unique IDs and object locations are used to determine the occurrence of vicinal events (objects that enter/exit a region of interest).

These vicinal events are then transmitted to the *DANCER* coordinator, who uses the events to update watchbox states and perform complex event reasoning. In addition, the coordinator is capable of simulating complex events that are normally difficult to capture in reality, as well as fine-tuning object detection models to perform better under perceptual domain shift. The

next sections describe each of the main functionalities of the *DANCER* coordinator.

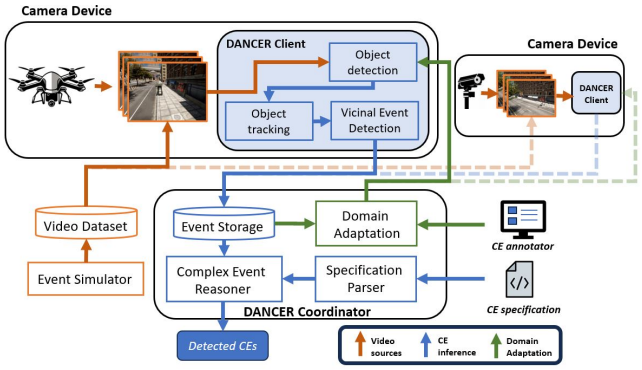

Fig. 3: The different processes present in DANCER. It first takes in multiple different sources of video, produces complex event inferences, and enables fine-tuning of neural predictors

A. CE Specification and Reasoning

Users may specify different atomic and complex events, which are defined in Figure 4. Atomic events (AE) involve multiple watchbox states *wbs*, a predicate *p*, with each event being evaluated at the current time step *i*. Each watchbox maintains a history of tracked objects within a region of interest in a video, with each predicate *p* targeting a particular watchbox and reasoning about the composition of objects (e.g. how many vehicles are present) describing the watchbox state *wbs*. Complex events (CE) are built using multiple operators (op) acting over both atomic events and complex events (evs).

We implement these terms and operators in LanguageCE, [2] where complex events are written and evaluated as Python programs, enabling both parsing of specifications and complex event reasoning given watchbox states. However, in our implementation, only the SEQUENCE/SET operators may be evaluated over both complex and atomic events; the remaining operators only use atomic events.

Event Types	Event value	Event Detection time
$AE := < wbs, p >$	$AE.value(i) = AE.p(wbs, i)$	$Min(i)\ s.t.\ AE.value(i) == True\ and\ AE.value(i-1) == False$
$CE := < op, evs >$	$CE.value(i) = CE.op(evs)$	$Min(i)\ s.t.\ CE.value(i) == True\ and\ CE.value(i-1) == False$

Operators	Result
$CE = AND([AE_1, ..., AE_k], i)$	$AE_1.value(i)\ AND\ AE_2.value(i)\ ...$
$CE = OR([AE_1, ..., AE_k], i)$	$AE_1.value(i)\ OR\ AE_2.value(i)\ ...$
$CE = HOLDS(AE_1, i, L)$	$True\ if\ AE_1.value(j) == True\ for\ all\ j = i - L\ to\ I, False\ otherwise$
$CE = SEQUENCE([AE_1, ..., AE_k], i)$	$True\ if\ AE_1.start < AE_2.start < ... < AE_k.start < i$
$CE = SET([AE_1, ..., AE_k], i)$	$True\ if\ all\ AE_1.start, AE_2.start, ..., AE_k.start < i$
$CE = SEQ\_TIMED([AE_1, ..., AE_k], i, L)$	$True\ if\ AE_1.start < AE_2.start < ... < AE_k.start < i\ and\ for\ all\ j,k\ \|AE_j.start - AE_k.start\| <= L$
$CE = SET([AE_1, ..., AE_k], i, L)$	$True\ if\ all\ AE_1.start, AE_2.start, ..., AEk.start < i\ and\ for\ all\ j,k\ \|AE_j.start - AE_k.start\| <= L$

Fig. 4: Definitions of atomic events (AE), complex events (CE), and various operators

B. Domain Adaptation for Complex Events

In *DANCER*, the key to achieving domain adaptation in complex event settings is done via a suite of cascading interfaces requiring different amounts of annotation effort, and a backend algorithm that aims to minimize such effort.

Algorithm 1 Domain Adaptation

Input: List of vicinal events *LVE*, complex event program *CP*

1: $CVE = find\_critical\_events(LVE, CP)$
2: $to\_annotate = [\,], confirmed = [\,]$
3: **for** v_e in *CVE* **do**
4: $\quad res = user\_verify(v\_e)$
5: $\quad$ **if** $res.val$ **then**
6: $\quad\quad$ add *res.data* to *confirmed*
7: $\quad$ **else**
8: $\quad\quad$ add *res.data* to *to_annotate*
9: $statuses = verify\_ae\_statuses(confirmed, CP)$
10: $missed\_times = get\_t\_intervals(statuses)$
11: **for** *interval* in *missed_times* **do**
12: $\quad v\_e = get\_events(LVE, interval.start, interval.end)$
13: $\quad res = user\_verify(v\_e)$
14: $\quad$ **if** $res.val$ **then**
15: $\quad\quad$ add *res.data* to *confirmed*
16: $\quad$ **else**
17: $\quad\quad$ add *res.data* to *to_annotate*
18: $statuses = verify\_ae\_statuses(confirmed, CP)$
19: $missed\_times = get\_t\_intervals(statuses)$
20: **for** *t_i* in *missed_times* **do**
21: $\quad res = user\_check\_video(t\_i.start, t\_i.end)$
22: $\quad$ add *res.data* to *to_annotate*
23: **for** *curr* in *to_annotate* **do**
24: $\quad labelled = user\_annotate\_image(curr.image)$
25: $\quad$ add *labelled* to *confirmed*
26: $add\_to\_dataset(confirmed)$

Algorithm 1 shows our domain adaptation algorithm. It first determines the set of vicinal events that contributed to an atomic event being detected (line 1). Then, for every vicinal event, we ask the user to verify if a given statement is true or false for an image (lines 3-8). Using the user-confirmed vicinal events, we verify which atomic events have occurred and which were missed (line 9). We then obtain intervals of time where an atomic event could have been missed and ask users to verify vicinal events within that range (lines 10-17). We re-verify which atomic events have occurred, and ask users to annotate potions of a video where relevant vicinal events could have occurred (18-22). Finally, any vicinal events that were incorrectly detected trigger an image labeling process (lines 23-25). We then save all user-confirmed events into our dataset (line 26), which is used to fine-tune the object detection model.

C. Simulation-based Generation of Complex Events Traces

We generate two types of complex events. The first type involves simulating the movement and behaviors of military entities across a map, as captured by multiple surveillance drones. The second type of complex event involves simulating pedestrians, packages, and vehicles across

[2] https://github.com/nesl/LanguageCE

ID	Description	# samples for each case
1	Prepare to Destroy Bridge	38 / 54 / 27
2	Move column of tanks across bridge	41 / 0 / 0
3	Set up stationary defensive position	52 / 0 / 0

TABLE I: Complex event examples for military. There are 3 cases: no domain shift, alternative tank appearances, and smoky visual conditions.

multiple intersections in a city, as captured by multiple traffic cameras. These datasets are publicly available at https://github.com/nesl/ComplexEventDatasets.

1) Complex Event Generation: Military Settings: We have developed a Unity-based [42] tool called NPSim, which defines a library of behaviors for different types of vehicles, including both civilian and military vehicles. Vehicles seek out various waypoints as well as custom maneuvers for evasion. Behaviors may then be stitched together to produce complex events. NPSim introduces two examples of perceptual domain shift. The first example is intermittent wildfire smoke, which partially occludes the visibility of different vehicles (shown in the right of Figure 5. The second example is a variation of enemy entity appearance, which the trained object detection model we use has never seen. In this case, we replace the regular appearance of tanks with a new design, shown on the left of Figure 5. We only simulate these domain shift examples for complex event ID 1. Our dataset and complex events are described in Table I. Each sample involves 3 cameras recording a 10 minute video, resulting in roughly 106 hours of video data.

Fig. 5: Domain shift example for object appearances (left) and environmental conditions (right)

2) Complex Event Generation: Urban Settings: We also develop a set of behaviors on top of Scenic [43], which is a programming language for generating complex scenarios in the CARLA [44] simulator while allowing for probabilistic behavioral policies of different agents (such as pedestrians or vehicles). Variation across the same complex event is achieved by introducing varying object appearances (such as car model or pedestrian clothing), differences in timing for spawning and movement, and finally weather and time of day conditions. In this dataset, we have two types of domain shift. The first is rainy weather conditions, and the second is the same environment set to nighttime. We create a dataset of various complex events, described in Table II. Each sample involves 2-3 cameras and ranges between 30 seconds to 3 minutes, with a total of roughly 30 hours of video data.

ID	Description	# samples for each case
4	Flash robbery	35 / 20 / 20
5	Street takeover	20 / 20 / 19
6	Package theft	20 / 20 / 20
7	Coordinated attack	18 / 17 / 19
8	Hit and Run	20 / 20 / 20

TABLE II: Complex event examples for urban settings. We consider 3 cases: no domain shift (sunny weather), rainy weather, and nighttime

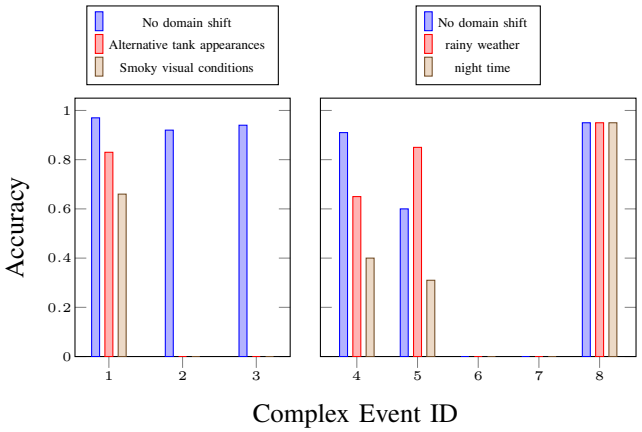

Fig. 6: CE detection accuracy prior to domain adaptation

While all the simulated complex events showcased disjoint camera viewpoints, we consider the possibility of leveraging multiple overlapping viewpoints in future work. This would allow us to fuse predictions across cameras, improving the accuracy of complex event detection.

V. Evaluation

For the *DANCER* client devices, we use Nvidia's Jetson TX2 [45] equipped with 8GB VRAM. These devices have two different object detection models. The first is the default YOLOv5s [40] detection model trained on the COCO dataset [46], while the second is a YOLOv5s detection model fine-tuned on several different classes of military vehicles. When running experiments involving urban scenarios, we utilize the former object detection model, while military scenarios utilize the latter. For every complex event (CE) case, we replay each video on the client device to mimic capturing a real-time camera feed. Our experiments demonstrate that YOLOv5s tracking and vicinal event detection on the edge devices are capable of achieving 12-15fps, which is sufficient for our dataset. Results from the *DANCER* client are then transmitted over WiFi to the CE coordinator device, which is a desktop machine equipped with two Nvidia RTX A5000 GPUs [47] to perform domain adaptation, complex event reasoning, as well as simulation and storage of new complex events.

A. Accuracy of Detected CEs

Figure 6 shows the results in capturing complex events across both military and urban scenarios using *DANCER*. Note that complex IDs 2 and 3 do not have any domain shift data,

	Short Duration CE		Long Duration CE	
	Subsample	NDA	Subsample	NDA
Avg. Time (sec)	96.7	**86.7**	674.3	**244.6**
Avg. # Annotated Images	9.3	3.1	5.6	3.3

TABLE III: User annotation effort across different annotation methods

and thus performance can not be measured. Complex IDs 6 and 7 involve an entirely new object (a package) that is unseen by the object detector during its training, resulting in zero accuracy.

B. Domain Shift

When performing adaptation, we measure the amount of time and number of annotations required using our method, called Neurosymbolic Domain Adapt (***NDA***). We compare NDA against a method that involves subsampling frames from a video and annotating images. We call this method ***subsample***. In the subsample method, we sample 1 video frame every 10 seconds for the military scenarios (complex IDs 1-3) and 5 seconds for urban scenarios (complex IDs 4-8). We use 2 human annotators, each of whom uses both the subsample and our domain adaptation method. Annotators were permitted to skip over images if they had annotated a similar one within the same CE example. We measure the average amount of time taken to annotate a single complex event as well as the average number of images that the user must annotate. We annotate 2 examples from every CE ID and leave the remaining cases for evaluation. The results of the annotation effort are shown in Table III. Urban CE scenarios take less than 3 minutes (which we name Short Duration CE), while military scenarios take 10 minutes (which we name Long Duration CE). Urban scenarios typically experience more dynamic objects, resulting in more images to label in the subsample method. This does not affect our method, as we focus strictly on images related to the CE. Our method is able to reduce the amount of time taken for CE annotations by a factor of 1.1-2.7x while reducing the number of images to label by 1.6-3x. Note that users produce fewer annotated images yet take longer to annotate complex events for long duration CEs of the *subsample* method. This is expected, since users must still spend time *viewing* each subsampled frame before annotation, whereas our method produces a minimal set of frames to view and annotate.

After the annotation process, we retrain our object detection models using the provided annotations from both the subsample method and our domain shift method. We then test the fine-tuned models on an unseen set of complex events and show their performance in Figure 7.

C. Accuracy After Domain Adaptation

Figure 7 shows the performance of different domain adaptation approaches. The colored bars refer to the same domain shift descriptions from the legend in Figure 6. In nearly all cases, our method of domain shift is able to improve the accuracy of complex event detection, particularly in cases

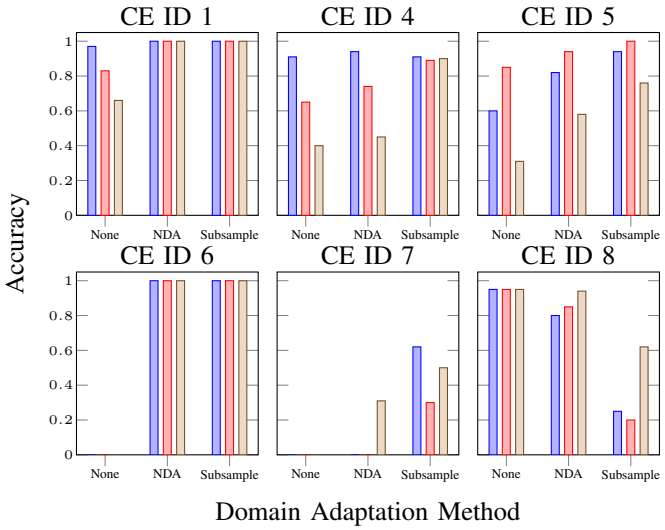

Fig. 7: Accuracy of CE detection without domain adaptation, with the subsample method, and with NDA(ours)

where new objects emerge and there are significant visual differences (e.g. smoke or night). While non-domain adaptation has an average 50.17% accuracy across all complex event and domain shift types, our method yields an average 74.3% accuracy (a 48% increase). However, our method does not always outperform the subsample approach which has the advantage of being able to annotate a significantly larger number of images at the cost of annotation effort. The last case (ID 8) is also an interesting case - both methods fail to outperform the non-domain adaptation case. This is possibly due to catastrophic forgetting [48] in neural networks as some prior information is overwritten.

VI. Conclusion

The DANCER neurosymbolic architecture for complex event detection that we presented is capable of supporting domain adaptation with minimal user annotation effort. Our proposed domain adaptation method decreases annotation time by up to 2.7x compared to a subsampling approach, while simultaneously improving complex event detection accuracy under perceptual domain shift by 48%. Domain shifts are however not limited to perception, and can also arise from concept drifts due to changes in tactics and procedures, which would require fine-tuning the symbolic stage. Our future research will explore this topic besides also generalizing DANCER to other complex event types and sensing modalities.

Acknowledgements

The research in this paper was sponsored in part by the DEVCOM Army Research Laboratory (cooperative agreements W911NF1720196 and W911NF2220243, and STTR contract W911NF22P0064) and the Air Force Office of Scientific Research (award FA95502210193).

REFERENCES

[1] https://www.latimes.com/people/nathan solis and https://www.latimes.com/people/melissa hernandez, "Inside L.A.'s deadly street takeover scene: 'A scene of lawlessness'," Aug. 2022, section: California. [Online]. Available: https://www.latimes.com/california/story/2022-08-22/street-takeovers-sideshow-deadly-toll-los-angeles

[2] A. Bhattarai and G. D. Vynck, "'Flash mob' robberies roiling U.S. retailers, traumatizing workers," *Washington Post*, Dec. 2021. [Online]. Available: https://www.washingtonpost.com/business/2021/12/03/retail-theft-organized-crime/

[3] E. Reed, "APTA SS-SRM-RP-007-12." [Online]. Available: https://www.apta.com/research-technical-resources/standards/security/apta-ss-srm-rp-007-12/

[4] "Planning Considerations: Complex Coordinated Terrorist Attacks."

[5] S. D. Inc, "Military Drone Swarm Intelligence Explained," Sep. 2022. [Online]. Available: https://sdi.ai/blog/military-drone-swarm-intelligence-explained/

[6] "Sports." [Online]. Available: https://ptzoptics.com/solutions/sports/

[7] A. Demers, J. Gehrke, B. Panda, M. Riedewald, V. Sharma, and W. White, "Cayuga: A General Purpose Event Monitoring System."

[8] D. Gyllstrom, E. Wu, H.-J. Chae, Y. Diao, P. Stahlberg, and G. Anderson, "SASE: Complex Event Processing over Streams," Dec. 2006, arXiv:cs/0612128. [Online]. Available: http://arxiv.org/abs/cs/0612128

[9] "SpaTeL | Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control." [Online]. Available: https://dl.acm.org/doi/abs/10.1145/2728606.2728633

[10] L. Nenzi, L. Bortolussi, V. Ciancia, M. Loreti, and M. Massink, "Qualitative and Quantitative Monitoring of Spatio-Temporal Properties with SSTL," *Logical Methods in Computer Science*, vol. Volume 14, Issue 4, Oct. 2018, publisher: Episciences.org. [Online]. Available: https://lmcs.episciences.org/4913/pdf

[11] "Monitoring mobile and spatially distributed cyber-physical systems | Proceedings of the 15th ACM-IEEE International Conference on Formal Methods and Models for System Design." [Online]. Available: https://dl.acm.org/doi/abs/10.1145/3127041.3127050

[12] M. Ma, E. Bartocci, E. Lifland, J. Stankovic, and L. Feng, "SaSTL: Spatial Aggregation Signal Temporal Logic for Runtime Monitoring in Smart Cities," in *2020 ACM/IEEE 11th International Conference on Cyber-Physical Systems (ICCPS)*, Apr. 2020, pp. 51–62, iSSN: 2642-9500.

[13] "Apache Flink® — Stateful Computations over Data Streams." [Online]. Available: https://flink.apache.org/

[14] S. Chaudhuri, K. Ellis, O. Polozov, R. Singh, A. Solar-Lezama, and Y. Yue, "Neurosymbolic Programming," *Foundations and Trends® in Programming Languages*, vol. 7, no. 3, pp. 158–243, Dec. 2021, publisher: Now Publishers, Inc. [Online]. Available: https://www.nowpublishers.com/article/Details/PGL-049

[15] T. Xing, L. Garcia, M. R. Vilamala, F. Cerutti, L. Kaplan, A. Preece, and M. Srivastava, "Neuroplex: learning to detect complex events in sensor networks through knowledge injection," in *Proceedings of the 18th conference on embedded networked sensor systems*, 2020, pp. 489–502.

[16] T. Xing, M. R. Vilamala, L. Garcia, F. Cerutti, L. Kaplan, A. Preece, and M. Srivastava, "Deepcep: Deep complex event processing using distributed multimodal information," in *2019 IEEE international conference on smart computing (SMARTCOMP)*. IEEE, 2019, pp. 87–92.

[17] M. R. Vilamala, T. Xing, H. Taylor, L. Garcia, M. Srivastava, L. Kaplan, A. Preece, A. Kimmig, and F. Cerutti, "Deepprobcep: A neuro-symbolic approach for complex event processing in adversarial settings," *Expert Systems with Applications*, vol. 215, p. 119376, 2023.

[18] G. Apriceno, A. Passerini, and L. Serafini, "A neuro-symbolic approach for real-world event recognition from weak supervision," in *29th International Symposium on Temporal Representation and Reasoning (TIME 2022)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2022.

[19] "Metropolis Spotlight: Sighthound Enhances Traffic Safety with NVIDIA GPU-Accelerated AI Technologies," Oct. 2021. [Online]. Available: https://developer.nvidia.com/blog/metropolis-spotlight-sighthound-enhances-traffic-safety-with-nvidia-gpu-accelerated-ai-technologies/

[20] D. Scimeca, "Repurposed traffic cameras anonymously track social distancing during COVID outbreak," Dec. 2020. [Online]. Available: https://www.vision-systems.com/boards-software/article/14188528/traffic-cameras-track-social-distancing-vivacity-labs

[21] X. Liu, P. Ghosh, O. Ulutan, B. Manjunath, K. Chan, and R. Govindan, "Caesar: cross-camera complex activity recognition," in *Proceedings of the 17th Conference on Embedded Networked Sensor Systems*, 2019, pp. 232–244.

[22] Y. Li, A. Padmanabhan, P. Zhao, Y. Wang, G. H. Xu, and R. Netravali, "Reducto: On-camera filtering for resource-efficient real-time video analytics," in *Proceedings of the Annual conference of the ACM Special Interest Group on Data Communication on the applications, technologies, architectures, and protocols for computer communication*, 2020, pp. 359–376.

[23] X. Zeng, B. Fang, H. Shen, and M. Zhang, "Distream: scaling live video analytics with workload-adaptive distributed edge intelligence," in *Proceedings of the 18th Conference on Embedded Networked Sensor Systems*, 2020, pp. 409–421.

[24] J. Jiang, G. Ananthanarayanan, P. Bodik, S. Sen, and I. Stoica, "Chameleon: scalable adaptation of video analytics," in *Proceedings of the 2018 conference of the ACM special interest group on data communication*, 2018, pp. 253–266.

[25] D. Kang, P. Bailis, and M. Zaharia, "Blazeit: optimizing declarative aggregation and limit queries for neural network-based video analytics," *arXiv preprint arXiv:1805.01046*, 2018.

[26] T. Sautory, N. Cingillioglu, and A. Russo, "Hyster: A hybrid spatio-temporal event reasoner," *arXiv preprint arXiv:2101.06644*, 2021.

[27] A. Boulis, C.-C. Han, R. Shea, and M. B. Srivastava, "SensorWare: Programming sensor networks beyond code update and querying," *Pervasive and Mobile Computing*, vol. 3, no. 4, pp. 386–412, Aug. 2007. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1574119207000314

[28] T. Liu and M. Martonosi, "Impala: A middleware system for managing autonomic, parallel sensor systems," in *Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming*, 2003, pp. 107–118.

[29] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: an acquisitional query processing system for sensor networks," *ACM Transactions on database systems (TODS)*, vol. 30, no. 1, pp. 122–173, 2005.

[30] C.-L. Fok, G.-C. Roman, and C. Lu, "Agilla: A mobile agent middleware for self-adaptive wireless sensor networks," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 4, no. 3, pp. 1–26, 2009.

[31] J. Noor, H.-Y. Tseng, L. Garcia, and M. Srivastava, "DDFlow: visualized declarative programming for heterogeneous IoT networks," in *Proceedings of the International Conference on Internet of Things Design and Implementation*, ser. IoTDI '19. New York, NY, USA: Association for Computing Machinery, Apr. 2019, pp. 172–177. [Online]. Available: https://dl.acm.org/doi/10.1145/3302505.3310079

[32] W. Liu, G. Ren, R. Yu, S. Guo, J. Zhu, and L. Zhang, "Image-adaptive yolo for object detection in adverse weather conditions," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, 2022, pp. 1792–1800.

[33] X. Li, W. Chen, D. Xie, S. Yang, P. Yuan, S. Pu, and Y. Zhuang, "A free lunch for unsupervised domain adaptive object detection without source data," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 10, 2021, pp. 8474–8481.

[34] M. Khodabandeh, A. Vahdat, M. Ranjbar, and W. G. Macready, "A robust learning approach to domain adaptive object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 480–490.

[35] X. Yang, X. Liu, M. Jian, X. Gao, and M. Wang, "Weakly-supervised video object grounding by exploring spatio-temporal contexts," in *Proceedings of the 28th ACM international conference on multimedia*, 2020, pp. 1939–1947.

[36] H. Li, X. Pan, K. Yan, F. Tang, and W.-S. Zheng, "Siod: Single instance annotated per category per image for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 197–14 206.

[37] "Utilizing Motion Zones With Your Video Doorbells and Security Cameras." [Online]. Available: https://support.ring.com/hc/en-us/articles/360021842611-Utilizing-Motion-Zones-With-Your-Video-Doorbells-and-Security-Cameras

[38] "Streaming Motion Imagery Alternatives | SBIR.gov." [Online]. Available: https://www.sbir.gov/node/1482209

[39] P. Oza, V. A. Sindagi, V. V. Sharmini, and V. M. Patel, "Unsupervised domain adaptation of object detectors: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.
[40] G. Jocher, "YOLOv5 by Ultralytics," May 2020. [Online]. Available: https://github.com/ultralytics/yolov5
[41] Y. Zhang, P. Sun, Y. Jiang, D. Yu, F. Weng, Z. Yuan, P. Luo, W. Liu, and X. Wang, "Bytetrack: Multi-object tracking by associating every detection box," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXII*. Springer, 2022, pp. 1–21.
[42] "Unity Real-Time Development Platform | 3D, 2D, VR & AR Engine." [Online]. Available: https://unity.com
[43] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: a language for scenario specification and scene generation," in *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, 2019, pp. 63–78.
[44] C. Team, "Carla." [Online]. Available: https://carla.org/
[45] "Jetson TX2 Module," May 2017. [Online]. Available: https://developer.nvidia.com/embedded/jetson-tx2
[46] "COCO - Common Objects in Context." [Online]. Available: https://cocodataset.org
[47] "Introducing RTX A5000 Graphics Card | NVIDIA." [Online]. Available: https://www.nvidia.com/en-us/design-visualization/rtx-a5000/
[48] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.