Conditionally Augmented Temporal Anomaly Reasoner And Convolutional Tracking System (CATARACTS)

Dwight Temple
ExoAnalytic Solutions

Abstract

Applications of artificial intelligence have been gaining extraordinary traction in recent years across innumerable domains. These novel approaches and technological leaps permit leveraging profound quantities of data in a manner from which to elucidate and ease the modeling of arduous physical phenomena. ExoAnalytic collects over 500,000 resident space object images nightly with an arsenal of over 300 autonomous sensors; extending the autonomy of collection to data curation, anomaly detection, and notification is of paramount importance if elusive events are desired to be captured and classified. Efforts begin with rigorous image annotation of observed glints, streaking stars, and resident space objects; synthetic plumes were generated from both Generative Adversarial Networks as well as manual image augmentation techniques. Preliminary results permitted the successful classification of observed debris generating events from AMC-9, Telkom-1, and Intelsat-29e. After initial proof-of-concept, these events are incorporated into the training pipeline in order to characterize potentially unknown debris generating or anomalous events in future observations. The inclusion of a visual tracking system aides in reducing false alarms by roughly 30%. Future efforts include applications on both historical datamining as well as real-time indications and warnings for satellite analysts in their daily operations while maintaining a low probability of false alarm through detection and tracking algorithm refinement.

1 INTRODUCTION

An abundance of papers, proposals, and presentations regarding deep learning have inundated the recent literature of numerous conferences. Typically, these papers introduce novel methods to simplify arduous tasks with no closed-form mathematical solution. Convolutional, Temporal Convolutional and Recurrent Neural Networks (CNNs, TCNs and RNNs)[1], Generative Adversarial Networks (GANs)[2], Reinforcement Learning (RL)[3], and attention-based decoder algorithms have been developed to solve tasks from image classification and segmentation, maneuvering target tracking, language translation and prediction, speech synthesis and emulation, as well as robotic action emulation. In the realm of space situational awareness (SSA), previous work has been completed for sensor tasking using RL [4], maneuver detection using basic CNNs [5] and anomaly emulation and detection with static-pattern GANs and CNNs [6]. ExoAnalytic Solutions collects over 500,000 resident space object images nightly using their arsenal of over 300 ground based autonomous telescopes. With this enormous onslaught of incoming imagery and information, dissemination and discrimination are of paramount importance to prevent operator information overload. Creating a self-perpetuating anomaly detection algorithm improves timeliness, detectability, and unexploited anomalous resident space object (RSO) behavior. As opposed to relying on a static image classification system that is operator supervised and disseminated, ExoAnalytic utilizes a semi-supervised training architecture composed of region proposals for image segmentation and anomaly detection. This paper incorporates additional stringency to the anomaly detection task. Instead of characterizing the unresolved satellite image as a whole, a region proposal neural network explicitly labels regions in the image with their proposed classification and confidence. For example, close-approaches, debris shedding events, and star-streaking can all be annotated with a probability and pinpointed on the image itself. This method, when utilized iteratively, allows for the incorporation of temporal context into decision-chains. This method produces an anomaly detection algorithm less susceptible to false alarms as valuable contextual information is inferred from integration through time using the proposed regions from the neural network.

2 BACKGROUND

CNNs have inundated the deep learning literature in the past decade where they have successfully and continually beaten all previously held image classification benchmarks. Additionally, these networks have been honed using various elaborate methods such as Neural Architecture Search [7] ever increasingly complex network layers, and novel feature extraction techniques. Typically, CNNs targeted performance metrics are the correct classification of imagery with the goal being to categorize, label, or highlight existing components within datasets. Labeling MRIs as anomalous and reporting to a physician, classifying the breed of dog in an image, and identifying the existence of dangerous pests in beehives are all practical applications of CNNs in industry.

While labeling images in their entirety is useful for extracting information, identifying specific regions of pixels within the image proves more useful in intelligence applications. Instead of having the operator inspect flagged images for the occurrence of the class, this portion of labor will instead be completed by the neural network. This application was first introduced by [8] and has since been dramatically improved. Various renditions have been implemented and deployed to an array of problems from self-driving vehicles, pedestrian tracking, and less critical ones such as identifying the Millennium Falcon.

3 CLASSIFICATION ALGORITHM

3.1 ANNOTATION

The primary task of initiating the algorithm development framework is the proper construction of a supervised learning task. Because the goal is to explicitly and simultaneously propose, classify, and regress bounding box labels to objects of interest within an image, appropriate ground-truth labels are necessary. To label a region in an image, five variables are required: object class, box centroid in X and Y, and the box dimension in X and Y, in pixels.

$$Label = (int(object\ class), x_{centroid}, y_{centroid}, x_{width}, y_{width})$$

Additionally, where multiple objects occupy a single image, multiple labels are required. The result is an $N_{labels} * label_{width}$ text file for each image in the dataset.

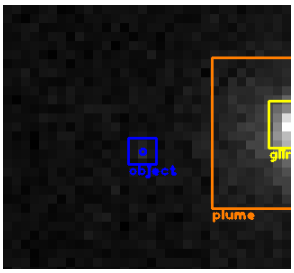

Figure 1. Image used for Annotation Table

Table 1. Example Image Annotation

Class	X Coord.	Y Coord.	X Width	Y Width
0	0.868320	0.443972	0.280415	0.522862
1	0.484716	0.511466	0.078608	0.090849
2	0.965095	0.414408	0.082109	0.164061

The detections in Table 1 are coordinates normalized with respect to the image in Fig. 1. Accordingly, the detection algorithm is capable of operating on various-sized input images given the standardized label format.

The dataset used in subsequent experiments began with 1,000 hand-selected and annotated images. This process proves to be arduous in both time and discernment of specific classifications. For example, when labeling different regions, one must consistently annotate similar ambiguous patterns and occurrences throughout the dataset. This becomes problematic when deciding to label a smudged RSO as a "streak" or as two closely space objects (CSOs). Examples of this can be seen in Fig. 2 and Fig. 3.

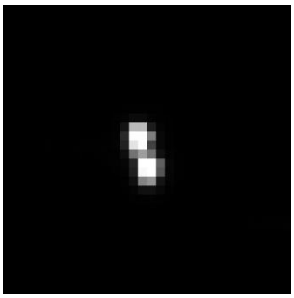

Figure 2. CSO

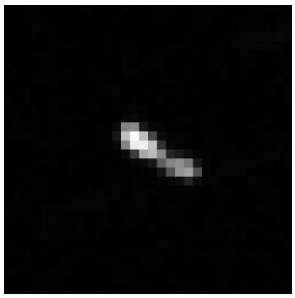

Figure 3. Smudged RSO

3.2 MODEL CONFIGURATION

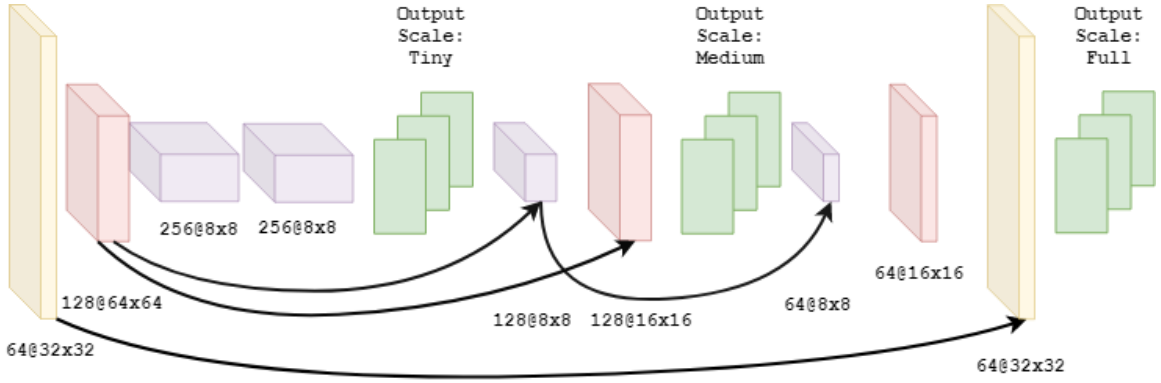

Figure 4. Multiscale Architecture for Small Objects

For classification, the standard CNN archetype is leveraged and utilized as shown in Fig. 4. Specifically, the YoloV3 [8] architecture is used as a fundamental backbone for object segmentation. Additionally, the Darknet [9] neural network library is used for all training and inference. Indeed, architectural best-practices are abided by in order to maximize the downstream classification performance by metrics of both accuracy and timeliness through the reduction in floating point operations. Primarily, the network utilizes residual connections, batch normalization, and smaller convolutional filters to achieve the aforementioned outcomes.

Residual connections, also known as identity connections [10], are the operation of mapping the layer input through a nonlinearity while adding the original input to the nonlinearity output. This operation increases the flow of unhindered imagery information to deeper filter layers while also increasing the gradient availability for successful backpropagation.

Batch normalization is a statistical technique employed in most DNN applications that are non-recurrent [11]. Z-normalizing one's dataset is a beneficial preprocessing tasks for reducing the variance in network weights and biases. I.E, learning weights with smaller variances also for faster model convergence and more generalization. However, once the data is input into the model, as each additional nonlinear operation occurs, the data is no longer consistently z-normalized. Therefore, after each major set of nonlinearities, the dataset can be z-normalized again so that subsequent layers are working with whitened data. Because the data is only passed in a minibatch at a time during training, the batch normalization mean and standard deviation must be trainable network parameters where the proper convergence value is the training *population's* mean and standard deviation at that specific layer in the network.

The input is a dimension of M by M pixels where M must be a multiple of 32. For our application, images used for classification are 32 pixels. As depicted in Fig. 4, convolutions with increasing depth are succeeded by down sampling and batch-normalization layers until the image dimensionality is M by V. V is the necessary output dimensionality defined by the number of classes desired in the output.

3.3 OBJECTIVE FUNCTION

$$\textbf{Loss} = \lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] +$$

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left[\left(\sqrt{w_i} - \sqrt{\widehat{w_i}} \right)^2 + \left(\sqrt{h_i} - \sqrt{\widehat{h_i}} \right)^2 \right] +$$

$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{obj} \left(C_i - \widehat{C_i} \right)^2 +$$

$$\lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} 1_{ij}^{noobj} \left(C_i - \widehat{C_i} \right)^2 +$$

$$\lambda_{noobj} \sum_{i=0}^{S^2} 1_{i}^{obj} \sum_{c \, \epsilon \, classes}^{B} \left(p_i(c) - \widehat{p_i(c)} \right)^2$$

Equation 2

Equation 2 depicts the multifaceted objective function used for optimizing the image classification, localization, and confidence projection. Because each of these loss components can be explicitly untangled, they will be explored and explained individually for further clarification.

3.4 CLASSIFICATION

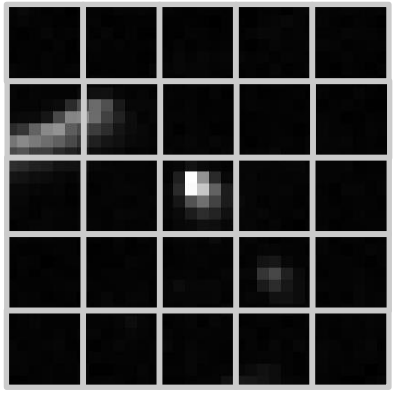

Figure 5. Initial Grid Mask

While most architectures are required to classify the entire image as belonging exclusively to a single class, the task at hand requires multiple classes for numerous regions within the image. This is accomplished by subdividing the image into a gridded mask like Fig. 5 and assigning classification probabilities to each of these regions like Fig. 6.
While this single gridded mask *could* be sufficient for the task, using more proves more granular object segmented image is produced.

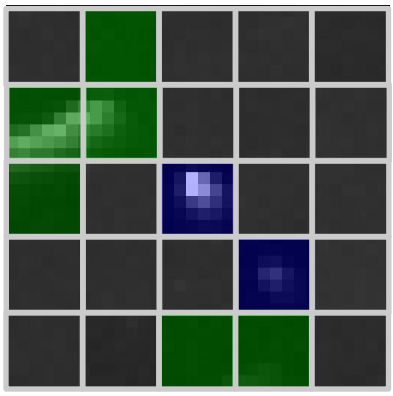

Figure 6. Grid Class Likelihoods

The output layers are scaled and reconfigured in such a manner that this gridded mask is applied at a multi-scale level; this allows for multiple network interpretations in a single pass. This technique can greatly impact the network's ability to reason with objects that appear at various scales throughout training and inference. More specifically, Equation 2 portrays the cost of image classification. If an object exists within a predicted bounding box, the sum squared difference in probabilities is weighted by how much the bounding box overlaps the true target object.

3.5 LOCALIZATION

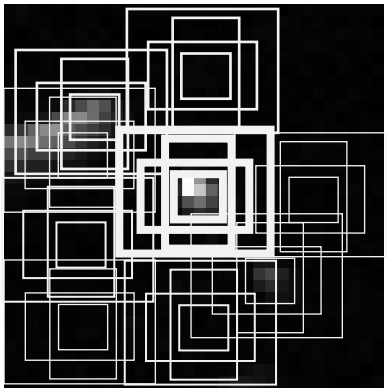

Figure 7. Proposed Bounding Boxes

Once the probability regions are proposed, the next task is to assign bounding boxes to these specific regions. The output of these bounding box proposals is a seemingly chaotic, but well-suited overlay of boxes. This layer is scored on how well the predicted boxes overlay the annotated truth boxes. This score is called "objectness" and is expressed by Equations lines 3 and 4. The former sums the square of the difference in predicted confidence scores for each box where an object truly lies over the entire image. The latter performs the same but over the background portions where no objects lie. It is additionally weighted to assure the model learns the proper interpretation of regions in the image that are not classes, but the weight is small so that this portion of loss does not dominant the landscape detrimentally.

As Fig. 7 depicts, there are now numerous boxes that contain the desired information; it must be dealt with in the proper manner such that one obtains the desired and best-fitting result with maximal accuracy and human interpretability.

3.6 NON-MAXIMAL SUPPRESSION

Fig. 8 displays the predicted bounding boxes and a subset of the class probabilities coded by color. The center region contains two highly overlapped regions predicting the same object class with difference confidences. Suppressing and combining these redundant portions is necessary in order to reduce the interpretability burden on the analyst. Overlapping regions with an intersection over union (IoU) greater than a user-defined threshold, typically 0.5, are sorted descendingly by the computed objectness score. IoU is calculated as the area of intersection of the predicted and truth bounding boxes divided by the unionized area of the predicted and truth bounding boxes as shown in Fig. 9.

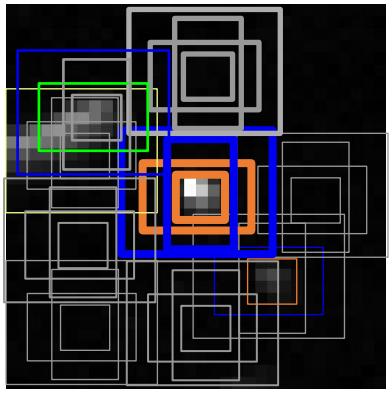

Figure 8. Predicted Bounding Boxes and Classes

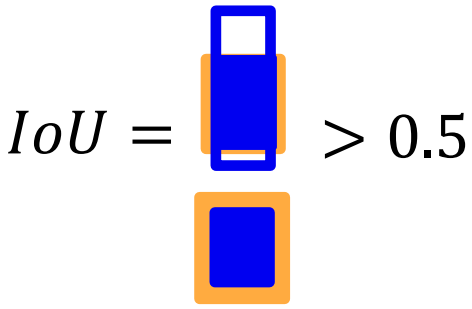

Figure 9. Example Intersection over Union

This process of sorted and selecting the highest probability with the highest IoU is iterated until all detected objects have been accounted for.

The result of this non-max-suppression can be observed in Fig. 10. This process is very algorithmic and rigid; however, it is a user-adjustable portion of the algorithm. Because this threshold is user-adjustable, it can be adapted to problems where either precision or recall is desired more. Precision represents the model's ability to only classify desired classes and not allow false alarms. Recall represents the model's ability to classify as many of the desired class as possible while permitting false alarms. These metrics are shown below in Equations 3 and 4.

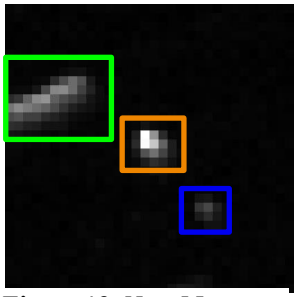

Figure 10. Non-Max-Suppression Results

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \qquad \text{Equation 3}$$

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \qquad \text{Equation 4}$$

Fig. 11 depicts the parameterization of this threshold from low to high from the upper left corner to the bottom right corner. Specifically, it ranges from 0.01 to 0.95 in roughly 0.1 confidence value increments

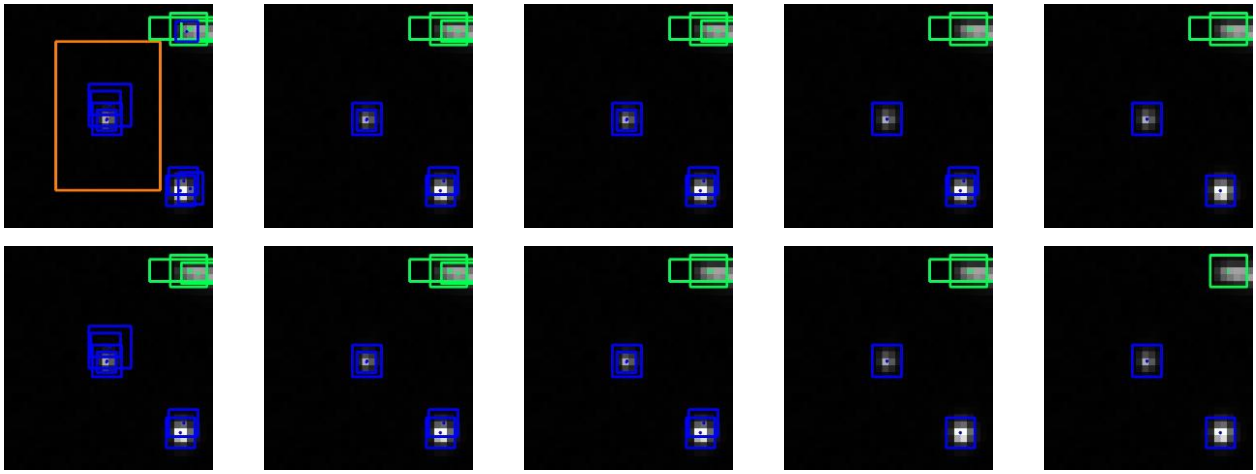

Figure 11. Altering Suppression Threshold

Clearly, the leftmost results host far too many false alarms for a deployed system; however, the bottom right corner, in this image, seems to do exactly as desired. It encapsulates the necessary information to alert an operator of two objects and a likely star in the current image. Now that the algorithm has been discussed regarding its application to properly annotating, regressing, and suppressing for optimal performance, its application and ability to detect real-world events is explored in subsequent sections.

4 TRACKING ALGORITHM

The detection algorithm can be interpreted as a measurement device of a scene; detections and features are extracted on a frame-by-frame basis where the frame's boresight is focused on the primary RSO. Algorithm measurements are represented by (x, y) pixel coordinates, the width and height of the corresponding bounding-box, as well as the objectness score. Utilizing the Simple Online and Realtime Tracking (SORT) algorithm from [5], these observations are mapped to an appropriate temporal state-space model in order to perform multi-target tracking and association. The state-space is represented by pixel coordinates, bounding box aspect ratio and height as well as their respective velocities, $(x, y, a, h, \dot{x}, \dot{y}, \dot{a}, \dot{h})$. The Kalman Filtering scheme used for track prediction and updating is a simple linear model with static uncertainties in object position, velocity, and bounding box dimension. The state-transition and uncertainty are depicted in Equations 5 and 6 where $(\sigma_x, \sigma_y, \sigma_a, \sigma_h, \dot{\sigma_x}, \dot{\sigma_y}, \dot{\sigma_a}, \dot{\sigma_h})$ are $\left(\frac{1}{160}, \frac{1}{160}, 1e-2, \frac{1}{160}, \frac{1}{240}, \frac{1}{240}, 1e-5, \frac{1}{240}\right) * box_{height}$, respectively.

Therefore, the larger a target's bounding box is, the larger the transition uncertainty is from frame to frame.

$$A = \begin{matrix} 1 & 0 & 0 & 0 & dt & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & dt & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & dt & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & dt \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}, \quad Q = \begin{matrix} \sigma_x & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma_y & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_a & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_h & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dot{\sigma_x} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \dot{\sigma_y} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \dot{\sigma_a} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \dot{\sigma_h} \end{matrix} \quad \text{Equation 5 and 6}$$

SORT utilizes a cascading Hungarian assignment algorithm for the multi-target matching. This approach minimizes the Mahalanobis distance between assignable tracks and accounts for measurement frequency in these assignments. The latter is a notably important feature because less frequently observed tracks are more easily associable to incoming confirmed tracks due to the spread of probability mass over the state-space.

In addition to the cascading Mahalanobis metric, SORT can leverage the latent features from a CNN. For example, the final outputs may be pixel coordinates, but the layers just prior to this output can serve as meta-information from

which to associate tracks on a deeper level. The current tracking implementation does not utilize this capability, but it will be explored seriously in future works.

While the original implementation of SORT was used primarily for pedestrian tracking, this implementation is observing another set of processes entirely. Target occlusion and spurious detections still arise; however, the most important differentiator is the multi-class tracking performance. Plumes, glints, and objects are of high interest; likewise, it is crucial to avoid false alarms when deciding whether to report warnings to the operator. Through incorporating a more rigorous track formation framework, these false alarms can be reduced significantly while still maintaining the ability of detecting these events. Fig. 12 and Fig. 13 depicts a sequence of frames with and without the tracking methodology applied to the sequence. The results are a smoother frame-to-frame bounding box scaling as well as less spurious object detections. Within the six sample frames, there is a 37.5% *decrease* in spurious tracks formed as a result of the tracker. Over the course of a night, this reduction aids significantly in reducing false alarms.

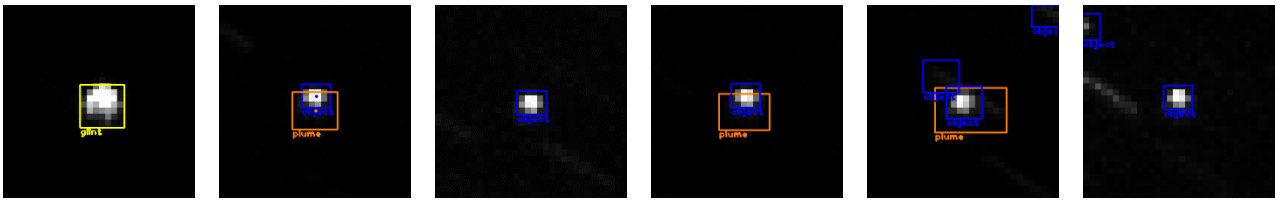

Figure 12. Intelsat-29e without Tracker

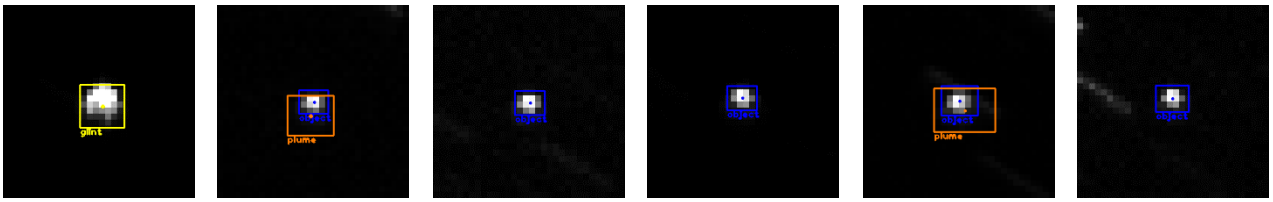

Figure 13. Intelsat-29e with Tracker

5 ANALYSIS OF RESULTS

The results presented below come with an associated and important caveat; the network was *never* trained on imagery collected from these events or any other debris shedding events. The network was exclusively trained using nominal operation patterns alongside synthetically generated imagery to include debris-like clouds and CSOs.

5.1 TELKOM-1

August 25th, 2018 marked the occurrence of a dramatic antenna anomaly for geostationary satellite Telkom-1. The timeline of interest alongside the multivariate metrics captured using the chip segmentation algorithm are portrayed in Fig. 14.

As Fig. 14 displays, there were indications of debris prior to the dramatic material ejection observed at roughly time 48 minutes. Subsequent to the ejection, a field of debris fills the surrounding area with a detectable cloud. Both the ejection and emitted cloud are detectable without prior knowledge. Additionally, a piece of dim and tumbling debris was tracked traveling vertically in the frame as indicated by the "multi-object" time series.

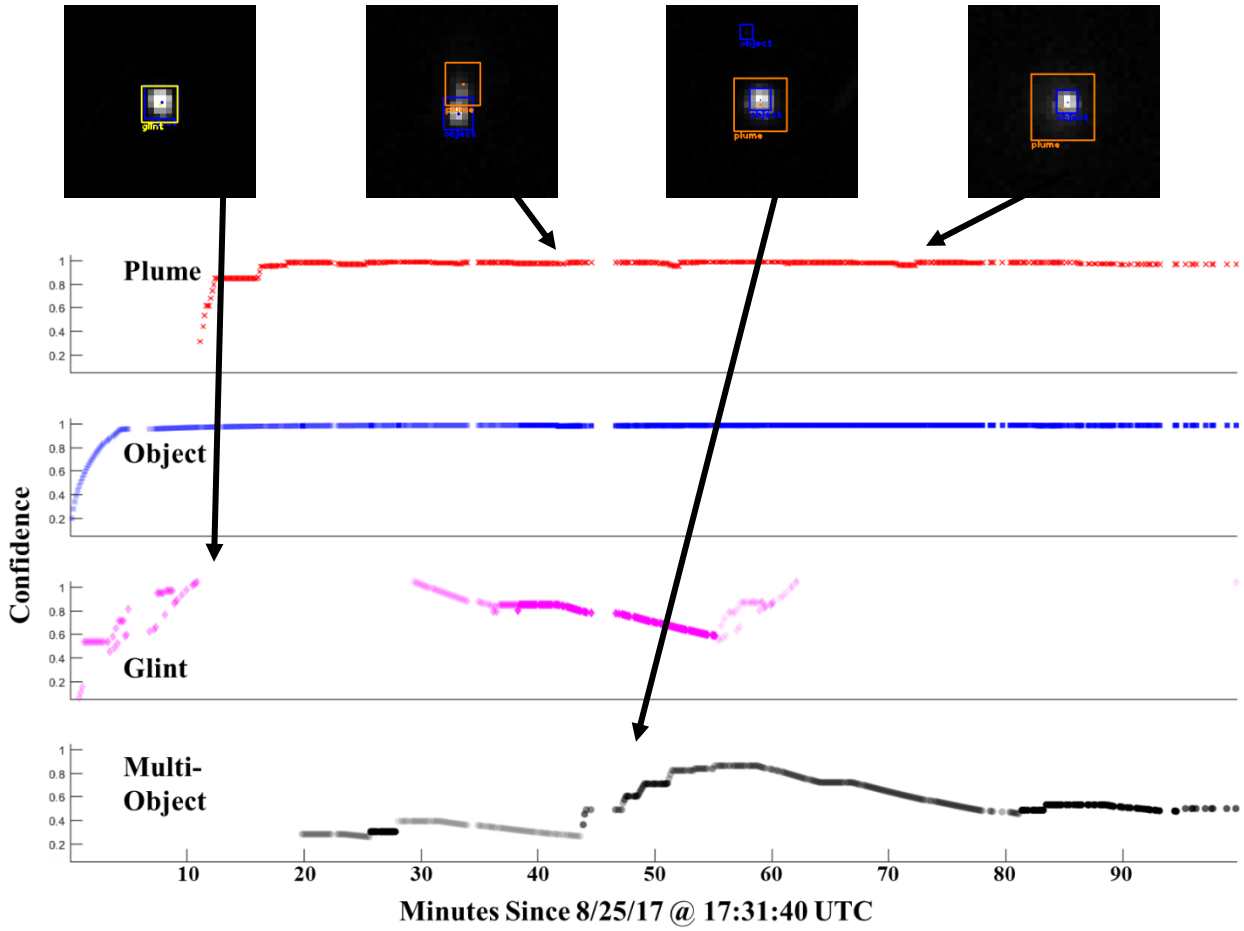

Figure 14. Telkom-1 Timeline of Events

5.2 AMC-9

The first photometrically documented debris generating event captured is that of AMC-9 on July 1st, 2017. Immediately preceding the event were a series of glints; these glints are uncharacteristically bright and are not a result of seasonal and anticipated glinting due to the solar declination angle. CATARACTS detects these glints and subsequently detects a plume track at 280 minutes since June 30th, 2017 at 23:28:25 UTC. Subsequent to this, there are numerous other glints and a detectable and trackable object is tracked from the emitted plume.

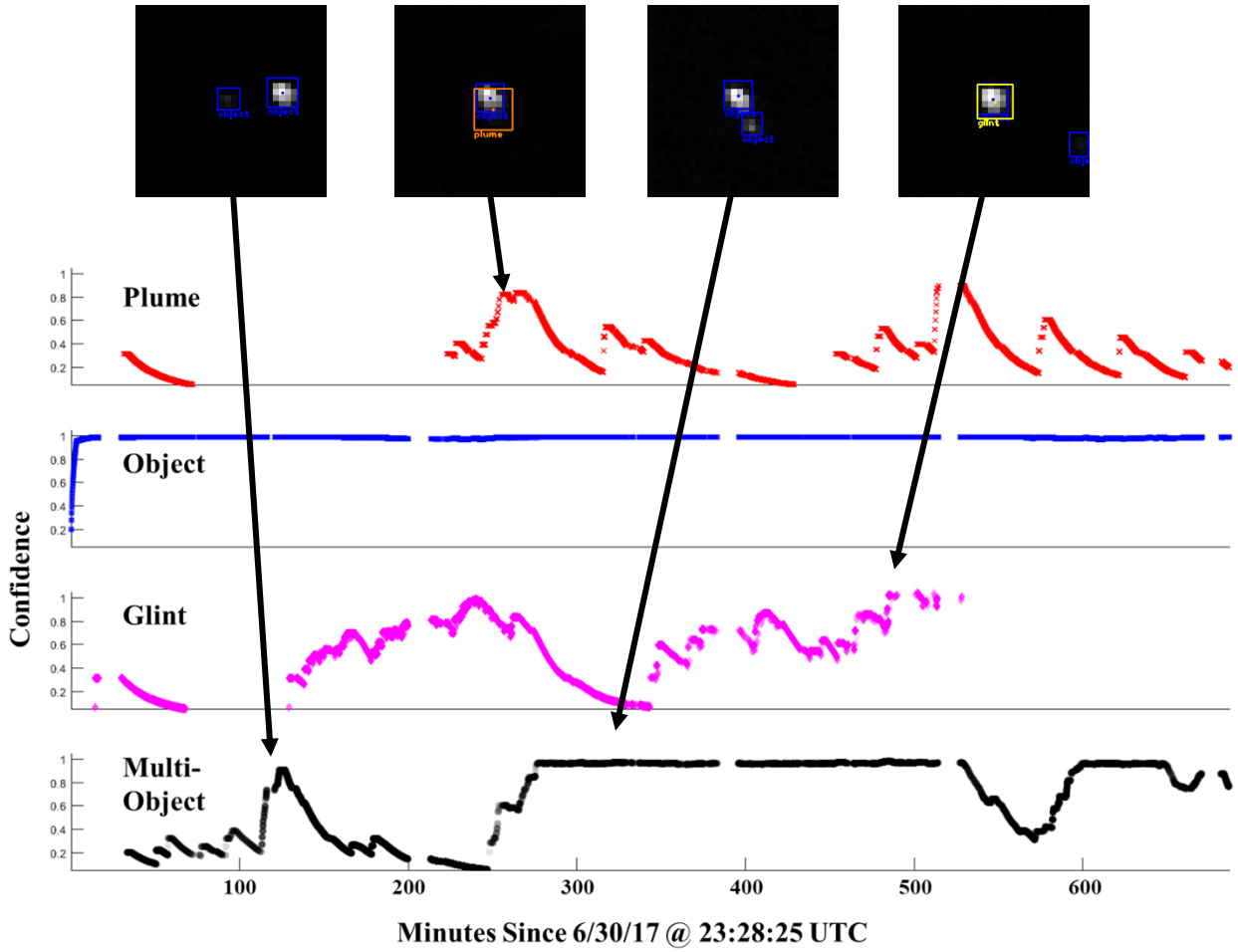

Figure 15. AMC-9 Timeline of Events

5.3 INTELSAT-29E

From April 8th to April 11th, 2019, Intelsat-29e suffered several debris generating events with the culminating and catastrophic failure occurring on April 11th, 2019 3:37:12 UTC. The events preceding the final spectacle were innocuous in comparison; nevertheless, they were characterized using astrometric and photometric standards. It was not until the final anomaly occurred that the entire series was analyzed, and other plumes were detected. This highlights a critical application for a plume detection algorithm.

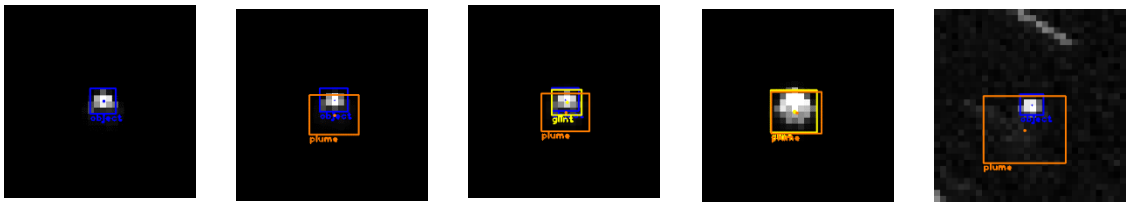

Figure 16. April 8th, 2019 1:21:48 UTC

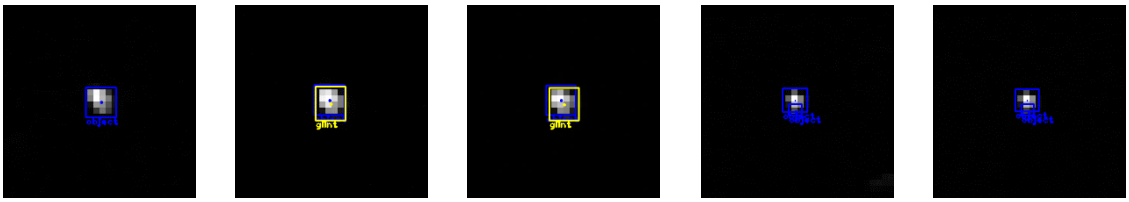

Figure 17. April 9th, 2019 3:53:06 UTC

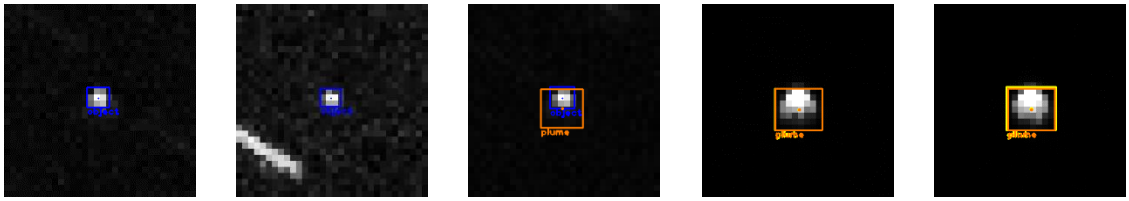

Figure 18. April 10th, 2019 7:46:40 UTC

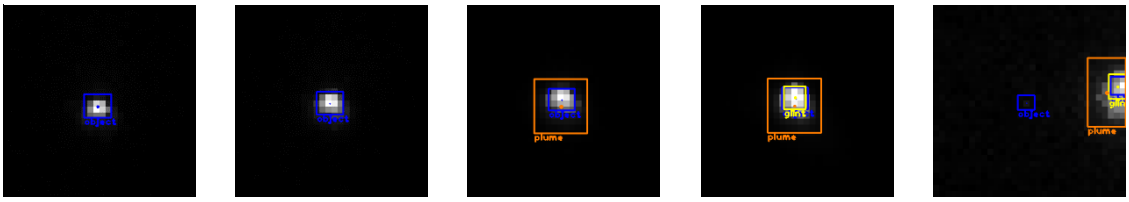

Figure 19. April 11th, 2019 3:37:12 UTC

Fig.16 clearly shows the first event in the series to follow with what appears to be a detected debris field around Intelsat-29e followed by a more abrupt plume and glint event. Over 24 hours later, Fig. 17 shows one of numerous periodic glints; however, this glint was followed by a trackable object emerging from the satellite itself. Similar to April 8th, Fig. 18 display an additional expulsion of material and glinting behavior. Fig. 19 depicts the catastrophic failure of Intelsat-29e. The fifth chip in the sequence is an off-center chip where the object to the right is Intelsat-29e and the center object is a piece of expelled debris being tracked.

6 IMPLICATIONS

6.1 DATABASE MINING

The completion of a tool that is capable of rapidly scouring, filtering, and reporting specific photometric events of interest is a powerful capability when paired with an insurmountable volume of data. ExoAnalytic Solution stores over one petabyte of imagery; these images can be decomposed into image chips. The number of image chips available for analysis is on the order of 500 million and grows at an accelerating number each day with the addition of new sensors. While there are numerous cases of un-cued anomaly detection utilizing standard photometric and astrometric methods, this algorithm provides an additional dimension of robustness. Therefore, one can use this to iterate over the database and point to images where events of interest likely occurred. When events that were once unknown are uncovered, these can be further analyzed and additionally used to train the model. As a result, the algorithm will become increasingly capable at detection unseen anomaly types for the application of real-time indications and warnings.

6.2 REAL-TIME INDICATIONS & WARNINGS

Subsequent to the utilization of this algorithm for scanning the database for previously undetected maneuvers, the additional collected information can be leverage for future events. While the network will be the same architecture used, it will now need to be retrained to include the additionally labeled anomalous events that were verified by a human-in-the-loop. The newly trained architecture, which takes roughly four hours to converge, is deployable to a data-stream where it persistently stares, scans, and segments the focal planes and anomalous events of interest. These classifications are flagged for further analysis; ultimately, this process is recursive and unendingly improving as greater volumes of data are utilized.

7 REFERENCES

1. C. Pelletier, G. I. Webb and F. Petitjean. Temporal Convolutional Neural Network for the Classification of Satellite Image Time Series. *arXiv:1811.10166v2 [cs.CV].* 31 Jan 2019

2. I. J. Goodfellow, J. Pouget-Abadie, M.Mirza, B. Xu, D. Warde-Farley, S. Ozair , A. Courville, Y. Bengio. Generative Adversarial Nets. *arXiv:1406.2661v1 [stat.ML].* 10 Jun 2014.

3. Y. Li. Deep Reinforcement Learning. *arXiv:1810.06339v1 [cs.LG].* 15 Oct 2018.

4. B. D. Little, C.Frueh. SSA Sensor Tasking: Comparison of Machine Learning with Classical Optimization Methods. *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, held in Wailea, Maui, September 2018.

5. D.Temple, M.Poole, M. Camp. Network Enabled - Unresolved Residual Analysis and Learning. *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, held in Wailea, Maui, September 2017.

6. D. Temple. Synthetic Heterogenous Anomaly and Maneuver – Neural Network Event Winnowing System. *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, held in Wailea, Maui, September 2018.

7. B. Zoph, Q. V. Le. Neural Architecture Search with Reinforcement Learning. *arXiv:1611.01578v2 [cs.LG]* 15 Feb 2017

8. S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Network. *arXiv:1506.01497v3 [cs.CV].* 6 January 2016.

9. J. Redmon, A. Farhadi. Yolov3: An incremental improvement. *arXiv:1804.02767 [cs.CV].* 2018.

10. PJ, Reddie. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016.

11. K. He, X. Zhang, S. Ren, J. Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385v1 [cs.CV]*. 10 Dec 2015

12. S. Ioffe, C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167 [cs.LG]*. 11 Feb 2015.

13. Wojke, Nicolai and Bewley, Alex and Paulus, Dietrich. Simple Online and Realtime Tracking with a Deep Association *Metric 2017 IEEE International Conference on Image Processing (ICIP) 3645—3649*