Chapter

Game Theoretic Training Enabled Deep Learning Solutions for Rapid Discovery of Satellite Behaviors

Dan Shen, Carolyn Sheaff, Genshe Chen, Jingyang Lu, Mengqing Guo, Erik Blasch and Khanh Pham

Abstract

The chapter presents a game theoretic training model enabling a deep learning solution for rapid discovery of satellite behaviors from collected sensor data. The solution has two parts, namely, Part 1 and Part 2. Part 1 is a PE game model that enables data augmentation method, and Part 2 uses convolutional neural networks (CNNs) for satellite behavior classification. The sensor data are propagated with the various maneuver strategies from the proposed space game models. Under the PE game theoretic framework, various satellite behaviors are simulated to generate synthetic datasets with labels for the training to detect space object behaviors. To evaluate the performance of the proposed PE model, a CNN model is designed and implemented for satellite behavior classification. Python 3 and TensorFlow are used in this implementation. The simulation results show that the trained machine learning model can efficiently and correctly classify the satellite behaviors up to 99.8%.

Keywords: space situational awareness, satellite characterization, sensor models, simulated training data, training performance, general-sum games, CNN

1. Introduction

Since space has already been fully utilized, society has become increasingly dependent on space advantages in various industrial, civil, and commercial applications. This dependence brings an essential vulnerability, especially shortage of continuous situational awareness of the space environment to ensure freedom of movement. Due to the fact that information from space is crucial for key decision-making, such as urban, agricultural, and responsive planning, space is regarded as a significant frontier. In addition to real-time and hidden information constrains, the existence of space object density significantly produces the complexity of the space situational awareness (SSA). Understanding the position of space objects from low-level information fusion can support high-level information fusion SSA missions of sensor, user, and task refinement [1]. In order to implement SSA accurately, it is possible to coordinate the evaluation of residential space objects (RSO) through user-defined operation pictures (UDOP) [2].

Space control and SSA are required for space prevalence, which depend on fast and precise space object behavioral discovery. Developing a theoretical approach for fast discovery of variation of satellite behaviors is the main task of this book

chapter. The machine learning methods with novel neural networks are proposed in this case. However, there are numerous challenges for constructing the tools because of the following reasons: (i) partially observable movements, (ii) resident space objects (RSOs), (iii) uncertainties modeling and propagation, (iv) real-time response, and (v) computationally intractable algorithms.

Space access investigation and mission trade-off considers are imperative for the victory of space-borne operations. The tracking algorithms of space object can be measured depending on collecting data to track satellites, debris, and natural phenomena (such as comets, asteroids, and solar flares). Tacking is related with sensor administration, which can point sensors to observation points to decide the circumstance and to aware threatens. SSA improvements consist of models (such as orbital mechanics), measurements, computational software (such as tracking), and application-based system coordination (such as situations). For instance, *game-theory* methods for SSA can be utilized for pursuit-evasion analysis [3].

This book chapter creates and establishes game theoretic training enabled deep learning (GTEL) methodologies for fast discovery of the behaviors from the satellites. GTEL is an adaptive feedback adversarial theoretic method, which acquires data from sensors related to the relationship between resident space objects (RSOs) of interest and sensing assets from ground and space (GSAs). Thus, a game theory is modeled instead of a control problem for this circumstance. Game reasoning uses data-level fusion, random modeling/propagation, on the other hand, RSO detection/tracking predicting the future RSOs-GSAs relationships. The adversarial engine also supports optional space pattern dictionary/semantic rules for adaptive transition in the Markov game. In the event that no existing pattern dictionary is accessible, GTEL will construct an initial pattern and modify it during the game inference. The output of GTEL inference consists of two parts of control methods: (i) measurements processing and (ii) RSOs localization. The two parts establish a *game-equilibrium*, one of which is sensing asset management, the other of which is the estimation of RSO behaviors.

The chapter is organized as the following. Section 2 introduces the comprehensive system design for our methods. Section 3 presents the Markov game theory with satellite maneuvering. Section 4 proposes the details of our machine learning model for space behavior detection. And the numerical results and analysis are displayed in Section 5. At the end, Section 6 draws the conclusion of this chapter.

2. Overall system architecture

The proposed methodology of GTEL is shown as **Figure 1**. The core piece of the method for detecting ***unknown*** patterns of space objects is Markov Game Engine. Due to the patterns are obscure, there is no preparing training data accessible. Therefore, the Markov Game Engine makes use of zero-shot learning [4] and transfer learning to unsupervised classify unavailable target domain data by training available data from source domain (i.e. simulate data). The knowledge adaptation or domain transfer is performed through manifold learning to share intermediate semantic embeddings (such as attributes) between labeled and unlabeled data. On one hand, the manifold learning can reduce the dimension of sensing data (such as azimuth angle, elevation angle, range, and range rate). On the other hand, the manifold learning can also mitigate the difficulties of object tracking and detection with fast space object behaviors' detection. Moreover, in order to solve the ***uncertainties*** in this task, the $\mathbb{R}^5 \times \mathbb{S}$ coordinate system is used with filtering technology on *optimal transport (OT)*. The essential components of our methodology are shown as the following:

1. *Markov game* – The space conflicting situation is inferred by the Markov game structure [5, 6], in which system state is represented by distributions rather than deterministic values.

2. *Uncertainty modeling and propagation* – The uncertainties are expressed by the production of Von-Mises and independent Gaussian distribution for both measurement interference and original state uncertainty that are identified on cylindrical manifold $\mathbb{R}^5 \times \mathbb{S}$ [7]. Convergence is not guaranteed with relaxed synchronization for uncertainty spread of information.

3. *Optimal transport based tracking of space objects in cylindrical manifolds* – Compared with ensemble Kalman filter (EnKF) methods for tracking space objects, optimal transport (OT) [8, 9] is much more precise and robust. Furthermore, it is way broader than the algorithms with uncertainties assumption in $\mathbb{R}n$, because OT is not related to the distribution (since OT is focused on a transformation instead of importance sampling).

4. *Course of actions for behavior modeling* – The behavior of RSO is specified a course of action (CoA), which in turn dictates to what the RSO may appear to perform or is doing within subsequent a couple of steps.

5. *Manifold learning for data level sensor fusion* – The raw sensor data is generally high-dimensional. Considering that the measurement data is reflections of witnessed satellites (only several parameters will determine their states), so it can be reasonably assumed that the inherent dimension of the measurement domain is low. The communication bandwidth can be saved with decreasing the number of dimensions by using Manifold learning algorithms [10].

The simulated positions of satellite are used to generate sensor measurements, and the results will be utilized to track several space objects to complete the

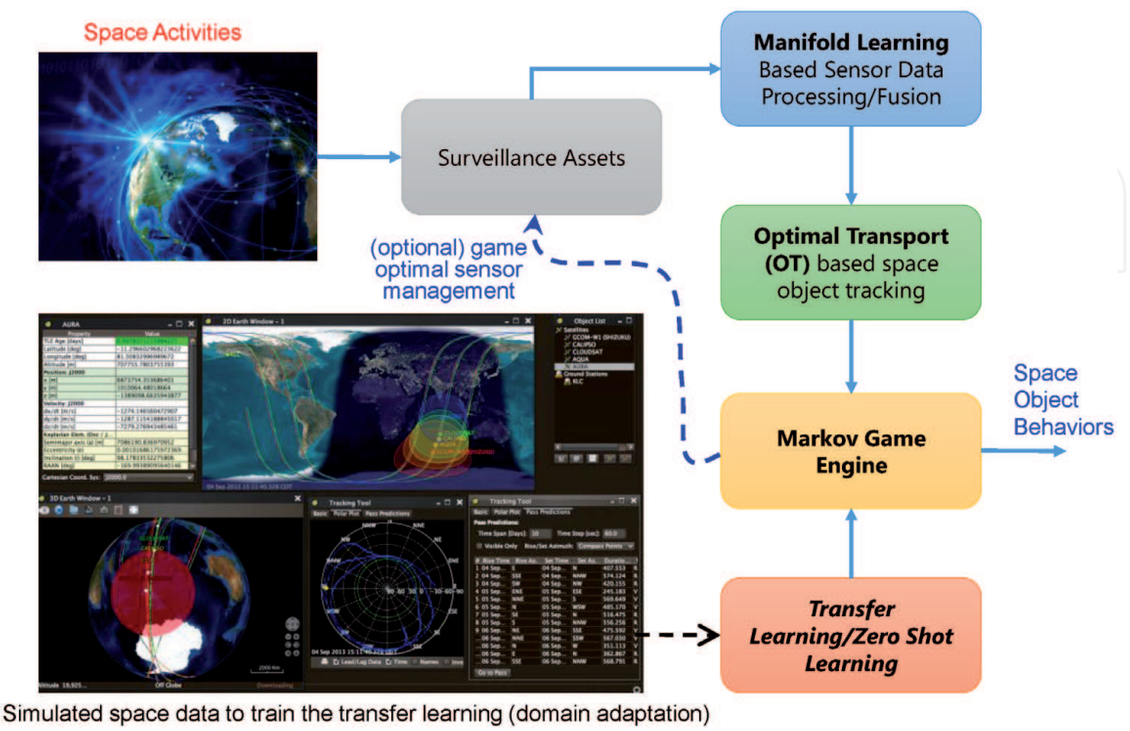


Figure 1.
GTEL system architecture.

estimation of the position. Afterwards, the estimations of the orbits are used for collision alert and maneuver detection of space objects. The satellite maneuvers are going to be interpreted as platform commands to perform course of actions and space object movements.

Figure 1 shows an adaptive feedback approach with the game theory enabled. It utilizes sensors to obtain information about the relationships between the ground/space sensing assets and the RSOs of interest [11]. RSOs and GSAs determine the relations. Thus, rather than saying it is a control problem, it is a dynamic game. Data-level fusion, game reasoning, RSO detection/tracking, and uncertainty modeling/propagation are combined to predict the RSO-SA relationships in the future. Optional space behavior dictionary/semantic rule for adaptive transition matrices in our Markov game is also supported by our game engine. In addition, the game solution is an equilibrium, which is controlled by both space sensing asset management and the (estimated) RSO behaviors.

3. Markov game in space situational awareness

Lloyd Shapley has invented a concept of stochastic game [6], which is a dynamic game played by one or several players focusing on probabilistic transitions. There are several stages for this game. At first, the game is set in one state. Then, the participated players should select an action individually. Based on the current state and the actions players chosen, each player will receive a reward. Therefore, after the chosen from each player, the game comes to a new random state, where the previous state and previous actions chosen by the players determine a distribution of the new random state. Afterwards, the above action will be repeated again for the new state. After finite or infinite number of stages of playing, the total reward for each player is obtained using the discounted sums of each stage reward or the averages of every stage rewards. In this way, each player gets a reward and the reward is compared with each other. The aforementioned sequences are the procedures for the stochastic game, which can be generalized by Markov decision processes (MDP) with repeated games. Our space situational awareness (SSA) would utilize this game tool for intent prediction [12].

The Markov game engine extracts specific information from each event as the following: (*i*) a finite set of players N, (*ii*) a finite or infinite set of states, S, (*iii*) a finite set of accessible actions for each player in N set, D^{i} (the overall action space is $D = \times_{i \in N} D^{i}$), (*iv*) a transition rule $q: S \times D \rightarrow \Delta(S)$, (where $\Delta(S)$ is the space of all probability distributions over S), and (*v*) a reward function $r: S \times D \rightarrow R^{N}$.

Figure 2 shows a visual description of the simple game states with only two players, who have only two options of actions for each player. The arrows in the graph indicates the probable transitions from one state to the other state. The states with red color means that only player 1 changes the approach. On the contrary, the states with blue color indicates only player 2 changes the approach. The state with green color indicates both players change the approach.

The GTEL solution uses a two-player Markov game to investigate the sensor management for tracking space objects. Whether deliberate or unintentional, some of space objects may cause confusion to observers (sensors) when the orbital maneuvers are performed. In general, spatial object tracking can be assumed as an optimal control problem (one side optimization) or a game problem (two side optimization). For the settings of optimal control, the position and velocity of the space objects will be calculated (filtered) as the states dependent on the measurement from the sensor. However, this method ignores the possibility that the space objects may alter their orbits purposely with intelligence. It may cause difficulties for the

satellite to track the space objects. Therefore, the Markov game method provides a solution for these difficulties. In this approach, on the one hand, the observed satellite will utilize the tracking and sensing model to destroy the tracking estimations by confusing the observer. On the other hand, the observer figures out ways to minimize the uncertainties of tracking, where the uncertainties are dependent on the entropy of tracking.

In this chapter, the information uncertainty of the GTEL pursuit-evasion (PE) game method [13] was exercised with a circumstance of two satellites, one of which is Geostationary Earth Orbit – GEO (observed satellite), the other of which is space based Low Earth Orbit-LEO satellite (observer satellite). **Figure 3** provides an illustration of a space based optical (SBO) sensor measurement model. The angle from the line from the object to the SBO and the line from the sun with object is defined as Bistatic Solar Angle, represented by θ. The smaller the angle is, the stronger the lighting conditions. Therefore, it causes difficulties for observations when the angle is large because of saturation of lighting. As shown in **Figure 4**, the scenarios of light have shown. In the graph, the blue line is for the LEO orbit, the green line

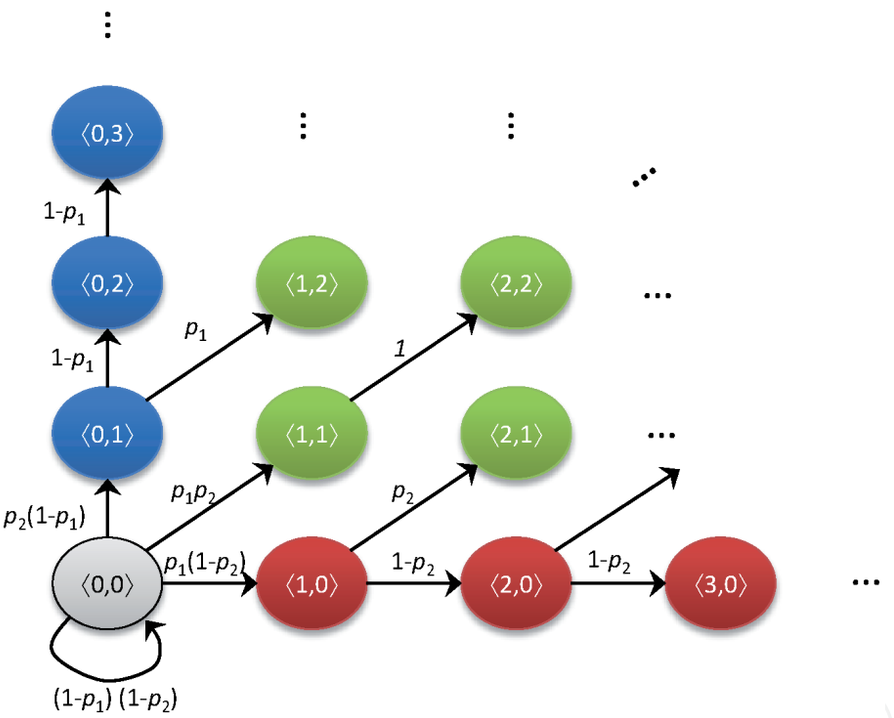


Figure 2.
A diagram of states in a Markov game.

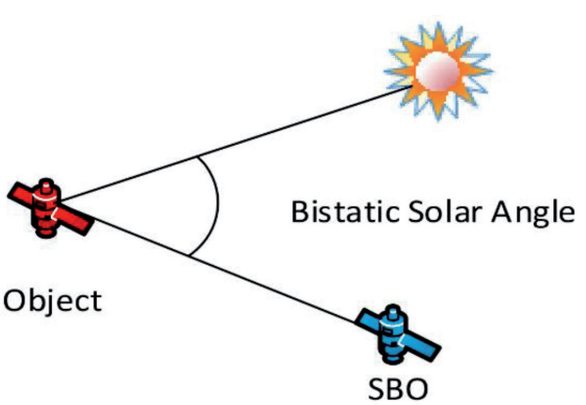


Figure 3.
SBO with a Bistatic solar angle.

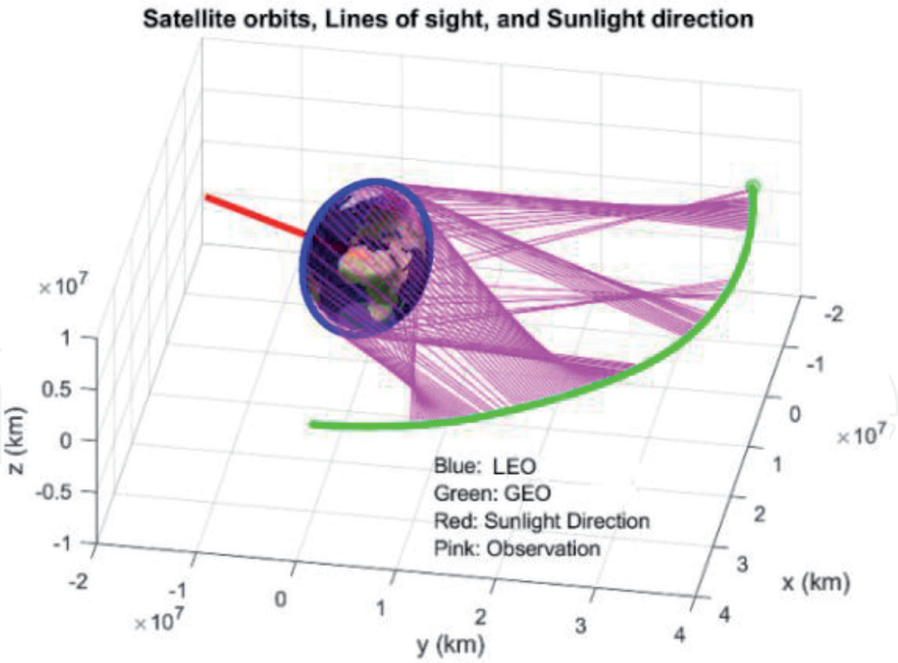


Figure 4.
LEO and GEO based on sensor management and maneuver strategies with game theory.

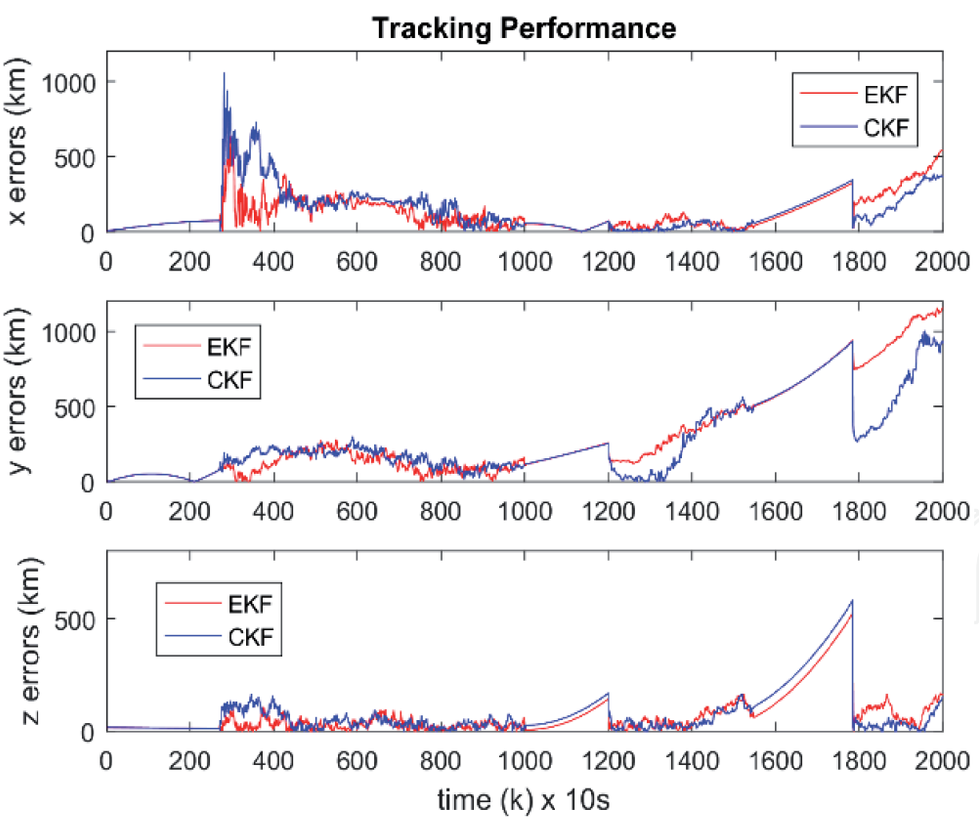


Figure 5.
The performance of tracking using theoretical Markov game strategies.

indicates the GEO orbit with maneuvers, the pink lines are the SBO sensor performed to track the GEO in order to lower the uncertainty, and the red line displays the sunlight from the sun to the earth.

The research scenario is shown in **Figure 4**, where the red line indicates the direction of sun light, green color is for the GEO orbit with maneuvers, blue color

for LEO orbit, and pink lines indicate when the SBO sensor resource is used to track the GEO (use the sensor data to lower the uncertainty).

Figure 5 shows the results of tracking based on intermitted measurements. Both cubature KF (CKF) and extended Kalman Filter (EKF) trackers are shown. With the increase of the period without measurement due to the Earth blockage, the tracking errors increased as well. In addition, the tracking errors increased with the maneuver actions from the satellite. On the contrary, informational entropy decreased by sensor measures in this process.

Figure 6 top graph displayed the PE game control results, with α and β as angles of the maneuver thrust. The zoom-in view of the game optimal controls is shown

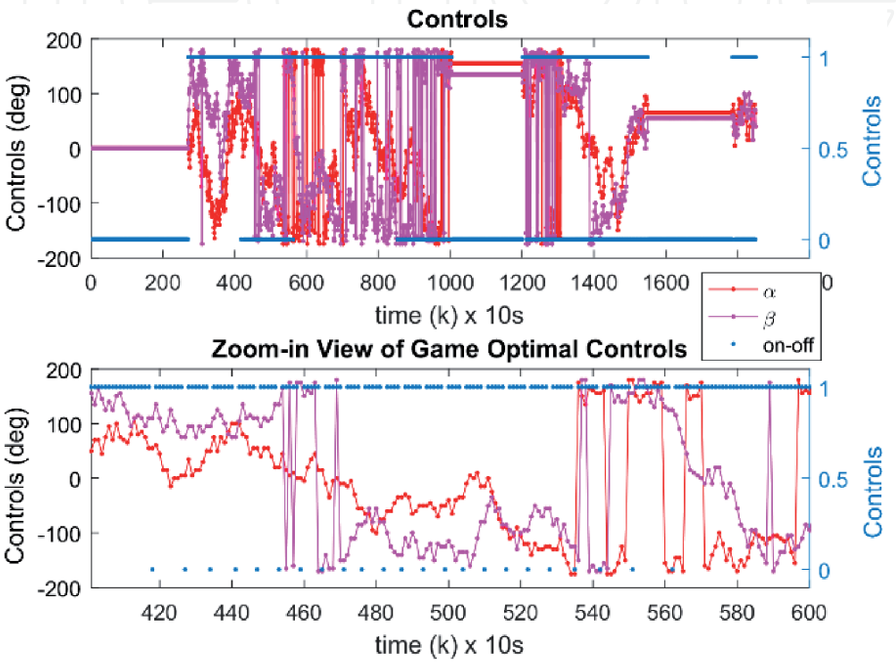


Figure 6.
The maneuver controls dependent on the PE game solution for the satellite direction.

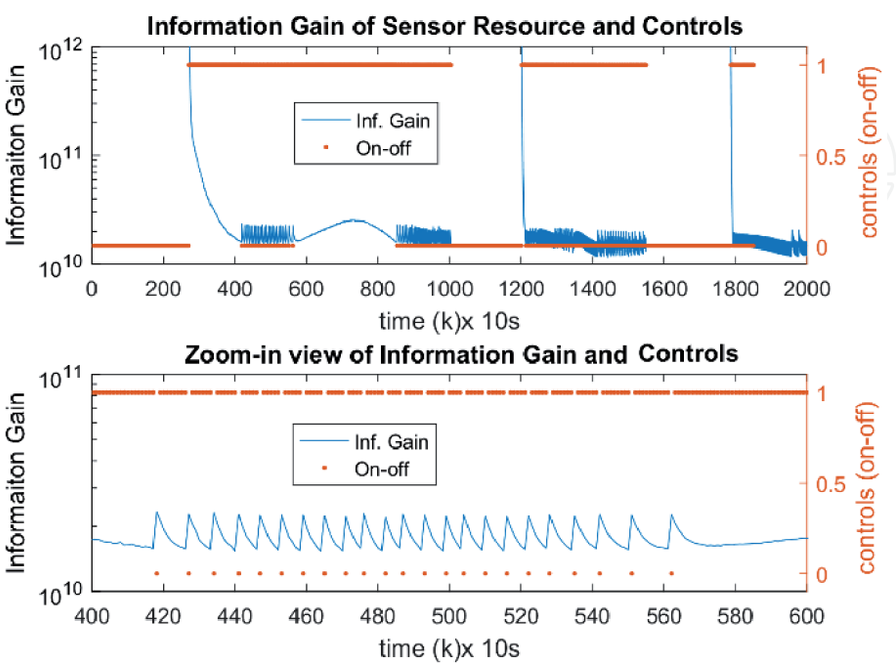


Figure 7.
Sensor's game theoretic on–off controls and associated information gains.

Figure 6 below, which exhibited that the observer satellite can keep the tracking uncertainty within a desired level while saving the resources of sensor. **Figure 7** shows the on–off sensor controls and the associated information gains. The results show that with the larger potential information gain, the sensor would use the resources to take measures (for the observer's on–off control, 0 means turning off and 1 indicates turning on).

4. Machine learning based RSO behavior pattern classification

4.1 The scheme of machine learning

The machine learning (ML) details will be described in this section. The main purpose of construction this machine learning scheme is to detect the behaviors of the resident space object (RSO) by fusing sensors data from multiple sources, including the velocity, orbital energy, angular momentum, and the position of RSO compared to the station. Similar with other machine learning model, this model is generally trained off-line by using generated data. Then the generalized weights will be deployed in the application using TensorFlow deployment. In addition, in order to improve the robustness of the trained system, we proposed a neural network scheme with the ability to train the newly-added unknown pattern online with only tiny modifications of the weights.

As shown in **Figure 8**, the RSO pattern classification architecture is displayed. It consists of two separate parts, one of which is offline part as *Modeling*, the other of which is online part as *Monitoring*, to detect the RSO behavior pattern.

As the matter of the offline part, named as *Modeling RSO Behavior Pattern*, our neural networks will be trained as a classifier by using the collected simulated data. The data specifically indicates the different behaviors of the RSO. As shown in the graph, some useful features are obtained from different sensors in the sessions of feature extraction. Subsequently, the extracted feature will be sent into the training model for neural network tuning and training to generate a classifier, which is used to identify the RSO behavior with fine grained size. On the contrary, the online part, *Monitoring RSO Behavior Pattern*, acquires RSO patterns in real-time to distinguish the abnormal behaviors. In this way, a warning message would be prompted if there are any abnormal behaviors detected.

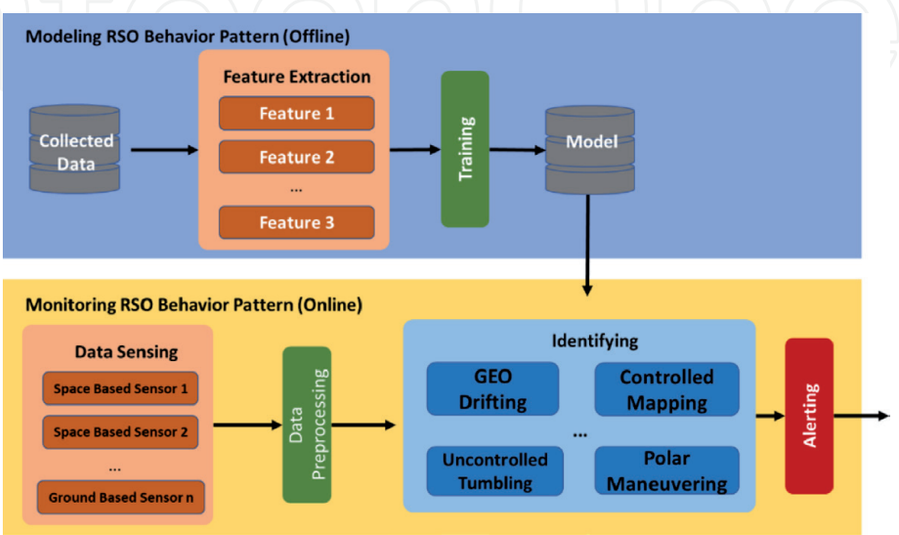


Figure 8.
The scheme of RSO behavior classification.

Additionally, the classifier generated by the machine learning methods has several properties as the following: (a) ability to fuse heterogeneous and complex input data, (b) scalability, (c) robustness with perturbations relative to the data, (d) high accuracy, and (e) explicitly. However, it is hard to fulfill all the requirements as shown above, therefore, some trade off would be considered for our model. Among different structures of neural networks, convolutional neural networks fit our case appropriately.

4.2 Framework of convolutional neural networks (CNNs)

The structures of Convolutional Neural networks (CNN) as well as Dense Neural Networks (DNN) are shown in **Figure 9**. The DNN neural networks consist of several hidden layers with hidden neurons. Every neuron in the subsequent layer is fully connected to the neurons in the previous hidden layer. The neurons contain the linear functions with activation function for each neuron, which is completely independent with the other neurons. Finally, after going through several layers, the input data is generated to the output data as the "output layer", which utilizes softmax activation function to produce classification probabilities.

However, the DNN (as top image shown in **Figure 9**) has several drawbacks, such as scaling difficulties for large images. For instance, for an image with 18 × 18 × 3 dimensions, DNN for the first layers will have 18 × 18 × 3 = 972 neurons. If the next DNN layer has 30 neurons, the weight parameters will be 972 × 30 + 30 = 29190. With a larger size of image, the weight parameters will increase a lot as well. Moreover, the 30 neurons may not have enough complexity to generalize the accuracy of our classification model.

On the other hand, the CNN (as bottom graph shown in **Figure 9**) can solve the aforementioned issues. A simple CNN [14] consists of several different filters as convolutional layer. In addition, the pooling layer can decrease the dimensions of the input data. Meanwhile, since the adjacent data in an image has similar values,

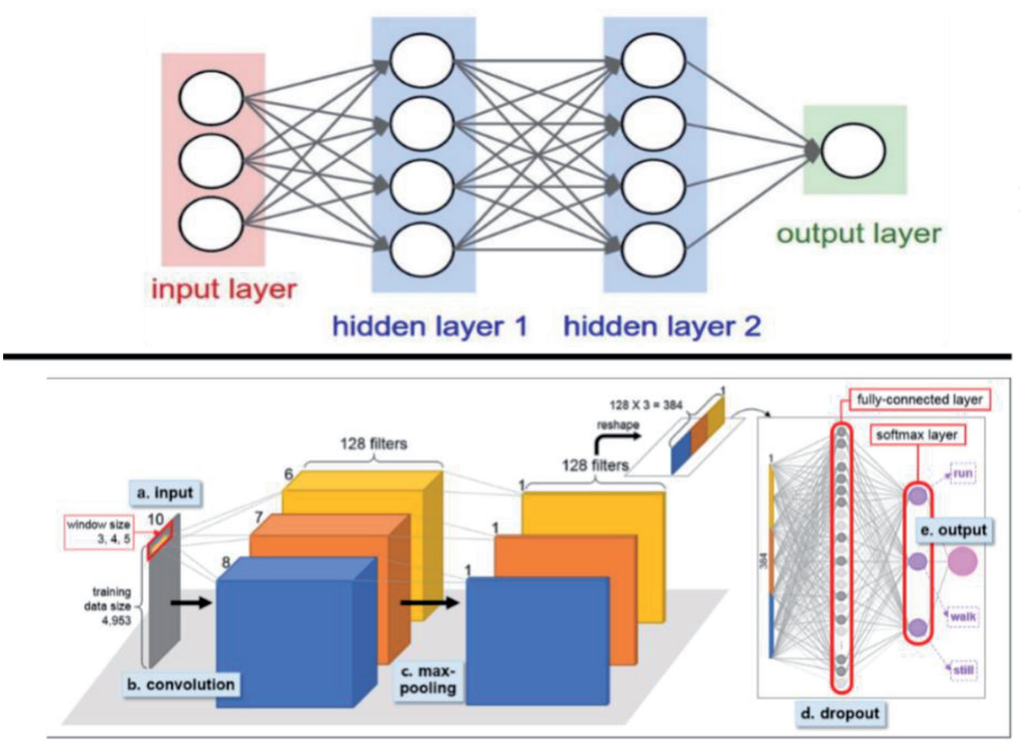


Figure 9.
The framework of the convolutional neural network.

the filter can extract the features reasonably with few weight parameters. Thus, using the CNN filters can increase the accuracy of our neural network classifier.

With the development of CNN, it successfully proceeds to be the best model structure for computer vision and image processing tasks. Moreover, CNN has also been used by Natural Language Processing (NLP) area. With small filters moving across the input dataset, the filters, with only small numbers of parameters, are re-utilized to recognize patterns for the large image. Therefore, with the similar classification abilities, CNN network is faster to train and predict compared with DNN. After several filters, the output part is flattened to bypass several dense layers to produce the softmax activation at the last layer with the sparse cross-entropy loss function to backpropagation for the behavior of RSO pattern prediction.

Our GTEL PE method utilizes the CNN architecture to classify the RSO pattern with observed data. Compared with the other conventional methods, the convolutional neural networks can process the RSO observation much faster. Therefore, we utilize the Python and TensorFlow with Keras [15] as the fundamentals for code write up. Although, each filter is computationally expensive to be trained, the overall CNN architecture is faster to be trained to provide similar accuracy classification.

A typical CNN structure is shown in **Figure 10**. After training our CNN-DNN with our training data, the test data (with 10–20%) will be employed to evaluate the generalization of our CNN-DNN model. Additionally, in order to solve the

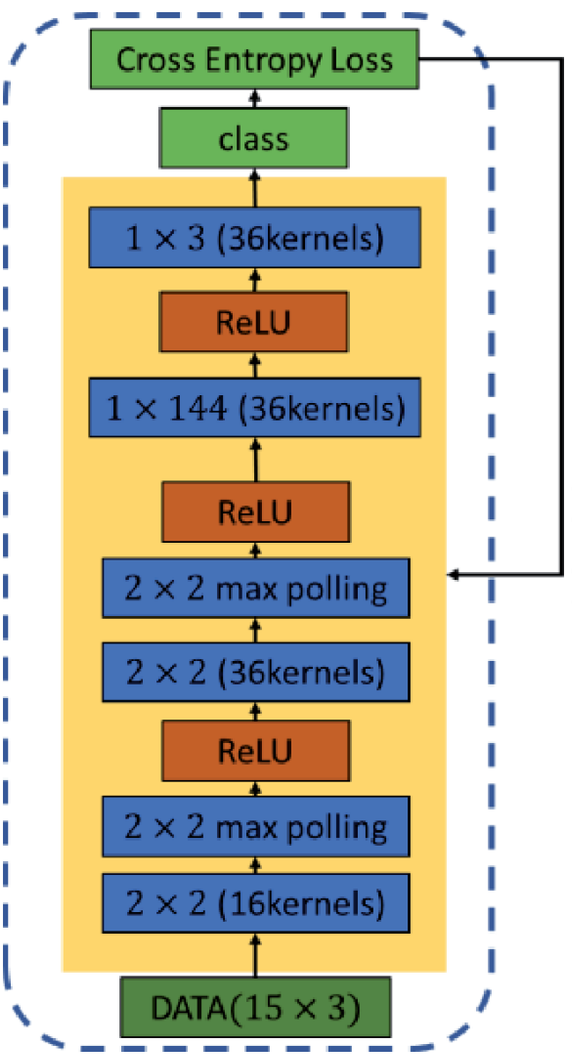


Figure 10.
CNN architecture for RSO behavior classification.

overfitting problem of our model, several dropout layers were added after each layer. This model outperforms other traditional methods with better accuracy and higher computationally efficiency.

The revised deep learning neural network structure for 143-sapce-behavior (we partition the pointing angles of α and β into cells with 15 degrees each and the total cell number is 143) is shown in **Figure 11**. At first, there are 3 different dimensions in our raw data, which are 3 parameters dimension, 15 times (each track consists of 15 observation measurements) interval dimension, and 1 number of channels dimension. Therefore, the convolutional neural network (CNN) comprises an initial 3*15*1 input dimension with 72,000 samples as the raw dataset. In order to distinguish the 143 different labels of satellite behaviors (15-degree difference for between each behavior), 128 "2 × 2" filters are utilized with the same padding for the first convolutional layer. Thus, there are 3*15*128 dimensions of output

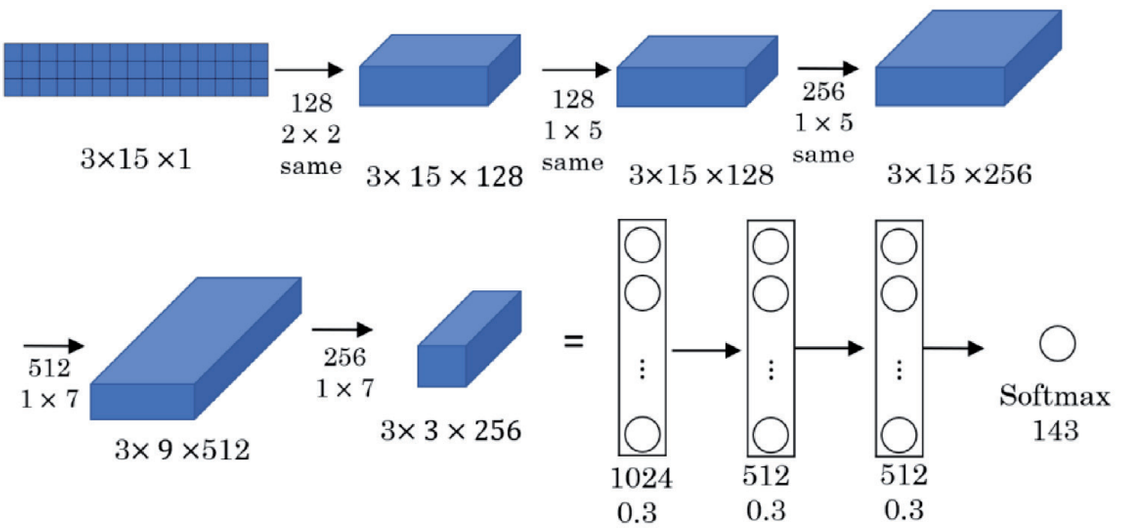


Figure 11.
Revised CNN structure for 143-label data.

Layer (type)	Output shape	Param #
Conv2d	(None, 3, 15, 128)	640
Conv2d	(None, 3, 15, 128)	82,048
Conv2d	(None, 3, 15, 256)	164,096
Conv2d	(None, 3, 9, 512)	918,016
Conv2d	(None, 3, 3, 256)	917,760
Flatten	(None, 2304)	0
Dense	(None, 1024)	2,360,320
Dropout	(None, 1024)	0
Dense	(None, 512)	524,800
Dropout	(None, 512)	0
Dense	(None, 512)	262,656
Dropout	(None, 512)	0
Dense	(None, 143)	73,359

Table 1.
The parameters of our 143 classification CNN-DNN model.

after the first layer. Then the previous output bypasses another two convolutional layers with 128 and 256 "1 × 5" filters together with the same paddings. After the first three convolutional layers, the dimension of the data exploded to 3*15*256.

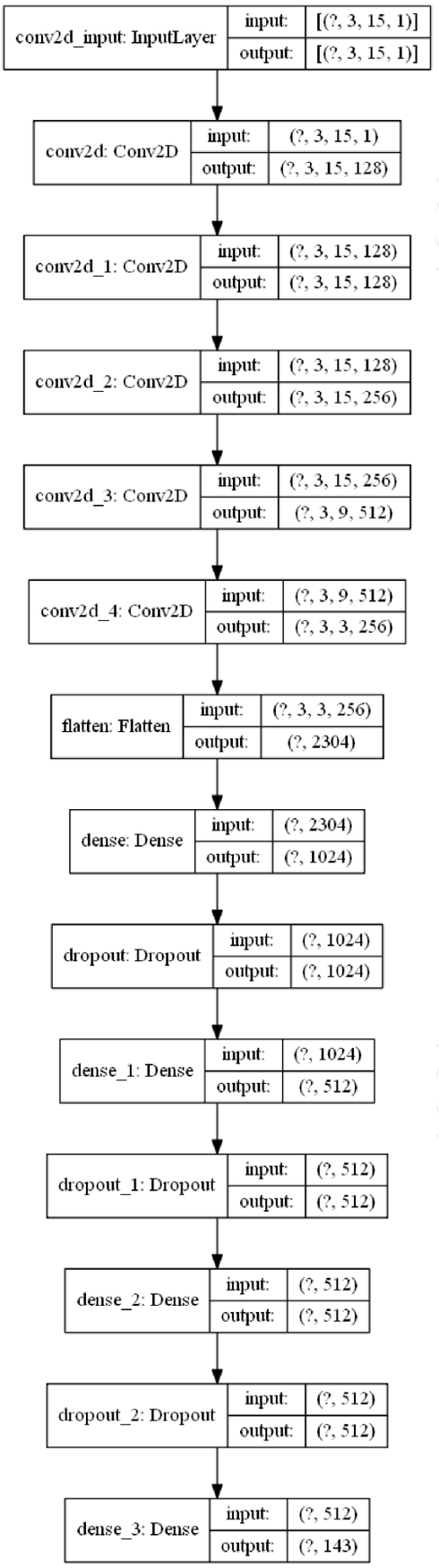


Figure 12.
Model structure for the CNN_DNN network.

Afterwards, another two convolutional layers are added with the same padding methods to downsample the dimensions to 3*3*256 for future operation. The two layers both has 1 × 7 filters with 512, and 256 third dimension respectively. Finally, the 3*3*256 data is flattened to one dimension to bypass three dense layers with 1024, 512, and 512 neurons, respectively. All these layers utilized 30% dropout parameters after dense layers. The employment of the dropout layer is used for solving the overfitting problem during the training process. In order to predict the 143 labels classifications, there is a 143-degree softmax attached at the last layer of the CNN-DNN model. Additionally, the cross-entropy loss function with the Adam Optimizer minimizes the result with the gradient of the next calculation points.

The details of the parameters of the CNN are shown in **Table 1** and **Figure 12**. There are 5,303,695 parameters that need to be trained in this large NN for detecting the different behaviors. In order to train the GTEL CNN much faster and more accurately, a learning rate decay is used during the training process. For the first 75 epochs, the learning rate is $1e^{-4}$, which can learn faster after the weight initializing. After 75 epochs, the exponentially decay of learning rate finds the optimized path for decreasing the loss for the CNN model.

5. Numerical results and discussion

5.1 Training data generation

In order to capture the training data with the two-line elements (TLEs) from the space-track.org, we modified the catalog tracking by the addition of the maneuvers. The different maneuvers indicate different labels in our training data. As an example, the maneuver to increase the energy of the orbital is labeled as 1. On the contrary, the maneuver to decrease the energy of orbital is labeled as 2. In this way, adding no maneuver is labeled as 3. The details related to adding maneuvers are shown as the following:

a. The earth-centered inertial coordinates (ECI) is converted from the TLEs at 0 time step;

b. Use the specified methods (Markov game) shown in Section 3 to propagate the satellites;

c. Convert the 16 waypoints back to azimuth, elevation, range, range rate relative to a ground site.

Therefore, almost 57,332 tracks were generated for training dataset with the other 6371 tracks as the testing dataset. The original first 10,000 tracks are shown in **Figure 13** for 143-space-behavior maneuvers.

The data format is list as:

- Column 1: track id
- Column 2: observation id (from 1 to 15)
- Column 3: Azimuth angle (rad)
- Column 4: Elevation angle (rad)

- Column 5: Range (km)
- Column 6: Training label (from 1 to *143*)

5.2 Results and analysis of our CNN_DNN neural networks

After 1000 epoch optimization, the GTEL systems achieved almost 98% training accuracy with 96% test accuracy as shown in **Figure 14**. There is a little overfitting after 400 epochs of training, which can be solved by adding more dropout layers and other methods. **Figure 15** displays the cross-entropy loss during the training process. The training set loss is always decreasing; however, the loss of validation set is flat after 200 epochs of training, indicating some overfitting issue after 200 epochs of training of the data.

Figure 16 shows the confusion matrix to evaluate our model's performance. As shown, for both training and test datasets, the true labels are almost the same with the predicted label by the GTEL CNN model. The performance evolution pattern is shown in **Figure 17**.

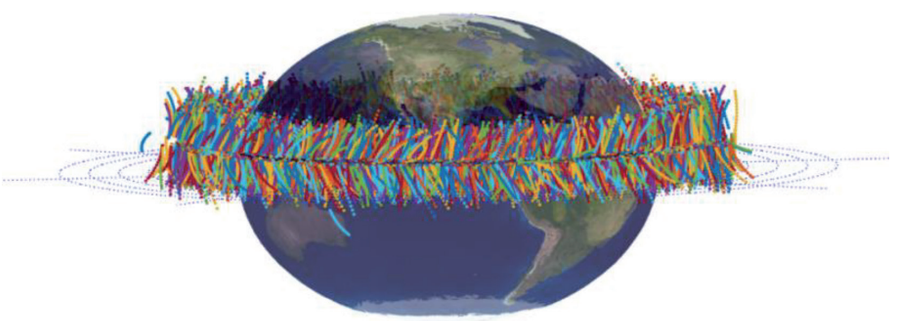


Figure 13.
The first 10,000 training tracks with various space behaviors.

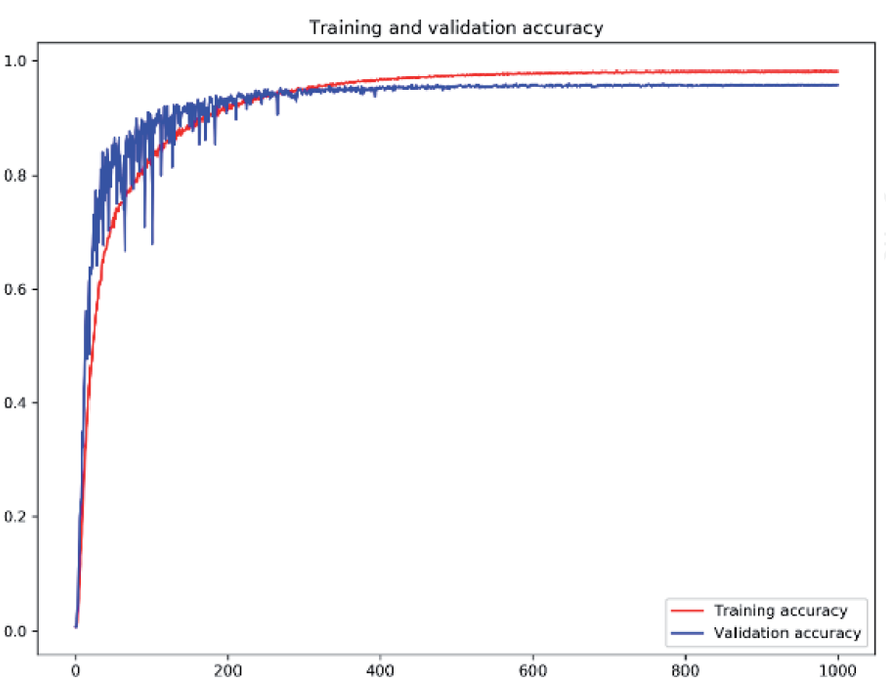


Figure 14.
Training and validation accuracy during the training process.

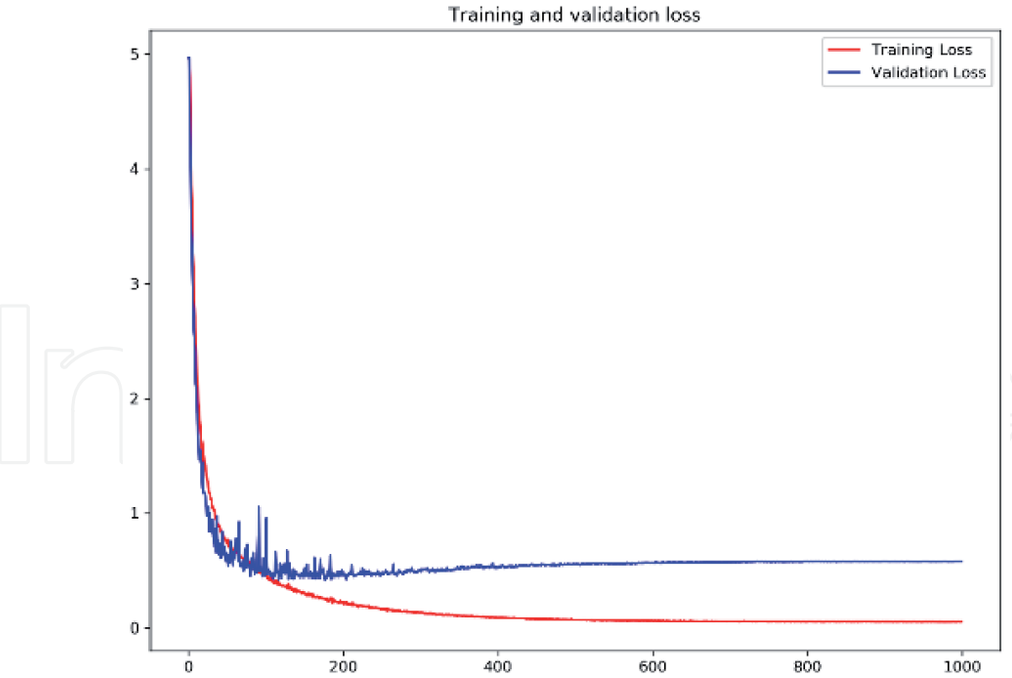


Figure 15.
Training and validation loss during the training process.

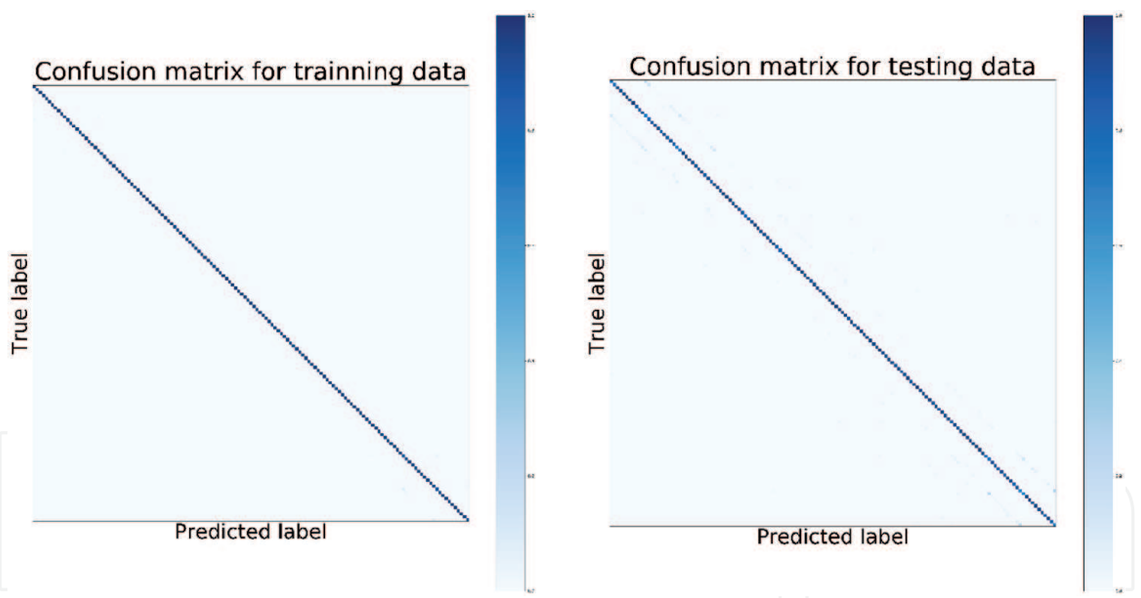


Figure 16.
Confusion matrix of training data and testing data.

For filter-based normalization, changes are grouped normalized by filters, which aims to display the change distributions over iterations for individual filters. For instance, as shown in **Figure 18**, the filter changes in the convolutional layer and dense layers are visualized. The changes are drastic in the first several iterations and become relatively small in the later stages (after 400 epochs) for most of the layers due to learning rate decay as well as the convergence of the GTEL CNN model.

Notice in **Figure 18** (the color maps are shown on the right side of each plot, the whiter the more stable of the training process), there is no constant deep blue color

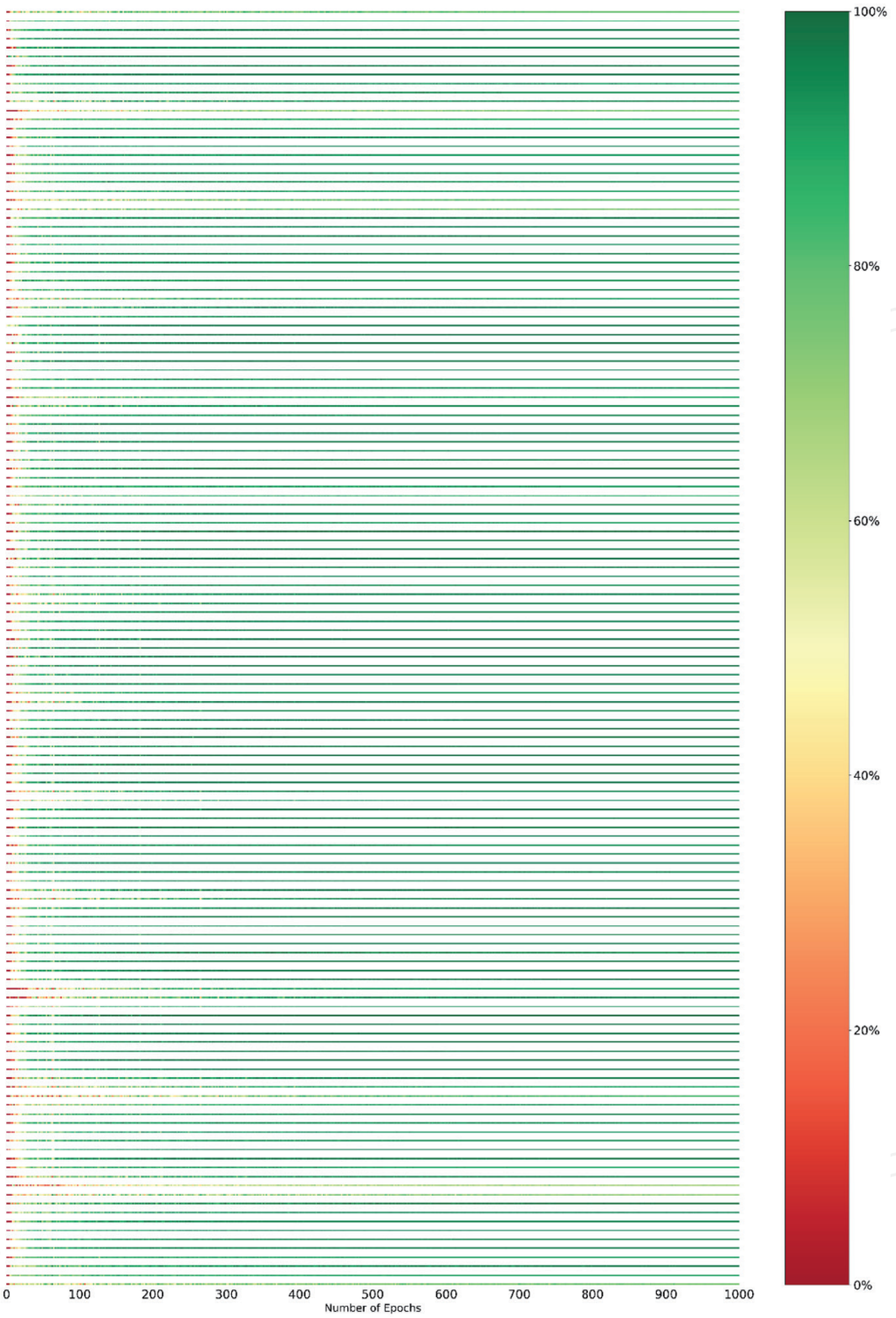
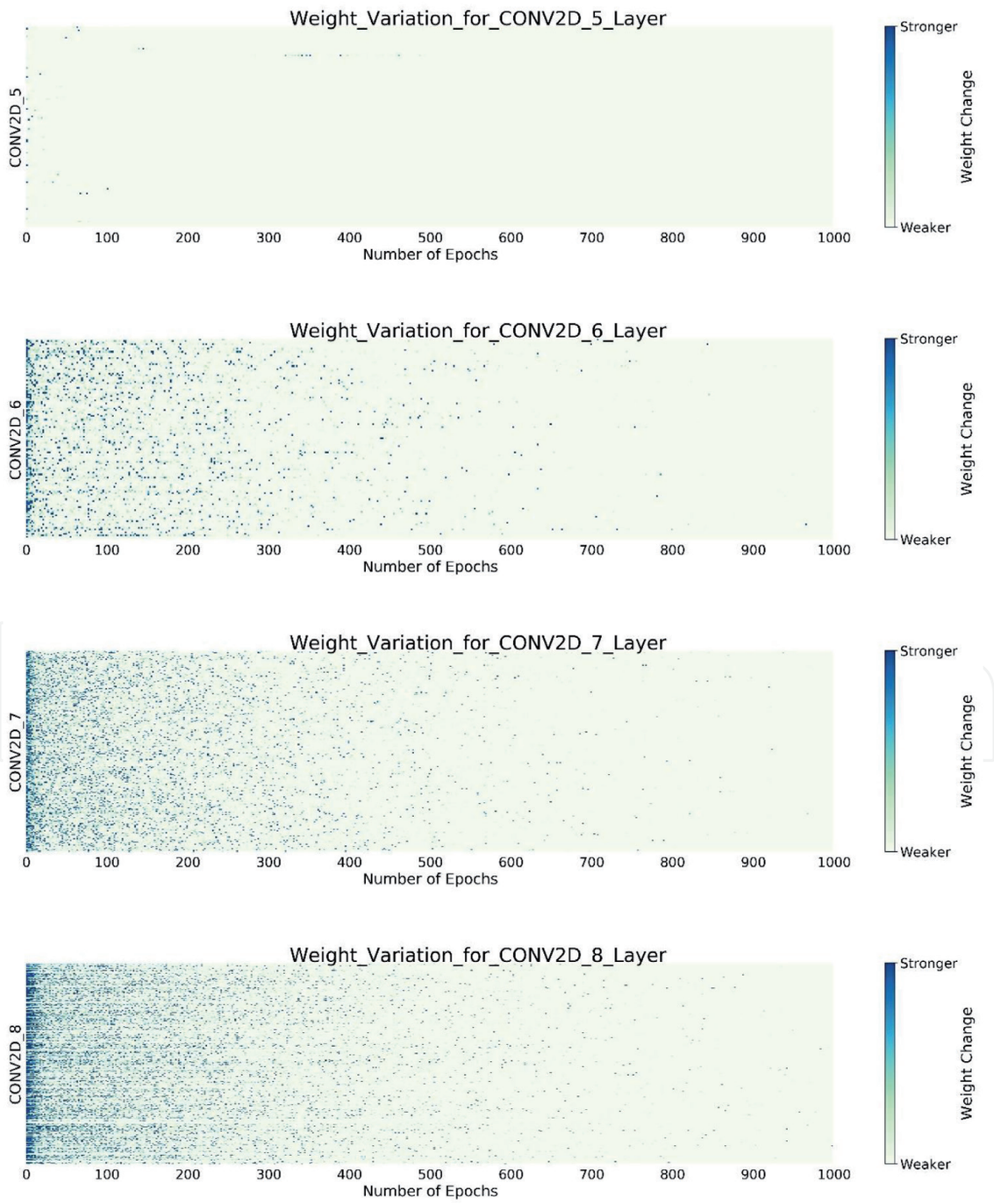


Figure 17.
Overview of validation classes prediction accuracy during training epochs.

for all training process, indicating the great performance and reasonable training configuration during training of the deep learning neural networks. Understanding the meaning for these weights during training process with further checking is very important in the CNN model design, which leads to more reasonable explanation in the deployment of the model.

In addition, we investigated the filter image correlations in a broad overview. As shown in **Figure 19**, rows and columns represent layers and image classes, respectively. A sequential color scheme is used to encode the number of anomaly

filters to intuitively represent these relationships between satellite behavior labels and anomaly filters. Using a grid-style visualization, interpretability is possible where rows and columns represent layers and classes of behaviors, respectively. The number of rows and columns equal to the number of layers with anomaly filters and classes with anomaly iterations each class, respectively. In **Figure 19**, the darker the color it is, the more anomaly weights and filters appear in that layer, which are related to that class. From this visualization, it is easy to observe that the second fully connected dense layer has the most anomaly weights and filters among all the other layers. Hence, there are some trends for most anomaly weights and filters during training processes, especially the middle layers of the deep convolutional neural networks. In addition, the anomaly class seems to have the same interval in the dataset, indicating more work is needed for these classes and layers to build a better network. It displays a similar trend compared to the results as shown in **Figure 17**.



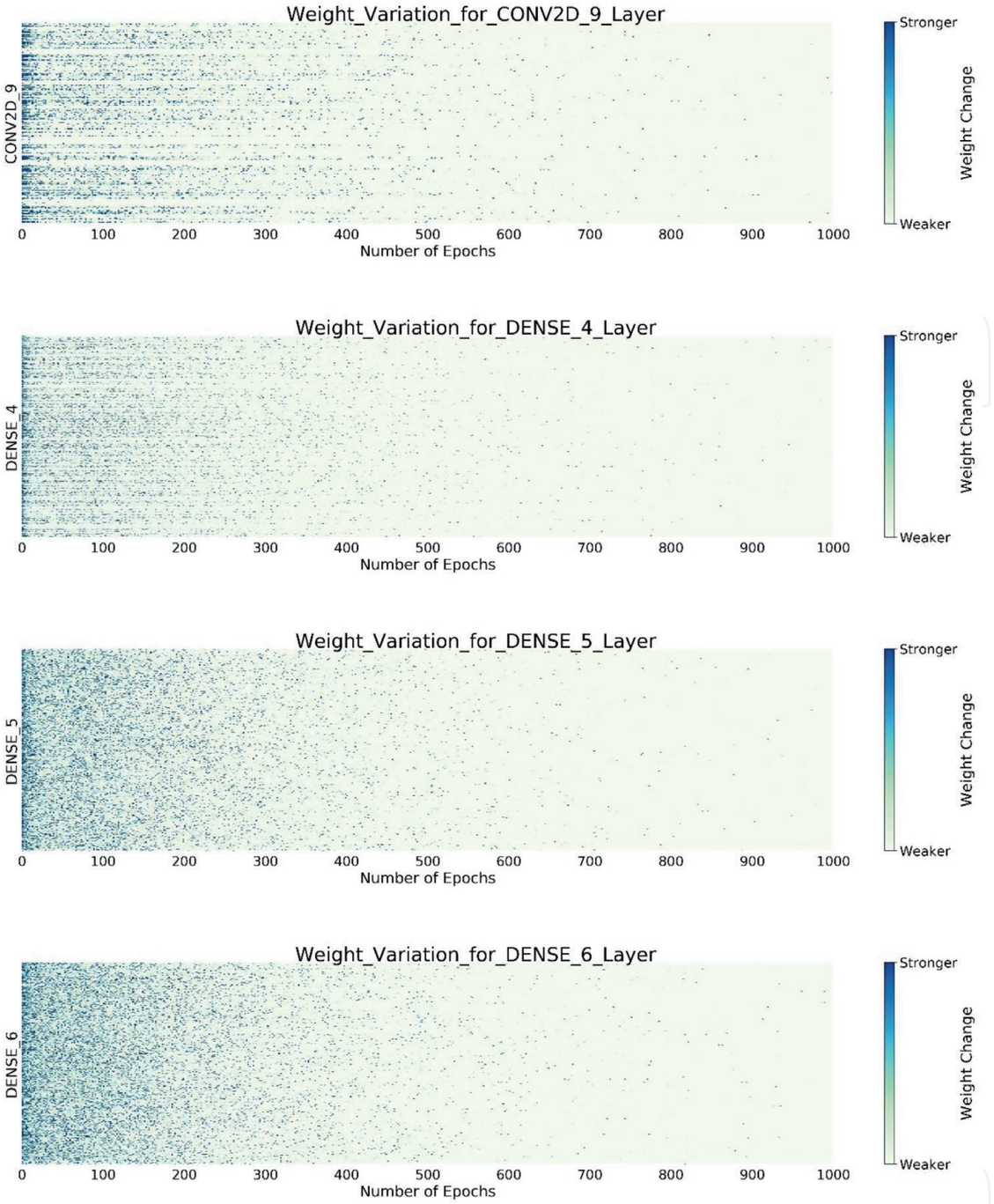


Figure 18.
The weight changes in filters during iterations. More blue color indicates stronger variation for the filters during iterations.

Figure 19.
The abstract version of correlation view.

6. Conclusions

In this chapter, a machine learning design has been presented and implemented to discover space object behaviors. Our GTEL methods using CNN models the circumstance with PE game rather than a control problem. The stochastic modeling/propagation, RSO tracking, and data level fusion are utilized to predict the relations for future space by game reasoning. In order to generate the data for training, the Marko game approach is used with maneuvering strategies. The method provides a way to solve the SSA using unknown behaviors. Additionally, the unknown behaviors exist where a satellite employs the tracking and sensing way to corrupt the tracking estimates to perturb the sensors. On the other hand, the space sensors decrease the uncertainties during tracking process. Finally, the CNN-DNN is used to train the numerical results, where the accuracy is 98% for our classification RSO model with 143 labels.

In the future, a multi-player game theory with adversarial network with SSA will be employed to enhance the deep learning for sensor management, combined tracking, as well as secure communications. Methods for diffusion-based cooperative space object tracking [16] and block chain [17] are emerging as methods for game-theoretical methods.

Acknowledgements

The work was supported under contract FA9453-15-C-0459, FA9453-15-C-0423, and FA8750-19-C-1000. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of AFRL, or the U.S. Government.

Author details

Dan Shen[1], Carolyn Sheaff[2], Genshe Chen[1*], Jingyang Lu[1], Mengqing Guo[1], Erik Blasch[3] and Khanh Pham[4]

1 Intelligent Fusion Technology, Inc., Germantown, MD, USA

2 Information Directorate, Air Force Research Laboratory (AFRL), Rome, NY, USA

3 Air Force Office of Scientific Research (AFOSR), Arlington, VA, USA

4 Space Vehicles Directorate, Air Force Research Laboratory (AFRL), Albuquerque, NM, USA

*Address all correspondence to: gchen@intfusiontech.com

References

[1] Blasch E, Bosse E, Lambert DA. High-Level Information Fusion Management and Systems Design. Norwood, MA: Artech House; 2012

[2] Blasch E. Enhanced air operations using JView for an air-ground fused situation awareness UDOP. In: AIAA/IEEE Digital Avionics Systems Conference, October. 2013

[3] Shen D, Pham K, Blasch E, Chen H, Chen G. Pursuit-evasion orbital game for satellite interception and collision avoidance. Proceedings of the SPIE. 2011;**8044**:89-97

[4] Palatucci M et al. Zero-shot learning with semantic output codes, advances in neural information processing systems. 2009

[5] Mertens JF, Neyman A. Stochastic games. International Journal of Game Theory. 1981;**10**(2):53-66. DOI: 10.1007/BF01769259

[6] Shapley LS. Stochastic games. Proceedings of the National Academy of Sciences of the United States of America. 1953;**39**:1095-1100

[7] Horwood JT, Poore AB. Gauss von mises distribution for improved uncertainty realism in space situational awareness. SIAM/ASA Journal on Uncertainty Quantification. 2014;**2**(1):276-304

[8] Das N, Ghosh RP, Guha N, Bhattacharya R, Mallick B. Optimal transport based tracking of space objects in cylindrical manifolds. The Journal of the Astronautical Sciences. 2019;**66**:582-606

[9] Das N, Deshpande V, Bhattacharya R. Optimal-transport-based tracking of space objects using range data from a single ranging station. Journal of Guidance, Control, and Dynamics. 2019;**42**(6):1237-1249

[10] Shen D, Blasch E, Zulch P, Distasio M, Niu R, Lu J, et al. A joint manifold leaning-based framework for heterogeneous upstream data fusion. Journal of Algorithms and Computational Technology (JACT). 2018;**12**(4):311-332

[11] Jia B, Pham K, Blasch E, Shen D, Wang Z, Chen G. Cooperative space object tracking using space-based optical sensors via consensus-based filters. IEEE Transactions on Aerospace and Electronic System;**52**(4):1908, 2016-1936

[12] Chen G, Shen D, Kwan C, Cruz J, Kruger M, Blasch E. Game theoretic approach to threat prediction and situation awareness. Journal of Advances in Information Fusion. 2007;**2**(1):35-48

[13] Shen D, Jia B, Chen G, Pham K, Blasch E. Game optimal sensor management strategies for tracking elusive space objects. In: Proceedings of IEEE Aerospace Conference. 2017:1-8. DOI: 10.1109/AERO.2017.7943676

[14] Zou H, Lang H, et al. Covert photo classification by deep convolutional neural networks. Machine Vision and Applications. 2017;**28**(5-6):623-634

[15] Available from: https://www.tensorflow.org/

[16] Jia B, Pham K, Blasch E, Shen D, Chen G. Diffusion-based cooperative space object tracking. Optical Engineering. 2019;**58**(4):041607

[17] Xu R, Chen Y, Blasch E, Chen G. An exploration of Blockchain-enabled decentralized capability-based access control strategy for space situation awareness. Optical Engineering. 2019;**58**(4):041609