Aircraft Classification Using radar from small Unmanned Aerial Systems for Scalable Traffic Management Emergency Response Operations

Chester V. Dolph[1], George N. Szatkowski[2], Henry Holbrook[3]
NASA Langley Research Center, Hampton, VA, 23681, USA

Chris M. Morris [4]
Analytical Mechanics Associates, Hampton, VA, 23666, USA

Larry A. Ticatch [5]
Teams3 NASA LaRC, Hampton, VA, 23681, USA

Mahyar R. Malekpour [6] and Robert G. McSwain.[7]
NASA Langley Research Center, Hampton, VA, 23681, USA

This work investigates two machine learning techniques: Support Vector Machine (SVM) and Autoencoders (AE) with SVM layer for classification of radar trajectories as General Aviation (GA), fixed-wing small Unmanned Aerial System (sUAS), or not-an-aircraft using radar data recorded from sUAS. Onboard identification of intruder aircraft type is useful for planning avoidance maneuvers and is necessary to provide autonomous systems to meet or exceed the avoidance capability of a human pilot. Aircraft classification can identify intruder aircraft that are not part of the team and may be violating a Temporary Flight Restriction. Aircraft classification is needed in monitoring an airspace where multiple aircraft are teaming on a shared task. Scalable Traffic Management for Emergency Response Operations (STEReO) is a NASA project aimed at improving disaster response by enabling large scale aircraft operations through the teaming of manned aircraft with sUAS to maximize emergency response resources. To this end, this work uses trajectories and radar derived features to classify aircraft from a multirotor sUAS. The AE + SVM generated the strongest classification overall accuracy of 93.5% using the first 4 seconds of radar track data for tracks that activated the avoidance system. Subsampling the available track data increased the available training data with the maximum aircraft recall of 0.94 achieved using the SVM with 1 second track data.

I. Introduction

The NASA UAS Traffic Management (UTM) project developed an integrated concept leveraging research from air traffic management, aircraft autonomy, and human factors [1]. The NASA Scalable Traffic Management for Emergency Response Operations (STEReO) project uses UTM technologies to improve emergency response for aviation missions during disasters to help save lives and protect property [2]. One challenge for emergency response air operations is identifying and avoiding non-cooperative aircraft, such as privately-owned hobby sUAS, that pose a hazard to first responders and manned aircraft operations. One research thrust of STEReO is to develop an autonomous detection and tracking system onboard an sUAS to assist the emergency response aircraft operations by surveying an airspace for non-cooperative aircraft prior to the arrival of manned emergency aircraft and to continue monitoring the

[1] Aerospace Engineer, Aeronautics Systems Engineering Branch, AIAA Member.
[2] Electrical Engineer, Safety Critical Avionics Branch.
[3] Computer Vision Intern, Aeronautics Systems Engineering Branch, AIAA Member.
[4] Engineer Level IV, Safety Critical Avionics Branch.
[5] Advanced Aircraft Engineer, Electromagnetics Sensors Branch.
[6] Research Aerospace Technologist, Safety-Critical Avionics Systems Branch, AIAA Member.
[7] Aerospace Engineer, Aeronautics Systems Engineering Branch.

airspace for intruder aircraft after the arrival of the emergency responder aircraft. STEReO project plans include evaluating optical-radar aircraft detection systems. This work focuses on radar-based classification.

Under NASA's UTM project, onboard flight testing of the Independent Configurable Architecture for Reliable Operations of Unmanned Systems (ICAROUS) Avoidance system [3] was demonstrated to utilize radar track data to create evasive maneuver trajectories to avoid well clear incursions from other aircraft. This research was conducted under the project: Radar on Autonomous Aircraft to Verify ICAROUS Navigation (RAAVIN) [4]. The RAAVIN flight tests were conducted in coordination between NASA and the Mid-Atlantic Aviation Partnership (MAAP) at the Virginia Tech Kentland Farm test range in Blacksburg, VA to demonstrate the performance of a commercially available sUAS radar to detect and track encroaching targets near the ownship's well clear volume. During the tests, the radar track data was ingested in real time by NASA's ICAROUS algorithm to perform autonomous ownship maneuvering based on radar track data. The radar system adequately detected and tracked the intruder aircraft, however, the abundance of not-an-aircraft (NAA) tracks created numerous unnecessary avoidance maneuvers. To improve the chances that the ICAROUS maneuvers were only performed on radar tracks from the pre-planned intruder flights and not from ground clutter, extensive filtering was applied within the radar to only report tracks meeting a set of complex bounding conditions. Even with the bounding conditions applied to the radar track data, activation of the ICAROUS avoidance system from non-intruder tracks occurred eleven times during the twenty-five aircraft encounters that were analyzed. To improve methods to distinguish between legitimate well clear encroachments from other aircraft rather than from radar clutter, the data was analyzed to establish criteria for feature extraction for the purpose of target type identification. In this paper, a method of aircraft type identification using machine learning is proposed using features extracted from radar using one to five seconds of tracking data. This paper is organized as follows: Methods, Results, Discussion, and Conclusion sections.

II. Methods

This section begins with a description of the radar and flight testing followed by a description of the machine learning approach.

A. Radar and Flight Testing

The airborne RAAVIN research system consisted of the EchoFlight radar operating with firmware 13.0.0, an ANRA mesh radio telemetry link, and a Jetson/Orbitty Tx2 onboard computer running ICAROUS. The research system included an ethernet switch, serial interrupt switch, and power regulators. ICAROUS located on the onboard computer was used to monitor air traffic and executed avoidance maneuvers via the flight controller and the logic defined in Detect and Avoid Alerting Logic for Unmanned Systems (DAIDILUS) [5]. The research system was built and bench validated in the MAAP lab for basic functionality and is shown in Figure 1. The UAS selected to carry the research system, referred to as the "Ownship", was the BFD 1400-SE8-E. The BFD is a heavy-lift multirotor with 8 propulsion motors, maximum takeoff weight of 25 kg, and is capable of carrying a 9 kg payload. The research system alone weighed 2.7 kg while an additional 1.4 kg was required for hardware to adapt the radar and research components to the airframe, resulting in a total payload weight of 4.1 kg. Two relay interrupt switches were included in the payload design and accessible from the pilot's controller to ensure ICAROUS could be disconnected in case the pilot needed to take back control.

The EchoFlight radar used in the RAAVIN flight tests was a commercially available Frequency Modulated Continuous Wave radar with 120° azimuth and 80° elevation field of view. The EchoFlight radar operated at 24.55 GHz center frequency with a 45 MHz swept bandwidth. The radar transmits 2 watts of peak power through a Metamaterial Electronically Scanning Array antenna oriented in horizontal polarization. The radar physical dimensions are 18.7 cm x 12.1 cm x 4.1 cm and it weighs less than 820 grams.

The radar is interfaced through the API via Ethernet transmission control protocol (TCP) and allows bidirectional communication between the radar and Jetson/Orbitty Tx2 onboard computer as shown in Figure 1. Radar detection data are generated every time the radar is able to detect a Doppler target. If the radar detection data meets appropriate criteria established by the Kalman filter settings a radar track is formed. The radar track and detection data packets are recorded in binary files. By default, detection packets are transmitted at a rate of every beam step period, 6.72 milliseconds, while track data packets are received at 9 Hz.

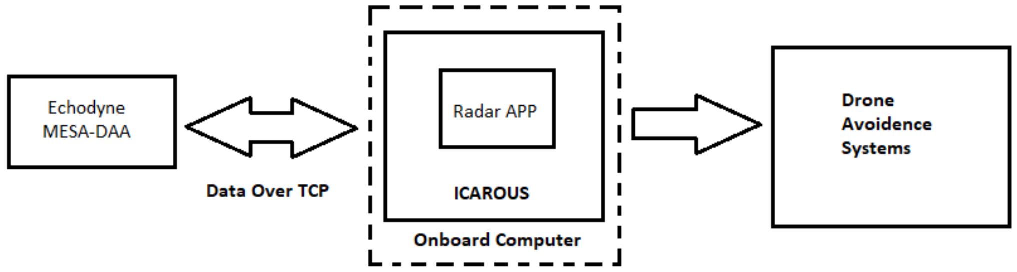


Figure 1 RAAVIN Hardware Configuration

Flight testing took place on 8-1-2019 and 8-2-2019 at the MAAP using the BFD ownship, the intruder GA plane Cessna 172s, and the fixed-wing tempest sUAS shown in Figure 2. During the RAAVIN Phase 2 flight tests, two different flight scenarios were conducted to engage the ICAROUS system, one where the two aircraft were flown head-on and one where they were intercepting perpendicularly. The scenarios were designed to ensure an ICAROUS maneuver while maintaining safe separation between aircraft. The C172 was flown 577ft above the BFD for safety.

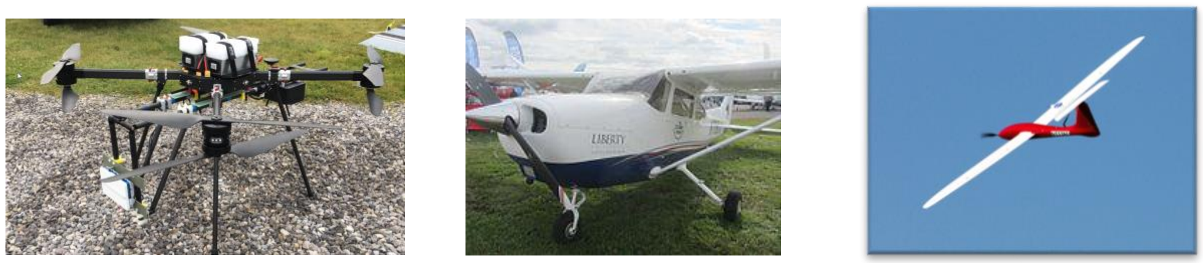


Figure 2 Aircraft: (a) ownship BFD 1400 SE-8 equipped with radar, (b) Intruder GA (Cessna 172S) , (c) Intruder UAS (UASUSA Tempest).

During the tests, the radar's operational parameters were optimized to reduce the number of ICAROUS maneuvers due to non-intruder tracks by applying Boolean logic on a combination of the reported radar data measurement range, velocity and track confidence level values to create bounding conditions on the criteria of when a new track was established. The track confidence level is a measure of how many repeated detections are associated to a specific track. Raising the confidence level to prevent short lived tracks from being reported to ICAROUS greatly reduced the number of potential ICAROUS maneuvers due to non-intruder tracks. The detection points range and velocity provided additional methods to bound which detections would be tracked for ICAROUS reporting. Range masking was used to prevent any tracks from being generated that were closer than about 280 meters from the ownship. This eliminated many of the direct ground clutter targets that would cause an ICAROUS maneuver to be triggered. The velocity of the incoming radar target detection was also effective at reducing potential tracks from being reported. However, filtering tracks based on velocity has ramifications since aliased velocities will also be filtered. The radar scripts were tailored for each specific planned encounter. The same script was not used for all flights. The head-on and crossing flights for the C172 and Tempest all required a unique set of bounding conditions to reduce the chance of ICAROUS being engaged from a non-intruder aircraft radar track. Even with the radar tracking bounding conditions in place for a specific flight encounter, the ICAROUS avoidance system was engaged 11 times from non-intruder tracks during the twenty-five aircraft encounters that were analyzed. A much more sophisticated approach to eliminate non-aircraft tracks will be needed to make autonomous maneuvering vehicles more universally operational.

From the RAAVIN flight tests, a total of 25 runs were included in our analysis. In these runs the intruder aircraft could be clearly identified. Table 1 shows the total number of runs for each scenario and aircraft. The table also shows the number of times ICAROUS maneuvered correctly against an intruder and when it maneuvered against a radar target that was not the intruder. An ICAROUS maneuver can occur multiple times during a flight encounter. A non-intruder track could occur before the actual intruder track is approaching the ownship's well clear volume causing two ICAROUS maneuvers during the flight.

Table 1: Flight Operations Summary

Intruder aircraft	Scenario	Number of Runs	Number of times ICAROUS Activates on intruder Aircraft	Number of times ICAROUS Activates on non-intruder aircraft
Tempest	Head-On	6	4	2
Tempest	Intersecting	4	3	0
C172	Head-On	6	4	4
C172	Intersecting	9	5	5
Total	----------->	25	16	11

The track data is divided into 4 subsets: initial track data from instances where the ICAROUS avoidance system activated shown in Table 2, initial track data using all available tracks shown in Table 3, time interval dataset where the track data is subdivided by the time interval where the ICAROUS avoidance system activated shown in Table 4, and time interval dataset where the track data is subdivided by the time interval using all available tracks shown in Table 5. The initial datasets are useful for determining aircraft classification using the first one to five seconds of available radar track. Early classification of intruder tracks is useful for avoidance maneuvering planning because it may improve safety and conserve onboard resources such as battery or overall mission completion time. The complete datasets utilize all available tracks including ones that did not cause ICAROUS to activate, which is useful for traffic monitoring around the ownship and because machine learning models can benefit from additional data. The time interval datasets utilize all available data by subdividing the tracks by t_{track}.

Table 2: Initial Avoidance Activation Dataset

t_{track}	No. Tempest	No. C172	No. of NAA During Tempest Flight	No. of NAA During C172 Flight
5	8	6	4	12
4	8	6	4	13
3	9	6	6	14
2	9	6	6	15
1	10	6	8	15

Table 3: Initial Complete Dataset

t_{track}	No. Tempest	No. C172	No. of NAA During Tempest Flight	No. of NAA During C172 Flight
5	19	18	18	98
4	19	20	20	113
3	20	20	24	131
2	21	21	30	159
1	22	21	43	189

Table 4: Time Interval Avoidance Activation Dataset

t_{track}	No. Tempest	No. C172	No. NAA During Tempest Flight	No. NAA During C172 Flight
5	28	25	7	50
4	36	32	8	65
3	51	44	13	92
2	77	67	20	140
1	163	136	48	285

Table 5: Time Interval Complete dataset

t_{track}	No. Tempest	No. C172	No. NAA During Tempest Flight	No. NAA During C172 Flight
5	74	80	45	254
4	95	105	57	337
3	132	142	83	473
2	203	219	133	764
1	418	446	295	1631

B. Machine Learning

The machine learning approach is described in detail in the following subsections: features, Support Vector Machine, and Stacked Auto-encoder.

1. *Features*

Twenty-four features are used as the input of SVM or AE + SVM as shown in Table 6. These features capture the movement of the radar track in terms of dynamic behavior and radar return with Radar Cross Section (RCS) over time intervals ranging 1 to 5 seconds. The radar tracker is based on Kalman filtering and updates at a rate of 9Hz. The magnitude of the velocity vector and the estimated radar cross-section were used to generate first order statistical features. The statistical features used in this work are average, maximum, minimum, and variance for one to five seconds of the track data.

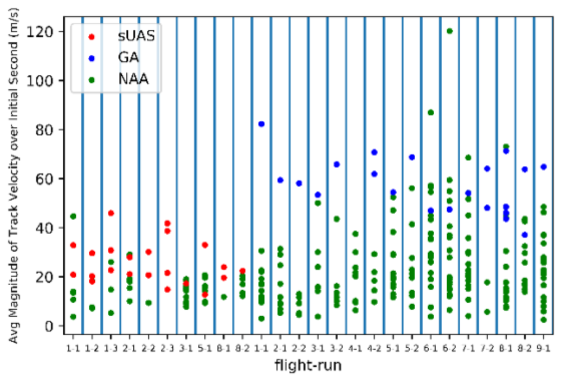
Table 6: Features

Average Velocity for XZ component	Maximum Velocity for XZ component	Minimum Velocity for XZ component	Variance of Velocity for XZ component
Average Velocity for Y component	Maximum Velocity for Y component	Minimum Velocity for Y component	Variance of Velocity for Y component
Magnitude of Average Velocity	Magnitude of Maximum Velocity	Magnitude of Minimum Velocity	Magnitude of Variance of Velocity
Average Velocity	Maximum Velocity	Minimum Velocity	Variance of Velocity
Average Estimated radar Cross Section	Maximum Estimated radar Cross Section	Minimum Estimated radar Cross Section	Variance of Estimated radar Cross Section
No. times the Velocity vector changed signs	No. times the x component of Velocity vector changed signs	No. times the y component of Velocity vector changed signs	No. times the z component of Velocity vector changed signs

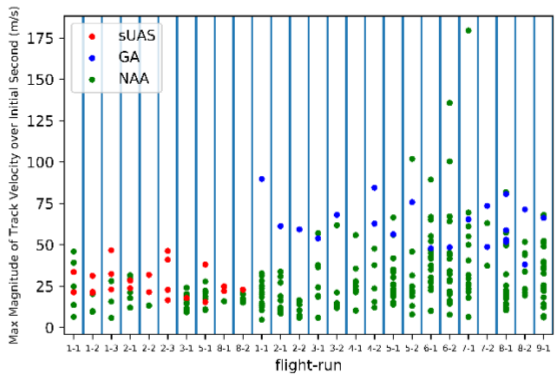
The magnitude of velocity, $\boldsymbol{v_{track}}$, was computed in this way:

$$\|\boldsymbol{v_{track}}\| = \sqrt{v_x^2 + v_y^2 + v_z^2} \tag{1}$$

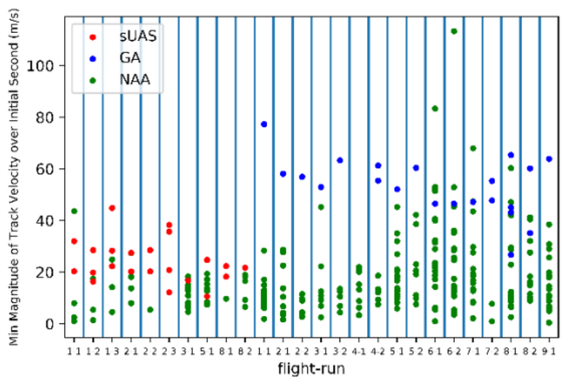
Fig. 3 presents the absolute, average, maximum, and minimum track velocity for a GA, fixed-wing sUAS and “not an aircraft” NAA for flights conducted during the Phase II RAVVIN tests. Each point represents one second of track velocity as computed by Equation 1. The GA frequently has greater velocity than the fixed-wing sUAS and NAA in terms of average, maximum, and minimum as shown in Fig. 3. Discrimination between GA and NAA is not as clear with maximum, minimum, and variance of the magnitude of velocity.



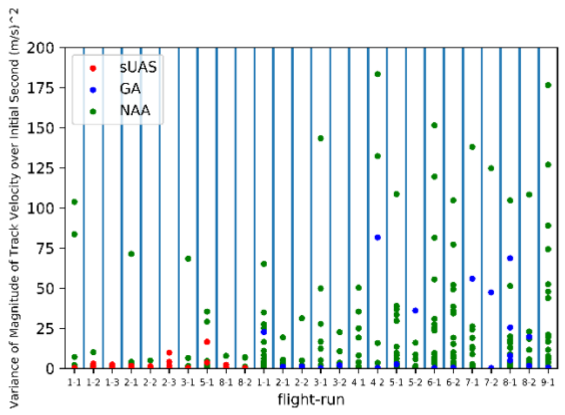
(a)



(b)



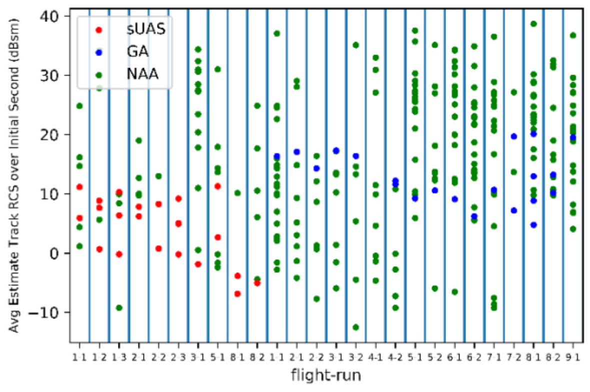
(c)



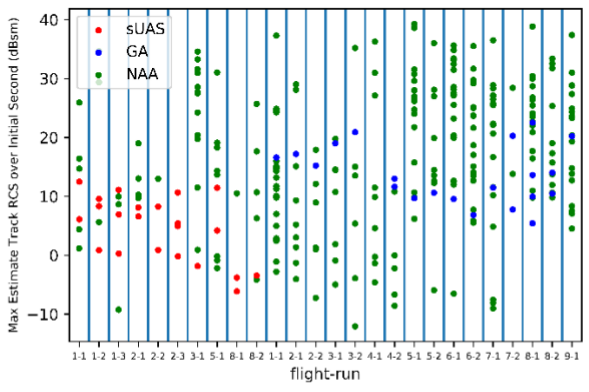
(d)

Fig. 3. Magnitude of Velocity from the Initial Track data: (a) average, (b) maximum, (c) minimum, (d) variance, where red is the tempest, GA is blue, and green is the not-an-aircraft class.

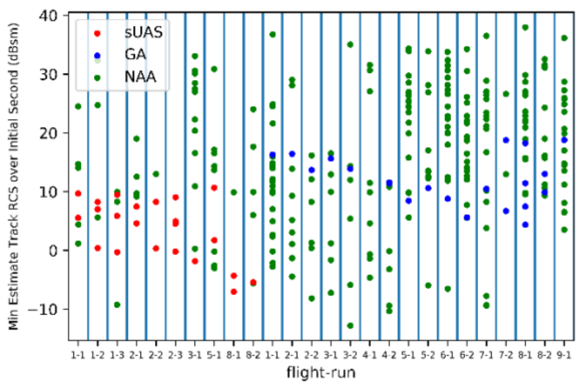
RCS is a measure of the reflected power from the target. Larger aircraft generally will have higher RCS numbers than smaller aircraft. The estimated RCS is calculated based on the received power of the targets, its range distance, antenna gain, and high pass filter losses. The estimated RCS is a non-coherent value at the targets range bin location and is not derived from a known calibration standard such as a metal sphere or cylinder. Figure 4 shows the absolute, average, maximum, and minimum RCS values for the same 25 runs. The Tempest (red dots) and Cessna (blue dots) cluster fairly well within 20 dB. Most aircraft will expect to have RCS signatures which vary within ±10 dB value around a quotient relating to its physical size. If the maximum and minimum RCS values are collected, they can provide an indication of the physical size of the vehicle. The RCS and velocity of the track combined should be relatable features for estimating the class of aircraft.



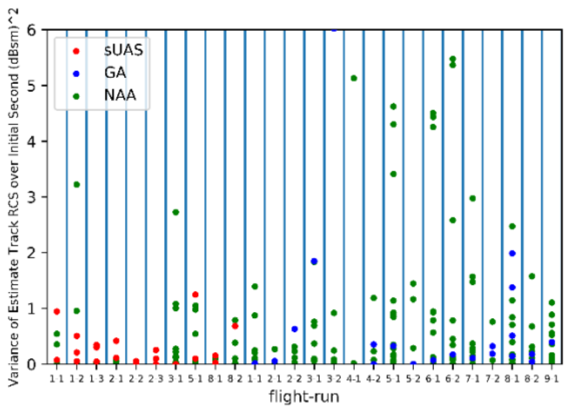
(a)



(b)



(c)



(d)

Fig. 4. Radar cross-section: (a) average, (b) maximum, (c) minimum, (d) variance, where red is the tempest, GA is blue, and green is the not-an-aircraft class. dBsm = decibal square meter.

In addition to the track velocity and RCS, the track confidence level defined in section IIA can provide flight behavior characteristics from the Kalman filter positional variance.

2. *Support Vector Machine*

A Support Vector Machine (SVM) is trained on the features for this work. SVM is a supervised learning technique that extracts patterns from training data and the corresponding labels (e.g. GA, Tempest fixed wing, and NAA). The SVM creates a set of hyper-planes in feature space for classification. A Radial Basis Function was used to extract the pattern during training of the SVM in this work with the regularizer, ***C***, = 1000 and γ is defined:

$$\gamma = \frac{1}{(No.\ features * Variance\ of\ feature\ vector)} \tag{2}$$

The ***c*** parameter trades the penalty of the misclassification with model stability and γ is the hyperparameter that controls the curvature of decision boundary for the SVM. The SVM was implemented per [6] [7] , in Python 3.7 [8] using the scikit-learn and libsvm libraries [9] [10].

The leave-one-out cross classification validation technique is used for evaluation because the dataset is small with 22 GA tracks, 21 Tempest tracks, and 275 NAA tracks. Here the model is trained on every sample except for 1, then that left out sample has its class predicted, and the process continues until each sample has a class prediction.

3. *Auto-encoder*

An Auto-encoder (AE) is used for extraction along with an SVM for classification. The encoder part of the neural network extracts patterns and reduces the dimensionality of the input data to a latent-space representation. The decoder takes information from the compressed latent-space and generates an output. The encode-decode process extracts non-linear patterns from the input features, reduces the impact of features that capture the same patterns, and helps prevent over-fitting of the input-features. The work in [11] showed improvement using SAE over SVM for emotion recognition in speech while the work in [12] showed SVM as having higher performance over SAE in some circumstances for remote sensing image classification.

The autoencoder implemented in this work is a multilayer autoencoder that uses batch normalization, and uses Leaky Rectified Linear Unit (Leaky ReLu) as its activation function as implemented in [13]. This model has two networks: the encoder and the decoder as shown in Figure 5. The encoder takes the original feature vector as input, and outputs a compressed version of that same vector packed with only the relevant information. The decoder takes the compressed vector as input and recreates the input feature vector. The difference between the recreated vector and the input feature vector one is used as the loss function to update both the decoder network to recreate the original vector from the compressed vector and the encoder network to include the information from the compressed vector. After training the autoencoder, the decoder portion of the network is replaced with an SVM as a classification layer.

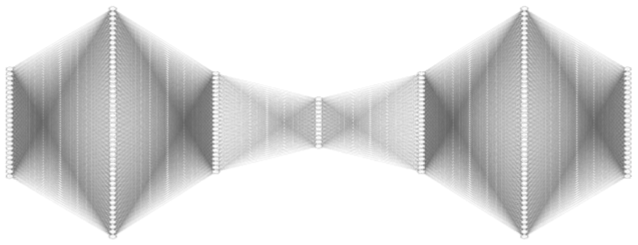


Figure 5: Autoencoder

The encoder has an input layer, two hidden layers, and an output layer (bottleneck layer). These layers have node counts of 24, 48, 24, and 12, respectively, with the 24 nodes of the input layer being the features shown from Table 6, and the 12 nodes of the output layer representing the compressed feature vector. The decoder mirrors this, as it takes the bottleneck layer as input, then has two hidden layers, and an output layer, with node counts of 12, 24, 48, and 24, respectively, with the 24 nodes of the output layer being the recreated feature vector.

Figure 6 shows the learning curves of the autoencoder for the t_{track} =1 from the initial dataset. The graph shows Loss vs Epochs, with Loss being the difference between the input vector and the output vector of the entire autoencoder as a function of the number epochs the model has been trained on. There are two types of loss plotted: training loss and testing (or validation) loss. The training loss is the error between the input and output vector for the training set of data, while testing loss is that same error but on the test set of data. As the number of epochs increase, the loss for both the training and testing set decrease as the model learns to encode the most relevant information within the compressed size of the bottleneck, as well as how to decode that information back into the full size of the input vector.

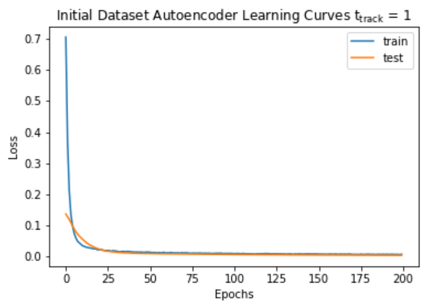


Figure 6: Loss function

C. Evaluation Methodology

The machine learning models are evaluated using leave-one-out accuracy computations and precision-recall analysis compared to the experimental testing as shown in Table 7, where the Tempest aircraft may be replaced by C172 to find the equivalent C172 performance metrics. The leave-one-out accuracy captures overall machine learning performance across Tempest, C171, and NAA trajectories. A class break-down accuracy is computed for leave-one-out analysis as this provides insight into how the model performs against specific aircraft or NAA. There is bias for overall leave-one-out metric because there is substantially more NAA trajectories over Tempest and C171 trajectories. However, the overall accuracy is useful for understanding how the model performs on the experimentally collected data. In addition to accuracy, precision and recall are computed for the experimental activations of ICAROUS and the machine learning model classification. The experimental precision and recall captures system performance of the radar and the onboard collision avoidance system.

Table 7: Performance Metrics

Performance Measure or Metric	Definition
Overall leave-one-out accuracy	Computed by training classifier on all but one trajectory, testing on left-out trajectory, and then iterating through the entire dataset so that each trajectory becomes a test. The number of correctly classified trajectories is then divided by the total number of trajectories in the dataset.
Class leave-one-out accuracy (Tempest, C172, or NAA)	Class accuracy using leave-one-out method of training on entire dataset except for the left-out trajectory.
Experimental Precision	$\frac{No.ICAROUS\ Aicraft\ Activations}{No.\ ICAROUS\ Aicraft\ Activations\ + No.NAA\ activations}$
Experimental Recall	$\frac{No.ICAROUS\ Aicraft\ Activations}{No.ICAROUS\ Aicraft\ Activations\ + No.missed\ Aircraft\ Activations}$
Tempest* Experimental Precision	$\frac{No.ICAROUS\ Tempest\ Activations}{No.\ ICAROUS\ Tempest\ Activations\ + No.NAA\ activations\ from\ Tempest\ flights}$
Tempest* Experimental Recall	$\frac{No.Tempest\ Aicraft\ Activations}{No.ICAROUS\ Tempest\ Activations\ + No.missed\ Tempest\ Activations}$
Machine Learning Model Precision	$\frac{No.Aicraft\ Correctly\ Classified}{No.Aircraft\ Correctly\ Classified + No.NAA\ Incorrectly\ Classified\ as\ Aircraft}$
Machine Learning Model Recall	$\frac{No.Aicraft\ Correctly\ Classified}{No.Aircraft\ Correctly\ Classified + No.Aircraft\ incorrectly\ classifed\ as\ NAA}$
Machine Learning Model Tempest* Precision	$\frac{No.Aicraft\ Correctly\ Classified}{No.Aircraft\ Correctly\ Classified + No.NAA\ Incorrectly\ Classified\ as\ Aircraft}$
Machine Learning Model Tempest* Recall	$\frac{No.Tempest\ Correctly\ Classified}{No.Tempest\ Correctly\ Classified + No.Tempest\ Incorrectly\ classifed\ as\ NAA}$

***** Tempest is replaced by C172 to find the C172 equivalent performance metrics

III. Results

This section presents the results for the SVM and AE + SVM first by the initial dataset followed by the time interval dataset.

A. Initial Dataset

Tables 8 through 11 show the initial dataset results. Overall accuracy for SVM and AE+SVM is strong with ~80% overall leave-one-out accuracy for t_{track} values of 1 through 5 seconds as shown in Table 8. The strongest overall classification accuracy is 85.5% with AE + SVM using 4 seconds of track data. The strongest class accuracies are: 89.5% for the Tempest with the AE + SVM using 4 seconds of track data, 66.7% for the C172 with AE + SVM using 5 seconds of data, and the SVM generated the strongest NAA accuracy using 3 seconds of data. The difference between SVM and AE+ SVM for the strongest respective overall leave-one-out accuracy is ~5% accuracy.

Table 8: Total Initial Dataset Acc.

	SVM				AE + SVM			
t_{track}	Overall Leave-one-out Acc.	Leave-one-out Tempest Accuracy	Leave-one-out C172 Accuracy	Leave-one-out NAA Accuracy	Leave-one-out Accuracy	Leave-one-out Tempest Accuracy	Leave-one-out C172 Accuracy	Leave-one-out NAA Accuracy
5	0.824	0.842	0.611	0.853	0.797	0.789	0.667	0.819
4	0.820	0.737	0.600	0.865	0.855	0.895	0.650	0.880
3	0.831	0.700	0.500	0.890	0.795	0.650	0.600	0.839
2	0.835	0.667	0.571	0.884	0.771	0.714	0.476	0.810
1	0.844	0.773	0.667	0.866	0.756	0.818	0.524	0.772

The AE+SVM performed the strongest for the Activation Dataset with accuracy of 93.5% for 4 seconds of track data as shown in Table 9. The AE+SVM generated the strongest results for the Tempest and C172 accuracies at 4 to 5 seconds. The SVM tended to have the strongest performance for NAA for the total initial and initial activation datasets.

Table 9: Initial Activation Dataset Acc.

	SVM				AE + SVM			
t_{track}	Overall Leave-one-out Accuracy	Leave-one-out Tempest Accuracy	Leave-one-out C172 Accuracy	Leave-one-out NAA Accuracy	Overall Leave-one-out Accuracy	Leave-one-out Tempest Accuracy	Leave-one-out C172 Accuracy	Leave-one-out NAA Accuracy
5	0.833	1.000	0.833	0.750	0.800	0.875	0.833	0.750
4	0.871	0.875	0.833	0.882	0.935	1.000	1.000	0.882
3	0.829	0.778	0.833	0.850	0.771	0.667	0.833	0.800
2	0.889	0.778	1.000	0.905	0.778	0.778	1.000	0.714
1	0.846	0.900	1.000	0.783	0.795	0.900	1.000	0.696

The experimental precision recall is shown in Table 10 for the initial dataset. Overall precision is weak ~ .45 while overall recall is strong at 1 indicating that the avoidance system always activated to the intruder aircraft, however, the system reacted to many NAA and executed avoidance maneuvers to tracks that were not hazards.

Table 10: Total Initial Dataset Experimental Precision-Recall

t_{track}	Overall Precision	Overall Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall
5	0.467	1	0.667	1	0.333	1
4	0.452	1	0.667	1	0.316	1
3	0.429	1	0.600	1	0.300	1
2	0.417	1	0.600	1	0.286	1
1	0.410	1	0.556	1	0.286	1

The SVM and AE + SVM overall precision in Table 11 typically exceeded the overall experimental precision values, however, the machine learning overall recalls ranged to 0.83 to 0.61.

Table 11: Total Initial Dataset Machine Learning Precision-Recall

	SVM						AE + SVM					
t_{track}	Overall Precision	Overall Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall	Overall Precision	Overall Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall
5	0.614	0.771	0.516	0.842	0.846	0.688	0.563	0.771	0.455	0.789	0.800	0.750
4	0.591	0.722	0.467	0.778	0.857	0.667	0.652	0.833	0.567	0.895	0.813	0.765
3	0.585	0.686	0.500	0.778	0.769	0.588	0.500	0.676	0.382	0.722	0.750	0.632
2	0.542	0.667	0.500	0.737	0.600	0.600	0.410	0.610	0.357	0.750	0.526	0.476
1	0.500	0.756	0.472	0.810	0.538	0.700	0.354	0.725	0.305	0.857	0.478	0.579

B. Time Interval Dataset

The SVM and AE + SVM achieved higher leave-one-out accuracy along with the individual class accuracies when training on the time interval dataset relative to the results from the initial dataset as shown in Table 12.

Table 12: Total Time Interval Dataset Acc.

	SVM				AE + SVM			
t_{track}	Overall Leave-one-out Acc.	Leave-one-out Tempest Acc.	Leave-one-out C172 Acc.	Leave-one-out NAA Acc.	Overall Leave-one-out Acc.	Leave-one-out Tempest Acc.	Leave-one-out C172 Acc.	Leave-one-out NAA Acc.
5	0.890	0.878	0.850	0.903	0.890	0.865	0.863	0.903
4	0.894	0.916	0.867	0.896	0.887	0.884	0.857	0.896
3	0.918	0.955	0.887	0.917	0.899	0.917	0.873	0.901
2	0.906	0.926	0.877	0.909	0.908	0.941	0.886	0.906
1	0.902	0.940	0.917	0.890	0.901	0.926	0.917	0.891

The SVM and AE + SVM performed similarly with the time interval activation dataset as shown in Table 13, and outperformed the accuracies reported by the initial dataset.

Table 13: Time Interval Activation Dataset Acc.

	SVM				AE + SVM			
t_{track}	Overall Leave-one-out Acc.	Leave-one-out Tempest Acc.	Leave-one-out C172 Acc.	Leave-one-out NAA Acc.	Overall Leave-one-out Acc.	Leave-one-out Tempest Acc.	Leave-one-out C172 Acc.	Leave-one-out NAA Acc.
5	0.909	0.893	1.000	0.877	0.900	0.929	0.960	0.860
4	0.887	0.972	0.969	0.808	0.879	0.944	0.969	0.808
3	0.915	0.980	0.977	0.857	0.890	0.902	0.977	0.848
2	0.908	0.948	0.970	0.863	0.891	0.948	0.970	0.831
1	0.880	0.926	0.978	0.817	0.883	0.933	0.978	0.820

Permutating the track data for the experimental precision recall shown in Table 14 yielded similar results to the initial datasets.

Table 14: Time Interval Experimental Precision Recall

t_{track}	Precision	Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall
5	0.482	1.000	0.800	1.000	0.333	1.000
4	0.482	1.000	0.818	1.000	0.330	1.000
3	0.475	1.000	0.797	1.000	0.324	1.000
2	0.474	1.000	0.794	1.000	0.324	1.000
1	0.473	1.000	0.773	1.000	0.323	1.000

The time interval precision and recall for the machine learning models shown in Table 15 are higher than the initial dataset.

Table 15: x

	SVM						AE + SVM					
t_{track}	Overall Precision	Overall Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall	Overall Precision	Overall Model Recall	Tempest Precision	Tempest Recall	C172 Precision	C172 Recall
5	0.821	0.881	0.739	0.890	0.919	0.872	0.821	0.875	0.753	0.877	0.896	0.873
4	0.813	0.899	0.750	0.916	0.883	0.883	0.809	0.883	0.757	0.894	0.865	0.874
3	0.846	0.926	0.808	0.955	0.887	0.900	0.817	0.901	0.766	0.924	0.873	0.879
2	0.823	0.916	0.780	0.931	0.869	0.901	0.821	0.923	0.776	0.946	0.870	0.902
1	0.791	0.940	0.730	0.945	0.859	0.936	0.792	0.932	0.726	0.930	0.867	0.934

IV. Discussion

The overall-classification accuracy results shows that it is feasible to classify a radar trajectory with limited data of ~1 second given the overall leave-one-out accuracies for initial and time interval datasets of 84.4% and 90.2% for SVM, and 75.6% and 90.1% for AE+SVM respectively. The lower ~10% AE+SVM overall classification accuracy relative to SVM for the initial dataset may be related to the limited training data of 250 samples for the neural networks, which benefit from larger datasets to extract patterns. To address limited data problem from the initial dataset, the time interval dataset aims to extract more training and testing data. There is bias in the time interval dataset because the same radar track is being sampled at different time intervals and that is not the same as acquiring trajectories from new aircraft or encounters. However, the time interval dataset maximizes the utility of the available data because the trajectories are dynamic through time, and this provides an opportunity to train neural networks using this data. The tempest classification accuracy results tended to be higher than the C172 classification accuracies. One possible explanation may be that RCS values from ground clutter resemble the RCS values from C172. Future work will utilize INS to correct for the attitude for radar to suppress features from ground data.

For the activation dataset, the time interval results were generally higher than initial dataset overall classification accuracies. This may be explained by additional training data for extracting patterns for the machine learning models. The 93.5% classification accuracy using 4 seconds for the SVM + AE for the initial dataset is promising discriminating different intruder types using onboard radar. The overall classification accuracies tended to be in the 80-90% accuracy ranges. The overall classification accuracy for the one second in the initial data of 84.6% and 79.5%, which would be useful for planning avoidance maneuvers and communicating relative intruder aircraft position.

The SVM and AE + SVM outperformed the experimental precision, however, recall dropped from experimental value 1 to a maximum of 0.940. The precision-recall analysis revealed that precision doubled for the SVM and AE + SVM for the time interval dataset over the experimental results. Avoidance applications need to balance over-reacting to potential intruders and ensuring the avoidance system meets adequate safety performance standards. As the minimum operational performance standards are evolving through government-industry-academic partnerships, the results and analysis herein are useful for understanding how machine learning results may be used to understand radar trajectory data. Future work will expand current radar dataset to include more diverse collision geometries and aircraft as this will enable machine learning models to extract more comprehensive patterns of actual flight geometries. Additionally, the Kalman filter model parameters will be used as features as the motion models provide insight of the character of the radar trajectory. Integrating an INS to correct for changes attitude of the radar will improve classification and precision-recall performance by eliminating radar detections from the ground and improving the accuracy of the velocity-based features.

V. Conclusion

The overall classification accuracies of approximately 90% show that it feasible to use SVM and AE + SVM to discriminate between GA, fixed wing sUAS, and NAA trajectories from an aerial multirotor sUAS. The maximum overall recall of 0.940 using 1 second of track data for the time interval dataset indicates that the SVM model is correctly identifying the aircraft at high rate relative to misclassifying aircraft as NAA. Typically, it is better to activate a safety critical system occasionally for non-intruder aircraft provided it captures all the true intruder aircraft. This balance of capturing intruder aircraft vs not-an-aircraft would likely improve with a larger training dataset as machine learning models benefit from more data for pattern recognition.

Acknowledgments

The authors would like to acknowledge Lou Glaab (NASA) for managing the RAAVIN program, Dave Bradley (NIA) for piloting sUAS, Dave Hare (NASA) for operating ground control station, Kyle Smalling (NIA), and Swee Balachandran (NIA) for their help integrating ICAROUS onto the vehicle and support during flight testing, and Andrew Kriz (Virginia Tech) and John Coggin (Virginia Tech) for his support at MAAP.

References

[1] P. Kopadekar, J. Rios, T. Prevot, M. Johnson, J. Jung and J. Robinson, "Unmanned Aircraft System Traffic Management (UTM)," in *AIAA Aviation*, Washington DC, 2016.

[2] NASA, "Airspace Operations Lab," [Online]. Available: https://humansystems.arc.nasa.gov/groups/AOL/research/stereo.php. [Accessed 11 Nov. 2020].

[3] S. Balachandran, C. Munoz, M. Consiglio, M. Feliu and A. Patel, "Independent Configurable Architecture for Reliable Operation of Unmanned Systems with Distributed Onboard Services," in *38 IEEE/AIAA 37th Digital Avionics Systems Conference* , London, UK, 2018.

[4] G. N. Szatkowski, A. Kriz, L. A. Ticatch, R. B. J. Coggin and a. C. M. Morris, "Airborne Radar for sUAS Sense and Avoid," in *IEEE/AIAA 38th Digital Avionics Systems Conference*, San DIego, 2019.

[5] C. Munoz, A. Narkawicz, G. Hagen, J. Upchurch, A. Dutle and M. Consiglio, " DAIDALUS: Detect and Avoid Alerting Logic for Unmanned Systems," in *Proceedings of the 34th Digital Avionics Systems Conference* , Prague, 2015.

[6] I. Guyon, G. Boser and V. V., "Automatic Capacity Tuning of Very Large VC-dimension Classifiers," in *Advances in neural information processing* , Denver, 1993.

[7] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, pp. 273-297, 1995.

[8] P. S. Foundation, "Python 3.7.0," [Online]. Available: https://www.python.org/downloads/release/python-370/.

[9] F. Pedregosa, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825--2830, 2011.

[10] C.-J. L. Chih-Chung Chang, "LIBSVM : a library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1--27:27, 2011.

[11] H. Aouani and Y. Ayed, "Emotion Recognition in Speech Using MFCC with SVM, DSVM and Auto-encoder," in *th International Conference on Advanced Technologies for Signal and Image Processing (ATSIP)*, Sousse, 2018.

[12] P. Liu, K. Choo, L. Wang and F. Huang, "SVM or deep learning? A comparative study on remote sensing," *Soft Computing*, vol. 21, p. 7053–7065, 2017.

[13] J. Brownlee, "Autoencoder Feature Extraction for Classification," 7 Dec 2020. [Online]. Available: https://machinelearningmastery.com/autoencoder-for-classification/.

[14] P. Molchanov, K. Egiazarian, J. Astola, R. Harmanny and J. de Wit, "Classification of small UAVs and birds by micro-Doppler signatures," in *Proceedings of the 10th European Radar Conference*, Nuremberg, 2013.