CLASSIFICATION OF RESIDENT SPACE OBJECTS BY SHAPE AND SPIN MOTION USING NEURAL NETWORKS AND PHOTOMETRIC LIGHT CURVES

Katiyayni Balachandran*and Kamesh Subbarao†

Department of Mechanical and Aerospace Engineering, The University of Texas at Arlington, Arlington, TX, 76019, USA

ABSTRACT

Accurate identification of Resident Space Objects (RSOs) through ground-based photometric light curve measurements can aid in characterizing collision-prone objects to orbital assets, planning for commercial use of space, and possibly understand the intent of foreign objects. Given an unresolved object's light curve, in low earth orbit (LEO), we can characterize it by shape and spin rate using a hybrid approach of a recurrent neural network and a hidden Markov model. This integrated model has been developed to identify tumbling and stabilized objects by testing for aliasing, periodicity, and feature extraction. The performance of this approach for RSO classification is demonstrated via simulations. The model is designed to train and validate using synthetic light curve measurements of objects such as an ellipsoidal asteroid, rocket upper-stage, CubeSat, torus, probe, and box-wing satellite, among others. The fidelity of these simulations ensures it is both realistic and accurate to within 10% error, and the computation speed is faster with flexibility to deal with a larger amount of data.

Keywords: RSO; classification; neural networks; light curve.

1. INTRODUCTION

In recent years, the number of active and defunct Resident Space Objects (RSOs) have been increasing. Additional artificial space objects orbiting Earth is becoming an issue for space situational awareness (SSA). Accurate tracking and identification of RSOs will eliminate potential threats and enable a better understanding of the object's intent. Typically, optical sensors are utilized to collect astrometric and photometric measurements of existing space objects. The light detected in these sensors is significant to determining characteristics of the object. The brightness (flux of photons) measurements reflected by an RSO and seen by the observer as a function of time is defined as a photometric light curve.

Optical measurements for space object tracking are sensitive to shape, attitude, angular velocity, and surface parameters [11]. Current state-of-the-art in RSO characterization relies heavily on nonlinear state estimation theory [15] , multiple models [22], and full light curve inversion (LCI) which is computationally expensive [11]. Additionally, physical models are used in LCI [17, 18] and to estimate the model parameters, *a priori* information is needed. A data-driven approach for improved accuracy with a large volume of objects employs the use of deep neural networks.

As the number of artificial space objects increase, the availability of astrometric and photometric data obtained commercially along with high cadence of observations makes it difficult to sustain continuous and real time analysis [6]. The state of a satellite cannot be assumed constant and therefore requires machine classification if near real time assessment of a larger number of objects is desired. Machine learning techniques have been used to analyze variable stars [19], however there isn't a big data problem with artificial space objects unlike stellar objects. As deep space becomes increasingly congested and anomalous events are common, it is essential that the SSA community has the tools to reduce information latency [6]. The requirement of timeliness and the consideration of a non-stationary state are unique to man-made objects. As such, we are considering both active, stabilized objects as well as inactive, tumbling objects.

Time series classification is a supervised machine learning problem aimed for labeling multivariate series of variable length [3]. Time series data often have a very high dimensionality and thus, a broad number of features can be extracted making classification arduous [21, 9]. Instead of applying classification methods on raw time series data, as it's not very practical, we can employ a higher-level representation [8]. One such type of higher-level representation is window-based representation. In window-based representation, the whole time series is divided into a sequence of equal sized windows (segments). One or more features are extracted from each frame, and a vector of these features becomes the data-reduced representation. Subsequently, the created vectors are used to train a classifier. A few commonly used classifiers are listed as follows: Rule Induction, Support Vector Machine (SVM), Neural Network (NN). When using any of the above-mentioned classifiers, the temporal relations

*Graduate Student, Email:katiyayni.balachandran@mavs.uta.edu

†Professor, Fellow of American Astronautical Society, Email: subbarao@uta.edu

Proc. 8th European Conference on Space Debris (virtual), Darmstadt, Germany, 20–23 April 2021, published by the ESA Space Debris Office
Ed. T. Flohrer, S. Lemmens & F. Schmitz, (http://conference.sdo.esoc.esa.int, May 2021)

present in time series data are ignored, leading to inaccurate results. To solve this issue, multiple classifiers can be utilized to improve the accuracy of time series classifiers. The viability of using multiple classifiers are evidenced in [24, 31] and have been found better performing than those of single classifier systems. Adding a Hidden Markov Model (HMM) to a Recurrent Neural Network (RNN) is proposed for this study. This hybrid approach improves the accuracy of time series classification by executing a secondary screening accounting for the temporality in the data. Redundancy in the classifiers from multiple layers of training and prediction can be eliminated.

This paper is organized as follows. First, the acquisition technique for light curves is briefly explained, followed by the mathematical background for the components of the hybrid model. Details of the classification approach are included with the aggregation of Long-Short Term Memory (LSTM) RNN to the HMM. Identifiers (labels) by which classification of light curves is performed as well as the tests to ensure accuracy are then explained. Simulation results for the proposed method applied to a set of synthetic light curves of artificial objects in LEO are shown, and future improvements will conclude this work.

2. LIGHT CURVE ACQUISITION

Consider a reference frame attached to the object, with respect to whom the illuminating source (Sun) and the observer viewing on Earth are described. The direction unit vectors to the source and observer from the object are $\boldsymbol{\omega}$ and $\boldsymbol{\omega_0}$ respectively. The unit normal vector of a facet, as depicted in Fig. 1 (modified image from [27]), is defined as $\boldsymbol{\eta}$ (specifically $\boldsymbol{\eta}(\theta, \psi)$ where θ is measured from the pole - see Fig. 1b).

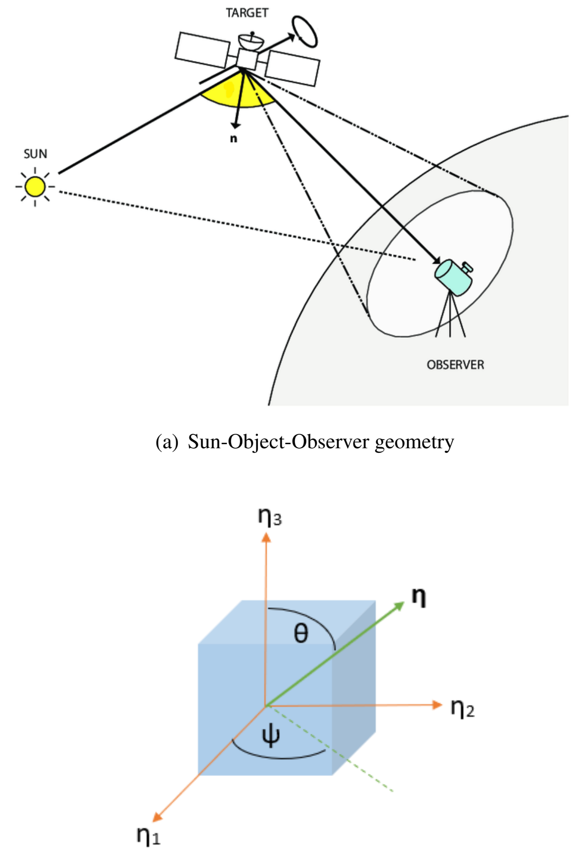
$$\begin{aligned} \eta_1 &= \sin\theta\cos\psi & (1) \\ \eta_2 &= \sin\theta\sin\psi & (2) \\ \eta_3 &= \cos\theta & (3) \end{aligned}$$

The visible, illuminated section of the surface is given by $\boldsymbol{A}$. The integrated total brightness of the target object (made up of a finite number of facets) as seen by the observer at a given time is

$$L(\boldsymbol{\omega_0}, \boldsymbol{\omega}) = \int_{\boldsymbol{A}} S(\mu, \mu_0, \alpha) G(\theta, \psi) \sin\theta d\theta d\psi \qquad (4)$$

and the scattering function $S(\mu, \mu_0, \alpha)$ is dependent on the viewing geometry where

$$\begin{aligned} \mu &= \boldsymbol{\omega} \cdot \boldsymbol{\eta} & (5) \\ \mu_0 &= \boldsymbol{\omega_0} \cdot \boldsymbol{\eta} & (6) \\ \cos\alpha &= \boldsymbol{\omega_0} \cdot \boldsymbol{\omega} & (7) \end{aligned}$$



(a) Sun-Object-Observer geometry

(b) Object's normal direction vector and astrometric angles

Figure 1. Simplified geometry involved in light curve acquisition

The scattering function or bidirectional reflectance distribution function (BRDF) uses the area of the facet and α is the solar phase angle. The BRDF model describes light reflection off a surface which comprises the shape's reflectance properties. There are different models to describe the scattering function (i.e. Hapke model) and a simplified model [23] is depicted as

$$S(\mu, \mu_0, \alpha) = f(\alpha) \left(\frac{\mu\mu_0}{\mu + \mu_0} + C\mu\mu_0 \right) \qquad (8)$$

where $f(\alpha)$ is the phase function and C is a relative weighting factor. The illuminated surface, $\boldsymbol{A}$, encompasses points on the surface where $\mu, \mu_0 \geq 0$. The surface curvature function $G(\theta, \psi) \geq 0$ is denoted by $G(\theta, \psi) = \dfrac{J(\theta, \psi)}{\sin\theta}$ where $J = |\mathbf{J}|$ is the norm of the Jacobian vector $\mathbf{J}(\theta, \psi) = \dfrac{\partial \mathbf{x}}{\partial \theta} \times \dfrac{\partial \mathbf{x}}{\partial \psi}$ and $\mathbf{x}(\theta, \psi)$ gives the surface as a function of the surface normal direction.

To numerically compute the integrated total brightness (equation 4), a more analytical approach of using the Lebedev quadrature is formulated and presented in [1] and is chosen for the purpose of generating synthetic light curves of artificial objects to populate the training and test datasets.

3. BACKGROUND ON MODEL COMPONENTS

The following section presents the details of the hybrid model's components, which include the recurrent neural network (RNN) and the hidden Markov model (HMM).

3.1. Recurrent Neural Network (RNN)

Of the three important types of neural networks, RNN has a recurrent connection on the hidden state, which allows for feedback. The outputs of previous time steps will be considered as inputs in the current time step, and the results of the current time step will impact the calculation of the next time step. This looping constraint ensures that sequential information is captured in the input data.

Given $\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}$ are the input vectors, $\boldsymbol{h_1}, \boldsymbol{h_2}, \ldots, \boldsymbol{h_n}$ are the hidden cell vectors and $\boldsymbol{y_1}, \boldsymbol{y_2}, \ldots, \boldsymbol{y_n}$ are the output vectors, where n represents the total number of steps, the equations computing the resulting output vectors [28] are defined below:

$$\begin{aligned} \boldsymbol{h}_t &= \theta \phi_{(\boldsymbol{h}_{t-1})} + \theta_x \boldsymbol{x}_t \\ \boldsymbol{y}_t &= \theta_y \phi_{(\boldsymbol{h}_t)} \end{aligned}$$

where $\theta, \theta_x, \theta_y$ are weights, ϕ is the activation function ($\tanh$ in most RNNs). The self-connection weight θ is simply initialized as 1. The subsequent back-propagation will adjust all the weights in every iteration.

RNNs share the parameters across different time steps. This results in fewer parameters to train and decreases the computational cost. Deep RNNs suffer from the vanishing and exploding gradient problem.

A comparison of the characteristics of the artificial, recurrent and convolutional neural networks (ANN, RNN and CNN) are summarized in Table 1. Although all three have a common disadvantage of vanishing and exploding gradient, RNN is the optimal choice for time-series data with the recurrent connections and less computational burden from parameter sharing.

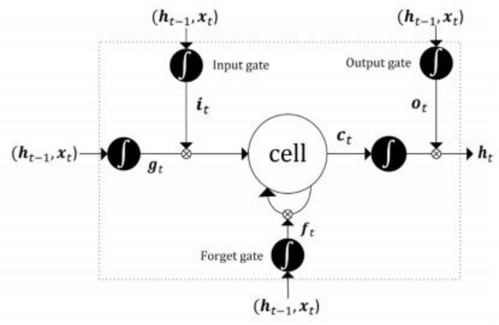
Table 1. Attributes of ANN vs. CNN vs. RNN

Attribute	ANN	RNN	CNN
Data compatibility	-	Yes	Yes
Recurrent connections	-	Yes	-
Parameter sharing	-	Yes	Yes
Spatial relationship	-	-	Yes
Vanishing & Exploding gradient	Yes	Yes	Yes

Of the numerous types of RNNs, Long Short-term Memory (LSTM) has been chosen for this application and the logistics and equations are detailed in the following section.

3.1.1. Long Short-term Memory (LSTM)

Long Short-term Memory RNN is specifically designed to avoid the long-term dependency problem [12] and there are many slight variations [16]. We will use the definition from [13] in this context. As seen in Fig. 2, a cell of LSTM RNN has three gates, which control the involvement of the past context information: input gate, output gate, and forget gate. The latter is used to scale the influence of the previous cell on the current cell state.



$(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$
$(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$
Input gate
Output gate
$\boldsymbol{i}_t$
$\boldsymbol{o}_t$
$(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$
$\boldsymbol{g}_t$
cell
$\boldsymbol{c}_t$
$\boldsymbol{h}_t$
$\boldsymbol{f}_t$
Forget gate
$(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$

Figure 2. LSTM cell structure [28]

The equations for computing the gate outputs [28] are defined below:

$$\begin{gathered} \boldsymbol{i}_t = \sigma\left(\theta_{xi} \boldsymbol{x}_t + \theta_{hi} \boldsymbol{h}_{t-1} + b_{\mathrm{i}}\right) \\ \boldsymbol{f}_t = \sigma\left(\theta_{xf} \boldsymbol{x}_t + \theta_{hf} \boldsymbol{h}_{t-1} + b_{\mathrm{f}}\right) \\ \boldsymbol{o}_t = \sigma\left(\theta_{xo} \boldsymbol{x}_t + \theta_{ho} \boldsymbol{h}_{t-1} + b_{\mathrm{o}}\right) \\ \boldsymbol{g}_t = \tanh\left(\theta_{xg} \boldsymbol{x}_t + \theta_{hg} \boldsymbol{h}_{t-1} + b_{\mathrm{g}}\right) \\ \boldsymbol{c}_t = \boldsymbol{f}_t \cdot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \cdot \boldsymbol{g}_t \\ \boldsymbol{h}_t = \boldsymbol{o}_t \cdot \tanh\left(\boldsymbol{c}_t\right) \end{gathered} \tag{9}$$

where $\boldsymbol{h}_{t-1}$ is the output of the last time step, $\boldsymbol{x}_t$ is the cell input at the current step, and $\boldsymbol{h}_t$ is the cell output. The t in the subscripts represents the current step number. The $\boldsymbol{i}, \boldsymbol{f}, \boldsymbol{o}$, and $\boldsymbol{g}$ respectively denote the output vectors of input gate, forget gate, output gate and the cell itself. θ are the weights. For example, θ_{xi} is the weight between the input vector $\boldsymbol{x}_t$ and the input gate vector $\boldsymbol{i}_t$, whereas θ_{hi} is the weight between the output vector $\boldsymbol{h}_{t-1}$ and the gate vector $\boldsymbol{i}_t$. b represents biases, $\boldsymbol{c}_t$ and $\boldsymbol{c}_{t-1}$ are the cell outputs in the current step and the previous step, respectively, and σ represents a sigmoid function (output 0 to 1). LSTM usually limits the activation function to $\tanh$ (output -1 to 1) for $\boldsymbol{g}_t$ and $\boldsymbol{h}_t$, and sigmoid for $\boldsymbol{i}_t$, $\boldsymbol{f}_t$, and $\boldsymbol{o}_t$. Other activation functions, like rectified linear unit (ReLU), make LSTM diverge [5], hence, the sigmoid and tanh are suitable choices.

Generally, the RNN cannot retrieve data deep into its memory and LSTM RNN helps mitigate that problem as shown in [14]. LSTM RNN has become a popular choice for modeling inherently dynamic processes [28] and is expected to result in a high accuracy.

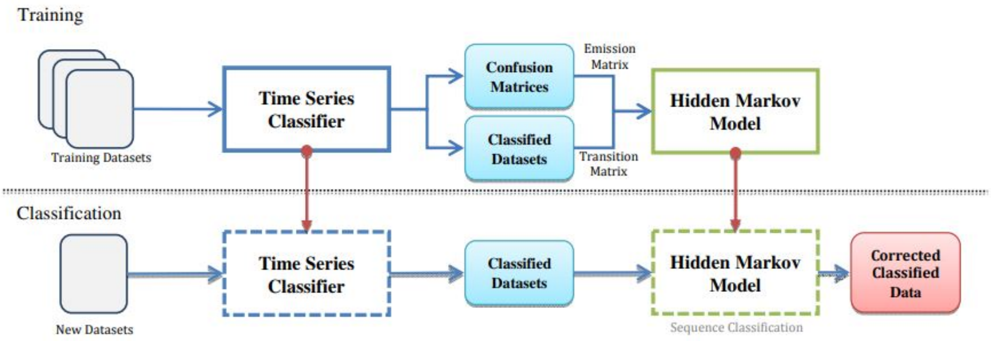


Figure 3. General Framework with two phases: Training and Classification

3.2. Hidden Markov Model (HMM)

A Markov chain satisfies the Markov property (Eq. 10) where the next state depends only on the current state and not on the sequence of states that preceded it [25]. The model consists of a finite number of states $\{s_1, s_2, \ldots, s_n\}$ and some known probabilities $\boldsymbol{P} = \{p_{ij}\}$ where p_{ij} is the probability of moving from state s_i to the state s_j [8].

$$P\left(s_{t+1} \mid s_0, s_1, \ldots, s_t\right) = P\left(s_{t+1} \mid s_t\right) \tag{10}$$

Hidden Markov model (HMM) is a Markov chain in which the observations, dependent on the state, are visible instead of states. Each state has a probability distribution over the possible output observations. The sequence of observations is linked to its corresponding sequence of states. In a HMM, the observer does not know which state the system is in, but only a probabilistic insight on where it should be. It is defined by equation 11 as formulated in [8].

$$\boldsymbol{\mu} = (\boldsymbol{S}, \boldsymbol{O}, \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi}) \tag{11}$$

where $\boldsymbol{S} = \{s_1, s_2, \ldots, s_n\}$ is the set of hidden states, $\boldsymbol{O} = \{o_1, o_2, \ldots, o_m\}$ is the set of observations, $\boldsymbol{A}$ is the transition matrix, $\boldsymbol{B}$ is the emission matrix, and $\boldsymbol{\pi}$ is the initial state matrix, where π_i is the probability that state s_i is a start state. The matrices have been defined in the following equations.

$$\boldsymbol{A} = \{a_{ij}\}, \quad a_{ij} = P\left(s_j \mid s_i\right) \quad for \;\; i, j = 1, \ldots, n \tag{12}$$

$$\boldsymbol{B} = \{b_{ij}\}, \quad b_{ij} = P\left(o_j \mid s_i\right) \quad for \;\; i = 1, \ldots, n, \; j = 1, \ldots, m \tag{13}$$

$$\pi_i = \frac{1}{\sum S_i} \quad i = 1, \ldots, n \tag{14}$$

All states have equal probabilities to be the initial state as depicted in the equation for π_i.

Hidden Markov Models are used to model temporal and sequence data. In order to improve the accuracy of the time series classifier, the HMM is trained to evaluate, confirm and correct the classification results performed by the LSTM (initial classifier). The general framework of the approach is sketched in Fig. 3 (from [8]) which shows the two phases of this approach. During training phase, the confusion matrix and the classified data outputted by the first classifier will be used. In the classification phase, the trained HMM will be used to reclassify the sequence of classified samples. The correct samples will be confirmed and the misclassified samples will be corrected. Classifying any time series data will require two stages.

A confusion matrix $\boldsymbol{C} = c_{ij}$ for $i, j = 1, 2, \ldots, n$ is denoted such that c_{ij} represents the number of samples that belong to state s_i but they have been misclassified as s_j. Confusion matrix is also used in computing the emission matrix, $\boldsymbol{B}$.

Unlike traditional classifiers (i.e. support vector machine (SVM)), HMM classifies the data on the basis of their temporal relations [8]. To classify a given sequence of observations, we need to find the most likely sequence of states (path) that produces these observations.

4. CLASSIFICATION SCHEME

There are different traits by which classification can be performed. Using the taxonomy and classification scheme for artificial space objects [10] as a reference, we have selected attitude and shape as classifiers. We will primarily use light curves of artificial space objects in LEO as the training and testing datasets. The first classification determination is made from the control states. The control profiles include (three-axis) spin stabilized, and tumbling/irregular. The control states are not limited to ones used in this work and other states are possible (i.e. regularly spinning/rotating), but for this study, these are sufficient. The next classification is determined using the shape model, either separating it as regular or irregular. Under regular, for example, we have objects in the bank such as ellipsoid, cylinder, cone, cubesat, torus, probe, box-wing satellite, probe, disk and rocket body. A select few of these models have been included in Fig. 4. If the object cannot be identified as one from the aforementioned list, it will be categorized as an irregular object, which includes fragments.

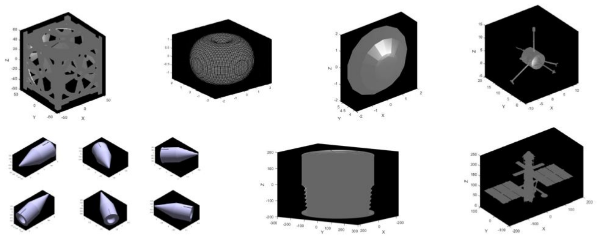


Figure 4. Models of artificial space objects. From left to right, (1st row): 1U Cubesat, torus, disk, Pioneer 9 probe. (2nd row): rocket nose cone at different viewing geometries, NASA RASSOR drum, box-wing satellite.

Light curves of tumbling objects have a wide variety of features in contrast with stable objects. Stable satellites tend to have a single main peak around low solar phase angles caused by a solar panel glint [6]. Additionally, observed brightness values tend to have relatively low variance about their trend. On the other hand, tumbling object light curves tend to take one of two forms: sinusoidal, or aliased. A sinusoidal signature will have several peaks and troughs. An aliased light curve is caused by the sensor sampling at a rate below the Nyquist frequency unable to properly capture the periodic behavior. These light curves do not have a distinguishable trend and look like scattered data points. To enhance our ability to identify these forms, we utilize tests for periodicity, aliasing and feature selection. A schematic of the decision flowchart is depicted in Fig. 5 encompassing all the tests, given an object's light curve as the input. These additional tests have been discussed in further detail below.

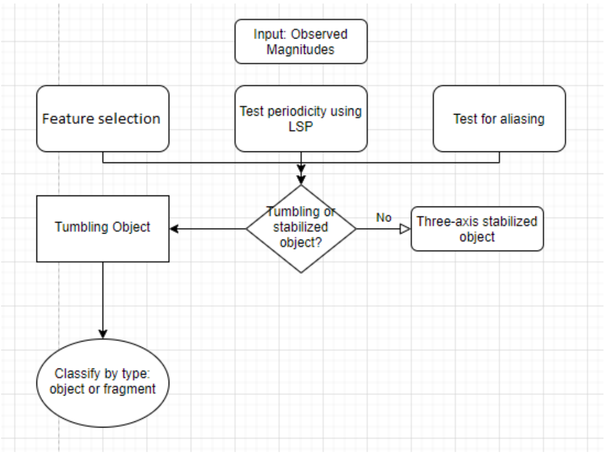

Input: Observed Magnitudes
Feature selection
Test periodicity using LSP
Test for aliasing
Tumbling Object
Tumbling or stabilized object?
No
Three-axis stabilized object
Classify by type: object or fragment

Figure 5. Classification flow diagram

4.1. Periodicity Test

Assuming the data is not aliased, we can test for periodicity to detect the presence of periodic waveforms, which differentiates tumbling from stable objects. One test that can be used to confirm the significance of a signal's periodic component is Fisher's Exact Test for Periodicity [20]. The test statistic is found by extracting the most significant frequencies. As noted from the analysis in [2], the Lomb-Scargle periodogram has proven to be successfully consistent in finding dominant frequencies as it does not require observations to be uniformly spaced, which is rare in real photometric data. Furthermore, heavy user influence in trimming the search frequency range is not present unlike other fourier and phase-folding methods. If the light curve has a significant periodic component, we will assume that it came from a tumbling object. Since stable objects can have a single peak in its periodogram, a threshold must be set on the period length that would be tested such that the test does not indicate that stable objects are periodic (with a long period). This eliminates false positives.

Outliers are inevitable in photometry and should be removed because they can distract from periodic components that are truly present in a light curve. They also cause some undesired results in the periodogram and in turn, the test for periodicity itself. Additionally, there are some frequencies that we do not expect in the light curve from a tumbling object. For instance, if the fundamental frequency of a light curve is too low, that frequency may correspond to one or fewer periods occurring within the observation window. Then this frequency is likely not representative of the tumble rate. Generally, there is a clear peak close to zero for the unstable object's period while the stable objects' periods tend to be much longer [6].

4.2. Test for aliasing

An assumption of stationarity is sufficient to give us the ability to detect aliasing in a temporally sampled signal process [29]. We can ensure stationarity using the Augmented Dicky Fuller test. Once this is confirmed, we can use bispectral analysis to identify aliased signals [30]. The bispectrum of a discrete-time signal is a periodic function in two frequency indices ω_1, ω_2. The bispectrum, defined to be the triple Fourier transform of the third-order autocorrelation, reduces to a function of two frequencies since stationarity confines the spectrum to the origin of the plane.

Assuming a real-valued discrete time series, all non-redundant information is confined to the square $0 \leq \omega_1, \omega_2 \leq \pi$. Due to symmetries, the non-redundant information in the bispectrum is confined to a particular triangle inside this square. The triangle consists of two pieces (isosceles and irregular). The discrete bispectrum is non-zero only in the isosceles triangular when no aliasing is present.

4.3. Feature selection

Certain features in a light curve can be correlated to physical structures on objects. The primary light curve features are narrow and broad brightness peaks correlating to structures such as the main payload enclosure, solar panels, and communications antennas. Large antennas are often located on the eastern and/or western sides of the nadir-facing main body as seen in a box-wing satellite. This asymmetry is observed in the reflected light of the photometric curves. Narrow features are narrow peaks whereas broad features include either a broad peak or a shoulder [6]. Feature selection prior to classification is essential as it removes irrelevant and redundant features to improve accuracy [4]. We can extract the feature information via wavelet decomposition, which is elaborated in the next section.

4.3.1. Wavelet Analysis

A wavelet is a sum of elementary functions with two requirements that it should integrate to zero, and that the function has to be well-localized. A wavelet is mathematically described as,

$$X_{a,b} = \int_{-\infty}^{\infty} x(t)\psi_{a,b}(t)dt \tag{15}$$

where $x(t)$ is the real signal, $\psi_{a,b}(t)$ is an arbitrary mother wavelet, a is the scale and b is the translation. The scale is inversely proportional to the frequency of the mother wavelet and the same as the size of the window. The translation parameter is analogous to how far we ”slide” the window from the starting point.

The Discrete Wavelet Transform (DWT) is utilized to extract the wavelet coefficients: approximation and detail. By applying the DWT again on the approximation coefficients of the previous DWT, we get the wavelet transform of the next level. At each next level, the original signal is also sampled down by a factor of 2 (sometimes de-noising). The Wavelet Transform automatically adjusts the window width to give good time resolution and poor frequency resolution at high frequencies, and good frequency resolution and poor time resolution at low frequencies [7]. The limit of the time and frequency resolutions is given by $\Delta t \Delta f \geq \frac{1}{4\pi}$. If a coefficient scores a high statistic value, data (in terms of this coefficient) can be easily separated by the classifier.

5. SIMULATION RESULTS

The training dataset consists of light curve vectors as inputs and class vectors as outputs. The model is trained to assign each measurement vector to a class using a set of training examples. The classification approach is tested using simulated data as well. Synthesizing light curves for each of the models in the bank using Lebedev quadrature at differing angular rotation speeds (from $\omega = 1$ to $\omega = 20 \deg/s$) for revolutions about its own spin axis makes up a portion of the volume of the training dataset. Variations in viewing geometries, altering size dimensions and interpolated surface reflectance properties across the facets also populate the training dataset. Python and Matlab are used as the simulation environment for this work.

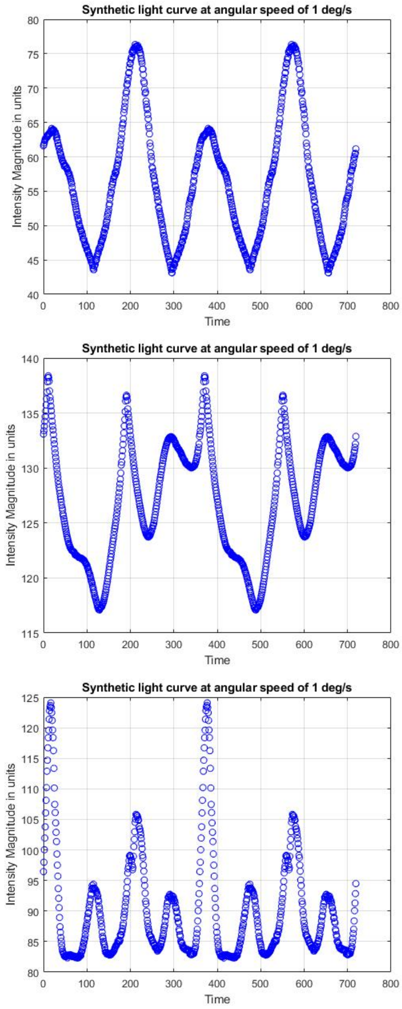


Figure 6. Synthetic light curves of Disk, RASSOR drum and Pioneer probe for training data set (Top to Bottom)

The simulation parameters used to generate the light curve measurements are as follows. The material surface properties were set to include ambient, diffuse and specular reflection ($k_a = 0.8, k_d = 0.1, k_s = 1$) respectively. Artificial space objects are primarily made of plastic, fiberglass and hard wood, hence the choices of coefficients to reflect these material properties. The lighting conditions are ideal to produce uniform white light at a point source that radiates as a headlight. The position of the object with respect to the observer's location on earth has coordinates to approximate an altitude of 1000 km. The spin-axis orientation is user-defined. A few examples of the simulated light curves are included in Fig. 6.

The various layers in the first classification stage using LSTM RNN is visualized in Figure 7.

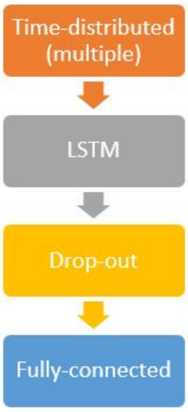


Figure 7. LSTM layers

The state space for the Markov chain is $\boldsymbol{S} = \{S, T\}$ where S represents stable and T represents tumbling RSOs. For the HMM, we assumed the initial probabilities are equal for both states giving $P(X_1 = S) = P(X_1 = T) = 0.5$, which forms our initial state matrix, $\boldsymbol{\pi}$. This means it is equally likely that the first time series came from a stable or a tumbling object. The transition matrix contains the probabilities that the underlying Markov sequence transitions from one state to another. We assumed the object is likely to stay in its stable state, unless it begins tumbling, in which case, it will very likely stay tumbling. This gives us the transition matrix, $\boldsymbol{A}$:

$$\boldsymbol{P} = \begin{bmatrix} P(S \mid S) & P(T \mid S) \\ P(S \mid T) & P(T \mid T) \end{bmatrix} = \begin{bmatrix} 0.9 & 0.1 \\ 0.01 & 0.99 \end{bmatrix}$$

where the notation $P(S \mid T)$ means the probability that the current state is S, given that the previous state was T. Finally, the emission matrix contains the probabilities of each possible observable, for each state.

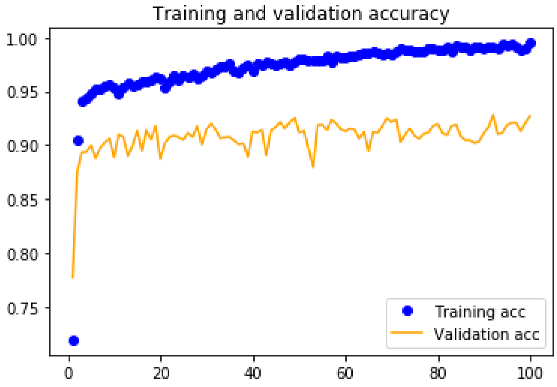


Figure 8. Training and Validation accuracy plots

After running 100 epochs, the resulting validation accuracy is 91.7%, which is an acceptable value by simulation standards in literature. The higher the number of epochs, the accuracy in turn increases. A more comprehensive visual depicting the accuracy and loss in training and validation is observed in Fig. 8.

6. CONCLUSION AND FUTURE WORK

A hybrid classification model, inclusive of an LSTM recurrent neural network and a HMM, is proposed to classify an unresolved object's light curve, in low earth orbit (LEO), by its shape and spin rate. It is distinguished between two control profiles (tumbling and stabilized) with additional tests for periodicity, aliasing and feature selection to enhance the process. This modified approach improves the accuracy of time series classification by executing a secondary screening accounting for the temporal relations in the data and eliminating redundancy in the classifiers. The fidelity of these simulations ensures it is both realistic and accurate to within 10% error, and the computation speed is faster with flexibility to deal with a larger amount of data.

Future improvements include incorporating real light curves in our test dataset. One thing to note with real data is that photometric observations with a gap in collection time can possess changes like slowing of tumbling rate. Here, it is important to use key identifiers in the light curve to match for resemblance. However, if the period is similar, we can correlate them to be the same object regardless of the gap in collection. Sample selection bias is also a parameter to remember. It can cause catastrophic errors in predictions on the testing data because standard assumptions for machine-learned model selection procedures break down, and dense regions of testing space might be completely devoid of training data [26]. An effective technique for artificial RSOs with minimal user interference to rectify sample selection bias is required.

ACKNOWLEDGMENTS

We acknowledge the support of the Natural Sciences and Engineering Research Council of Canada (NSERC), [funding reference number CGSD3-532704] along with Zonta International Foundation for the Amelia Earhart Fellowship.

BIBLIOGRAPHY

1. Balachandran K., & Subbarao K. (2019). Computing surface brightness integrals of artificial space objects with axisymmetric bodies using photometric light curves. In *International Astronautical Congress*.
2. Balachandran K., & Subbarao K. (2021). Comparison of deterministic methods to estimate sidereal rotation period from light curves. *Journal of Spacecraft and Rocketry*.

3. Batal L., Sacchi L., Bellazzi R., et al. (2009). Multivariate time series classification with temporal abstractions. In *Proceedings of the 22nd International Florida Artificial Intelligence Research Society Conference*, FLAIRS-22, pages 344–349. University of Pittsburgh.

4. Bennet J., Arul Ganaprakasam C., & Arputharaj K. (2014). A discrete wavelet based feature extraction and hybrid classification technique for microarray data analysis. *The Scientific world journal*, 2014.

5. Dahl G. E., Sainath T. N., & Hinton G. E. (2013). Improving deep neural networks for lvcsr using rectified linear units and dropout. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 8609–8613. IEEE.

6. Dao P., Haynes K., Gregory S., et al. (2019). Machine classification and sub-classification pipeline for geo light curves. *Proceedings of the Advanced Maui Optical and Space Surveillance Technologies Conference*, page 53.

7. Deokar S. & Waghmare L. (2014). Integrated dwt–fft approach for detection and classification of power quality disturbances. *International Journal of Electrical Power & Energy Systems*, **61**:594–605.

8. Esmael B., Arnaout A., Fruhwirth R. K., et al. (2012a). Improving time series classification using hidden markov models. In *2012 12th International Conference on Hybrid Intelligent Systems (HIS)*, pages 502–507. IEEE.

9. Esmael B., Arnaout A., Fruhwirth R. K., et al. (2012b). Multivariate time series classification by combining trend-based and value-based approximations. In *International Conference on Computational Science and Its Applications*, pages 392–403. Springer.

10. Früh C., Jah M., Valdez E., et al. (2013). Taxonomy and classification scheme for artificial space objects. Technical report, Air Force Research Lab, Kirtland AFB NM, Space Vehicles Directorate.

11. Furfaro R., Linares R., & Reddy V. (2018). Space objects classification via light-curve measurements: deep convolutional neural networks and model-based transfer learning. In *AMOS Technologies Conference, Maui Economic Development Board*.

12. Gers, F. A. & Schmidhuber J. (2000). Recurrent nets that time and count. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*. IJCNN 2000. Neural Computing: New Challenges and Perspectives for the New Millennium, volume 3, pages 189–194. IEEE.

13. Graves A., Mohamed A.-r., & Hinton G. (2013). Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE.

14. Hochreiter S. & Schmidhuber J. (1997). Long short-term memory. *Neural computation*, **9**(8):1735–1780.

15. Jah M. & Madler R. A. (2007). Satellite characterization: angles and light curve data fusion for spacecraft state and parameter estimation. In *Proceedings of the advanced Maui optical and space surveillance technologies conference*, volume 49.

16. Jozefowicz R., Zaremba W., & Sutskever I. (2015). An empirical exploration of recurrent network architectures. In *International conference on machine learning*, pages 2342–2350.

17. Kaasalainen M. & Torppa J. (2001). Optimization methods for asteroid lightcurve inversion: I. shape determination. *Icarus*, **153**(1):24 – 36.

18. Kaasalainen M., Torppa J., & Muinonen K. (2001). Optimization methods for asteroid lightcurve inversion: II. the complete inverse problem. *Icarus*, **153**(1):37 – 51.

19. Kim D.-W. & Bailer-Jones C. A. (2016). A package for the automated classification of periodic variable stars. *Astronomy & Astrophysics*, **587**:A18.

20. Liew A. W.-C., Law N.-F., Cao X.-Q., et al. (2009). Statistical power of fisher test for the detection of short periodic gene expression profiles. *Pattern Recognition*, **42**(4):549–556.

21. Lin J., Keogh E., Lonardi S., et al. (2003). A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery*, pages 2–11.

22. Linares R., Crassidis J. L., & Jah M. K. (2014). Space object classification and characterization via multiple model adaptive estimation. In *17th International Conference on Information Fusion (FUSION)*, pages 1–7. IEEE.

23. Lu X.-P. & Jewitt D. (2019). Dependence of light curves on phase angle and asteroid shape. *The Astronomical Journal*, **158**(6):220.

24. Polikar R. (2006). Ensemble based systems in decision making. *IEEE Circuits and systems magazine*, **6**(3):21–45.

25. Rabiner L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, **77**(2):257–286.

26. Richards J. W., Starr D. L., Brink H., et al. (2011). Active learning to overcome sample selection bias: Application to photometric variable star classification. *The Astrophysical Journal*, **744**(2):192.

27. Schafer E. (2017). *Stereoscopic light curve analysis of space debris objects*. Master's thesis, DLR Institute for Technical Physics.

28. Sun Z., Di L., & Fang H. (2019). Using long short-term memory recurrent neural network in land cover classification on landsat and cropland data layer time series. *International journal of remote sensing*, **40**(2):593–614.

29. Vixie K. (2007). *Detection of aliasing in signals*, Portland State University and Los Alamos National Laboratory.

30. Vixie K. R., Wolinsky M., & Sigeti D. E. (1999). The bispectral aliasing test: a clarification and some key

examples. In *ISSPA'99. Proceedings of the Fifth International Symposium on Signal Processing and its Applications (IEEE Cat. No. 99EX359)*, volume 1, pages 255–258. IEEE.

31. Xu L., Krzyzak A., & Suen C. Y. (1992). Methods of combining multiple classifiers and their applications to handwriting recognition. *IEEE transactions on systems, man, and cybernetics*, **22**(3):418–435