DETECTING ORBITAL ANOMALIES IN SPACE OBJECT BEHAVIOR USING RECURRENT NEURAL NETWORKS

Gaetano Calabrò[*], Riccardo Cipollone[†], and Pierluigi di Lizia[‡]

INTRODUCTION

As space exploitation has become a fundamental pillar of modern society and global economy, the continued use of the Near-Earth environment has resulted in a delicate and increasingly complex situation. The overpopulation that this region of space is experiencing is accompanied by a general increase in the risk of collisions and fragmentations that are progressively reinforcing the likelihood of the Kessler's syndrome.[1] As part of the efforts to address these challenges, the Space Surveillance and Tracking (SST) framework plays a pivotal role. The main outcome of the above activity is the creation and maintenance of space object catalogs, which act as repositories of relevant orbital information for all tracked targets. These databases are crucial to support and enhance key functionalities of Space Situational Awareness (SSA)[2] such as conjunction analysis, fragmentation forensics, and collision risk assessment that benefit from accurate and up-to-date orbital information. Generally, orbital data is stored in space object catalogs using various encoding formats that may differ in terms of accuracy and quality, with a common example being the Two-Line Element (TLE) set.

Recently, particular focus has been placed on the construction and exploitation of the so-called Pattern of Life (PoL), a long-term collection of orbital behavioral signatures of a resident object that may include maneuvers and routine operations. The widespread availability of thousands of TLE sets has enabled the investigation of data-driven methodologies to characterize hidden patterns of the operational behavior of satellites, primarily focusing on the identification and classification of maneuvering activities.[3] Within this context, maneuver detection not only is instrumental in improving the accuracy of RSO catalog maintenance pipelines but also in building and refining the Patter of Life for each space object. The information derived from the PoL can be leveraged to infer recurrent behavioral patterns including for example all routine maneuver cycles (e.g. orbit raising maneuvers, station-keeping, collision avoidance) that constitute an object nominal set of operations. In this framework, anomaly detection can be regarded both as a broader discipline encompassing detection of space events such as maneuvers, as well as a nested task focused on distinguishing nominal behavior from out-of-nominal activities. By considering the latter definition, anomaly detection methods can be then leveraged to detect deviations from expected maneuvering patterns registered in the PoL. These abnormalities could signal suspicious activities that may pose a risk towards other resident space objects. In addition, once detected, out-of-nominal behaviors must be

[*]Ph.D. student, Department of Aerospace Science and Technology, Via Giuseppe La Masa 34, 20156, Milan, Italy.
[†]Space Safety Engineer, Look Up Space, 18 Rue des Cosmonautes, Toulouse
[‡]Associate Professor, Department of Aerospace Science and Technology, Via Giuseppe La Masa 34, 20156, Milan, Italy.

excluded from Pattern of Life refinement process or can enforce the need of initiating a new PoL. Addressing this increasingly urgent topic is crucial to prevent degradation of space object catalogs if enhanced with the PoL data and to maintain reliability and safety in the space domain.

A traditional approach in anomaly detection involves the analysis of time series in a data-driven framework, which is common for telemetry applications and other fields. Therefore, given the novelty of the topic, research in maneuver detection can be extended to detect deviations from PoL-predicted maneuvers or from an assumed nominal behavior when the PoL data is unavailable. As for maneuver detection, early techniques focused on pure statistical analysis time series derived from Keplerian orbital elements within sliding window frameworks.[4] However, a significant limitation of these methods lies in the need of proper tuning of the threshold used to distinguish maneuver signatures from the natural orbital dynamics. To address this drawback, methods that involve adaptive thresholding have been developed and tested to specifically recognize maneuvering objects in the Low Earth Orbit (LEO) regime.[5] As a natural evolution of classical statistical approaches, machine learning techniques have also been extensively investigated to overcome these limitations. R. Mital et al.[6] leveraged both unsupervised clustering and supervised machine learning techniques to characterize satellite's behaviors. PoL-based supervised approaches have been explored using more advanced architectures as Convolutional Neural Networks (CNNs). For instance, T. Roberts[7] applied CNNs to detect maneuvers and anomalies in Geostationary Earth Orbit (GEO) satellites. Similarly, R. Cipollone et al.[8,9] investigated both CNNs and Recurrent Neural Networks (RNNs), specifically the Long Short-Term Memory (LSTM) cells, to perform accurate maneuvering epoch detection by learning from historical patterns. LSTM cells were also embedded by R.Kato et al.[10] into an Autoencoder architecture to detect station-keeping maneuvers in GEO satellites within an anomaly detection framework without relying on historical maneuvering records.

The presented work builds on the application of two LSTM-based neural network architectures combined with an adaptive thresholding technique to detect anomalous segments in TLE data. Assuming a fully nominal training dataset, the first network is trained to predict the next set of Keplerian elements, while the second also incorporates the aleatoric uncertainty into its predictions. The prediction error is then evaluated against both real and simulated ground-truth data through an unsupervised dynamic thresholding technique. To reduce the number of false positives, the score difference of adjacent anomalies is compared against a pruning parameter, which is finetuned according to two alternative strategies: for the first network, the optimal pruning parameter is selected enforcing an anomaly-free validation dataset; in the second, aleatoric uncertainty is leveraged in the computation of the detection metric. Ground-truth anomalies are manually identified through visual inspection of the test dataset and used for validation.

MATHEMATICAL FRAMEWORK

This section briefly introduces all mathematical tools of the proposed methodology, offering insights on key concepts of Long Short-Term Memory (LSTM) cells, as well as an explanation of the dynamic thresholding technique developed by K. Hudman et al.[11]

LSTM Neural Networks

Long Short-Term Memory are a specialized class of Recurrent Neural Networks, both designed to capture temporal dependencies in sequential data. This ability stems from the structure of the hidden recurrent layers, which propagate temporal information by passing the hidden state $\mathbf{h}$ from consecutive layers. However, traditional RNNs struggle to effectively model long-term dependencies due to the vanishing or exploding gradient problem.[12] To address this limitation, LSTM introduces a mechanism to preserve both short term and long term temporal patterns. As illustrated in Figure 1, a typical LSTM cell structure is composed by 3 gates: the input gate $\mathbf{i}_t$, the forget gate $\mathbf{f}_t$ and the output gate $\mathbf{o}_t$. At each timestep t, the input to the cell $(\mathbf{x}_t, \mathbf{h}_{t-1})$ is passed through the gates that are parameterized by each own set of weight matrices $\mathbf{W_g} = \{\mathbf{W}_{gx}, \mathbf{W}_{gh}\}$ and the bias vectors $\mathbf{b}_g$, where $g \in \{i, f, o\}$. These parameters make up the set of trainable parameters of the network that are adjusted during the learning process to minimize a prescribed loss function.

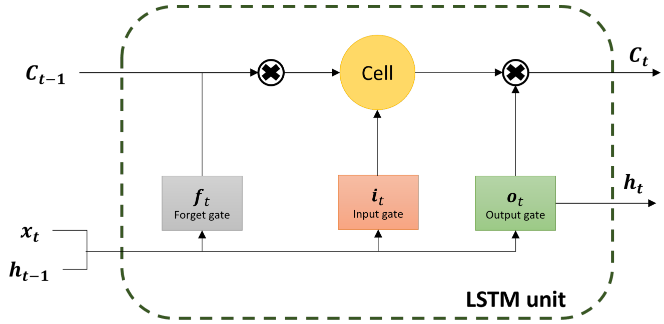

Figure 1: Simplified block diagram of a LSTM cell. It includes the input x_t, both previous and current hidden and cell states: h_{t-1}, h_t and C_{t-1}, C_t. The cell state represent the long term memory of LSTM models

The loss function is typically selected depending on the task for which the network is trained. For a prediction task as the one presented in this work, a common choice consists in the Mean Square Error (MSE) loss, which measures the difference between predicted $(\hat{\mathbf{y}})$ and true values $(\mathbf{y})$. The expression of the MSE loss is reported in Eq. (1), where N is the number of samples in each training batch.

$$\mathcal{L}_{MSE} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 \tag{1}$$

To include uncertainty in the prediction, an alternative LSTM network is developed by leveraging the principles of the so-called Bayesian Neural Networks (BNNs). Within this framework, two primary sources of uncertainties are recognized: aleatoric and epistemic uncertainty.[13] Aleatoric uncertainty captures the inherent noise of the data sequence, while epistemic uncertainty reflects the lack of knowledge of the model, which is typically addressed by providing a large amount of training data.

Although accounting for both sources is crucial for a comprehensive representation of the model uncertainty, this work focuses specifically on the aleatoric uncertainty. To model this uncertainty, assumed to be sample-dependent (*e.g. heteroscedastic*) noise, the MSE loss can be modified by including some additional weighting and penalty terms, defining the Heteroscedastic Mean Squared Error (HSMSE) loss.[14]

$$\mathcal{L}_{HSMSE} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2\sigma^2(x_i)} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|^2 + \frac{1}{2} \log \sigma^2(x_i) \tag{2}$$

In Eq. (2), a weight factor based on the i-th sample noise standard deviation $\sigma(x_i)$ is applied to the mean square error to penalize less terms with high variance. At the same time an additional contribute increments the loss for predicting terms with high variance.

Dynamic Thresholding

As previously mentioned, a common criticality of detection algorithms is the selection of an appropriate threshold. Whether in purely statistical methods or machine learning-based approaches, a common solution involves assuming a certain form of the expected distribution of the monitored value-typically a Gaussian distribution-followed by the use of σ-based thresholds. To avoid such parameterization, distance-based methods (*e.g.* K-nearest, Isolation Forests, *etc.*) can be leveraged, despite the increased computational complexity and a generally reduced sensitivity. To address all these limitations, this work adopts a non-parametric dynamic thresholding approach.[11]By letting the prediction errors speak for themselves, the threshold is continuously adapted within batches of windows sliding over the error sequence, which is first smoothed using an Exponentially-Weighted Moving Average (EWMA) algorithm.[15] The aleatoric uncertainty predicted by the network is used to normalize the prediction error and obtain a standardized residual (analogous to a z-score) with respect to the ground truth (Eq. (3)):

$$\tilde{e} = \frac{\hat{x} - x}{\hat{\sigma}} \tag{3}$$

As anomalies are detected based on the amplitude of the error, the absolute value of both the error and the standardized residual is extracted and used as a detection metric. In both cases, by defining all candidate thresholds ϵ as a function of a parameter set $\mathbf{z}$:

$$\epsilon = \mu(\mathbf{e}_s) - \mathbf{z}\sigma(\mathbf{e}_s) \tag{4}$$

The threshold ϵ applied for the current batch is found by fixing the z parameter as a result of the Eq. (5):

$$\epsilon = argmax(\epsilon) = \gamma \left(\frac{\Delta\mu(\mathbf{e}_s)}{\mu(\mathbf{e}_s)} + \frac{\Delta\sigma(\mathbf{e}_s)}{\sigma(\mathbf{e}_s)} \right) \tag{5}$$

Where:

$$\begin{aligned} \gamma &= \frac{1}{n_a N_{a,adj}} \\ \Delta\mu(\mathbf{e}_s) &= \mu(\mathbf{e_s}) - \mu(\{e_s \in \mathbf{e}_s | e_s < \epsilon\}) \\ \Delta\sigma(\mathbf{e}_s) &= \sigma(\mathbf{e_s}) - \sigma(\{e_s \in \mathbf{e}_s | e_s < \epsilon\}) \end{aligned} \tag{6}$$

The factor γ penalizes long sequences of anomalies (n_a) and the number of consecutive windows with at least one anomalous value inside ($N_{a,adj}$). The delta terms describe the difference between both mean and standard deviation of the whole window of smoothed errors with respect to the ones below the threshold value, capturing the statistical contrast that potentially caused by the presence of anomalies.

To reduce the rate of false positives, the sequence of detected anomalies undergoes a pruning process that assesses each anomaly based on the local behavior within the window. Hence, the algorithm consists in creating a set containing in descending order the maximum error above the threshold $e_{max} = max(\mathbf{e}_{seq})$ for all candidate anomalous sequences $\mathbf{e}_{seq}$ in the batch. Then, by appending the maximum non anomalous value on the right of the set, each value is then used to compute a distance score with respect to its neighbor, as in Eq. (7).

$$d = \left\| \frac{e^i_{max} - e^{i+1}_{max}}{e^i_{max}} \right\| \tag{7}$$

If this value is greater than a fixed percentage p named as the *pruning parameter*, then the anomalous sequence is retained. On the contrary, if the percentage decrease is less the the pruning value, the sequence is reclassified as nominal and the maximum non anomalous value is updated.

SETUP AND RESULTS

The developed method evaluates the effectiveness of the LSTM-based networks in predicting the evolution of Keplerian element to find evidences of anomalous behaviors. The analysis assumes that the training dataset describes the nominal orbital motion of these objects, implicitly including routine operations. The section also outlines the dataset investigated for a preliminary result, describing the whole pre-processing pipeline and model selection.

Dataset and pre-processing

The dataset used to test the approach includes two LEO objects. The first object is synthetic and was generated by randomly selecting a TLE of the International Space Station (ISS) from Space-Track website*, and propagating it over a time span of approximately 6 years using Simplified General Perturbation-4 (SGP4) propagator. To simulate an active satellite, a maneuvering pattern was introduced by applying randomized along-track maneuvers with intensity on the order of tens of centimeters per second. These maneuvers are simulated at at regular intervals ranging from 35 to 45 days. Additionally, six artificial anomalies were generated at random epochs after the fourth year of propagation, simulating sudden deviations from the routine patterns. The evolution of the semi-major axis of the simulated object (NORAD 99999) is shown in Figure 2, with both routine maneuvers and anomalies clearly highlighted.

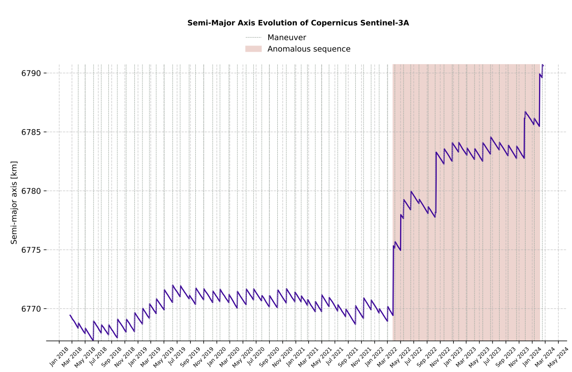

Figure 2: Evolution of the simulated satellite semi-major axis. Both routine maneuvers (green) and anomalous sequence (red) are shown as vertical lines

*Space-Track Webpage: `https://www.space-track.org/`

The second object is a real satellite, Copernicus Sentinel-3A, selected after a visual inspection of its orbital behavior. The analysis of its semi-major axis evolution (Figure 3) reveals an unexpected sequence of maneuvers — confirmed by the International Laser Ranging Service (ILRS)* maneuvering database — that began after November 15th, 2021, likely due to a change in the environmental conditions. The selected input and output features consist of Keplerian orbital elements,

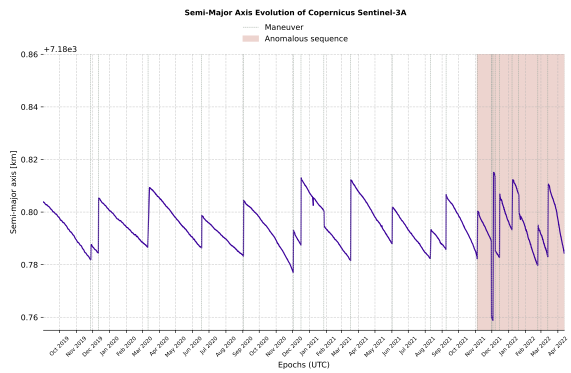

Figure 3: Evolution of the Sentinel-3A semi-major axis. Both routine maneuvers (green) and supposed anomalous sequence (red) are shown as vertical lines.

which have proven effective in representing anomalous behavior effects, as shown in the examples above. Therefore, given N available samples, the input sequence is defined as $N \times 6$ parameters, including:

$$x_i = [a, e, i, \Omega, \omega, \theta] \text{ for i = 1, ..., N} \tag{8}$$

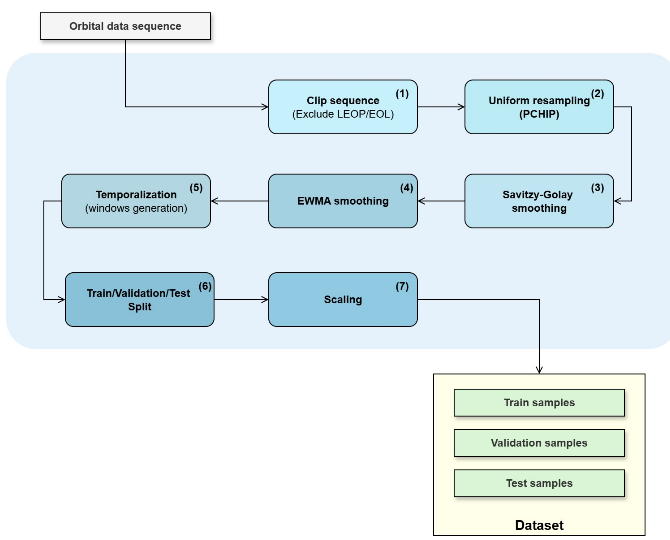

Figure 4: Pre-processing pipeline used to generate the dataset

Before feeding the dataset into the network, a pre-processing, represented in 4, cleans and setup the data format to properly match the required one from the network. First, the sequence of Ke-

*ILRS webpage: `https://ilrs.gsfc.nasa.gov/`

plerian elements is cleaned from samples not belonging to the operative regime, such the ones representing Launch and Early Orbit Phase (LEOP) activities.

The data sequence is successively resampled onto a uniform temporal grid. Uniform sampling is key requirement when working with LSTM-based networks, which inherently assume that the time intervals between adjacent data points are constant.[16] However, as TLEs are often irregularly sampled in time, a resampling procedure is crucial to prevent additional sources of error during prediction. To accommodate the uniformity requirement, both real and simulated datasets are resampled at six hours intervals using a Piecewise Cubic Hermite Interpolating Polynomial (PCHIP) fitting method.[17] This technique has the advantage to preserve the underlying shape of the time series and provide a solution to the presence of missing data points.

Following resampling, a two-stage smoothing filter procedure is applied to reduce high-frequency noise that may stem from both TLEs and interpolation artifacts. The first stage involves a Savitzky-Golay polynomial filter[18] , followed by an EWMA algorithm. Both filters employ a small window size to retain local, low-frequency components while effectively removing faster oscillations.

After the cleaning process is terminated, the dataset is reshaped to match the tensor format required for any LSTM input. Specifically, the original $N \times 6$ sequence is converted into a $\tilde{N} \times W_{lb} \times 6$ using a sliding window algorithm with stride $s = 1$, where $W_{lb} = 128$ represents the look-back window size. The dimension of the new dataset, $\tilde{N}$, after the conversion is computed as:

$$\tilde{N} = \frac{N - d + 1}{s} \tag{9}$$

One fundamental step in machine learning pre-processing pipelines is splitting the dataset into training, validation, and test sets. In this work the 80%-20% rule of thumb is adopted to split the data into training and test sets. Then, the training set is further partitioned using the same rule to create the validation subset.

Once the three sets are separated, the final step consists in data normalization, which prevents features with large scales from dominating the training process. In this case, a MinMax normalization is applied to each feature, limiting all values between 0 and 1, as defined in Eq. (10).

$$x_i = \frac{x_i - x_{i_{min}}}{x_{i_{max}} - x_{i_{min}}} \tag{10}$$

Model selection

The two neural network architectures share the same core structure, consisting on multiple stacked Bi-Directional LSTM (Bi-LSTM) layers with the only difference being the loss function and the number of predicted features — adding six parameters to model noise. Unlike vanilla LSTM cells that learn temporal patterns in the forward direction, the selected architecture adds an additional layer of detail by capturing time dependencies in reverse order. To achieve this, the number of traditional LSTM cells is doubled: one set processes the input in its natural sequential order, while the other analyses it in reverse.[19]

The introduction of two different loss functions stems from the idea of incorporating noise levels inside the detection metric. In this framework, less confident predictions weight down the contribution of the prediction error. As a result, sequences that are inherently uncertain but not anomalous do not dominate the anomaly score, allowing the method to better distinguish between actual anomalies and noisy predictions. The complete Bi-LSTM architecture can be summarized as follows:

- A first Bi-Directional LSTM layer with 8 cells.
- A second Bi-Directional LSTM layer with 6 cells.
- 6 dense layers of 1 neuron each to encode the features separately. This solution lies somewhere between a multi-head architecture and a single final dense layer.
- An additional head made of 6 dense layers with 1 neuron each for the second network, encoding the noise level of each feature.

The network training and dynamic thresholding algorithm hyperparameters are reported in Table 1 and Table 2. In both cases, their selection results from a refinement process carried out through an iterative trial-and-error procedure.

Hyperparameter	Value
Error window size	64
Batch size	10
Smoothing percentage	0.3
Pruning parameter	0.5

Table 1: (a) Dynamic Thresholding hyperparameters

Hyperparameter	NN 1	NN 2
Training epochs	500	500
Lag Window	64	64
Batch size	1	1
Optimizer	Adam	Adam
Loss Method	MSE	HSMSE

Table 2: (b) Neural Network hyperparameters

Simulation and Results

Two testing scenarios on both simulated and real satellites are described in this subsection. Both neural networks are trained and evaluated independently on the two datasets, consisting of approximately 4200 and 6500 data points for the simulated and real satellite scenarios, respectively. Both training and inference were conducted on the same platform, equipped with an Intel® Core™ i7-1370P (20 CPUs), 16GB of RAM and an integrated graphic card, using the Tensorflow Keras framework. The training process required approximately one hour for the simulated case and for the real one, mainly due to the larger volume of available data. In both scenarios, the inference times were less than one second, confirming that the deployment of the model does not require any dedicated Graphic Processing Units (GPUs).

Considering the simulated scenario first, Figure 5 and Figure 6 show the predicted trends for each Keplerian parameters according to $NN1$ and $NN2$, respectively. Observing the behavior close to the first anomaly epoch, it is clear evident that while the network attempts to follow the ground truth evolution, the prediction error becomes significant for some parameters, particularly those strongly affected by the anomalous maneuver. Additionally, as can be noted from Figure 6, the predicted uncertainty evolves close to the mean for all the orbital elements, with noticeable deviations for the inclination during maneuvering epochs. Therefore, incorporating the estimated noise into the detection metric is acts as a regularization factor, mitigating false positives in sequences that are characterized by discontinuities (e.g. jumps). One particular behavior consists in the leakage that can be observed on the orientation parameters i, Ω and ω that are not directly involved from the anomalous maneuvering pattern. A specific behavior is the leakage observed in the orientation parameters i, Ω, and ω, which are not directly affected by the anomalous maneuvering pattern. This phenomenon originates naturally from the architecture of the neural network, which learns a condensed latent representation incorporating contributions from all parameters. Consequently, this issue could be mitigated by increasing the depth of the separated head or, alternatively, by adopting a fully multi-head architecture.

As shown in Figure 7, the prediction error exhibits a growing trend and spikes in correspondence of each anomalous epoch, reflecting the network inability to capture the continuous unforeseen behavior. As illustrated, the detection algorithm correctly identifies the abrupt anomalous shift in the Keplerian elements. However, it gradually adapts to the new sequence and does not flag any successive anomaly. This behavior stems from the nature of the dynamic threshold, which raises the baseline according to the errors inside the window. Therefore, the result can be interpreted as the indication of a new operational regime, with the highest anomalous peaks marking its beginning.

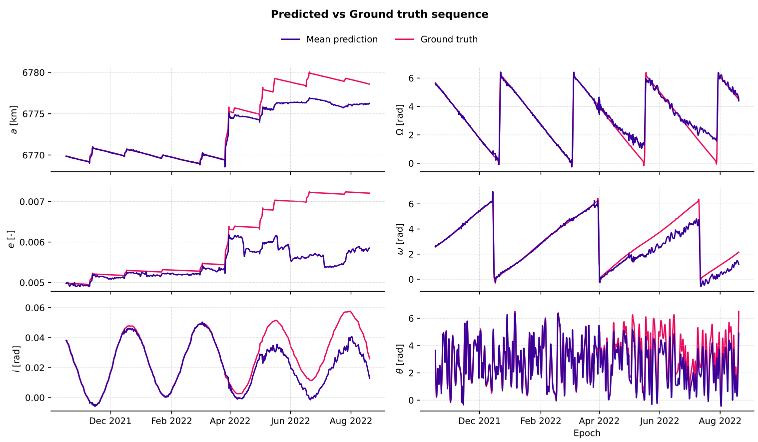

Figure 5: Prediction trends on each Keplerian parameter for the simulated satellite. The Network is optimized using the MSE loss function

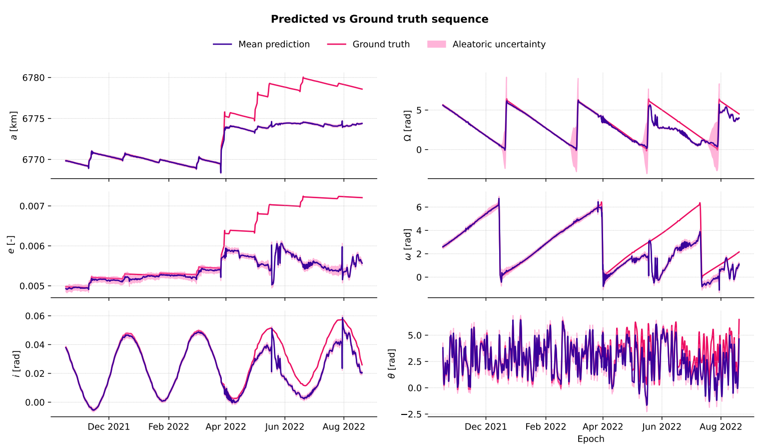

Figure 6: Prediction trends and inherent aleatoric uncertainty for each Keplerian parameter for the simulated satellite

As for the `Sentinel-3A` real scenario, the prediction trends shown in Figure 8 reveal that the Neural Network accurately follow the parameters behaviors, including anomalous changes that occurs at the beginning. However, by looking at the prediction errors and the results of the detec-

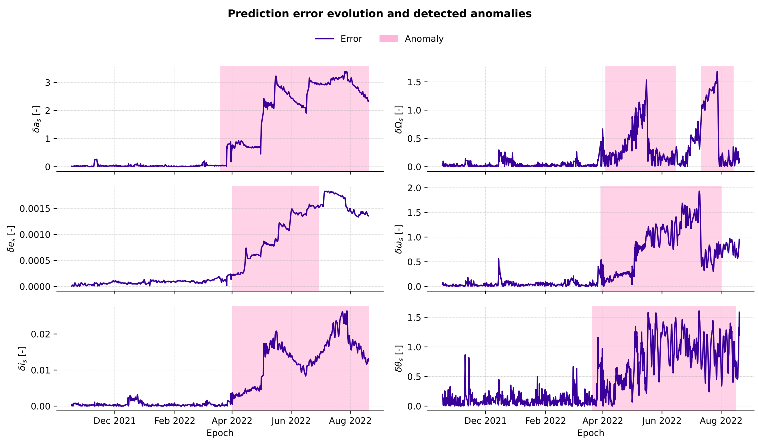

Figure 7: Prediction error (from $NN1$ with highlighted detected anomalous sequences. The found anomalous epochs are grouped together to generate a unique sequence. The results from $NN2$ are equivalent and thus omitted

tion algorithm for $NN2$ illustrated in Figure 9, a spike appears in correspondence to the known anomalous maneuvering sequence. The detection algorithm correctly identifies the true anomalous sequence, along with additional detections primarily due to the high prediction noise. This suggests that, for real objects, noise levels can significantly impact the prediction performance and should be included as features.

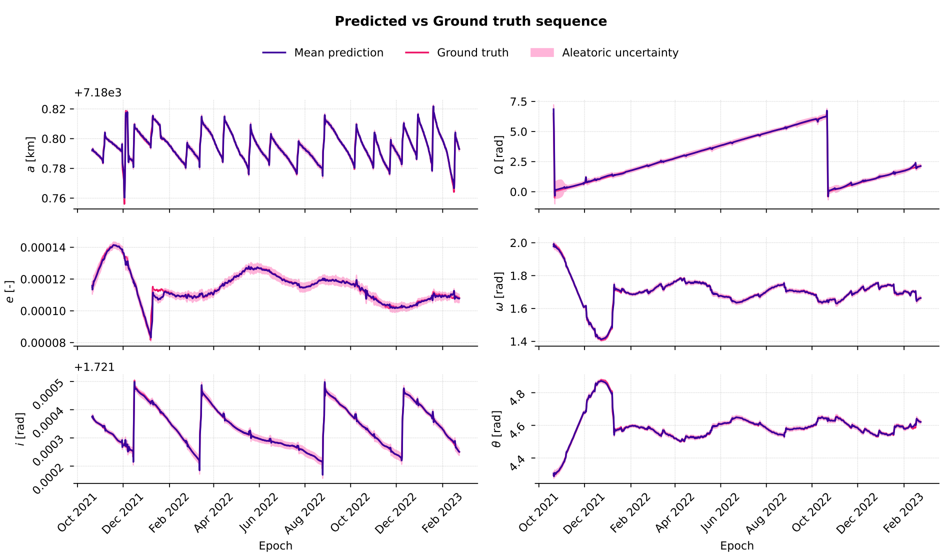

Figure 8: Prediction trends on each Keplerian parameter for the `Sentinel-3A`

Sensitivity analysis

To evaluate the impact of the temporal context provided as input to the prediction model on the anomaly detection performance, a sensitivity analysis is carried out only on the real satellite test scenario by varying the number of lagged time steps. Four configurations were tested (64, 128,

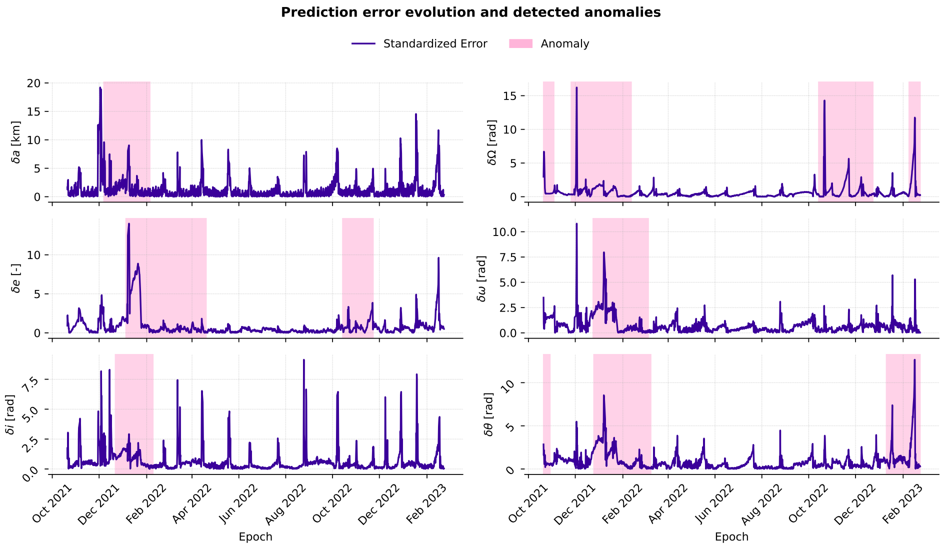

Figure 9: Prediction error with highlighted detected anomalous sequences. Adjacent anomalous sequence are grouped together

256 and 512 time steps), and for each case the prediction error was computed and compared on the most relevant orbital parameters: semi-major axis, eccentricity and inclination. The resulting error trends represented in Figure 10 highlight how the different window lengths affect both the amplitude and the temporal behavior of the residuals. This analysis provides insight into the trade-off between longer historical dependencies, which improve stability but increase computational cost, and shorter lags, which enhance responsiveness to rapid dynamical changes at the cost of higher fluctuations. Therefore, selection of intermediate values (e.g. 128 or 256) represents a suitable compromise between stability and reactivity to actual anomalies.

To further assess the sensitivity of the anomaly detection algorithm to its hyperparameters, a two-dimensional sweep was performed varying both the window size (2 to 128 samples) and the error batch size (1 to 32 samples) while keeping the lag time steps fixed to 64. For each parameter combination, the anomaly detection pipeline was executed and the maximum anomaly score observed across the semi-major axis, eccentricity and inclination was recorded. The computed values were aggregated into a heatmap, where each cell represents the maximum score for a specific window–batch configuration. This visualization (Figure 11) highlights how the algorithm responsiveness changes with different temporal dimensions, enabling the identification of hyperparameter regions that maximize sensitivity to anomalies while reducing false positives (e.g. detection of anomalies in the inclination). Specifically, regions with small window sizes are characterized by high anomaly scores, indicating that the reduction of the smoothing horizon and limited local statistics tend to generate more responses. In contrast, longer temporal windows reduce anomaly scores down to zero. This highlight that the dynamic detection technique no longer adapts the threshold to the local statistics but retains one or slightly more threshold values, computed over larger sequences.

However, the illustrated heatmaps alone are not sufficient for selecting an optimal combination of the two hyperparameters as the anomaly score does not explicitly reflect either the detection rate of actual anomalies or the false positive rate. Nevertheless, the embedded information can be leveraged as a first guess when employed within a grid search optimization framework, particularly

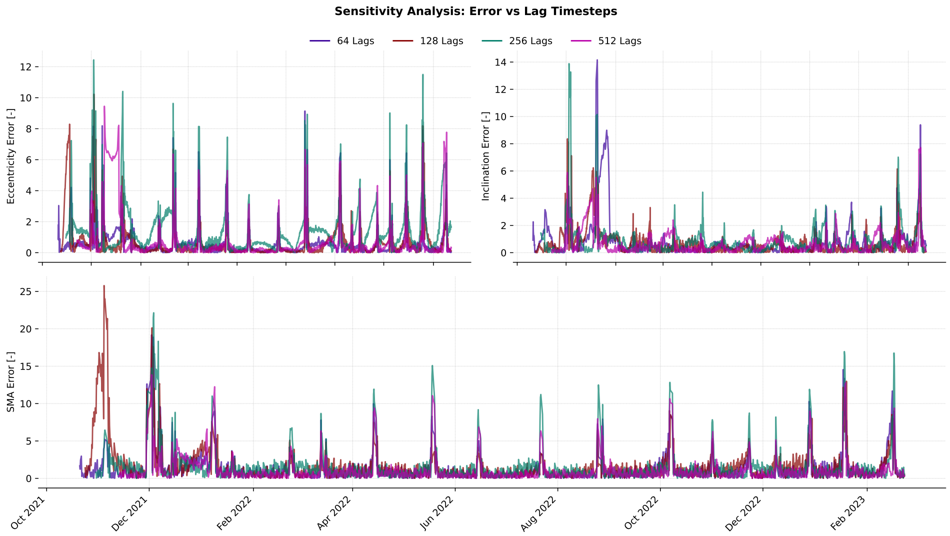

Figure 10: Prediction error behavior with respect to the temporal context

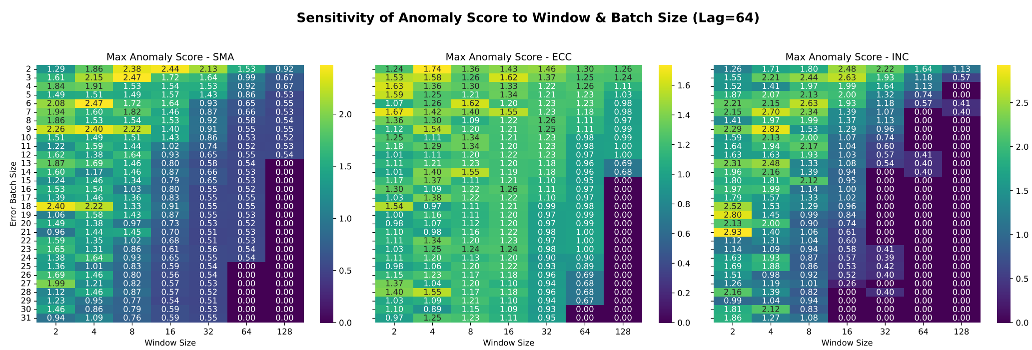

Figure 11: Heatmaps for semi-major axis (SMA), eccentricity (ECC) and inclination (INC) anomaly scores. It is obtained by varying window size and batch size hyperparameters of the detection algorithm

when combined with additional tunable parameters such as the pruning threshold. Stemming from this idea, a further sensitivity analysis is carried out by fixing again the above hyperparameters while varying the pruning parameter within 0.1 and 1. As described in previous sections, this threshold variable prunes candidate anomalies— those errors that exceed the dynamic threshold — based on the relative separation with respect to non-adjacent anomalies in the sequence. The effects of this parameter are illustrated in Figure 12, highlight an inverse relation with the number of detected anomalies. As illustrated in the figure, a pruning parameter between 0.42 and 0.46 effectively eliminates false positives in the inclination, while still preserving the true detections associated with the eccentricity and semi-major axis. Nevertheless, this outcome is specific to the satellite under analysis and may vary when considering different cases.

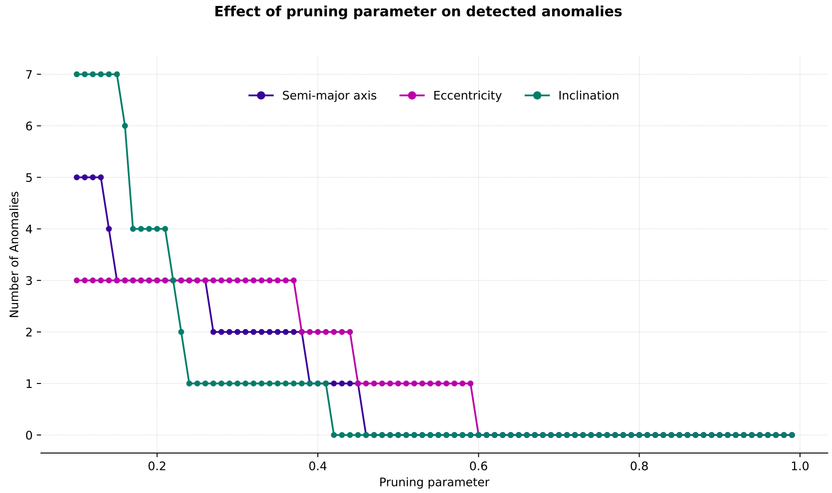

Figure 12: Effect of the selection of the pruning parameter on the number of detected anomalies for the `Sentinel-3A`

CONCLUSIONS

This study demonstrates the capability of detecting anomalous behaviors in space object histories, particularly in identifying new operational regimes and environment change effects. The Bayesian Neural Network introduces an additional layer of adaptivity through noise prediction, which is crucial when dealing with noisy datasets as the TLEs. In fact, the integration of the aleatoric uncertainty is crucial to limit the number of false positives that may arise from abrupt yet nominal changes (e.g. routine impulsive maneuvers). Future works could also explore different Neural Network architectures such as Graph Neural Networks (GNN) and Transformers, which are currently reaching state-of-the-art performances in multiple disciplines. In addition, to avoid manual selection of the hyperparameters of the presented detection method, clustering techniques or unsupervised methods could be investigated and jointly employed with the dynamic thresholding algorithm. To further improve the algorithm, one promising path could involve the integration of an enhanced PoL dataset, containing labeled maneuver characteristics (*e.g.* intensity, frequency, intent). This labeled data could serve as an additional source of information for the presented algorithm, enabling the characterization of anomalies in addition to their detection. Advancing this discipline is nevertheless fundamental for enhancing Space Situational Awareness (SSA) capabilities, build up more comprehensive and detailed catalogs to be used for many other applications, such as conjunction analysis, orbit determination and more.

REFERENCES

[1] D. J. Kessler and B. G. Cour-Palais, “Collision frequency of artificial satellites: The creation of a debris belt,” *Journal of Geophysical Research: Space Physics*, Vol. 83, No. A6, 1978, pp. 2637–2646. _eprint: https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1029/JA083iA06p02637, https://doi.org/10.1029/JA083iA06p02637.

[2] European Space Agency, “ESA’s Annual Space Environment Report,” technical Report, Space Debris Office, Mar. 2025.

[3] R. P. Patera, “Space Event Detection Method,” *Journal of Spacecraft and Rockets*, Vol. 45, May 2008, pp. 554–559, 10.2514/1.30348.

[4] T. Kelecy, D. Hall, K. Hamada, and M. D. Stocker, "Satellite Maneuver Detection Using Two-line Element (TLE) Data," *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2007.

[5] S. Lemmens and H. Krag, "Two-Line-Elements-Based Maneuver Detection Methods for Satellites in Low Earth Orbit," *Journal of Guidance, Control, and Dynamics*, Vol. 37, May 2014, pp. 860–868, 10.2514/1.61300.

[6] R. Mital, K. Cates, J. Coughlin, and G. Ganji, "A Machine Learning Approach to Modeling Satellite Behavior," *2019 IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT)*, Pasadena, CA, USA, IEEE, July 2019, pp. 62–69, 10.1109/SMC-IT.2019.00013.

[7] T. G. Roberts, *Geosynchronous Satellite Maneuver Classification and Orbital Pattern Anomaly Detection via Supervised Machine Learning*. PhD thesis, Massachusetts Institute of Technology, Department of Aeronautics and Astronautics, 2021.

[8] R. Cipollone, I. Leonzio, G. Calabrò, and P. Di Lizia, "An LSTM-based Maneuver Detection Algorithm from Satellites Pattern of Life," *2023 IEEE 10th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, Milan, Italy, IEEE, June 2023, pp. 78–83, 10.1109/MetroAeroSpace57412.2023.10189993.

[9] R. Cipollone, C. Frueh, and P. D. Lizia, "LSTM-based maneuver detection for resident space object catalog maintenance," *Neural Computing and Applications*, 2025.

[10] R. Kato, T. Sakaue, D. Mori, M. Tanaka, T. Sano, K. Taya, M. Ebara, R. Nakazawa, J. Yoshida, and R. Togawa, "Validity Evaluation of Anomaly Detection Using LSTM AutoEncoder for Maneuver Detection," *Advanced Maui Optical and Space Surveillance Technologies Conference (AMOS)*, 2023.

[11] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, and T. Söderström, "Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding," *CoRR*, Vol. abs/1802.04431, 2018.

[12] S. Hochreiter, "The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions," *International Journal of Uncertainty Fuzziness and Knowledge-Based Systems*, Vol. 6, April 1998, pp. 107–116, 10.1142/S0218488598000094.

[13] E. Hüllermeier and W. Waegeman, "Aleatoric and epistemic uncertainty in machine learning: an introduction to concepts and methods," *Machine Learning*, 2021, pp. 457–506, 10.1007/s10994-021-05946-3.

[14] A. Kendall and Y. Gal, "What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision?," 2017.

[15] S. W. R. and, "Control Chart Tests Based on Geometric Moving Averages," *Technometrics*, Vol. 1, No. 3, 1959, pp. 239–250, 10.1080/00401706.1959.10489860.

[16] D. Neil, M. Pfeiffer, and S.-C. Liu, "Phased LSTM: Accelerating Recurrent Network Training for Long or Event-based Sequences," 2016.

[17] F. N. Fritsch and J. Butland, "A Method for Constructing Local Monotone Piecewise Cubic Interpolants," *SIAM Journal on Scientific and Statistical Computing*, Vol. 5, No. 2, 1984, pp. 300–304, 10.1137/0905021.

[18] A. Savitzky and M. J. E. Golay, "Smoothing and Differentiation of Data by Simplified Least Squares Procedures.," *Analytical Chemistry*, Vol. 36, No. 8, 1964, pp. 1627–1639, 10.1021/ac60214a047.

[19] A. Graves, S. Fernández, and J. Schmidhuber, "Bidirectional LSTM Networks for Improved Phoneme Classification and Recognition.," *Artificial Neural Networks: Formal Models and Their Applications - ICANN 2005, 15th International Conference*, 01 2005, pp. 799–804.