Machine Learning Based Orbital Behavior Modeling and Anomaly Detection Using TLE Data

Abstract

Continuous monitoring of satellite orbits is essential for Space Situational Awareness (SSA), mission safety, and detection of unexpected or abnormal behavior. Traditionally, this task relies on physics-based propagation models such as SGP4. In this work, we present a purely data-driven approach that learns orbital evolution patterns directly from historical Two-Line Element (TLE) and CCSDS OMM data obtained from Space-Track.org. A deep learning model based on Long Short-Term Memory (LSTM) networks is trained to predict the next orbital state from a sequence of past states. The prediction residual is then monitored over time and used for statistical anomaly detection using a three-sigma rule. The system is further supported by a comprehensive visualization dashboard for orbital parameter trend analysis and error diagnostics. Experiments performed on a fleet of 59 Indian satellites demonstrate that the proposed method can automatically identify stable satellites, maneuvering behavior, and major mission phase changes without using any physics-based propagation model inside the learning system.

Keywords: Satellite Orbits, TLE, LSTM, Anomaly Detection, Space Situational Awareness, Machine Learning

1. Introduction

Modern satellites operate over long periods and experience a variety of orbital changes such as atmospheric drag induced decay, station-keeping maneuvers, orbit corrections, and sometimes major mission phase changes like orbit raising or disposal maneuvers. Continuous monitoring of such behavior is essential for Space Situational Awareness (SSA), mission safety, and operations monitoring. Traditionally, orbit prediction and monitoring relies on physics-based propagation models such as SGP4. While these models are highly accurate, they require explicit physical modeling and parameter tuning.

In this project, we explore a purely data-driven approach using Machine Learning and Deep Learning. We build a system that learns the normal orbital evolution pattern of a satellite from historical TLE data, predicts the next orbital state using a deep learning model, monitors the prediction residual over time, detects abnormal orbital behavior using

statistical analysis of prediction error, and performs detailed orbital parameter trend analysis through a visualization dashboard.

2. Data Source

The data used in this work is obtained from Space-Track.org, specifically from the GP History and CCSDS OMM history services. Each record contains one EPOCH (timestamp) and one complete set of orbital elements. For each satellite, a separate data.csv file is constructed containing fields such as Inclination, Right Ascension of Ascending Node, Eccentricity, Argument of Perigee, Mean Anomaly, Mean Motion, and BSTAR. Each row corresponds to one historical orbital state of the satellite.

This entire project was made possible due to the availability of historical orbital data from Space-Track.org, and we express special thanks and acknowledgement to Space-Track.org for providing access to this invaluable dataset for research and educational purposes.

3. Problem Statement

Given a time-ordered sequence of historical orbital elements of a satellite, the objectives are to learn the normal orbital evolution pattern, predict the next orbital state using a deep learning model, monitor the prediction error (residual) over time, flag abnormal behavior when this error becomes statistically large, and analyze long-term orbital parameter trends and error behavior visually.

4. Feature Engineering and Data Preparation

From each TLE/OMM record, a seven-dimensional feature vector is constructed: [i, Ω, e, ω, M, n, B*], where i is inclination, Ω is right ascension of ascending node, e is eccentricity, ω is argument of perigee, M is mean anomaly, n is mean motion, and B* is the drag term.

The data is sorted by time (EPOCH) and normalized using standard score normalization to ensure stable training and balanced influence of all features. A sliding window of size K = 10 is used. For each time index t, the input consists of the previous ten states and the target is the next state. This converts the learning problem into a sequence-to-one regression task.

5. Model Architecture

A Long Short-Term Memory (LSTM) neural network implemented in PyTorch is used. The network has an input size of 7, hidden size of 64, two LSTM layers, and an output size of 7

corresponding to the predicted next orbital state. The model learns a nonlinear mapping from a sequence of past orbital states to the next state.

6. Training Pipeline

The data preparation script loads and preprocesses the data, builds sliding windows, splits the dataset into 80% training and 20% testing sets, and saves the processed arrays and scaler. The training script trains the LSTM model using Mean Squared Error loss and the Adam optimizer and saves the trained model. The evaluation script loads the trained model, performs prediction on the test set, and computes the prediction residual as the L2 norm between predicted and true state vectors.

7. Evaluation Metrics

From the error sequence, the mean error, median error, 95th percentile error, maximum error, standard deviation, and anomaly count are computed and stored for further analysis and comparison.

Calculations and Mathematical Formulation

Let the orbital state vector at discrete time index t be defined as:

$x(t) \in R^7$

$x(t) = [\, i(t), \Omega(t), e(t), \omega(t), M(t), n(t), B^*(t) \,]$

where i is inclination, Ω is right ascension of ascending node, e is eccentricity, ω is argument of perigee, M is mean anomaly, n is mean motion, and B* is the drag term.

Each feature is standardized using z-score normalization:

$x\_norm = (x - \mu) / \sigma$

where μ and σ are the mean and standard deviation computed from the training dataset.

A sliding window of length K = 10 is used to construct the input sequence. For time index t, the input tensor is defined as:

$X\_t = [\, x(t-9), x(t-8), ..., x(t) \,] \in R^{10\times7}$

The prediction target corresponding to this input is:

$$Y_t = x(t+1)$$

The LSTM network learns a nonlinear mapping from the past 10 states to the next state:

$$\hat{x}(t+1) = f_\theta(X_t)$$

The model is trained by minimizing the Mean Squared Error (MSE) loss over the training set:

$$L = (1 / N) \cdot \Sigma || \hat{x}_i - x_i ||^2$$

where N is the number of training samples, $\hat{x}_i$ is the predicted state vector, and x_i is the true state vector.

During evaluation, the prediction residual (error) at time t is computed as the L2 norm:

$$e(t) = || \hat{x}(t) - x(t) ||_2$$

Let μ_e and σ_e denote the mean and standard deviation of the residual sequence e(t) over a reference window. The anomaly detection threshold is defined using a three-sigma rule:

$$T = \mu_e + 3 \cdot \sigma_e$$

A time step is flagged as anomalous if:

$$e(t) > T$$

8. Anomaly Detection Method

Normal orbital behavior is assumed to produce small residuals, while abnormal behavior results in large residuals. A statistical three-sigma rule is used to define the anomaly

threshold as mean plus three times the standard deviation of the error. Any time step whose error exceeds this threshold is flagged as anomalous.

9. Orbital Parameter and Error Analysis Dashboard

A visualization dashboard is developed to plot RAAN, inclination, mean motion, altitude, eccentricity, BSTAR, and prediction error versus time. Additional plots show relationships between error and orbital parameters and allow visual comparison between ML-based prediction behavior and traditional SGP4-based propagation. This enables physical interpretation and visual validation of detected anomalies.

10. Experimental Results and Interpretation

Three typical behaviors are observed. Stable satellites show smooth error curves with very few spikes and gradual orbital evolution. Maneuvering satellites show frequent sharp spikes and many anomalous points, corresponding to station-keeping maneuvers. Satellites undergoing mission phase changes show one or more very large spikes followed by a long period of elevated error and eventual stabilization, indicating major orbital regime changes.

11. Fleet-Level Processing

The same processing pipeline is applied to a fleet of 59 Indian satellites. Each satellite is processed independently, and the results are aggregated into a fleet-level comparison and analysis dashboard.

12. Tech Stack

The system is implemented using Python with NumPy, Pandas, Scikit-learn, PyTorch, Matplotlib, and Streamlit.

13. Discussion and Limitations

The system uses TLE data, which is noisy and not a precise ephemeris. The time sampling is irregular, and the model does not identify the physical cause of anomalies; it only flags abnormal behavior. The ML model is not intended for long-term orbit propagation but rather for short-term prediction and behavioral analysis.

14. Conclusion

This work demonstrates that deep learning models can learn orbital evolution patterns directly from historical TLE data and can be effectively used for automated orbital behavior modeling, error diagnostics, and anomaly detection. Supported by a comprehensive visualization framework, the system provides both statistical and physical insight into satellite behavior without using any physics-based propagation model inside the machine learning pipeline.

Acknowledgement

The authors would like to specially thank Space-Track.org for providing access to historical TLE and OMM data, without which this project would not have been possible.

RAHUL JASWAL