Systematic TLE Data Improvement by Neural Network for Most Cataloged Resident Space Objects

Hilaire Bizalion[1,2], Anteo Guillot[1], Alexis Petit[1], and Romain Lucken[1]

[1]Share My Space, 32 boulevard du Port 95032 Paris, FRANCE. , Email: romain.lucken@sharemyspace.space
[2]École polytechnique, Route de Saclay 91128, Palaiseau, FRANCE , Email: hilaire.bizalion@polytechnique.edu

August 10, 2021

Summary

This paper shows the performance of a neural network trained on more than 4500 inactive resident space objects (RSO) on 10 months of orbital data, on all types of orbits. It is shown that the state vector errors are corrected by at least 40% for 70% of the TLE and that the residual error distributions are well described by a Student's t-distribution for which covariance elements are defined consistently.
Keywords: *TLE improvement, orbital data catalogue, neural network, multi-layer perceptron.*

1 Introduction

Two-line elements (TLE) released by the US Air Force are the most complete source of information for space situational awareness in the public domain. They are used in a wide range of contexts that include navigation of nanosatellites, ground communication, and collision risk awareness. However, the TLE data is known for having significant bias and errors. Predicting the bias of TLE can be a way to correct them automatically and hence optimize the systems that rely on them at low cost, for instance for reentry calculations and possibly for collision avoidance. As shown by previous publications [1–3], TLE contain a systematic bias that is correlated with environmental parameters, including the position of an object along its trajectory. Ly et al. [1] have demonstrated that up to 80% of the TLE error could be predicted by polynomial regressions for the GPS satellites, while Lucken et al. [2] have presented some preliminary results for machine learning on LEO satellites while taking into account solar activity parameters. Levit and Marshall [3] employed accurate orbit determination to improve the accuracy of TLE extrapolation over multiple days. Peng et al. also employed a multi-layer perceptron (MLP) to model the TLE error for a limited set of objects and in controlled conditions. The novelty of the present paper is to apply machine learning techniques on a large set of objects across all orbits to correct TLE data massively.

First, we highlight the biases that exist in TLE positions, relative to SP ephemerides. In the next section, we show that a neural network enables us to infer these biases based on well chosen parameters. Finally, we will extract covariance matrices from this training pipeline. We will show that the trained neural networks can then be used to improve the state vectors and derive meaningful covariance elements for objects whose TLE data are known, when no other source of information is available.

2 Inputs and outputs characterisation

During the preliminary study, 12 parameters were found to have an influence on the resulting difference between the TLE and the precise state vectors extracted from the SP ephemerides. The orbital elements are computed using SGP4 and Space-Track TLEs, and the other parameters are either computed using the argument of latitude, the longitude and the latitude (λ, ϕ, θ), or using Astropy Python library [4]. These parameters are then stored in a database for quicker further access. The input vectors are hence of dimension 12 and there are noted

$$x = (a, e, i, \omega, \Omega, \nu, \lambda, b^*, \phi, \theta, D_{\text{Moon}}, D_{\text{Sun}}) \quad (1)$$

A set of vectors (x_n) will be referenced to as X. For example, X_{input} designates a set of input vectors. X_T designates the training input sample, and X_V the validation input sample.
For characterizing the TLE bias, the difference between the state vector obtained from the TLE after conversion with SGP4 and a fifth order polynomial interpolation of the SP ephemerides at the TLE epoch is computed in the local *RSW* frame [5] relative to the accurate state vector. The output data is then stored in the database.
The input and output data was normalized to increase the performance of the training algorithm. In order to improve the computation and training performances, outliers are suppressed using the following rule:

$$Y_{filtered} = \{y \in Y_{output} \text{ where } ||y|| < y_{max}\} \quad (2)$$

having $y_{max} = Q^3 + 3 \times \text{IQR}$, Q^3 being the third quartile and IQR being the inter quartile range, as proposed in *Exploratory Data Analysis* (Tukey, 1977) [6]. Therefore, inconsistent data (e.g. too high error due to an incorrect SGP4 conversion, a corrupted TLE, or a maneuver) can be safely eliminated to ensure clean training data. Data is then cleaned by discarding values whose Mahalanobis distance [7] to the mean position-velocity error is above 5 standard deviations.

3 Neural Network Training

A neural network that has been implemented is a multi-layer perceptron (MLP) with three hidden layers of 2000, 4000 and 2000 neurons respectively, defined using *PyTorch* [8] tensor-based architecture. Each layer consists of a linear layer, followed by ReLU non-linear stage.
The size of the neural network has been chosen in order to match the number of connections, and the number of learning points and the learning hyperparameters such as batch size, and learning rate using Bengio's (2012) [9] recommendations for MLP usage. The loss value $\mathcal{L}_n = \frac{1}{b}\sum_{x_i \in X}(\epsilon_n(x_i))^2$ is the first performance indicator. It should decrease as n increases as it represents the squared distance between the expected output and the prediction.
The histogram of the residual error $\{\epsilon(x_i), x_i \in X\}$, compared to the histogram of the initial error $\{y_i \in Y\}$ can also be useful to visualize the effect of the neural network on the uncertainty of position of the object: in fact, the new error on the position of an object x_i is this residual error $\epsilon(x_i)$.
Finally, the correlation map $y_i \mapsto H_n(x_i), x_i, y_i \in X \times Y$ provides a visual representation of the quality of the model. It should be close to the first bisector. However, if it is too linear, there is a risk that the model has overfitted the training set. If it is not linear and if the figure looks like a cloud of random points, the model underfits and has not learned.
In order to train and test the capacities of the model, a set of debris, decommissioned satellites, breakup residues and rocket second stages has been randomly selected. 4500 of those have been used for training, 500 for validation. The TLE epoch for these objects are between 2020, July 12th to 2021, May 3rd for these objects. The performances of the trained network are shown in Figure 1. A mean improvement of the TLE of about 55% to 60% has been observed, and the ratio between the mean residual error and the mean initial error is calculated to be 45%, reaching up to 70% for some objects. The performances of the neural network on individual objects depend on the semi-major axis a and on the inclination i strongly.

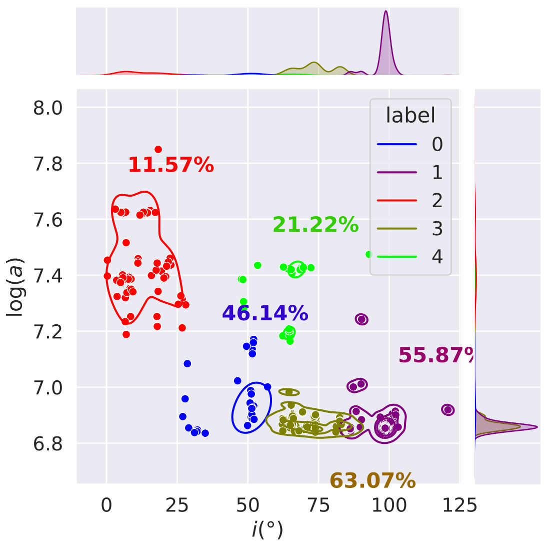

Figure 1: TLE improvement percentage on different clusters of the 500 RSO of the validation set.

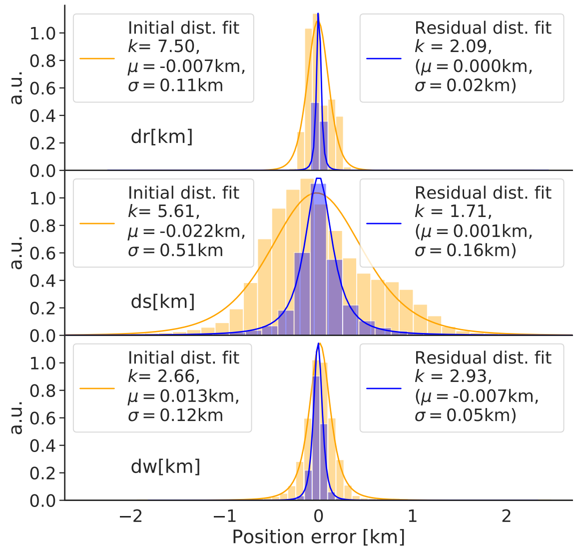

Figure 2: Residual error (blue) for the validation dataset compared to the initial error (orange), for the position coordinates. The solid lines represent the fits performed using the Student's t-distribution law

4 Use of the trained Neural Network for TLE improvement

The effect of the correction on the individual coordinates can be seen in Figure 2. In this figure, the distributions of the initial error are plotted together with the residual error for the six coordinates dr, dw, ds, dv_r, dv_w, and dv_s. The error distributions are fitted with a Student's t-distribution [10] and show that the covariance elements are not only smaller, but also better defined.

A reduction of 60 to 80% of the full width at half maximum between the initial position and velocity error, and the residual error after correction by the neural network is observed.

Figure 3 shows the proportion of corrected TLE as a function of the maximum position error. The residual median maximum error (frequency = 50%) is here equal to 0.16 km whereas the initial median maximum error is equal to 0.45 km.

The trained network is also able to correct the TLE error for manoeuvering satellites between 27% and 80%.

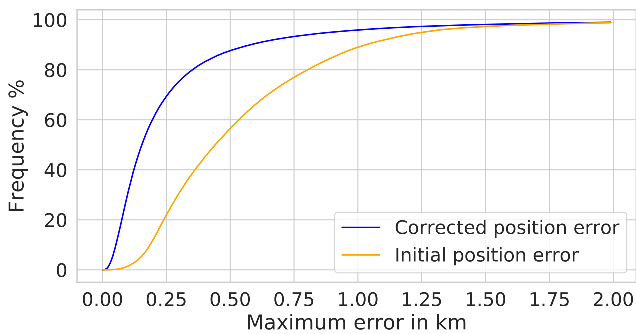

Figure 3: Frequency of TLE having a maximal error as a function of this error.

5 Conclusion

After showing the strong correlation that existed between the position of objects in space and their TLE bias, we used MLP to improve TLE for many objects with a good accuracy. On the contrary to previously cited studies, we used a large number of objects extracted from the public catalog across all types of orbits, and validated the algorithm against similarly diverse objects. This method can be used to reduce the systematic error of more than 70% of the TLE by at least 40%, and yields a much more significant covariance estimator.

6 Acknowledgements

This project was partially funded by BPI France in the frame of the *CATUSSA* project. The authors would like to thank the teams of Telespazio France and Thales Services Numériques.

7 References

[1] Ly, D., Lucken, R., and Giolito, D. Correcting TLEs at epoch: Application to the gps constellation. *Journal of Space Safety Engineering* **7** 08 (2020).

[2] Lucken, R., Viens, A., Ly, D., Giolito, D., Petit, A., Deleflie, F., and Fontaine, D. Investigation and prediction of two-line element errors for objects in the ionosphere, (2020).

[3] Levit, C. and Marshall, W. Improved orbit predictions using two-line elements. *Advances in Space Research* **47**, 1107–1115 04 (2011).

[4] Astropy website. `https://www.astropy.org/`.

[5] Vallado, D. A. and McClain, W. D. *Fundamentals of astrodynamics and applications*, volume The space technology library. Microcosm Press, 4th ed edition, (2013).

[6] Tukey, J. W. *Exploratory Data Analysis*. Addison-Wesley, (1977).

[7] Mahalanobis, P. C. On the generalized distance in statistics. National Institute of Science of India, (1936).

[8] Pytorch website. `https://pytorch.org/`.

[9] Bengio, Y. Practical recommendations for gradient-based training of deep architectures, (2012).

[10] Student. The probable error of a mean. *Biometrika* **6**(1), 1–25 (1908).