Artificial Neural Network–Based Machine Learning Approach to Improve Orbit Prediction Accuracy

Article *in* Journal of Spacecraft and Rockets · June 2018

DOI: 10.2514/1.A34171

2 authors:

Hao Peng
Rutgers, The State University of New Jersey
46 PUBLICATIONS **388** CITATIONS
SEE PROFILE

Xiaoli Bai
Rutgers, The State University of New Jersey
75 PUBLICATIONS **1,106** CITATIONS
SEE PROFILE

JOURNAL OF SPACECRAFT AND ROCKETS
Vol. 55, No. 5, September–October 2018

Artificial Neural Network–Based Machine Learning Approach to Improve Orbit Prediction Accuracy

Hao Peng* and Xiaoli Bai†
Rutgers, The State University of New Jersey, Piscataway, NJ 08854

DOI: 10.2514/1.A34171

A machine learning (ML) approach has been proposed to improve orbit prediction accuracy in previous studies. In this paper, the artificial neural network (ANN) model is investigated for the same purpose. The ANNs are trained by historical orbit determination and prediction data of a resident space object (RSO) in a simulated space catalog environment. Because of ANN's universal approximation capability and flexible network structures, it has been found that the trained ANNs can achieve good performance in various situations. Specifically, this study demonstrates and validates the generalization capabilities to future epochs and to different RSOs, which are two situations important to practical applications. A systematic investigation of the effect of the random initialization during the training and the ANN's network structure has also been studied in the paper. The results in the paper reveal that the ML approach using ANN can significantly improve the orbit prediction.

I. Introduction

THE number of space objects larger than 10 cm is presently approaching 29,000, the estimated population of objects between 1 and 10 cm is about 750,000, and for objects smaller than 1 cm the number exceeds 166 million [1]. These objects distribute in different regions from low to very high Earth orbits, and threatens the safety of satellites in or crossing these regions [2]. At the heart of the challenges for Space Situational Awareness (SSA) is to predict resident space object's (RSO's) orbit efficiently and accurately. Current orbit predictions that are solely grounded on physics-based models, however, fail to achieve required accuracy for collision avoidance and have already led to collisions. Several incidents have painfully demonstrated the limitations of our current orbit prediction capability, such as the February 2009 collision of a U.S. Iridium communications satellite and a Russian Cosmos 2251 communications satellite [3]. In the coming decades, both the number of space objects and the number of conflicts between these objects will increase rapidly. Our ability to generate accurate and timely predictions of the trajectory of RSOs is the key factor of the SSA system.

The limitation of current prediction approach arises from the assumed dynamic models, measurement errors, orbit determination errors, and others. In reality, we can only get limited information from the physical world, while these limited information will be further twisted by various assumptions during the modeling process. For example, whatever accurate measurements or parameters we have about the solar activity, the model is only statistically approximating the data with limited variables that appear to be dominant. Another good example is the atmosphere drag model that usually causes great uncertainty for orbit predictions of LEO objects.

Considering that certain information embedded in historical data has not been used during the conventional orbit prediction process, we have proposed a machine learning (ML) approach to capture and extract these extra information in previous studies [4–6]. The ML approach has presented a different modeling and prediction capability compared with the physical model based approach. The improvement on orbit prediction can be made without explicitly modeling forces or perturbations. Instead, models of orbit prediction errors are directly learned based on large amounts of historical data.

In one study, we have discovered that the information of the area-to-mass ratio (AMR) can be recovered from the consistent error between two estimated states in a catalog [6]. In another study using a simulation-based space catalog environment as the test bed, we have discovered three types of generalization capability for the proposed ML approach [4,7]. In the above studies, the support vector machine (SVM) method [8–10] has been chosen as the specific ML algorithm. The SVM is a universal approximation method, which means that theoretically it could approximate any continuous function with arbitrary accuracy. We have found that the SVM can capture the underlying relationship between the available parameters and the prediction error, and can reduce the orbit prediction error greatly.

In this paper, as an extension to the previous studies, we have examined the artificial neural network (ANN) to improve orbit prediction accuracy. The ANN is basically multiple layers of neurons that connect to neurons at the previous layers, and the inputs are processed from the first to the last layer, by weighted summation and nonlinear mapping at each neurons.

Recently, as represented by the great achievement of AlphaGo [11], attentions have been drawn on a specialized kind of ANN, the deep neural network, which is good at tasks such as classification, decision making, or picture comprehensions. It is well-known that the ANN can be superior to human performance in various classification problems, where the outputs are discrete categories. The ANN has also been used in other fields dealing with continuous outputs, that is, the regression problem. The ANN has proven to be a powerful universal approximation method [10,12–14], able to approximate any smooth functions with adequate neurons and layers. In fact, it has been a long time since the function approximation capability of shallow ANNs with just one hidden layer has been applied to various problems in aerospace fields. Williams has trained a shallow neural network to predict the future solar activity based on historical data, and his experiment shows that using this technique the prediction of the lifetime of satellite is more accurate [15]. Considerable improvements have been reported when compared with the conventional linear regression. Macpherson et al. have extended the method to predict both the solar and geomagnetic activity [16]. They have used ANNs with just 1 hidden layer of no more than 10 neurons. Lu et al. have proposed a pruning learning strategy to find the minimal radial basis function neural network [17], which has been also used in control and function approximation problems [18,19]. The ANN method has also been integrated into Kalman filters to improve the accuracy of GPS applications when there are model uncertainties [20]. Recently, Sanchez-Sanchez and Izzo have used deep neural networks to learn the solution of an optimal landing problem, so that real-time on-board trajectory planning can be

Presented as Paper 2018-1966 at the 2018 Space Flight Mechanics Meeting, Kissimmee, FL, 8–12 January 2018; received 17 December 2017; revision received 27 February 2018; accepted for publication 4 March 2018; published online 31 May 2018.

*Postdoctoral Associate, Department of Mechanical and Aerospace Engineering; Astro.H.Peng@gmail.com.

†Assistant Professor, Department of Mechanical and Aerospace Engineering; xiaoli.bai@rutgers.edu.

achieved [21,22]. Mereta et al. have studied various machine learning methods, including the ANN, to provide initial guess of the optimal final mass for low-thrust trajectory design between asteroids [23].

We make three contributions in this paper: 1) the ANN method is demonstrated to have good performance on reducing orbit prediction errors; 2) the effect of the number of neurons and hidden layers has been explored; and 3) ANN's generalization capabilities to future epochs and different RSOs have been rigorously investigated. Furthermore, combining the performance of the ANN method in this paper and that of the SVM method in our previous studies, we verify that the proposed ML approach is feasible and correct, with great potential to advance practical applications.

This paper is organized as follows. In Sec. II, we briefly summarize the previously proposed ML approach, and introduce the basic concept of the ANN. In Sec. III, we present the generalization capability study of the trained ANN models onto orbit predictions at future epochs, and the effect of randomness and network structure are investigated. In Sec. IV, the generalization capability of the trained ANN models to different RSOs are investigated. At last, in Sec. V, we summarize the paper and provide insights and suggestions on future research.

II. Backgrounds

In this section, we first briefly review the ML approach to improve orbit prediction accuracy using a simulated space catalog environment. More details can be found in our previous studies [4,5]. Next, design of learning variables and the dataset structure are presented. Last, basic concepts of the ANN are introduced.

A. Machine Learning Approach for Orbit Prediction Problem

The concept of the proposed ML approach to directly modify the orbit prediction is illustrated in Fig. 1. When the RSO enters the detection range of the ground station, we will generate an estimate of the state of the RSO at a certain epoch using a conventional orbit determination method. This estimated state will deviate from the true state, which is usually referred to as an estimation error. Then the estimated state is propagated, usually under the same dynamics to the determination process, to a future epoch to generate the predicted state, which will further deviate from the true state because of the error propagation. In practice, the atmosphere drag force, the solar radiation pressure, and the shape and attitude motion of the RSO are difficult to be accurately modeled.

With the speculation that the effect of these unmodeled factors is hidden in the historical data of measurements, estimations, and orbit predictions, the ML approach is proposed to discover the hidden relationship between the available variables and the orbit prediction error from historical data. Once the ML model has been well trained, as shown in Fig. 1, the ML-modified state is generated by eliminating the ML-generated error from the original predicted state. This ML-modified orbit prediction is expected to be closer to the true state. We note that the ML approach is not restricted to specific dynamic models or underlying relationships, but can be applied to different catalogs and embedded in current procedures.

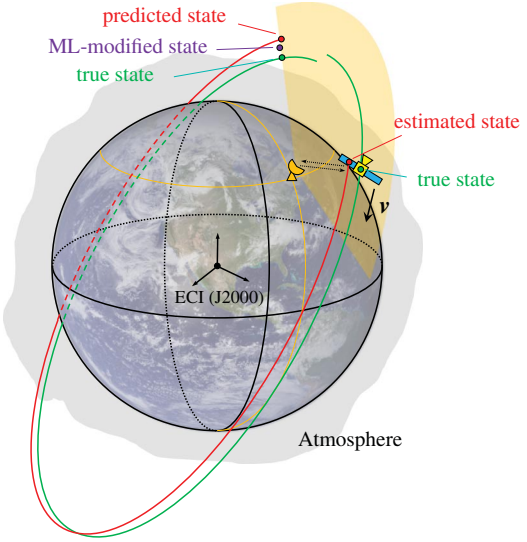


Fig. 1 Illustration of machine learning approach to directly improve the orbit prediction [7].

B. Simulation Environments

The flowchart of our simulation environment for the machine learning approach is demonstrated in Fig. 2. The top four blocks show the sequential simulations of the conventional orbit prediction:

1) The true orbit of an RSO in the "truth" dynamics models
2) The measurement of the RSO from ground stations
3) The estimation process in an assumed dynamic model different from the "truth"
4) The orbit predictions are generated

The bottom three blocks show the machine learning enhancement procedure.

This simulation environment is established with the goal that the simulated "truth" dynamic models can capture not only main factors contributing to the orbit prediction error in reality, but also the difference between the assumed models and the truth models.

The setups of the truth and assumed model are summarized in Table 1. The nonspherical effect of the Earth gravity is modeled using spherical harmonic functions, with coefficients provided by the EIGEN-6S model [24]. Third-body perturbations of all major solar system bodies are considered, including the Sun, all the planets, the Pluto, and the Moon. The position of these bodies are provided by DE430 data file from the JPL [25]. The Drag Temperature Model 2000 (DTM2000) model is used to approximate the atmosphere, where the Marshall Solar Activity Future Estimate Solar (MSAFE) data from NASA are used to provide solar activity information, which has significant effect on the density and the speed of the atmosphere. A different atmosphere model, NRLMSISE-00, is used for the assumed model. We note that we just use the two different models to mimic the differences between the empirical atmosphere model and the truth. The solar radiation pressure is calculated with the reference value as 4.56×10^{-6} N/m^2 at 1 AU (149,597,870.0 km) from the Sun. And the effects of the penumbra and eclipse are considered. During the generation of true orbit, a spherical RSO with a constant area-to-mass ratio of 0.05 is assumed, and the drag coefficient C_d and single-parameter reflection coefficient C_r are assumed to be constant.

Three ground radar stations are simulated using topocentric frames to generate tracks of measurement of an RSO, with their parameters summarized in Table 2 [26]. The stations will generate discrete measurements, including the azimuth α, the elevation η, and the range ρ, at each step of the measurement gap, when the target RSO is visible to the ground stations. A series of consecutive measurements are organized as a track, and one track can include measurements collected from different stations if they could all detect the RSO. The measurement errors are simulated as normal distributions with zero biases and standard deviations of σ_α, σ_η, and σ_ρ for the azimuth, elevation, and range, respectively, as summarized in Table 2. The batch Least Square (LS) estimator [27] is used to estimate the state of the RSO at the beginning of each track. In this study, all tracks in the past 12 h are used as the input of the LS estimator. The output of the LS estimator includes the orbit state $\boldsymbol{X}$ and drag coefficient parameter C_d. Then the prediction process is carried out in the same assumed dynamic model. The prediction error is generated by comparing the predicted state and the recorded true state at the same epoch. A maximum prediction duration of $\Delta t_{\max} = 7$ days (1 week) is used in this paper, which is suitable for general surveillance and conjunction analysis of LEO objects. We note that all the simulations are implemented using the Orekit, which is a low-level space dynamics library written in Java [28].

C. Dataset Structure of Machine Learning Approach

At first, we define the notations that will be used throughout the following paper. We use the symbol $\boldsymbol{X}(t)$ to denote the state of the RSO

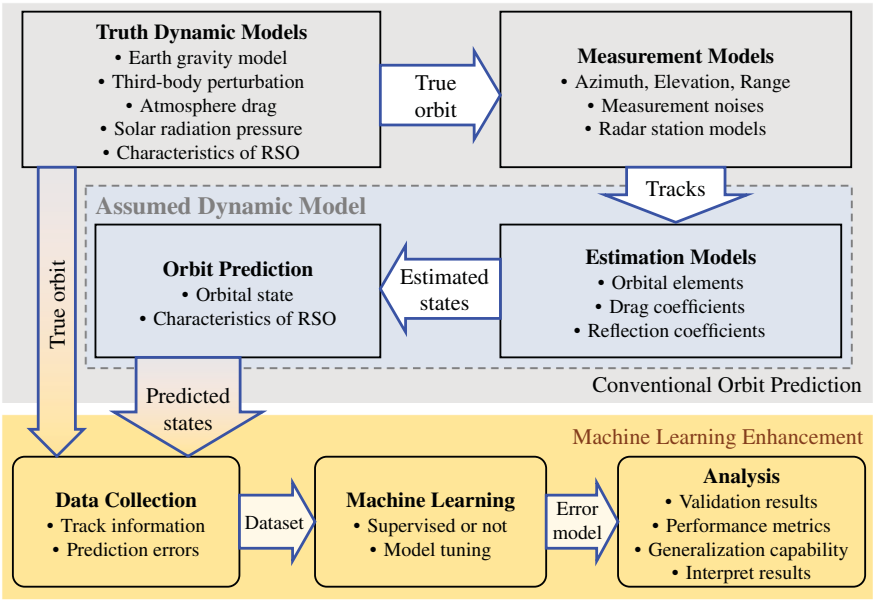


Fig. 2 Flowchart of the simulation environment for the machine learning approach [4].

at time t, without expressing it in a coordinate frame. The state $\boldsymbol{X}(t)$ can be expressed in the classical orbital element (COE) form as ${}^{\text{COE}}\boldsymbol{X}(t) = [a, e, i, \omega, \Omega, \nu]^T$, or in the Earth-centered inertial (ECI) frame as ${}^{\text{ECI}}\boldsymbol{X}(t) = [X, Y, Z, V_X, V_Y, V_Z]^T$. The difference $\delta\boldsymbol{X}(t)$ between two states, $\boldsymbol{X}_1(t)$ and $\boldsymbol{X}_2(t)$, at the same epoch t will be expressed in the RSW frame as ${}^{\text{RSW}}\delta\boldsymbol{X}(t) = [\delta x, \delta y, \delta z, \delta v_x, \delta v_y, \delta v_z]^T$ in this paper, where x axis (radial) is the radial direction, the y axis (along-track) is perpendicular to the x axis in the orbital plane and points to the inertial velocity direction, and the z axis (cross-track) is along the angular momentum direction [29]. The above symbols without any modifier indicate that they are true value of the orbit, such as the true state $\boldsymbol{X}(t_i)$. The hat over a symbol is used to indicate that it is an estimated value, such as the estimated state $\hat{\boldsymbol{X}}(t)$ or drag coefficient $\hat{C}_d$. We use an additional time variable after a semicolon to indicate that this value is based on a previous estimate, such as $(t_j; t_i)$ in $\hat{\boldsymbol{X}}(t_j; t_i)$, which indicates that this state is predicted at t_j based on $\hat{\boldsymbol{X}}(t_i)$.

Table 1 Parameters of the "truth" model used to generate orbits and measurements, and the assumed model used in the estimation and prediction

Parameters	Truth model	Assumed model
Earth shape	WGS84	WGS84
Harmonic gravity field	40×40	10×10
Third-body perturbation	Sun + solar planets +Pluto + the Moon	Sun + Jupiter +the Moon
Atmosphere model	DTM2000	NRLMSISE-00
Solar activity	MSAFE	$(F_{10.7}, K_p) = (150.0, 3.0)$

Table 2 Ground-based radar stations modeled in the paper [4,26]

Station	Eglin, FL	Clear, AK	Kaena point, HI
Latitude, deg	30.57	64.29	21.57
Longitude, deg	−86.21	−149.19	−158.27
Altitude, m	34.7	213.3	300.2
Max range ρ, m	13,210	4,910	6,380
Elevation η, deg	1–90	1–90	0–90
σ_ρ, m	32.1	62.5	92.5
σ_α, deg	0.0154	0.0791	0.0224
σ_η, deg	0.0147	0.024	0.0139

The choice of learning variables is identical to our previous study on SVM's limits [7]. To be self-consistent, this is briefly summarized here. With above notations, all learning variables at the current epoch t_i are collected and represented as the vector $\boldsymbol{\Lambda}(t_i)$, whose components include the following:

1) Prediction duration $\Delta t = t_j - t_i$ to the future epoch t_j $(t_i < t_j)$

2) Estimated state $\hat{\boldsymbol{X}}(t_i)$ at the current epoch t_i, expressed as both ${}^{\text{COE}}\hat{\boldsymbol{X}}(t_i)$ and ${}^{\text{ECI}}\hat{\boldsymbol{X}}(t_i)$

3) Estimated drag coefficient $\hat{C}_d(t_i)$ at the current epoch t_i

4) Maximal measured elevation in the current ith track η, and the corresponding range ρ and azimuth α at that epoch, denoted by $\boldsymbol{\theta}_i = [\rho, \alpha, \eta]^T$

5) Predicted state $\hat{\boldsymbol{X}}(t_j; t_i)$ at the future epoch t_j, based on the current epoch $\hat{\boldsymbol{X}}(t_i)$, expressed as both ${}^{\text{COE}}\hat{\boldsymbol{X}}(t_j; t_i)$ and ${}^{\text{ECI}}\hat{\boldsymbol{X}}(t_j; t_i)$

Then, all the target variables are:

6) True predicted error $\boldsymbol{e}(t_j; t_i)$ at the future epoch t_j, expressed as ${}^{\text{RSW}}\boldsymbol{e}(t_j; t_i) = [e_x, e_y, e_z, e_{vx}, e_{vy}, e_{vz}]^T$.

Because ${}^{\text{RSW}}\boldsymbol{e}$ has six components, totally six ANN models will be trained for each components. We note that these learning variables are chosen through a trial-and-error process, and the result is not meant to be optimal.

With the chosen learning and target variables, the illustration of the training and testing process of the ML approach is demonstrated in Fig. 3. During the collection of training data, each estimated state is propagated to the epoch of all following estimates with $\Delta t < \Delta t_{\max}$; then the learning variable $\boldsymbol{\Lambda}(t_i)$ and the true prediction error $\boldsymbol{e}(t_j; t_i)$ are collected as a data point $(\boldsymbol{\Lambda}(t_i), \boldsymbol{e}(t_j; t_i))$. The total dataset will be used to train ANN models to approximate $\boldsymbol{e}(t_j; t_i)$ based on $\boldsymbol{\Lambda}(t_i)$. After the ANN model has been trained, it can generate the ML-predicted error $\hat{\boldsymbol{e}}_{\text{ML}}(t_j; t_i)$, as shown in Fig. 3. In the ideal situation, $\hat{e}_{\text{ML}}(t_j; t_i)$ should be equal to $\boldsymbol{e}(t_j; t_i)$, leading to that the residual error $\boldsymbol{e}_{\text{res}}(t_j; t_i) = \boldsymbol{e}(t_j; t_i) - \hat{\boldsymbol{e}}_{\text{ML}}(t_j; t_i)$ becomes zero. In practice, the statistical properties of $\boldsymbol{e}_{\text{res}}(t_j; t_i)$ will be analyzed to evaluate the performance of the trained ANN model.

D. Artificial Neural Network

The artificial neural network (ANN) can approximate any smooth transfer functions with adequate neurons and layers [12]. In this subsection, a brief summary of ANN is presented (more details and discussions can be found in Refs. [10,12,13]).

Figure 4 shows a neural network with L layers. The first layer with the learning variables $\boldsymbol{a}^1 = \boldsymbol{\Lambda}(t_i)$ is the input layer, and the last layer that generates the output $\boldsymbol{a}^L$ is the output layer. Intermediate layers

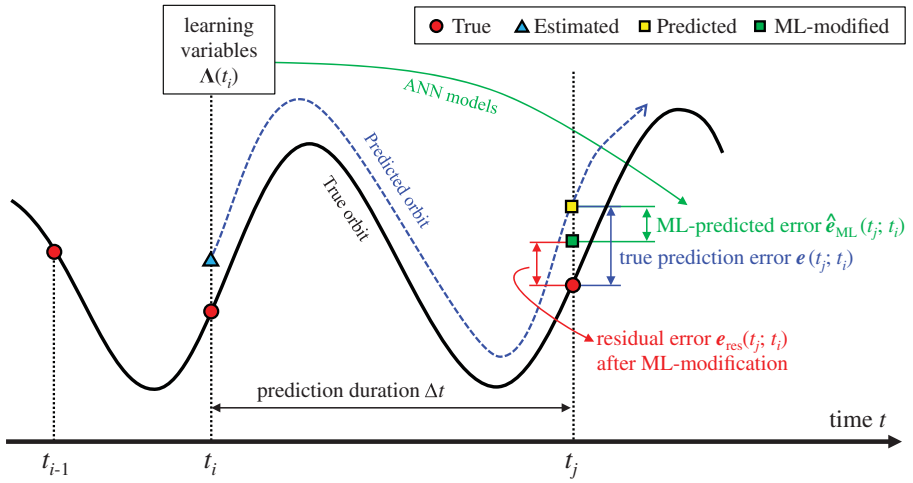


Fig. 3 Illustration of learning and target variables for ANN models [7].

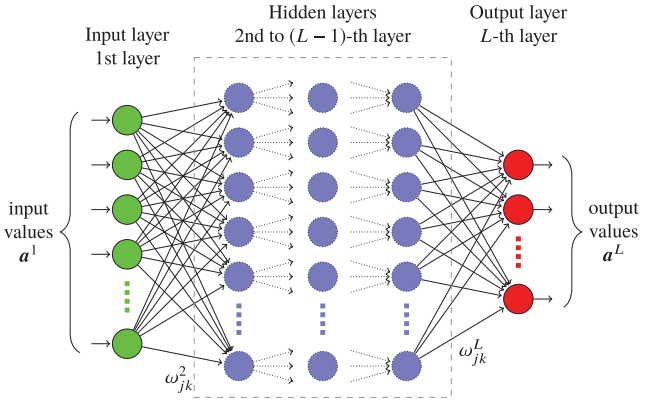


Fig. 4 Illustration of an artificial neural network (ANN) with L layers.

from $l = 2$ to $L - 1$ are usually referred to as hidden layers. The network will be trained to generate outputs $\boldsymbol{a}^L$ as close as possible to the corresponding target variable $\boldsymbol{e}(t_j; t_i)$ for each data point in the training data. In other words, the ANN is trained to capture the underlying relationship of $(\boldsymbol{\Lambda}(t_i, t_j), \boldsymbol{e}(t_j; t_i))$.

Denote the output of the jth neuron in the lth layer by a^l_j, then we have

$$a^l_j = \mathcal{F}^l_j\left(\sum_k \omega^l_{jk} a^{l-1}_k + b^l\right) \tag{1}$$

where a^{l-1}_k is the output of the kth neuron in the $(l-1)$th layer, ω^l_{jk} is the weight from the kth neuron in the $(l-1)$th layer to jth neuron in the lth layer, b^l_j is the bias of the jth neuron in the lth layer, and $\mathcal{F}^l_j(\cdot)$ is the activation or transfer function of the lth layer. This relationship connects the two adjacent layers in the ANN.

The activation function $\mathcal{F}$ can be chosen from many functions, while different choices will affect the capability and performance of the resulted ANN. Depending on the activation functions used at each layer, the ANN model can be used for both classification and regression problems. The orbit prediction problem in this paper is modeled as a regression problem; the log-sigmoid transfer function (`logsig` in MATLAB) is chosen for the hidden layers (layers 2 to $L - 1$),

$$\mathcal{F}^l_j(\xi) = \frac{1}{1 + e^{-\xi}} \tag{2}$$

and the pure linear function (`purelin` in MATLAB) is chosen for the output layer (the Lth layer),

$$\mathcal{F}^L_j(\xi) = \xi \tag{3}$$

The cost function of the ANN is chosen as the mean square of error C_{MSE} (mse in MATLAB):

$$C_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N} C(\boldsymbol{\Lambda}_i) = \frac{1}{N}\sum_{i=1}^{N} \frac{\|\boldsymbol{e}_i - \boldsymbol{a}^L_i\|^2}{2} \tag{4}$$

where N is the number of data point in the training data, $\boldsymbol{\Lambda}_i$ is the ith data point, $\boldsymbol{e}_i$ and $\boldsymbol{a}^L_i$ are the true and ANN-generated error corresponding to $\boldsymbol{\Lambda}_i$, and $\|\cdot\|$ represents the 2-norm. The goal of a training algorithm is to minimize the cost function C_{MSE}. In this study, the back propagation technique is used to solve for the gradient of the cost function with respect to all the weights ω^l_{jk} and the biases b^l_j. And the Levenberg-Marquardt optimization method (trainlm in MATLAB) is used to update the weights and biases of the ANN.

The great flexibility of the ANN also implies that it is very likely to overfit the training data. To avoid the potential overfitting, the training data are divided into a validation data set and a real training data set. The validation data are not used for training. Instead, during the training, the performance of the ANN on the validation data is monitored so that an early stop of the training will be triggered when overfitting is detected according to some criterion. A common indication of overfitting is that the cost function keeps decreasing but that on the validation data starts to increase, which is the criterion used in this paper.

III. Generalization Capability at Future Epochs

As has been proposed and discussed in our previous study [4], for the orbit prediction problem, there exist three types of generalization capabilities of the ML approach:

Type I: Generalization capability within the same dataset. The total dataset is divided randomly into a training set and a testing set. The testing data are within the same time interval as the training set. This is the common practice in machine learning area.

Type II: Generalization capability to future epochs. The testing data are restricted to be in the future epoch with respect to the training data. This reflects the realistic orbit prediction problem.

Type III: Generalization capability to different but nearby RSOs in future epochs. This is a generalization type that can be very important for SSA. Basically, this capability means learning some patterns from a limited number of RSOs, and then applying the knowledge onto other RSOs.

According to our previous results [4,7], the Type I generalization is relatively easy to achieve by the ML approach, and so we will focus on the Type II and III generalization capabilities in this paper.

In this section, the Type II generalization capability will be investigated. Besides of the performance of the ANN, the effect of random initialization and the number of neurons and hidden layers are also systematically explored.

Table 3 Parameters of ENVISAT[a]

Parameters	Values
Name	ENVISAT
NORAD ID	27,386
Orbit	SSO
Launch date	1 March 2002
Altitude	∼796 km
Period	∼100 min
Weight	∼8211 kg
Inclination	∼98.54 deg
Eccentricity	∼0.001165

[a]Extracted from the international laser range services (ILRS) website, https://ilrs.cddis.eosdis.nasa.gov/missions/satellite_missions/current_missions/env1_general.html.

A. Simulation Parameters and Performance Metric

The RSO ENVISAT, which has a Sun-synchronous orbit (SSO), is chosen as the study object in this paper. General information and parameters of ENVISAT are summarized in Table 3. For numerical simulations, the initial state of the orbit of ENVISAT is extracted from the following TLE set, using the standard SGP4 model:

```
0 ENVISAT
1 27386U 02009A 17238.87724141 +.00000000 +00000-0 +13804-4 0 9992
2 27386 098.2254 281.1813 0001340 081.3267 278.8065 14.37901343811233
```

Using this initial state, the orbit of the RSO in the simulated environment for 4 weeks is demonstrated in Fig. 5 in the ECI frame. The orbit is colored by red at the beginning, and is seen to gradually change to blue at the end, which reveals the procession of the orbit due to the perturbation forces.

The metric P_{ML} used to quantify the performance of the trained ANN is defined as the total absolute residual error divided by the total absolute true error, which can be expressed as

$$P_{\mathrm{ML},\xi} = 100\% \cdot \frac{\sum_{i=1}^{n} |e_\xi - \hat{e}_{\mathrm{ML},\xi}|}{\sum_{i=1}^{n} |e_\xi|} \tag{5}$$

where n is the size of the testing data and $\hat{e}_{\mathrm{ML},\xi}$ ($\xi \in \{x, y, z, vx, vy, vz\}$) is the ML-predicted orbit prediction error [4]. This metric shows the percentage of the errors that remain after the ML modification. We note that if the metric is computed using the training data, it actually reflects the learning capability of the ANN.

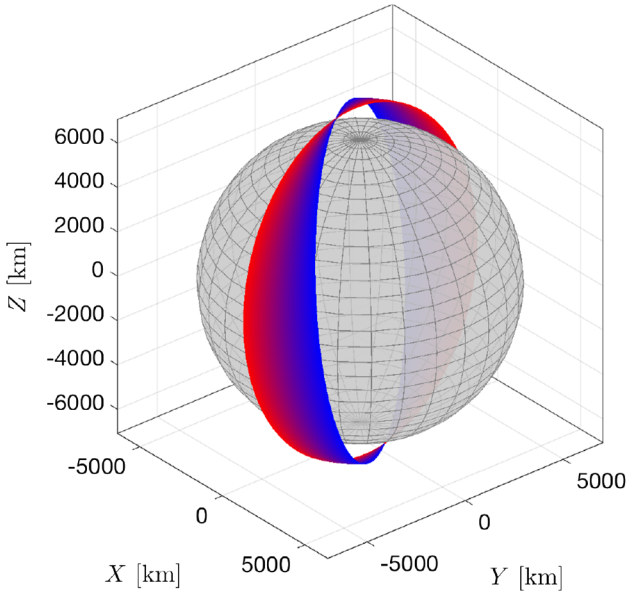


Fig. 5 Orbit of ENVISAT for 4 weeks in ECI frame [7].

More important, according to our experience, this definition can avoid numerical problems with small denominators and thus stably evaluate the average performance of the trained model in most situations.

B. Generalization Performance at Future Epochs

In this subsection, 6 three-layer ANN networks with 1 hidden layer of 20 neurons are trained for each component of the true orbit prediction error $\boldsymbol{e}$. The training data are the historical orbit prediction data ($\boldsymbol{\Lambda}$, $\boldsymbol{e}$) of the RSO in the first 4 weeks, that is, weeks 1–4, and the testing data are the data in the following 1 week, that is, week 5. Additionally, the data in the last day of the training data are used as the validation data for the early stopping criterion, in order to guarantee a better generalization capability. This design leads to that the numbers of data points for the training, validation, and testing data are 39,979, 1633, and 6239, respectively. For simplicity, the similar choice of validation data will be used for the remaining paper without clarification. The early stop parameter is chosen as 20, which means that the training will be terminated if the cost function on the validation data has not been reduced in the following 20 consecutive iterations.

Figure 6 demonstrates the performance of the trained ANN on the training data of the radial axis error e_x. The horizontal axis represents the prediction duration Δt. The vertical axis shows true, ML-predicted, and residual along-track errors e_x. The left plot shows the scatter plot, where black circles represent the true error $e_{1,x}$, green dots represent the ML-predicted error $\hat{e}_{\mathrm{ML},x}$, and red dots represent the residual error $e_{\mathrm{res},x}$ after ML modifications. The right plot shows the errorbar plot of the left scatter plot, where the center point represents the mean value of each clustered group of the error, and the length from the middle of the bar to the top (or the bottom) represents the standard deviation of the corresponding clustered group. For clarity, the three curves have been slightly displaced along the horizontal axis to avoid overlapping. The performance metric $P_{\mathrm{ML}}(e_x) = 7.5\%$ is shown above axes. Because these two plots reveal similar information in most time, for clarity, only the errorbar plot will be demonstrated in the following paper.

About the residual errors for all experiments, we conjecture that they are possibly due to, but not limited to, the following factors: 1) the intrinsic randomness of the data that cannot be completely removed, such as the measurement noise; 2) limited information of errors captured by the current choice of available learning variables; 3) the early stopping criterion introduced to guarantee a better generalization capability; and 4) the theoretical limits of the ANN model, such as the approximation capability with chosen finite number of neurons and layers. Therefore, these factors may not be completely overcome theoretically.

Figure 7 demonstrate the performance of ANNs for all the position and velocity error components on the training data (weeks 1–4), which reveals the learning capability of the ANNs. The mean values of residual error $\boldsymbol{e}_{\mathrm{res}}$ have all been reduced to almost zero, and the standard deviations have also been significantly reduced. The result reveals that the ANN has captured the underlying relationship between learning and target variables in the training data. However, the generalization capability onto future data remains to be examined.

Figure 8 demonstrates the performance of the trained ANN that has been generalized onto the testing data (week 5). For e_x, e_y, e_{vx}, and e_{vy}, both the mean values and the standard deviations have been significantly reduced. The metric P_{ML} are slightly larger than corresponding metric on training data in Fig. 7. This is expected because the testing data have not been used during the training and also may contain information different from the training data. For e_z and e_{vz}, the metrics are 55.9% and 60.0%, respectively. It is clear in the figure that the biases and standard deviations have also been reduced to some extent, although the performances are not as good as that of the other four components.

To have a clearer impression of the capability of the trained ANNs, a comparison of the true (left) and residual (right) errors is presented in Fig. 9. The vertical axes are in logarithmic scale. It is clear that among the magnitudes of three true position errors, we have

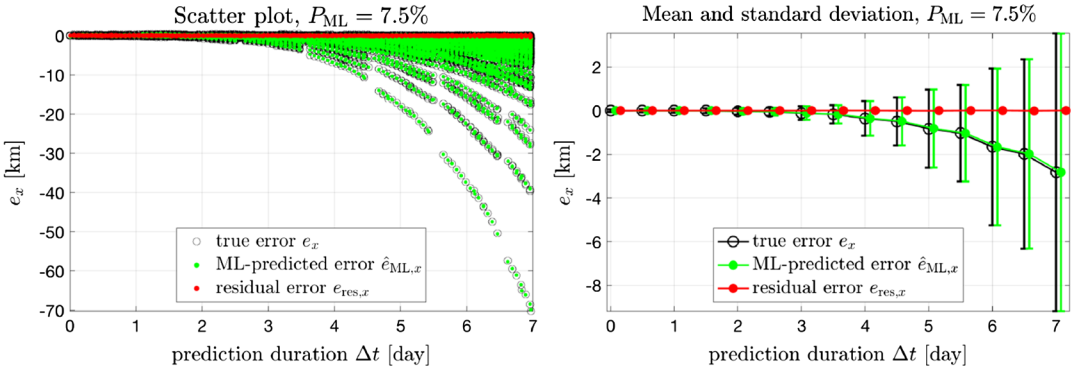


Fig. 6 Example of performance $P_{\mathrm{ML}}(e_x)$ of ANN on the training data (weeks 1–4).

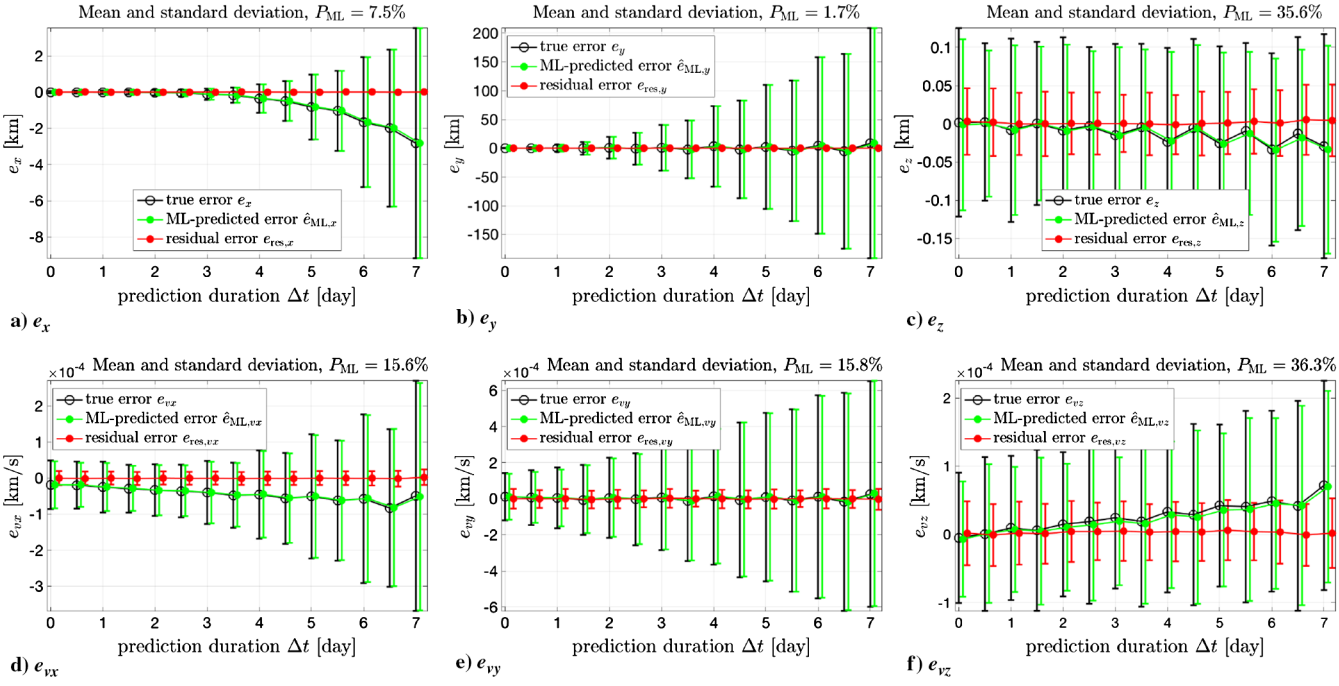


Fig. 7 Performance of ANNs for different components on the training data (weeks 1–4).

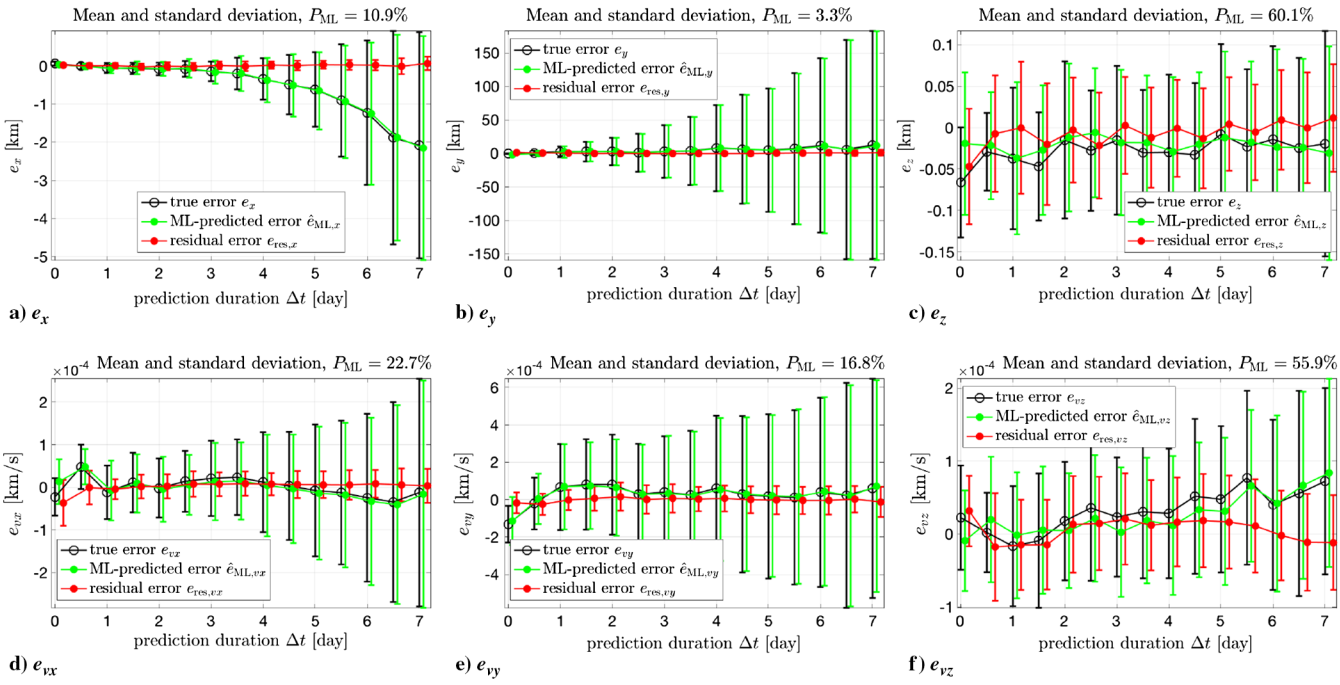


Fig. 8 Performance of ANNs for different components on the testing data (week 5).

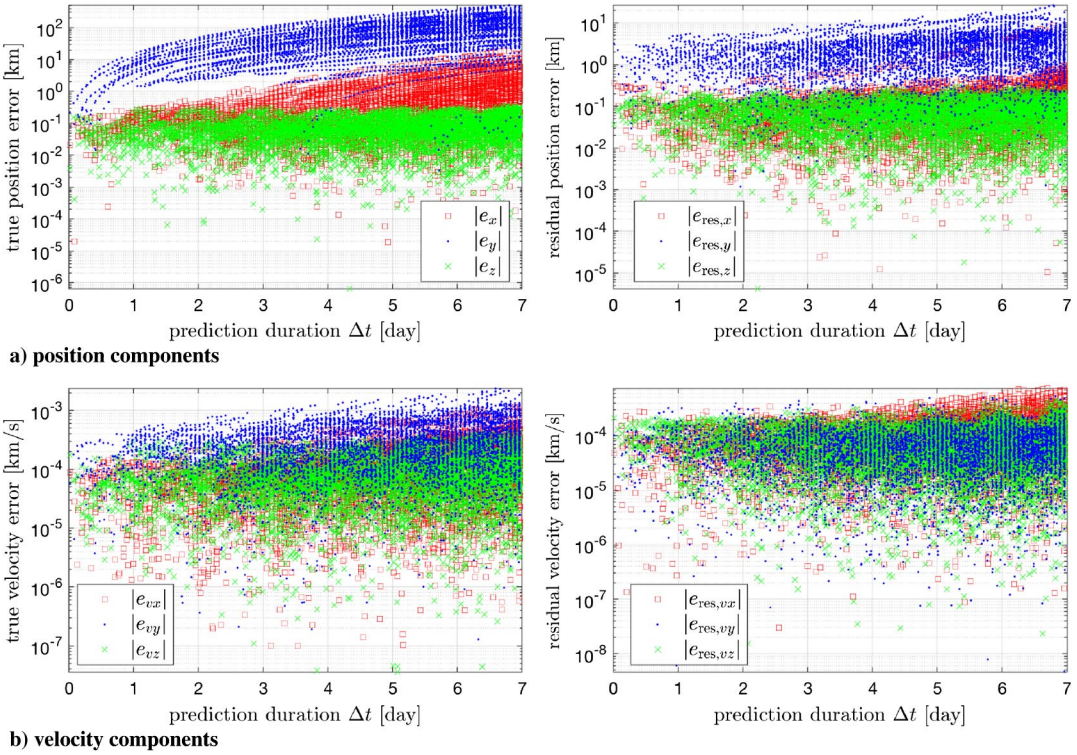


Fig. 9 Comparisons between the true (left) and residual (right) orbit prediction error using trained ANNs.

$e_y > e_x > e_z$, and after the modification using ANNs, $e_{\text{res},x}$ and $e_{\text{res},y}$ have been significantly reduced, and $e_{\text{res},x}$ and $e_{\text{res},z}$ have come to the same magnitude. It is also interesting to note that the residual velocity errors $e_{\text{res},vx}$, $e_{\text{res},vy}$, and $e_{\text{res},vz}$ have been reduced to almost the same magnitude.

C. Effect of Random Initialization

There is randomness in most of the training algorithms for ANN. Before the training, all the weights ω_{jk}^{l} of the network have to be initialized. The training result depends on the initialization of weights and is usually different for each initialization, which is due to many local optimal solutions of ANN. So a practical guideline is that the performance of ANN should not be evaluated by a single experiment. In fact, it is a common practice to improve the performance of an existing trained ANN by trying different initializations, until the performance is acceptable.

In our experiments, the Nguyen-Widrow initialization algorithm (`initnw` in MATLAB) is used to initialize the weights at each layer. By setting various random seeds for the global random stream of MATLAB, we can achieve different initializations. For all the trainings of ANNs in this paper, 10 random seeds are used and then fixed to explore the effect of randomness. The chosen seeds are [42, 375, 951, 732, 599, 157, 156, 59, 867, 602] (for reproduction purpose). All the trainings of the ANN in the following study, either with different network structures or for different components, will be carried out 10 times using these 10 seeds (set by `rng` in MATLAB).

For the ANNs in the previous subsection, which have 1 hidden layer of 20 neurons, the results of 10 tests are summarized in Fig. 10. The dashed line with circles represents the training results, and the solid line with asterisks represents the testing results. The results reveal that the trained ANN have captured the underlying relationship between the learning and target variables, regardless of a particular

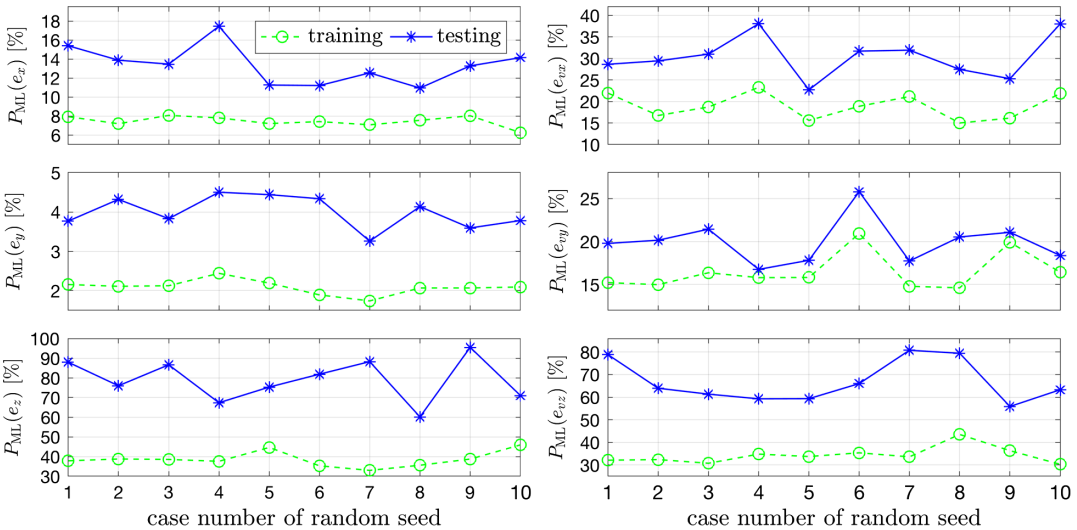


Fig. 10 Performances of ANNs initialized with different random seeds, trained with the same training data in weeks 1–4 and tested with the same testing data in week 5.

initialization. The training results are more consistent, whereas the testing results are relatively more sensitive to the initialization. There are some cases where the performance is obviously worse than others, for example, case 4 of e_x and case 6 of e_{vy}. From a practical point of view, a proper initialization of the ANN can always be found through a trial-and-error process, and a random initialization will usually not lead to too bad performance.

D. Effect of Network Structure

In this subsection, we investigate the effect of network structure of ANN on the generalization capability, by varying both the number of neurons at each layer and the number of layers. We note that having more neurons and layers will introduce more weights to be adjusted in the network, therefore more flexibility. This provides the ANN model more powerful approximation capability. However, as we have mentioned in Sec. II, more flexibility also means that it is more likely to overfit the training data, which will lead to a bad generalization performance.

First, we varied the number of neurons at all the layers from 10 to 30 with a step of 5. The results of the trained ANNs are summarized in Fig. 11. The horizontal axis represents the number of neurons at each hidden layer. The circles represent the training results for each random seed, and the asterisks represent the testing results. The dashed and solid lines represent medians of each set of results corresponding to the 10 random seeds. Note that the minimum P_{ML} among all the random initialization represents the best performance of the ANN, and the maximum P_{ML} reflects the capability of the ANN in the worst situation. Because we are more interested in the expectation of the performance, we have chosen the median of the metric P_{ML} to represent the performance of the ANN with a certain structure.

In Fig. 11, a general trend is that the performance on training data (dashed lines) of ANN is becoming better when more neurons are used. However, this is not exactly the same for the performance on the testing data (solid lines). For e_x, e_z, and e_{vy}, the metrics P_{ML} on testing data tend to decrease with respect to neuron numbers. But for other components, the performance only changes slightly. This in fact is good, revealing that ANN is actually learning the underlying relationship, rather than just fitting particular data patterns. If on the contrary, better performance on the training data corresponds to worse generalization results, it will imply the overfitting. On the other hand, this phenomenon also shows that the early stop criterion works well.

Second, the number of hidden layers is varied from 1 to 3 to explore relatively deeper networks, while the number of neurons is still varied from 10 to 30. We note that incorporating even more hidden layers requires longer computation time and also more training data to avoid overfitting, but the performance improvement can be limited. The results are summarized in Fig. 12. For clarity, only the median metrics P_{ML} are demonstrated. The annotations are similar to previous figures except that different markers are used to represent the results of ANNs with different hidden layers, where circles, asterisks, and triangles correspond to one, two, and three hidden layers, respectively.

In Fig. 12, the metrics P_{ML} on training data (dashed lines) are almost monotonously decreasing as both layers and neurons increases. So, generally, more hidden layers and more neurons will lead to better learning capability. Comparing the dashed curves horizontally, it is shown that an ANN with fewer layers but more neurons can have learning capability equivalent to an ANN with more layers but fewer neurons. For example, for e_x, the case with 1 hidden layer with 20 neurons and the case with 2 hidden layers with 10 neurons both have P_{ML} of about 8%. Such phenomena are expected from the theory about ANN but it is still helpful and interesting to be discovered for the orbit prediction problem for the first time.

In Fig. 12, the generalization performance on testing data (solid lines) shows different trends. For e_z and e_{vz}, the metrics P_{ML} are decreasing with respect to the number of both hidden layers and neurons. But for e_x, e_y, e_{vx}, and e_{vy}, the ANN with one hidden layer either has the best performance in most cases or appears most stable with respect to the neuron number. We note again that a good learning performance does not always guarantee a good generalization capability, especially for orbit predictions at future epochs. And the results indicate that introducing more layers does not always lead to a better performance for all the components of the orbit prediction error. Therefore, in order to make the best use of historical data, it is helpful to design different structures for different error components. It remains for future studies to develop a systematic procedure to design optimal structures of ANN.

IV. Generalization onto Different RSOs

In this section, the type III generalization capability of ANN from the training RSO onto other RSOs is investigated. First, a coverage analysis procedure to check learning variables for the ML approach is presented. Then, we demonstrate and analyze the performance of the trained ANNs on RSOs with different altitudes or right ascension of ascending nodes (RAANs).

We have varied the semi-major axis deviation Δa from -100 to 100 km with a step size of 10 km, and the RAAN deviation $\Delta\Omega$ from

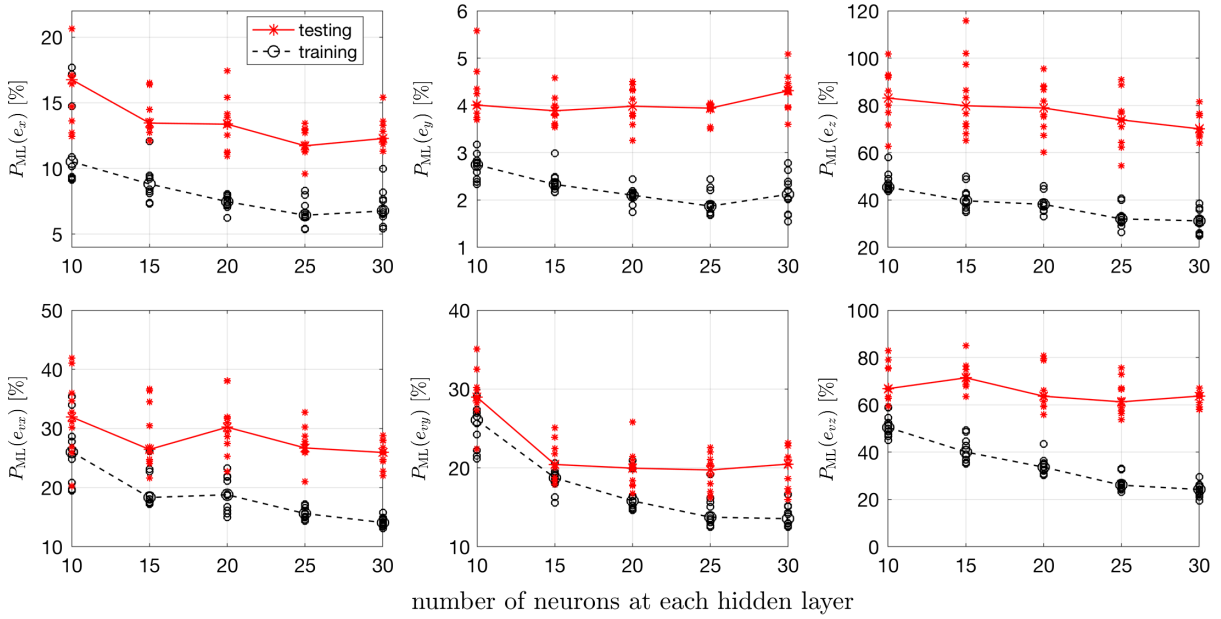


Fig. 11 Performance of trained ANNs with different numbers of neuron on the same testing data (week 5).

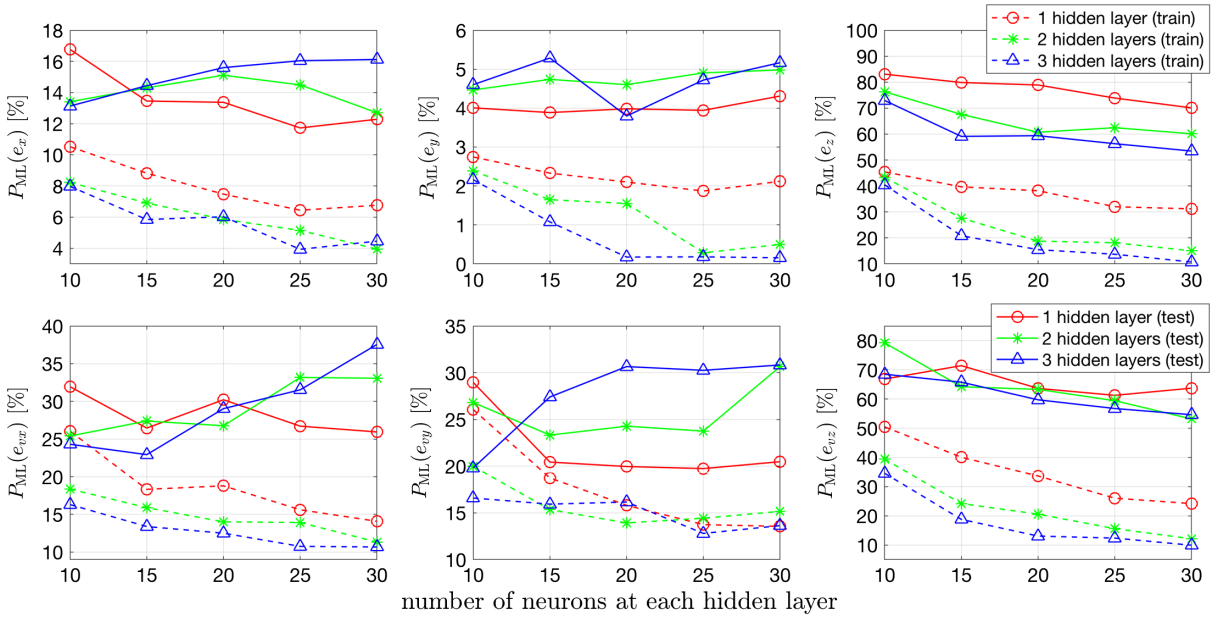
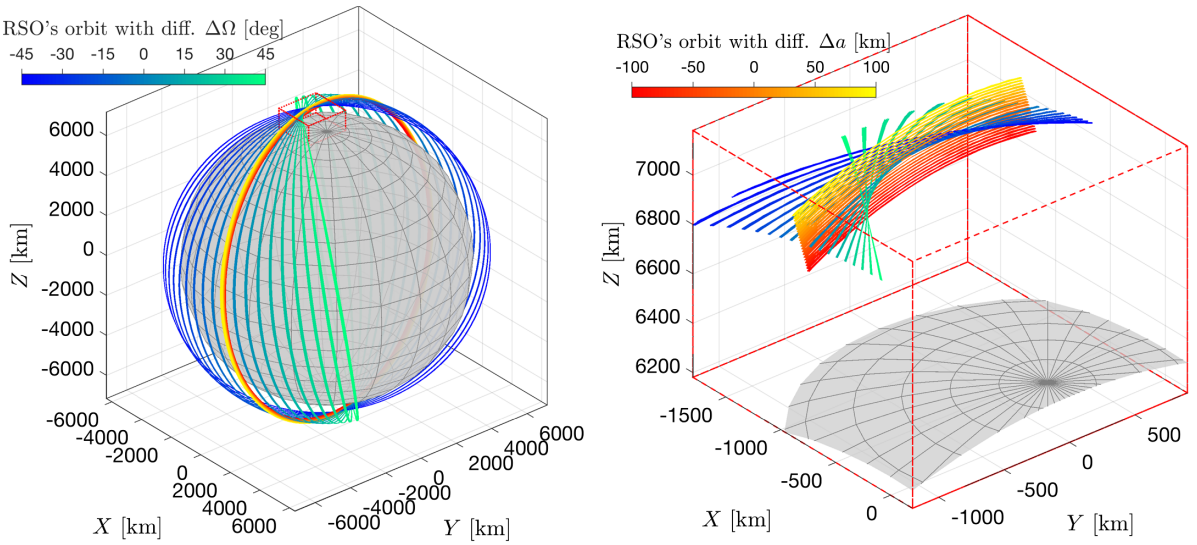


Fig. 12 Performance metrics P_{ML} of different components on the training data, using different network structures.



a) All RSOs **b) Zoom-in at the framed region of (a)**

Fig. 13 Orbits of RSOs with Δa and $\Delta\Omega$ based on ENVISAT for 1 day in the ECI frame.

−45° to 45° with a step size of 5°. Figure 13 shows all the orbits for 1 day in the ECI frame. Figure 13a shows the orbits with different $\Delta\Omega$, and Fig. 13b shows the orbits with different Δa by a zoom-in plot at the framed region of Fig. 13a. We note that the orbit of the training RSO locates at the center of all orbits, and the new orbits are different enough from ENVISAT's orbit for the testing purpose.

A. Coverage Analysis of Learning Variables

As has been discovered in our earlier study, checking the range of the learning variables is critical. This is because the range of the variables on the generalized RSOs can be significantly different from those of the training RSO. Here, a brief review is included.

Take a new RSO, which has an altitude deviations of 10 km from the training RSO, ENVISAT, as an example. The coverage analysis of the learning variables is demonstrated in Fig. 14. Each vertical bar represents the range of a learning variable in the training data, normalized to the range of [0, 1] by its minimum and maximum values. The dots represent the variables in the testing data that have been well-covered by the range of the training data. And the crosses represent the variables, for which at least 10% of data are out of the range [−0.1, 1.1]. This criterion has been chosen through a trial-and-error process. The badly covered variables are also marked with a cross symbol at the bottom of the axis. The bars of the variable 4 and 20, which have no dots or crosses, indicate that the corresponding variables are far out of the ranges.

Therefore, in this example for ANNs trained by the ENVISAT and then generalized to the new RSO, variables labeled as {2, 4, 18, 20, 22, 27} should be excluded from the learning variables (see Fig. 14). This procedure will be applied for all the experiments of the Type III generalization.

B. Varying Semi-Major Axis

In this section, we study the case with different semi-major axes, or the altitude, which is one of the important parameters for RSOs in SSO. As demonstrated in Fig. 13b, the altitude has been varied −100 to 100 km with a step size of 10 km.

We simulate each new RSO for 5 weeks. The ANNs will be trained by the data of ENVISAT in weeks 1–4, and then the data of the new RSO in week 5 will be used to test the Type III generalization capability of the trained ANNs. Considering that the altitude of the

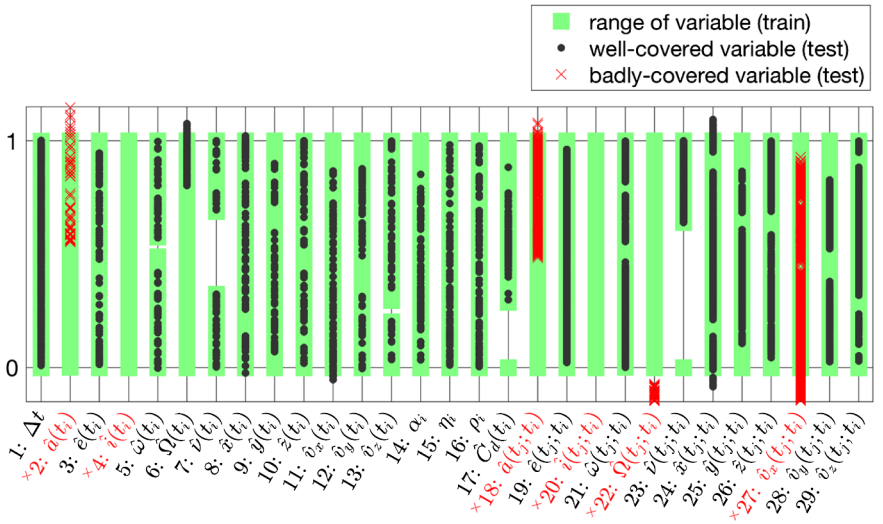


Fig. 14 Coverage analysis example of learning variables in training and testing data.

RSO ENVISAT is just about 800 km, as shown in Table 3, a maximum variation of 100 km is relatively large.

The variables that are badly covered for at least one of the new RSOs with varied semi-major axis will be removed from the redesigned learning variables. In this way, the redesigned learning variables are the same for all RSOs, and so we can compare the performance of ANNs on different RSOs. As a result, the variables labeled as {2, 4, 18, 20, 22, 24, 27} are excluded for the redesigned ANNs. Based on the results in the previous section in Fig. 12, for e_x and e_{vx}, the network structure is chosen to be 1 hidden layer with 25 neurons, which gives the best performance on the testing data; for e_y and e_{vy}, the chosen structure is 1 hidden layer with 20 neurons, because its performance is similar compared with those with more neurons and is better or more stable with respect to the neuron number than those with more layers; for e_z and e_{vz}, the chosen structure is 2 hidden layers with 20 neurons as a compromise between performance and computational burden.

The performance metrics for all the cases are summarized in Fig. 15. The horizontal axis is the changes of altitude, or semi-major axis, Δa. Similar to previous annotations, the circles (or asterisks) represent performance metrics P_{ML} on the training (or testing) data in experiments with 10 different random seeds; the solid lines represent the medians of all performance metrics on testing data; and the dotted lines represent the critical metric $P_{\mathrm{ML}} = 100\%$. As expected, the metrics P_{ML} on new RSOs increases as $|\Delta a|$ increases, which means that fewer orbit prediction errors can be removed when the new RSO has larger semi-major axis difference. When P_{ML} is larger than 100%, the average residual error after ML modification has not been reduced compared with the original error. For such cases, we will claim the generalization capability as invalid. For other cases with $P_{\mathrm{ML}} < 100\%$, we shall claim the generalization capability as valid. However, we remark that because the performance metric P_{ML} is defined as the average performance of all data points, and due to the probabilistic nature of the ML approach, orbit prediction errors of some data points can be increased even when P_{ML} is smaller than 100%. For e_x and e_y the generalization capability is valid to all negative Δa and to positive Δa until about 30 and 70, respectively. The scattering asterisks also reveal that this generalization capability is, to some extent, robust with respect to the initialization of ANN. The results of e_{vx} and e_{vy} are less exciting but still show good or at least feasible generalization capabilities for Δa from about -60 to 20 km and -100 to 60 km, respectively.

The results of e_z and e_{vz} in Fig. 15 show that the model cannot be used to correct errors for these two components. We investigate these cases further. First, taking the case with $\Delta a = 10$ km and the random seed 42 as example, the detailed results are demonstrated in Fig. 16.

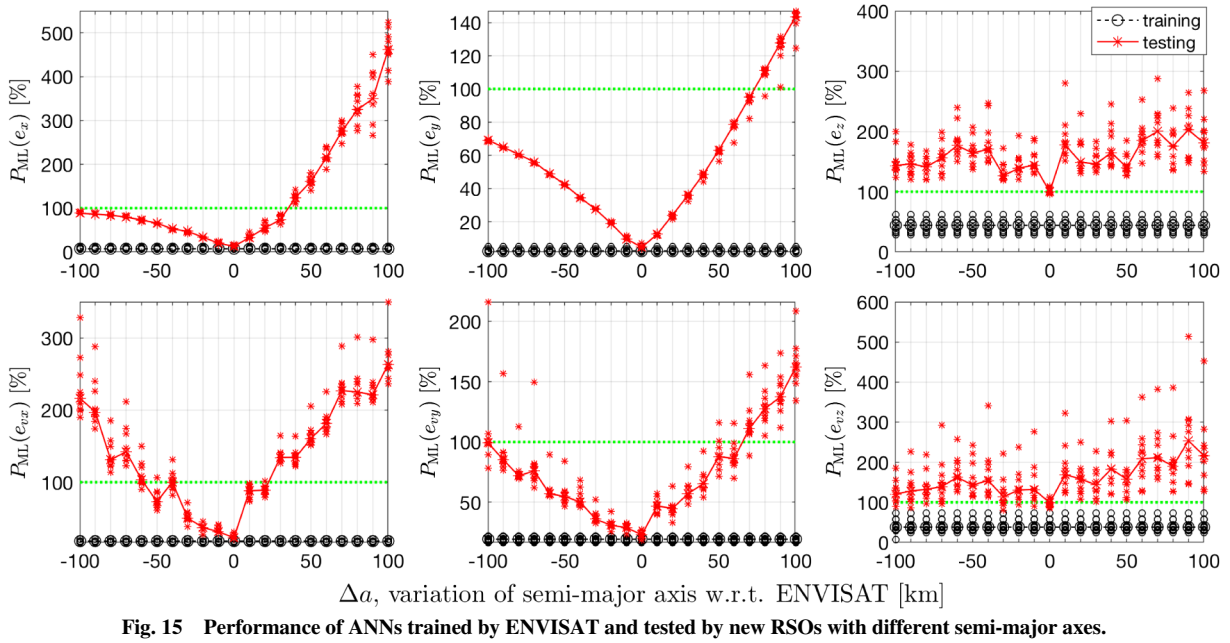


Fig. 15 Performance of ANNs trained by ENVISAT and tested by new RSOs with different semi-major axes.

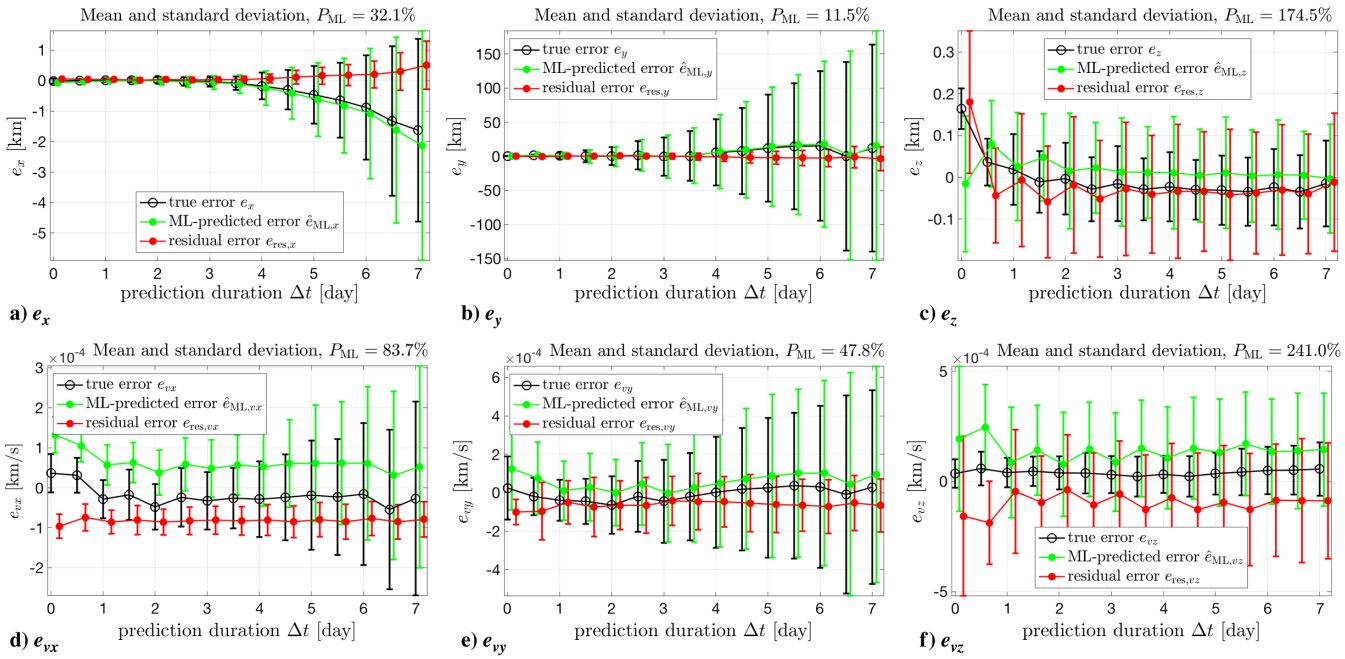


Fig. 16 Example of Type III generalization capability to the new RSO with $\Delta a = 10$ km.

We note that, for all the components except e_z and e_{vz}, their standard deviations tend to increase with the prediction duration Δt, and a main effect of ANNs is to reduce the standard deviations. However, the behavior of the standard deviations of e_z and e_{vz} is different, almost constant, or only slightly increase with Δt. A physical conjecture is provided here. The motion along z axis, the cross-section axis, reflects the procession of the orbit, which is mainly caused by the nonspherical shape of the Earth. Because the procession motion has already been accurately modeled in the assumed model, the orbit prediction error of e_z and e_{vz} will mainly contain intrinsic random errors such as the measurement noise, which cannot be eliminated by the proposed ML approach.

Using the same case as in Fig. 16, an experiment that considers only the J_2 perturbation in the assumed model was carried out to verify the above conjecture. The results are summarized in Fig. 17, where the ANNs are trained by ENVISAT and the new RSO has a Δa of 10 km. Now, as the orbit prediction errors are much larger due to the less accurate assumed models, the performance of ANNs for all components is significant, especially for e_z and e_{vz}. Furthermore, the performance metrics when using only J_2 perturbation in the assumed model is demonstrated in Fig. 18, which reveals that the trained ANNs have good Type III generalization capabilities to nearby RSOs with different semi-major axes for all the components.

C. Varying Right Ascension of Ascending Node

Another important parameter of RSOs in SSO we have examined is the right ascension of ascending node (RAAN). The variation of RAAN, $\Delta\Omega$, is varied from $-45°$ to $45°$ with a step size of $5°$, as shown in Fig. 13a. The orbits are propagated for 5 weeks and the data in week 5 are used as the testing data. Because they are all SSOs, their relative geometries in Fig. 13a are constant during the propagation.

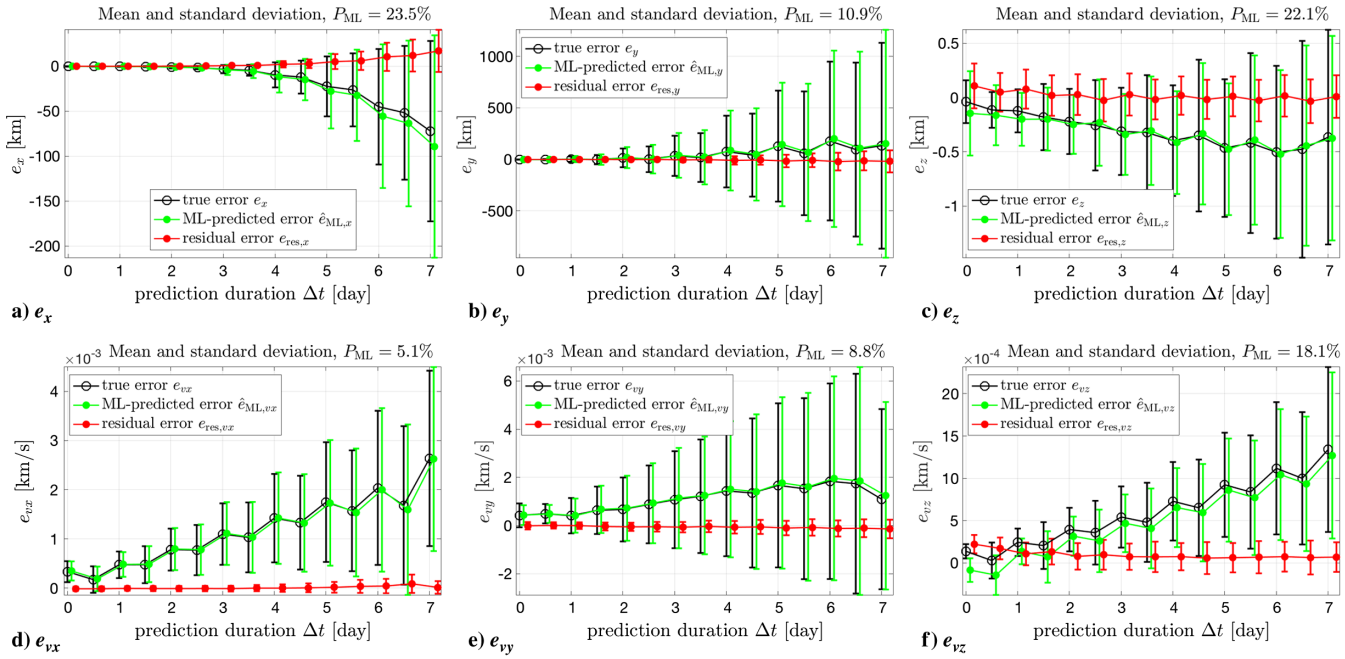


Fig. 17 Example of Type III generalization capability to the new RSO with $\Delta a = 10$ km, using J_2 perturbation in assumed model.

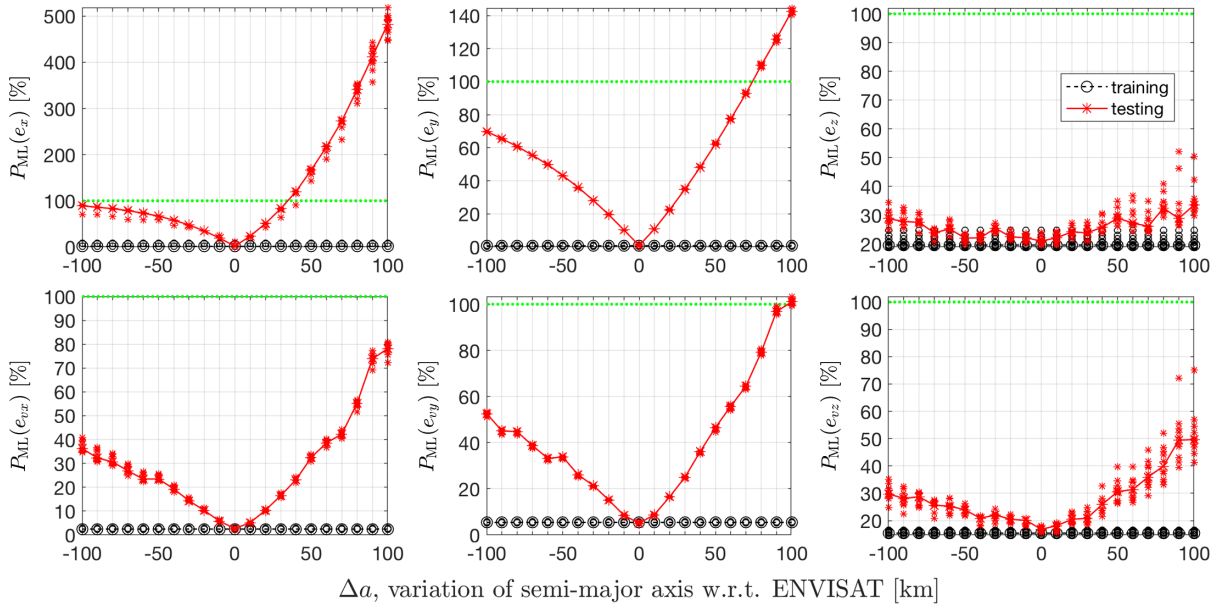


Fig. 18 Performance of ANNs trained by ENVISAT and tested by new RSOs with different semi-major axes, using J_2 perturbation in assumed model.

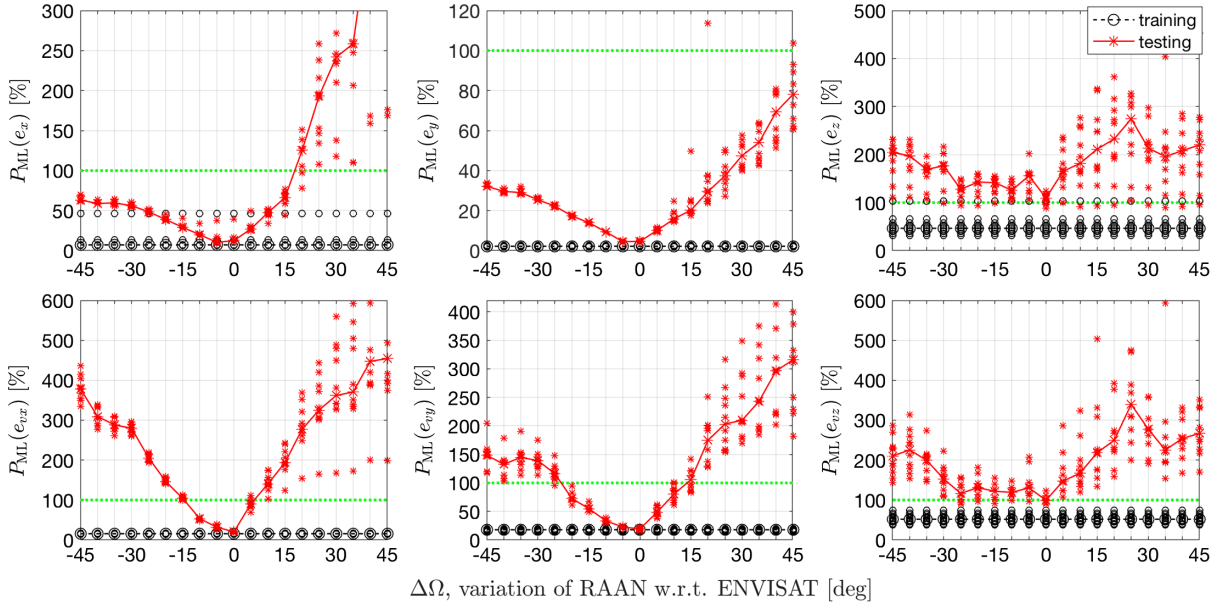


Fig. 19 Performance of ANNs trained by ENVISAT and tested by RSOs with different RAANs.

As before, for all the new RSOs with varied RAAN, all badly covered variables have been excluded in the redesigned ANNs, which are the variables labeled as $\{4, 6, 8, 11, 20, 22, 24, 27\}$. And the network structures for each component are the same as those in the previous subsection. The results are summarized in Fig. 19. Similar results are observed to show that ANNs have good generalization capabilities around the original RSO for all the components except e_z and e_{vz}. The reason is similar to what we have discussed in the previous subsection.

V. Conclusions

In this paper, the ML approach using ANN to improve the orbit prediction accuracy has been examined and demonstrated with good performance. The ANN models for all the components of the orbit prediction error are trained by the historical data of an RSO in SSO. Two types of generalization capabilities of the trained ANNs have been studied. First, the generalization capability to future epochs is found to be good. And the effect of the random initialization and the network structure has been systematically studied. The initialization is critical to guarantee a good performance, and the best network structures are different for individual error components. Second, the generalization capability to other RSOs is studied, where the ANNs are trained by the same RSO but tested on new RSOs with different semi-major axes or right ascension of ascending nodes. The results reveal that the ANNs could be generalized to a relatively wide range of nearby RSOs that have not been used for training.

Together with our previous studies using the support vector machine algorithm, it is further demonstrated that the ML approach is a feasible and promising method in improving orbit prediction accuracy. Besides of the good performance, the results also raise questions for future research. For example, is there a way to find an optimal initialization algorithm or network structure? And how shall we interpret the relationship between trained ANNs (especially with multiple hidden layers) and the physical laws?

Acknowledgments

The authors acknowledge the research support from the Air Force Office of Scientific Research (AFOSR) FA9550-16-1-0184 and the Office of Naval Research (ONR) N00014-16-1-2729. Large amount of simulations have been carried out on the SOE (School of Engineering) HPC cluster at Rutgers University.

References

[1] "Space Debris by the Numbers," Oct. 2017, http://www.esa.int/Our_Activities/Operations/Space_Debris/Space_debris_by_the_numbers/ [retrieved 19 Oct. 2017].

[2] Wang, Y., and Gurfil, P., "Dynamical Modeling and Lifetime Analysis of Geostationary Transfer Orbits," *Acta Astronautica*, Vol. 128, Nov.–Dec. 2016, pp. 262–276. doi:10.1016/j.actaastro.2016.06.050

[3] Kelso, T., "Iridium 33/Cosmos 2251 Collision," July 2009, http://celestrak.com/events/collision.asp.

[4] Peng, H., and Bai, X., "Improving Orbit Prediction Accuracy Through Supervised Machine Learning," 2018, pp. 1–30, http://arxiv.org/abs/1801.04856.

[5] Peng, H., and Bai, X., "Exploring Capability of Support Vector Machine for Improving Satellite Orbit Prediction Accuracy," *Journal of Aerospace Information Systems*, 2017. doi:10.2514/1.I010616

[6] Peng, H., and Bai, X., "Recovering Area-to-Mass Ratio of Resident Space Objects Through Data Mining," *Acta Astronautica*, Vol. 142, Jan. 2018, pp. 75–86. doi:10.1016/j.actaastro.2017.09.030

[7] Peng, H., and Bai, X., "Limits of Machine Learning Approach on Improving Orbit Prediction Accuracy," *Advanced Maui Optical and Space Surveillance Technologies (AMOS) Conference*, Maui Economic Development Board, Inc., Wailea Marriott, Sept. 2017, https://amostech.com/TechnicalPapers/2017/Astrodynamics/Bai.pdf.

[8] Vapnik, V. N., *The Nature of Statistical Learning Theory*, Springer, New York, 2000, Chap. 6. doi:10.1007/978-1-4757-3264-1

[9] Abu-Mostafa, Y. S., Magdon-Ismail, M., and Lin, H.-T., *Learning from Data*, AMLBook, 2012, Chap. 8, AMLbook.com.

[10] Du, K.-L., and Swamy, M. N. S., *Neural Networks and Statistical Learning*, Springer, London, 2014, Chap. 16. doi:10.1007/978-1-4471-5571-3

[11] Huang, A., Guez, A., Maddison, C. J., Silver, D., Hassabis, D., Grewe, D., van den Driessche, G., Sutskever, I., Antonoglou, I., Nham, J., Schrittwieser, J., Kavukcuoglu, K., Sifre, L., Leach, M., Lanctot, M., Kalchbrenner, N., Dieleman, S., Graepel, T., Lillicrap, T., and Panneershelvam, V., "Mastering the Game of Go with Deep Neural Networks and Tree Search," *Nature*, Vol. 529, No. 7587, 2016, pp. 484–489. doi:10.1038/nature16961

[12] Abu-Mostafa, Y., Magdon-Ismail, M., and Lin, H., "E-Chapter 7: Neural Networks," *Learning from Data*, 2015, http://amlbook.com/.

[13] Nielsen, M., "Neural Networks and Deep Learning," 2015, http://neuralnetworksanddeeplearning.com.

[14] Huang, G.-B., Chen, L., and Siew, C.-K., "Universal Approximation Using Incremental Constructive Feedforward Networks with Random Hidden Nodes," *IEEE Transactions on Neural Networks*, Vol. 17, No. 4, 2006, pp. 879–892. doi:10.1109/TNN.2006.875977

[15] Williams, K., "Prediction of Solar Activity with a Neural Network and Its Effect on Orbit Prediction," *Johns Hopkins APL Technical Digest*, Vol. 12, No. 4, 1991, pp. 310–317, http://www.jhuapl.edu/techdigest/views/pdfs/V12_N4_1991/V12_N4_1991_Williams.pdf.

[16] Macpherson, K. P., Conway, A. J., and Brown, J. C., "Prediction of Solar and Geomagnetic Activity Data Using Neural Networks," *Journal of Geophysical Research: Space Physics*, Vol. 100, No. A11, 1995, pp. 21735–21744. doi:10.1029/95JA02283

[17] Lu, Y., Sundararajan, N., and Saratchandran, P., "A Sequential Learning Scheme for Function Approximation Using Minimal Radial Basis Function Neural Networks," *Neural Computation*, Vol. 9, No. 2, 1997, pp. 461–478. doi:10.1162/neco.1997.9.2.461

[18] Nigel, C. B. H., Sundararajan, N., and Saratchandran, P., "Nonlinear Aircraft Control Using a Minimal Radial Basis Function Neural Network," *Proceedings of the 1998 American Control Conference. ACC (IEEE Cat. No. 98CH36207)*, Vol. 4, IEEE Publ., Piscataway, NJ, 1998, pp. 2589–2590. doi:10.1109/ACC.1998.703102.

[19] Pashilkar, A. A., Sundararajan, N., and Saratchandran, P., "A Fault-Tolerant Neural Aided Controller for Aircraft Auto-Landing," *Aerospace Science and Technology*, Vol. 10, No. 1, 2006, pp. 49–61. doi:10.1016/j.ast.2005.05.002.

[20] Jwo, D.-J., Chang, C.-S., and Lin, C.-H., "Neural Network Aided Adaptive Kalman Filtering for GPS Applications," *2004 IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, IEEE Publ., Piscataway, NJ, 2004, pp. 3686–3691, http://ieeexplore.ieee.org/abstract/document/1400916/.

[21] Sánchez-Sánchez, C., and Izzo, D., "Real-Time Optimal Control via Deep Neural Networks: Study on Landing Problems," 2016, https://arxiv.org/abs/1610.08668.

[22] Sánchez-Sánchez, C., Izzo, D., and Hennes, D., "Learning the Optimal State-Feedback Using Deep Networks," *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE Publ., Piscataway, NJ, 2016, pp. 1–8. doi:10.1109/SSCI.2016.7850105

[23] Mereta, A., Izzo, D., and Wittig, A., "Machine Learning of Optimal Low-Thrust Transfers Between Near-Earth Objects," *Hybrid Artificial Intelligent Systems*, Springer, Cham, 2017, pp. 543–553. doi:10.1007/978-3-319-59650-1_46

[24] Förste, C., Bruinsma, S. L., Shako, R., Abrikosov, O., Flechtner, F., Marty, J.-C., Lemoine, J.-M., Dahle, C., Neumeyer, H., Barthelmes, F., Biancale, R., Balmino, G., and König, R., "A New Release of EIGEN-6: The Latest Combined Global Gravity Field Model Including LAGEOS, GRACE and GOCE Data from the Collaboration of GFZ Potsdam and GRGS Toulouse," *American Geophysical Union, General Assembly 2012*, Vol. 14, European Geosciences Union, Vienna, Austria, 2012, p. 2821, http://adsabs.harvard.edu/abs/2012EGUGA..14.2821F.

[25] Folkner, W. M., Williams, J. G., Boggs, D. H., Park, R. S., and Kuchynka, P., "The Planetary and Lunar Ephemerides DE430 and DE431," Rept. 42-196, JPL, Pasadena, CA, 2014.

[26] Hill, K., Sabol, C., and Alfriend, K. T., "Comparison of Covariance Based Track Association Approaches Using Simulated Radar Data," *The Journal of the Astronautical Sciences*, Vol. 59, Nos. 1–2, 2012, pp. 281–300. doi:10.1007/s40295-013-0018-1

[27] Crassidis, J. L., and Junkins, J. L., *Optimal Estimation of Dynamic Systems*, 2nd ed., Chapman and Hall/CRC Applied Mathematics & Nonlinear Science, CRC Press, Boca Raton, FL, 2011, http://ebooks.cambridge.org/ref/id/CBO9781107415324A009.

[28] Maisonobe, L., Pommier, V., and Parraud, P., "Orekit: An Open Source Library for Operational Flight Dynamics Applications," *4th International Conference on Astrodynamics Tools and Techniques*, 2010, https://www.researchgate.net/publication/310250345_OREKIT_AN_OPEN_SOURCE_LIBRARY_FOR_OPERATIONAL_FLIGHT_DYNAMICS_APPLICATIONS.

[29] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, McGraw–Hill, Inc., New York, 1997, pp. 42–44, http://www.springer.com/us/book/9780387718316.

M. Xin
Associate Editor