On the Use of Machine Learning Algorithms to Improve GNSS Products

On the Use of Machine Learning Algorithms to Improve GNSS Products

Andrea Nardin
Politecnico di Torino, DET
Turin, Italy
andrea.nardin@polito.it

Fabio Dovis
Politecnico di Torino, DET
Turin, Italy
fabio.dovis@polito.it

Diego Valsesia
Politecnico di Torino, DET
Turin, Italy
diego.valsesia@polito.it

Enrico Magli
Politecnico di Torino, DET
Turin, Italy
enrico.magli@polito.it

Chiara Leuzzi
ALTEC
Turin, Italy
chiara.leuzzi@altecspace.it

Rosario Messineo
ALTEC
Turin, Italy
rosario.messineo@altecspace.it

Hugo Sobreira
European Space Agency, ESTEC
Noordwijk, The Netherlands
hugo.sobreira@esa.int

Richard Swinden
European Space Agency, ESTEC
Noordwijk, The Netherlands
richard.swinden@esa.int

***Abstract*—This paper presents the most relevant results on the investigation of possible uses of machine learning based techniques for the processing of data in the field of Global Navigation Satellite Systems. The work was performed under funding of the European Space Agency and addressed different kind of data present in the entire chain of the positioning process, as well as different kind of machine learning approaches. This paper presents the most promising results obtained for the prediction of ionospheric maps for the correction of the related error on the pseudorange measurement and for the forecast of fast corrections normally present in the EGNOS messages, when the latter might be missing. Results show how, based on the historical data and the time correlation of the values, machine learning methods outperformed simple regression algorithms, improving the positioning performance at GNSS user level. The work results also confirmed the validity of this approach for the automatic detection of outliers due to ionospheric scintillation phenomena.**

***Index Terms*—Machine-learning, GNSS, Ionosphere, Positioning**

I. Introduction

It is well known that machine learning (ML) methods are a powerful tool demonstrating their value when dealing with large amount of data. In particular, they show their effectiveness for the prediction of future evolution of the data series or to identify, in an automated way, "patterns" in the data or outliers. The *GNSS ML Demonstrator (GMLD)* aimed at investigating possible applications in the Global Navigation Satellite System (GNSS) domain that could benefit from ML capabilities, to improve some elements of the entire GNSS process to provide better positioning, navigation, and timing (PNT) services. Indeed, GNSS-based PNT has become fundamental for a wide range of applications, from critical infrastructures [1], [2] to intelligent transportation systems [3]–[5], to name a few, and, nonetheless, an effective resource for many other purposes [6]–[8]. Such ubiquity is therefore fostering an increased need to access correct GNSS data.

This work has been developed in the framework of the activity "Machine-Learning to model GNSS systems" funded by the European Space Agency (NAVISP-EL1-035.02).

ML-based techniques have been investigated for GNSSs and also proved their effectiveness on the receiver side [9], [10]. Nevertheless, this study aims at investigating and demonstrating the use of ML in the broader area of the GNSS domain. In particular, one of the main goals of the investigation is to improve the GNSS data that affect the quality of the GNSS measurements used for the position estimation. Such data might be either included in the navigation message or coming from external sources, and be needed for the construction of the pseudorange or to correct the latter from predictable error contributions. When such pieces of information are either missing—due to communication or demodulation issues—or outdated, the impact on the estimation of the position, velocity and time (PVT) could be disruptive. In the last years, the improvement of GNSS data and measurements has been pursued even by proposing space segment modernization [11], [12] or alternative PNT systems [13], [14], but failing to convey such information to users would still be a major waste of system's resources.

The investigated solutions aim at mitigating the impact of these issues by predicting some parameters, when missing, or implementing proper countermeasures at receiver level when the presence of errors in the data or in the measurements is detected/predicted (e.g. excluding some pseudoranges from the PVT computation or adapting the receiver tracking loops parameters). To this purpose we investigated a set of case-studies of applications that implement ML models able to predict missing information or the presence of possible error sources. In particular, four applications have been identified in order to cover a variety of diverse error sources, as well as different data structures.

The first application targets the prediction of the orbital parameters at receiver level, when the broadcasted ephemeris are not available, comparing the improvement of orbit prediction with respect to basic regression or the use of old (but valid) values. The second area investigated is the prediction of daily maps of the ionosphere (TEC maps), while the third is the estimation of the satellite-based augmentation system (SBAS)

correction parameters in the missed messages (fast and slow corrections in particular). In order to investigate the potential of automatic ML algorithms on large data series, the fourth area of investigation is the detection of signal distortions due to the presence of ionospheric scintillations and multipath [15].

In this work we focus on the most promising solutions that were found for the prediction capabilities, reporting the results of the second and third application. As for the detection of outliers in the GNSS signals, the investigation confirmed the ability of this method as already presented in the recent scientific literature as for example in [16]–[21], even if more efficient solutions were tested.

To deeply analyze the effectiveness of using the ML algorithm for each specific GNSS application, it is important to assess how the effect, for example, of a wrong prediction of a parameter could affect the GNSS application and if, in case of deviation, it still falls within an acceptable confidence interval. The assessment is based on the generation of scenarios via simulation or real data that are processed by the previously trained ML algorithms, or on the comparison of the results with the current state-of-the-art methods. Such outputs are compared with the expected performance to derive the performance metrics values.

The paper is organized as follows: Section II will describe the objectives and the methodology used in developing the work. Section III and Section IV will describe the use of ML algorithms to GNSS applications discussing the chosen algorithms and the results obtained, regarding the prediction of TEC maps and the prediction of fast corrections in SBAS messages, respectively. Eventually, Section V will summarize the conclusions and provide some insights on the future work.

II. Problem Description and Approach

The general object of this work has been to investigate the use of ML techniques in the GNSS framework, as a mean to alleviate the performance degradation effects of different error sources that appear along the processing chain, from the satellite transmission to the final PVT solution provided to the user. The use of smart algorithms can then act at different levels of the processing chain. The ML algorithm could work at signal level (i.e. on the digital stream of samples at the output of the front-end) to automatically assess the quality of the received signal, but also act as a processor of the post-correlation measurements (observables) to characterize the operating environment, providing, meanwhile, prediction of the future trends of the time-varying parameters. Furthermore, ML algorithms could act on aiding data that are provided by external sources to improve the PVT solution, either mitigating error contribution through estimated corrections, or easing the calculation of the solution (e.g. reduced TTFF, or increased sensitivity). There is then a large variety of cases that can be taken into account. Within this work, in order to address different data structures, rates and complexity of the data stream, applications to receiver observables, external data sources, and raw signal samples processed by a software GNSS receiver, were considered.

Figure 1 depicts a high level scheme of the GMLD implementation showing the data sources, the main software components, the data flows and the two different application pipeline configurations (i.e. development and production) needed to build the ML models and to run them inside the application, respectively. The GMLD consists of the software modules implementing the four selected applications plus several other software modules that are used across applications and, therefore, can be shared among them. The shared software modules provide capabilities to read and write GNSS data and manage ML model definition, training, validation and run. The input of the GMLD are files of GNSS data products stored in several repositories that are accessed to retrieve historical datasets. They are stored in raw format and then pre-processed to feed the ML application pipeline. After data ingestion and transformation, data are consumed by the ML applications, which consist of the following three stages. The *data preprocessing* block performs filtering, sampling, and cleaning operations to prepare the data for the specific application. It has the capability to read the GNSS data product formats involved in the application and store preprocessed data using intermediate formats that are suitable to feed the subsequent stage. The *feature extraction* is dedicated to the extraction of information which is provided as input to the ML based models. Finally, the *machine learning module* implements training and validation, in the development pipeline, and prediction, in the production pipeline.

The software modules defined so far were implemented and validated to fulfill the application objectives. They were built using existing algorithm implementations available in open source ML frameworks and libraries that are properly instantiated, configured and tuned to reach the expected application performance. ML applications are orchestrated through a flow engine that allows to define, coordinate, and monitor the production application pipeline and the other data management flows. To record all the experiments of the training and validation activities and to catalog the generated models, the GMLD integrates a *ML registry*. The *post-processing module* consists of a *data product generator* and other tools needed to support the GNSS performance assessment. It is in charge of reading the ML model output and generating the data products according to the GNSS data products format (e.g. RINEX). This module is useful to explore and analyse the application output with external tools that are common in the GNSS domain. The GMLD has a front-end based on interactive computing technologies (i.e. Jupyter Notebook) that is used for data exploration of all GMLD data, ML models development from preprocessing to model validation, and production application execution.

In addition to the standalone software, the applications have been deployed into the GNSS Science Support Centre (GSSC) platform [22]. For this purpose, each application has been refactored in order to be independent from the GMLD software components and integrated as GSSC DataLab tool.

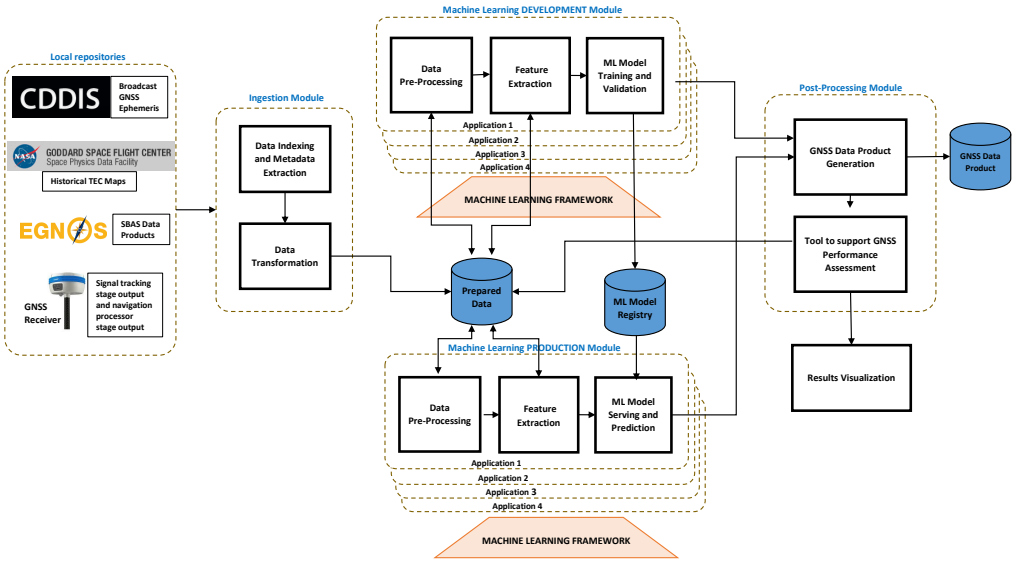


Fig. 1: Conceptual scheme of the demonstrator.

III. TEC MAPS PREDICITION

A. Application Description

It is known that total electron content (TEC) in ionosphere is defined as the total number of electrons integrated along the path from each GNSS satellite to the receiver. It is one of several indicators of ionospheric variability that impacts GNSS signals traveling through this layer of the athmosphere and interacting with the free electrons.

The objective of this application is to predict global TEC map sequences given the previous states of the ionosphere. When looking at the temporal evolution of TEC maps, they represent a time series of image frames, and the current status maps can be inferred from the ionosphere previous states. Temporal prediction of TEC maps is analogous to the well-studied problem of future frame prediction in video. This topic has recently received great attention by the ML community as it is a challenging problem that substantially benefits from the powerful representation learning capabilities of convolutional neural networks (CNN). TEC maps have a simpler content and smoother evolution than arbitrary video sequences and therefore it is possible to leverage the knowledge and architectures recently developed for video frame prediction, such as [23], [24] to effectively solve this task. In the approach proposed, the TEC map is treated as a whole, and the objective is the prediction of the entire map, exploiting the spatial and temporal correlation among the different points of the grid.

B. State of the Art

Forecasting of the TEC maps on a daily basis is a service already provided by GNSS services. This forecasting is based on classical regression methods, and recently methods based on ML and NN have been proposed. In [25]–[27], a large number of possible features are listed for this purpose: in addiction to base parameters like TEC value, other time series such as first- or second-time derivatives of TEC, some geomagnetic indices (e.g., Dst, Kp, ap, and AE), solar activity parameters (e.g. solar wind speed, sun spot number (SSN), solar flux index F10.7p, etc.), the solar radio emission at particular wavelength, etc. are considered. In [27], the authors extended forecasting TEC at a global scale, and the technique they proposed makes use of the NeQuick2 model [28] fed by an effective sunspot number R12 (R12eff), estimated by minimizing the root mean square error (RMSE) between nonlinear autoregressive with external input (NARX) neural network output and NeQuick2 applied at the same Global Ionospheric Map (GIM) grid points. These approaches basically implement regression of parameters that can be used to generate the TEC maps based on these estimated indices injected in the ionospheric model (e.g. NeQuick). The regression is applied on a selected grid of point from which the values for the global map are extrapolated, by utilizing, for example, NeQuick model as in [27], or by utilizing Bezier surface-fitting technique [29].

As previously remarked, the proposed application undertakes a different approach, treating each TEC map as a frame of a video thus leveraging effective ML and NN methods developed in the field of temporal motion (frame) prediction. The proposed application can be seen as extension of the approach proposed in [30], where a method based on deep neural network (DNN) has been proposed by combining several state-of-the-art architectures that process TEC maps treated as images. On the other hand, instead of a recurrent neural networks (RNN), the use of an architecture based on ResNets enables a more flexible design in terms of the memory model, the “prior” imposed on the data, and the ability to simultaneously obtain predicted outputs at multiple future time instants. The results presented in the following show

the capability to forecast at different time steps ahead, thus making this method applicable in a potential GNSS service, or in a receiver unable to get updated information neither from a SBAS satellite nor from a third party service.

The series of samples of the maps at a given time of the day can be used as an input for the forecast of the following days, at the same epoch. Prediction of future TEC maps from a buffer of past maps by means of frame prediction techniques allows to exploit both the spatial and temporal correlations present in the sequence, as illustrated in Figure 2.

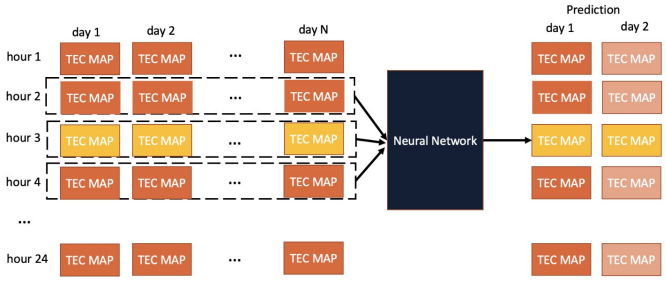


Fig. 2: Conceptual scheme of the TEC map prediction.

C. Algorithm selection method

The ResNet backbone architecture was chosen from experience as a suitable approach, as it is used in a wide variety of regression problems concerning images/videos (e.g., denoising and super-resolution). In this type of problems, ResNets are known to perform better than architectures without skip connections or without normalization layers, as (roughly stated) the skip connections enable each layer to learn just what is not already present in the input, rather than learning a model of the whole data [31]. Therefore, leveraging ResNets, the model selection has been focused on how to exploit the temporal correlation, rather than on the comparison of different architectures (e.g., ResNet versus a conventional CNN). Indeed, the scope of this preliminary approach was only showing that it is possible to improve over conventional, non-ML, approaches and that 3D convolutions over a multichannel input where the channels are 2 hours apart are more effective at short-term TEC prediction than the 2D convolution approach that only uses the same hour of the day. Vice versa, it was also shown that the latter is advantageous for long-term prediction. Variations on this architecture, such as different connections, number of layers, type of layers, hyperparameters, etc. have been the subject of the design stage for this application. We analyzed two candidate methods based on residual convolutional neural networks. The two models are customized to the TEC map prediction problem and make different assumptions about spatio-temporal correlation patterns in TEC maps, leading to different performance tradeoffs.

The first model concatenates TEC maps from the same hour-of-the-day as time T from the last K days, resulting in an input image with K channels. It is well known that due to Earth's rotation, the highest temporal correlation between TEC maps can be found between maps that are 24 hours apart. The model considers K maps each 24 hours apart to capture long-term trends in the evolution of TEC. This K-channel input image is then fed to a residual network such as the one in Figure 3. A ResNet architecture composed of

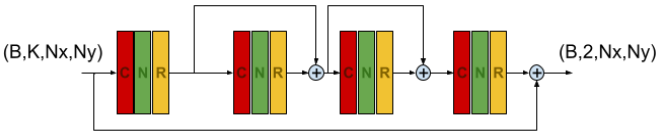


Fig. 3: Neural network architecture. C: conv2D layer (model 1), conv3D layer (model 2); N: Batch Normalization; R: ReLU non-linearity. Input and output tensor shapes follow the channel-first dimensional convention.

alternating a 2D convolutional layer, batch normalization, and ReLU non-linearity is used. We can notice that two residual blocks of convolution, normalization, non-linearity are used to build a deeper model. The output convolution returns an image with two channels. Each channel is summed with the image representing the TEC map at T-22 hours to generate the output which can be interpreted as the predictions for time T+2 hours and T+26 hours. This global skip connection allows to exploit the strong correlation between the map at T-22 hours and the future maps so that the neural network only has to estimate a residual correction. The filters in the convolutional layers have spatial size 5x5 to ensure that a sufficiently large receptive field is available to extract meaningful spatial features.

The second model concatenates all the available past TEC maps from T to T-K hours, creating an input image with K channels. Notice that, in contrast to the first method, even maps at hours-of-the-day that are different from the target one are exploited. The reason for this choice is that it allows better exploitation of short-term spatial and temporal correlation patterns thanks to the smooth evolution of TEC over temporally subsequent maps. In order to accomplish this, this second model uses a neural network architecture similar to the one in the first method but using 3D convolutional layers instead of 2D convolutional ones (see Figure 3). Convolution over the time dimension allows to track the smooth temporal evolution of the features extracted by spatial convolution. This justifies the need for the 3D operator in place of the 2D, which would have merged all the time steps in the first layer, losing descriptive power. As before, batch normalization and ReLU non-linearities are used, and a global skip connection ensures that the model only computes the residual with respect to the T-22 hours map. The filters in the convolutional layers have spatial size 5x5x5 to ensure that a sufficiently large receptive field is available to extract meaningful spatial features, and that its temporal extension is longer than 24 hours so that it can capture the strong correlations with 24h periodicity.

Both models are trained by minimizing the mean squared error (MSE) between the predictions and the ground truth TEC maps at T+2 hours and T+26 hours, available in the training set. During training, TEC maps sampled at steps of 2 hours

over the January 2001 - December 2003 time period were used, while the testing stage was fed with the maps from the January 2004 – December 2004 time period. It should be remarked that both models have a modest number of trainable parameters since the amount of training data is limited. Since this amount cannot be increased by orders of magnitude to what is typically available for video sequences, it is argued that more sophisticated models in the video frame prediction literature cannot be reliably trained or would not provide significant advantages due to overfitting.

According to the preliminary analysis, both candidate ML models provide improvements over the benchmark methods, but they offer different tradeoffs. The second model design favors the exploitation of the smooth temporal evolution via 3D convolution to extract short-term correlation patterns and, therefore, it provides more accurate predictions for maps in the near future. Vice versa, the first candidate model shows slightly better performance on the long-term prediction of the T+26 hours map. This is due to the design which targets finding long-term trends, as previously mentioned. Hence, the first model is recommended if there is an interest in long-term prediction, while the second is preferable for short-term prediction. The presented work favors short-term prediction, and therefore the choice of the second model was done. In particular, the final model manages also the presence of additional parameters, i.e. the sunspot number in our case. Input data are prepared by creating additional layers of repeated values to be stacked with the TEC maps, according to the scheme in Figure 4, where $N_x = 71$ and $N_y = 73$ are the dimensions of the TEC map, K is the number of previous TEC maps to be used as input for the model, that is the length of the TEC maps series. At each time step, a map containing the repeated values of the series is created and added as an additional layer. Combined data have thus input size (N_x, N_y, K, m) where $m = 2$ is the number of features of the ML model in the channels-last convention, i.e. TEC and sunspot number. Following this approach, it is easy to scale the input size when more features are considered.

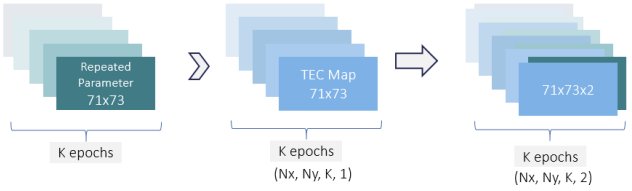


Fig. 4: Input data preparation for the final model.

D. Results

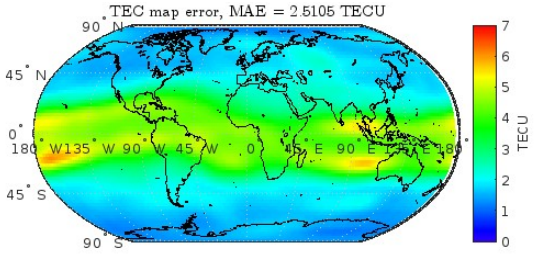
The dataset selection has been realized by considering the maximum periods of the Solar Cycles 23 and 24 (e.g., November 2001 and April 2014), and 4-year datasets (i.e., 2001-2004) have been created by processing downloaded CODE IONEX data. Then, the datasets are divided into training (70 %), validation (20 %), and test (10 %) sets. The results shared in the following belong to the test results.

The analysis of the overall TEC prediction performance has been carried out with respect to a ground truth of TEC maps provided as an IGS product. The estimated maps provided by the ML model have been used to compute the average vertical ionospheric bias error (AVIBE) affecting the pseudorange. The average error has been computed for each latitude and longitude pair, on a grid of 71x73 points, averaging over 4279 subsequent observations of the TEC map. TEC maps have been estimated as a forward prediction of 2 or 4 hours. The common scenario settings are reported in Table I.

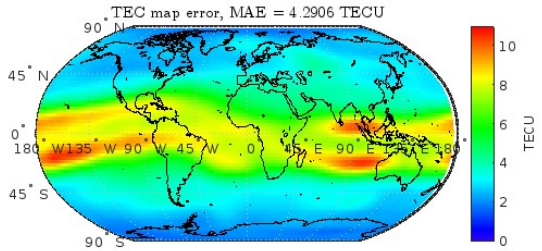
TABLE I: TEC maps prediction. Performance evaluation scenario.

Parameter	Value
Pseudorange measurement noise variance	0 m
User motion	static
TEC map observations	4379
PVT method	LMS
Forward prediction	2-4 hours

The resulting AVIBE has been drawn onto an Eckert map projection. Figure 5a shows the AVIBE values for a 2 hours forward prediction of the TEC map. The whole error map can be aggregated into a mean absolute error (MAE) value of 2.5105 TECU, which corresponds to a 40.8 cm of vertical ionospheric bias. An additional analysis was also performed



(a) 2 hours forward prediction.



(b) 4 hours forward prediction.

Fig. 5: Average vertical ionospheric bias error. Average performed over consecutive snapshots.

by characterizing the vertical ionospheric bias error for each latitude. Hence, TEC values over the longitudinal coordinate and consecutive snapshots were aggregated to perform a statistical characterization, whose result is reported in the boxplots of Figure 6a. This summary statistics highlights the median value as the central red line mark, and the 25th and

75th percentiles indicated as the bottom and top edges of the box, respectively. The whiskers extend to the most extreme data points not considered outliers, and the outliers are plotted individually using the '+' marker symbol. A similar analysis was repeated for a 4 hours forward prediction. The Results are shown in Figures 5b and 6b. In this case a slightly higher MAE of 4.3 TECU is obtained over the whole map, corresponding to a vertical ionospheric bias error of 69.7 cm.

The map representation of the AVIBE highlights how the TEC is harder to be predicted in the equatorial region. The equatorial region is subject to the most intense ionospheric variations and the prediction of such a complex behaviour is a harder task. Moreover, the statistical analyses showed that this area is also subject to a more widespread distribution of the estimation error. Even in the worst case however, all the error in the 75th percentile are in the order of meters and even the outliers are bounded to less than 12 and 15 meters for the 2 and 4 hours prediction, respectively.

The assessment of the performance (predicted vs. actual) has been complemented evaluating the impact on the slant TEC computation within the grid in order to use the classical correction formula:

$$I_{(\rho_j)} = 40.3\frac{sTEC}{f^2} \tag{1}$$

Given a TEC map, three different user positions on Earth have been chosen to grant the analysis of three different case studies:

i. a scenario where the user is located in correspondence of the minimum vertical TEC prediction error (best case);
ii. a scenario in which the user is located where the vertical TEC prediction error is maximum (worst case);
iii. a scenario in which the user is located in correspondence to the mode value of the vertical TEC prediction error.

For all the case studies, a satellite with medium elevation (38°-46°) has been selected, given the observation date and time. After the selection of the satellite and the user location, the ionospheric pierce point is computed. The grid points surrounding the pierce point are interpolated and used to compute the vertical ionospheric delay. The slant factor is then computed and used to compute the slant delay. Given a randomly selected snapshot of the predicted vertical TEC (vTEC) map (see Figures 7a and 7a), the user position was set at three different locations, as reported in Tables II and III for 2 and 4 hours forward prediction. The elevation of the selected satellites are also reported.

TABLE II: Scenario settings for each case study. 2 hours forward prediction.

Case study	User location (lat,lon)	Selected satellite elev. (deg)
Best vTEC	84°, -129°	46°
Worst vTEC	9.4°, -99°	46°
Mode vTEC	82°, -129°	46°

The final results are reported in Table IV. This result has no general validity, but provides a relevant evaluation of the impact of the prediction error once the slant factor is applied. Over an entire TEC map, the worst prediction error led to a slant ionospheric bias error of less than 6 meters for the 2 hours prediction case. While for the 4 hours prediction a 10.6 m error has been found.

TABLE III: Scenario settings for each case study. 4 hours forward prediction.

Case study	User location (lat,lon)	Selected satellite elev. (deg)
Best vTEC	-48°, -164°	42°
Worst vTEC	7°, -104°	38°
Mode vTEC	82°, -129°	44°

TABLE IV: Slant ionospheric bias error for each case study.

Case study	Slant ionospheric bias error (cm)	
	2 hours	*4 hours*
Best vTEC	6.5	41.4
Worst vTEC	584.7	1060
Mode vTEC	9.7	113.9

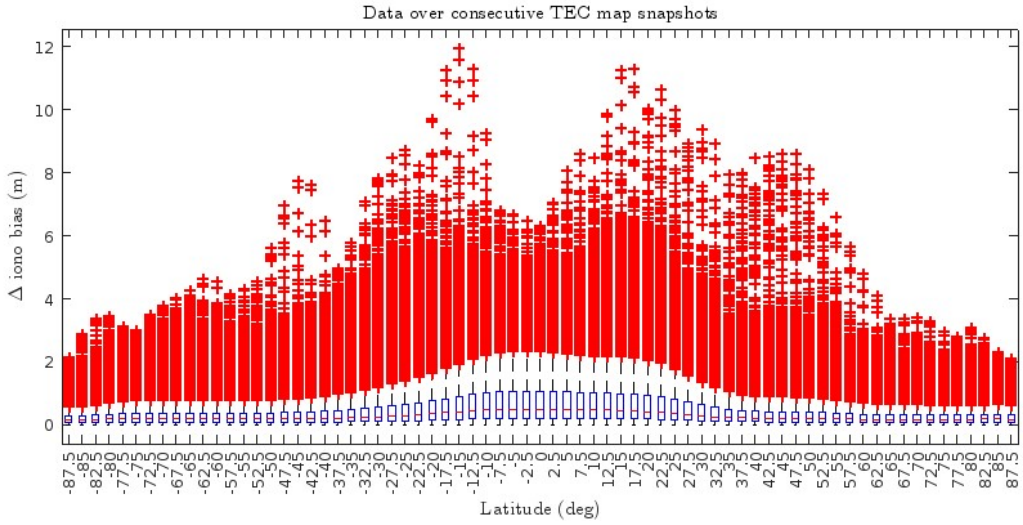
IV. SBAS CORRECTIONS PREDICTION

A. Application Description

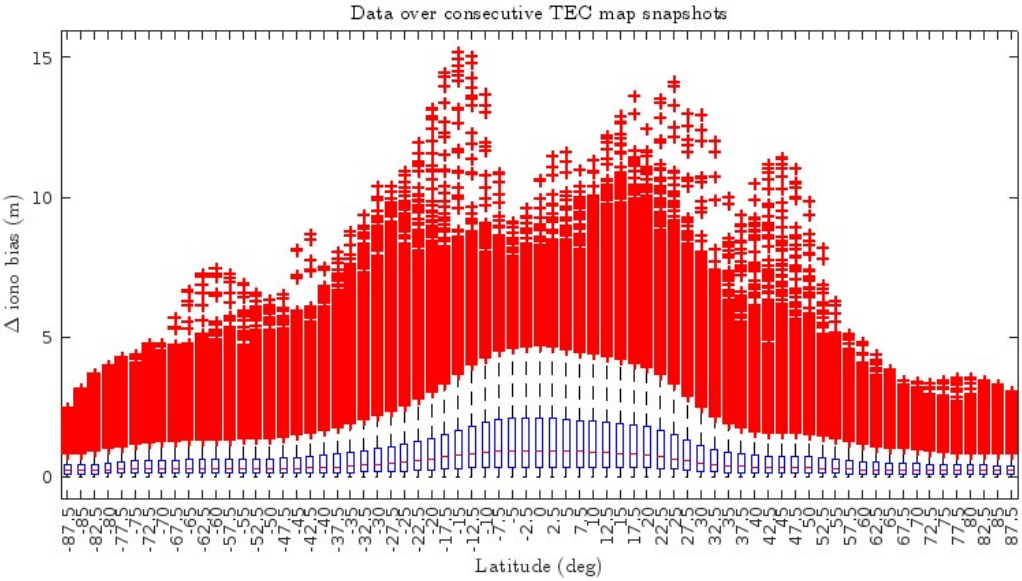
The SBAS navigation chain includes a network of reference stations aiming at collecting some data from a space segment (GPS and in the future Galileo) and a processing facility implementing the navigation algorithms that process this data to produce the SBAS navigation message. The processing facility is in charge of estimating the satellite corrections, ionospheric model and error variance related terms by processing all the data from the monitoring stating network. Moreover, advanced models for satellite orbit determination, satellite clock corrections, and integrity estimation are implemented in the processing cycles of the SBAS. From time to time, the user could miss some SBAS messages broadcasted. The use of the old data yields an increased error. The prediction of the parameters in the missing messages by utilizing the ML algorithms can be used to fill the gaps in the information flow. The main target users for this application, are non safety-of-life (SoL) users of wide area differential corrections, such as, for example, in the field of precision farming. In fact, SoL applications have strict requirements on the reliability of the data used for integrity assessment and the use of extrapolated information is questionable and would need a deep assessment of the algorithm confidence intervals that was out of the scope of this work.

B. State of the Art

In [32], the orbit, clock and ionospheric corrections derived from SBAS messages are comprehensively evaluated using data collected from the SBAS systems (e.g., WAAS, EGNOS, MSAS, GAGAN, SDCM) over 181 consecutive days. Mainly, the SBAS corrections include satellite orbit correction, satellite clock correction, and ionospheric correction. The satellite orbit and clock corrections consist of long-term and fast correction



(a) 2 hours forward prediction.



(b) 4 hours forward prediction.

Fig. 6: Statistical characterization of vertical ionospheric bias error over longitude and consecutive snapshots.

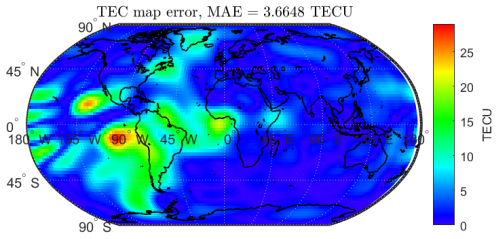
data. The long-term corrections are intended to correct satellite orbit errors and the slow-varying part of satellite clock errors, while the fast corrections are designed to compensate for rapidly changing part of GPS clock errors. Furthermore, in the experimental tests and analysis, the number of missing epochs for each SBAS satellite is counted and given in detail, which constituted a basis for the development of this application. In [33], Support Vector Machines (SVM) for long-term GNSS clock offset prediction were investigated along with polynomial regression, and Kalman filtering, and tested for GPS and GLONASS satellites. It is also indicated that the obtained results significantly improve the clock predictions relative to extrapolation with the basic clock model of the last obtained broadcast ephemerides.

C. Algorithm selection method

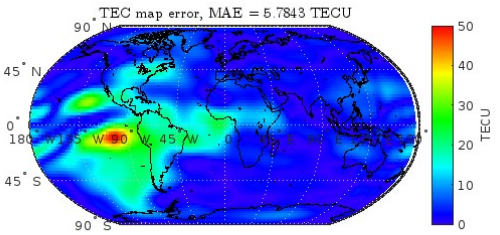
This problem is a typical case of a time series prediction for forecasting the future values that can also be used to train a regressive model able to provide the prediction of the missing data. Only the corrections from satellites included in the PRN mask are used to feed the ML algorithm. The prediction of the trend of corrections over time has been approached through supervised methods, three of which were considered and compared, here listed in increasing order of complexity.

i. *Linear regression*, which is a type of supervised ML algorithm, describes the relationship between a response and predictors. The linearity in a linear regression model refers to the linearity of the predictor coefficients β:

$$y_i = \beta_0 + \beta_1 X_{i1} + ... + \beta_p X_{ip} + \epsilon_i, \quad i = 1, ..., n \quad (2)$$



(a) 2 hours forward prediction.



(b) 4 hours forward prediction.

Fig. 7: Snapshot of a vertical TEC predicton map.

where y_i is the i-th response and X_{ij} is the i-th observation on the j-th predictor variable where $j = 1, .., p$.

ii. A *multistep dense network* learns an arbitrary model of the relation between the past and the future samples; the learned model is fixed, as there is no conditional dependence on the past samples. In other terms, it is a non-linear model without memory. A multi-step dense model is a sequential model of which layers are given in Table V. To address the issue of how the input features are changing over time, the model needs access to multiple time steps when making predictions, as depicted in Figure 8.

TABLE V: Main layers in the multi-step dense model architecture

No.	Layer
1	Flattening
2	Deeply connected neural networks (NNs) layers
3	Batch normalization (BN)
4	Rectified linear unit (ReLU)

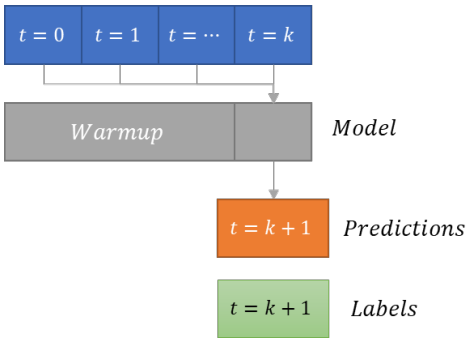


Fig. 8: Multi-step dense model architecture.

iii. *Long short-term memory (LSTM)* is a type of RNN well-suited to make predictions based on time series data. RNNs process a time series step-by-step, maintaining an internal state from time-step to time-step, as depicted in 9. An RNN is a non-linear model with memory; indeed, it can be seen as a machine with state, where the state is changed as a function of the previously observed values of the time series.

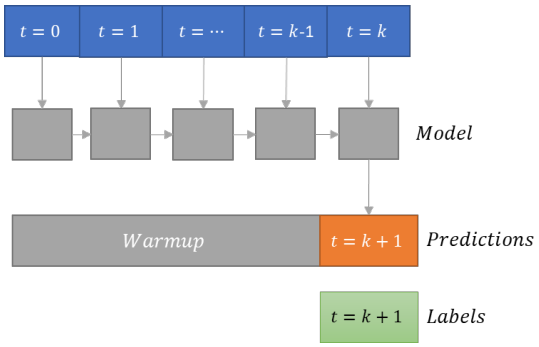


Fig. 9: RNN Architecture.

For the fast corrections, the results indicated that for the cases in which the prediction time ahead is equal to or lower than 8 seconds, the LSTM method performs better than the linear regression and multi-step dense models. However, when the prediction time increases, the accuracies of the linear regression model is better than the LSTM and multi-step dense models. For the slow corrections, the results are also in line with the ones obtained for fast corrections. However, in this case, multi-step dense performs better than the linear regression algorithm for the cases in which the prediction time ahead is equal to or lower than 60 seconds. If the prediction time is at two-time steps ahead or closer (e.g., ≤ 8 s for fast corrections, ≤ 60 s for slow corrections), LSTM method could be the primary model. Whereas linear regression could be a second choice for fast corrections, multi-step dense could be preferred as a secondary method for slow corrections. For more time-steps ahead prediction, linear regression could be preferred as the primary model.

D. Results

European Geostationary Navigation Overlay Service (EGNOS) .ems files that belong to the period between October 16, 2020, and November 15, 2020, have been downloaded from the EMS FTP service. They were decoded and parsed into the different Message Types (MTs) and fed to the ML engine. Then, the datasets were divided into training (70 %), validation (20 %), and test (10 %) sets. The results shared in the following belong to the test sets. For the sake of brevity, we discuss hereafter the results on the fast correction prediction performance.

The assessment of the model performance at a GNSS-user level was performed by setting up a proper simulation environment. Synthetic pseudoranges, corrected with SBAS fast corrections, have been used to compute the user position.

Satellite positions and user location have been generated consistently with such corrected pseudoranges. The given user location is then used as a ground truth with respect to which all the positioning errors are computed. The PVT computation with such exact measurements allows to define

i. a "best case scenario", where the SBAS messages are received and pseudoranges are profitably corrected;

ii. a "worst case scenario" in which the SBAS corrections are missing, and therefore the PVT is computed using non-corrected pseudoranges, leading to positioning errors that are proportional to the magnitude of such missed corrections;

iii. a benchmark scenario, in which the SBAS corrections are missing, but the corrections are estimated through a linear extrapolation formula, as defined in [34]

$$PR_{corr}(t) = PR_{meas}(t) + PRC(t_{of}) + RRC(t_{of}) \cdot (t - t_{of}) \quad (3)$$

with

$$RRC(t_{of}) = (PRC_{current} - PRC_{previous})/\Delta t \quad (4)$$

where PR is the pseudorange, PRC is the fast correction, RRC is the range-rate correction, t_{of} is the time of the last received correction and t is the current time;

iv. a scenario in which, despite the missed reception of SBAS messages, the correction terms are predicted through a ML approach.

The comparative analysis benchmarks the estimated user position comparing the predicted corrections (*iv*) to case (*i-iii*). The SBAS corrections in (*iii*) and (*iv*) have been computed as a forward prediction of 4, 8,12 and 16 s. Table VI summarizes the common parameters for scenarios (*i*)-(*iv*).

TABLE VI: SBAS corrections prediction. Performance evaluation scenario.

Parameter	Value	Notes
Pseudorange measurement noise variance	0 m	
User motion	static	
Observation time	20 hours	4 s step
PVT method	LMS	
Forward prediction	4-16 s	
No. of satellites	6	corrections applied to all

The estimated user position is shown in a 3D ECEF frame in Figure 10. It is worth noting that the position computed under the circumstances of case (ii) (absence of fast corrections) is more erratic around the true user position. Conversely, the position estimated in case (iv) is more concentrated around the true value. A better evaluation can be done by examining the user position error over its spatial coordinates. Figure 11 shows the user state error for the ground truth case (i), while Figures 12 and 13 show the user state error in scenarios (ii-iv) for 4, 8, and 12, 16 seconds forward prediction, respectively.

The resulting MAE are also summarized in Table VII for ease of comparison. Looking at MAE values, the prediction

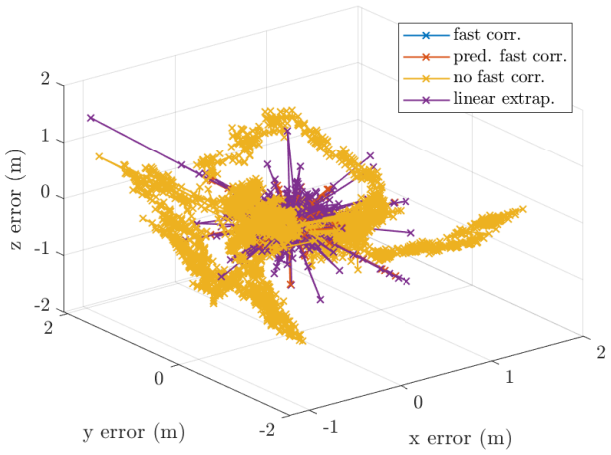


Fig. 10: Estimated user position error in ECEF coordinates. 4 seconds prediction.

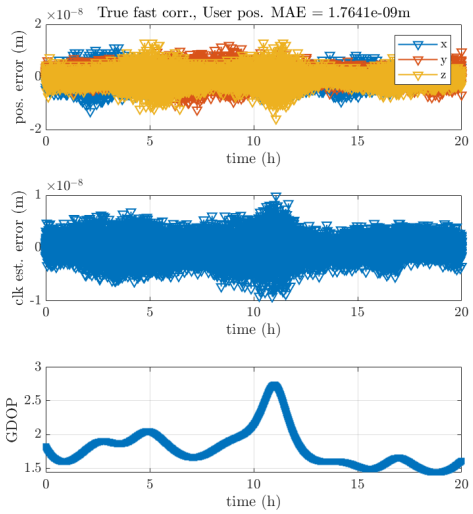
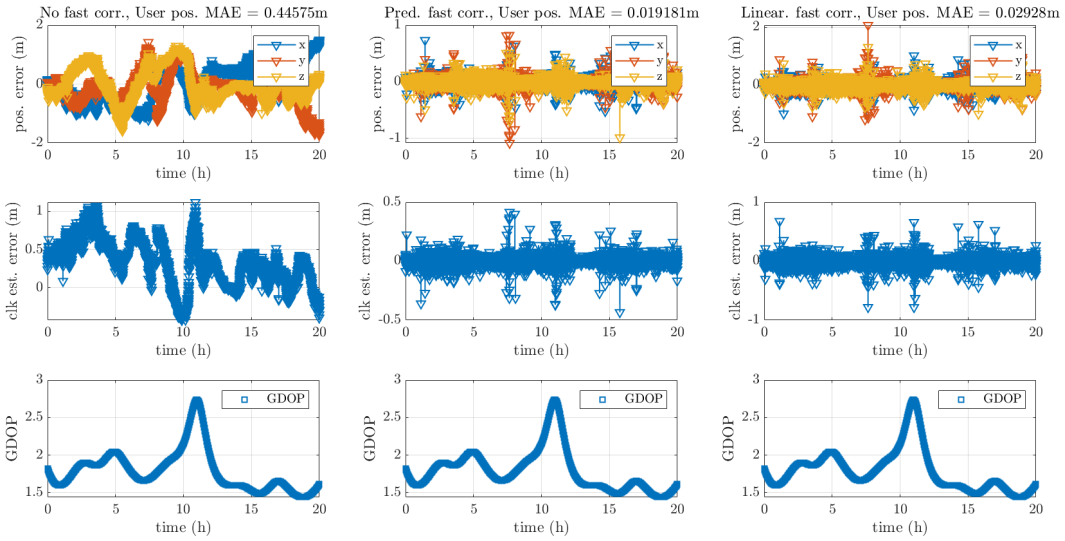
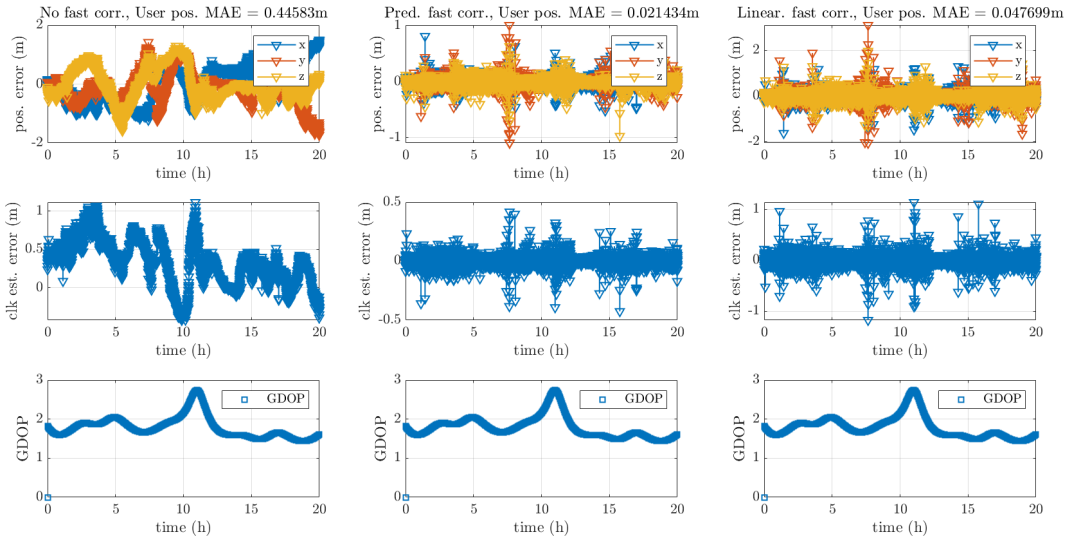


Fig. 11: User state estimation error. Ground truth scenario (i). 4 seconds forward prediction.

of the fast corrections can provide a MAE of position improvement of more than one order of magnitude, even for a forward prediction interval of 16 seconds. Moreover, the investigated methods outperform the MAE of user positions obtained by linearly propagating old fast corrections (case (iii)). Nonetheless, the performance gap becomes larger by increasing the forward prediction window. This means that a linear prediction is prone to a fast degradation with respect to the proposed ML approach, which on the contrary shows to cope well with longer burst of missing data. If we look at the positioning errors plots in Figures 12 and 13, we can see that, while the position computed without fast corrections is fluctuating around a null error, the correction prediction leads to position errors over spatial coordinates that are often very close to zero. This behavior is consistent with what seen in



(a) 4 seconds forward prediction.



(b) 8 seconds forward prediction.

Fig. 12: User position estimation error. Case studies (ii-iv) comparison.

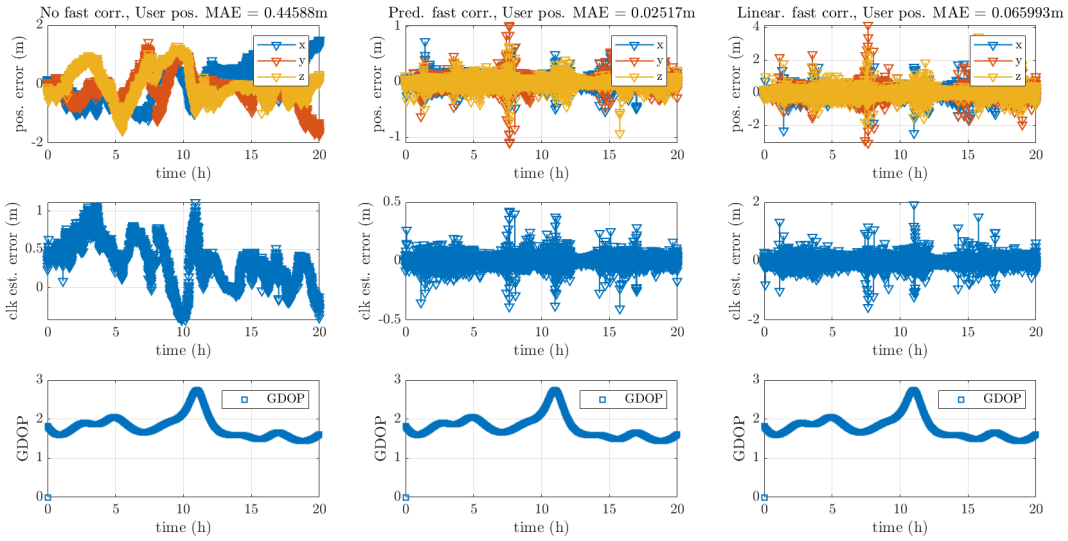
TABLE VII: MAE of user position estimation for different case studies and forward prediction interval.

Forward prediction	**MAE (m)**			
	Case (i)	***Case (ii)***	***Case (iii)***	***Case (iv)***
4 s	1.7e-9	0.45	0.029	0.019
8 s	1.8e-9	0.45	0.048	0.021
12 s	1.8e-9	0.45	0.066	0.025
16 s	1.7e-9	0.45	0.084	0.028

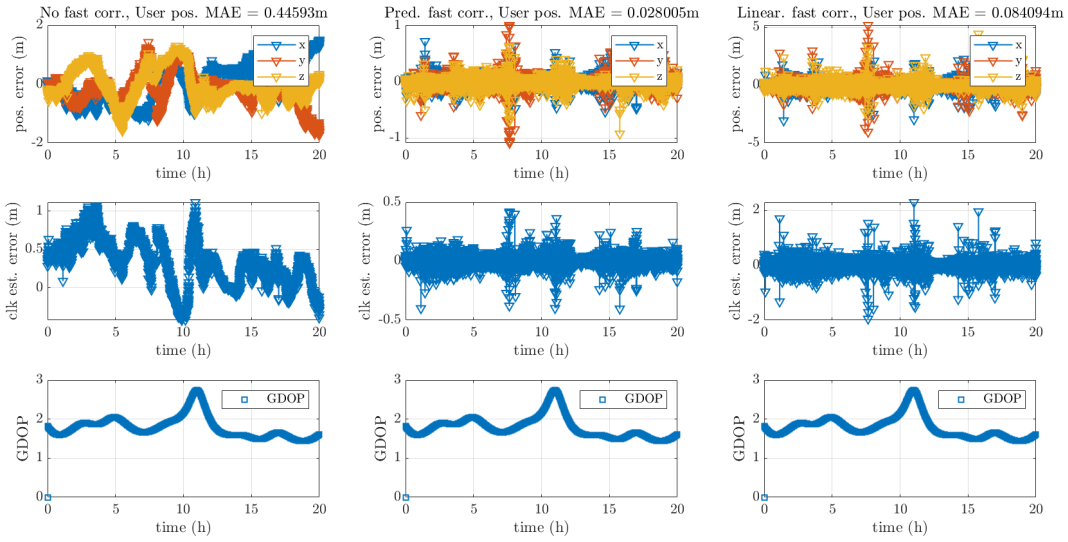
Figure 10, for the 3D position. The technique used for the correction prediction is not only effective from an average performance point of view (MAE of few centimeters) but it also provide an estimation that does not show evidence of systematic errors.

V. Conclusion

The method for the estimation of the ionospheric maps proposed provided satisfactory results, compared to the complexity of the input data, which are the time series of the previous TEC maps and sunspot number values. Other methods in the literature aimed at predicting the different variables that are used to construct such a map. But as highlighted while discussing the input data for this work, the retrieval of historical time series of such parameters is not always trivial. Results showed how TEC is harder to be predicted in the equatorial region. The equatorial region is subject to the most intense ionospheric variations and the prediction of



(a) 12 seconds forward prediction.



(b) 16 seconds forward prediction.

Fig. 13: User position estimation error. Case studies (ii-iv) comparison.

such a complex behaviour is a harder task. Moreover, the statistical analyses showed that this area is also subject to a more widespread distribution of the estimation error. A hybrid approach in which the predicted TEC map is fine tuned by the prediction of some geophysics parameters would be worth to be investigated, with the specific goal of having a prediction suitable also to the use in such areas of the world.

The results obtained for the fast corrections have demonstrated the suitability of the method for non-SoL applications, especially when compared to the absence of such corrections. The results obtained suggest that could be interesting to use ML based prediction for other kind of corrections, even obtained by sources different from the EGNOS messages.

ACKNOWLEDGMENT

This work has been developed in the framework of the activity “Machine-Learning to model GNSS systems” funded by the European Space Agency (NAVISP-EL1-035.02). The project was carried out in the scope of NAVISP Element 1, which is dedicated to technology innovation of the European industry in the wide PNT sector. The authors would like to thank Dr. Caner Savas for his contributions to develop this work, during the early stages of the project.

REFERENCES

[1] R. Ioannides, T. Pany, and G. Gibbons, “Known vulnerabilities of global navigation satellite systems, status, and potential mitigation techniques,” *Proceedings of the IEEE*, vol. 104, no. 6, pp. 1174–1194, 2016.

[2] M. Pini, A. Minetto, A. Vesco, D. Berbecaru, L. M. C. Murillo, P. Nemry, I. De Francesca, B. Rat, and K. Callewaert, "Satellite-derived time for enhanced telecom networks synchronization: the root project," in *2021 IEEE 8th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 2021, pp. 288–293.

[3] A. Minetto, A. Nardin, and F. Dovis, "GNSS-only collaborative positioning among connected vehicles," in *Proceedings of the 1st ACM MobiHoc Workshop on Technologies, MOdels, and Protocols for Cooperative Connected Cars*, ser. TOP-Cars '19. New York, NY, USA: Association for Computing Machinery, 2019, p. 37–42. [Online]. Available: https://doi.org/10.1145/3331054.3331552

[4] N. Williams, P. B. Darian, G. Wu, P. Closas, and M. Barth, "Impact of Positioning Uncertainty on Connected and Automated Vehicle Applications," *SAE International Journal of Connected and Automated Vehicles*, vol. 6, no. 12-06-02-0010, 2022.

[5] A. Minetto, A. Nardin, and F. Dovis, "Tight integration of gnss measurements and gnss-based collaborative virtual ranging," in *Proceedings of the 31st International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2018)*, Miami, Florida, September 2018, pp. 2399–2413.

[6] ——, "Modelling and experimental assessment of inter-personal distancing based on shared gnss observables," *Sensors*, vol. 21, no. 8, 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/8/2588

[7] A. Nardin, A. Minetto, O. Vouch, M. Maiani, and F. Dovis, "Snapshot acquisition of GNSS signals in space: a case study at lunar distances," in *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022)*, Denver, Colorado, September 2022, pp. 3603 – 3617.

[8] A. Minetto, F. Dovis, A. Nardin, O. Vouch, G. Impresario, and M. Musmeci, "Analysis of GNSS data at the Moon for the LuGRE project," in *2022 IEEE 9th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 2022, pp. 134–139.

[9] P. Borhani-Darian and P. Closas, "Deep neural network approach to gnss signal acquisition," in *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, 2020, pp. 1214–1223.

[10] A. Nardin, T. Imbiriba, and P. Closas, "Jamming source localization using augmented physics-based model," 2022. [Online]. Available: https://arxiv.org/abs/2212.08097

[11] G. Michalak, M. Dassie, A. Testa, G. Giorgi, C. Günther, and K. H. Neumayer, "Signal-in-space range error analysis of the simulated broadcast ephemerides for the kepler system." in *Proceedings of the 35th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2022)*, Denver, Colorado, September 2022.

[12] A. Nardin, J. A. Fraire, and F. Dovis, "Contact plan design for GNSS constellations: A case study with optical intersatellite links," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 58, no. 3, pp. 1981–1995, 2022.

[13] A. Nardin, F. Dovis, and J. A. Fraire, "Empowering the tracking performance of LEO PNT by means of meta-signals," in *2020 IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)*, 2020, pp. 153–158.

[14] ——, "Empowering the tracking performance of LEO-based positioning by means of meta-signals," *IEEE Journal of Radio Frequency Identification*, vol. 5, no. 3, pp. 244–253, 2021.

[15] N. Linty, A. Minetto, F. Dovis, and L. Spogli, "Effects of phase scintillation on the gnss positioning error during the september 2017 storm at svalbard," *Space Weather*, vol. 16, no. 9, pp. 1317–1329, 2018.

[16] Y. Jiao, J. J. Hall, and Y. T. Morton, "Automatic equatorial gps amplitude scintillation detection using a machine learning algorithm," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 53, no. 1, pp. 405–418, 2017.

[17] ——, "Performance evaluation of an automatic gps ionospheric phase scintillation detector using a machine-learning algorithm," *NAVIGATION*, vol. 64, no. 3, pp. 391–402, 2017. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/navi.188

[18] N. Linty, A. Farasin, A. Favenza, and F. Dovis, "Detection of GNSS ionospheric scintillations based on machine learning decision tree," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 55, no. 1, pp. 303–317, 2019.

[19] C. Savas and F. Dovis, "The impact of different kernel functions on the performance of scintillation detection based on support vector machines," *Sensors (Switzerland)*, vol. 19, no. 23, 2019.

[20] L.-T. Hsu, "Gnss multipath detection using a machine learning approach," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, 2017, pp. 1–6.

[21] G. Franzese, N. Linty, and F. Dovis, "Semi-supervised GNSS scintillations detection based on deepinfomax," *Applied Sciences (Switzerland)*, vol. 10, no. 1, 2020.

[22] V. Navarro, J. Ventura-Traveset, R. Prieto, J. Cegarra, M. d. M. Millan, and S. del Rio, "Esa gnss science support centre a world-wide reference gnss environment for scientific communities," in *Proceedings of the 33rd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2020)*, September 2020, pp. 1174–1199.

[23] H. Choi and I. V. Bajić, "Deep frame prediction for video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 7, pp. 1843–1855, 2020.

[24] M. Oliu, J. Selva, and S. Escalera, "Folded recurrent neural networks for future video prediction," in *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part XIV*. Berlin, Heidelberg: Springer-Verlag, 2018, p. 745–761.

[25] A. Zhukov, D. Sidorov, A. Mylnikova, and Y. Yasyukevich, "Machine learning methodology for ionosphere total electron content nowcasting," *International Journal of Artificial Intelligence*, vol. 16, pp. 144–157, 04 2018.

[26] R. Orus Perez, "Using tensorflow-based neural network to estimate gnss single frequency ionospheric delay (iononet)," *Advances in Space Research*, vol. 63, no. 5, pp. 1607–1618, 2019.

[27] Cesaroni, Claudio, Spogli, Luca, Aragon-Angel, Angela, Fiocca, Michele, Dear, Varuliator, De Franceschi, Giorgiana, and Romano, Vincenzo, "Neural network based model for global total electron content forecasting," *J. Space Weather Space Clim.*, vol. 10, p. 11, 2020. [Online]. Available: https://doi.org/10.1051/swsc/2020013

[28] B. Nava, P. Coïsson, and S. Radicella, "A new version of the nequick ionosphere electron density model," *Journal of Atmospheric and Solar-Terrestrial Physics*, vol. 70, no. 15, pp. 1856–1862, 2008, ionospheric Effects and Telecommunications.

[29] E. Tulunay, E. T. Senalp, S. M. Radicella, and Y. Tulunay, "Forecasting total electron content maps by neural network technique," *Radio Science*, vol. 41, no. 4, 2006.

[30] A. Boulch, N. Cherrier, and T. Castaings, "Ionospheric activity prediction using convolutional recurrent neural networks," 2018.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[32] Z. Nie, P. Zhou, F. Liu, Z. Wang, and Y. Gao, "Evaluation of orbit, clock and ionospheric corrections from five currently available sbas l1 services: Methodology and analysis," *Remote Sensing*, vol. 11, no. 4, 2019.

[33] J. Pihlajasalo, H. Leppäkoski, S. Kuismanen, S. Ali-Löytty, and R. Piché, "Methods for long-term gnss clock offset prediction," in *2019 International Conference on Localization and GNSS (ICL-GNSS)*, 2019, pp. 1–6.

[34] SC-159, *Minimum Operational Performance Standards for Global Positioning System / Wide Area Augmentation System Airborne Equipment (RTCA MOPS DO 229 (Revision D)*, RTCA, Inc., 1828 L Street, NW, Suite 805, Washington, D.C. 20036, USA, 2006.