Prediction of geomagnetic events from solar wind data using deep learning

Prediction of geomagnetic events from solar wind data using deep learning

Maurizio Lo Schiavo
Department of Electronics and Telecommunications
Politecnico of Turin
Torino, Italy
maurizio.loschiavo@polito.it

Enrico Magli
Department of Electronics and Telecommunications
Politecnico of Turin
Torino, Italy
enrico.magli@polito.it

Silvano Fineschi
Astrophysical Observatory of Torino
National Institute for Astrophysics
Pino Torinese, Italy
silvano.fineschi@inaf.it

Gianalfredo Nicolini
Astrophysical Observatory of Torino
National Institute for Astrophysics
Pino Torinese, Italy
gianalfredo.nicolini@inaf.it

Daniele Telloni
Astrophysical Observatory of Torino
National Institute for Astrophysics
Pino Torinese, Italy
daniele.telloni@inaf.it

***Abstract*—The recent technological maturity attained by deep learning drew the attention of numerous scientific communities. Among these, Space Weather is leveraging such tools to support its activities related to forecasting harmful events. Coronal mass ejections (CMEs) are one of the most critical phenomena occurring in the solar system: their propagation may impact the Earth, altering the equilibrium of the terrestrial surface under different aspects, requiring their prediction to take countermeasures accordingly. Classical forecasting methods are built upon solar remote-sensing observations to forecast the CME onset, intensity, and arrival time. Although such methods could provide alerts within 1-4 days in advance, their estimations are affected by large uncertainties. On the other hand, deep learning has been offering valid alternatives through recent studies, devising data-driven models to obtain real-time alerts while monitoring such events remotely.**

The goal of this work is that of developing neural network architectures able to offer CME predictions leveraging the Lagrangian point L1 measurements, taking advantage of historical data related to these phenomena past behaviors to predict future trends.

In this paper, two main phases may be distinguished: first, the manipulation of the dataset - mostly through augmentation techniques - to make it more suitable for the proposed prediction steps. Second, the implementation of different network structures for multiple classification tasks concerning various aspects of CMEs, to prove the effectiveness of deep learning algorithms in reaching the desired goal.

***Index Terms*—coronal mass ejection, data augmentation, deep learning, neural network**

I. Introduction

CMEs are one of the largest expressions of solar activity, defined as large eruptions of magnetized plasma from the sun's atmosphere - the corona - into interplanetary space. Their origin is similar to solar flares, which are bursts of electromagnetic radiation, deriving from the twisting and realignment of the sun's magnetic field.

The hypothetical harms are well exposed in [1], stating that "those solar activities could result in potential slowdown and orbital decay of the low-Earth-orbiting satellites, induction of very harmful electric currents in power transmission grids and pipelines, disruption of satellite signal propagation with severe implications for positioning and communication systems, and unrecoverable failures of electronics onboard spacecraft".

Compared to solar flares, whose travel occurs at the speed of light and reaches Earth in approximately 8 minutes, CMEs travel at a more leisurely pace: "at their highest speeds of almost 1,900 miles per second (3,000 kilometers per second), CMEs can reach Earth in about 15 to 18 hours whilst slower CMEs traveling around 155 mi/s (250 km/s) can take several days to arrive" [2]. Such an aspect represents a beneficial factor, as the relatively slow travel speed gives those concerned more time to prepare for potential arrivals.

Classical CME forecasting methods are categorized into three types: physics-based, event-based, and drag-based; their implementation is built upon remote-sensing observations of such ejections providing expectations of their onset, intensity, and arrival time. One of the biggest deficiencies of current forecasting methods is represented by the scarce usage of in-situ solar wind data measured at the Lagrangian point L1, essentially due to the challenging work of locally identifying CMEs from in-situ observations, whose deployment could potentially provide more accurate warnings concerning CME arrival and strength.

Nowadays, deep learning techniques have been increasingly adopted in many different research fields, providing cross-disciplinary research between computer science and solar and heliospheric physics [3]. In this regard, [1] highlights a technological readiness in implementing deep learning tools for real-time prediction of geomagnetic events. Also, [4] remarks that the recent advances in deep learning can significantly benefit solar sciences.

A. Related work

Multiple works have been presented over the last years, applying deep learning tools on various datasets to predict specific geomagnetic disturbances, mainly CMEs and solar flares.

In this regard, some results using deep learning architectures for prediciton of geomagnetic events have been presented by Telloni, Lo Schiavo et al. in [7]. Dhuri et al. [8] implement logistic regression and gradient boosting classifier to understand the underlying mechanisms governing flares. Sudar et al. [9] analyze transit times of CMEs using a simple feed-forward neural network, providing as input only the CME velocity and the central meridian distance of its associated flare, and observing that transit time dependence on velocity is showing a typical drag-like pattern in the solar wind. Delouille et al. [10] demonstrate that coronal holes and filaments could be distinguished in solar extreme ultraviolet images through the usage of a machine learning algorithm combined with segmentation techniques. A new approach is proposed in [11], with Liu et al. building a prediction engine for CME arrival time forecasting, equipped with the support vector machine algorithm and based on partial-/full halo CME. Wang et al. [12] make use of a convolutional neural network (CNN) regression model, giving as input only the instances of the white-light observation (images) of CMEs to predict their arrival time; Shi et al. [13] utilize logistic regression on a set of CME parameters like central position angle, angular width, and linear velocity (derived from large angle and spectrometric coronagraph LASCO images) to predict whether a CME will hit or miss the Earth's surface, and in case of a hit, their expected arrival time. A new tool for CME detection and tracking is presented in [3], composed of three modules. Firstly, a LeNEt network solves a supervised image classification problem, considering images containing CME structures. CME regions are then spotted through a deep descriptor transforming (DDT), able to localize the object of interest in the image set. In the end, a graph-cut technique is applied to finely tune the detected CME region. Again, Besliu-Ionescu et al. [14] developed a logistic regression model to forecast whether a CME will impact the terrestrial surface and will be associated with geomagnetic storms. In [15], Sun et al. propose a multimodality solar wind prediction method that jointly learns vision and sequence information in a unified end-to-end framework. Specifically, their prediction tool consists of three modules: Vmodule, Tmodule, and Fusion module. Vmodule uses pre-trained GoogLeNet to learn visual representation from the extreme ultraviolet images, Tmodule applies a combination of a one-dimensional CNN and a bidirectional LSTM (BLSTM) for learning sequence representation on a multivariate time series, and a Fusion module to improve the overall performance.

B. Purposes of the work

The goal of the present paper is to develop deep learning tools for the prediction of geomagnetic events, leveraging temporal time series composed of solar wind data collected in situ at the L1 point along with measurements of geomagnetic indexes to be used as ground truth. The collection took place over 15 years, with observation recorded every minute, resulting in more than 7 million data points.

By digesting the data, the models should be able to forecast the future evolution of solar events, and specifically to predict future values of geomagnetic indexes, which can be used to generate alerts in case of detection of potentially critical events that might be harmful to the Earth.

The prediction task is actually cast as a classification problem, where deep models such as neural networks are trained to determine the intervals, corresponding to different danger degrees, which the targeted geomagnetic index will belong to.

Training a model for this task requires overcoming two issues. First, the data are highly non-stationary, making for a challenging prediction problem. Second, since this is an anomaly detection problem the dataset is highly unbalanced, as the CMEs are rather rare with respect to regular behavior, thus requiring the careful design of the training procedures in order to obtain an informative prediction model.

II. Methodology

This work aims to classify future CME intensity, spotting the ones with a critical degree, using deep learning tools like neural networks. The targeted feature is represented by the so-called SYM metric for H component (SYM-H), one of the most important indices for Space Weather, indicating the intensity of the magnetic storm, similar to the disturbance storm time (DST) index but with a much higher time-resolution [5]. Rather than predicting its continuous value, the values assumed by the index are divided into intervals corresponding to different levels of danger, and models are trained to estimate the index class in a given time in the future.

Specifically, multiple classification problems have been considered, aiming to perform forecasts differing in the temporal extension - how long in the future the estimation is pointing at - and its degree of severity. Regarding this latter, we considered binary scenarios with the critical (SYM-H value lower than -50nT) and non-critical classes, and categorical scenarios including four classes: non-critical, moderate ($-100nT \leq \mathrm{SYM-H} \leq -50nT$), intense ($-250nT \leq \mathrm{SYM-H} \leq -100nT$), and super ($\mathrm{SYM-H} \leq -250nT$) events [6]. A model is trained for a specific scenario adopting a specific look-ahead time for the prediction.

A. Dataset

The dataset used for this work consists of a multivariate time series represented by a collection of solar wind and geomagnetic indices, made available by the astrophysics observatory located in Pino Torinese. The observations were gathered in situ L1 from the 1st of January 2005 to the 31st of December 2019. Each observation includes 20 features measured with a one-minute resolution, resulting in 7.888.320 data points overall. All the features composing the dataset are listed in Table I.

TABLE I
FEATURES COMPOSING THE DATASET

Index	Name	Measure unit
0	Time	[s]
1	Time shift	[s]
2	Magnetic intensity B	[nT]
3	Bx	[nT]
4	By	[nT]
5	Bz	[nT]
6	Solar wind bulk speed	[km/s]
7	Vx	[km/s]
8	Vy	[km/s]
9	Vz	[km/s]
10	Proton density	$[cm^{-3}]$
11	Proton temperature	[K]
12	Flow pressure	[nPa]
13	Plasma pressure	[nPa]
14	Magnetic pressure	[nPa]
15	Total pressure	[nPa]
16	Plasma beta	A-dimensional
17	Alfvénicity	[s]
18	AE	[nT]
19	SYM-H	[nT]

As pointed out before, the SYM-H is not directly estimated; instead, the danger interval at which it belongs is targeted. This means that the target value is a class label, which can assume 1 out of 2 possible values for the binary scenarios and 1 out of 4 for the categorical ones. Regarding the input, the availability of a collection of temporal-related observations leads to the construction of a set of time windows, obtained by extracting from the overall time series a specific number of consecutive equally-spaced data points. Given the physical properties of the problem, a reasonable setup for those windows includes 24 observations taken every hour, so as to represent the daily trend. After this data windowing step, the reshaped dataset will have three dimensions: the number of windows extracted from the entire time series (7886371), the length of each window (24), and the number of features for each point within the window (20). The next analyses regarded the dataset composition, as it should be noted that training deep networks implies the usage of a dataset with a sufficient heterogeneity of samples characterizing all event classes. Unfortunately, the physical nature of CMEs implies a rare manifestation of dangerous phenomena, resulting in a high unbalance between the critical and non-critical classes, constituting respectively the 2% and 98% of the overall set. Therefore, the next work step addressed the implementation of augmentation strategies to combat the imbalance. The augmentation procedure has been executed on just a part of the dataset, corresponding to its 90%. This fraction will constitute the train and validation set for the algorithm training. The remaining 10% will instead compose the test set, representing the future events, which should preserve the unbalanced property; hence, no augmentation has been done on the test set.

Most of the techniques employed belong to the random transformation category, both along the magnitude and time domains [16]. These are the jittering, with random noise added to windows samples, scaling, magnitude, and time warping to compress/extend somewhat the discrete signal representing the event.

Moreover, an augmentation strategy called SMOTE (synthetic minority oversampling technique) [16] has been used, this belonging to the pattern mixing category. Such a technique interpolates two windows considered neighbors according to the dynamic time warping distance, better than Euclidean distance at catching resemblance among patterns with similar but misaligned trends.

In the end, undersampling the majority class first, and using augmentation techniques for oversampling the minority led to a more balanced distribution of more than 4 million windows, composed of 32% critical and 68% not critical events.

After achieving a satisfactory balance, multiple neural network architectures have been designed and trained for each task, differing in model architecture and complexity. These include simpler models such as linear network, multilayer perceptron (MLP), CNN, and long short-term memory (LSTM). More complex compositions were developed by combining previous ones, i.e. a sequential cascade of CNN (deep CNN), the adoption of a skip connection (deep CNN with skip, similar in spirit to a ResNet), and the combination of LSTM and a fully convolutional network (LSTMFCN).

A detailed description of the architecture settings is offered in Table II.

Other parameters related to the training procedure and common to all networks are the loss function used (cross-entropy), the number of epochs equal to 100, the optimizer (Adam), and the set of values proposed for the batch size and learning rate hyperparameters, respectively (32, 64, 128, 256) and (0.001, 0.0001, 0.00001).

B. Description of experiments

Five scenarios were defined for the classification task. In scenarios 1 and 2, the models forecast the criticality class 1 hour ahead, adopting a binary and categorical classification, respectively. As defined earlier, in the binary case the classifier only estimates whether the event will be critical or not, in the categorical one it will distinguish among multiple ranges.

Scenario 3 and scenario 4 repeated the previous two tasks, providing this time estimations with a longer time horizon equal to 8 hours. Those four scenarios are expressions of single-step models, where future index estimations refer to only one specific time instant. A multi-step approach is instead pursued in scenario 5, predicting for several consecutive time instants whether the event will be critical or not.

The multi-step prediction could be carried out in two different ways: on one hand, through an iterated multi-step (IMS) forecasting, where the model learns a single-step and iteratively applies it to obtain multi-step prediction through an autoregressive approach; on the other hand, direct multi-step (DMS) to estimate all the numerous predictions at once. This latter tends to suffer less from the error accumulation effect; moreover, it is preferred when the future number of time instants is relatively large. Those considerations led to

TABLE II
ARCHITECTURAL COMPOSITION

NETWORK	Feature extraction layers	Input layer nodes	Hidden layer nodes	Output layer nodes	Activation function
LINEAR	0	0	144	$N_{classes}$	ReLU and Softmax
MULTIDENSE	2 dense	96 + 80	80	$N_{classes}$	ReLU and Softmax
CNN	1 conv	(5,24,6)	64	$N_{classes}$	ReLU and Softmax
LSTM	1 lstm	72	64	$N_{classes}$	ReLU and Softmax
DeepCNN	4 conv	9216x4	12	$N_{classes}$	ReLU and Softmax
LSTMFCN	4 conv + 1 lstm	1536x4 + 120	64	$N_{classes}$	ReLU and Softmax
DeepCNN$_{Skip}$	4 conv	9216x4	12	$N_{classes}$	ReLU and Softmax

the choice of the DMS approach, dealing with up to 8 future values.

Seeking the optimal model setup implied the adoption of a trial and error approach for scenarios 1 and 2, tuning the architectures to a set of factors, composed of the **input features**, the structure of the feature **extraction block**, the hyperparameters for the network training procedure **batch size** and **learning rate**, and other options like the implementation of a **class weight** and **adaptive learning rate** strategy. The best configuration of each model was obtained in these two scenarios, using precision, recall, F1 score, and balanced accuracy as evaluation metrics. This latter is necessary as overall accuracy would be misleading due to the unbalanced nature of the problem. Such setups were then employed also in the other three scenarios.

III. RESULTS

A. Parametric analyses

The preliminary phase for the architectural parameter research performed in scenarios 1 and 2 pointed out the difficulty of choosing the feature set. Giving all the twenty solar wind features as input resulted in training issues, preventing models from inferring representative patterns. Therefore, two different subsets were tested: set A with 11 features (**Total pressure**, **SWB Speed**, **B**, **Bx**, **By**, **Bz**, **Vy**, **Vz**, **Plasma Pressure**, **Proton Density**, **SYM-H**) and set B with 6 (**Total pressure**, **SWB Speed**, **B**, **Bz**, **Proton Density**, **SYM-H**), employing fewer features whose correlation with the physical phenomenon is easier to learn. Likewise, two strategies for the feature extraction block were considered, consisting in learning a multi-feature representation (JOINT approach) in one case and treating each feature independently (DISJOINT) in the other.

A graphical representation of the architectural composition in the two cases is offered in Fig. and 1 and 2.

Other solutions were investigated, like the adoption of a class weight strategy, which aims at penalizing misclassifications made on the minority class by setting a higher class weight while reducing weight for the majority class. During the training, a higher weight will be given to the minority class in the algorithm's loss function, leading to a higher penalty for such class, making the algorithm focus on reducing the errors for this latter.

The last solution regards the adaptive learning rate, which dynamically changes the learning rate whenever the loss value is not significantly decreasing anymore, contributing to accelerate training and alleviating some of the pressure of choosing a learning rate and learning rate schedule.

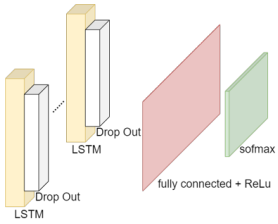

Fig. 1. Architecture implementing DISJOINT extractor

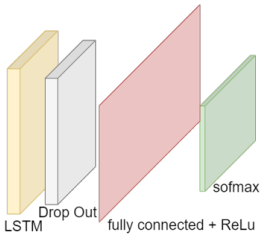

Fig. 2. Architecture implementing JOINT extractor

The best configuration for each model is summarized in Tables III and IV.

A first investigation regards the best combination of options, considering each batch size and learning rate value for each model, combined with each feature extractor block type and set of inputs. In particular, the gap among model performance equipped with the two feature extractor blocks is rather obvious. Either implementing set A or set B, simpler models behave better when adopting a feature extractor for every single input. Conversely, complex models work differently in accordance with the scenario: in the case of binary classification, performance is slightly better with the JOINT approach, while the DISJOINT feature extraction is more effective during the categorical one.

A comparison between input subsets highlighted better performance with set B: dealing with an intricate dataset suggests working with fewer, more relevant inputs. Equipping each of the seven best configurations with the class weight strategy improved performance; on the contrary, the adaptive

TABLE III
SCENARIO 1: BEST MODEL CONFIGURATIONS

NETWORK	FEATURE SET	FEATURE EXTRACTION	(BS, LR)	CLASS WEIGHT	ADAPTIVE LR
LINEAR	SET B	DISJOINT	(128, 0.0001)	YES	NO
MULTIDENSE	SET B	DISJOINT	(128, 0.00001)	YES	NO
CNN	SET B	DISJOINT	(256, 0.0001)	YES	NO
LSTM	SET B	DISJOINT	(256, 0.0001)	YES	NO
DeepCNN	SET B	JOINT	(256, 0.001)	YES	NO
LSTMFCN	SET B	JOINT	(128, 0.00001)	YES	NO
DeepCNN$_{Skip}$	SET B	JOINT	(64, 0.00001)	YES	NO

TABLE IV
SCENARIO 2: BEST MODEL CONFIGURATIONS

NETWORK	FEATURE SET	FEATURE EXTRACTION	(BS, LR)	CLASS WEIGHT	ADAPTIVE LR
LINEAR	SET B	DISJOINT	(64, 0.001)	YES	NO
MLP	SET B	DISJOINT	(256, 0.00001)	YES	NO
CNN	SET B	DISJOINT	(128, 0.00001)	YES	NO
LSTM	SET B	DISJOINT	(64, 0.00001)	YES	NO
DeepCNN	SET B	JOINT	(128, 0.001)	YES	NO
LSTMFCN	SET B	DISJOINT	(256, 0.00001)	YES	NO
DeepCNN$_{Skip}$	SET B	DISJOINT	(128, 0.001)	YES	NO

learning rate did not lead to any enhancement.

B. Performance evaluation

Results of scenarios 1 and 2 can be seen in Tables V and VI. Regarding the binary case, P_{NC}, P_C, R_{NC}, R_C and $F1_{NC}$, $F1_C$ are respectively the precision, recall, and F1 score of critical and non-critical events. Regarding the categorical one, P_{NC}, $P_{Moderate}$, $P_{Intense}$, P_{Super}, R_{NC}, $R_{Moderate}$, $R_{Intense}$, R_{Super} and $F1_{NC}$, $F1_{Moderate}$, $F1_{Intense}$, $F1_{Super}$ are respectively the precision, recall, and F1 score of non-critical, moderate, intense, and super events. In both scenarios, $Acc_{Balanced}$ is the balanced accuracy.

In scenario 1, LSTM performed slightly better than other models, achieving a balanced accuracy equal to 93%; conversely, the MLP and deep CNN showed worse performance. The very good results achieved by most models highlight that this first scenario is not so challenging. Moreover, all models present a recall value higher than the precision, highlighting their non-conservative behavior in predicting critical events. Such an outcome is in agreement with the different weight given to the two event classes: obtaining a false alarm may result in less harm than missing a critical event.
In scenario 2, simpler models confirm their suitability to the single-step problem, with LSTM showing the highest performance again, with an accuracy of around 70%.
However, the categorical classification turned out to be problematic, as the scarcity of some event classes, even after the augmentation, did not allow the models to return proper estimations.
A specific aspect that stands out from all architecture results is the high ability to detect intense events and the complete inability in spotting super events.
Fortunately, the super class represents not even 1% of the overall events, limiting the scope of the problem.

Examining the feature extraction block alternatives, simpler models showed better performance when treating the features disjointly. Unlike scenario 1, the precision values are higher than the recall ones, resulting in weaker predictive capacities for critical events.
In both scenarios 1 and 2, the most complex models did not show improved results, displaying a particular deficiency at detecting super events. Overall, MLP and deepCNN showed lower predictive capacity, suggesting their exclusion for the following scenarios.

Moving to the next scenarios, what immediately stood out was the increased level of difficulty of the problem, as general performance is worse than in previous ones, as is visible in the Tables VII and VIII.

The recurrent approach showed the best results, where scenario 3 reveals the highest balanced accuracy of about 69% by the LSTMFCN model, thus showing its effectiveness at capturing longer temporal dependencies.
Such performance drops to 44% in scenario 4, remarking the lack of balance among the critical event classes: the longer time horizon shows the necessity of increasing the number of intense events too, as none of the models has been able to detect them with such a long time length in scenario 4. Deep CNN with the skip connection offers good results, showing the same effectiveness as the LSTM. It is useful to look at the precision values, which are higher than the recall ones like in scenario 2, thus expressing a more conservative approach

TABLE V
SCENARIO 1: MODEL RESULTS

NETWORK	[P_{NC}, P_C]	[R_{NC}, R_C]	[$F1_{NC}$, $F1_C$]	$Acc_{Balanced}$
LINEAR	[0.999, 0.641]	[0.997, 0.857]	[0.998, 0.734]	0.927
MLP	[0.999, 0.652]	[0.997, 0.819]	[0.998, 0.726]	0.908
CNN	[0.999, 0.648]	[0.997, 0.843]	[0.998, 0.733]	0.920
LSTM	[0.999, 0.620]	[0.997, 0.876]	[0.998, 0.736]	0.936
DeepCNN	[0.999, 0.628]	[0.997, 0.772]	[0.998, 0.693]	0.885
LSTMFCN	[0.999, 0.614]	[0.997, 0.806]	[0.998, 0.697]	0.902
$DeepCNN_{Skip}$	[0.999, 0.628]	[0.997, 0.823]	[0.998, 0.712]	0.910

TABLE VI
SCENARIO 2: MODEL RESULTS

NETWORK	[P_{NC}, $P_{Moderate}$, $P_{Intense}$, P_{Super}]	[R_{NC}, $R_{Moderate}$, $R_{Intense}$, R_{Super}]	[$F1_{NC}$, $F1_{Moderate}$, $F1_{Intense}$, $F1_{Super}$]	$Acc_{Balanced}$
LINEAR	[0.999, 0.534, 0.943, 0]	[0.996, 0.861, 0.943, 0]	[0.998, 0.659, 0.943, 0]	0.701
MLP	[0.999, 0.534, 0.921, 0]	[0.997, 0.796, 0.878, 0]	[0.998, 0.639, 0.899, 0]	0.667
CNN	[0.999 0.533 0.989, 0]	[0.996, 0.824, 0.920, 0]	[0.998, 0.647, 0.954, 0]	0.685
LSTM	[0.999, 0.509, 0.932, 0]	[0.996, 0.878, 0.943, 0]	[0.998, 0.645, 0.938, 0]	0.704
DeepCNN	[0.998, 0.603, 0.952, 0]	[0.998, 0.646, 0.865, 0]	[0.998, 0.624, 0.906, 0]	0.627
LSTMFCN	[0.999, 0.557, 1, 0]	[0.997, 0.752, 0.866, 0]	[0.998, 0.640, 0.928, 0]	0.654
$DeepCNN_{Skip}$	[0.999, 0.548, 0.962, 0]	[0.997, 0.708, 0.871, 0]	[0.997, 0.618, 0.914, 0]	0.6440

TABLE VII
SCENARIO 3: MODEL RESULTS

NETWORK	[P_{NC}, P_C]	[R_{NC}, R_C]	[$F1_{NC}$, $F1_C$]	$Acc_{Balanced}$
LINEAR	[0.995, 0.550]	[0.998, 0.311]	[0.997, 0.397]	0.5256
CNN	[0.996, 0.397]	[0.997, 0.301]	[0.997, 0.342]	0.649
LSTM	[0.995, 0.588]	[0.999, 0.324]	[0.997, 0.418]	0.662
LSTMFCN	[0.996, 0.694]	[0.999, 0.391]	[0.998, 0.500]	0.695
$DeepCNN_{Skip}$	[0.996, 0.708]	[0.999, 0.283]	[0.998, 0.404]	0.641

TABLE VIII
SCENARIO 4: MODEL RESULTS

NETWORK	[P_{NC}, $P_{Moderate}$, $P_{Intense}$, P_{Super}]	[R_{NC}, $R_{Moderate}$, $R_{Intense}$, R_{Super}]	[$F1_{NC}$, $F1_{Moderate}$, $F1_{Intense}$, $F1_{Super}$]	$Acc_{Balanced}$
LINEAR	[0.996, 0.571, 0, 0]	[0.999, 0.252, 0, 0]	[0.998, 0.369, 0, 0]	0.404
CNN	[0.996, 0.130, 0, 0]	[0.993, 0.227, 0, 0]	[0.994, 0.166, 0, 0]	0.407
LSTM	[0.995, 0.406, 0, 0]	[0.999, 0.201, 0, 0]	[0.997, 0.269, 0, 0]	0.400
LSTMFCN	[0.995, 0.253, 0.290, 0]	[0.997, 0.300, 0.015, 0]	[0.996, 0.223, 0.028, 0]	0.440
$DeepCNN_{Skip}$	[0.996, 0.282, 0, 0]	[0.997, 0.232, 0, 0]	[0.996, 0.255, 0, 0]	0.410

adopted by the models; however, this results in less ability to predict critical events. Overall, the necessity of increasing the number of events characterized by higher intensity stood out as a limiting factor.

Scenario 5 introduces a new level of complexity, requiring the models to predict several labels corresponding to the event class at i-hour ahead, with $i = 1, 2, \ldots 8$ through the DMS approach. Once completed the training for the five architectures, the test step provides balanced accuracy as a function of the look-ahead time, here displayed in Fig. 3.

As expected, models present a decreasing accuracy trend. The deep CNN with the skip connection presents the best result, maintaining an accuracy higher than 0.7 for all eight predictions. On the contrary, LSTM does not appear quite

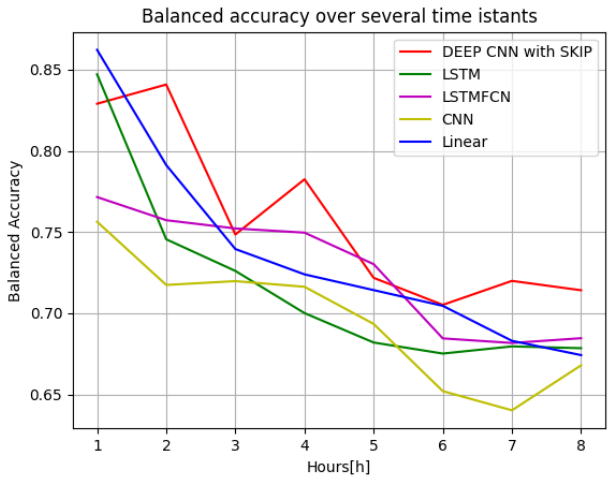

Fig. 3. Scenario 5: balanced accuracy over time

as good as in the single-step problem, resulting even worse than the linear model, despite the good prediction capabilities shown for the very first value ahead. In this regard, the LSTMFCN offers interesting prediction capability with the hours in the middle of the period considered; however, it does not perform as good as the deep CNN with the skip connection.

IV. Conclusions and Future Works

The goal of this work was that of classifying future CME intensity, spotting the ones with a critical degree, employing deep learning tools like neural networks fed with a time series dataset. The main challenge regarded the data augmentation, highly needed given the strongly imbalanced nature of such measurements. Multiple classification problems have been designed to infer different pieces of information about the CME arrival time and its degree of severity. Scenario 1 highlighted the high difficulty in detecting the optimal set of input features, ending up with the one containing fewer but more relevant ones. Considering also scenario 2, the single-feature extraction block offered on average high performance, as the conflict among the set of features suggested treating them independently at the feature extraction stage. Looking at the results of these two scenarios, LSTM performed slightly better than other models; however, the method does not work well in the categorical classifications, as the scarcity of super events, even after the augmentation, did not allow to classify them. Scenarios 3 and 4 remarked the higher effectiveness of the recurrent approach for single-step prediction, where LSTMFCN outperformed all the other methods.

In the end, what stood out is the necessity of increasing the number of events characterized by higher intensity, both super and intense, as fulfilling a categorical task requires a better balance among the strongest disturbances. Scenario 5 highlighted the insufficient multi-prediction capabilities of the LSTM model - slightly enhanced by the implementation of the LSTMFCN - and the good performance of the deep CNN with the skip connection, which outperformed all the others, guaranteeing a balanced accuracy for the detection of criticality in at least the 70% of cases.

Overall, deep learning tools proved to be a valid alternative to classical physics-based models, pointing out how the exponential growth of data and computing capacity offers valid support for very complex problems.

Further enhancement of this study could be oriented toward a deeper understanding of the features to be used for the prediction, analyzing one-by-one the features excluded and testing whether their individual contribution could be constructive or disruptive.

Other augmentation techniques like generative models could be contemplated, mainly to increase the diversity of intense and super events. The integration of different datasets could be taken into consideration, combining the constructed time series windows with coronagraph images.

References

[1] Telloni, D., 2022, Frontiers in Astronomy and Space Sciences, 9, 865880, doi: 10.3389/fspas.2022.865880

[2] NoAA, Coronal mass ejections, https://www.swpc.noaa.gov/phenomena/coronal-mass-ejections#:~:text=Coronal%20Mass%20Ejections%20(CMEs)%20are,magnetic%20field%20(IMF)%20strength

[3] Wang, P., Zhang, Y., Feng, L., Yuan, H., Gan, Y., Li, S., Lu, L., Ying, B., Gan, W., & Li, H., 2019, A New Automatic Tool for CME Detection and Tracking with Machine Learning Techniques, Astrophys. J., 244, 1, doi: 10.3847/1538-4365/ab340c

[4] Camporeale, E., 2019, Space Weather, 17, 1166, doi: 10.1029/2018SW002061

[5] Cai, L., Ma, S., Cai, H., Zhou, Y., & Liu, R., 2009, Prediction of SYM-H index by NARX neural network from IMF and solar wind data, Sci. China Ser. E-Technol. Sci. 52, 2877–2885, doi: 10.1007/s11431-009-0296-9

[6] Cander, L. R., & Mihajlovic, S. J., 1998, J. Geophys. Res.: Space Phys., 103, 391, doi: 10.1029/97JA02418

[7] Telloni, D., Lo Schiavo, M., Magli, E., Fineschi, S., Gustavino, S., Nicolini, G., Susino, R., Giordano, S., Amadori, F., Candiani, V., Massone, A., & Piana, M.,, 2023, Prediction Capability of Geomagnetic Events from Solar Wind Data Using Neural Networks, Astrophys. J., 952, 111, doi: 10.3847/1538-4357/acdeea

[8] Dhuri, D. B., Hanasoge, S. M., & Cheung, M. C. M., 2019, Machine learning reveals systematic accumulation of electric current in lead-up to solar flares, Proceedings of the National Academy of Science, 116, 11141. doi: 10.1073/pnas.1820244116

[9] Sudar, D., Vrsnak, B., & Dumbovic, M., 2016, Predicting coronal mass ejections transit times to Earth with neural network, Monthly Notices of the Royal Astronomical Society, Mon. Not. R. 440 Astron. Soc., 456, 1542, doi: 10.1093/mnras/stv2782

[10] Delouille, V., Hofmeister, S., Reiss, M., Mampaey, B., Temmer, M., & Veronig, A., 2018, Coronal holes detection using supervised classification, Catalyzing Solar Connections, doi: 10.1016/B978-0-12-811788-0.00015-9

[11] Liu, J., Ye, Y., Shen, C., Wang, Y., & Erdélyi, R., 2018, A New Tool for CME Arrival Time Prediction using Machine Learning Algorithms: CAT-PUMA, Astrophys. J., 855, 2, doi: 10.3847/1538-4357/aaae69

[12] Wang, Y., Liu, J., Jiang, Y., & Erdélyi, R., 2019, CME Arrival Time Prediction Using Convolutional Neural Network, Astrophys. J., 881, 1, doi: 10.3847/1538-4357/ab2b3e

[13] Shi, Y., Wang, J., Chen, Y., Liu, S., Cui, Y., & Ao, X., 2022, Impacts of CMEs on Earth Based on Logistic Regression and Recommendation Algorithm, Space: Science and Technology, 2022, doi: 10.34133/2022/9852185

[14] Besliu-Ionescu, D., & Mierla, M., 2021, Geoeffectiveness prediction of cmes, Frontiers in Astronomy and Space Sciences, 8, doi: 10.3847/1538-4357/ac7962

[15] Yanru, S., Zongxia, X., Yanhong, C., & Qinghua, H., 2022, Accurate Solar Wind Speed Prediction with Multimodality Information, Space Sci Technol., doi: 10.34133/2022/9805707
[16] Iwana, B.K., & Uchida, S., 2021, An empirical survey of data augmentation for time series classification with neural networks, PLoS ONE 16(7): e0254841, doi: 10.1371/journal.pone.025484