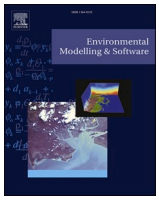
Environmental Modelling and Software 183 (2025) 106203

Contents lists available at ScienceDirect

Environmental Modelling and Software

journal homepage: www.elsevier.com/locate/envsoft



An integrated, automated and modular approach for real-time weather monitoring of surface meteorological variables and short-range forecasting using machine learning

R. Tsela, S. Maladaki, S. Kolios *

National Kapodistrian University of Athens, Faculty of Sciences, Department of Aerospace Science and Technology, 34400, Greece

ARTICLE INFO

Keywords:
Applied meteorology
Commercial off-the-shelf structures
Weather monitoring
Nowcasting
Machine learning algorithms
Ground stations

ABSTRACT

Weather monitoring and forecasting plays a vital role in a great variety of human activities such as agriculture, transportation, and extreme weather phenomena. This study presents the first outcomes of the development of a fully automated system regarding the real-time recording of basic meteorological parameters and their short-range forecasting (nowcasting). The system itself is divided into five core components: a hardware system for monitoring atmospheric conditions (Commercial Off-The-Shelf structures), a system for storing and managing data, a module for distributing data to support applications, a machine learning algorithm for nowcasting, and a user-friendly interface, all made by modern tools and methods, described analytically. Finally, the nowcasting procedure along with the relative accuracy results, is presented. The nowcasting procedure is based on a Long Short-Term Memory (LSTM) model scheme which is parametrized in such a way that reliable forecasts, up to 2 h ahead of time, can be provided.

1. Introduction

The meteorological parameters are essential physical factors for studying various phenomena within the atmosphere which affect weather, climate and consequently the physical environment on Earth's surface. These parameters play pivotal roles in understanding climate change, the intensification of the greenhouse effect, detecting local disturbances in climatic profiles, and predicting severe and extreme weather events. These examples comprise only a small number of situations where the monitoring of basic meteorological parameters is necessary. Moreover, the forecasting of these parameters offers valuable informational background for sustainable environmental management and future planning as well as warning about extreme weather phenomena.

There are large numbers of studies in international literature that analyze time-series weather related datasets with many meteorological parameters targeting different objectives. Significant is the category of studies where the central scope is to classify weather types and estimate potential climate change (e.g. Naik and Pathan, 2012; Olaiya and Adeyemo, 2012; Kotsias et al., 2022). Another characteristic category of studies based in meteorological parameters concern the analysis of extreme weather events (e.g. Goubanova and Li, 2007; Kalimeris et al., 2011; Karagiannidis et al., 2012; Varfi et al., 2009). Time-series of meteorological data are also used for short-term (or long-term) forecasting where the key objective is the determination of the evolution of these parameters in space and time. In such applications the integration of multivariate data from various sources and the utilization of several data management methods and algorithms is very common (Pourmousavi et al., 2011; Guo et al., 2014; Pierre and Monbet, 2012; Chattopadhyay et al., 2011; Almonacid et al., 2013; Abhishek et al., 2012; Brendel et al., 2019).

The correlations among meteorological variables are widely recognized and well-studied, however they can be notably intricate especially due to their localized nature and their rapid fluctuations on short timescales. The irregular variations of meteorological parameters require nonlinear and multi parametric statistical procedures to improve these conventional statistical methodological approaches. Here's where automated computer intelligent data processing, powered by advanced Machine Learning algorithms like LSTM networks can be invaluable. These statistical algorithms enable the generation of detailed models capable of capturing the intricate relationships driving short-range weather forecasts without requiring extremely powerful computers and a lot of processing time. In addressing this issue, solutions such as Artificial Neural Networks (ANN), fuzzy logic, and genetic algorithms (e.

* Corresponding author.
E-mail address: skolios@aerospace.uoa.gr (S. Kolios).

https://doi.org/10.1016/j.envsoft.2024.106203
Received 24 March 2024; Received in revised form 1 August 2024; Accepted 31 August 2024
Available online 10 September 2024

g. Saima et al., 2011; Aggarwal and Kumar, 2013; Bushara and Abraham, 2013; Ghafarian et al., 2022) can be employed. Indeed, in recent years, the field of weather prediction has witnessed significant advancements driven by the integration of cutting-edge technologies and various data-driven methodologies (e.g. Karvelis et al., 2017; Wu and Levinson, 2021; Chen et al., 2021; Nandi et al., 2024). Among these, Long Short-Term Memory (LSTM) artificial neural networks have emerged as a promising tool for enhancing the accuracy of short-range weather forecasting (e.g. Salman et al., 2018; Hoang et al., 2020; Tekin et al., 2023; Zenkner and Navarro-Martinez, 2023). Short-range weather prediction, encompassing forecasts over a span of a few hours to a couple of days, holds pivotal importance in sectors ranging from agriculture and energy management to transportation and disaster preparedness.

In particular, Recurrent Neural Networks (RNNs) are considered the state-of-the-art method for analyzing sequential time-dependent data such as the weather-data we are using here. What makes RNNs so special among a series of Machine Learning techniques is its ability to retain old information, making it capable of capturing connections and extracting patterns from sequentially structured data (Hochreiter and Jurgen Schmidhuber, 1997). This ability arises from the feedback mechanism, where the output from a previous step serves as an input to the current step and contributes as well to the formation of the next step. This mechanism forms something relative to our memory and emulates somehow the way we perceive and from thoughts about events that had not occurred yet, but we are still able to predict the outcome.

This study provides the methodological steps for the development of an integrated and accurate service regarding the nowcasting of basic meteorological parameters and their extremes. We describe the underlying hardware and software components used to monitor weather variables and collect real-time measurements. These components provide the building blocks for higher level front-end real-time interactive applications through well-defined interfaces, as well as a valuable data source that can be used to construct powerful prediction models.

In section 2, a high-level overview of the developed system is provided. In section 3, the study results are described. In section 4 the study outcomes are provided and in section 5, the conclusions of this study are presented.

2. Data and methods

All the weather data used in this study were collected from custom-made ground-based meteorological stations. The design of every aspect of our system incorporates four key principles: automation, sustainability, modularity, and simplicity. The system itself is divided into five core components: a hardware system for recording meteorological parameters, a system for storing and managing data, a module for distributing data to support applications, machine learning algorithms for short-range forecasting, and a user-friendly interface. In this section, a top-down overview of each component, beginning with the hardware system, is presented.

2.1. The hardware

We engineered custom monitoring systems that comprehensively handle everything from data collection to transmission across designated networks. Each independent monitoring unit (meteorological station) is equipped with basic sensors including thermometers, hygrometers, barometers, anemometers, wind vanes, and rain gauges. The entire structure of each unit was designed specifically to ensure compatibility between the selected sensors, accessibility to the external interfaces, and to supply the required operational power to the various system components. Each entity can operate autonomously, generate its own energy and provide remote access for data and control.

Each structure was built with Commercial Off-The-Shelf (COTS) components as well as a few custom electronic components with modularity in mind. Each station is composed of five modules as depicted in Fig. 1 (a power generation, storage and distribution module, a system controller, a set of internal state monitors, a set of sensor devices and a network adapter module).

Energy generation is exclusively based on solar radiation, where a solar panel is utilized for collecting solar energy through the day and converting it into valuable electric power for the system operation needs. The storage unit consists of a battery that can provide continuous power supply to the station for at least 2 days per full charge at 12VDC supporting the continuous operation of the structure even with low solar radiation through cloudy days and throughout the nighttime.

All actions of the station are controlled by an embedded 8-bit AVR microcontroller. Every unit can operate standalone without external intervention, controlling and monitoring its own state and adjusting the operation flow accordingly. Efficient algorithms control the inner state of the station programmed to protect the units from harsh conditions such as extreme heat, under-charging or over-charging, and loss of connection. At this point it is mentioned that custom software was developed to control all components utilizing standardized protocols such as I2C and SPI for interfacing.

Data transmission is based on the General Packet Radio Service (GPRS) through a Global System for Mobiles (GSM) adapter module embedded in every monitoring unit. Utilizing this approach the data collection is independent of the internal mass storage and the local network infrastructure, permitting in that way real time meteorological data collection at any geographical position of choice, since data is transmitted to remote backend systems but at the same time we can transmit control commands to them.

Three virtual links (one physical) were established between each station entity and the backend system: data uplink, telemetry uplink and command downlink respectively. We use TCP/IP over HTTP for all the aforementioned transfers following a client-server approach, keeping the design as simple as possible.

2.2. Backend data management system

The client-server approach utilized for the data transfer between the system controller and the monitoring units permits a highly abstract level and differentiation between the monitoring system and the data handling system, reducing in that way the design and implementation complexity of both entities. Also, that kind of modularity provides great scalability and easy upgradability for future expansion and studies.

The backend data management system is implemented on the server side and consists of a database for storing the related weather parameter values (temperature, humidity, barometric pressure, wind speed, wind direction and rainfall) that are recorded and transmitted from the

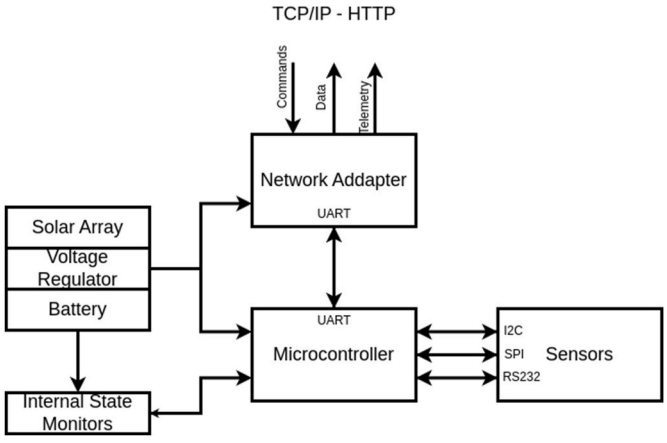


Fig. 1. The physical layer architectural block-diagram of the weather monitoring units.

different monitoring units. Stations only collect measurements (see Table 1). A series of low-level routines (hereinafter, "handlers") handle the data coming from these stations and redirect commands that go to these units. All low-level handlers are implemented in PHP and designed to be utilized through the HTTP application-layer protocol. The high-level of abstraction between the monitoring system and the data handling system achieved reduces the design and implementation complexity of both parts.

Each station transmits the measurements (Table 1) in a well-defined directory within the server (POST request) which is managed by the "store_data" handler. It receives the request from each of the stations concurrently, and extracts the measurements, organizes them and storing them within the aforementioned database (also stores these data into a file for backup purposes).

Along with the measurement data, telemetry is also provided by each sensor in order to monitor their state. Telemetry is handled by the "telemetry" handler. Telemetry data include the internal temperature, the voltage level, the current intensity, the heartbeat counter from the last system reset and the operational mode that the station is currently in. It is important to note here that each station structure is defined by a unique station identification code.

Moreover, for controlling several station's operations such as the definition of sampling period, the shutdown operation, the halting operation and the system reset, through the telecommand channel another handler is employed for both receiving the command "cmd_tx" from the user/controller and transmitting the command to the respective stations "cmd_rx". Commands have the form of "$station_id, command_id $" and are executed asynchronously by the stations.

Furthermore, we implemented a series of middle-ware handler-functions to simplify the interaction of applications with our system, constructing in that way a series of primitives that support easy application development, such as real-time monitoring systems and training machine learning algorithms for weather forecasting as we will present later in this paper. These handlers include the "daily_data" which is used to retrieve data for the current day from the database, the "stats" which provides statistics on a daily basis including mean, maximum and minimum values as well as rain accumulation over the selected time period. Another handler is the "dataset" which is used to create a dataset for a selected period of time of data measurements collected by a specific station - which implies a specific geographical region. Fig. 2 depicts the high-level architecture diagram of the backend data management system as described above.

2.3. The Application Programming Interface (API)

Along with the modular backend system and the physical entities an API(Application Programming Interface) was also implemented as part of this project. The API, referred to as pyWSN (Python Weather Station Network), is a Python package that encapsulates the complexity of accessing the backend system and requesting specific operations from

Table 1
The collection of COTS sensors employed for monitoring the relevant weather variables in each station unit.

Parameter	Range	Accuracy/ Resolution	Device/Component
Humidity	0-100 [%]	+-2 [%]	AM2315
Temperature	−20- 80 [C]	+-0.1 [C]	AM2315
Barometric Pressure	0.3–110 [KPa]	+-1 [hPa]	BMP280
Rainfall Rate	–	0.29 [mm]	SparkFun Electronics Tipping bucket
Wind speed	–	2.4 [Km/h]	SparkFun Electronics Cup Anemometer
Wind direction	0–360 [deg]	22.5 [deg]	SparkFun Electronics wind vane

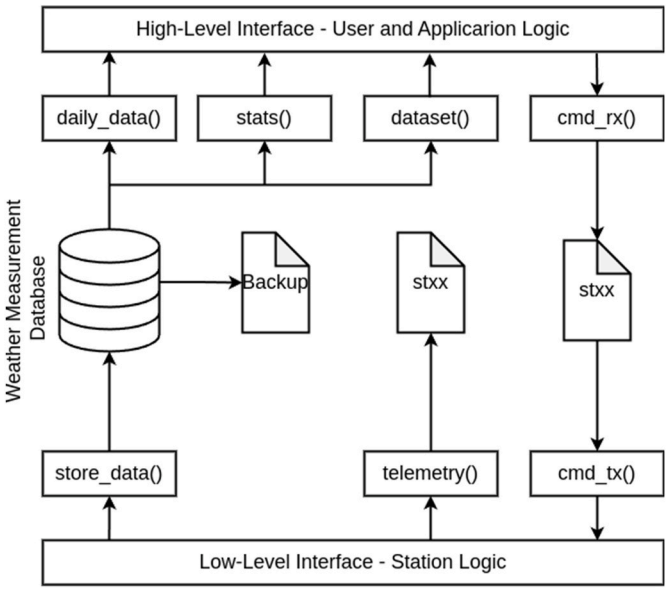


Fig. 2. The backend data management system architectural block-diagram.

our backend data handlers (as described in subsection II). The package is designed with modularity in mind and consists of a single class named "Weather_Station_Backend_Controller". This class includes a set of getter and setter member functions, alleviating the user from the need-to-know which handler to invoke and what parameters to use through the HTTP requests for a specific task.

The main target of the API is to provide a high-abstraction to the application developer by hiding all the complexity behind the low-level and web interfaces, speeding up in that way the application development as well as the debug time of web components (which is relatively hard). The API is designed to interface with the backend system running locally on the user's machine, managing all necessary HTTP requests for specific tasks. It facilitates access to daily data through a set of functions provided in Table 2.

The API enables users to build meteorological data-driven applications and conduct atmospheric-related studies without spending time on searching for data sources, formatting them, cleaning datasets, measuring statistics, or, in our case, learning how to invoke specific handlers for certain requests. We showcase the effectiveness of the API in reducing the development time for related applications by developing a prototype web application on top of it.

2.4. Nowcasting procedure

2.4.1. Long - short term memory networks

The LSTM unit consists of a sequence of distinct processes through which data passes to generate a prediction. Fig. 3 depicts a computational graph illustrating the internal architecture of an LSTM unit in means of dataflow. Within the unit, data undergoes filtration across multiple stages, with the flow being regulated by weighted gates. These gates incorporate a sigmoid function that receives a weighted input,

Table 2
API functions for accessing the collected data of the meteorological stations.

Command function	Description
get_data	access to daily data
get_dataset	retrieval of historical data for generating a well-structured dataset in "csv" (Comma Separated Values) format
get_stats	access to daily statistics
get_telemetry	access to telemetry and internal conditions of stations
set_command	ability to transmit commands to specific station units

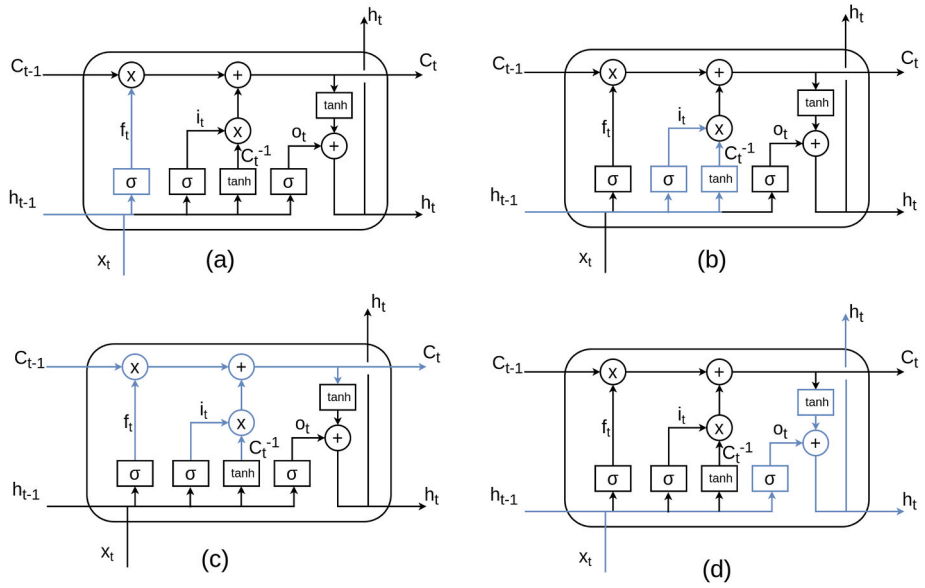


Fig. 3. The LSTM computational graph/unit. 3(a) depicts the forget state, 3(b) depicts the dataflow for the candidate state, 3(c) depicts the process of updating the cell state and 3(d) represents the output from the neuron.

generating an output that represents the quantity of information to be utilized in the subsequent stage (or state).

We provide here a brief description of the functionality of the LSTM cell. Wherein the classic recurrent unit ("Vanilla" RNN) has a single hidden state, LSTM cells have: one for the basic operation and one for extended timesteps. The current state is combined with historical data to form the new state that will be the output. The dataflow inside this unit is controlled by weighted operators, simply referred to as gates. Gates are composed out of a sigmoid layer and a pointwise multiplication operation. The values generated by the gates range from 0 to 1 and refer to the percentage of data passing through, guided to the next layer. Hyperbolic tangent functions are also used to filter furthermore the data at each state.

The cell state (C_t) traverses the entire neuron and it is that part of the unit that acts like a memory by adding or removing information to/from the output as regulated by the pre-referenced gate layers. The output is formed by measuring how much of the historical data and how much of the input data is required taking also into account how strong the correlation between them is. To do that the processing chain can be described by four main procedures/states ("Forget", "Candidate", "Update", and "Output").

The Forget State: Is the first step and refers to the amount of historical information that should be removed from the cell state. The process of "forgetting" information is described by equation (1) and depicted in Fig. 3a.

$$f_t = \sigma\left(w_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{1}$$

Where f_t is the amount of information the cell should "forget", w_f is the set of trainable weights that control the forget gate, b_f the bias added to the gate and x_t and h_{t-1} the input and hidden states of the cell respectively. A sigmoid activation function consists of the gate for this level.

The Candidate State: Is the next state after the forget and determines how much of the input information should be added in the cell state. The LSTM neurons' operation under this condition is described by equation (2) and equation (3).

$$i_t = \sigma(w_i \cdot [h_{t-1}, x_t] + b_i) \tag{2}$$

$$C_t^{-1} = \tanh(w_c \cdot [h_{t-1}, x_t] + b_c) \tag{3}$$

Where the current input x_t is filtered twice producing the input portion i_t at time t and the candidate value C_t^{-1} respectively. The gates are controlled by the weights w_i, w_c respectively and thresholded by the bias values b_i, b_c. Sigmoid activation function is utilized to determine the portion of input information and a tanh activation function for the candidate state gate respectively. The candidate state process is depicted in Fig. 3b.

The update State: operates by combining and updating the above values defined from (1), (2) and (3) of the cell state (i.e. the memory). By multiplying these two parameters the candidate state is produced. The update state process is given by equation (4) and its visual representation of data flow can be seen in Fig. 3c.

$$C_t = f_t \cdot C_{t-1} + i_t \cdot C_t^{-1} \tag{4}$$

Where C_t is the "memory" of the cell, f_t the portion of data that needs to be excluded (forget) by the previous state of the cell C_{t-1}, i_t is the new portion of information that will be included to the current state and C_t^{-1} is the candidate state as defined in the previous step.

The Output state: the output state is formed mainly by the updated version of the cell state which passes through a tanh filter coupled with the respective input and the recurrent value. Since correlation patterns in sequential data are strong, the input itself is utilized to define how strong this correlation is resulting in a distilled output that takes into account not only the historical and current dependencies but also the relationship between them. Furthermore, LSTM is a recurrent neural meaning that the produced output will be fed back in the input line as described in the beginning of this section. Equation (5) and equation (6) describe the procedure of generating the output and Fig. 3d depicts the aforementioned data flow representation.

$$o_t = \sigma(w_o \cdot [h_{t-1}, x_t] + b_o) \tag{5}$$

$$h_t = o_t \cdot \tanh(C_t) \tag{6}$$

Where o_t is the portion of the input that will form the output of the cell, as a result of filtering the pats prediction h_{t-1} coupled with the current input instance x_t through a sigmoid gate controlled by the w_o weights and b_o intercept respectively. The new predicted output h_t is finally a combination of the output of the cell o_t and a filtered portion of the memory.

By employing the set of equations described above, a powerful model (LSTM) for analyzing highly correlated time-series datasets emerges. For

these equations to be capable of modeling a specific problem however the right weights (w_i) and biases (b_i) (where $i = f,i,c,o$) must be chosen.

2.4.2. Implementation procedure

The LSTM neural network was implemented using the "Keras" package in Python (https://keras.io/api/). To train and evaluate the algorithms, we utilized the same dataset, splitting it into 80% for training and 20% for validation respectively. Moreover, additional testing is performed on different datasets taken from different days respectively.

The main training dataset comprises one month (from December 1, 2023 to January 1, 2024) of 5-min recordings, including temperature, humidity, barometric pressure, wind speed, wind direction, and rainfall, along with date and time information for a single station unit. The data were collected from our first monitoring system established in Psachna at the Department of Aerospace Science and Technology. After ensuring the integrity of the data, we generated training and validation sets, focusing solely on the recordings of temperature, humidity, pressure and windspeed.

The selection of a relatively small dataset for training our algorithms is intended and not a coincidence or lack of data samples. We wanted to provide as accurate as possible short range weather forecasts with the lowest resources (both computational and time) possible, also including data samples. Training the algorithms for the purpose of our project is not a one-time procedure but a periodical retaining is required to stay "updated" and continue accurate predictions. The last must be performed on runtime by the application and following our 4-rule development principles set - automation, sustainability, modularity, and simplicity. Season changes are strongly affecting the performance of the forecasting algorithm, where we observed high fluctuations in predicted meteorological variables, which are significantly reduced when the model is retrained on related to that season datasets.

2.4.3. Training procedure

We constructed a many-to-one bi-directional stacked LSTM deep network comprising 4 input neurons, one per input feature followed by two layers of 128 and 64 LSTM cells respectively with the ability of holding 2 h of historical measurements, which is 24 time-steps (of 5 min). The output of the last LSTM hidden layers ends in a fully connected layer of 4 neurons which produces predictions for all four input features in a pre-defined future timestep. Table 3 depicts the architecture of the network constructed with Keras primitives. A dropout layer is also included and activated during training with a dropout rate of 0.2 to avoid overfitting.

In addition to the training dataset creation, another significant method applied to the data samples is normalization. Normalization is critical when dealing with RNN (and deep networks in general). To normalize the data the minimum-maximum scaling approach was utilized along all features of the dataset. The result is that all data is represented in range [0, 1]. Equation (7) describes the normalization method applied to data.

Table 3
The architecture of the LSTM - based network for short range forecasting of temperature, humidity, wind and pressure using 2 h of historical 5-min data.

Model: "Bi-Directional LSTM"		
Layer (type)	Output Shape	Param #
lstm_0 (LSTM)	(None, 4, 128)	90624
lstm_1 (LSTM)	(None, 64)	49408
dropout (Dropout)	(None, 64)	0
dense (Dense)	(None, 4)	260
Total params	: 140,292 (512.76 KB)	
Trainable params	: 140,292 (421.76 KB)	
Non-trainable params	: 0 (0.00 Byte)	

$$\widehat{X} = \frac{x - min\{X\}}{max\{X\} - min\{X\}} \tag{7}$$

Where $\widehat{X}$ is the new transformed value of the vector/matrix $\widehat{X}$, x is the previous non-normalized value of the vector/matrix $\widehat{X}$" and min, max are the minimum and maximum values contained in the initial vector/matrix. After a prediction is made, the result must be converted back into actual values. That is simply done by inverting the (7).

$$\widehat{y} = y \cdot (max\{X\} - min(X)) + min\{X\} \tag{8}$$

Where $\widehat{y}$ is the actual predicted value and y the normalized and scaled down predicted value. After training the model through 50 epochs with a batch size of 64 at a time to furthermore speed-up the process the results of evaluating the model on a non-seen model dataset are depicted in Figs. 6 and 7. Table 4 contains relative statistics.

2.5. User interface

A key objective for this system was to design a simple and user-friendly interface. Along with the dedicated API we implemented and a graphical web interface (GUI).

Specifically, the application was developed with a primary focus on ensuring easy data access and displaying fast real-time weather data and forecasts for both users and our team members. The application features two interfaces: one for authorized users and another for the system administrator and system managers. Authorized users can view real-time daily measurements through graphical representations, access various statistics, and download datasets containing daily data collections for the station of their choice, from available geographic locations. The main panel also provides a short-range prediction extending up to 2 h (Fig. 5b).

On the other hand, administrators have full access to the entire system. They can manage user registration requests by adding or deleting users, command stations by enabling or disabling them, add new station entities on the user view, monitor station conditions through telemetry, and access the entire measurement database.

In designing this web application, we settled to use the Django framework (https://www.djangoproject.com/) which aligns with the system design principles. Django is a robust Python-based framework that facilitates the rapid development of web applications. At this point it is mentioned that the backend system is hosted on a commercial "TopHost" (https://top.host) server, the application is deployed using the "PythonAnywhere" hosting approach which is an online integrated development environment (IDE) and web hosting service (platform as a service) based on the Python programming language.

3. Results

In this section, the results of the system operation are presented, after installing the first meteorological station (monitoring unit) in the specific geographical region of interest, establishing the connections with the backend system and deploying the graphical web application platform. Moreover, in this section the accuracy of the prediction models for both the LSTM-based artificial neural network is evaluated.

3.1. Real time monitoring

One meteorological monitoring unit was initially constructed to develop and test all the system components. We consistently monitored the operational status of the installed station and assessed the accuracy of the sensors. In Fig. 4(a)–(f), we present an example of measurements recorded from December 1, 2023, to December 31, 2023.

Table 4
Evaluation statistics regarding the outcomes of the nowcasting procedure. The size of the evaluation dataset is 9200 pairs of values for each parameter.

Prediction Range	Temperature (°C)		Humidity (%)		Pressure (hPa)		Wind Speed (km/hr)	
	ME	MAE	ME	MAE	ME	MAE	ME	MAE
30 min	−1.09	2.42	−1.17	7.29	−1.84	0.82	0.74	5.57
60 min	−2.1	3.6	1.93	7.44	−0.6	1.22	−1.07	5.26
120 min	−2.74	3.03	3.15	7.32	0.29	0.57	−0.33	5.05
Total Mean	−1.97	3.01	1.3	7.35	−0.71	0.87	−0.22	5.29

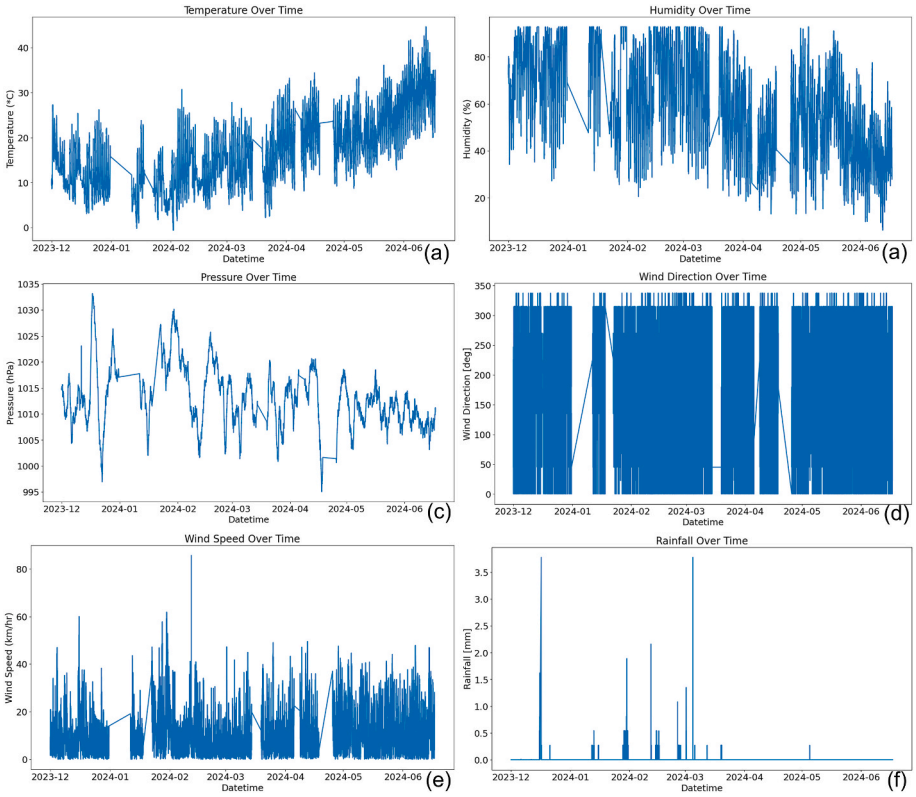


Fig. 4. Measurements for the dates 2023-12-20 to 2023-12-31. Fig. 4(a) depicts temperature, Fig. 4(b) depicts humidity, Fig. 4(c) depicts pressure, Fig. 4(d) depicts wind speed, Fig. 4(e) depicts wind direction and Fig. 4(f) depicts rainfall.

3.2. User graphical interface and data access

The user interface, as depicted in Fig. 5, comprises an interactive map featuring all the deployed meteorological stations and a panel on the right side, providing users access to graphs and statistics for the collected daily data for a selected station. The statistics comprise the

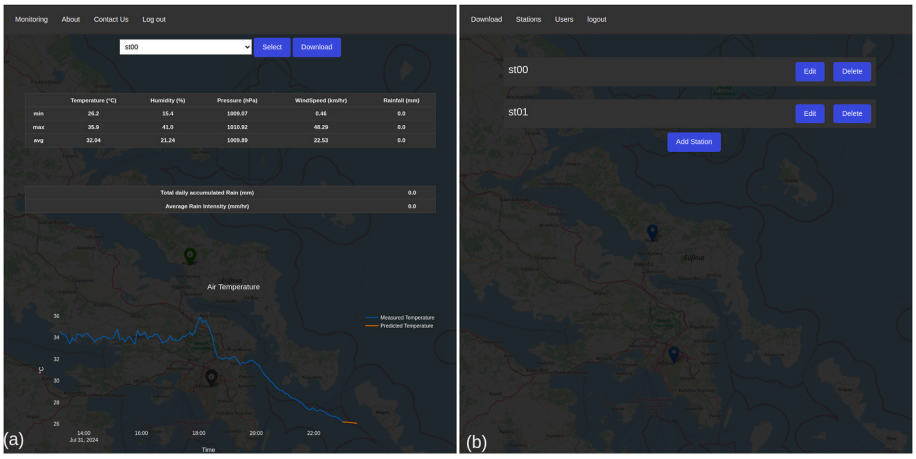


Fig. 5. The main web graphical user interface view. (a) The user-panel presents the statistics in a Table and the real-time graphs of the meteorological variables (b) The primary view of the administrator panel for controlling the station units and registered users.

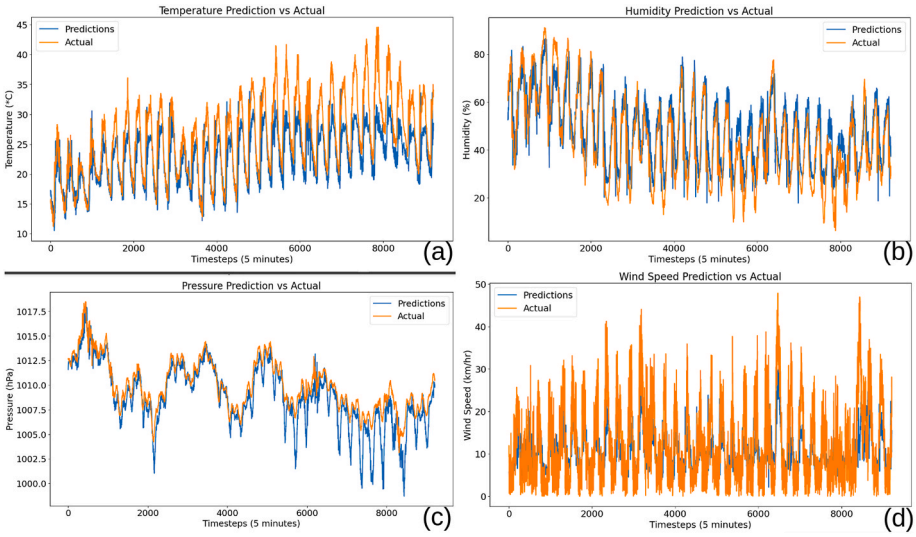


Fig. 6. Prediction results for (from December 1, 2023, to December 31, 2023) 1 h for LSTM compared with the actual measurements for the specific timestamp. Fig. (a) depicts results for temperature, Fig. (b) depicts the results for humidity, Fig. (c) depicts the results for barometric pressure, and Fig. (d) the results for wind speed.

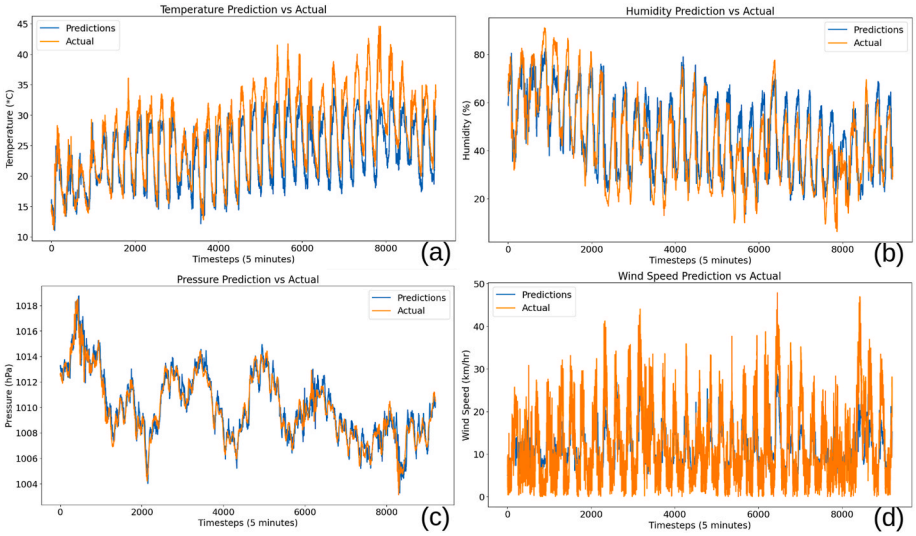


Fig. 7. Prediction results for 2 h for LSTM (from December 1, 2023, to December 31, 2023) compared with the actual measurements for the specific timestamp. Fig. (a) depicts results for temperature, Fig. (b) depicts the results for humidity, Fig. (c) depicts the results for barometric pressure, and Fig. (d) the results for wind speed.

minimum, maximum, and averages of measured parameters such as temperature, humidity, pressure, rainfall, and wind speed for the respective day. Additionally, it displays the total daily accumulated rain and average rain intensity. Plots represent the current day's measured data and the short-range forecast up to 2-h. In the current version, the LSTM-based algorithm that was previously described is employed.

The web application interface can be visited through the link: uoawsn.pythonanywhere.com. Interested users can make a registration request. Interested end users can visit the site through the aforementioned link and using the username "*visitor*" and password "*uoawsn@-freeaccess2023*" can access the interface without registering. In nominal operation however, a user must submit a registration form. Our team will process the request and send a message to the interested user with its credentials for accessing the interface. Upon registration, registered users can have access to daily data measurements, real-time monitoring, short-range forecasts in temperature, humidity, and pressure as well as to daily statistics (Fig. 5a). It is also mentioned that data and availability of the system and its services is freely available through this link: https://github.com/RonT23/WSN.

The administrator's main page, as illustrated in Fig. 5b, grants the administrator the ability to modify characteristics, create a new view and send commands to active station units remotely. It also provides access to the latest telemetry sent by a station for real-time state monitoring. An additional administrator view includes other features, such as the ability to access all data for a given station and a given date range. At this point it is noted that the visualization of real-time data is being made using an interactive, open-source, and browser-based graphing library for Python programming language which is called "Plotly" (https://plotly.com/).

3.3. Short range forecasting

Table 4 illustrates the absolute mean validation error measured for temperature, humidity, and pressure predictions at intervals of 30 min,

1 h and 2 h for both the LSTM and linear regression models. Both models are trained using the same dataset and evaluated on the same validation dataset. The input for both models consists of 2 h of past measurements obtained at 5-min intervals.

$$MAE = \frac{1}{N}\sum_{i=0}^{N-1} |\widehat{y_i} - y_i| \quad (9)$$

$$ME = \frac{1}{N}\sum_{i=0}^{N-1} (\widehat{y_i} - y_i) \quad (10)$$

Where N is the total number of samples for the evaluation, $\widehat{y_i}$ the actual measurement for the respective i-th time instance-sample and y_i the predicted outcome.

Figs. 6 and 7 are graphical examples that illustrate the satisfactory fitting of the timeseries predictions to the relative measurements for the worst cases which are the 60- and 120-min predictions according to Table 4. These Figures and the relative Table show that the Mean Error (ME) regarding air temperature vary from 1 °C to 3 °C (minus symbol depicts underestimation). Humidity has similar mean errors while the barometric pressure and the wind speed is even better. The relative Mean Absolute Errors (MAE) are quite larger than ME due to their "nature. More specifically, this statistical parameter is focusing on the mean divergence of the estimations from the real measurements not considering if there is overestimation or underestimation. At this point it must be referred that the error statistics (Table 4) although are notable, can be characterized as quite satisfactory considering the large dispersion of their distribution of values due to the seasonality of the parameter values. Moreover, these statistics were calculated by using the evaluation datasets with total size of 9200 pairs of values for each parameter. This size is acceptable from a statistical point of view, but larger datasets is intended to improve these evaluation statistics in near future, as the system continues to operate and produce larger timeseries of data.

4. Conclusions

This study presents the outcomes of the development of a fully automated system regarding the real-time recording of basic meteorological parameters and their short-range forecasting (nowcasting). To achieve this, we have developed a series of Commercial Off-The-Shelf (COTS) structures equipped with an array of sensors to gather pertinent variables. One primary key outcome of this study involves the training of an LSTM model scheme to predict future weather behaviors in a short time scale up to 2 h (nowcasting). This model is operational for extrapolating the real-time measurements for each active ground-station of the network. For the selected period of data used to train and evaluate the LSTM, ME of −1.97 °C for the air temperature, 1.3% for the humidity, −0.71 hPa for the barometric air pressure and −0.22 km/h for the wind speed was achieved regarding the total mean values of all the forecasting cycles (Table 4).

Another key outcome is the development of a fundamental API that facilitates the creation of user-level applications atop our systems. To ensure accessibility and user-friendliness, a modern relative framework ("Django") for the development of this application, was utilized. The whole application was developed with a central aim to provide a graphical user interface aligning with the four mentioned design principles: automation, sustainability, modularity, and simplicity.

A third key element consists of the visualization of real-time data. As significant as numerical data are, a concise representation of measurements is of great importance to understand the underlying information. Graphs and plots clearly depict the patterns present in meteorological data. Therefore, all collected samples (in 5 min interval) for all the basic meteorological variables are plotted in a concise way using a standard tool in front-end graphical user interfaces design ("Plotly").

While this study can be considered a comprehensive undertaking, an ongoing refinement and expansion has already been planned. Some of the future plans encompass the evaluation of the LSTM by using larger datasets, and the integration with satellite datasets in order to further improve and extend in-time the short-range forecasts. Also, new meteorological units (stations) have already been planned to be connected. Finally, an upgrade of the sensors is also planned, replacing the existing with new ones of ultrasonic technology, which will improve the accuracy of the measurements enhancing the usefulness of the whole system.

Software and data availability

Name of the software: WSN (Weather Station Networks) Project.
Developer: Tesla R., Maladaki R. S.
Contact information: ron-tsela@di.uoa.gr, maladakistella@gmail.com, skolios@aerospace.uoa.gr.
Year first available: 2024.
Program languages: Python, PHP, C, HTML, CSS, JavaScript;
Cost: free;
Software requirements: Python3;
Hardware requirements: Linux Native Machines/Arduino Mega and sensors;
Software availability: https://github.com/RonT23/WSN.

Data availability

Daily data collection from https://uoawsn.pythonanywhere.com/

CRediT authorship contribution statement

R. Tsela: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Formal analysis, Data curation. **S. Maladaki:** Visualization, Validation, Software, Methodology, Formal analysis, Data curation. **S. Kolios:** Writing – review & editing, Writing – original draft, Validation, Supervision, Resources, Project administration, Investigation, Conceptualization.

Declaration of competing interest

There is no conflict of interests between the authors.

Data availability

I have shared code/data in the link it is referred in the manuscript: https://github.com/RonT23/WSN.

References

Abhishek, K., Singh, M.P., Ghosh, S., Abhishek, A., 2012. Weather forecasting model using artificial neural network. Procedia Tech. 2, 311–318.
Aggarwal, R., Kumar, R., 2013. A comprehensive review in weather prediction models. International J. Comput. Appl. 74 (18), 44–48.
Almonacid, F., Perez-Higueras, P., Rodrigo, P., Hontoria, L., 2013. Generation of ambient temperature hourly time series for some Spanish locations by artificial neural networks. Renew. Energy 51, 285–291.
Brendel, E.C., Dymond, L.R., Aguilar, F.M., 2019. An interactive web app for retrieval, visualization, and analysis of hydrologic and meteorological time series data. Environ. Model. Software 117, 14–28. https://doi.org/10.1016/j.envsoft.2019.03.003.
Bushara, N.O., Abraham, A., 2013. Computational intelligence in weather forecasting: a review. J. Net. Innov. Comput. 1, 320–331.
Chattopadhyay, S., Jhajharia, D., Chattopadhyay, G., 2011. Univariate modelling of monthly maximum temperature time series over northeast India: neural network versus Yule-Walker equation-based approach. Meteorol. Appl. 18, 70–82.
Chen, X., Zhao, J., He, M., 2021. Ensemble learning of numerical weather prediction for improved wind ramp forecasting. In: IEEE Green Technologies Conference. https://doi.org/10.1109/GreenTech48523.2021.00031. Denver, CO, USA, 07-09 April 2021.
Ghafarian, F., Wieland, R., Luttschwager, D., Nendel, C., 2022. Application of extreme gradient boosting and Shapley Additive explanations to predict temperature regimes

inside forests from standard open-field meteorological data. Environ. Model. Software 156, 105466. https://doi.org/10.1016/j.envsoft.2022.105466.
Goubanova, K., Li, L., 2007. Extremes in temperature and precipitation around the Mediterranean basin in an ensemble of future climate scenario simulations. Global Planet. Change 57, 27–42.
Guo, Z., Chi, D., Wu, J., Zhang, W., 2014. A new wind speed forecasting strategy based on the chaotic time series modelling technique and the Apriori algorithm. Energy Convers. Manag. 84, 140–151.
Hoang, T., Yang, L., Cuong, D.P., Trung, P.D., Tu, N.H., Truong, V., Hien, T., Nha, T., 2020. Weather prediction based on the LSTM model implemented AWS Machine Learning Platform. Int. J. Res. Appl. Sci. https://doi.org/10.22214/ijraset.2020.5046. Engineer. Tech. ISSN: 2321-9653; Volume 8 Issue V.
Hochreiter, S., Schmidhuber, J., 1997. Long shot-term memory. Neural Comput. 9 (8), 1735–1780. https://www.bioinf.jku.at/publications/older/2604.pdf.
Kalimeris, A., Founda, D., Giannakopoulos, C., Pierros, F., 2011. Long term precipitation variability in the Ionian islands (Central Mediterranean): climatic signal analysis and future projections. Theor. Appl. Climatol. 109, 51–72.
Karagiannidis, A.F., Karacostas, T., Maheras, P., Makrogiannis, T., 2012. Climatological aspects of extreme precipitation in Europe, related to mid-latitude cyclonic systems. Theor. Appl. Climatol. 107, 165–174.
Karvelis, P., Georgoulas, G., Kolios, S., Stylios, C., 2017. Ensemble learning for forecasting main meteorological parameters. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC) Banff Center. Banff, Canada, October 5-8, 2017, 978-1-5386-1645-1/17.
Kotsias, G., Lolis, J.C., Hatzianastassiou, N., Lionell, P., Bartzokas, A., 2022. A comparison of different approaches for the definition of seasons in the Mediterranean region. Int. J. Climatol. 42 (3), 1954–197415.
Naik, A.R., Pathan, S.K., 2012. Weather classification and forecasting using back propagation feed-forward neural network. Int. J. Sci. Res. 2 (12), 2250–3153.
Nandi, S., Patel, P., Swain, S., 2024. IMDLIB: an open-source library for retrieval, processing and spatiotemporal exploratory assessments of gridded meteorological observation datasets over India. Environ. Model. Software 171, 105869. https://doi.org/10.1016/j.envsoft.2023.105869.
Olaiya, F., Adeyemo, A.B., 2012. Application of data mining techniques in weather prediction and climate change studies. Int. J. Inform. Engineer. Elect. Business 1, 51–59.
Pierre, A., Monbet, V., 2012. Markov-switching autoregressive models for wind time series. Environ. Model. Software 30, 92–101.
Pourmousavi, Kani, S.A., Ardehali, M.M., 2011. Very short-term wind speed prediction: a new artificial neural network-Markov chain model. Energy Convers. Manag. 52 (1), 738–745.
Saima, H., Jaafar, J., Belhaouari, S., Perak, T., Jillani, T.A., 2011. Intelligent methods for weather forecasting: a review. In: Proceedings of the National Postgraduate Conference, (Tronoh Perak, Malaysia) Sept, pp. 19–20, 2011.
Salman, A.G., Heryadi, Y., Abdurahman, E., Supatra, W., 2018. Weather forecasting using merged long short-term memory model (LSTM) and autoregressive integrated moving average (ARIMA) model. J. Comput. Sci. 14 (7) https://doi.org/10.3844/jcssp.2018.930.938, 2018.
Tekin, F.S., Fazla, A., Kozat, S.S., 2023. Numerical weather forecasting using convolutional-LSTM with attention and context matcher mechanisms. IEEE Trans. Geosci. Rem. Sens. https://arXiv:2102.00696v2.
Varfi, M.S., Karacostas, T.S., Makrogiannis, T.J., Flocas, A.A., 2009. Characteristics of the extreme warm and cold days over Greece. Adv. Geosci. 20, 45–50.
Wu, H., Levinson, D., 2021. The ensemble approach to forecasting: a review and synthesis. Transport. Rsearch Part C: Emerging Tech. 132, 103357 https://doi.org/10.1016/j.trc.2021.103357.
Zenkner, G., Navarro-Martinez, S., 2023. A flexible and lightweight deep learning weather forecasting model. Appl. Intell. 53, 24991–25002. https://doi.org/10.1007/s10489-023-04824-w.