e-Science workflow: A semantic approach for airborne pollen prediction

Sandro Hurtado[a], María Luisa Antequera-Gómez[a], Cristóbal Barba-González[a,*], Antonio Picornell[b], Ismael Navas-Delgado[a]

[a]*Dept. de Lenguajes y Ciencias de la Computación, ITIS Software, Universidad de Málaga, Málaga - 29071, Spain,* cbarba@uma.es

[b]*Dept. de Botánica y Fisiología Vegetal, Universidad de Málaga, Málaga - 29071, Spain*

Abstract

Allergic rhinitis has become a global health problem in recent decades because airborne pollen is a primary trigger of this respiratory disorder. Moreover, pollinosis can exacerbate the symptoms of asthma and favour respiratory infections. Seasonal pollen trends and climatic circumstances (such as temperature, precipitation, relative humidity, wind speed and direction, and other variables) can impact daily airborne pollen concentrations, influencing local pollen emission and dispersion. Because of that, pollen monitoring and prediction are becoming more relevant to the urban population and scientific interest is put into them. Due to such tasks' high volume of data, scientists are starting to use computational tools like workflows to automate and speed up the process. Furthermore, using the expert scientific domain is critical for improving the analysis, allowing, among others, a better workflow configuration and data provenance. As semantic web technologies have been revealed as an essential means for knowledge representation, we implemented this workflow information as an ontology using formats like RDF(S) and OWL. Consequently, this paper provides a semantic-enhanced e-Science workflow based on the TITAN framework for pollen forecasting analysis using meteorological data. Furthermore, a catalogue of components is developed on the TITAN framework, which allows the creation of different workflow versions. Two case studies of pollen prediction were developed to test the implementation of the aforementioned methodologies. Both were elaborated with airborne pollen data obtained in the city of Málaga (Spain). Still, one was elaborated for *Platanus* pollen type (narrow annual main pollination period), while the other was done for Amaranthaceae pollen type (extensive annual main pollination period). The predictions have been conducted using machine and deep learning algorithms like SARIMA or CNN-LSTM that intend to optimise the pollen prediction procedure depending on its stational and seasonal profile.

Keywords: Big Data Analytics, Semantics, e-Science, Pollen prediction

1. Introduction

How science and engineering performed their experiments has changed dramatically over the last decades. Scientific workflow technology has been developed to automate computational operations in the scientific domain (e-Science). E-Science refers to the increasingly global collaboration of people and shared resources required to solve new science and engineering problems. These e-Science problems range from the simulation of entire engineering or biological systems to bioinformatics, proteomics, and pharmacogenetics research [1]. A scientific workflow is, at its core, a technology that automates the execution of an experiment [2]. As a result, it can provide many benefits throughout an experiment. During the composition phase, the scientists have access to a collection of tried-and-true workflows from which to choose. Because experimenting is a repeating activity, workflows can relieve scientists of these repetitive chores while keeping track of all intermediate steps and data during the execution phase as (meta)-data provenance [3]. These traces can be used later to ensure that the experiment is reproducible.

During the analysis phase, provenance information can be used to examine the evolution of the research effort, determine the source of an error, or go back to a prior stage and change the investigation's path. In this sense, semantic web technologies have become especially interesting for data traceability and provenance. For instance, the provenance ontology (Prov-o) [4] defines the classes and properties necessary for this goal. Besides, this phase also includes visualisation tools to aid in evaluating the results [5].

Thus, a typical workflow scenario in e-Science applications is a data analysis procedure that includes preprocessing, analysis, simulation, and post-processing processes [6]. Due to the wide range of studies performed by scientists and the diverse requirements associated with their automation, scientific workflow systems are forced to address an enormous variety of complex issues, such as sequence alignment [7], agricultural experiments [8], ex-

*Corresponding author.

perimental images [9], clinical data analysis [10], etc.

In the present work, we adopt a semantic approach for the e-Science workflow that solves some research challenges such as: 1) enhancing data and analysis results quality, validating the analysis processes (during design, configuration, and execution phases); 2) traceability of the analysis results, repeatability of the workflow, and the ability to generate not only an explanation of results but also a "confidence level" which depends on, among others, data provenance and domain semantics. We employ TITAN [11], a general-purpose software platform for managing the entire life cycle of research workflows in the context of Big Data and scientific applications, from deployment through execution. This Workflow Management System (WMS) is distinguished by a semantic-driven design and operation paradigm at many levels: data sources, issue domain, and workflow components. Furthermore, this ontological meta-data framework manages processes and models consistently while leveraging domain knowledge [11]. BIGOWL [12] is the ontology behind TITAN, which defines all the relevant (meta)-information to support the analysis, data provenance and help in result interpretation. This will facilitate the integration (in a standardised way) with third-party data, algorithms, business intelligence (BI) and visualisation services. Besides, TITAN allows the development of the components using any programming language. This is possible because those components will be wrapped in docker containers [13], which TITAN executes as described in [11].

Allergic rhinitis has become a global health problem in recent decades, being airborne pollen one of the primary triggers of this respiratory disorder [14]. Moreover, pollinosis can exacerbate the symptoms of asthma and favour respiratory infections [15].

Previous studies suggest that the pollen release rate is influenced by seasonal pollen trends and meteorological conditions like temperature, relative humidity, precipitation, and wind speed [16, 17]. On the other hand, airborne pollen concentrations show different trends depending on the plant species and local impacts of climate change. In fact, in the Iberian Peninsula, different trends were detected for arboreal (e.g., Cupressaceae family, *Platanus*, *Quercus* and *Olea*) and herbaceous (e.g., Urticaceae, Poaceae and Amaranthaceae families) pollen types. Different trends were seen for the same pollen type depending on the sampling location [18]. Therefore, creating a workflow instead of performing individual and separate studies of pollen forecasting would be of great interest in reducing the scientific effort and optimising the process. We have developed an e-Science workflow in a real case of use. We forecast the trend and the likelihood of detecting daily high levels of pollen concentrations in the atmosphere, taking into account data from the amount of grain of pollen in the air as well as meteorological data such as precipitation, wind speed and direction, or temperature [19].

Scientists are starting to leverage workflows to forecast pollen concentration [20, 21, 22]. However, as far as our knowledge goes, those works do not consider relevant aspects such as domain context, data provenance or reuse of the workflow components. For those reasons, we focus on developing a catalogue of components in TITAN for designing e-Science workflows to forecast airborne pollen, considering the seasonal pollen trends and meteorological variables.

The challenge in this work is the development of mechanisms that facilitate the pollen prediction to domain experts and, in addition, ease the application and parameter configuration of different algorithms without knowing anything about their parameterisation or their development. The main contributions of this work are described next:

- Development of an ontology which extends BIGOWL [12] to represent knowledge on atmospheric pollen concentration and its prediction. This extension provides annotations of the machine and deep learning algorithms, data management algorithms and data provenance algorithms, together with their parameters. This includes providing validation mechanisms to ensure the quality of workflows from the design stages, before their execution, by taking advantage of the operational semantics of BIGOWL.

- Development of a catalogue of open source components to design e-Science workflows to support the analysis of pollen forecasting in TITAN [11]. The catalogue contains several components for supporting different data formats (CSV, JSON, XLSX), data storage (file, database), gap detection in the data (NaN, missing data), discovery seasonality and stationarity in the data, machine and deep learning algorithms (LSTM, Random Forest, CNN, SVM), data interpolation, and data visualisation.

- Development of a semantic-enhanced e-Science workflow based on TITAN for allowing repeating the analysis and the contextualisation of the data. Besides, keeping a chain of data provenance for the data themselves and for the workflow parameters, hyperparameters and results.

- As a proof of concept, we show how the workflow designed with the catalogue of components is used to predict pollen in real-world data. Users benefit from tailored, data-driven guidance. PPWO guides users in designing the workflow specifically according to the pollen data's characteristics in the time series. This semantic guidance ensures that users choose appropriate techniques precisely suited to handle these patterns, enhancing the accuracy and relevance of predictions. A first case study consists of the prediction of *Platanus* pollen, whose taxon is often planted in many cities for shade as ornamentals. This pollen type is widely distributed in southern Europe and

substantially contributes to pollinosis symptoms occurring between February and April [23]. The second case of study is based on the Amaranthaceae pollen type. Many of the species belonging to Amaranthaceae are ubiquitous and cosmopolitan. . Their flowering season does not overlap in all species, which causes a long pollen season with several peaks within the same natural year [17]. This pollen type is present in the atmosphere of many European cities and causes allergic rhinitis and conjunctivitis, and it exacerbates symptoms in the asthmatic population [24, 25].

The rest of the paper is structured as follows. In Section 2, background concepts and literature overview are presented. Section 3 describes the semantic-enhanced workflow for e-Science, focusing on the semantic design and the steps of the workflow. Section 4 depicts the two use cases for validation, which is followed by discussions in Section 5. Finally, conclusions and future work are drawn in Section 6.

2. Background and Related Work

This section includes background concepts in time series prediction and the Semantic Web field. Furthermore, it is provided with an overview of the main related works.

2.1. Background concepts

In line with [26], a formal representation of the real world is provided by an **ontology**. It defines a description of terms in a concrete domain (classes or concepts), properties of each term (descriptions of various aspects and attributes of the concept), and property limitations. An ontology accompanied by a set of individual instances of classes composes a knowledge base and offers services to facilitate interoperability across multiple heterogeneous systems and databases.

The W3C proposal that provides a vocabulary for defining online resources includes the Resource Description Framework (**RDF**) [27]. RDF specifies resources as triples, consisting of a subject, a predicate, and an object. The Ontology Web Language (**OWL**) [28] defines ontologies in the context of the Semantic Web, and was approved by the World Wide Web Consortium (W3C) in 2004. In 2012, appears **OWL2** which is an extension of OWL but with much more expressiveness.

SPARQL is a query language for interacting with RDF data. Furthermore, SPARQL is the query language advocated by W3C to work with RDF graphs [29], allowing queries and web data sources to be identified using URIs.

2.2. BIGOWL

The semantics in the TITAN framework is defined by employing BIGOWL, an OWL2 ontology that defines a language for describing workflows and entities that incorporate software components. Tasks represent how these components are instantiated (parameter setup) and performed. Some components, such as algorithms, are processing (transformation) components. This ontology has five primary classes with the goal of describing components:

- *Data Collection*, which annotates the initial data to serve as the input for the following components (e.g., metadata about a CSV file or database credentials);
- *Data Processing*, which defines the behaviour of the processing stage (e.g., clean dataset, data normalisation, call an external API, etc.);
- *Data Analysis*, which represents all analytic operations (e.g., CNN, SVN, LSTM, etc.);
- *Data Flow*, which annotates the process of selecting between various data for being used in the workflow (e.g., selection between different datasets following quality criteria like the number of empty cells or the number of rows);
- *Data Sink*, whose instances describe the process of persisting data in databases or visualisation.

It should be pointed out that the main classes in BIGOWL are *Data*, *Component*, *Task*, *Parameter* and *Workflow*. They are defined as follows:

- ***Data***. It is devoted to annotating all the data flowing throughout the analytic workflows.
- ***Component***. This class represents each processing stage in the analytic workflow. It encapsulates a specific functionality, its parameters, and the inputs and outputs it considers.
- ***Task***. A task is a single instance of a component that can run in a workflow. It is defined by parameters, inputs and outputs with concrete values.
- ***Workflow***. This class is used to guide the correct orchestration of those tasks involved in data analysis workflows.
- ***Parameter***. This class is used to annotate any values that must be passed to a component and, it is instantiated with concrete values.

2.3. Workflows in e-Science

Modelling and design support in workflows in e-Science is provided through scientific workflow systems: clarity, predictability, recordability, reportability, reusability, scientific data modelling, and automatic optimisation are all factors to consider [30].

There exist plural e-Science workflows and, in general, they are described, implemented and executed using Workflows Management Systems (WMS), such as Taverna [31]. Taverna is an open-source workflow tool suite with a wide

user base in the e-Science field that mixes distributed web services and local tools into complicated research pipelines. In [32], Taverna is leveraged for e-Science in the following ways: personal reuse, reuse by collaborators and reuse by third parties whom the workflow author never met. Multiple e-Science workflows are defined with Taverna [1] in domains such as protein sequence analysis, gene expression profile, and sequence alignment.

Another WMS widely employed in e-Science is VLAM-G [33], an e-Science-focused decentralised data flow-driven workflow engine. VLAM-G is built on various essential Grid services (resource management, data access, and resource information services). For example, in [33], the workflows focus on analysing gene expression data and local time series temperatures in Amsterdam from 1995 to 2006.

Scientific procedures today use a variety of grid and cloud interfaces to access distributed heterogeneous resources, which can be challenging to program. Furthermore, consistent quality of service across these distant resources is required, especially for time-sensitive critical applications. The virtual grid execution system (vgES) developed by VGrADS [34] provides a consistent qualitative resource abstraction across grid and cloud platforms, for assessing VGrADS, the authors conduct an analysis of regional weather forecast but with small datasets.

Because many data-driven applications require analysis of scientific data received from several sources and generated by computations on distributed resources, data provenance is critical in e-Science [35], covering the data and process provenance needs and difficulties for workflows, as well as how generic provenance capture can be enabled in Kepler's actor-oriented workflow environment.

Some works are focused on analysing time series data, for instance, in [36] the authors define a set of cooperating web services as a workflow that analyses time series data to detect real-time storms and activate weather forecasts through data mining and events processing.

2.4. Pollen prediction

In the field of pollen, forecasting has employed different statistical methods. For instance, in [37] the models are constructed using the partial least squares regression method (PLSr). It takes crop production as a dependent variable and meteorological and aerobiological data as the independent variables. Besides, in [38], the models are built by linear regression of daily pollen concentration. However, in recent years, scientists have employed machine learning methods such as neural nets and random forests because scientists are not experts in computational data analysis [39].

Deep learning models have recently been used for pollen prediction analysis [40]. Deep learning-based models have been effectively implemented in various sectors relevant to time series prediction, including remote sensing [41], and others. However, these approaches are not provided as easy-to-use tools for scientists.

Time-series analytic techniques may be used to analyse aerobiological datasets, and various approaches have been devised to determine the seasonal component [42]. The study of time series forecast begins with a regression equation [43]. Nonetheless, it is challenging to achieve high-precision prediction through complex models due to the high complexity, irregularity, randomness and nonlinearity of actual data. Machine learning methods [44] can obtain more accurate forecasts than traditional statistical-based models through repeated training cycles and learning approximations. In [45], time series prediction is performed by Support Vector Machine (SVM), Elman recurrent neural networks, and AutoRegressive Moving Average (ARMA) models. In this work, SVM demonstrates to be an excellent candidate for the prediction of time series. However, SVM needs previously determined parameters and constant values for appropriate performance [46]. ARIMA (AutoRegressive Integrated Moving Average) models were used in ARMA models to forecast pollen concentrations. However, SARIMA (Seasonal AutoRegressive Integrated Moving Average) is a modification of the ARIMA model that explicitly supports seasonal invariable time series data.

Additionally, to determine if a time-series is stationary, the Dickey-Fuller test is commonly used [47]. Time series prediction with Deep Learning methods such as Recurrent Neural Networks (RRNs), especially Long Short-Term Memory Neural Networks (LSTM), has significant achievements in recent years [48]. Even though LSTM can help capture long-term dependencies, its ability to pay different degrees of attention to sub-window features within multiple time steps is insufficient. An evolutionary attention-based LSTM training with competitive random search is proposed in [49] for multivariate time series prediction to address this issue. Besides, that paper uses an inductive and transductive LSTM to create a data-driven forecasting model for a weather forecasting application [50].LSTM offers the most accurate one-day forecast and essential information over time to present a good candidate when determining the forecast [51]. Thus, LSTM bypasses extensive preprocessing to uncover influential features by allowing the network to extract them regardless of the different variables [52].

3. e-Science pollen prediction workflow in TITAN

This section describes the development of e-Science pollen prediction workflows in TITAN. This development is focused on producing a concrete forecasting workflow and the semantic annotation of the data managed in this workflow.

[1] `https://www.myexperiment.org/workflows`

3.1. Semantic design

To exploit all the semantic capacity of TITAN, we have developed an ontology, Pollen Prediction Workflow Ontology (PPWO), which joins pollen domain ontologies with BIGOWL for annotating from the input data to the last results, including workflow, components and their parameters.

Several competence questions were examined for designing the domain ontology in order to follow the best practices and principles in ontological engineering [53]. Without loss of generality, these questions are formulated in the context of the pollen prediction domain of knowledge as a proof of concept. However, it could be conducted on other different domains (earth and environment, plants, organisms, etc.). The questions are:

- What is the set of characterising classes for the pollen prediction domain?
- What is the set of object properties for the pollen prediction?
- What is the set of data properties for the pollen prediction?
- Are there ontologies which PPWO can be aligned with?
- How do we align the pollen prediction domain and BIGOWL?

As a result, Figure 1 depicts the PPWO, where all the ontologies aligned for developing it, reusing existing knowledge in the form of public ontologies. Thus, several classes have been selected from a set of other ontologies to reuse existing knowledge:

- Big Data analytics OWL ontology (BIGOWL) [12]. All the classes and properties related to the analysis have been selected for this work, such as Workflow, Task, Component, Data, Parameter, Column and Data Type.
- Semantic Web for Earth and Environment Terminology Ontology (SWEET) [54]. We have reused terminologies from this ontology regarding weather and climate data such as Atmospheric Pressure, Temperature, Wind, Insolation or Precipitation.
- Plant Ontology [55]. Plant ontology vocabulary is used to describe botanical terms. In this case, PPWO reuses the Pollen concept.
- National Center for Biotechnology Information Organismal Classification ontology (NCBITAXON) [56]. This ontology provides a curated classification and nomenclature for organisms. The organisms chosen for the study from this ontology are Amaranthaceae and *Platanus*, and their common class Mesangiospermae.

Considering this, we developed our ontology using the standard Ontology 101 development process [53] in an interdisciplinary manner involving ontology and biology specialists. There are seven steps in this procedure:

1. *To determine the domain and scope of the ontology.* The scope is to define an ontology that describes, in this case, the pollen prediction domain. It will let us annotate any analysis related to this domain. In particular, it focuses on Amaranthaceae and *Platanus* pollen types, together with meteorological data such as *Temperature*, *Insolation*, *Wind*, *Atmospheric Pressure* and *Precipitation*.
2. *To consider reusing existing ontologies.* The ontology is aligned with BIGOWL and a set of biological and meteorological ontologies. On the one hand, BIGOWL has been endorsed to describe the concepts related to the workflow (task, component, data and parameters). On the other hand, the set of the aligned ontologies defines, among other ideas, key terms in a pollen prediction analysis such as *Temperature*, *Atmospheric Pressure*, *Pollen*, etc.
3. *To enumerate important terms in the ontology.* Important terms were selected from the literature: *Task*, *Component*, *Parameter*, *Column*, *Row* and *Cell* in the *Tabular Dataset*, amount of *Pollen* collected from different plant organisms like *Platanus* or Amaranthaceae.
4. *To define the classes and the class hierarchy.* The key concepts described above are specified in our ontology, which includes BIGOWL and other biological and meteorological ontologies.
5. *To define the properties of classes and slots.* We have considered the BIGOWL data and object properties to define relationships between classes and attributes in the workflow. Nevertheless, we have defined new data and object properties to connect the different classes aligned in PPWO ontology, for example, to indicate whether the pollen data are stationary or seasonally or to define the pollen of a plant or the different representation of the domain data in the analysis. Table 1 contains the object and data properties defined for PPWO ontology.
6. *To define the facets of the slots.* As this step includes the definition of value type restrictions for the ontology's properties, we have defined one among others to indicate that the temperature can only be the double type.
7. *To create instances.* To describe the domain of the analysis, the ontology contains some instances such as data recollected from *Platanus* or Amaranthaceae pollen, temperature, precipitation, etc.

The e-Science workflow was designed using PPWO, describing fifteen steps for determining the pollen prediction workflow. A diagram of the pollen forecasting generic process is depicted in Figure 2. It is divided into five sections.

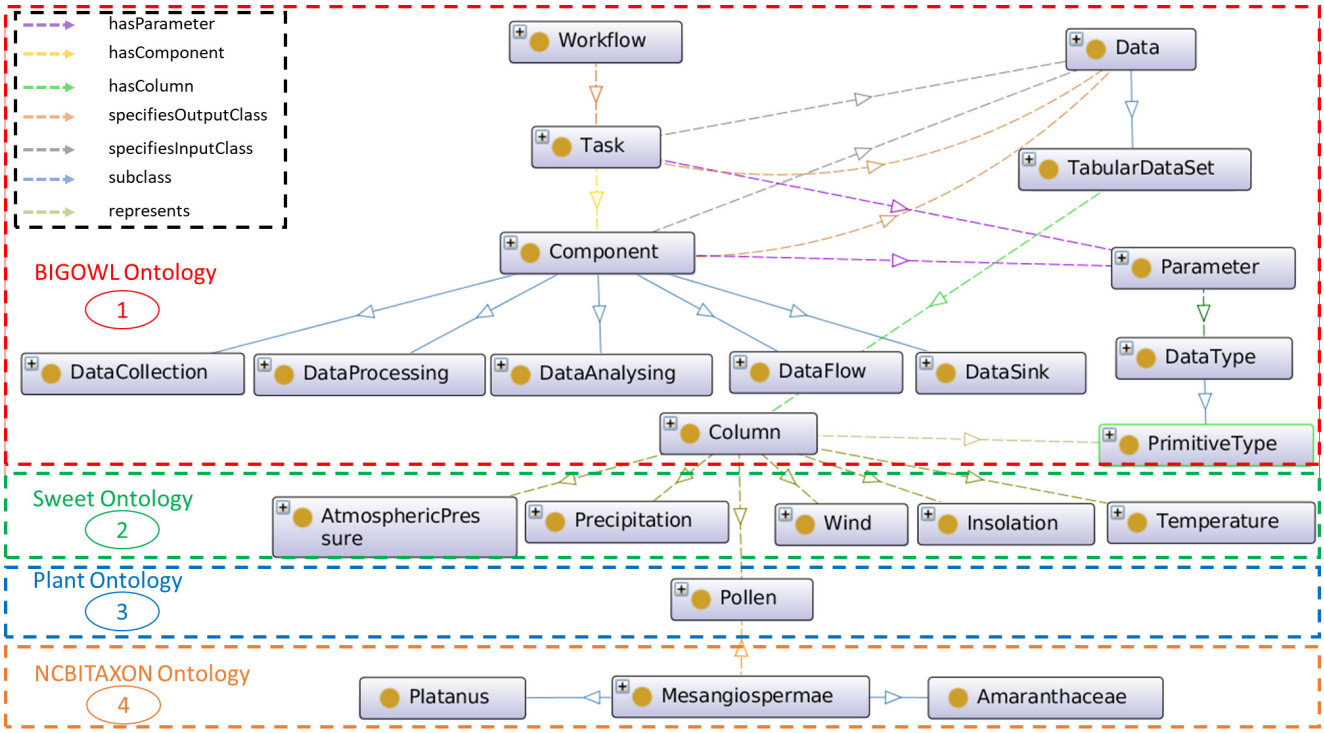

Figure 1: Overview of the Pollen Prediction Workflow Ontology. Subclasses are shown by straight arrows, while dotted arrows denote characteristics. The pollen prediction workflow ontology is aligned with four ontologies: BIGOWL (1), SWEET (2), Plant (3) and NCBITAXON ontologies (4).

Table 1: PPWO ontology: object and data properties

Object Properties	Description Logic
hasPollen	$\exists$ hasPollen Thing $\sqsubseteq$ 71274 $\top \sqsubseteq \forall$ hasPollen PO_0025281
hasPrecipitation	$\exists$ hasPrecipitation Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasPrecipitation Precipitation
hasInsolation	$\exists$ hasInsolation Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasInsolation Insolation
hasAtmosphericPressure	$\exists$ hasAtmosphericPressure Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasAtmosphericPressure AtmosphericPressure
hasPlant	$\exists$ hasPlant Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasPlant PO_0025281
hasTemperature	$\exists$ hasTemperature Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasTemperature Temperature
hasWind	$\exists$ hasWind Thing $\sqsubseteq$ Column $\top \sqsubseteq \forall$ hasWind Wind
Data Properties	**Description Logic**
seasonaly	$\exists$ seasonaly $\sqsubseteq$ PO_0025281 $\top \sqsubseteq \forall$ seasonaly
stationary	$\exists$ stationary $\sqsubseteq$ PO_0025281 $\top \sqsubseteq \forall$ stationary
represents	$\top \sqsubseteq \forall$ represents string

Firstly, the workflow receives and combines the data from different sources (Figure 2, blue). Secondly, this data is homogenised and filled with selected data from a database of interest (Figure 2, orange). The missing data are interpolated, and the optimal final dataset is selected according to the best metrics of a Random Forest analysis (Figure 2, green). The interpolated data are then transmitted to the prediction phase (Figure 2, pink). Finally, the workflow ends with visualising the pollen prediction results (Figure 2, grey).

TITAN has been extended to support the automatic annotation of data, tasks, and their parameters during the execution of a workflow using the PPWO. In BIGOWL and also in PPWO, a task is a single instance of a component that can be run in a workflow. Therefore, a component is a template for one or more tasks in a workflow that will carry out its specific functionality.

Concerning the data, different types of data are defined, such as XLSX, CSV, JSON or PNG files. In the case of CSV files, their characteristics have also been described like delimiter, encoding, etc.

It is worth mentioning that the workflow's components can be interchangeable, or even it is possible to remove some of them or include new ones. TITAN has been updated, using the classes and properties of BIGOWL included in PPWO, with an option for assessing the components' compatibility during the design stage. This is implemented as a SPARQL query on the semantic repository. Listing 1 depicts the SPARQL query for checking the compatibility between components and tasks. To do that, first, it is verified that the input data model is consistent with the task, which is a component instance (or implementation). Second, it is checked that the input/output data connections of each pair of subsequent tasks are "semantically" comparable, that is, the input data of `task2` is covered by the output data of `task1`, depending on the

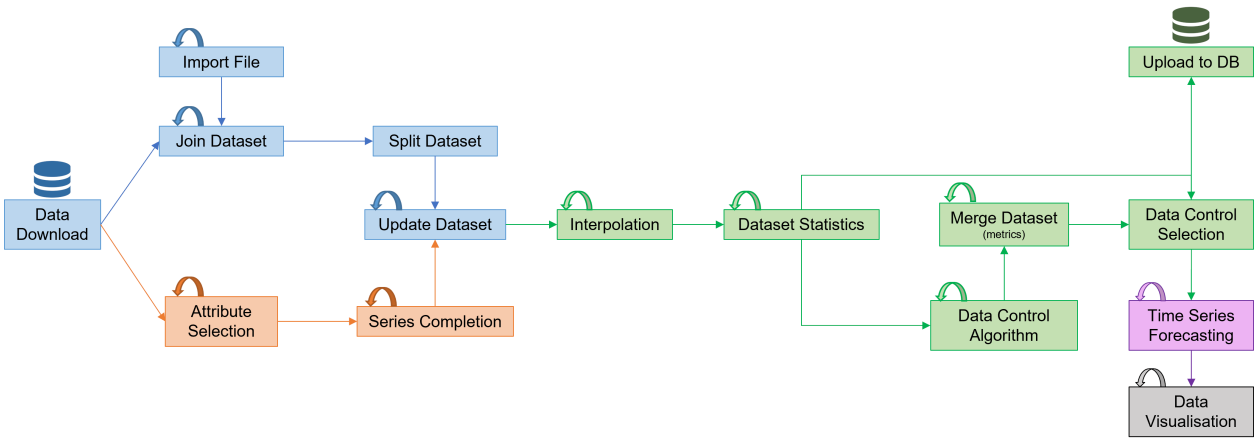

Figure 2: Snapshot of the pollen forecasting generic workflow. It is split into five sections. The blue steps receive data from various sources, such as databases and Excel files, merging them into a CSV file followed by its sectioning into a more straightforward version. The orange steps select data of interest previously obtained from a database, detect data gaps, and fill them using Linear Regression techniques. The data from the blue and orange paths are combined. The stages in green are the ones that interpolate missing data using different techniques, such as linear, spline, or average, and then determine which interpolation is optimal based on R^2, Mean Absolute Error (MAE), and Root-Mean-Square Error (RMSE) metrics. The interpolated data are then transmitted to the prediction phase. Deep learning and machine learning methods are used to anticipate the pollen counts (pink). Finally, charts are used to display the results (grey). The top arrows pointing to themselves indicate that the corresponding step could be repeated in parallel branches of the workflow if necessary; for example, the workflow can have more than one input source or the analyst could utilise more than one interpolation method like linear, time, quadratic, spline, pad, etc. Or even different algorithms for times series forecasting like LSTMs, SARIMA, or any other that can be included in the open-source components catalogue.

workflow's execution sequence.

3.2. Semantic management of data provenance and data lineage

The annotations with PPWO let TITAN enhance the workflows with semantic knowledge as it explicitly describes and registers the data lineage (data provenance in database systems) from sources to results. Furthermore, all the workflow tasks and their parameters are annotated with the semantics. This work exploits those annotations with twofold goals. First, to provide the analyst with all the parameters and their values from already designed and executed workflows for replicating the experiments. Second, the management of the data lineage and provenance of all the components of the workflows.

Table 2 shows the SPARQL query applied to obtain all the tasks involved in the workflow. The tasks are connected between them through the data and their semantic compatibility. Thus, the output data of a task is the input data of its following tasks. One task can connect with zero or more tasks. It is worth noting that the query handles the scenario in which the tasks have no output data, such as the case of *Data Sink* like *Upload to DB* or *Data Visualisation*.

Once all the tasks involved in a workflow are fetched, the following steps focus on obtaining their related tasks, data lineage, and provenance of each task. Table 3 shows how to get SPARQL queries as follows:

1. The first query focuses on a task and obtains all its parameters with their values; thus, the analyst can reproduce the workflow. For instance, this query

Listing 1: SPARQL query for checking the compatibility between tasks and components.

```
SELECT ?compatible
WHERE{
{SELECT DISTINCT (count(*)+1 =
    ?num_task as ?compatible)
  WHERE {
   ?wf rdf:type dmop:Workflow .
   ?wf bigowl:numTask ?num_task .
   ?wf bigowl:hasTask ?task1 .
   ?wf bigowl:hasTask ?task2 .
   ?task1 rdf:type bigowl:Task .
   ?task2 rdf:type bigowl:Task .
   ?task1 bigowl:connectedTo ?task2 .
   ?task1 bigowl:hasComponent ?comp1 .
   ?task2 bigowl:hasComponent ?comp2 .
   ?comp1 bigowl:specifiesOutputClass ?outputC .
   ?inputC rdf:type* ?outputC .
   ?comp2 bigowl:specifiesInputClass ?inputC .
   ?task1 bigowl:specifiesOutputClass ?outputTask .
   ?task2 bigowl:specifiesInputClass ?inputTask .
   FILTER (?outputC = ?inputC &&
   ?outputTask = ?inputTask ) .
  }
 GROUP BY ?task1 ?task2 ?num_task
 }
FILTER (bound(?compatible))
}
```

Table 2: SPARQL queries for obtaining all tasks and data of a workflow given.

SPARQL

```
SELECT DISTINCT ?task ?data ?taskConnected
WHERE {
 ppwo:WorkflowPollen rdf:type dmop:Workflow .
 ppwo:WorkflowPollen bigowl:hasTask ?task .
 ?task rdf:type bigowl:Task .
 OPTIONAL{
   ?task bigowl:specifiesOutputClass ?data .
   ?taskConnected ?bigowl:specifiesInputClass ?data .
   ?task bigowl:connectedTo ?taskConnected .
 }
}
```

Result

Task	Data	Task connected
TaskDataDownload	FileDownload	TaskJoinDataset
TaskImportFile	FileImport	TaskJoinDataset
TaskJoinDataset	FileJoin	TaskSplitDataset
TaskSplitDataset	FileSplit	TaskUpdateDataset
TaskDataDownload	FileDownload	TaskAttributeSelection
TaskAttributeSelection	FileAttribute	TaskSeriesCompletion
TaskSeriesCompletion	FileCompletion	TaskUpdateDataset
TaskUpdateDataset	FileUpdate	TaskInterpolation
TaskInterpolation	FileInterpolation	TaskDatasetStatistics
TaskDatasetStatistics	FileStatistics	TaskDataControlSelection
TaskDatasetStatistics	FileStatistics	TaskDataControlAlgorithm
TaskDataControlAlgorithm	FileControl	TaskMergeDataset
TaskMergeDataset	FileMerge	TaskDataControlSelection
TaskDataControlSelection	FileSelection	TaskUploadToDB
TaskDataControlSelection	FileSelection	TaskTimeSeriesForecasting
TaskTimeSeriesForecasting	FileForecasting	TaskDataVisualisation
TaskUploadToDB		
TaskDataVisualisation		

fetches the parameters and values of the interpolation task, such as the method name, the limit of the maximum number of consecutive NaNs to fill, the axis to interpolate, consecutive NaNs that will be filled in a direction, etc.

2. The following steps are related to data lineage and provenance. The second query fetches, given a task, where its output data goes. For example, the output data of the task *TaskDataDownload* is the input data of two tasks: *TaskJoinDataset* and *TaskAttribute-Selection*. Furthermore, they have as output data *FileJoin* and *FileAttribute*, respectively. This query aims to obtain the data lineage in the following steps.
3. The target of the third step is, given a task and its input data, to fetch the tasks where the data are obtained. For instance, the task *TaskImportfile* reads a file, such as a CSV or XLSX file, and this is the input data of the *TaskJoinDataSet*.
4. The fourth query focuses on obtaining the following tasks connected to a given task. For example, *TaskJoinDataset* has as output data a CSV file, and that file is the input data of *TaskSplitDataset*.

3.3. Component implementation

As described in the above section, a generic workflow for pollen forecast is designed using PPWO and TITAN's design tool, allowing the workflow's development by interchangeable components following their semantic compatibility. However, the workflow can only be used if the defined components have an available implementation that could be executed in the TITAN execution platform. As such, we have extended TITAN catalogue of components with new components for the designed workflow [2] and, our e-Science workflow is online available [3]. The new components are implemented in Python and wrapped in docker containers for flexible execution in TITAN. Each component is described as follows:

- ***Data Download***. This component fetches and obtains the data from an external database. It supports requesting the data using an API REST or other types of connections. The output data can be a JSON or CSV file.
- ***Import File***. This component is in charge of providing local files to the workflow. It supports different types of files such as XLSX, CSV, JSON, TXT, ZIP, etc.
- ***Join Dataset***. This component combines two CSV files into a new one. The defined parameters should be configured to determine how the two datasets are combined. That is, parameters will define which columns are used to join the files. Furthermore, it supports the left, right, upper, and inner join modes.
- ***Split Dataset***. Given a CSV and a list of column names, this component returns a new CSV with only the columns specified on the list.
- ***Update Dataset***. This component has two input CSV files and replaces the contents of the column given in the first CSV with the values specified in the same column provided in the second CSV.
- ***Attribute Selection***. Given a JSON file and a list of attribute names, this component returns a CSV file with the data selected from the JSON file.

[2] https://github.com/KhaosResearch/e-Science-workflow

[3] https://enbic2lab.uma.es/titan/

Table 3: SPARQL queries for fetching related tasks, data lineage and provenance of each task. The first (1) is focused on obtaining parameter values; the second (2) is related to seeking data lineage and provenance. Furthermore, the goal of the third (3) query is to bring the precedence of the input data of a task and finally, the fourth (4) is focused on obtaining the tasks connected to a given task.

SPARQL | **Result**

(1)

```
SELECT DISTINCT ?parameter ?value
WHERE {
    ppwo:WorkflowPollen rdf:type dmop:Workflow .
    ppwo:WorkflowPollen bigowl:hasTask
    ppwo:TaskInterpolation.
    OPTIONAL{
    ppwo:TaskInterpolation
       bigowl:hasParameter ?parameter .
    ?parameter bigowl:hasValue ?value}
}
```

Parameter	Value
method	'linear'
axis	None
limit	3
inplace	False
limit_direction	'both'
limit_area	None
downcast	None

(2)

```
SELECT DISTINCT ?task ?data ?taskConnected
WHERE {
    ppwo:WorkflowPollen rdf:type dmop:Workflow .
    ppwo:WorkflowPollen
        bigowl:hasTask ppwo:TaskDataDownload .
    ppwo:TaskDataDownload
       bigowl:specifiesOutputClass ?data .
    OPTIONAL{
      ?task bigowl:specifiesInputClass ?data .
    }
}
```

Task	Data Name
TaskJoinDataSet	FileJoin
TaskAttributeSelection	FileAttribute

(3)

```
SELECT DISTINCT ?taskFrom ?inputdata
WHERE {
  ppwo:WorkflowPollen rdf:type dmop:Workflow .
  ppwo:WorkflowPollen
    bigowl:hasTask ppwo:TaskJoinDataSet .
  ppwo:TaskJoinDataSet
      bigowl:specifiesInputClass ppwo:FileDownload .
  OPTIONAL{
     ppwo:TaskJoinDataSet
     bigowl:specifiesInputClass ?inputdata .
    ?taskFrom bigowl:specifiesOutputClass ?inputdata .}
  FILTER ( titan:FileDownload != ?inputdata).
}
```

Task	Data Name
TaskImportFile	FileImport

(4)

```
SELECT DISTINCT ?taskNext ?outputdata
WHERE {
    ppwo:WorkflowPollen rdf:type dmop:Workflow .
    ppwo:WorkflowPollen
        bigowl:hasTask ppwo:TaskJoinDataSet .
    ppwo:TaskJoinDataSet
    bigowl:specifiesInputClass
         ppwo:FileDownload .
    OPTIONAL{
     ppwo:TaskJoinDataSet
     bigowl:specifiesOutputClass ?outputdata .
     ?taskNext bigowl:specifiesInputClass ?outputdata .
    }
}
```

Task	Data Name
TaskSplitDataset	FileSplit

- ***Series Completion***. This component fills the missing values of a CSV column using the data from other different columns. It supports four completion criteria: R^2, slope and pair of shared data.
- ***Interpolation***. This component completes the missing values of a CSV using different interpolation techniques such as linear, time, quadratic, spline, pad, etc.
- ***Dataset Statistics***. Given a CSV file and a list of column names, this component performs a statistical analysis using the moving average and accumulative sum of column values.
- ***Data Control Algorithm***. This component aims to show the analyst a quality metric created by a machine learning algorithm, which is the Random Forest algorithm by default. The metrics provided by the component are R^2, MAE, RMSE, and RMSE/MAE.
- ***Merge Dataset***. Given two CSV files, this component aims to add the rows of the second CSV into the first one.
- ***Data Control Selection***. Given two CSV files, the goal of this component is to select the best dataset according to the selected metrics (R^2, MAE, RMSE and RMSE/MAE).
- ***Upload to DB***.This component aims to upload a JSON file to a database.
- ***Time Series forecasting***. This component will behave as a complete sub-workflow in time series forecasting. This sub-workflow will have several processing and analysis components, as shown below in a real-world use case in pollen prediction.
- ***Data Visualisation***. The goal of this component is to depict the prediction result through plots.

4. Pollen forecast experimentation

The Department of Botany and Plant Physiology of the University of Málaga (Spain) provides the pollen data to assess the workflow designed in this work. The pollen data have been collected in Málaga city centre from 1991 to 2022. The selected pollen types are *Platanus* and Amaranthaceae.

In order to try to predict the atmospheric daily pollen concentrations of the previously cited group of interesting allergenic plants (*Platanus* and Amaranthaceae), we implement, for each use case, a version of the generic workflow described in 3.3. In both approaches, we used multivariate methods for pollen predictions since univariate time-series forecasting cannot exploit any cross-series information available in a set of time series that may be correlated and share many standard features. In contrast, multivariate methods can include key patterns and structures that a collection of time series may share. In this sense, the meteorological data have been obtained from the airport meteorological station, which belongs to the Spanish State Meteorological Agency (AEMET)[4] from a target meteorological station (the nearest station to where the pollen data has been collected): temperature (maximum, minimum and mean in Celsius degrees), precipitation (mm), wind speed average (m/s), maximum gust wind speed (m/s), direction of maximum gust wind speed (tens degree), insolation (hours), maximum and minimum pressure (hPa), relative humidity (%), calm (percentage of winds below 1 km/h) and time percentage of the day where the wind has passed through a specific quadrant. The day of the year (DOY) was also added to the study. Additionally, to address missing data in the weather variables, we employed a two-step approach. First, *Series Completion* component was used to fill missing values with nearby meteorological stations, choosing stations with stable, consistent data over time. However, due to incomplete data from these stations, *Interporlation* component was also included, involving linear interpolation, spline interpolation, and regression-based methods. Finally, linear regression was selected as the most effective interpolation method, using all known data information to fill the gaps.

The designed workflow can be used by different time-series from diverse pollen types using the AEMET meteorological variables and others that the user considers to include within the workflow. However, depending on the optimal interpolation method for the dataset and the seasonality profile of the selected pollen type, the data flow and parameter settings for pollen prediction analysis might change. Therefore, the workflow must be configurable to accomplish the study with different component settings and datasets. Generally, the workflow considers two possible strategies to process and analyse our data: one based on auto-regressive models and another based on advanced deep-learning models. The idea is to allow the analyst to differentiate between two types of use cases. When pollen data represent clear stationarity and seasonality, the scientist can consider using auto-regressive models such as SARIMA, as they are easy to understand and interpret and will help the analyst to understand the outcome of the predictions.

Nonetheless, when pollen data have a very complex seasonal behaviour, these models will usually not be able to find an optimal solution. That is why we propose using more advanced deep learning models, such as Recurrent Neural Networks (RNNs), in cases where the data does not represent clear stationarity and seasonality. Additionally, both strategies can be used for the same data set and to compare results. Nevertheless, it should be noted that the execution time increases exponentially with the complexity of the problem. Furthermore, because of the semantics

[4] `https://opendata.aemet.es/centrodedescargas/inicio`

and the data provenance, the analysts can fetch the configurations used in previous experiments, which helps them repeat and improve their results.

For this reason, we conduct two use cases according to seasonal analysis results, the first one seasonal data pollen like *Platanus* and the second one non-seasonal data pollen as Amaranthaceae. Both use cases share all the workflow steps listed in Figure 2 except for *Time Series Forecasting*. Therefore, we focus on describing that step, comprised of different components (Figure 3), as detailed below.

4.1. Seasonal pollen: Platanus

According to the state-of-the-art, the pronounced main pollen season of the ornamental taxa *Platanus* is associated with February-April months which would indicate a marked seasonality. The Augmented Dickey-Fuller (ADF) test of *Platanus* pollen rejects the null hypothesis ($P - value < 0.05$), indicating that the given time-series data is stationary. Besides, the pollen trend of the main pollination period follows a normal distribution with a high amplitude difference (Figure 4A and Figure 4B). Looking at the mean and standard deviation (SD) range in the time series, a marked single peak above this series appeared every year (Figure 4A). Besides, pollen integral during a specific period (sum of daily pollen concentrations over a period of time) (Figure 4B) and seasonal decomposition confirms that this *Platanus* pollen accumulation is concentrated during February-April repeating every year, which indicates a marked seasonality because the trend fluctuation is positive (Figure 4C). In this case, we propose using an auto-regressive model such as SARIMA since *Platanus* represents clear stationarity and seasonality. Although RRN-based algorithms can obtain excellent predictions, in terms of complexity, they are costly and require a complex adjustment of the model hyperparameters.

The time-series forecasting step in the workflow is represented in Figure 3A) as a sub-workflow where all the components are arranged according to the seasonal pollen forecasting workflow. Then, it is possible to follow the complete process step-by-step.

As the first step of this workflow, we have used the component *Scaled by Months* to scale the dataset by months. Once the pollen dataset is ready to work, the *Split by Datetime* component splits the dataset into training and test set by DateTime. Then *Stationarity and Seasonality* component performs a stationarity ADF test [57] and seasonal decomposition to decompose time series data into its seasonal, trend, and residual components [58]. As we have seen above, *Platanus* pollen indicates stationarity and seasonality. For this reason, we propose a *SARIMA Model Building* component to build a SARIMA model as a specific solution for *Platanus* pollen prediction (Figure 3A). SARIMA is represented in equation 1.

$$SARIMA(p,d,q)(P,D,Q)s \quad (1)$$

Where p stands for the order of the non-seasonal auto-regressive model that decides how many prior data points will be used. q is the order of the non-seasonal moving average model, P is the order of the seasonal auto-regressive model, Q is the order of the seasonal moving average model, d is the number of non-seasonal differences, D is the number of seasonal differences, and s is the periodic term that indicates the seasonal length in the data.

SARIMA Hyperparameter Tuning component (Figure 3A) has been used for efficient hyperparameter tuning, which performs a Grid Search method to select a final model with the best parameters. The only parameter we have not inserted in the Grid Search method is s since we know that the seasonality is fulfilled yearly in the *Platanus* pollen. Therefore, we set s to 12 (12 months). Therefore, multiple models have been run with different parameter settings to check which parameter combination is the best. Consequently, the Akaike information criterion (AIC) was used to select the best model since AIC estimates each model's quality relative to the other models.

The *Model Training* component (Figure 3A) trains the model with the best parameter configuration. The final model is trained with variables previously obtained from AEMET as exogenous variables and the pollen variable as an endogenous variable from training data.

The *Predictions and Metrics Evaluation* component is carried out as an evaluation step regarding the test set, where deviated metrics are obtained (R^2, MAE, and RMSE).

To evaluate the performance of the proposed workflow, we make predictions for the last four years from our pollen data set to ensure the results' quality and the model's knowledge generalisation. In this sense, we trained our model from 1991 to 2017 and will try to predict 2018, and we will move the time window from year to year, as observed in Figure 5.

Table 4 shows the results of the evaluation of the proposed workflow as an auto-regressive modelling approach using SARIMA for the last four years of the *Platanus* pollen dataset.

Table 4: SARIMA results for *Platanus* pollen (pollen grains/m^3)

Years	R^2	MAE	RMSE
2018	0.906	26.42	53.84
2019	0.958	15.85	28.72
2020	0.905	17.46	32.06
2021	0.947	13.05	22.59

The result of the pollen prediction of the selected four last years is shown in Figure 6 by using the *Data Visualisation* component (Figure 2), which helps users to depict the prediction results through plots.

4.2. Non-seasonal pollen: Amaranthaceae

Previous studies suggest that Amaranthaceae pollen has no specific pollination period. Studying the time series profile of this pollen family, the ADF test accepts the null hypothesis ($P - value > 0.05$), indicating that the

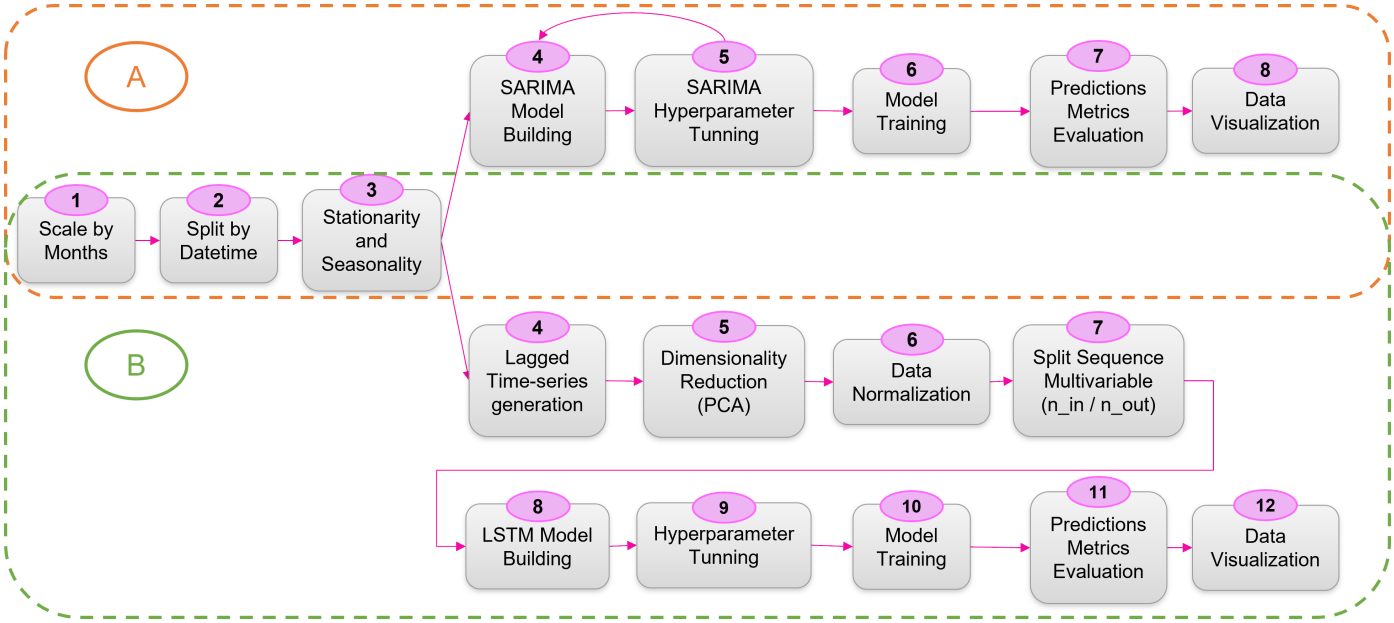

Figure 3: The Time Series Forecasting step's workflow design comprises 17 components. The components are interchangeable following their semantic compatibility. The firsts three elements are in charge of detecting the stationarity and seasonality in the data. Branch A makes an analysis using the SARIMA algorithm when data are stational and seasonal. Branch B uses an approach based on Recurrent Neural Networks and, in this case, is used for non-stational and non-seasonal data.

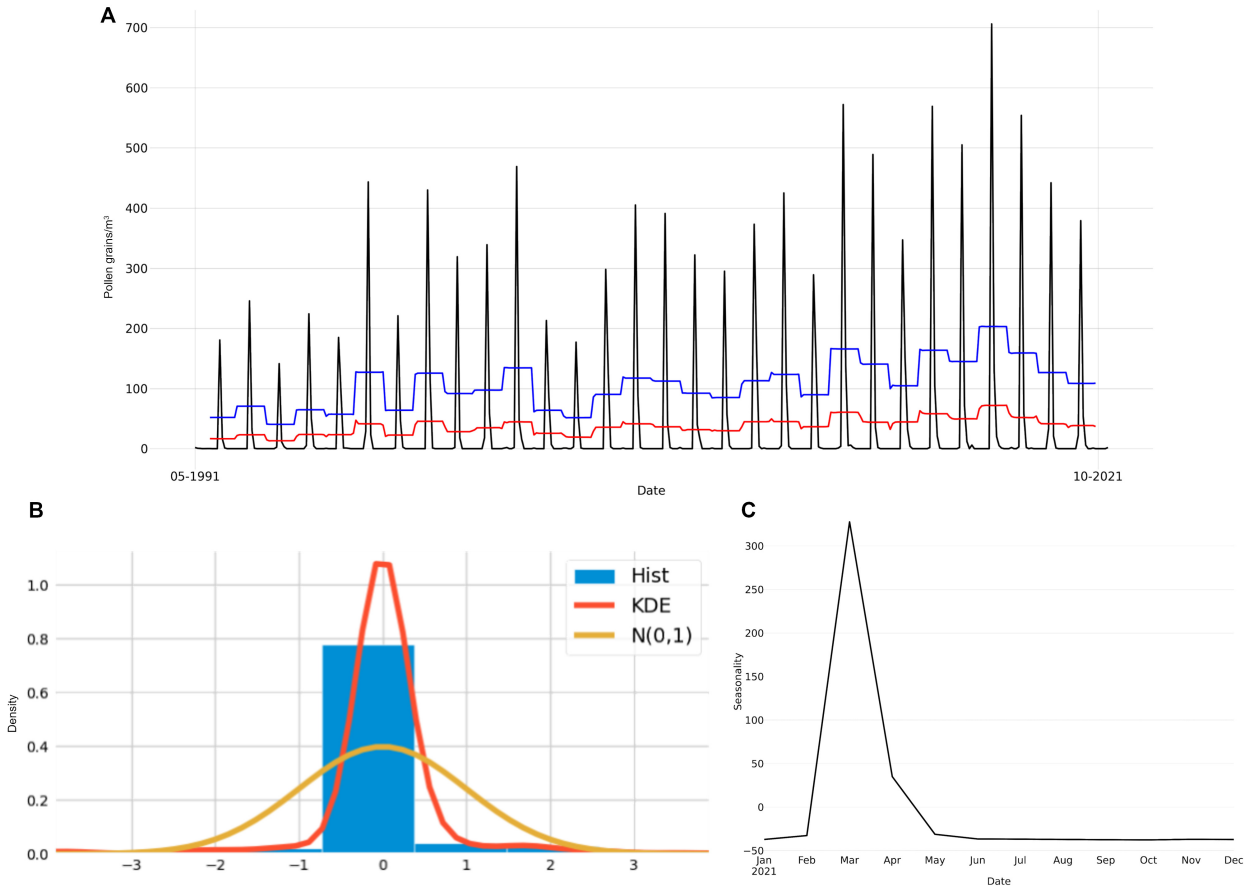

Figure 4: Stationarity and seasonality analysis of *Platanus* pollen. **A)** Time series profile (black line) of *Platanus* pollen counts from 1991 to 2022. Red and blue lines show the time series mean and standard deviation (SD), respectively. **B)** SARIMA histogram of the Kernel Density Estimation (KDE) of *Platanus* pollen. **C)** Seasonal decomposition of *Platanus* pollen time series throughout 2021.

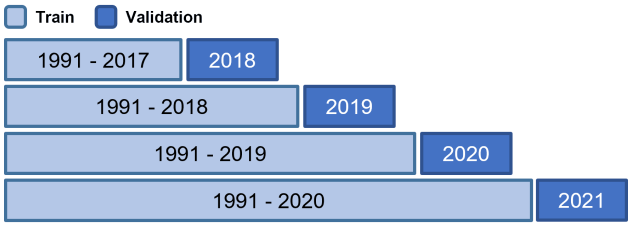

Figure 5: Pollen predictions are made separately for the last four years [2018-2019-2020-2021]. For this purpose, we selected 1991 to 2017 as the training set and evaluated the algorithm in 2018. Likewise, we shifted the time window one year and trained from 1991-2018 and predicted 2019. And so on for each of the experiments. In this way, the quality of the results could be ensured.

Amaranthaceae pollen times series is non-stationary. In addition, the trend distribution has a normal and irregular profile with a high amplitude variation (Figure 7A and Figure 7B). The time series shows two or more peaks each year of very different sizes which fall within or outside the mean and SD range (Figure 7A) which support the ADF results. Besides, pollen integral is mainly divided into two different periods of time (Figure 7B). However, the seasonal decomposition varies among the collected years; a first peak with a positive seasonality always appears following additional small peaks below the trend (negative). Therefore Amaranthaceae pollen is not seasonal, according to seasonal analysis results (Figure 7C). In this sense, as previously indicated for these cases we use an approach based on Recurrent Neural Networks (RNNs). RNNs can retain a memory of what they have already processed and thus can learn from previous iterations during training. Therefore, we propose using RNNs models that generalise traditional linear time series ARIMA models and do not require stationarity data to work as the latter. These models can be great options to help improve seasonality detection and reduce overall forecast error. Hence, we have implemented a globally trained Long Short-Term Memory network (LSTM), concretely a CNN LSTM Encoder-Decoder approach, where a single prediction model is built across all the available time series (AEMET meteorological variables) to exploit the knowledge in a group of related time series.

The sub-workflow implementing this case study is represented in Figure 3B), where all the components are arranged according to the non-seasonal pollen forecasting workflow. Then it is possible to follow the complete process step-by-step.

Like the previous workflow, the component *Scaled by Months* is used to scale the dataset by months. Then, the *Split by Datetime* component splits the dataset into training and test set by DateTime. Afterwards, *Stationarity and Seasonality* component performs the stationarity Augmented Dickey-Fuller (ADF) test and seasonal decomposition (Figure 3). As observed in Figure 7, Amaranthaceae does not represent stationarity and seasonality. This pollen is more complex than the previous use case (*Platanus*). The CNN LSTM Encoder-Decoder approach for sequence-to-sequence learning fits well in real-world use cases for non-seasonal pollen prediction. Before building and training the model, the workflow performs feature generation and preparation of the data to put it into a correct format for the model.

The data preparation process starts with the component *Lagged Time-series generation* (Figure 3B). Sometimes incorporating lagging features can improve performance. This could have an intuitive cause. Even though the lag feature values are duplicates, they are stored in vectors that can be uniquely weighted, which offers the possibility of a unique contribution. However, RNN architectures such as LSTM include a forget gate which causes them to remember values over multiple time intervals. Thus, depending on the situation, having any lagged variables in your feature set is unnecessary since LSTM can discover relevant time-dependent relationships independently. For this reason, we conduct these two approaches (with/without lagged variables) in this use case and explore which one works better. Besides, when using the *Lagged Time-series generation* component, the number of features at a particular prediction also increases at every time stamp that lag is considered. For example, in our use case, we have 18 independent features (AEMET meteorological variables), and we consider $n_lag = 12$ (12 months lag). The overall features post is 18+18*(n_lag), in case 234 features. In this regard, each time this component is used, it has to be followed by the *Dimesionality Reduction (PCA)* component to reduce the dimensionality of the dataset (Figure 3B).

In addition, the preparation steps include a *Data Normalisation* component that uses the mean and standard deviation of the training set. This component becomes necessary in multivariate time series datasets, where each time series may possess observations with different scales that are nonlinear and highly dynamic in behaviour. Even more, when applying Deep Neural Networks as LSTM, data normalisation significantly impacts the time series forecasting [59] (Figure 3B).

Moreover, feature generation includes the component of *Split Sequence Multivariable*. We employ this component to generate time series of our feature set with a sequence length of 12 months (Figure 3B). LSTM expects input data to be a 3D tensor such that *[batch size, time steps, features]. Batch size* means how many samples are in each batch during training and testing. *Time steps* means how many values exist in a sequence. *Features* mean how many dimensions are used to represent data in one time step. Also, we need to indicate n_out equivalent to how many months we want to forecast in the future. In our case, an entire year or 12 months.

Once the data is in the correct format, the workflow moves on to the next component of the *LSTM Model Building* (Figure 3B). The CNN LSTM Encoder-decoder architecture contains two subsequent 1D convolutional layers (Conv1D) placed at the beginning to extract features with a kernel size of 3. Then it is flattened after *MaxPooling*

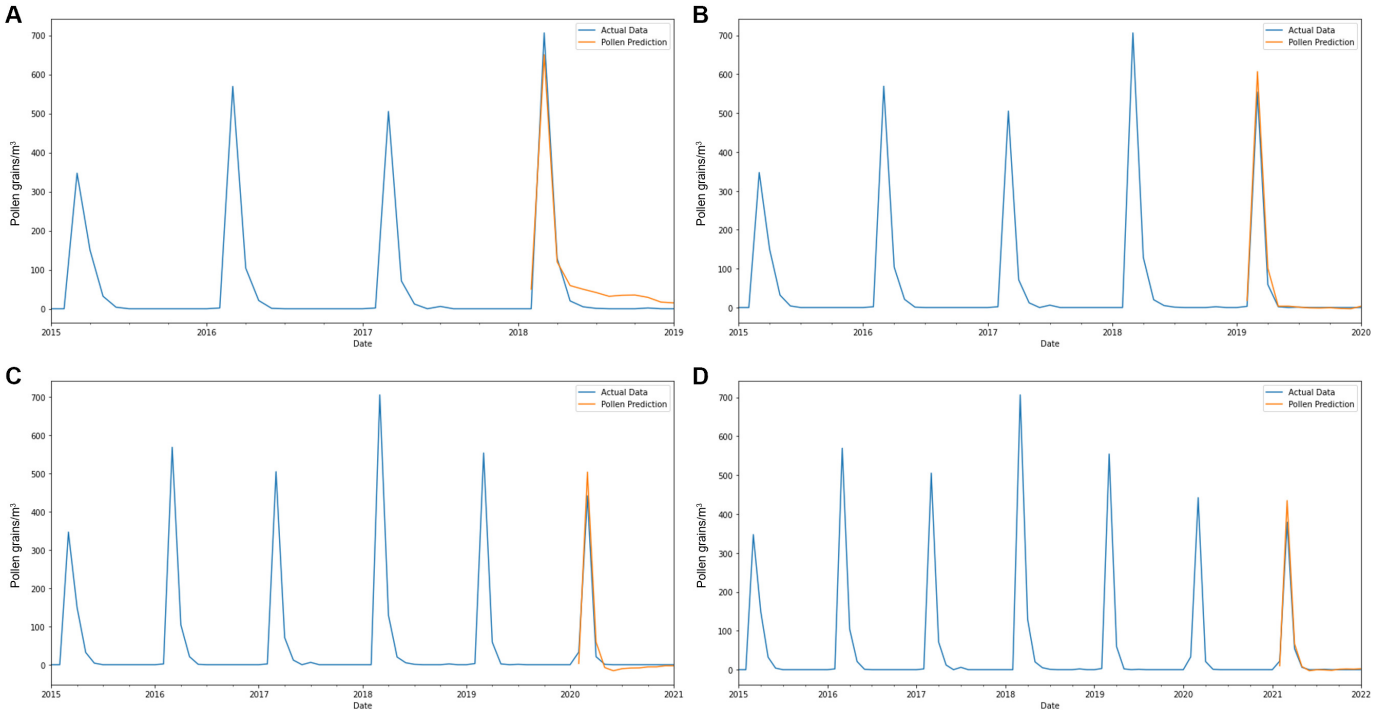

Figure 6: Selection of time window plots from training and prediction results of the *Platanus* pollen data scaled by month, according to Figure 5. **A)** Model training with 1991 to 2017 period and prediction with 2018. **B)** Model training with 1991 to 2018 period and prediction with 2019. **C)** Model training with 1991 to 2019 period and prediction with 2020. **D)** Model training with 1991 to 2020 period and prediction with 2021.

the results of Conv1D. This encoder part is responsible for reading and interpreting the input. The encoder part compresses the information into a small representation (a fixed-length vector) of the original input. This context vector is given to the decoder as input to interpret and forecast. A *RepeatVector layer* is used to repeat the context vector we obtain from the encoder part. It is repeated for the number of future steps you want to forecast and fed into the decoder. The output received from the decoder in each time step is mixed. A fully connected Dense layer is applied to each time step via the *TimeDistributed* wrapper; consequently, it separates the output for each time step.

LSTM Hyperparameter Tuning component is used for efficient hyperparameter tuning (Figure 3B). Hyperparameter Tuning is executed by varying the number of neurons in the LSTM layer by 100, 200, 300, 400, and 500 neurons and also varying the filter size in 1D convolutional layers, obtaining, in general, the best result for a filter size of 64. The hyperparameters are tuned by performing 30 evaluations for each configuration on the training set using Grid Search. Thus, we avoid the stochastic nature of the algorithm or evaluation procedure. A detailed description of the experimental settings of the CNN-LSTM model can be observed in Table 5.

The model is trained in each iteration using the Adam optimiser and MSE as loss function during 500 epochs. Moreover, the component implements early-stopping to allow the training job to finish early if there are no significant performance improvements after 10 epochs. Furthermore, the component implements *ReduceLROnPlateau* decreasing learning rate in 0.001 with a minimum learning rate of $1 \exp -8$ if there are no significant performance improvements after 5 epochs.

Like in *Platanus* use case, the performance of the proposed LSTM-based strategy is evaluated over the last four years (2018-2019-2020-2021) of the Amaranthaceae pollen dataset (with/out lagged variables) to ensure the quality of the results, as seen in Figure 5. The table 6 shows the metrics results of the predictions of the best configurations obtained by the *LSTM Hyperparameter Tuning* component for each of the last four years. The metrics are the results of the mean and standard deviation of the 30 evaluations performed. It is worth noting that in the year 2019, the model obtained the lowest R^2 of 0.88 with a deviation of 0.08; this seems reasonable since that particular year, a "jump" can be observed in the graph (Figure 8). However, the model fits reasonably well thanks to the help of the additional meteorological variables that have been inserted in the multivariate model to give added information to the model and thus obtain a more robust model. The model has obtained excellent results since the predictions are properly adjusted to the actual values.

Moreover, as shown in Table 6, the use of the lagged Time-series generation component has performed better in two of the four predicted years. Therefore, we con-

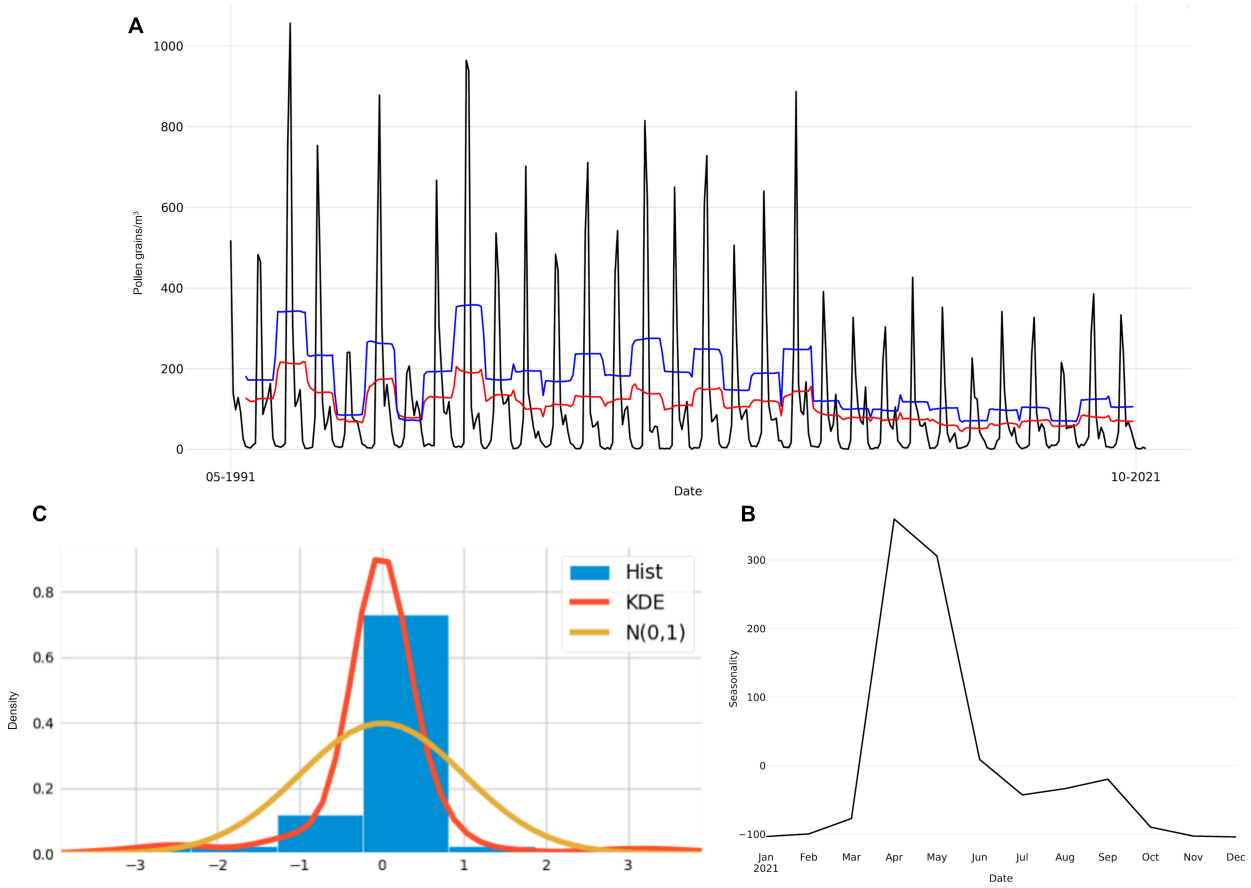

Figure 7: Stationarity and seasonality analysis of Amaranthaceae pollen. **A)** Time series profile (black line) of Amaranthaceae pollen counts from 1991 to 2022. Red and blue lines show the time series mean and standard deviation (SD), respectively. **B)** SARIMA histogram of the Kernel Density Estimation (KDE) of Amaranthaceae pollen. **C)** Seasonal decomposition of Amaranthaceae pollen time series throughout 2021.

sider that it can be helpful in other scenarios with different pollen types since the new lag feature values created, despite being values coming from other variables, offer the possibility of being a unique contribution and uniquely weighted in the model.

The result of the pollen prediction of the selected four last years of Amaranthaceae pollen is shown in Figure 8 by using the *Data Visualisation* component (Figure 2), which helps users to depict the prediction result through plots.

5. Discussions

One of the leading research findings we claim with the design and implementation of this e-Science workflow is the ability to provide a methodology for analysing time series data, which allows not only supplying a catalogue of interoperable components, but also the semantic capacity for annotating all the meta-data during the process to let replicate and improve the analysis. The meta-data is integrated following the PPWO structure and stored in an RDF repository. This enables the publication of the results as Linked Open Data.

The alignment of PPWO with SWEET, Plant, NCBITAXON and BIGOWL ontologies allows the creation of all the meta-data and semantic knowledge that explicitly describes and registers the data lineage from data sources to visualisation results as well as algorithms parameters configuration. Furthermore, all these features facilitate the analysis replication and indeed enhance the e-Science workflow due to the component configurations of former experiments helping in new ones.

This work also provides the scientific community with a catalogue of open-source components for data integration, processing and analysing pollen data. They are included in the open-source TITAN framework, which allows designing and executing the proposed workflow using the created group of components. In addition, the workflow configuration is widely flexible, allowing a parameterisation and combination of components according to researchers' requirements.

Based on the analysis provided in the two case studies, the scientists can identify the correct path the data follow and how they are modified to obtain added value for the pollen domain of knowledge. For example, a series of data sources involving information about pollen (with data collected by the scientists) are semantically related (or linked) to the results obtained from AEMET in the form of time

Layers	**Parameters settings**	
Conv1D	Filter Kernel Activation	32, 64 3 relu
Conv1D	Filter Kernel Activation	32, 64 3 relu
MaxPooling	Pooling Size	2
Flatten		
RepeatVector		
LSTM	Units Activation Return Sequence	100, 200, 300, 400, 500 relu True
Time Distributed	Dense Activation	100 relu
Time Distributed	Dense	1

Table 5: CNN-LSTM parameter settings

Table 6: CNN LSTM Encoder-Decoder results for Amaranthaceae pollen (pollen grains/m^3)

Years	MAE (mean)	MAE (SD)	RMSE (mean)	RMSE (SD)	R^2 (mean)	R^2 (SD)
2018 (with lagged)	20.06	4.61	29.4	8.02	0.905	0.05
2019 (without lagged)	21.55	4.15	28.81	7.21	0.88	0.08
2020 (without lagged)	18.02	5.05	22.31	6.23	0.92	0.03
2021 (with lagged)	20.47	7.02	28.5	8.51	0.904	0.04

series. Furthermore, in each step of the workflow, that data is attached as well to the component configuration; thus, in this way, the data lineage is registered with semantic annotations, enabling the experimentation replicability.

In terms of practical implications, two possible alternatives for pollen prediction have been proposed, trying to cover the different generic pollination profiles of a wide range of plant species. The first proposal is based on a strategy based on autoregressive models, where we have mainly focused on the SARIMA model, although it is not limited to this model. This proposal is presented as an alternative to predict pollen types with stationarity and seasonality characteristics since these models work pretty well in these cases where the seasonality of the time series is clearly defined. As seen in the case of *Platanus* pollen (seasonal data pollen), the SARIMA model can predict pollen values with promising results.

However, for other pollen types where seasonality is not so clear, these models can not fit their predictions as well as more complex deep learning models such as LSTM. In contrast, RNN models can extract internal features from the time series over short and long distances in time. Furthermore, RNN can detect patterns in the time series even if the time series does not demonstrate stationarity or seasonality. Therefore, these models help improve the detection of these pollen types and reduce the overall forecast error.

Nevertheless, it is recommended to re-train the workflow from time to time or evaluate the models with pollen data from the last 10-15 years to avoid masking new pollen trends that may appear due to climate change's local effects or possible changes in the flowering season.

Therefore, our e-Science workflow provides the expert biologist with multiple possible strategies for pollen prediction at future stages, helping to anticipate allergy incidence by estimating the main pollen season of different types of plants. The different features of our analysis strategy can be summarised as follows:

- Our method based on autoregressive models has demonstrated excellent efficiency in predicting pollen with stationarity and seasonality features.
- In complex pollen application cases with neither stationarity nor seasonality, our strategy based on RNN models has established its ability to produce good outcomes. These outcomes perform better than those attained using conventional autoregressive models.
- The strategy based on RNN models can be used regardless of the type of pollen. However, in this work, we have concluded that for seasonal pollen use cases, it is not worth using complex deep learning models

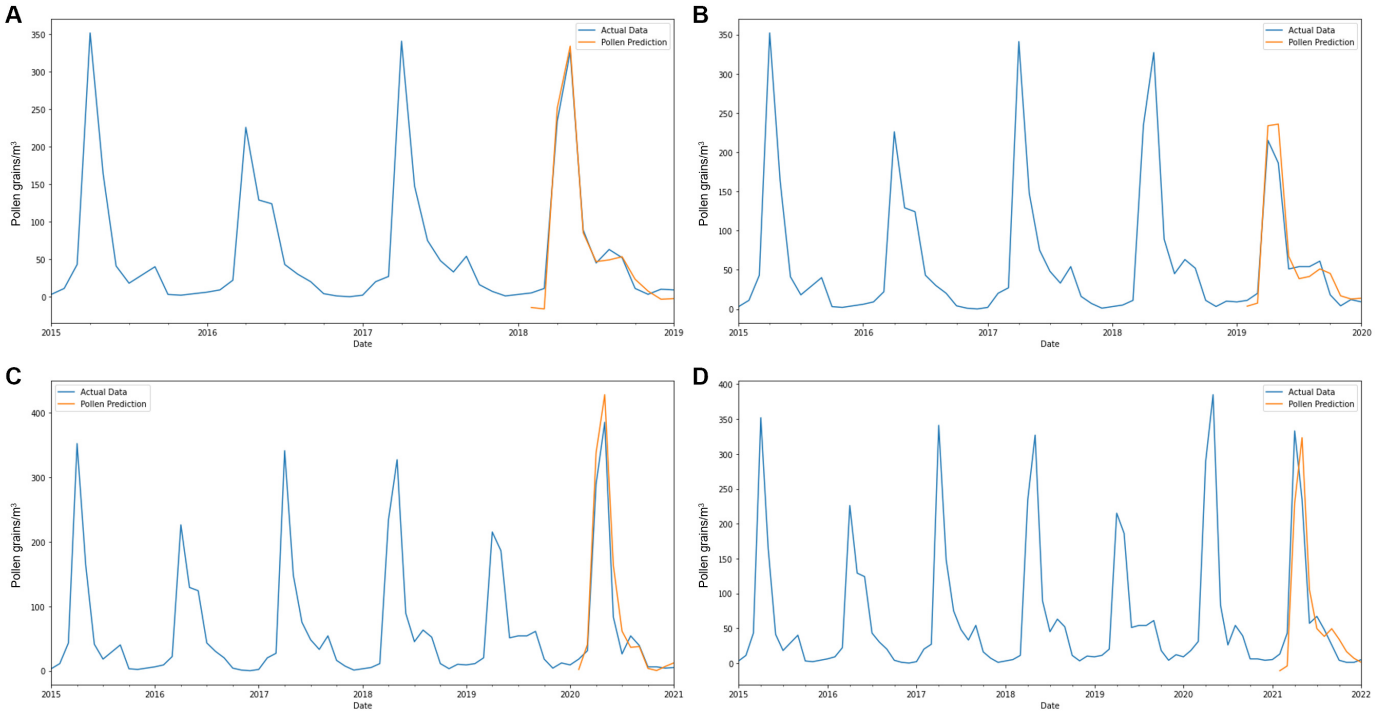

Figure 8: Selection of time window plots from training and prediction results of the Amaranthaceae pollen data scaled by month, according to Figure 5. **A)** Model training with 1991 to 2017 period and prediction with 2018. **B)** Model training with 1991 to 2018 period and prediction with 2019. **C)** Model training with 1991 to 2019 period and prediction with 2020. **D)** Model training with 1991 to 2020 period and prediction with 2021.

due to their complexity in hyperparameterisation, high computational cost and poor interpretability of the results.

6. Conclusions and future work

This work presents a semantic-enhanced e-Science workflow and an ontology to analyse data pollen. Both have been designed and developed using the TITAN framework. The ability to integrate knowledge domain data with processing components, and monitor and maintain data lineage and consistency at any stage of the data life cycle, are just a few of the significant benefits of semantic annotation of all the meta-data involved in an e-Science workflow.

Furthermore, the TITAN catalogue has been increased in new 32 components, created for designing and building the proposed e-Science workflow. There are components for collecting, transforming, analysing and visualising data. Although the e-Science workflow has been assessed with pollen data, it can be used with any type of time series data, enabling the design of other novel workflows based on the available component catalogue in TITAN.

Future work includes creating multiple example workflows focused on processing and analysing time-series data. The number of functional components in the internal TITAN catalogue will increase due to all these additional samples. These components will be made available to the scientific and industrial communities.

Acknowledgements

This work has been partially funded by the Spanish Ministry of Science and Innovation via grant (funded by MCIN/AEI/10.13039/501100011033/) PID2020-112540RB-C41, AETHER-UMA (A smart data holistic approach for context-aware data analytics: semantics and context exploitation), and grant "Environmental and Biodiversity Climate Change Lab (EnBiC2-Lab)" LIFEWATCH-2019-11-UMA-01 (AEI/FEDER, UE). A. Picornell has been supported by a postdoctoral grant financed by the Consejería de Transformación Económica, Industria, Conocimiento y Universidades (Junta de Andalucía, POSTDOC_21_00056).

References

[1] A. J. Hey, A. E. Trefethen, The data deluge: An e-science perspective.

[2] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, J. Myers, Examining the challenges of scientific workflows, Computer 40 (12) (2007) 24–32.

[3] Y. L. Simmhan, B. Plale, D. Gannon, A survey of data provenance in e-science, ACM Sigmod Record 34 (3) (2005) 31–36.

[4] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, J. Zhao, Prov-o: The prov ontology.

[5] K. G. Achilleos, C. C. Kannas, C. A. Nicolaou, C. S. Pattichis, V. J. Promponas, Open source workflow systems in life sciences informatics, in: 2012 IEEE 12th International Conference on Bioinformatics & Bioengineering (BIBE), IEEE, 2012, pp. 552–558.

[6] I. J. Taylor, E. Deelman, D. B. Gannon, M. Shields, et al., Workflows for e-Science: scientific workflows for grids, Vol. 1, Springer, 2007.

[7] I. Wassink, H. Rauwerda, P. Van Der Vet, T. Breit, A. Nijholt, e-bioflow: Different perspectives on scientific workflows, in: International Conference on Bioinformatics Research and Development, Springer, 2008, pp. 243–257.

[8] L. Ambrósio, H. Linhares, J. M. N. David, R. Braga, W. Arbex, M. M. Campos, R. Capilla, Enhancing the reuse of scientific experiments for agricultural software ecosystems, Journal of Grid Computing 19 (4) (2021) 1–24.

[9] E. Deelman, T. Peterka, I. Altintas, C. D. Carothers, K. K. van Dam, K. Moreland, M. Parashar, L. Ramakrishnan, M. Taufer, J. Vetter, The future of scientific workflows, The International Journal of High Performance Computing Applications 32 (1) (2018) 159–175.

[10] R. Pandey, S. Silakari, Investigations on optimizing performance of the distributed computing in heterogeneous environment using machine learning technique for large scale data set, Materials Today: Proceedings.

[11] A. Benítez-Hidalgo, C. Barba-González, J. García-Nieto, P. Gutiérrez-Moncayo, M. Paneque, A. J. Nebro, M. del Mar Roldán-García, J. F. Aldana-Montes, I. Navas-Delgado, Titan: A knowledge-based platform for big data workflow management, Knowledge-Based Systems 232 (2021) 107489.

[12] C. Barba-González, J. García-Nieto, M. del Mar Roldán-García, I. Navas-Delgado, A. J. Nebro, J. F. Aldana-Montes, Bigowl: knowledge centered big data analytics, Expert Systems with Applications 115 (2019) 543–556.

[13] B. B. Rad, H. J. Bhatti, M. Ahmadi, An introduction to docker and analysis of its performance, International Journal of Computer Science and Network Security (IJCSNS) 17 (3) (2017) 228.

[14] C. A. Akdis, P. W. Hellings, I. Agache, Global atlas of allergic rhinitis and chronic rhinosinusitis, European Academy of Allergic Rhinitis and Chronic Rhinosinusitis, 2015.

[15] S. Gilles, C. Blume, M. Wimmer, A. Damialis, L. Meulenbroek, M. Gökkaya, C. Bergougnan, S. Eisenbart, N. Sundell, M. Lindh, et al., Pollen exposure weakens innate defense against respiratory viruses, Allergy 75 (3) (2020) 576–587.

[16] M. Sofiev, P. Siljamo, H. Ranta, T. Linkosalo, S. Jaeger, A. Rasmussen, A. Rantio-Lehtimaki, E. Severova, J. Kukkonen, A numerical model of birch pollen emission and dispersion in the atmosphere. description of the emission module, International journal of biometeorology 57 (1) (2013) 45–58.

[17] K. Piotrowska-Weryszko, E. Weryszko-Chmielewska, A. Sulborska, A. Konarska, M. Dmitruk, B. M. Kaszewski, Amaranthaceae pollen grains as indicator of climate change in lublin (poland), Environmental Research 193 (2021) 110542.

[18] C. Galán, P. Alcázar, J. Oteros, H. García-Mozo, M. Aira, J. Belmonte, C. D. de la Guardia, D. Fernández-González, M. Gutierrez-Bustillo, S. Moreno-Grau, et al., Airborne pollen trends in the iberian peninsula, Science of the Total Environment 550 (2016) 53–59.

[19] F. J. Rodríguez-Rajo, R. M. Valencia-Barrera, A. M. Vega-Maray, F. J. Suárez, D. Fernández-González, V. Jato, Prediction of airborne alnus pollen concentration by using arima models, Annals of Agricultural and Environmental Medicine 13 (1) (2006) 25.

[20] A. Bhawnani, P. Parwani, N. Bhavnani, K. Rupreja, S. Sahu, Forecasting pollen concentration using ml and pollen taxa classification using cnn, Available at SSRN 3882612.

[21] A. S. Lewis, W. M. Woelmer, H. L. Wander, D. W. Howard, J. W. Smith, R. P. McClure, M. E. Lofton, N. W. Hammond, R. S. Corrigan, R. Q. Thomas, et al., Increased adoption of best practices in ecological forecasting enables comparisons of forecastability, Ecological Applications (2021) e02500.

[22] L. Ascari, C. Siniscalco, G. Palestini, M. J. Lisperguer, E. S. Huerta, T. De Gregorio, S. Bregaglio, Relationships between yield and pollen concentrations in chilean hazelnut orchards, European Journal of Agronomy 115 (2020) 126036.

[23] P. Alcázar, H. García-Mozo, M. del Mar Trigo, L. Ruiz, F. J. González-Minero, P. Hidalgo, C. D. de la Guardia, C. Galán, Platanus pollen season in andalusia (southern spain): trends and modeling, Journal of Environmental Monitoring 13 (9) (2011) 2502–2510.

[24] P. Pyšek, P. W. Lambdon, M. Arianoutsou, I. Kühn, J. Pino, M. Winter, Alien vascular plants of europe, in: Handbook of alien species in Europe, Springer, 2009, pp. 43–61.

[25] T. Stemeseder, W. Hemmer, T. Hawranek, G. Gadermaier, Marker allergens of weed pollen–basic considerations and diagnostic benefits in the clinical routine, Allergo journal international 23 (8) (2014) 274–280.

[26] N. F. Noy, D. L. McGuinness, et al., Ontology development 101: A guide to creating your first ontology (2001).

[27] B. McBride, The resource description framework (rdf) and its vocabulary description language rdfs, in: Handbook on ontologies, Springer, 2004, pp. 51–65.

[28] G. Antoniou, F. v. Harmelen, Web ontology language: Owl, in: Handbook on ontologies, Springer, 2004, pp. 67–92.

[29] S. Harris, A. Seaborne, E. Prud'hommeaux, Sparql 1.1 query language, W3C recommendation 21 (10).

[30] T. McPhillips, S. Bowers, D. Zinn, B. Ludäscher, Scientific workflow design for mere mortals, Future Generation Computer Systems 25 (5) (2009) 541–551.

[31] D. Turi, P. Missier, C. Goble, D. De Roure, T. Oinn, Taverna workflows: Syntax and semantics, in: Third IEEE International Conference on e-Science and Grid Computing (e-Science 2007), IEEE, 2007, pp. 441–448.

[32] A. Goderis, P. Li, C. Goble, Workflow discovery: the problem, a case study from e-science and a graph-based solution, in: 2006 IEEE International Conference on Web Services (ICWS'06), IEEE, 2006, pp. 312–319.

[33] V. Korkhov, D. Vasyunin, A. Wibisono, A. S. Belloum, M. A. Inda, M. Roos, T. M. Breit, L. O. Hertzberger, Vlam-g: Interactive data driven workflow engine for grid-enabled resources, Scientific Programming 15 (3) (2007) 173–188.

[34] L. Ramakrishnan, C. Koelbel, Y.-S. Kee, R. Wolski, D. Nurmi, D. Gannon, G. Obertelli, A. YarKhan, A. Mandal, T. M. Huang, et al., Vgrads: enabling e-science workflows on grids and clouds with fault tolerance, in: Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis, IEEE, 2009, pp. 1–12.

[35] I. Altintas, O. Barney, E. Jaeger-Frank, Provenance collection support in the kepler scientific workflow system, in: International Provenance and Annotation Workshop, Springer, 2006, pp. 118–132.

[36] X. Li, B. Plale, N. Vijayakumar, R. Ramachandran, S. Graves, H. Conover, Real-time storm detection and weather forecast activation through data mining and events processing, Earth Science Informatics 1 (2) (2008) 49–57.

[37] J. Oteros, F. Orlandi, H. García-Mozo, F. Aguilera, A. B. Dhiab, T. Bonofiglio, M. Abichou, L. Ruiz-Valenzuela, M. M. Del Trigo, C. D. de la Guardia, et al., Better prediction of mediterranean olive production using pollen-based models, Agronomy for sustainable development 34 (3) (2014) 685–694.

[38] A. Picornell, J. Oteros, M. Trigo, D. Gharbi, S. D. Fernández, M. M. Caballero, F. Toro, J. García-Sánchez, R. Ruiz-Mata, B. Cabezudo, et al., Increasing resolution of airborne pollen forecasting at a discrete sampled area in the southwest mediterranean basin, Chemosphere 234 (2019) 668–681.

[39] R. Ruíz Mata, A. Picornell, M. Recio, B. Cabezudo, M. d. M. Trigo-Perez, et al., Comparative between forecast methods in aerobiology.

[40] Y. Lops, Y. Choi, E. Eslami, A. Sayeed, Real-time 7-day forecast of pollen counts using a deep convolutional neural network, Neural Computing and Applications 32 (15) (2020) 11827–

11836.
[41] L. Zhang, L. Zhang, B. Du, Deep learning for remote sensing data: A technical tutorial on the state of the art, IEEE Geoscience and Remote Sensing Magazine 4 (2) (2016) 22–40.
[42] J. Rojo, R. Rivero, J. Romero-Morte, F. Fernández-González, R. Pérez-Badia, Modeling pollen time series using seasonal-trend decomposition procedure based on loess smoothing, International journal of biometeorology 61 (2) (2017) 335–348.
[43] G. U. Yule, Vii. on a method of investigating periodicities disturbed series, with special reference to wolfer's sunspot numbers, Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character 226 (636-646) (1927) 267–298.
[44] M. I. Jordan, T. M. Mitchell, Machine learning: Trends, perspectives, and prospects, Science 349 (6245) (2015) 255–260.
[45] U. Thissen, R. Van Brakel, A. De Weijer, W. Melssen, L. Buydens, Using support vector machines for time series prediction, Chemometrics and intelligent laboratory systems 69 (1-2) (2003) 35–49.
[46] N. I. Sapankevych, R. Sankar, Time series prediction using support vector machines: a survey, IEEE Computational Intelligence Magazine 4 (2) (2009) 24–38.
[47] B. Paudel, T. Chu, M. Chen, V. Sampath, M. Prunicki, K. C. Nadeau, Increased duration of pollen and mold exposure are linked to climate change, Scientific reports 11 (1) (2021) 1–12.
[48] F. A. Gers, D. Eck, J. Schmidhuber, Applying lstm to time series predictable through time-window approaches, in: Neural Nets WIRN Vietri-01, Springer, 2002, pp. 193–200.
[49] Y. Li, Z. Zhu, D. Kong, H. Han, Y. Zhao, Ea-lstm: Evolutionary attention-based lstm for time series prediction, Knowledge-Based Systems 181 (2019) 104785.
[50] Z. Karevan, J. A. Suykens, Transductive lstm for time-series prediction: An application to weather forecasting, Neural Networks 125 (2020) 1–9.
[51] X.-H. Le, H. V. Ho, G. Lee, S. Jung, Application of long short-term memory (lstm) neural network for flood forecasting, Water 11 (7) (2019) 1387.
[52] R. Navares, J. L. Aznarte, Predicting air quality with deep learning lstm: Towards comprehensive models, Ecological Informatics 55 (2020) 101019.
[53] N. Noy, D. M. (Hrsg.), Ontology development 101: A guide to creating your first ontology. Technical report, 2001.
[54] R. G. Raskin, M. J. Pan, Knowledge representation in the semantic web for earth and environmental terminology (sweet), Computers & geosciences 31 (9) (2005) 1119–1125.
[55] P. Jaiswal, S. Avraham, K. Ilic, E. A. Kellogg, S. McCouch, A. Pujar, L. Reiser, S. Y. Rhee, M. M. Sachs, M. Schaeffer, et al., Plant ontology (po): a controlled vocabulary of plant structures and growth stages, Comparative and functional genomics 6 (7-8) (2005) 388–397.
[56] Database resources of the national center for biotechnology information, Nucleic acids research 46 (D1) (2018) D8–D13.
[57] R. Mushtaq, Augmented dickey fuller test.
[58] R. B. Cleveland, W. S. Cleveland, J. E. McRae, I. Terpenning, Stl: A seasonal-trend decomposition, J. Off. Stat 6 (1) (1990) 3–73.
[59] S. Bhanja, A. Das, Impact of data normalization on deep neural network for time series forecasting, arXiv preprint arXiv:1812.05519.