This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

ExtremeEarth Meets Satellite Data From Space

Desta Haileselassie Hagos, Theofilos Kakantousis, Vladimir Vlassov, Sina Sheikholeslami, Tianze Wang, Jim Dowling, Claudia Paris, Daniele Marinelli, Giulio Weikmann, Lorenzo Bruzzone, Salman Khaleghian, Thomas Kræmer, Torbjørn Eltoft, Andrea Marinoni, Despina-Athanasia Pantazi, George Stamoulis, Dimitris Bilidas, George Papadakis, George Mandilaras, Manolis Koubarakis, Antonis Troumpoukis, Stasinos Konstantopoulos, Markus Muerth, Florian Appel, Andrew Fleming, and Andreas Cziferszky

***Abstract*—Bringing together a number of cutting-edge technologies that range from storing extremely large volumes of data all the way to developing scalable machine learning and deep learning algorithms in a distributed manner, and having them operate over the same infrastructure poses unprecedented challenges. One of these challenges is the integration of European Space Agency (ESA)'s Thematic Exploitation Platforms (TEPs) and data information access service platforms with a data platform, namely Hopsworks, that enables scalable data processing, machine learning, and deep learning on Copernicus data, and development of very large training datasets for deep learning architectures targeting the classification of Sentinel images. In this paper, we present the software architecture of *ExtremeEarth* that aims at the development of scalable deep learning and geospatial analytics techniques for processing and analyzing petabytes of Copernicus data. The *ExtremeEarth* software infrastructure seamlessly integrates existing and novel software platforms and tools for storing, accessing, processing, analyzing, and visualizing large amounts of Copernicus data. New techniques in the areas of remote sensing and artificial intelligence with an emphasis on deep learning are developed. These techniques and corresponding software presented in this paper are to be integrated with and used in two ESA TEPs, namely Polar and Food Security TEPs. Furthermore, we present the integration of Hopsworks with the Polar and Food Security use cases and the flow of events for the products offered through the TEPs.**

***Keywords*—Copernicus, ExtremeEarth, Hopsworks, Earth Observation, Linked Geospatial Data, Deep Learning, Artificial Intelligence, Remote Sensing, Food Security, Polar Regions**

Desta Haileselassie Hagos, Vladimir Vlassov, Sina Sheikholeslami, Tianze Wang, and Jim Dowling are with the KTH Royal Institute of Technology, Stockholm, Sweden (e-mail: {destah, vladv, sinash, tianzew, jdowling}@kth.se).

Theofilos Kakantousis and Jim Dowling are with the Logical Clocks, Stockholm, Sweden (e-mail: {theo, jim}@logicalclocks.com).

Claudia Paris, Daniele Marinelli, Giulio Weikmann, and Lorenzo Bruzzone are with the University of Trento, Trento, Italy (e-mail: {claudia.paris, daniele.marinelli, giulio.weikmann, lorenzo.bruzzone}@unitn.it).

Salman Khaleghian, Thomas Kræmer, Torbjørn Eltoft, and Andrea Marinoni are with the UiT The Arctic University of Norway, Norway (e-mail: {salman.khaleghian, thomas.kramer, torbjorn.eltoft, andrea.marinoni}@uit.no).

Despina-Athanasia Pantazi, George Stamoulis, Dimitris Bilidas, George Papadakis, George Mandilaras, and Manolis Koubarakis are with the National and Kapodistrian University of Athens (UoA), Athens, Greece (e-mail: {dpantazi, gstam, d.bilidas, gpapadis, gmandi, koubarak}@di.uoa.gr).

Antonis Troumpoukis and Stasinos Konstantopoulos are with the National Center for Scientific Research - Demokritos (NCSR-D), Paraskevi, Greece (e-mail: {antru, konstant}@iit.demokritos.gr).

Markus Muerth and Florian Appel are with the VISTA Remote Sensing in Geosciences GmbH, Munich, Germany (e-mail: {muerth, appel}@vista-geo.de).

Andrew Fleming and Andreas Cziferszky are with the British Antarctic Survey, Cambridge, UK (e-mail: {ahf, ancz}@bas.ac.uk).

Submitted to IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

I. Introduction

FOR more than 20 years, Earth Observation (EO) satellites developed and operated by the ESA have provided a wealth of data. In the coming years, the Sentinel missions[1], along with the Copernicus contributing missions[2], as well as earth explorers [1] and other third-party missions [2] will provide routine monitoring of our environment at a global scale, thereby delivering an unprecedented amount of data. This expanding operational capability of global monitoring from space, combined with data from long-term EO archive will provide users with unprecedented insight into how oceans, atmosphere, land, and ice operate and interact as part of an interconnected earth system. The Copernicus mission, with its free and open Sentinel satellite data, is opening the way towards systematic large-scale scientific EO analysis relying on quality-controlled and calibrated data.

Copernicus. The European Union's EO flagship programme for monitoring the planet earth and its environment is called Copernicus. The Copernicus programme consists of a complex set of systems that collect data from satellites and in-situ sensors, process this data, and provide users with reliable and up-to-date information on a range of environmental and security issues. The EO satellites that provide the data of Copernicus are the Sentinels, which are developed for the specific needs of the Copernicus programme, and the contributing missions, which are operated by national, European, or international organizations. The access to Sentinel data is regulated by the EU law and it is full, open, and free. Information extracted from Copernicus data is made available to users through the Copernicus services addressing six core thematic areas: land monitoring, marine environment monitoring, atmosphere monitoring, emergency management, security, and climate change.

The land monitoring service provides geographical information on land cover and its changes, land use, water management, forest monitoring, agriculture and food security, soil quality, etc. The marine environment monitoring service includes the monitoring for marine safety and transport, variability, and dynamics of the ocean and marine ecosystems for the global ocean, ocean forecasting, and the polar environment. The atmosphere monitoring service provides continuous data and information on atmospheric composition. The emergency management service provides all actors involved in the management of natural disasters, man-made

[1] https://sentinels.copernicus.eu
[2] https://www.copernicus.eu/en

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

disasters such as floods, forest fires, and earthquakes, and humanitarian crises with timely and accurate geospatial information collected from satellite remote sensing. The Copernicus service for security applications supports security challenges such as maritime surveillance and border control. Lastly, the climate change service of Copernicus provides authoritative information about the past, present, and future climate in Europe and the rest of the world.

An important activity related to Copernicus is ESA's TEPs. A TEP is a collaborative, virtual work environment addressing a class of users and providing easy access to EO data, algorithms, computing, and networking resources required to work with them, through one coherent interface. The fundamental principle of the TEPs is to move the user to the data and tools as opposed to the traditional approach of downloading, replicating, and exploiting data locally. Another important development in the context of Copernicus is the implementation of five Copernicus *Data and Information Access Services (DIAS)*.

The availability of the growing volume of diverse environmental data from multiple distributed sources in space represents a unique opportunity for scientific research and applications. Nevertheless, the task of achieving its full potential in terms of extracting valuable knowledge and commercial value at the extreme scale of data expected in Copernicus is a major challenge associated with this opportunity. In this paper, we present the *ExtremeEarth* software infrastructure that builds on seamless integration of both existing and novel software platforms and tools for storing, accessing, processing, analyzing, and visualizing large amounts of Copernicus EO data. *ExtremeEarth* facilitates the application of deep learning techniques for the analysis of EO data, in particular classification tasks. The knowledge extracted from the satellite data is then encoded as linked geospatial data and integrated with other open linked data sources. Relevant technologies enable both EO and non-EO expert users to pose queries over this information in order to develop prototype environmental and business applications. One of the main objectives of the *ExtremeEarth* project is the development of distributed deep learning classification architectures tailored to the specific properties of Copernicus EO data using deep learning techniques and extreme geospatial analytics. The information provided by the Copernicus satellite data is then used for knowledge extraction, focusing explicitly on the following two use cases.

Polar Use Case. This use case exploits long time series of Sentinel-1 Synthetic-aperture radar (SAR) images to model processing architectures and algorithms to cope with the extreme analytics and big data issues affiliated with sea ice monitoring [3]. High quality, timely and reliable information about sea ice and iceberg conditions is significantly important to ensure that vessels can navigate efficiently and safely with reduced risk to the environment. This information is used by vessels in many fields, including cargo transport, fisheries, tourism, research vessels, resource exploration and extraction, destination shipping, and national coast guard vessels. The purpose of the *ExtremeEarth* Polar use case is to generate high-resolution ice charts from large volumes of hybrid Copernicus data.

Food Security Use Case. This use case aims at the assessment of high-resolution water availability maps for agricultural areas, allowing a new level of detail for wide-scale irrigation support for farmers [4] and agricultural stakeholders by combining Sentinel-2 multispectral images with crop growth modeling to provide water availability. Water availability is an important EO-based product that can support farmers in decision-making and irrigation information management. The deep learning architecture presented for this use case enables a processing chain that generates crop type and crop boundaries maps, using Sentinel-2 data for training.

The implementation of the Food Security use case draws on the following information: *(i)* crop type and leaf area index computed using Sentinel-2 images, *(ii)* biomass, water demand, soil moisture, snow storage, snow runoff, and groundwater computed using the proprietary land surface modeling software PROMET of VISTA[3], *(iii)* snowmelt from Sentinel-1 data, (*iv*) snow cover products from the Copernicus services, and (*v*) snow water equivalent from in-situ sensors[4]. Activities have been started in the two European catchments Danube and Douro, where irrigation has a significant role in Food Security.

End-User Services. The end-user services that *ExtremeEarth* targets are related to the previous two use cases. For example, as part of the Polar use case, deep learning techniques for automated sea ice mapping have been developed. National sea ice services around the world produce operational sea ice charts, often on a daily basis by manual and human-oriented interpretation. These usually contain information about sea ice concentration or a combination of sea ice concentration, ice edges, and ice type. For sea ice data analysis, SAR imaging plays a key role as the images acquired by satellite can continue to be collected during all weather conditions and through the polar night [5]. However, manual interpretation is a time-consuming task, and the ice information can be out of date with low coverage of the earth's surface. By taking advantage of a high volume of satellite SAR images and high-performance computing, the Polar use case aims to propose robust models considering advanced deep learning architectures to automate sea ice analyses with Arctic-wide coverage. For the Food Security use case, deep neural network architectures tailored to Copernicus data have been developed. These techniques can help the end-users to integrate different EO information and modeling using big data for watershed monitoring and irrigation recommendation of agricultural crops in drought-volatile areas by combining long time series of Sentinel-2 multispectral images with crop growth modeling to provide large scale as well as high-resolution water availability and water demand maps. Also, the deep learning architectures trained on Sentinel-2 data will generate crop type and crop boundaries maps.

[3] https://www.vista-geo.de/en/
[4] https://www.snowsense.de

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

Contributions. Our main contributions are centered around the two use cases of the *ExtremeEarth* software architecture. We summarize our main contributions below.

- We present the software architecture of *ExtremeEarth* that aims at the development of scalable deep learning and geospatial analytics techniques for processing and analyzing petabytes of Copernicus data.
- For the Polar use case, we present deep learning-based models for sea ice classification considering SAR imagery. The most important capability of these models is to classify sea ice and water distinctively in SAR images representing different geographic locations and timing.
- We extensively evaluate our deep learning-based models using our collected dataset to extract more relevant features when sea ice analysis is performed. Meanwhile, we cope with the overfitting problem by proposing a modified version of the VGG-16 model and the augmentation technique.
- For the Food Security use case, a machine learning-based approach has been defined to automatically extract a large training set for crops classification data leveraging publicly available crop type maps. The training set is obtained without the need for additional inputs.
- To generate the crop type and crop boundary maps, a Long Short Term Memory (LSTM)-based architecture is used for the classification of dense time series of Sentinel-2 images. This peculiar multi-temporal deep learning architecture is able to capture the temporal evolution of the phenological characteristics of the different crop types, thus producing accurate crops classification map.
- We compared the proposed deep learning model for the Food Security use case with state-of-the-art multitemporal models used for crop type classification.
- We present a set of linked data tools and methods that go beyond the state-of-the-art in the area of big linked geospatial data. These tools are used to transform the knowledge produced by the deep learning algorithms into linked data, interlink this knowledge with other available datasets in order to increase its value, store these datasets and allow the user to pose rich geospatial queries, and finally define a federation of such endpoints and pose geospatial queries over combining information from all these endpoints.

Outline. The rest of the paper is organized as follows. Section II discusses related work, and Section III provides a detailed overview of the high-level layered architecture of the *ExtremeEarth* platform architecture. Section IV explains how the TEPs applications work and provides a description of their architecture in the context of *ExtremeEarth*. Section V describes how the *ExtremeEarth* infrastructure has been architectured around Hopsworks to tackle large-scale machine learning and deep learning challenges in a way that takes the processing and management of earth data and knowledge beyond the current state-of-the-art. Linked geospatial data applications are presented in Section VI and Section VII describes in detail how different components of the *ExtremeEarth* platform fully integrate into the project's overall infrastructure to provide unified EO data and knowledge solution with Copernicus data. Section VIII highlights the deep learning classification models developed in the *ExtremeEarth* infrastructure. Finally, Section IX concludes our paper and suggests directions for future research work.

II. Related Work

Deep learning techniques have been used in analyzing satellite data in different areas. Highly scalable Artificial Intelligence techniques based on deep neural network architectures have recently been developed by companies such as Google and Facebook. However, similar architectures for satellite images, that can manage the extreme scale and characteristics of Copernicus data are missing today. The deep neural network architectures can classify effectively and efficiently multimedia images because they have been trained using extremely large benchmark datasets consisting of millions of images (e.g., ImageNet [6]) and have utilized the power of big data, cloud, and GPU technologies. Training datasets consisting of millions of data samples in the Copernicus context do not exist today and published deep learning architectures for Copernicus satellite images typically run on one GPU and do not take advantage of recent advances like distributed scale-out deep learning [7]. In addition to this, many techniques for knowledge discovery and data mining from satellite images and related geospatial data sets, as well as tools for linked geospatial data integration, querying, and analytics have been developed so far. However, none of these tools scales to the many petabytes of data, information, and knowledge present in the Copernicus context. For example, the state-of-the-art geospatial Resource Description Framework (RDF) store Strabon can only handle up to 100GBs of geospatial data [8]. To address these issues, we present the *ExtremeEarth* software architecture for Copernicus EO data by leveraging machine learning and deep learning-based techniques. We show how we go beyond the state-of-the-art by scaling to the petabytes of data using Hopsworks and demonstrate our big data technologies in two of the ESA TEPs: Polar and Food Security.

Polar Use Case. Over the past decade, probabilistic/statistical methods, classical machine learning methods, and deep learning-based methods have been developed for sea ice classification. For example, Moen et al. [9] introduced a Bayesian classification algorithm based on statistical and polarimetric properties for automatic segmentation of SAR sea ice scenes into a specified number of ice classes. Fors et al. [10] investigated the ability of various statistical and polarimetric SAR features to discriminate between sea ice types and their temporal consistency within a similar Bayesian framework. Yu et al. [11] presented a sea ice classification framework based on a projection matrix, which preserves spatial localities of multi-source image features from SAR and multispectral images. They obtained a set of fusion vectors that preserved the local similarities. The classification was then obtained in a sliding ensemble strategy.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

Considering the machine learning methods, Lit et al. [12] introduced a sea ice classification method based on the extraction of local binary patterns. They used a bagging principal component analysis (PCA) to generate hashing codes of the extracted features. Finally, these hashing codes were fed into an extreme learning machine for classification. Park et al. [13] extracted texture features from SAR images and trained a random forest classifier for sea ice classification. Their method classified a SAR scene into three generalized cover types, including ice-free water, integrated first-year ice, and old ice. Zhang et al. [14] introduced a conditional random fields classifier for sea ice classification using Sentinel-1 (S1) data. The literature is very limited when it comes to deep learning-based methods for sea ice classification. Castelluccio et al. [15] fine-tuned two existing architectures to perform semantic classification of remote sensing data, namely the CaffeNet and the GoogLeNet, and showed significant performance improvements. Wang et al. [16, 17] used deep ad-hoc Convolution Neural Networks (CNNs) for ice concentration estimation. Kruk et al. [18] used DenseNet [19] for finding ice concentration and ice types considering dual-polarization RADARSAT-2 SAR imagery by combining the HH and HV polarizations for the input samples. Han et al. [20] used a hyperspectral sea ice image classification method based on the spectral-spatial-joint features with deep learning. Gao et al. [21] proposed a deep fusion network for sea ice change detection based on SAR images. They exploited the complementary information among low, mid, and high-level feature representations.

Food Security. When considering the production of crop classification maps using remote sensing data, the temporal dimension plays an important role since the spectral and textural properties of the crop change according to the corresponding growth cycle [22]. In this context, the dense time series provided by Sentinel-2 at high spatial resolution allow one to accurately model the phenological trends of different cultivation [23]. Due to their spectral properties and several multitemporal radiometric indices that can be extracted (e.g., Normalized Vegetation Index or Enhanced Vegetation Index), several approaches have been defined to exploit such data for crop mapping [22] and monitoring [23, 24]. Although shallow models based on hand-crafted features can achieve good crop classification results [22], deep learning models outperformed such methods because of their enhanced capability of modeling the temporal information and extracting high-level abstract features [25]. Standard deep learning architectures such as CNNs [26] are not suited for this peculiar classification task, since they focus only on spatial and spectral information. By neglecting the temporal one, they fail in discriminating crop types indistinguishable when relying on single date remote sensing data. To effectively address the crop type mapping, two main categories of multitemporal deep learning architectures have been considered in the literature: (1) recurrence-based deep learning models, and (2) time convolutional-based deep learning models.

The most relevant property of recurrence-based deep learning models is the looped connections between neurons, which allows for modeling the sequential or temporal information using an internal state memory. In [27], the authors present the STAR recurrent neural network (StarRNN) composed of STAckable Recurrent cells which allows the accurate classification of temporal sequences of data. Gated recurrent unit (GRU)-based models have been successfully used to perform temporal analysis [28]. The two gates (i.e., reset gate and update gate) are used together with the hidden state to keep track of the relevant temporal information. However, one of the most used Recurrent Neural Networks (RNNs) for crop classification is the LSTM due to its capability of preserving the memory of a large number of past observations, making it suited for the problem under analysis [29, 30]. The effectiveness of these types of networks for crop classification of Sentinel-2 time series has been proved by a comparison with other several methods presented in [29].

In time convolutional deep learning models, a 1D convolution applied over the temporal dimension is used to extract information from the profiles [31, 32]. Such an approach has been integrated into different architectures and layers. Among them, we recall the network proposed in [32] where 3 convolutional layers, with equal-sized filters, are followed by dense and softmax layers. A similar structure is provided by the OmniscaleCNN [33], which is composed of 3 convolutional layers having different filter sizes followed by a global average pooling and a dense layer. In [31], the authors proposed an architecture composed of temporal convolutional layers and an inception module to model both the short temporal variations and seasonal trends. A comparison of RNNs and time convolutional neural networks is presented in [25] showing in general higher performances for the former groups.

III. Layered Architecture of the *ExtremeEarth* Software Infrastructure

This section presents a detailed overview of the high-level layered architecture of the *ExtremeEarth* software infrastructure.

The *ExtremeEarth* software architecture builds on the integration of ESA TEPs, DIAS, and Hopsworks. It enables high-performance scalable distributed data processing and deep learning on Copernicus EO data. As shown in Figure 1, the *ExtremeEarth* infrastructure consists of *four* layers with the TEPs in the *product layer*, Hopsworks in the *processing layer*, CREODIAS in the *data layer*, and Infrastructure as a Service (IaaS) in the *physical layer*.

A. Product layer

This layer provides a collaborative virtual work environment, through TEPs, that operates in the cloud and enables access to the products, tools, and relevant EO, and non-EO data. For more details about TEPs, we refer our readers to Section IV.

B. Processing layer

The processing layer provides the Hopsworks data-intensive Artificial Intelligence (AI) platform. Hopsworks is an

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

open-source platform that provides an execution environment for data science and data engineering, which supports the design, distributed training, and operation of both end-to-end machine learning and deep learning pipelines at scale. As part of *ExtremeEarth*, Hopsworks is deployed on CREODIAS. Hopsworks has certain unique features that make it appropriate for the development of deep learning algorithms for EO data. It provides tools to:

- Build end-to-end machine learning pipelines.
- Manage feature store, machine learning artifacts, and assets such as experiments and models.
- Build first-class support for popular open-source machine learning frameworks such as TensorFlow, PyTorch, Keras, and scikit-learn.
- Integrate with data science tools such as Jupyter notebooks, and infrastructure monitoring functionalities.

C. Data layer

The data layer offers a cloud-based platform that facilitates and standardizes access to EO data through a DIAS. It also provides centralized access to Copernicus data and information, as well as to processing tools. TEPs are installed and run on a DIAS infrastructure, which in the case of *ExtremeEarth* is CREODIAS.

CREODIAS platform. It is a cloud infrastructure adapted to the processing of big amounts of EO data, including an EO data storage cluster and a dedicated IaaS cloud infrastructure for the platform's users. The EO data repository contains Sentinel-1, 2, 3, and 5-P, Landsat-5, 7, 8, Envisat, and many Copernicus services data. The infrastructure and the services offered are optimized for use of EO data and support scientific, operational, and commercial applications. The main idea of the CREODIAS platform is based on combining a big data repository with customer accessible big processing power. CREODIAS offers several main categories of commercial services such as computing, storage, data, virtual networking, security, and monitoring services. It also offers three types of processing elements: virtual machines, dedicated servers, and containers.

D. Physical layer

The physical layer contains the cloud environment's compute, storage, networking resources, and hardware infrastructures. As explained below, Hopsworks users and developers use TEP to develop processing algorithms online, with all the processing done in the cloud, eliminating the need for people to download data to their local computers, and install the software. A TEP also hosts an online collaboration hub, where experts and users can exchange ideas, codevelop algorithms, and work together across the globe to refine and improve information extraction methods and develop new ones. The Hopsworks platform provides services to move the processing to where the data is and it is based on a cloud computing approach.

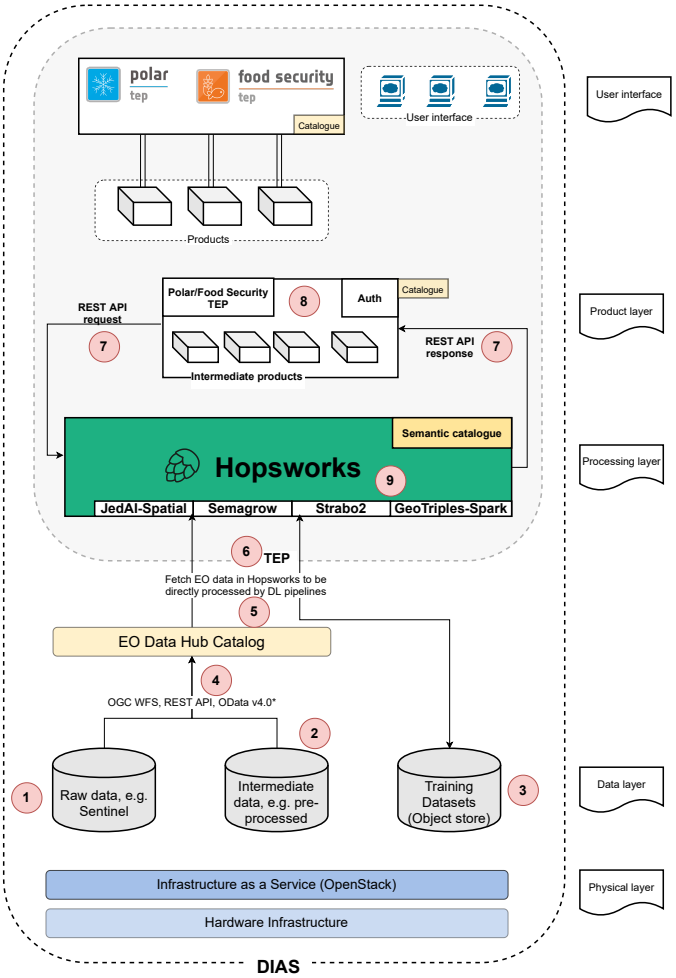


Fig. 1: Overview of *ExtremeEarth* platform infrastructure.

IV. TEPs in *ExtremeEarth*

Copernicus data today is freely available not only through the Copernicus Open Access Hub but also through the five DIAS where computing power is also available close to the data. Some related facilities of the EO ecosystem in Europe are the TEPs of ESA, where user communities can collaborate using a virtual workspace where EO data, non-EO data, tools, and computing power are available. TEPs provide users with access to pre-processed EO data and services which form the basis for creating training datasets. This pre-processed data is managed by TEPs but is stored in the CREODIAS infrastructure, for example in an object store. Today most of the TEPs run on a DIAS (e.g., the Food Security and the Polar TEPs run on CREODIAS). The CREODIAS infrastructure provides easy access and processing of petabytes of EO data in a scalable framework. As we can see in Figure 1, CREODIAS is the source of the raw EO data. It provides various protocols including OGC WFS, a REST API, and OData [34] as interfaces to the EO data. This helps users of Hopsworks to directly access raw EO data from their applications and services running on the Hopsworks platform. CREODIAS and its administrative tools enable TEPs to spawn and manage virtual machines and storage by using CloudFerro

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

which provides an OpenStack-based cloud platform to TEPs. Hopsworks is then installed on virtual machines within the TEP domain so that it can access compute and storage resources provided by the TEPs. In addition to this, TEPs also provide users with data to be served as input to the machine learning models from various sources. Such sources include the data provided by CREODIAS, higher-level satellite data products processed with TEP services, and external services that the TEP can connect to.

As explained in Section I, TEPs allow users to easily get access to satellite data and extract useful information out of them using analytic tools without the need to download any dataset. ESA's EO exploitation platforms initiative aims at creating an ecosystem of interconnected TEPs addressing seven main themes, i.e., Coastal, Forestry, Hydrology, Geohazards, Polar, Urban, and Food Security. In the context of *ExtremeEarth*, we focus only on how the Polar and Food Security TEPs are integrated into the *ExtremeEarth* infrastructure. We believe that the *ExtremeEarth* infrastructure can also be extended and used for deploying other TEPs as well. Even though the transfer and integration of Hopsworks to TEPs is quite new, we believe this is exploitable. As soon as Hopsworks is installed within the TEP environment, in principle it is possible to share the object store and data exchange. Since all TEPs have an API for data retrieval and passing of information by exploiting the APIs of Hopsworks, we argue that our infrastructure can also generally work with other TEPs. The use of the Hopsworks platform by the other TEPs can be accomplished in a similar way that the Polar and Food Security TEPs are explained in our paper.

The way the *ExtremeEarth* architecture's components interact with each other has been designed to be as agnostic as possible to the specific DIAS and TEPs involved. Hopsworks, as the key machine learning component of the architecture, is a cloud-agnostic platform that can be installed and operated on different cloud infrastructures. To achieve this, Hopsworks is shipped with its own installer software that is responsible for installing all the platform's components. In the *ExtremeEarth* project, we have demonstrated the successful deployment of Hopsworks on the CREODIAS cloud infrastructure which is based on OpenStack, an open standard cloud computing platform commonly used by other DIASes[5]. The communication between Hopsworks and the TEPs is done by using open standards and best practices, such as Representational state transfer (REST), which can easily be utilized to integrate additional TEPs in our proposed architecture. For more information about the users interaction with the *ExtremeEarth* architecture and flow of events, we refer the reader to Section VII.

A. Polar TEP

The Polar TEP is developed as part of ESA's activities to address the growth in volume and complexity of EO data available for the polar regions. EO is especially important in the polar regions at a time when climate change is having a profound impact, and excitement about new economic opportunities is driving increased attention and traffic, resulting in concerns about the state of the region's delicate ecosystems. Developing tools to model, understand, and monitor these changes is vitally important to better predict and mitigate the resulting global economic and environmental consequences. The Polar TEP provides new ways to exploit EO data for polar research scientists, industry, operational service providers, regional authorities, and in support of policy development. The demand for this functionality is expected to rise significantly, given plans for new polar-focused instruments, including anticipated Sentinel expansion missions. It facilitates the development of new processors, algorithms, and data products based on the data and knowledge resources available on the platform and enhances the collaboration between polar researchers. The Polar TEP also provides a complete working environment where users can access algorithms and data remotely, providing computing resources and tools that they might not otherwise have, avoiding the need to download and locally manage large volumes of data. This new way of working encourages wider exploitation of polar EO data.

The Polar TEP is based on the CREODIAS infrastructure, providing access to a wide range of EO data including the Copernicus Sentinel missions. It integrates data discovery and access for polar EO and complementary datasets, a scalable processor execution environment, analytical tools, and the possibility to share results and collaborate. Users are enabled to create, apply, and visualize their own data processors and products. The Polar TEP resources are accessed through a web portal. Its platform architecture is open, scalable, and infrastructure-independent to the maximum possible extent to allow easy expansion of the platform's capabilities. The high-level architecture of the Polar TEP is shown in Figure 3.

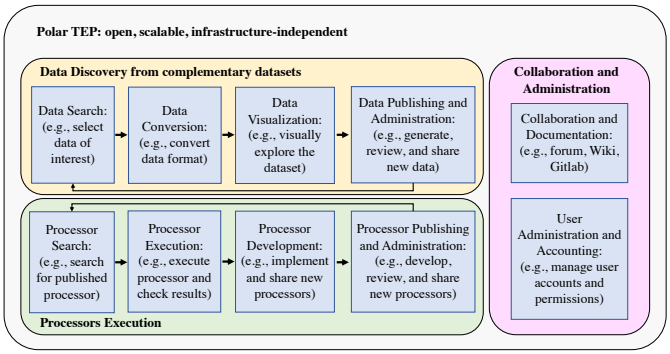


Fig. 2: Key functionality groups of the Polar TEP.

Polar TEP has mainly three different function groups that handles data discovery, processor execution, and collaboration and administration. The three key functionality groups of the Polar TEP, which are shown in Figure 2, are described as follows.

Data Functions. The first group of functions is concerned with searching, processing, exploring, and sharing data on the Polar TEP which include:

- *Data Search*. Users can search and retrieve polar EO and other data on the platform and from third-party

[5]https://www.copernicus.eu/en/access-data/dias

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

data providers. The User Access Portal contains a search interface with an interactive map to select areas of interest to search in and further parameters to narrow down the result list.

- *Data Conversion*. Users can convert data from and to well-known EO data formats using dedicated processors and libraries in the development environment.
- *Data Visualization*. Users can visualize EO data in order to explore it with a visualization tool integrated with the user access portal.
- *Data Publishing and Administration*. Users can generate new data using processors in the execution environment and publish and share this data. Resource administrators can check and review, publish and retract data to ensure compliance with the Polar TEP terms of service.

Processor Functions. The second group of functions is concerned with searching, executing, developing, and sharing processors on the Polar TEP.

- *Processor Search*. Users can search for processors already published on the platform to apply them to their own data. The corresponding components from the component view are the user access portal and execution environment.
- *Processor Execution*. Users can start the execution of existing processors in the execution environment through the user access portal, review the execution status based on feedback from the processor, and view the processor outputs.
- *Processor Development*. More experienced users can develop new processors on the Polar TEP, supported by development tools offered by the platform. They can review and adapt the algorithms published and shared by other platform users and benchmark and compare their developments. These activities are supported by documentation and community moderators.
- *Processor Publishing and Administration*. The developed processors from the previous section can be published and shared with the community. Community moderators can check and review, publish and retract processors to ensure compliance with the Polar TEP terms of service.

Collaborative and Administrative Functions.The last group of functions covers the collaborative and administrative functions:

- *Collaboration and Documentation*. Users and moderators can collaborate through the platform using forum, Wiki, and GitLab. They can review existing documentation, document their own algorithms and processors, share research results, and compare benchmarks.
- *Administration and Accounting*. Users have to accept the Polar TEP terms and conditions when logging in for the first time. User and resource administrators can activate and retire user accounts, assign permissions, provide resources, review resource usage, and use these tools to ensure compliance with the Polar TEP terms of service. They can also communicate with users through e-mail or through collaboration tools.

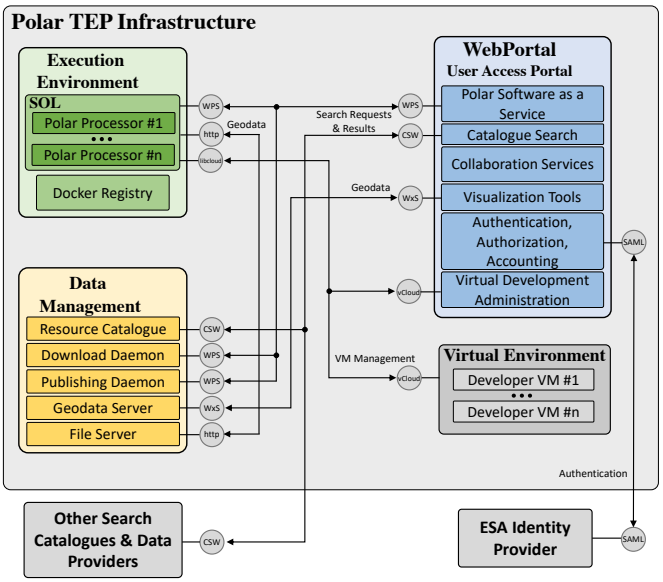


Fig. 3: High-level architecture of the Polar TEP.

Figure 3 summarizes the high-level architecture of the Polar TEP. Polar TEP uses open standard interfaces (WPS, OpenSearch/CSW, SAML) for communication between different components.

B. Food Security TEP

The Food Security TEP[6] provides a platform for the extraction of information from EO data for services in the food security sector mainly in Europe and Africa. Thereby it targets to foster smart, data-intensive agriculture and aquaculture applications in the scientific, private and public domain. The Food Security TEP user community is interested in applications ranging from small-scale farming advice up to regional and national analysis of crop development. Users accessing the platform are from a wide range of fields including public science and app developers, the finance and insurance sectors as well as local and national administration and international agencies.

Current and past projects in Europe and Africa using the Food Security TEP showed that a dedicated cloud-based platform for EO applications leads to easier cooperation between international project partners. The Food Security TEP Platform-as-a-Service (PaaS) increases the available information for agricultural and aquacultural management even in regions with relatively poor infrastructure, while securing data ownership for all partners [35].

The technical infrastructure is a web-based PaaS developed by CGI Italy[7], which leverages the most advanced cloud computing technologies. Food Security TEP provides easy access to all Copernicus data and a wealth of additional data sources. It facilitates the implementation of specific services, by adding new processing algorithms and allowing their execution, monitoring, and maintenance. The main point of

[6] https://foodsecurity-tep.net
[7] https://www.cgi.com/italy/en

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

access to the platform is the Open Expert Interface providing the main functionalities of the platform and access to a variety of tools and data sets. The platform allows data visualization also on mobile devices and the provision of customized products and services to selected users.

With regard to key functionalities, the Food Security TEP in principle provides the same functions as shown for the Polar TEP in Figure 2 with slight differences:

- The only missing function compared to Figure 2 is *Data Conversion* per se, which is not covered by explicit platform services. Developers need to exploit well-known, open libraries to create this function.
- Food Security TEP is currently enhanced to provide an additional function, namely the *Execution of Remote Processors* in other cloud environments provided as Web Processing Service (WPS)-based application deployment and execution services.

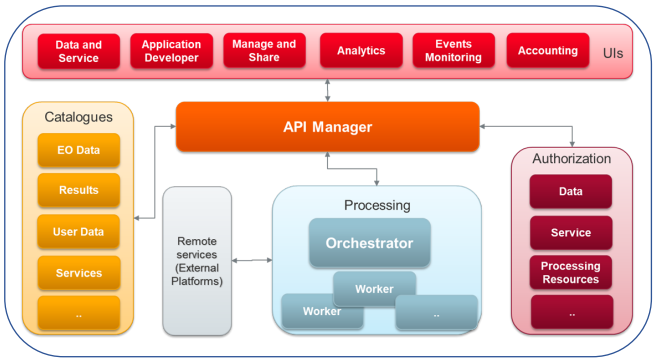


Fig. 4: High-level architecture of the Food Security TEP.

Figure 4 summarizes the high-level architecture of the Food Security TEP with its API Manager at the core. This allows users to access certain platform functionalities not only through the GUIs, but also through the Food Security TEP public REST API. Besides user interaction, the UIs component provides information on the accounting of resource consumption based on a virtual platform currency (TEP coin). As Food Security TEP follows the principle of a collaborative platform, Authorization to nearly all contents of the platform are managed by the respective components. The platform offers scaling capabilities by allocating and releasing Processing resources based on actual consumption and user configuration. In addition, the platform federates other platform services exploiting standard interfaces (WPS). Finally, all data, processing jobs, and services available on the platform are managed by the Food Security TEP's Catalogues, which allows sharing and publication based on flexible user grouping.

V. Hopsworks

This section describes how the *ExtremeEarth* infrastructure has been architectured around Hopsworks to tackle large-scale machine learning and deep learning challenges and take processing and management of earth data analytics and knowledge beyond the current state-of-the-art.

Hopsworks provides a horizontally scalable platform for building end-to-end machine learning and deep learning pipelines with GPUs and SDKs for hyper-parameter tuning and elastic model serving. It offers a convenient collaborative environment for developing machine learning models and putting them into production. For example, a user can import a specific dataset in a project, transform it into RDF[8] and securely share the results with other projects or users, who then can perform further processing, such as interlinking or querying. Hopsworks supports role-based access control with dynamic roles for users accessing and processing such datasets, which enables data owners to securely give access to datasets in a project, given that the data cannot be exported outside the project or cross-linked with other data sources outside the project. This security model is built on TLS certificates and enables both authorization and authentication in the underlying communication protocol. In order to perform these tasks, users and developers only need to interact through the self-service user interface of the platform, which offers ready-to-use deployments of popular cloud data storage and processing tools like Apache Hive, Apache Spark, and Apache Kafka as well as popular machine learning frameworks such as Tensorflow, PyTorch and scikit-learn. Also, using this interface the users can collaborate in order to specify and execute their data pipeline in a Jupyter Notebook and effortlessly monitor the execution progress and inspect the results.

Machine Learning Pipeline. Hopsworks provides SDKs in Python and Scala to help developers work with its provided set of services that support the full machine learning and deep learning lifecycle (shown Figure 5), including:

- Data management with HopsFS, Hive, Apache Kafka, and Elasticsearch.
- Training machine learning and deep learning models on both GPUs and CPUs, including distributed training on GPUs.
- Serving of models in production using Kubernetes, with Hopsworks providing authorized, audited access to scale-out models on TensorFlowServing, SparkML, or scikit-learn.
- Model management and monitoring with Apache Spark Streaming application analyzing model usage in near-realtime.

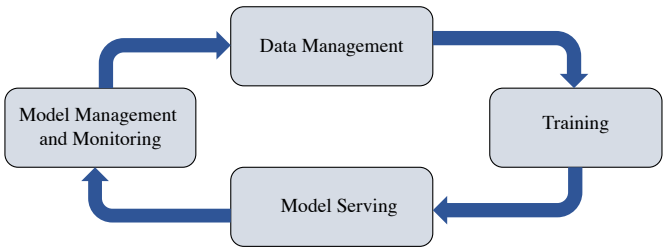


Fig. 5: Full machine learning and deep learning lifecycle.

Data Ingestion. The first step in building a scalable deep learning pipeline is to locate the sources where the input data

[8]https://www.w3.org/RDF/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

reside. Then, processes need to be established that ingest data from the sources into the platform where the deep learning pipeline runs. These sources can be quite diverse in the format they use to store data and the protocols they implement to deliver data over to other systems. Such sources include raw data which can come from devices connected to the Internet of Things (IoT), images from satellites, structured data from data warehouses, financial transactions from real-time systems, social media, etc. Figure 6 illustrates where in the deep learning pipeline the external systems reside. In the context of *ExtremeEarth*, we show the different ways in which Hopsworks has been extended to make satellite imagery data easily ingested in the platform for further processing.

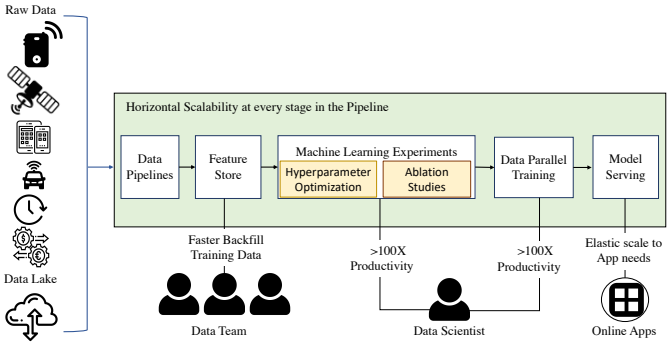


Fig. 6: Data Ingestion sources for a deep learning pipeline.

Feature Store. Feature engineering can be described as the process with which domain knowledge on ingested data is applied, in order to create features that are used in further training stages of the deep learning pipeline. With the continuous growth in input data and increased complexity of deep learning pipelines, the need for a framework that facilitates features engineering and reduces the complexity of managing features becomes evident. To improve the management of curated feature data, Hopsworks has been extended with a new framework called *Feature Store*. The Feature Store acts as the central management layer for curated feature data in an organization and it serves as the interface between data engineering and data science teams. As depicted in Figure 7, the motivation of feature engineering is to generate reusable features that can be shared across different teams in an organization and can facilitate the development of new machine learning models. The advantages of the Feature Store include reusing features across pipelines, feature discoverability with free-text search across an organization's feature data, applying software engineering principles onto machine learning features with versioning, documentation, and access-control, time-travel by fetching past feature data that were used for training particular model. It also brings scalability in terms of being able to manage multiple petabytes of feature datasets, so that data scientists can gather useful insights regarding data distribution and correlation.

To achieve all the aforementioned properties, the *Feature Store* is implemented on scalable and fault-tolerant services. Offline data is stored in Apache Hive [36], a scalable data warehouse built on top of Apache Hadoop, and online data

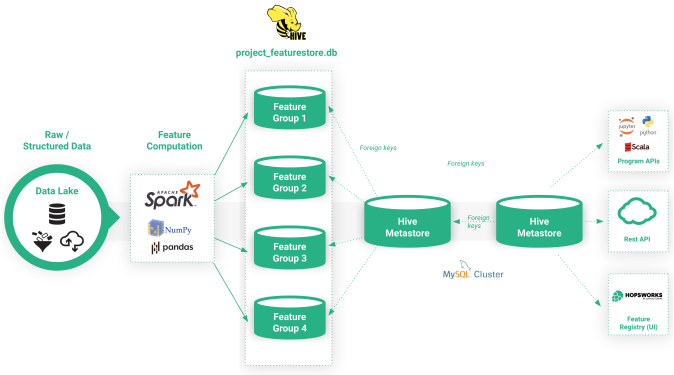


Fig. 7: The Feature Store as the link between Feature Engineering and Training.

are stored in MySQL Cluster (NDB). Offline features can be used for training and are used mostly in batch-oriented use cases where past feature data can be fetched or huge volumes of feature data can be analyzed to generate statistics. Online features need to be accessible in real-time for pipelines that need to get data at prediction time. Besides storing data, the *Feature Store* utilizes the Spark integration in Hopsworks to compute and analyze features. Figure 8 shows the main components of the Hopsworks Feature Store.

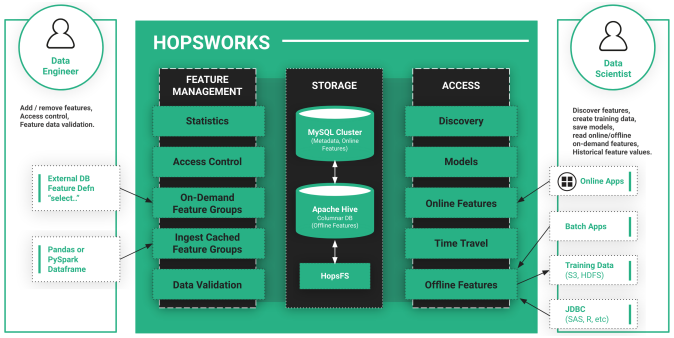


Fig. 8: Hopsworks Feature Store Architecture.

Training. Building machine learning models is an iterative process as models need to be build based either on new training data or feedback from model validation. As such, the task of doing distributed training on large datasets in order to build and deploy models is non-trivial. The process of using training samples, finding the best hyperparameters, and building a model can be referred to as an experiment. The experiments framework in Hopsworks has been extended with Maggy [37], an open-source PySpark-based framework for asynchronous hyperparameter tuning, ablation studies, and distributed training of machine learning models. The programming model of Maggy is based on the distribution oblivious training function [38], in which the users factor out dataset creation and model creation code into their own separate functions, and pass them as parameters to the training function. The resulting code can then be used to launch different experiments (hyperparameter tuning, ablation studies, or training) either in parallel or a single-host fashion, without requiring any further changes to the code.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

For hyperparameter tuning, Maggy currently ships with implementations of Random Search, and Bayesian Optimization (with Tree Parzen Estimators [39] and Gaussian Processes [40]) as optimizers, as well as HyperBand [41] and ASHA [42], and a median stopping rule for early stopping. However, Maggy provides base classes for both the optimizers and the early stopping rules as part of a developer API to make it extensible, so that users can implement their own optimizer or early stopping rules. To run a hyperparameter tuning experiment, the users have to define a search space for their target hyperparameters and specify the optimization algorithm and early stopping rule. For ablation studies, Maggy ships with a Leave-One-Component-Out (LOCO) ablator, which removes one component out of the training process at a time. A component can be one or a group of dataset features, one or a group of network architecture layers, as well as "modules" of network architecture, such as Inception-v3 inception modules [43]. Similar to hyperparameter tuning experiments, here, instead of a search space, the users have to define a list of the components that are to be ablated (i.e., excluded), and specify the ablation policy (e.g., LOCO). Maggy would then automatically create the corresponding trials for the ablation study and run them in parallel.

Model Serving and Monitoring. The last stage of the deep learning workflow is threefold: *export*, *serve* for inference, and *monitor* the model. Once a model is developed and exported using the stages in the deep learning pipeline, it needs to be served so that external clients can use it for inference. After the model is deployed, its performance also needs to be monitored in real-time so that users can decide when it would be the best time to trigger the training stage. Hopsworks has been extended to provide support for TensorFlow Serving and Flask model servers to deploy models trained with Tensorflow or any other Python library such as scikit-learn. Hopsworks has been extended with a Kubernetes cluster on which docker containers are deployed that run TensorFlow serving and a Flask server. Additionally, KFServing has been installed in the Kubernetes cluster to deploy more complex inference pipelines composed of the model server, an inference logger container, and a transformer component. KFServing provides additional features such as pre- and post-processing of inputs and outputs during model inference, multi-model serving, scale-to-zero support, multi-armed bandits, or A/B testing. Users have the option to choose whether to use KFServing or the docker containers provided by Hopsworks. Moreover, users can select the minimum number of instances for a model serving in runtime, therefore Hopsworks provides users with the important property of elasticity.

In the context of *ExtremeEarth*, inference requests are proxied through the Hopsworks REST API to provide secure multi-tenant access to Hopsworks where role-based access control is done based on projects. Project members are allowed to submit requests only to the models being served from within their projects. Inference requests are logged in Apache Kafka [44] which is provided as a multi-tenant service in Hopsworks. The overall architecture of model serving and monitoring in Hopsworks is depicted in Figure 9. The snippet of codes presented in Listing 1-3 shows end-to-end examples of model serving on Hopsworks using TensorFlow.

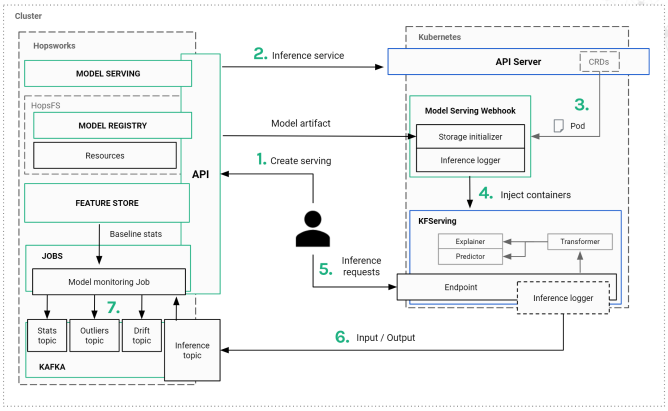


Fig. 9: Model serving and monitoring architecture [45].

```
from hops import model
from hops.model import Metric
MODEL_NAME="mnist"
EVALUATION_METRIC="accuracy"

best_model = model.get_best_model(MODEL_NAME,
                          EVALUATION_METRIC, Metric.MAX)

print('Model name: ' + best_model['name'])
print('Model version: ' + str(best_model['version']))
print(best_model['metrics'])
```

Listing 1: Querying the model repository for best MNIST model.

```
from hops import serving

# Create serving
# optionally, add the kfserving flag to deploy the model
# server using this serving tool. Iif not specified,
# it is deployed using the serving tool by default
# on the current Hopsworks version (docker or kubernetes)
serving_name = MODEL_NAME
model_path="/Models/" + best_model['name']
response = serving.create_or_update(serving_name,
                          model_path,
                          model_version=best_model['version'],
                          model_server="TENSORFLOW_SERVING",
                          kfserving=False)

# List all available servings in the project
for s in serving.get_all():
    print(s.name)
```

Listing 2: Creating model serving of exported model.

```
import numpy as np
import json

TOPIC_NAME = serving.get_kafka_topic(serving_name)
NUM_FEATURES=784

for i in range(20):
    data = {
        "signature_name": "serving_default",
        "instances": [np.random.rand(NUM_FEATURES).tolist()]
            }
    response = serving.make_inference_request(
        serving_name, data)
```

Listing 3: Sending prediction requests to the served model using Hopsworks REST API.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

Resource Management. When developers use TensorFlow/Keras/PyTorch to train deep neural networks in Hopsworks, the latter uses PySpark to transparently distribute the python code to containers with GPUs. Hopsworks enables Automated machine learning (AutoML), automated search for good neural network architectures and hyperparameters by running parallel experiments on different combinations of hyperparameters and model architectures.

In PySpark, Hopsworks runs a different experiment on each Executor. Not all of the experiments will finish at the same time, some experiments may finish early, some later. Moreover, GPUs cannot currently be shared by concurrent applications. Population-based approaches for AutoML, typically proceed in stages or iterations, meaning all experiments wait for other experiments to finish, resulting in idle GPU time. That is, GPUs lie idle waiting for other experiments to finish. As such, there is the problem of how to free up the GPUs as soon as its experiment is finished. Hopsworks leverages dynamic executors in PySpark/YARN to free up the GPUs attached to an Executor immediately if it sits idle waiting for other experiments to finish.

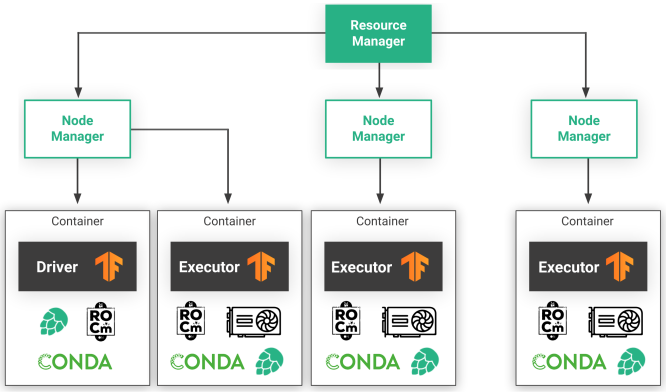


Fig. 10: Spark/TensorFlow on Hops.

As Figure 10 shows, each Spark Executor runs a local TensorFlow process. Hopsworks also supports managing a project's Python environment and dependencies across the cluster using Conda. Hopsworks supports the creation of Projects, and each Project has its own conda environment that lives inside a Docker container. When a PySpark job is launched, it pulls the Docker container of the project the job belongs to. This way, users can install whatever libraries they like using conda and pip, and then use them directly inside Spark Executors. It makes programming PySpark one step closer to the single-host experience of programming Python. Hopsworks also supports Jupyter and the SparkMagic kernel for running PySpark jobs.

Storage. Hops is a next-generation distribution of Apache Hadoop, with a heavily adapted implementation of Hadoop Filesystem (HDFS). HopsFS is a new implementation of the HDFS, that supports multiple stateless NameNodes, where the metadata is stored in an in-memory distributed database. HopsFS enables NameNode metadata to be both strongly consistent, customized and analyzed. HopsFS replaces HDFS's Active-Standby Replication architecture with a set of stateless NameNodes backed by an in-memory, shared-nothing NewSQL database. HopsFS provides the database abstraction layer (DAL)-API as an abstraction layer over the database and implements a leader election protocol using the database. This means that HopsFS no longer needs services like quorum journal nodes, Zookeeper, and the Snapshot server required by Apache HDFS. HopsFS has been proven to scale to millions of operations per second [46].

Storing Small Files. Big data developers may need to store and access many datasets as image files, stored in a distributed file system. However, according to Uber [36], it is complex and costly to implement multiple round-trips to the filesystem at a large scale especially using modern distributed file systems such as HDFS and object stores such as S3. Uber's proposed solution is to pack image files into larger Apache Parquet [47] files. HopsFS solves this problem by transparently integrating Non-Volatile Memory Express (NVMe) disks into its HDFS-compatible file system [48]. Modern distributed file systems such as HDFS and S3 are designed around large blocks optimized to overcome slow random I/O on disks, while new NVMe hardware supports fast random disk I/O and potentially small blocks sizes. However, as NVMe disks are still expensive, it would be prohibitively expensive to store tens of terabytes or petabyte-sized datasets on only NVMe hardware. The hybrid solution in Hops involves storing files smaller than a configurable threshold (default: 64KB, but scales up to around 1MB) on NVMe disks in its metadata layer. On top of this, files under a smaller threshold, typically 1KB, are replicated in-memory in the metadata layer due to their minimal overhead.

Storage Cost Reduction. HopsFS is extended with two ways for reducing storage costs that can occur due to the deluge of data provided by the Copernicus programme. Firstly, HopsFS is extended to provide native support for cloud-native storage solutions and protocols such as AWS S3 and Azure ADLS. Effectively, HopsFS is able to store its data that is split into blocks directly to the aforementioned storage services. This alleviates the need for acquiring additional local storage in the form of disks and also provides a more intuitive user and developer experience when working with the Hopsworks data storage layer. AWS S3 is of particular for the *ExtremeEarth* project as it is the protocol utilized by CREODIAS to provide the Copernicus data[9], for example, data retrieved from Sentinel satellites to the users and therefore Hopsworks.

Secondly, HopsFS provides erasure-coding functionality in order to decrease the storage capacity that is required for EO data without the loss of high-availability. HopsFS offers a powerful, on a per-file basis configurable, erasure coding API. Codes can be freely configured and different configurations can be applied to different files. Given that Hops monitors the erasure-coded files directly in the NameNode, maximum control over encoded files is guaranteed. Apache HDFS stores three copies of data to provide high-availability. So, for example, one petabyte of data actually requires three petabytes of storage. For many organizations, this results in enormous

[9]https://creodias.eu/faq-s3

storage costs. HopsFS also supports erasure-coding to reduce the storage required by 44% compared to HDFS, while still providing high-availability for the data. The erasure-coding API offers the ability to request the encoding of a file while being created. This has the benefit that file blocks can initially be placed in a way that meets placement constraints for erasure-coded files without needing to rewrite them during the encoding process. The actual encoding process will take place asynchronously on the cluster.

The erasure-coding API also offers the ability to request the encoding for existing files. A replication factor to be applied after successfully encoding the file can be supplied together with the desired codec. Here, the actual encoding process will take place asynchronously on the cluster too. Every path in HDFS has a replication factor. If erasure-coding is enabled for that path, then erasure-coding overrides the replication factor settings. However, if erasure-coding is disabled for a path, the erasure-coding API allows the default replication factor in HDFS for that specific path to take effect. A replication factor can be supplied and is guaranteed to be reached before deleting any parity information. Deletion of encoded files does not require any special care since the system automatically takes care of the process.

VI. Linked Geospatial Data

A. Linked Data Tools

The following linked data tools are integrated with Hopsworks.

Semantic Catalogue for EO data. In order to extend the capabilities for EO data discovery, and access utilizing information and knowledge extracted from Copernicus data, we have designed a semantic catalogue for the Polar and Food Security ontologies [49]. Based on these ontologies we take the results of the machine learning algorithms and translate them to the RDF model using the tool GeoTriples-Spark and store this information to a Strabo2 SPARQL endpoint. Finally, this Strabo2 SPARQL endpoint is deployed in Hopsworks and can be accessed by the Polar and Food Security TEPs through an HTTP-based RESTful API according to the W3C standard[10]. In both developed ontologies, all the extracted features are linked to the main observation class, which is linked to the respective satellite image it belongs to with the *hasImageID* property. This allows us to use the semantic catalogue in *two* ways:

- Access the Strabo2 SPARQL endpoint through its URL or HTTP requests and pose GeoSPARQL queries to retrieve satellite images based on the semantic information that the ontologies provide.
- Use the existing CREODIAS semantic search API[11] and access the Strabo2 SPARQL endpoint using SPARQL SERVICE[12] for federation. This method allows us to enhance the existing catalogue API with the semantic search features provided by the Strabo2 endpoint. The linking of information in the two SPARQL endpoints is achieved through the satellite image ID string.

GeoTriples-Spark. It is an extension of GeoTriples that enables the transformation of big geospatial data into RDF graphs. The implementation is based on Apache Spark and it currently supports the transformation of CSV, GeoJSON, and ESRI Shapefiles. The initial data sources first need to be loaded into HopsFS, and then GeoTriples-Spark runs as a Spark job handled by HopsYARN, without directly interacting with any other components on Hopsworks.

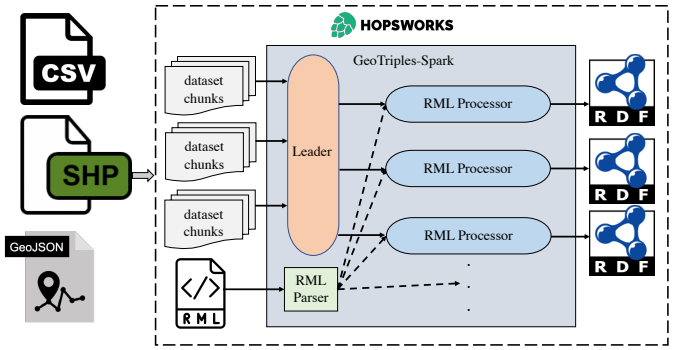


Fig. 11: GeoTriples-Spark Architecture.

As we can see in Figure 11, the GeoTriples-Spark loads the data from HopsFS as multiple partitions, where for each one of them a Spark Task is spawn, responsible for transforming the data of the partition. In each task, an RML Processor is employed which maps the input data into RDF triples by applying certain transformation rules. Those transformation rules, which are defined by the user in an RML mapping file, are functions that define how the input data will be mapped into RDF triples. This RML file is loaded by the driver which extracts the transformation rules and broadcasts them to all executors. Except for the broadcast of the transformation rules, there is no need for further data shuffling as the transformation of each partition does not affect the transformation of the rest. Moreover, regarding the execution of ESRI Shapefiles, we use the library Apache Sedona[13] (formerly known as GeoSpark) which is an in-memory cluster computing framework that extends Spark with spatial computations, like spatial indexing.

JedAI-spatial. It discovers spatial relations between two different input datasets of linked geospatial data. As shown in Figure 12, JedAI-spatial runs as a Spark job handled by HopsYARN, without directly interacting with any other component on Hopsworks.

Initially, the two input datasets to be interlinked, the source and the target one, are loaded as Resilient Distributed Datasets (RDDs) that are spatially partitioned according to Apache Sedona's Quad-Tree. All RDDs are partitioned using the same partitioner and thus, the topologically close geometries belong to partitions with the same partition id. The source and target RDDs with the same partition id are then merged such that each partition contains all geometries from both datasets that lie within its area. This way, we ensure that all geometries that

[10] https://www.w3.org/TR/sparql11-http-rdf-update/
[11] https://browser.creodias.eu/
[12] https://www.w3.org/TR/sparql11-federated-query/
[13] http://sedona.apache.org/

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

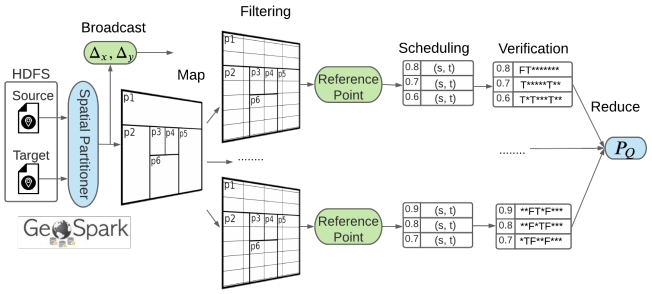


Fig. 12: JedAI spark on top of Hopsworks.

are likely to satisfy a topological relation coexist in the same partitions. Subsequently, each Executor receives one of these partitions as input, during the Map phase. To apply Filtering, it indexes the source geometries and for each target geometry t it estimates the tiles that intersect its Minimum Bounding Rectangle. Using the index, it retrieves the distinct source geometries in these tiles. Note that every geometry that crosses the borders between two partitions is added to both during spatial partitioning. To avoid the resulting redundancy, the reference point technique is used, i.e., JedAI-spatial ensures that every pair is verified only in the partition that contains the top left corner of their intersection. Note that the granularity of Filtering requires some simple computations by the Driver, which broadcasts the results (Δx, Δy) to the Executors. Finally, each Executor performs the necessary processing, such as Scheduling, and applies Verification to estimate the topological relations of every target geometry with all source geometries that pass the Filtering step. After examining the resulting Intersection Matrices, the set of qualifying pairs of geometries PQ is aggregated during Reduce phase.

Strabo2. It performs GeoSPARQL query answering on RDF datasets stored in Hopsworks. Strabo2 has 2 modules. The first module is responsible for data import and storage. This module is executed as a Spark job handled by HopsYARN, that reads the RDF datasets from HopsFS and creates tables in the HIVE database of the corresponding Hopsworks project, according to an optimized schema, stored in compressed PARQUET format, as shown in the bottom part of Figure 13. The second module is responsible for GeoSPARQL query execution. This module receives a GeoSPARQL query, and it performs a cost-based translation into Spark SQL, enhanced with spatial operators offered by the Apache Sedona framework. This module can also be executed as a Spark job, using as an input the GeoSPARQL query either directly, or by executing all GeoSPARQL queries contained in files in a given directory in HopsFS passed as an argument. Though this method of execution is simple and can be directly submitted to HopsYARN for management, it does not offer interactivity. For this reason, we offer a second way to execute this module in the form of an endpoint, where it accepts requests and returns results in an interactive manner. For example, using this mode of execution, Strabo2 can act as an endpoint for the Semagrow federation engine. In order to directly use the Spark installation running in Hopsworks, this module must start a Spark session in cluster mode, and keep it open, as it accepts and executes the GeoSPARQL queries. Unfortunately, creating Spark sessions in cluster mode cannot be integrated with the HopsYARN resource manager. For this reason, in order to achieve full integration with Hospworks, we are developing a solution that uses Apache Livy for maintaining the Spark session.

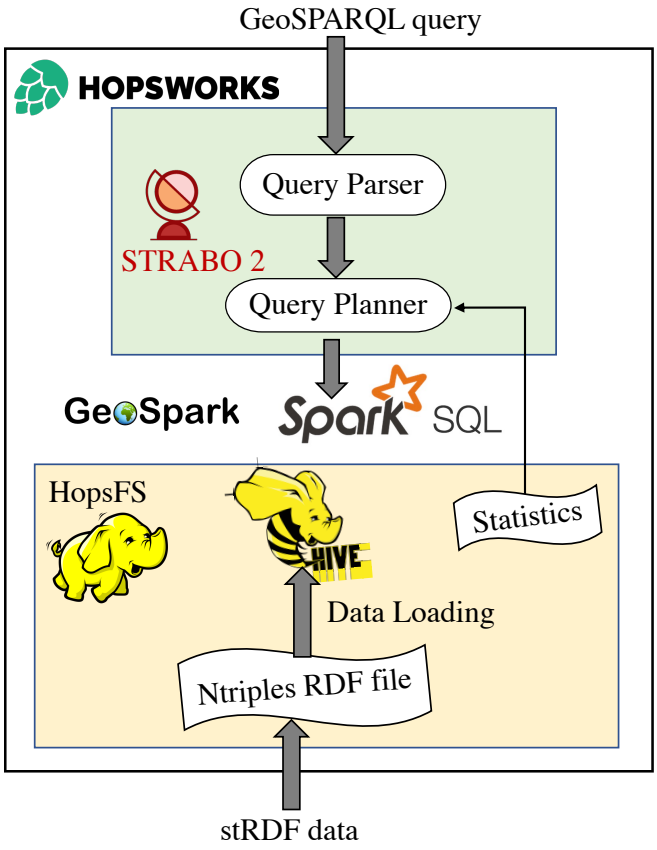


Fig. 13: Strabo2 on Hopsworks.

Semagrow. The GeoSPARQL query federation engine Semagrow [50] is used for integrating several heterogeneous big linked geospatial data sources under a single GeoSPARQL endpoint. Semagrow receives a GeoSPARQL query through its endpoint, decomposes the query, performs the necessary GeoSPARQL queries in the federated endpoints, combines the results accordingly, and presents the result to the user. The implementation of Semagrow is based in the rdf4j framework, including its geospatial support which has been extended with optimization techniques especially for geospatial linked data. In the context of integration with Hopsworks, we have used Semagrow to combine big linked geospatial data served by Strabo2 with external geospatial resources (Figure 14). Besides data integration between data served by Hopsworks and external data sources, this also achieves providing access to the data served by Hopsworks that completely implements the GeoSPARQL specification, as Semagrow is a complete GeoSPARQL implementation and automatically fills in any functionality missing in the access methods of the data sources it federates.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

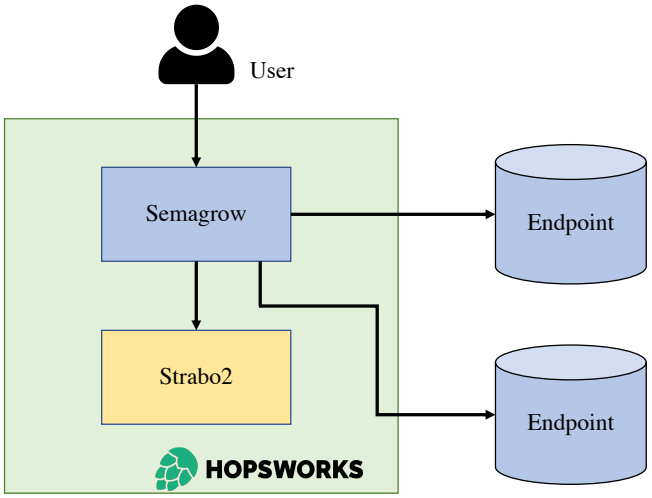


Fig. 14: Semagrow on Hopsworks.

B. Experimental Evaluation of Linked Geospatial Data Tools

Let us now discuss the performance of three of the linked geospatial data tools presented: GeoTriples-Spark, JedAI-spatial and Strabo2.

Table II shows the performance of GeoTriples-Spark in the transformation of big geospatial data into RDF. The input data consists of extracts of the OpenStreetMap[14] project in the form of multiple ESRI shapefiles. For the transformation, we used up to 30 Executors equipped with 2 cores each and 2GB of memory. In the end, we managed to transform 1TB of input geometries in 34 minutes. An important issue that arises with very large input files is the size of the output files, as this is significantly bigger than the initial input, due to the nature of RDF triples. To solve this issue, we are streaming the produced triples directly in Strabo2, instead of writing them on the disk. This will facilitate access to the produced graphs and will enable us to pose GeoSPARQL queries efficiently.

To evaluate the performance of Geospatial Interlinking, we used JedAI-spatial to link a source dataset of 72.3 million geometries with a target one containing 114.8 million geometries (D_6 in [51]). Restricting the maximum number of examined geometry pairs to 20 million, we achieved the performance in Table I. In all cases, the entire process is completed within 20 minutes, as JedAI-spatial fully exploits the parallelization capabilities of the *ExtremeEarth* platform. The original configuration of JedAI-spatial [51] corresponds to CF, JS, and χ^2, which select the geometry pairs to be investigated based on the tiles they share in the grid index. That is, they favor the geometry pairs with the highest co-occurrence frequency in the tiles of the index. We significantly improve their performance by introducing a new pair selector, called MBRO, which promotes the pairs with the largest overlap of their minimum bounding rectangles. MBRO raises both Recall and Precision of the best original configuration, χ^2, by 18%. Given that MBRO considers evidence that is complementary to that of the original pair selectors, we can combine them into hybrid selectors, namely CF·MBRO, JS·MBRO, and χ^2·MBRO. In all cases, the performance of the original selector raises to a significant extent. Most importantly, χ^2·MBRO improves the performance of MBRO by another 2%, thus corresponding to the best overall configuration. Note also that in most cases, JedAI-spatial improves the performance of the baseline random selector by three to two times.

Regarding the experimental evaluation of Strabo2, we have performed experiments with both real-world and synthetic datasets, with sizes that exceed the capabilities of the centralized Strabon system. Specifically, the real-world dataset used is the scalability dataset of the Geographica2 benchmark [8], which is a fusion of Open Street Maps with Corine Land Cover. The original datasets have been extended to more countries, now consisting of 1.08 billion triples (198 GB in plain text). For this dataset, we have performed the three queries of the benchmark. These are queries of varying selectivity. One query contains a spatial filter over the whole dataset, and the other two are heavy spatial join queries. Execution times in hops with 64 executors for these three queries are 30, 407, and 100 seconds respectively.

It is worth mentioning that centralized Strabon cannot import datasets of this size. The synthetic dataset used also comes from the Geographica2 benchmark and it contains a minimal ontology that follows a general version of the schema of Open Street Maps. It contains 36 queries of spatial selections and spatial joins with different selectivities and different spatial predicates (intersects, within, touches). We have successfully executed the query set in Hops for a dataset with a scale factor of 12228, which contains 2.35 billion triples (0.46 TB in plain text-scale factor 12228). In order to examine the scalability of Strabo2 with respect to dataset size, we used different scale factors of the scalability benchmark on a setting with 64 executors in Hops, where each executor contains 2 cores and 8 GB of memory. The average execution time for the 36 queries in each dataset is presented in Figure 15.

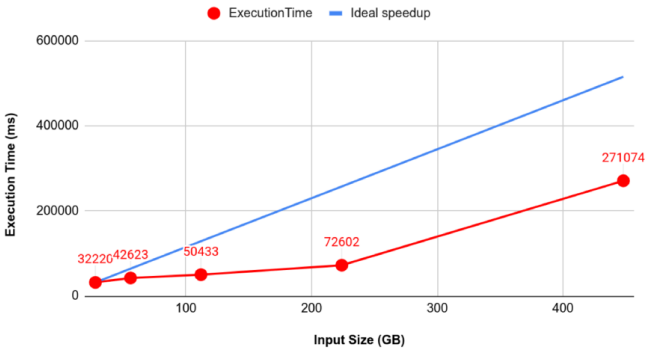


Fig. 15: Strabo2 scalability with varying dataset size (64 executors)

VII. Integration of the *ExtremeEarth* platform components and flow of events

In this section, we define the high-level architecture and semantics of the integration of Hopsworks with the Polar and Food Security TEPs in *ExtremeEarth*. The APIs used for the

[14] https://www.openstreetmap.org/#map=16/37.9745/23.7404

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

TABLE I: Performance of JedAI-spatial when interlinking 72.3M with 114.8M geometries, while considering up to 20M pairs.

	Random Selector	JedAI-spatial						
		CF	JS	χ^2	MBRO	CF·MBRO	JS·MBRO	χ^2·MBRO
Recall	0.106	0.131	0.376	0.387	0.456	0.454	0.408	0.465
Precision	0.006	0.007	0.019	0.020	0.024	0.024	0.021	0.024
Run-time (min)	-	18.6	19.8	18.5	20.6	22.2	19.7	19.8

TABLE II: Performance of GeoTriples-Spark for the transformation of big geospatial data.

Input Size (GB)	Overall Time (minutes)	Output Size (GB)
100	4.3	428
250	9.9	1,068
500	17.6	∼2,500
1000	34.3	∼5,100

integration of the *ExtremeEarth* components via the inter-layer interfaces of the software platform are also described here.

A. ExtremeEarth *infrastructure integration of components*

As shown in Figure 1, the *ExtremeEarth* infrastructure consists of *four* layers with TEPs in the *product layer*, Hopsworks in the *processing layer*, CREODIAS in the *data layer*, and IaaS in the *physical layer*. Figure 1 contains the numbered labels highlighting the different integration points of the DIAS, TEPs, and Hopsworks. Integration includes defining the user roles and processors to be implemented in order to provide a seamless experience for *ExtremeEarth* users. This integration is split into at least two iterations; the first iteration develops functional products that will satisfy the requirements of the use cases and the second iteration builds on the first one to automate as many of the manual processes of the first iteration as possible. A detailed description of each integration point is provided in the following:

1) **Raw EO data**. DIAS and CREODIAS in particular, provide EO data access to services and applications running on their platforms. This includes the downstream of Copernicus data as it is generated by satellites, such as the Sentinel constellations. At an infrastructure level, this data is persisted at an object store with an *S3* object interface, managed by CREODIAS. This infrastructure comes with *s3fs*, a tool that emulates filesystem storage using an S3 object interface[15].

2) **Pre-processed data**. TEPs provide the developers of the deep learning pipelines with pre-processed EO data which forms the basis for creating training and testing datasets. This pre-processed data is managed by TEPs but is stored in the CREODIAS infrastructure. Developers of the deep learning pipeline can also define and execute their own pre-processing, if the pre-processed data is not already available. To do that, various third-party libraries and frameworks can be implemented and used in different programming languages such as Python, C++, and Java. To further assist *ExtremeEarth* users with pre-processing data, Hopsworks provides APIs to run programs in languages other than Python, which is the de-facto language of machine learning frameworks and libraries.

3) **Object store**. CREODIAS provides an object store used for storing data produced and consumed by the TEPs services and applications. In *ExtremeEarth*, this object store is used primarily for storing training data required by the Polar and Food Security use cases. This training data is provided as input to the deep learning pipelines.

4) **EO Data Hub Catalog**. This service is provided and managed by CREODIAS. It provides various protocols including OGC WFS, a REST API, and OData [34] as interfaces to the EO data. The Data Hub uses cloud infrastructure and methods to efficiently process and distribute data in a matter of seconds. It removes the major hassle of downloading, archiving, and processing petabytes of data and simply makes the full and global archive easily available immediately via web services. EO Data Hub technology is designed to work with original EO data, avoiding the need for computing-intensive pre-processing and additional storage for processed tiles [34].

5) **TEP-Hopsworks EO data access**. Users can directly access raw EO data from their applications running on Hopsworks. Multiple methods, e.g., object data access API (*SWIFT/S3*), filesystem interface, etc., are provided for accessing Copernicus and other EO-data available on CREODIAS.

6) **TEP-Hopsworks infrastructure integration**. Hopsworks and both the Polar and Food Security TEPs are installed and operated on a DIAS infrastructure, which in the case of *ExtremeEarth* is the CREODIAS infrastructure. CREODIAS and its administrative tools enable TEPs to spawn and manage virtual machines and storage by using CloudFerro which provides an OpenStack-based cloud platform to TEPs. Hopsworks is then installed on virtual machines within the TEP domain so that it can access compute and storage resources provided by the TEPs.

7) **TEP-Hopsworks API integration**. Hopsworks is provided within the TEP environment as a separate application and is mainly used as a development platform for the deep learning pipelines and applications of the Polar and Food Security use cases. These applications are exposed in the form of processors to the TEP users. Practically, a processor is a Polar TEP abstraction that uses deep learning models that have previously been developed by the data scientists of the *ExtremeEarth* use cases. Hence, Polar TEP users are presented with a geographical map where they

[15] https://github.com/dask/s3fs

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

can select data from a specific region and timeframe as input to one of the available processors. This means that the machine learning model is developed within Hopsworks and then served to users as a processor in Polar TEP where they can make use of increased data access and options for batch or triggered processing capabilities.

8) **Hopsworks-TEPs datasets**. TEPs provide users with access to data to be served as input to processors from various sources. Such sources include the data provided by CREODIAS and external services that the TEP can connect to. For the Polar use case, Sentinel-1 data is the primary data source that is readily available on CREODIAS. For the Food Security use case, pre-processed Sentinel-2 data is the primary data source. The pre-processing takes place on the Food Security TEP, where cloud-masking, atmospheric correction, and extraction of leaf area index is performed before handing the data to Hopsworks. The pre-processed data is stored in an object storage provided by CREODIAS, thus made available to Hopsworks users by exchanging authentication information.

9) **Linked Data tools integration**. Linked data applications are deployed as Hopsworks jobs using Apache Spark. A data scientist, as shown in Figure 17, can use GeoTriples-Spark to transform big geospatial data into the RDF, according to the ontologies developed for the Food Security and the Polar use cases [49]. Then, JedAI-Spatial can be used for spatial interlinking. The resulting RDF data will be stored in Strabo2 [52], where the data scientist can pose GeoSPARQL queries using the Apache Livy interface for interacting with Spark. Moreover, a user can pose GeoSPARQL queries in Semagrow to combine the big linked geospatial data served by Strabo2 with external geospatial sources. As an example, such integration allows the validation of Food Security data using external geospatial datasets. Strabo2 and Semagrow will be deployed using the JBoss application server running in Hopsworks, and can be accessible as standard SPARQL endpoints from other modules of the Hopsworks platform, based on the SPARQL HTTP protocol[16].

Building and serving models in *ExtremeEarth*. The Polar and Food Security use cases are the two main scenarios of the *ExtremeEarth* software architecture for the application of large-scale Copernicus EO data. For the Polar use case, the Polar TEP has established an instance of the Hopsworks platform on CREODIAS to allow the development of deep learning models, as explained above. The resulting processor is integrated into the Polar TEP, as described above. For the Food Security use case, the instance of the Hopsworks platform on CREODIAS, set up for the Polar use case, is jointly used. There are *two* methods by which the trained model can be served via the Polar TEP: *(i)* The model can be exported from Hopsworks and make it fully embedded into the Polar TEP processor. *(ii)* The model can be served online on Hopsworks and a processor on Polar TEP submits inference requests to the model serving instance on Hopsworks and returns the results.

[16]https://www.w3.org/TR/sparql11-protocol/

In method *(i)* which is also supported in Food Security TEP, once the machine learning model is developed, it can then be transferred from Hopsworks to the Polar or Food Security TEP by using the Hopsworks REST API and Python SDK. TEP users can integrate the Hopsworks Python SDK into the processor workflow to further automate the machine learning pipeline lifecycle. In method *(ii)*, Polar TEP is able to submit inference requests to the model being served by the online model serving infrastructure run on Kubernetes and hosted on Hopsworks.

The Food Security TEP provides deployment of deep learning models as TEP services using method *(i)*. As the currently employed models require satellite data time series separately preprocessed on the Food Security TEP, training data is first made available to the Data Scientist on Hopsworks via exchange of authentication information. A trained model is then exported from Hopsworks and embedded as a processing service in Food Security TEP, where it can be independently run with preprocessed input data, shared, published, and further customized by the service owners for utilization by the wider TEP user community.

Hopsworks provides a REST API for clients to work with model serving, and authentication is done in the form of API keys managed by Hopsworks on a per-user basis. These keys can therefore be used by external clients to authenticate against the Hopsworks REST API. The TEP end-user would experience almost no difference between these two methods of model serving. However, the implementation is considerably different. Figure 16 shows how the machine learning model on Hopsworks can be linked to the Polar TEP processor. It shows how the classification step takes a patch or a set of patches (a batch) and transmits the image patches over the network to the Hopsworks REST API endpoint before the model is served by the Hopsworks.

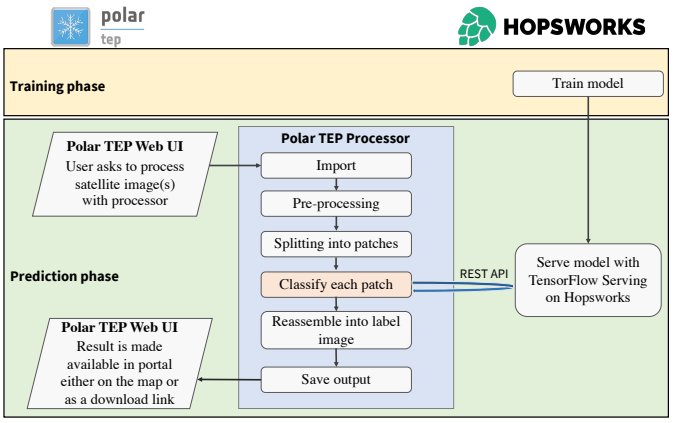


Fig. 16: Interaction of the machine learning model on Hopsworks with the Polar TEP processor.

B. ExtremeEarth *Flow of Events*

Figure 17 shows the flow of events of the products offered by the TEPs using Hopsworks. Details about the specific APIs and services used in each event are presented as follows.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

(1) EO data scientists log in to Hopsworks, (2) Read and pre-process raw EO data in Hopsworks, TEP, or in the local machine, (3) Create training datasets based on the intermediate pre-processed data, (4) Develop deep learning pipelines, (5) Perform linked data transformation, interlinking, and storage, (6) Log in to the Polar or Food Security TEP applications, (7) Select Hopsworks as the TEP processor. The processor starts the model serving in Hopsworks via the REST API. The processor also downloads the model from Hopsworks via the REST API and serving is done within the TEP application, (8) Submit federated queries with Semagrow and use the semantic catalogue built into Hopsworks.

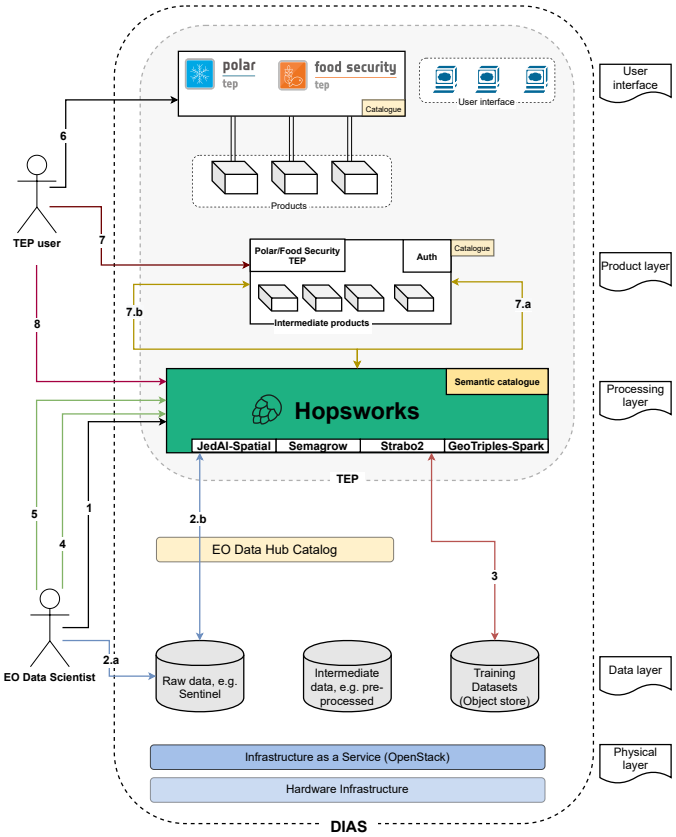


Fig. 17: *ExtremeEarth* architecture and flow of events.

VIII. Deep Learning Classification Models

ExtremeEarth aims to advance the state-of-the-art by developing ad-hoc distributed deep learning architectures tailored to the peculiar properties of Copernicus Sentinel satellite data. The *ExtremeEarth* infrastructure has developed and continues to develop prototype deep learning models for the Polar and Food Security use cases mostly targeting classification tasks to classify the objects, linked data to build maps (e.g., sea ice mapping for Polar use case and crop type maps for the Food Security use case).

A. Deep Learning Models for the Polar Use Case

The Polar use case exploits a large volume of Sentinel-1 (S1) SAR images to model various deep learning-based architectures. The purpose is to handle the extreme analytics and big data challenges associated with sea ice classification. By taking advantage of Hopsworks, we developed an ad-hoc architecture and explored an existing and deeper network for sea ice classification. The ad-hoc architecture is flexible but it generally requires the optimization of many hyperparameters. The existing deeper architecture reduces the modeling aspect of the network design. However, training existing architecture entails a large amount of data and long training times. Our ad-hoc architecture comprises three convolution layers and three max-pooling layers. For these layers, the number of kernels/filters are 32, 64, and 64, respectively. Our model also uses three fully connected layers with 1024, 512, and 2 nodes, respectively. We also exploit a regularization strategy namely to deal with the problem of overfitting. For the existing deeper architecture, we exploited the VGG-16 model [53] for sea ice classification. For the ad-hoc and existing VGG-16 architectures, the number of fully connected layers is the same. We trained the VGG-16 architecture from scratch using our sea ice training dataset. We also consider the transfer learning strategy to train the VGG-16 architecture. In fact, training the architecture from scratch provides insight into the impact of a deeper network on the sea ice classification task.

TABLE III: The comparison of the validation accuracy of different models: considering two different pixel resolutions.

Training Strategies	Validation Accuracy	Resolution in Pixels	Resolution in Meters
Ad hoc CNN	98.53%	32×32	1280
VGG-16 trained from scratch + augmentation	99.79%	32×32	1280
VGG-16 Modified + augmentation	**99.89%**	32×32	1280
VGG-16 Modified + augmentation	99.30%	20×20	800

For training the ad-hoc and existing architectures, we used an in-house dataset for sea ice classification [54]. This dataset is based on Sentinel-1 Extended Wide (EW) Level-1 Ground Range Detected (GRD) scenes. Training is done in the sensor range–azimuth geometry with a pixel spacing of 40 m $\times$ 40 m. The dataset was acquired north of the Svalbard archipelago in the winter months between September and March during the period 2015-2018. This dataset was pre-processed by applying the standard ESA thermal noise removal algorithm, calibrated using the σ_0 look-up table, and multi-looked using a 3×3 boxcar filter. We extracted patches from the images of this dataset for five classes namely: Water (including ice-free water (windy)), ice-free water (calm), and open water in leads, Brash/Pancake Ice, Young Ice, Level First-Year Ice, and Deformed Ice (including both first-year and multi-year ice). For binary Water/Ice classification, we combined the extracted patches into two classes, namely Water and Ice. The models take three channels as input, consisting of HH and HV σ_0 intensities in dB, as well as the angle of incidence. Additionally, we also modified the VGG-16 model by reducing the max-pooling layers. This modification allows us to feed smaller patches to get higher resolution results. We present the results of our ad-hoc and the existing VGG-16 model in Table III using different setups. Our modified VGG-16 model trained from scratch showed 99.89% validation accuracy, and the ad-hoc architecture achieved 98.53% validation accuracy.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

We also found the effect of using different patch sizes for the three-channel cases. Figure 18 depicts results from the modified VGG-16 model considering 32×32 patch size for binary classification. The first row indicates the input images and the second row indicates the processed images. In our experiments, the training and validation data are stored on Hopsworks and trained models are served via Hopsworks through the suggested pipeline.

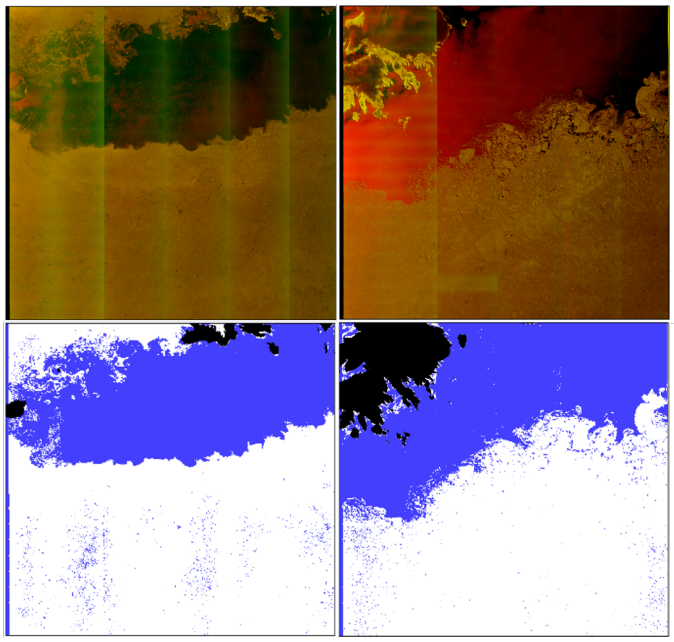


Fig. 18: Classification results from north of Svalbard using a to 32×32 patch size. Ice is annotated in white, water is annotated in blue and the land mask is shown in black.

B. Deep Learning Model for the Food Security Use Case

The deep learning model developed for the Food Security use case exploits the large volume of multitemporal multispectral Sentinel-2 images to generate high-spatial resolution crop type and crop boundaries maps. Although Sentinel-2 allows for accurate crop type mapping, this particular classification task presents many challenges. Agricultural areas are usually dominated by few common crops, which are extensively cultivated (e.g., corn, wheat). This leads to very different prior probabilities for different crop types. Such imbalanced datasets have to be properly handled to avoid low classification results on minority classes. Indeed, the accurate classification of minor crop types is still of interest to the local authorities and important to achieve accurate agricultural mapping. Moreover, one of the most critical issues of large-scale crop type mapping is the harmonization of the time series from the temporal viewpoint. To achieve accurate classification results, it is necessary to have a precise characterization of the phenological trends of the crop categories. The main problem is related to the fact that time series acquired over different Sentinel-2 tiles present unequal lengths and are characterized by variations in the temporal sampling rate. Moreover, for some geographical regions, the cloud and snow coverage can be so heavy in some months that several optical images should be discarded, thus affecting the temporal sampling of the tile.

To address these challenges, the proposed system architecture is based on two main steps: (1) the temporal harmonization of the time series of Sentinel-2 images acquired over different tiles, and (2) the training of the weighted multitemporal deep learning architecture able to handle imbalanced classification problems. For each Sentinel-2 tile, the corresponding time series is converted into a stack of 12 monthly composites considering a statistic-based approach that works at a pixel level. The images acquired within each month are collapsed into a single one computing their median per pixel per band [55]. This step allows us to sharply reduce the impact of cloud coverage while generating a consistent multitemporal spectral signature across tiles. Then, a multitemporal deep learning model designed for mitigating the imbalanced classification problem is trained. The adopted architecture is a multilayer LSTM, which performs a pixel-level classification [56]. The capability of the LSTM to handle the temporal information is crucial to accurately model the temporal dynamics of the crops and to capture their phenological growth [29]. To mitigate the problem of having severely imbalanced training data, we trained a weighted LSTM according to the procedure proposed in [57]. In the first training phase, the weights of the LSTM cost functions are defined per crop type according to the number of samples of each class. This step is carried out under the assumption that the number of samples per class is proportional to the prior probability of the different crops. Such weight initialization avoids that the gradient is mainly dominated by the contributions of the dominant classes. The network weights obtained at the end of the first phase are then used as the initial ones of the second training phase, which is performed using the standard cost functions.

To successfully train the deep network a considerably high amount of training samples is required. To solve this problem, we leveraged existing publicly available crop type maps based on farmer's declarations. Such maps are produced in the context of the *Common Agricultural Policy* of the European Union [58]. The crop types are collected with surveys within the subsidy application process, while the polygon field boundaries are the ones provided by the Land Parcel Identification System (LIPS). In particular, the 2018 Austrian crop type map was considered [59]. Although the reliability of the map is high, a machine learning-based procedure was defined to automatically extract labeled units having a high possibility to be correctly associated to their labels [55]. The obtained training dataset is made up of more than 1 million samples associated with 15 crop categories, namely: "grassland," "maize," "legumes," "winter caraway," "winter wheat," "rapeseed," "potato," "rye," "winter barley," "beet," "spring barley," "soy," "sunflower," "permanent plantations", "triticale" and "other crops". Please note that while the 2018 Austrian crop type map can be used to perform crop parameter estimation at country scale for one specific year, the extracted training set can be used to train deep learning models able to generate multi-years crop classification maps and to classify a study area larger than the Austrian country.

The preliminary experimental analysis presented has been carried out in the European Danube catchment, considering 15 Sentinel-2 tiles covering the Austrian country. Different tiles were used to define the training set (13 tiles), test set (1 tile), and validation set (1 tile) to consider spatially uncorrelated samples for training, validating, and testing the model. To accurately represent the phenological trend of the crop types, we focus the attention on the images acquired in the agronomic year from September 2017 to August 2018 (i.e., the period from one year's harvest to the next one for an agricultural commodity). The architecture has been trained and validated on the Hopsworks platform through a grid search approach for the selection of the best network parameters. The hyper-parameters were chosen by testing several combinations of network layers $L \in \{2, 3, 4\}$ and number of cells per layer $\in \{100, 125, 200, 225, 300\}$. At each training step, the calculation of the cross-entropy loss is computed and back-propagated through the network layers. The weights are optimized through an RMSprop optimizer [60]. The learning rate has been set to 10^{-3} and the weight decay to 0.4.

To assess the effectiveness of the proposed weighted LSTM, we validate the results obtained considering: (1) the Austrian crop type map, and (2) the 2018 Land Use and Cover Area frame Statistical Survey (LUCAS) database [61]. Both the validation analysis allow us to have statistical independent evaluation. While the results on the Austrian Map are computed considering the validation tile (T33UVP) not included in the training set, the LUCAS database is made up of reference samples collected with field surveys. In the latter, all the crop types were evaluated. In the former, only the the crop types present in the database were considered (i.e., "grassland", "maize", "winter wheat", "rapeseed", "potato", "rye", "beet", "soy", "sunflower", "permanent plantations").

What should be noted is how the integrated *ExtremeEarth* tools were used to carry out the quantitative evaluation of the obtained classification map considering the LUCAS database. LUCAS reports exact point coordinates, both the theoretical target and the actual GPS signal at the time of the observation. However, matching these data to the obtained crop classification map is not trivial. To handle the different spatial resolution of the two data sources, the standard approach typically used to integrate in-situ and EO data, associates each surveyed point to the closest instance present in the classification map. Although the application of this deterministic rule is conceptually straightforward, in operational scenarios this may lead to wrong results. As the surveyor is actually situated on the road and not inside the crops, in agricultural areas with several adjacent crops for each LUCAS point even GPS accuracy may lead to the wrong match. A more robust approach is looking for inconsistencies between LUCAS datapoints and the classification maps produced by EO data. Intuitively, the idea is that for every LUCAS point there must be at least one crop with the same label and in reasonable proximity to the LUCAS point; otherwise this LUCAS point is a negative validation point in the sense that at least one nearby shape is mis-labeled (although we do not automatically know which one). To achieve accurate validation results, for each LUCAS point we analyzed its surrounding area in a radius of 10m, i.e., the spatial resolution of the obtained classification map. From the computational point of view, executing distance queries at such a large scale can be challenging: the obtained classification map extends for 83.879 km^2 and is made up of 936566 instances, that has to be linked to the 8841 field samples for the considered study area. This was executed by using Strabo2 on Hops to serve the large-scale crop type classification map and Semagrow to federate the EO-generated maps with the LUCAS dataset while mapping between the different crops codelists.

The weighted LSTM was compared with two benchmark methods widely employed in the literature for crop type mapping, namely the time convolutional-based model TempCNN [32] and the recurrence-based model StarRNN [27]. Table IV shows the numerical results obtained on both the validation datasets. The LSTM obtained on the validation tile an Overall Accuracy (OA%) of 85.39%, outperforming both the benchmarks methods that achieved 81.71% and 81.17% for TempCNN and StarRNN, respectively. The capability of the weighted LSTM to better handle the severely unbalanced classification problem is confirmed by the median Fscore (F1%) which is 84.08%, 78.33% and 83.73% for the weighted LSTM, TempCNN and StarRNN, respectively. Similar results are achieved on the LUCAS database, where the weighted LSTM achieved the highest median Fscore (F1%) 84.20% compared to 80.79% and 82.24% for TempCNN and StarRNN, respectively. The highest OA% is obtained by the TempCNN (89.94 %) compared to 89.72% and 89.53% for the weighted LSTM and StarRNN, respectively. Figure 19 shows qualitative examples of the obtained crop type map result. Although the proposed deep learning models provide a result at pixel level, the crop types and crop boundaries are accurately detected by the proposed system architecture.

TABLE IV: Crop type classification results obtained on the map validation tile (T33UVP) and the LUCAS database. The proposed weighted LSTM is compared with two benchmark methods: (1) the time convolutional-based model TempCNN, and (2) the recurrence-based model StarRNN. The overall accuracy (OA%) and the median Fscore (F1%) are reported per method.

	Map (T33UVP)		**LUCAS**	
Method	**OA%**	**F1%**	**OA%**	**F1%**
TempCNN [32]	81.71	78.33	**89.94**	80.79
StarRNN [27]	81.17	83.73	89.53	82.24
LSTM Weig.	**85.39**	**84.08**	89.72	**84.20**

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

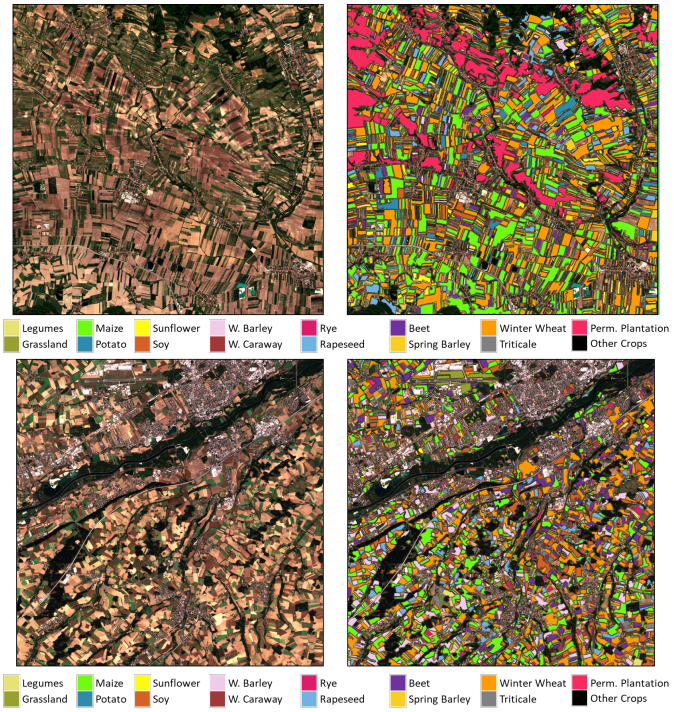


Fig. 19: Qualitative examples of the obtained crop type map are reported together with one Sentinel-2 image of the time series. The different crop types are depicted in different colors.

IX. Conclusion

In this paper, we present the software architecture of *ExtremeEarth* that aims at the development of scalable deep learning and geospatial analytics techniques for processing and analyzing petabytes of Copernicus data. The main contributions presented in this paper are centered around the Polar and Food Security use cases of the *ExtremeEarth* software architecture.

For the Polar use case, we investigated the potential of different CNN models for sea ice classification. The results showed that the complex architecture with deeper layers (such as those based on the VGG network) typically obtain more relevant features and subsequently show better classification results. Moreover, we evaluated the value of data augmentation and a modified VGG-16 architecture to prevent over-fitting caused by a scarce training dataset and obtain better resolution. We also showed the robustness of the proposed models when applied to SAR scenes collected at different spatial locations and times. Additive system noise in SAR images is a serious challenge to obtain refined sea ice maps. In this sense, scarce training data escalate this issue; therefore, we will address this specific issue in future works. Additionally, we note that sea ice classification represents a fairly new application area for deep learning architectures.

SAR input images do not always have clear boundaries between classes and are not as rich in distinct corners, edges, and line features as images in computer vision. Future work will carefully analyze the extracted feature extraction modules in order to find the proper depth needed for the architecture to extract the relevant information. We will furthermore investigate semi-supervised approaches to remedy the scarce training issue, and investigate the impact of adding additional input layers containing derived quantities motivated by a physics-based understanding of SAR imaging of sea ice.

For the Food Security use case, we presented an approach for crop type mapping using Sentinel-2 time series based on a deep learning architecture. The proposed method is able to generate a crop type map characterized by a very detailed classification scheme made up of 16 crop types. The proposed approach first harmonizes the time series producing monthly composite to guarantee homogeneous data across different tiles and mitigate the cloud cover issues. Then, it exploits a very large training dataset to successfully train the considered multitemporal deep learning architecture, an LSTM model. Finally, the LSTM is used to perform crop type mapping for the whole study area. The LSTM-based approach is proved to be effective for this task since its recurrent structure can model the phenological evolution of the crop types. This is confirmed by the numerical results, where the proposed deep learning model is able to achieve an OA of 85.39% and 89.72% on the map and the LUCAS validation datasets, respectively. As future developments, we aim to enlarge the considered study area to map the agricultural land surrounding the Austrian territory (i.e., including Hungary, Moravia, Slovakia, etc.). To this end, Sentinel-2 tiles where no ground reference data will be classified. Moreover, to provide a more comprehensive environmental monitoring of the considered study area, we aim to generate multi-year crop type maps. In particular, we aim to focus on the Sentinel-2 time series of the following agronomic years (i.e., from September 2018 to August 2019 and from September 2019 to August 2020). Such experimental analyses allow us to better assess and evaluate the robustness and the generalization capability of the considered multi-temporal deep learning model from the spatial and temporal viewpoint. To accurately achieve these goals, fine-tuning strategies will be investigated.

The implementations of the Polar and Food Security use cases utilize the four linked geospatial data systems presented: GeoTriples-Spark, JedAI-spatial, Strabo2, and SemaGrow. GeoTriples-Spark is a new version of GeoTriples which is able to transform big geospatial data into RDF. For this system, we showed that it is capable of transforming up to terabytes of input geospatial data in a reasonable amount of time. We have also presented JedAI-spatial, a system for interlinking geospatial data sources expressed in RDF. We have shown how to improve the performance of JedAI-spatial, significantly increasing the number of topological links it produces when allocated to limited computational resources. We have also presented Strabo2, the first distributed geospatial RDF store, and a preliminary evaluation of its performance and scalability. We are currently working on improving the efficiency of Strabo2 using data compression and better selectivity estimates for query planning, and we plan to perform a more thorough experimental evaluation with larger datasets. Finally, we presented SemaGrow, the only federation system for linked geospatial RDF stores available in the literature up to now. We are currently working on efficient federated query processing techniques and integrate them in SemaGrow.

Summarizing the contributions of ExtremeEarth, we can point out to its advances in deep learning for sea ice classification and crop type mapping, the new improved versions of the linked geospatial data systems that scale to big data, and the integration of these components in a software architecture that is used to implement the two use cases.

ACKNOWLEDGMENT

This work is supported by the *ExtremeEarth* project[17] funded by European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 825258.

REFERENCES

[1] "ESA Earth Observation Missions," https://earth.esa.int/web/guest/missions/esa-operational-eo-missions, 2021. [Online]. Available: https://sentinels.copernicus.eu

[2] "ESA Third Party Missions," https://earth.esa.int/web/guest/missions/3rd-party-missions/overview, 2021. [Online]. Available: https://earth.esa.int/web/guest/missions/3rd-party-missions/overview

[3] C. O. Dumitru, V. Andrei, G. Schwarz, and M. Datcu, "Machine Learning for Sea Ice Monitoring from Satellites." *Int. Arch. Photogrammetry, Remote Sensing & Spatial Inf. Sci.*, 2019.

[4] S. Migdall, S. Dotzler, C. Miesgang, F. Appel, M. Muerth, H. Bach, G. Weikmann, C. Paris, D. Marinelli, and L. Bruzzone, "Water stress assessment in Austria based on deep learning and crop growth modelling." *Submitted to BiDS, 2021*, 2021.

[5] J. L. Awange and J. B. K. Kiema, "Microwave remote sensing," in *Environmental Geoinformatics*. Springer, 2013, pp. 133–144.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[7] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He, "Accurate, large minibatch sgd: Training imagenet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.

[8] T. Ioannidis, G. Garbis, K. Kyzirakos, K. Bereta, and M. Koubarakis, "Evaluating geospatial RDF stores using the benchmark Geographica 2," *Journal on Data Semantics*, 2021.

[9] M.-A. Moen, A. P. Doulgeris, S. N. Anfinsen, A. H. Renner, N. Hughes, S. Gerland, and T. Eltoft, "Comparison of feature based segmentation of full polarimetric SAR satellite sea ice images with manually drawn ice charts," 2013.

[10] A. S. Fors, C. Brekke, A. P. Doulgeris, T. Eltoft, A. H. Renner, and S. Gerland, "Late-summer sea ice segmentation with multi-polarisation SAR features in C and X band," *The Cryosphere*, vol. 10, no. 1, pp. 401–415, 2016.

[11] Z. Yu, T. Wang, X. Zhang, J. Zhang, and P. Ren, "Locality preserving fusion of multi-source images for sea-ice classification," *Acta Oceanologica Sinica*, vol. 38, no. 7, pp. 129–136, 2019.

[12] Y. Li, F. Gao, J. Dong, and S. Wang, "A Novel Sea Ice Classification Method from Hyperspectral Image Based on Bagging PCA Hashing," in *2018 Fifth International Workshop on Earth Observation and Remote Sensing Applications (EORSA)*. IEEE, 2018, pp. 1–4.

[13] J.-W. Park, A. A. Korosov, M. Babiker, J.-S. Won, M. W. Hansen, and H.-C. Kim, "Classification of Sea Ice Types in Sentinel-1 SAR images," *The Cryosphere Discussions*, pp. 1–23, 2019.

[14] Y. Zhang, T. Zhu, G. Spreen, S. Zhang, F. Li *et al.*, "Sea ice-water classification on dual-polarized Sentinel-1 imagery during melting season." in *Geophysical Research Abstracts*, vol. 21, 2019.

[15] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva, "Land use classification in remote sensing images by convolutional neural networks," *arXiv preprint arXiv:1508.00092*, 2015.

[16] L. Wang, K. A. Scott, L. Xu, and D. A. Clausi, "Sea ice concentration estimation during melt from dual-pol SAR scenes using deep convolutional neural networks: A case study," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 8, pp. 4524–4533, 2016.

[17] L. Wang, K. Scott, and D. Clausi, "Sea ice concentration estimation during freeze-up from SAR imagery using a convolutional neural network," *Remote Sensing*, vol. 9, no. 5, p. 408, 2017.

[18] R. Kruk, M. C. Fuller, A. S. Komarov, D. Isleifson, and I. Jeffrey, "Proof of Concept for Sea Ice Stage of Development Classification Using Deep Learning," *Remote Sensing*, vol. 12, no. 15, p. 2486, 2020.

[19] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[20] Y. Han, Y. Gao, Y. Zhang, J. Wang, and S. Yang, "Hyperspectral Sea Ice Image Classification Based on the Spectral-Spatial-Joint Feature with Deep Learning," *Remote Sensing*, vol. 11, no. 18, p. 2170, 2019.

[21] Y. Gao, F. Gao, J. Dong, and S. Wang, "Transferred deep learning for sea ice change detection from synthetic-aperture radar images," *IEEE Geoscience and Remote Sensing Letters*, vol. 16, no. 10, pp. 1655–1659, 2019.

[22] Y. T. Solano-Correa, F. Bovolo, L. Bruzzone, and D. Fernández-Prieto, "A Method for the Analysis of Small Crop Fields in Sentinel-2 Dense Time Series," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 58, no. 3, pp. 2150–2164, 2020.

[23] B. Watkins and A. Van Niekerk, "A comparison of object-based image analysis approaches for field boundary delineation using multi-temporal Sentinel-2 imagery," *Computers and Electronics in Agriculture*, vol. 158, pp. 294–302, 2019.

[24] M. Belgiu and O. Csillik, "Sentinel-2 cropland mapping using pixel-based and object-based time-weighted dynamic time warping analysis," *Remote sensing of environment*, vol. 204, pp. 509–523, 2018.

[25] M. Rußwurm, S. Lefèvre, and M. Körner, "BreizhCrops: A Satellite Time Series Dataset for Crop Type Identification," in *Time Series Workshop of the 36th International Conference on Machine Learning (ICML)*, Long Beach, United States, 2019. [Online]. Available: https://hal.archives-ouvertes.fr/hal-02343898

[26] R. Nijhawan, D. Joshi, N. Narang, A. Mittal, and A. Mittal, "A futuristic deep learning framework approach for land use-land cover classification using remote sensing imagery," in *Advanced Computing and Communication Technologies*. Springer, 2019, pp. 87–96.

[27] M. O. Turkoglu, S. D'Aronco, J. D. Wegner, and K. Schindler, "Gating Revisited: Deep Multi-layer RNNs That Can Be Trained," 2021.

[28] E. Ndikumana, D. Ho Tong Minh, N. Baghdadi, D. Courault, and L. Hossard, "Deep recurrent neural network for agricultural classification using multitemporal sar sentinel-1 for camargue, france," *Remote Sensing*, vol. 10, no. 8, p. 1217, 2018.

[29] M. Rußwurm and M. Korner, "Temporal vegetation modelling using long short-term memory networks for crop identification from medium-resolution multi-spectral satellite images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 11–19.

[30] M. Rußwurm and M. Körner, "Multi-Temporal Land Cover Classification With Long Short-Term Mermory Neural Networks." *International Archives of the Photogrammetry,*

[17] Project website: http://earthanalytics.eu/

Remote Sensing & Spatial Information Sciences, vol. 42, 2017.
[31] L. Zhong, L. Hu, and H. Zhou, "Deep learning based multi-temporal crop classification," *Remote sensing of environment*, vol. 221, pp. 430–443, 2019.
[32] C. Pelletier, G. I. Webb, and F. Petitjean, "Temporal Convolutional Neural Network for the Classification of Satellite Image Time Series," *Remote Sensing*, vol. 11, no. 5, 2019.
[33] W. Tang, G. Long, L. Liu, T. Zhou, J. Jiang, and M. Blumenstein, "Rethinking 1D-CNN for Time Series Classification: A Stronger Baseline," 2021.
[34] CREODIAS, "Data and Processing," https://creodias.eu/data-related-services, 2021. [Online]. Available: https://creodias.eu/data-related-services
[35] M. Muerth, S. Migdall, M. Hodrius, F. Niggemann, M. Holzapfel, H. Bach, S. Gilliams, T. Van Roey, A. Cuomo, P. Harwood *et al.*, "Food Security TEP-Supporting sustainable intensification of food production from Space," in *IOP Conference Series: Earth and Environmental Science*, vol. 509, no. 1. IOP Publishing, 2020, p. 012038.
[36] G. Robbie, C. Owen, and L. Yevgeni, "Introducing Petastorm: Uber ATG's Data Access Library for Deep Learning," https://eng.uber.com/petastorm/, 2018. [Online]. Available: https://eng.uber.com/petastorm/
[37] M. Meister, S. Sheikholeslami, A. H. Payberah, V. Vlassov, and J. Dowling, "Maggy: Scalable Asynchronous Parallel Hyperparameter Search," in *Proceedings of the 1st Workshop on Distributed Machine Learning*, 2020, pp. 28–33.
[38] M. Meister, S. Sheikholeslami, R. Andersson, A. A. Ormenisan, and J. Dowling, "Towards Distribution Transparency for Supervised ML With Oblivious Training Functions," in *Workshop on MLOps Systems*, 2020.
[39] J. Bergstra, D. Yamins, and D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *International conference on machine learning*. PMLR, 2013, pp. 115–123.
[40] D. Ginsbourger, J. Janusevskis, and R. Le Riche, "Dealing with asynchronicity in parallel Gaussian process based global optimization," Ph.D. dissertation, Mines Saint-Etienne, 2011.
[41] L. Li, K. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6765–6816, 2017.
[42] L. Li, K. Jamieson, A. Rostamizadeh, E. Gonina, M. Hardt, B. Recht, and A. Talwalkar, "Massively parallel hyperparameter tuning," 2018.
[43] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
[44] N. Garg, *Apache kafka*. Packt Publishing Ltd, 2013.
[45] J. de la Rúa Martínez, "Scalable Architecture for Automating Machine Learning Model Monitoring," 2020.
[46] M. Ismail, S. Niazi, M. Ronström, S. Haridi, and J. Dowling, "Scaling HDFS to more than 1 million operations per second with HopsFS," in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 2017, pp. 683–688.
[47] D. Vohra, "Apache parquet," in *Practical Hadoop Ecosystem*. Springer, 2016, pp. 325–335.
[48] S. Niazi, M. Ronström, S. Haridi, and J. Dowling, "Size matters: Improving the performance of small files in hadoop," in *Proceedings of the 19th International Middleware Conference*, 2018, pp. 26–39.
[49] D.-A. Pantazi, G. Stamoulis, M. Koubarakis, N. Hughes, A. Everett, F. Appel, "D1.7 - Semantic catalogue design and implementation-v2." Available from: http://earthanalytics.eu/deliverables.html, 2020. [Online]. Available: http://earthanalytics.eu/deliverables.html
[50] A. Charalambidis, A. Troumpoukis, and S. Konstantopoulos, "Semagrow: optimizing federated SPARQL queries," in *Proceedings of the 11th International Conference on Semantic Systems, SEMANTICS 2015, Vienna, Austria, September 15-17, 2015*, 2015, pp. 121–128.
[51] G. Papadakis, G. Mandilaras, N. Mamoulis, and M. Koubarakis, "Progressive, Holistic Geospatial Interlinking," in *The Web Conference*, 2021.
[52] M.Koubarakis et al., "From Copernicus Big Data to Extreme Earth Analytics," *22nd International Conference on Extending Database Technology (EDBT). Lisbon, Portugal*, 2019.
[53] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[54] S. Khaleghian, H. Ullah, T. Kræmer, N. Hughes, T. Eltoft, and A. Marinoni, "Sea Ice Classification of SAR Imagery based on Convolution Neural Networks," *Remote Sensing*. [Online]. Available: https://www.mdpi.com/2076-3417/10/6/2020
[55] C. Paris, G. Weikmann, and L. Bruzzone, "Monitoring of agricultural areas by using Sentinel 2 image time series and deep learning techniques," in *Image and Signal Processing for Remote Sensing XXVI*, vol. 11533. International Society for Optics and Photonics, 2020, p. 115330K.
[56] A. Graves, "Long short-term memory," in *Supervised sequence labelling with recurrent neural networks*. Springer, 2012, pp. 37–45.
[57] L. Bruzzone and S. B. Serpico, "Classification of imbalanced remote-sensing data by neural networks," *Pattern recognition letters*, vol. 18, no. 11-13, pp. 1323–1328, 1997.
[58] K. Kocur-Bera, "Data compatibility between the Land and Building Cadaster (LBC) and the Land Parcel Identification System (LPIS) in the context of area-based payments: A case study in the Polish Region of Warmia and Mazury," *Land Use Policy*, vol. 80, pp. 370–379, 2019.
[59] "Invekos," https://www.data.gv.at/katalog/dataset/e21a731f-9e08-4dd3-b9e5-cd460438a5d9, 2020. [Online]. Available: https://www.data.gv.at/katalog/dataset/e21a731f-9e08-4dd3-b9e5-cd460438a5d9,
[60] F. Zou, L. Shen, Z. Jie, W. Zhang, and W. Liu, "A sufficient condition for convergences of adam and rmsprop," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11 127–11 135.
[61] "LUCAS," https://ec.europa.eu/eurostat/web/lucas/data/primary-data/2018. [Online]. Available: https://ec.europa.eu/eurostat/web/lucas/data/primary-data/2018

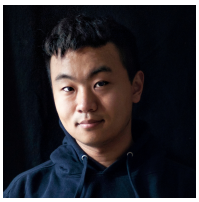
Desta Haileselassie Hagos received a Ph.D. degree in Computer Science from the University of Oslo, Faculty of Mathematics and Natural Sciences in April 2020. Currently, he is a Postdoctoral Research Fellow at the Division of Software and Computer Systems (SCS), Department of Computer Science, School of Electrical Engineering and Computer Science (EECS), KTH Royal Institute of Technology, Stockholm, Sweden, working on the H2020-EU project, ExtremeEarth: From Copernicus Big Data to Extreme Earth Analytics. He received his B.Sc. degree in Computer Science from Mekelle University, Department of Computer Science. He obtained his M.Sc. degree in Mobile Systems from Luleå University of Technology, Department of Computer Science Electrical and Space Engineering, Sweden in June 2012. His current research interests are in the areas of Machine Learning, Deep Learning, and Artificial Intelligence.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

Theofilos Kakantousis is a co-founder and VP of Product at Logical Clocks, the main developers of Hopsworks, an open-source Data and AI platform. He holds an MSc in Distributed Systems from KTH Royal Institute of Technology and has previously worked as a Middleware consultant at Oracle, Greece, and as a research engineer at SAP, Zurich, and at RISE SICS, Stockholm. He has published research papers on big data and machine learning platforms. He frequently gives talks on Hopsworks, and has presented Hopsworks at venues such as Strata San Jose/New York, Big Data Tech Warsaw, and BigData Moscow.

Vladimir Vlassov is a professor in Computer Systems at the Department of Computer Science, School of Electrical Engineering and Computer Science, KTH Royal Institute of Technology in Stockholm, Sweden. He was a visiting scientist with the Massachusetts Institute of Technology (1998) and at the University of Massachusetts Amherst (2004), USA. He has participated in a number of research projects funded by the European Commission, by Swedish funding agencies, and by the National Science Foundation (NSF) USA. He was one of the coordinators of the EMJD-DC Erasmus Mundus Joint Doctorate in Distributed Computing. Currently, he is a principal investigator from KTH in the H2020-EU project "ExtremeEarth: From Copernicus Big Data to Extreme Earth Analytics" (2018-2020). His research covers several areas in computer science, including data-intensive computing, stream processing, scalable distributed deep learning, distributed systems, autonomic computing, cloud, and edge computing.

Sina Sheikholeslami is a Ph.D. student in the Distributed Computing group of KTH Royal Institute of Technology, where his main focus is on the design of systems for distributed deep learning. He holds an M.Sc. in Data Science from KTH and TU Eindhoven.



Tianze Wang is a Ph.D. student in the Distributed Computing group at the Department of Computer Science of KTH Royal Institue of Technology. He received his B.Sc. degree in Software Engineering from Nankai University. He obtained his M.Sc. degree in Data Science from KTH Royal Institue of Technology and Eindhoven University of Technology. His current research interests are in Distributed Deep Learning.

Jim Dowling is CEO of Logical Clocks and an Associate Professor at KTH Royal Institute of Technology. He is the lead architect of the open-source Hopsworks platform, a horizontally scalable data platform for machine learning that includes the industry's first Feature Store. He received his Ph.D. in Distributed Systems from Trinity College Dublin and has worked at MySQL AB. He is a distributed systems researcher and his research interests are in the area of large-scale distributed systems and machine learning.

Claudia Paris (M'16) received the B.S. degree and M.S. (summa cum laude) degree in Telecommunication Engineering and the Ph.D. in Information and Communication Technology from the University of Trento, Italy, in 2010, 2012, 2016, respectively. She accomplished the Honors Master Program in Research within the Master Degree in Telecommunication Engineering in 2012. Since 2014 she is a teaching assistant at the Department of Information Engineering and Computer Science of the University of Trento, Italy, where she is currently an Assistant Professor. Her main research includes image and signal processing, machine learning, and deep learning with applications to remote sensing image analysis. The main research interests include remote sensing single date and TSs image classification, land cover map update, and fusion of multisource remote sensing data for the estimation of biophysical parameters. She conducts research on these topics within the frameworks of national and international projects. Dr. Paris won the very prestigious Symposium Prize Paper Award (SPPA) at the 2016 International Symposium on Geoscience and Remote Sensing (Beijing, China, 2016) and at the 2017 International Symposium on Geoscience and Remote Sensing (Fort Worth, Texas, USA, 2017). She has been a member of the program and scientific committee of several international conferences and workshops.

Daniele Marinelli received the "Laurea" (B.Sc.) degree in Electronics and Telecommunications Engineering, the "Laurea Magistrale" (M.Sc.) degree in Telecommunications Engineering (cum laude) and the Ph.D. in Information and Communication Technologies (cum laude) from the University of Trento, Italy, in 2013, 2015, 2019, respectively. Since 2017 he has been a teaching assistant at the Department of Information Engineering and Computer Science of the University of Trento where he is currently a postdoctoral researcher. His research interests are related to the multitemporal analysis of Hyperspectral images, LiDAR point clouds, and multispectral time-series. In 2017 he was a visiting Ph.D. student at the Integrated Remote Sensing Studio, University of British Columbia, Vancouver, Canada working on Change Detection in LiDAR data for forestry applications. He is the recipient of the prizes for the 2020 Best Italian Ph.D. Thesis and 2015 Best Italian master Thesis in the area of remote sensing awarded by the Italy Chapter of the IEEE Geoscience and Remote Sensing Society. He got Second Place in the Student Paper Competition at the 2018 IEEE International Geoscience and Remote Sensing Symposium (IGARSS 2018) hold in Valencia (Spain).

Giulio Weikmann received his "Laurea" (B.S.) degree and his "Laurea Specialistica" (M.S.) (summa cum laude) degree in Information and Communication Engineering from the University of Trento, Italy. He is currently a Ph.D. student in the Remote Sensing Lab (RSLab) and a teaching assistant at the Department of Information Engineering and Computer Science of the University of Trento, Italy. His major research interest concerns image and signal processing and the development of deep learning architectures for the automatic processing and classification of remote sensed optical data. He conducts research on these topics within the frameworks of national and international projects.

Lorenzo Bruzzone received the Laurea (M.S.) degree in electronic engineering (summa cum laude) and the Ph.D. degree in telecommunications from the University of Genoa, Italy, in 1993 and 1998, respectively. He is currently a Full Professor of telecommunications at the University of Trento, Italy, where he teaches remote sensing, radar, and digital communications. Dr. Bruzzone is the founder and the director of the Remote Sensing Laboratory (https://rslab.disi.unitn.it/) in the Department of Information Engineering and Computer Science, University of Trento. His current research interests are in the areas of remote sensing, radar and SAR, signal processing, machine learning, and pattern recognition. He promotes and supervises research on these topics within the frameworks of many national and international projects. He is the Principal Investigator of many research projects. Among the others, he is currently the Principal Investigator of the Radar for icy Moon exploration (RIME) instrument in the framework of the JUpiter ICy moons Explorer (JUICE) mission of the European Space Agency (ESA) and the Science Lead for the High-Resolution Land Cover project in the framework of the Climate Change Initiative of ESA. He is the author (or co-author) of 294 scientific publications in referred international journals (221 in IEEE journals), more than 340 papers in conference proceedings, and 22 book chapters. He is editor/co-editor of 18 books/conference proceedings and 1 scientific book. His papers are highly cited, as proven from the total number of citations (more than 39000) and the value of the h-index (91) (source: Google Scholar). He was invited as a keynote speaker in more than 40 international conferences and workshops. Since 2009 he has been a member of the Administrative Committee of the IEEE Geoscience and Remote Sensing Society (GRSS), where since 2019 he is Vice-President for Professional Activities.

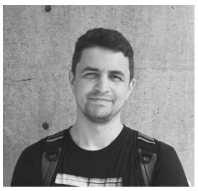
Dr. Bruzzone ranked first place in the Student Prize Paper Competition of the 1998 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Seattle, July 1998. Since that he was a recipient of many international and national honors and awards, including the recent IEEE GRSS 2015 Outstanding Service Award, the 2017 and 2018 IEEE IGARSS Symposium Prize Paper Awards, and the 2019 WHISPER Outstanding Paper Award. Dr. Bruzzone was a Guest Co-Editor of many Special Issues of international journals. He is the co-founder of the IEEE International Workshop on the Analysis of Multi-Temporal Remote-Sensing Images (MultiTemp) series and is currently a member of the Permanent Steering Committee of this series of workshops. Since 2003 he has been the Chair of the SPIE Conference on Image and Signal Processing for Remote Sensing. He has been the founder of the IEEE GEOSCIENCE AND REMOTE SENSING MAGAZINE for which he has been Editor-in-Chief between 2013-2017. Currently, he is an Associate Editor for the IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING. He has been Distinguished Speaker of the IEEE Geoscience and Remote Sensing Society between 2012-2016. He is a Fellow of IEEE.

Salman Khaleghian is a Ph.D. student in scalable computing for Earth observation at UiT The Arctic University of Norway, the faculty of Science and Technology. He received a bachelor's degree in Applied mathematics, majored in computer science, and an M.Sc. degree in Computer software engineering from the Science and Research Branch of Azad University, Iran. His research interest includes machine learning, deep learning, scalable deep learning, and computer vision.

Thomas Kræmer received the M.Sc. degree in data analysis and sensor technology in 2011 from the University of Tromsø (UiT), Tromsø, Norway, focusing on segmentation of sea ice using polarimetric synthetic aperture radar images. Since 2016 he has been head engineer with the Earth Observation Laboratory at UiT the Arctic University of Norway, Tromsø, Norway, where he is also pursuing a Ph.D. degree. His current research interests are on algorithms for automated analysis of SAR images for sea ice applications.

Torbjørn Eltoft (Member, IEEE) received the M.Sc. degree in 1981 and the Ph.D. degree in 1984 from UiT the Arctic University of Norway, where he joined the Department of Physics and Technology in 1988 as Associate Professor. he is now employed as Professor. Eltoft is the Director of CIRFA (Centre for Integrated Remote Sensing and Forecasting for Arctic Operations), a Centre for Research-based Innovation awarded by the Norwegian Research Council in 2014, developing knowledge and remote sensing technology for Arctic applications. From 2013-15 he was the Head of the Department of Physics and Technology at UiT. Eltoft has international research experience from the University of California, Irvine (1992-93 and 1997-98), and the University of California, San Diego (2004-05). He served as Associate Editor of the Elsevier journal Pattern Recognition for the period 2005-2011 and was guest-editor for the journal Remote Sensing's - Special Issue for the PolInSAR 2017 conference. Eltoft's research interests include multi-dimensional signal and image analysis, statistical modelling, neural networks, and machine learning, with emphasis on applications in multi-channel Synthetic Aperture Radar remote sensing. His current research focuses on multi-sensor remote sensing for Arctic applications. Eltoft has a significant publication record in the area of signal processing and remote sensing. He was co-recipient of the year 2000 Outstanding Paper Award in Neural Networks awarded by IEEE Neural Networks Council and of the Honourable Mention for the 2003 Pattern Recognition Journal Best Paper Award, and he was awarded The 2017 UiT Award for Research and Development from UiT the Arctic University of Norway.

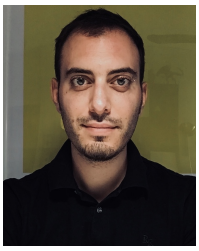
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing



Andrea Marinoni (S '07 - M '11 – SM '16) is an associate professor with the Earth Observation group, Centre for Integrated Remote Sensing and Forecasting for Arctic Operations (CIRFA), Department of Physics and Technology, UiT the Arctic University of Norway, Tromsø, Norway, and a visiting academic fellow with the Dept. of Engineering, University of Cambridge, UK. He received the B.S., M.Sc. "cum laude" and Ph.D. degrees in Electronic Engineering from the University of Pavia, Pavia, Italy, in 2005, 2007 and 2011, respectively. From 2013 to 2018, he has been a research fellow at Telecommunications and Remote Sensing Lab., Dept. of Electrical, Computer and Biomedical Engineering, University of Pavia, Pavia, Italy. In 2009, he has been a visiting researcher at the Communications Systems Lab., Dept. of Electrical Engineering, University of California – Los Angeles (UCLA), Los Angeles, CA, USA. In 2011, he has been the recipient of the two-year "Applied research grant", sponsored by the Region of Lombardy, Italy, and STMicroelectronics N.V.. In 2017, he has been the recipient of the INROAD grant, sponsored by the University of Pavia and Fondazione Cariplo, Italy, for supporting excellence in the design of ERC proposal. In 2018, he has been the recipient of the "Progetto professionalità Ivano Becchi" grant funded by Fondazione Banco del Monte di Lombardia, Italy, and sponsored by the University of Pavia, Italy and NASA Jet Propulsion Laboratory, Pasadena, CA, for supporting the development of advanced methods of air pollution analysis by remote sensing data investigation. He has been the recipient of Åsgard Research Programme and Åsgard Recherche+ Programme grants funded by Institut Français de Norvège, Oslo, Norway, in 2019 and 2020, respectively, for supporting the development of scientific collaborations between French and Norwegian research institutes.

From 2015 to 2017 he has been a visiting researcher at Earth and Planetary Image Facility, Ben-Gurion University of the Negev, Be'er Sheva, Israel; School of Geography and Planning, Sun Yat-Sen University, Guangzhou, P.R.C.; School of Computer Science, Fudan University, Shanghai, P.R.C.; Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, P.R.C.; Instituto de Telecomunicações, Instituto Superior Tecnico, Universidade de Lisboa, Lisbon, Portugal. In 2020 and 2021, he has been a visiting professor at the Department of Electrical, Computer and Biomedical Engineering, University of Pavia, Pavia, Italy. His main research interests are focused on efficient information extraction from multimodal remote sensing, nonlinear signal processing applied to large scale heterogeneous records, Earth observation interpretation and Big Data mining, analysis and management for human-environment interaction assessment. He is the founder and current chair of the IEEE GRSS Norway chapter. He is also an ambassador for IEEE Region 8 Humanitarian activities and a research contact point for the Norwegian Artificial Intelligence Research Consortium (NORA – nora.ai). He serves as a topical associate editor of machine learning for IEEE Transactions on Geoscience and Remote Sensing. He has been guest editor of three special issues on multimodal remote sensing and sustainable development for IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing. He is the leader of the GR4S committee within IEEE GRSS, coordinating the organization of schools and workshops sponsored by IEEE GRSS worldwide.

Despina-Athanasia Pantazi is a Research Associate in the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, and a Ph.D. candidate under the supervision of Professor Koubarakis. She holds a B.Sc. and M.Sc. from the same department. She has worked in various H2020 (Copernicus AppLab, AI4EU, ExtremeEarth, AI4Copernicus) projects to develop tools and techniques for making Earth Observation (Copernicus) data available as linked open data and create ontologies to enable semantic search catalogues over Earth Observation data. Her research interests focus in the areas of Artificial Intelligence and Semantic Web.



George Stamoulis is a Research Associate in the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, and a Ph.D. candidate under the supervision of Professor Koubarakis. He holds a B.Sc. and M.Sc. from the Department of Informatics and Telecommunications of the National and Kapodistrian University of Athens. He has worked in various FP7 (LEO, MELODIES) and H2020 (BigDataEurope, Copernicus AppLab, AI4EU, ExtremeEarth) projects to develop visualization tools that allow users to combine various linked data sources and EO data in digital maps. His research interests focus in the areas of Semantic Web, Data Visualization and Integration and User Interfaces.

Dimitris Bilidas is a postdoctoral researcher at the Management of Data, Information and Knowledge group, in the Dept. of Informatics and Telecommunications, National and Kapodistrian University of Athens. Mr. Bilidas obtained a B.Sc. from the Department of Informatics, University of Piraeus, an M.Sc. in Advanced Information Systems from the Department of Informatics and Telecommunications in National and Kapodistrian University of Athens, and a Ph.D. under the supervision of Professor Manolis Koubarakis, in the area of query languages for the Semantic Web and Distributed Query Processing. Mr. Bilidas has extensive research experience in several Greek and European research projects (DeepCube, ExtremeEarth, Optique, Copernicus AppLab, Indigo).

George Papadakis is an internal auditor of information systems and a research fellow at the National and Kapodistrian University of Athens. He has also worked at the NCSR "Demokritos", the National Technical University of Athens (NTUA), the L3S Research Center and the "Athena" Research Center. He holds a Ph.D. in Computer Science from Hanover University and a Diploma in Computer Engineering from NTUA. His research focuses on data integration and web data mining.

George Mandilaras is a Research Associate in the Department of Informatics and Telecommunications, National and Kapodistrian University of Athens. He holds a B.Sc. and M.Sc. from the same department. His work focuses on distributed tools for the transformation of large volumes of data in the RDF model and interlinking techniques for linked geospatial data. His research interests focus in the areas of Artificial Intelligence and Semantic Web.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JSTARS.2021.3107982, IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing

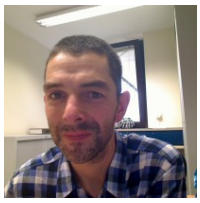
Manolis Koubarakis is a Professor and Director of Graduate Studies in the Dept. of Informatics and Telecommunications, National and Kapodistrian University of Athens. He leads the Artificial Intelligence team (http://ai.di.uoa.gr). He holds a Ph.D. in Computer Science, from the National Technical University of Athens, an M.Sc. in Computer Science, from the University of Toronto, and a diploma (B.Sc.) in Mathematics, from the University of Crete. He is a Fellow of EurAI (European Association for Artificial Intelligence) since 2015 and President of the Hellenic Association for Artificial Intelligence. He is a member of the Advisory Board that implements the Hellenic National Strategy for Artificial Intelligence. He has published more than 200 papers that have been widely cited (6863 citations and h-index 42 in Google Scholar) in the areas of Artificial Intelligence (especially Knowledge Representation), Databases, Semantic Web and Linked Geospatial Data (especially Earth observation data). His research has been financially supported with a total amount exceeding 8 million Euros by the European Commission, the Hellenic Foundation for Research and Innovation, the Greek General Secretariat for Research and Technology, the European Space Agency and industry.

Antonis Troumpoukis holds a B.Sc. in Informatics and Telecommunications (University of Athens, 2009), an M.Sc. in Theoretical Computer Science (University of Athens, 2011) and a Ph.D. on programming languages and knowledge representation (University of Athens, 2019). He is affiliated with the Institute of Informatics and Telecommunications, NCSR 'Demokritos', and has participated in several Greek and European research projects. Antonis' main research interests are knowledge representation, the semantics and implementation of programming languages and their various applications to data management, and he has published more than 15 papers in these fields.

Stasinos Konstantopoulos holds an MEng in Computer Engineering and Informatics from the University of Patras, Greece, an MSc in Artificial Intelligence from Edinburgh University, U.K., and a Ph.D. on machine learning and computational logic from Groningen University, the Netherlands. His main research interests are artificial intelligence and computational logic, and their applications to semantic modeling and to Big Data management and processing. In these fields, he has published more than fifty articles and full-length conference papers; and has served on the program committee or as a reviewer for major international conferences and journals. He is affiliated with the Institute of Informatics and Telecommunications, NCSR 'Demokritos' where he leads the Institute's Data Engineering Group. He has also served as a proposal evaluator and project reviewer for the European Commission and as a technical consultant for private companies.

Markus Muerth received the Ph.D. degree in Physical Geography from Ludwig Maximilians University in Munich, Germany in 2008. From 2004 to 2015, he was a researcher and lecturer for the Physical Geography and Remote Sensing Group of Ludwig Maximilians University, Munich. His worked focused on modelling energy transfer processes at the atmosphere-vegetation-soil boundary as well as using Regional Climate Model data for hydrological and land surface process models. Since 2016, he is a senior project manager of the R&D team of Vista Remote Sensing in Geosciences GmbH, Munich, Germany.

Florian Appel is senior scientist and product manager at VISTA Remote Sensing in Geosciences GmbH in Munich. He holds a Diplom (equivalent to M.Sc.) from the University of Munich in Geography and Remote Sensing. He joined VISTA with a focus on EO and hydrological applications in 2000. Florian Appel worked as senior scientist and project manager for international and national projects with the main emphasis on hydrological, e.g. snow, water, and energy applications, EO services, and businesses. Florian Appel is a member of the expert jury at Copernicus Masters since several years.



Andrew Fleming is the Head of the Mapping and Geographic Information Centre at the British Antarctic Survey. He leads the application of geospatial and remote sensing to science and operational projects in the polar regions. He has been the Manager of Polar View activities in the Antarctic since 2004, which develops and delivers near-real-time sea ice information to users operating in both polar regions. He also plays a key role in the related Copernicus Marine Environment Monitoring Service, plus EU ExtremeEarth and ArcticPASSION projects, and in the ESA Polar Thematic Exploitation Platform activities. He is a member of the BAS Artificial Intelligence Lab where he works on methods to detect and track icebergs. As part of his work he has participated in numerous field campaigns in the Arctic and Antarctic.

Andreas Cziferszky is a Geospatial Systems Architect at the British Antarctic Survey (BAS) in Cambridge, UK, and holds a M.Sc. in Geodesy and Geospatial Information Systems from the Technical University in Vienna, Austria. He has a wide-ranging understanding of satellite remote sensing and specializes in the design and implementation of operational applications for polar regions. Andreas provides considerable expertise in the design and implementation of information delivery services and the architecture of EO exploitation and processing platforms, focusing on the customization of open source solutions. Over the last 10 years, he has contributed to numerous EU FP7, H2020, ESA, UK Space Agency research projects. He leads the BAS component of the ESA-funded Polar Thematic Exploitation Platform. Andreas' expertise is supported by numerous fieldwork campaigns in the Antarctic and the Arctic.