International Journal of Geo-Information

MDPI

Article

Geoweaver: Advanced Cyberinfrastructure for Managing Hybrid Geoscientific AI Workflows

Ziheng Sun [1], Liping Di [1,*], Annie Burgess [2], Jason A. Tullis [3] and Andrew B. Magill [4]

[1] Center for Spatial Information Science and Systems, George Mason University, 4400 University Dr, Fairfax, VA 22030, USA; zsun@gmu.edu
[2] Earth Science Information Partners (ESIP), Raleigh, NC 27612, USA; annieburgess@esipfed.org
[3] Department of Geosciences and Center for Advanced Spatial Technologies, University of Arkansas, Fayetteville, AR 72701, USA; jatullis@uark.edu
[4] Texas Advanced Computing Center, The University of Texas at Austin, Austin, TX 78712, USA; amagill@tacc.utexas.edu
* Correspondence: ldi@gmu.edu; Tel.: +1-703-993-6114

Received: 24 December 2019; Accepted: 17 February 2020; Published: 21 February 2020

Abstract: AI (artificial intelligence)-based analysis of geospatial data has gained a lot of attention. Geospatial datasets are multi-dimensional; have spatiotemporal context; exist in disparate formats; and require sophisticated AI workflows that include not only the AI algorithm training and testing, but also data preprocessing and result post-processing. This complexity poses a huge challenge when it comes to full-stack AI workflow management, as researchers often use an assortment of time-intensive manual operations to manage their projects. However, none of the existing workflow management software provides a satisfying solution on hybrid resources, full file access, data flow, code control, and provenance. This paper introduces a new system named Geoweaver to improve the efficiency of full-stack AI workflow management. It supports linking all the preprocessing, AI training and testing, and post-processing steps into a single automated workflow. To demonstrate its utility, we present a use case in which Geoweaver manages end-to-end deep learning for in-time crop mapping using Landsat data. We show how Geoweaver effectively removes the tedium of managing various scripts, code, libraries, Jupyter Notebooks, datasets, servers, and platforms, greatly reducing the time, cost, and effort researchers must spend on such AI-based workflows. The concepts demonstrated through Geoweaver serve as an important building block in the future of cyberinfrastructure for AI research.

Keywords: geospatial artificial intelligence; cyberinfrastructure; geoprocessing workflows; provenance; full stack workflows; geographical information systems (GIS); remote sensing

1. Introduction

Artificial intelligence (AI) is profoundly reshaping the scientific landscape [1]. Adoption of AI techniques has brought a substantial revolution in how we collect, store, process, and analyze legacy and current datasets [2–5]. The beneficial features of AI, such as efficiency in execution time, resource usage, and automation, have made these techniques more attractive to researchers than traditional scientific analysis routines, and many scientists/engineers are considering or already using AI techniques in their research and industrial projects [6–8].

The internal mechanisms of AI models are not magic. It is a combination of hardware, software, and algorithms that simulates typical behaviors of the human brain when confronting problems. Machine learning is a popular AI approach and has been proven effective in solving many computer science problems. Much of the core algorithms are not new (e.g., since 1980s) [9–11]. However, only

in recent years has the advancement of hardware like GPU (graphics processing unit), CPU (central processing unit), memory, and storage disks allowed machine learning models to digest big data in a relatively short time [12,13]. The lower computational time costs and drastically improved results have spurred the wide use of machine learning models, with application in many science disciplines, from speech recognition [14]; driverless cars [15]; to smart cities [16]; intelligent homes [17]; and many other fields such as agriculture, environment, manufacturing, transportation, and defense [18–20]. Popular media illustrates an impression that machine learning is constantly changing our way of living, working, and relaxing from many aspects.

The application scope of machine learning is expanding rapidly in geospatial research. In recent studies, machine learning has been proven effective for extracting information from the immense amount of data collected within the geosciences [21–25]. The advantages of machine learning can support our efforts to develop the best possible predictive understanding of how the complex interacting geosystem works [23]. Bergen et al pointed out that the use of machine learning in geosciences could mainly focus on (1) performing the complex predictive tasks that are difficult for numeric models; (2) directly modeling the processes and interactions by approximating numerical simulations; and (3) revealing new and often unanticipated patterns, structures, or relationships [23]. In the foreseen future, machine learning will play a key role in the effort to better understand the complex interactions among solid earth, ocean, and atmosphere.

Many research investigations have been conducted on geospatial datasets in all the three categories. For instance, a team of Australian scientists created the first digital map of seabed lithologies by analyzing ~15,000 samples of sediments found in marine basins [26]. They used the support vector machine (SVM) model, one of the most-used machine learning models, to create the map (Figure 1), which formerly required much effort by many people. Besides SVM, there are many other machine learning techniques (e.g., decision tree, random forest, naive Bayes, neural networks) to implement artificial intelligence. Deep learning, a popular subset study area of machine learning, studies the neural networks with deep hidden layers (e.g., hundreds of layers) or wide hidden layers with many neurons. Typical models of deep learning include feedforward deep neural networks [27], convolutional neural networks [28], recurrent neural networks [14], and so on. Deep learning has been actively used in geoscientific studies [7,19]. For example, Li et al proposed a stacked autoencoder network that inherently considers spatial and temporal correlations and proved their model could predict the air quality of all stations simultaneously [29]. To address the challenges in cloud-resolving models on representing subgrid processes, deep learning is used for short-term simulations at a low computational cost [30]. Rasp et al trained a deep neural network to represent all atmospheric subgrid processes in a climate model by learning from a multiscale model [30]. Their results demonstrated the feasibility of using deep learning for climate model parameterization.

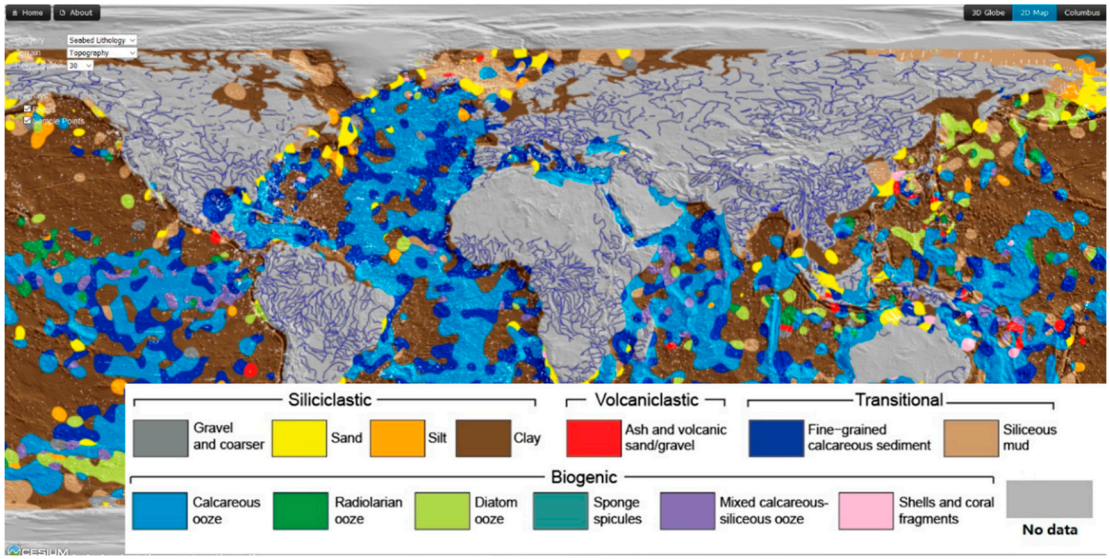

Figure 1. Seabed lithology map by support vector machine (SVM) (screenshot from gplates website (https://portal.gplates.org/cesium/?view=seabed)).

However, scientists also point out that current AI techniques alone are still unable to fully understand Earth's complexity [7]. Studying complex systems like climate and the environment calls for a hybrid approach of AI and numeric physical modeling, which presents challenges for the integration of AI in geosciences. Combining these techniques is not straightforward. Managing and streamlining hybrid AI workflows is presently a very challenging task.

It has been suggested that there are five general phases (https://skymind.ai/wiki/machine-learning-workflow) in a machine learning workflow: (1) objective definition—identifying a geoscientific problem and proposing a potential machine learning solution; (2) feasibility study—analyzing the risk of failures and bottleneck issues; (3) model design, training, and lab evaluation [31,32]; (4) model deployment, in-production evaluation, and monitoring [33]; and (5) model maintenance, diagnosis, share, and reusability. In phase 3 and 4, the processing workflow of AI includes not only AI algorithms, but also preprocessing and post-processing (https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94). The current AI processing is done by a combination of software, scripts, libraries, and command-line tools. AI practitioners often maintain their workflows relying on their own unique methods, which are commonly created without much priority given to accessibility or ease of use and understanding by others. As many disparate and distributed entities are involved, it becomes a challenge to streamline all the processes to help scientists to organize their deep learning projects into a manageable and clear manner. Meanwhile, it is still difficult to share and reuse the created AI workflows and results among the geoscientific community, which results in low efficiency in AI model training and application in geosciences. Because of the lack of the details about workflow and platform environments, it is difficult for scientists to reproduce the results when they have no access to the original AI workflows.

This paper proposes a scientific workflow management framework to address these AI workflow-related issues and a prototype system named Geoweaver for validation. We tested the system with a use case applying deep learning methods to agricultural land cover mapping. The task involves three distributed servers, GPUs, three programming languages, dozens of command tools, a geographical information system (GIS), a deep learning toolkit, and multiple public data sources. Geoweaver allows practitioners to manage all these entities in one place and record the provenance of every execution in a separate database for future examination. The results prove that the framework can bring great benefits to the AI community to build, run, monitor, share, track, modify, reproduce, and reuse their AI workflows in either a single-machine environment or a distributed environment.

The existence of Geoweaver could serve as a fundamental tool for the future AI workflow management and boost the practical application of AI in the geosciences.

2. Big Spatial Data and AI

AI models are a combination of universal mathematical equations. Theoretically, they can be used on any kind of datasets and fit in most scientific scenarios. One prerequisite of a successful AI model is a big, less biased, accurate, complete training dataset, which requires continuous and large-scale raw datasets [34]. An ideal training dataset collected in the previous century is often difficult to find, as 90% of the current data in the world were created in just the past few years (https://www.xsnet.com/blog/bid/205405/the-v-s-of-big-data-velocity-volume-value-variety-and-veracity). Every day, huge amounts of data are collected from mobile phones, cameras, laptops, satellites, and sensors from around the world and stored by data facilities under the management of tens of thousands of institutes, organizations, agencies, and individuals [35]. The term "big data" describes the large volume of data, both structured (clearly defined data types) and unstructured (everything else) (https://www.datamation.com/big-data/structured-vs-unstructured-data.html), accumulated by businesses on a day-to-day basis. Scientists summarized big data challenges into several "V"s, including volume, velocity, variety, veracity, and value [36]. Data owners, providers, or keepers have invested significantly to develop solutions that address these challenges [37]. The availability of big data is good news for AI applications, however, extracting a training dataset from big data presents a set of associated challenges [34].

In 2012, the U.S. government announced big data initiatives of more than $200 million in research and development investments for National Science Foundation [38]. Fundamental research innovations have been made in both the upstream and downstream of data, for example, data model, storing, query and transferring strategies, computing paradigm, and data infrastructure [39], which have resulted in the creation of a group of software tools, libraries, and hardware devices. Some commonly used tools include Apache Hadoop [40], Apache Spark [41], HBase [42], Hive [43], MongoDB [44], Google Earth Engine [45], Amazon S3 [46], Amazon EC2, Jupyter Notebook [47], and so on. These tools together have enabled the efficient processing and analysis of big datasets collected from the entirety of human society. Practitioners of the geosciences have been early adopters in the field of big data and AI-assisted analysis, with many instances of the use of big data techniques successfully addressing geographic information science problems [48–51]. Today, both the industrial and academic sectors are embracing big data challenges and developing new theories and technologies for information discovery from big spatial data. Many traditional analysis methods created for use on small-scale data and requiring manual processes are unable to cope with the increasing volume of geospatial data available today. It is a big challenge for conventional GIS operations to process such abundance of spatial datasets and deliver timely information to stakeholders [52]. AI techniques present a promising automatic solution: AI does not require manual processes and can digest large-scale data very quickly using modern hardware. The most important thing is that the tremendous live data traffic will not affect AI normal operation on a daily basis. From every aspect, AI seems to be an ideal solution for the current big spatial data issues.

However, the spatiotemporal features of geospatial datasets often create more difficulties for the AI models to successfully learn and predict. On the basis of our previous experiences, we conclude five reasons that spatiotemporal data require additional attention:

(1) ***Difficulty in calibrating training datasets***: This problem often occurs in remote sensing data. The spatial resolution determines the size of each pixel and how much land it covers. For instance, if AI models use remote sensing data as inputs and ground-collected data as outputs, the accuracy can be critically affected if images are slightly shifted, resulting in ground sample points matching the wrong pixels. Appropriate project transformation and resampling algorithms are required to ensure the location match between input and output is right. Otherwise, the model will be very hard to converge and the trained models are useless.

(2) ***Difficulty in synchronizing observation time***: Most spatial information is also time-sensitive. The observation time is important to associate observed events with triggering causes and later potential consequences. Dealing with spatial data must keep in mind the data's exact observation time, approximate observation period, and time zones (if the dataset is regional). In many studies, high temporal resolution granules are processed into more coarse resolution products, for example, daily, weekly, biweekly, monthly, or annual products [32]. The time processing could use maximum, minimum, average, standard deviation, and even some customized algorithms.

(3) ***Difficulty in reducing the bias in training datasets***: Most spatial datasets are naturally biased. For instance, in the North Dakota agriculture, the growing acres of canola are much smaller than soybean, creating bias in datasets that contain more soybean samples than canola. Bias creates problems for AI training, as AI will get much better accuracy on unbiased datasets. Extra processes need be done to reduce the bias in the training dataset such as batch normalization or restraining the representation sample numbers of each category in the training datasets [53]. However, one should be aware that bias is not the only reason for poor fitting performances and reducing bias might cause the trained model to underfit in the major classes and overfit in the minor classes. Scientists are still looking for solutions to balance between major and minor class samples.

(4) ***Difficulty in treating data gaps***: Data gaps caused by mechanical issues, weather, cloud, and human reasons are very common in long-term observatory datasets. A typical example of mechanics failure is the gap on the Landsat 7 imagery caused by the Scan Line Correction failure since 2003. Clouds are the major reason blocking satellites from observing the land surface. Misconduct by device operators can also cause gaps in the datasets. Data gaps may lead to missing information of key phenomenon and make AI models unable to capture the patterns. There are several proposed solutions to fill the gaps, but require human invention and take a long time, which is not very efficient for big data.

(5) ***Difficulty in dealing with spatiotemporal data fuzziness***: Fuzzy data, which are datasets using qualitative descriptions instead of quantitative measures, are everywhere [54]. Social media text [55], for example, will give a fuzzy location like "San Francisco" and fuzzy time like "yesterday" about some observed event. Spatiotemporal analysis normally requires a precise location (longitude/latitude) and time (accurate to hours/minutes/seconds) [56]. Feeding fuzzy information into AI models might make the models even more inaccurate. How to deal with fuzzy spatiotemporal data is also an important issue faced by AI today [57].

In summary, the location and time information inherent in spatiotemporal data, and the high dimensionality of these data, require additional attention by AI engineers compared with other traditional use cases. The difficulties enumerated above, present even when dealing with small volumes of data, become more serious when big data are involved. The main bottleneck of using AI in spatial data is the compatibility between the requirements of AI models and the complicated structure of big spatial data. To address these issues, specialized tools and software are essential, and may be better developed under open and sharing terms and conditions, so that the entire big data community can contribute to solutions, while avoiding private unique non-interoperable data types, interfaces, and structures. This work is one of these efforts to propose an open source software named GeoWeaver as a solution to the mentioned challenges. In the software, GeoWeaver integrates the capabilities of various existing libraries and software, for example, GDAL (geospatial data abstraction library) /GRASS (geographic resources analysis support system) for calibration; USGS (United States Geological Survey) EarthExplorer Python client to get the observations within the same period; Python scikit-learn package for clustering and smoothing the data gaps, among others; and CUDA (compute unified device architecture) DNN (deep neural network) for high performance model training, and so on. Compared with existing workflow management software (WfMS), GeoWeaver could better couple these heterogenous softwares into an automated workflow with less manual intervention and intuitive one-stop user experiences.

3. Workflow Management Software

Workflow management software (WfMS) is one of the urgently required software for big data AI application, and maybe the most important one in the foreseeable future [58–62]. In this section, we investigated the state-of-art of workflow systems that might be usable for managing AI workflows. "Workflow" is very general term and may be interpreted in many ways. For example, many Earth scientists often refer a Jupyter Notebook or a bash script as a workflow [47]. GIS users call the chains in ArcGIS ModelBuilder a workflow [63]. Businesspeople refer to their production lines as a workflow [64]. In the web environment, web service chains are workflows [59,65,66]. None of these interpretations are incorrect. Despite the big differences in the apparent forms, the so-called workflows by various groups of people share several common key components:

3.1. Atomic Process

As the fundamental elements in a workflow, atomic processes are the tasks that act on and in some way transform or create data. Most traditional workflow management systems expect all the atomic processes in a workflow to conform to specific predetermined types. For instance, the ArcGIS ModelBuilder allows the tools from ArcGIS (including customized tools) as atomic processes [63]. The Oracle business process execution language (BPEL) process manager can compose web services with SOAP-WSDL (Simple Object Access Protocol – Web Services Description Language) [67]. Information about other WfMS can be found in Table 1. Scientific software complexity in the context of geosciences breaks this conformity, with atomic processes of variable functionality and granularity. For example, Python has become the dominant language and environment for AI workflows [68]. AI workflows written in Python involve many calls of python libraries and distributed Jupyter Notebooks. Most of the computations must be completed remotely on systems with GPUs installed or capable of executing highly parallel codes. Shell scripts and command lines are another two important atomic processes for transforming big spatial data into an AI-ready format and post-processing results into understandable value-added products. The granularity of atomic processes in AI workflows is constantly changing according to the personal preferences and habits of the workflow authors. Practitioners may tend to encapsulate all the business logic into one huge atomic process, while some others prefer to itemize the business logic into smaller discrete tasks to attain more granularity in the workflow progress. WfMS should be designed to be flexible to grant the freedom of changing the granularity of atomic processes to AI engineers.

Table 1. Supported atomic processes by existing workflow management software (WfMS).

Name	Atomic Process	Workflow Language	License
ArcGIS Model Builder	ArcGIS toolbox	Self-Defined	Commercial
QGIS Processing Builder	GDAL, QGIS, GRASS, SAGA	Self-Defined	GNU GPL (General Public License)
Apache Taverna	Local Java code SOAP web services RESTful services R processor Shell scripts Xpath scripts	SCUFL2	Apache v2.0
Kepler	Web services Unix commands Shell scripts	Kepler Archive	BSD (Berkeley Software Distribution)

Table 1. *Cont.*

Name	Atomic Process	Workflow Language	License
Cylc	Shell scripts	Directed Acyclic Graph	GPL v3.0
Galaxy	Built-in bio process	Gxformat2	AFL (Academic Free License)
Pegasus-WMS	Local shell scripts Built-in processes	DAX	Apache v2.0
Apache Airflow	Bash, Python	Directed Acyclic Graph	Apache v2.0

3.2. Function Chain

Simply speaking, a workflow is a bunch of linked processes [69]. Some processes (downstream process) depend on the outputs of other processes (upstream process). This dependency is the main driver of the workflow composition. The links among the processes are normally called "data flow". Once the processes are chained together, WfMS will allow users to execute the workflow automatically and check the real-time status of the execution, which will require much less human intervention compared with manually executing the separate processes individually. To facilitate reuse and replication, the WfMS communities, for example, Workflow Management Coalition (WfMC) [70], World Wide Web Consortium (W3C) [71], Organization for the Advancement of Structured Information Standard (OASIS) (https://www.oasis-open.org/standards), myExperiment [72], Galaxy [73], and Kepler [74], proposed a series of workflow languages that describe and record these workflow links and the involved processes. The standard languages commonly used in the industrial sector include BPEL (business process execution language), BPMN (business process model and notation), common workflow language (CWL), and so on. For scientific workflows, most WfMS define their own languages, such as Taverna SCUFL2 (Simple Conceptual Unified Flow Language), YAWL (Yet Another Workflow Language), Kepler, and so on. These workflow languages provide abstractions and information models for processes. WfMS uses these abstractions and models to execute the corresponding workflows. However, hundreds of WfMS have been developed, but only a handful supports the common workflow language. Most created workflows cannot be shared among different WfMS. Therefore, when choosing WfMS, one needs to be very cautious because the course of transitioning workflows to another WfMS later would be difficult.

3.3. Error Control

Another important objective of WfMS is error control. Big data processing may execute for days or even months before desired results are achieved; it is inevitable that exceptions will occur during such a long computational duration. Any design of WfMS must anticipate and be prepared to handle the occurrence of errors in workflow execution. Owing to the common occurrence in big data processing environments that components are written in different programming languages and run in distributed virtual platforms, this can be a difficult challenge to realize and a non-trivial solution is required [75]. Strategies for error control are generally comprised of two steps: error capturing and error handling. WfMS must monitor the status of each individual process to capture the possible error events. This requires that the individual processes have a channel for WfMS to observe their real-time status (especially if the process will last very long time). Once an error occurs, the WfMS may handle it in one of three ways: (1) stop the entire workflow, (2) rerun the process, or (3) skip the failed process to continue. In the big data case, one recommended solution is to only rerun the failed processes to avoid duplicated time waste on the upstream successfully executed processes; however, this may depend on specific cases. For those common errors such as network disruption, memory leak, time out, or hardware failures, the WfMS should provide some predefined solutions to decrease the necessary interaction of users. In recent studies, many WfMS attempt to realize automatic error detection and recovery without human intervention. This may be achievable for basic errors, while for most high-complexity errors (in debugging phase), human analysis and operation is still required.

3.4. Provenance

Given the complex architectures common in machine learning, for example, in neural networks with sometimes hundreds or thousands of layers weighted with parameters that are not humanly interpretable, AI techniques are contributing to a widening crisis in repeatability, replicability, and reproducibility [76–80]. A key, but often misunderstood enabling concept in these issues is that of provenance [81–84]. High quality provenance records contain process and pedigree information about what, how, where, and when the geospatial or other data are derived. The records of components, inputs, systems, and processes that provide a historical context all belong to what may be termed retrospective provenance. Similar prospective provenance records may be used for possible future computational activities (refined workflow execution). Geospatial data are inherently problematic in terms of replicability (e.g., because of spatial heterogeneity [85]), and the corresponding need for high quality geospatial provenance information appears to be very high.

Provenance is extremely important in the world of data and geospatial data production and its benefits go beyond replicability-related challenges to include quality, transparency, and trust [82,83,86,87]. However, there are serious unanswered questions about potential conflict between provenance, privacy, and intellectual property. Furthermore, development of provenance standards for AI workflows has not reached maturity. At a large scope, W3C has released a recommendation for a web provenance data model called PROV-DM (Provenance Data Model) [71], which defines a basic universal core data model for almost all kinds of the provenance information. However, there is uncertainty about using such a recommendation to record the provenance generated by AI workflows that incorporate big geospatial data. More research is also needed to solve data representation issues associated with provenance compression, storage, query, retirement, and so on.

3.5. Big Data Processing Workflow

Training AI models requires big training datasets, which are derived by big data processing workflow. However, managing big data processing workflows is much more complicated than managing conventional small-scale workflows. This section reviews the current status of big data processing workflow management and analyzes the challenges we are facing today.

Many big data processing frameworks have been used in response to carrying large-scale computational experiments in geoscience [88]. Scientific workflows are used to combine traditional high-performance computing with big data analytics paradigms [89]. Computational-intensive scientific experiments are normally conducted in large data centers of national laboratories such as DOE (Department of Energy) Oak Ridge National Laboratory, San Diego Supercomputer Center, National Center for Supercomputing Applications in University of Illinois at Urbana-Champaign, and CISL (Computational and Information Systems Lab) Supercomputers in NCAR (National Center for Atmospheric Research). These supercomputer centers provide users with some client tools to manage their experiment workflow remotely, such as Cylc [90]. Owing to security and maintenance reasons, supercomputers can provide a limited environment for users to incubate their experiments.

Cloud computing is another popular paradigm for large-scale data processing today [91–93]. It is based on resource sharing and virtualization to provide a transparent, scalable, and elastic environment, which can be expanded or reduced on demand [94]. With cloud computing, clustered virtual machines, as well as big data processing frameworks like Hadoop and Spark, may be instantiated and installed to create a private big data processing architecture with little effort, and avoiding the need to enter into a queue for resource allocation [48]. The big data software tools can automatically enforce data locality and minimize data transfers, and allow scientific workflows to be executed in parallel. For instance, suppose there is a task to average the petabyte data of global precipitation from hourly to daily. The WfMS can help organize all the involved processes in the ETL (extract, transform, load) routine of MapReduce workflow [51,95]. Big data are first digested and split into data batches in distributed file systems like HDFS (Hadoop Distributed File System) [40]. Each batch is going to be mapped into key/value pairs, and fed into reduced nodes for applying predefined algorithms. The processing of all

the batches is carried out in parallel. The results are further reduced into tuples and stored in the HDFS. People can use tools like Apache Hive to query data from the HDFS using a language that is very similar to traditional SQL (Structured Query Language). Apache HBase is a popular NoSQL database that runs on top of HDFS and supports Hadoop to handle large volumes of structured, semi-structured, and unstructured data [42]. WfMS are mainly used for managing data ingest and data pipelines.

There are many other big data processing frameworks in addition to the Apache family [96]. For scientific workflow, Python plays an important role today. The ecosystem of numerous Python libraries allows scientists to manage their experiments just using Python code. Pangeo community is one of the largest fruits by the groups who try to use Python libraries to meet the big data challenges in geosciences [97]. Pangeo is not a software, but a Python environment in which people will get all the required libraries for processing large scale datasets, for example, global ocean remote sensing database. The core libraries for parallel processing are Xarray and Zarr. Running on cloud platforms or supercomputers, the Pangeo environment [97] has shown its feasibility in solving large-scale geoscience problems via processing a huge amount of Earth observations accumulated in the past decades.

4. Framework

To meet the challenges above, we propose a novel framework for managing large-scale AI-based geoprocessing workflow. The framework is designed to help scientists to sort out their AI experiment and improve the automation and reusability. As shown in Figure 2, the framework is based on the existing software ecosystem of AI and big spatial data by integrating many mature systems together in one place for users to call. The core design is divided into three modules according to the entities they deal with; that is, host, process, and workflow. The details are introduced below.

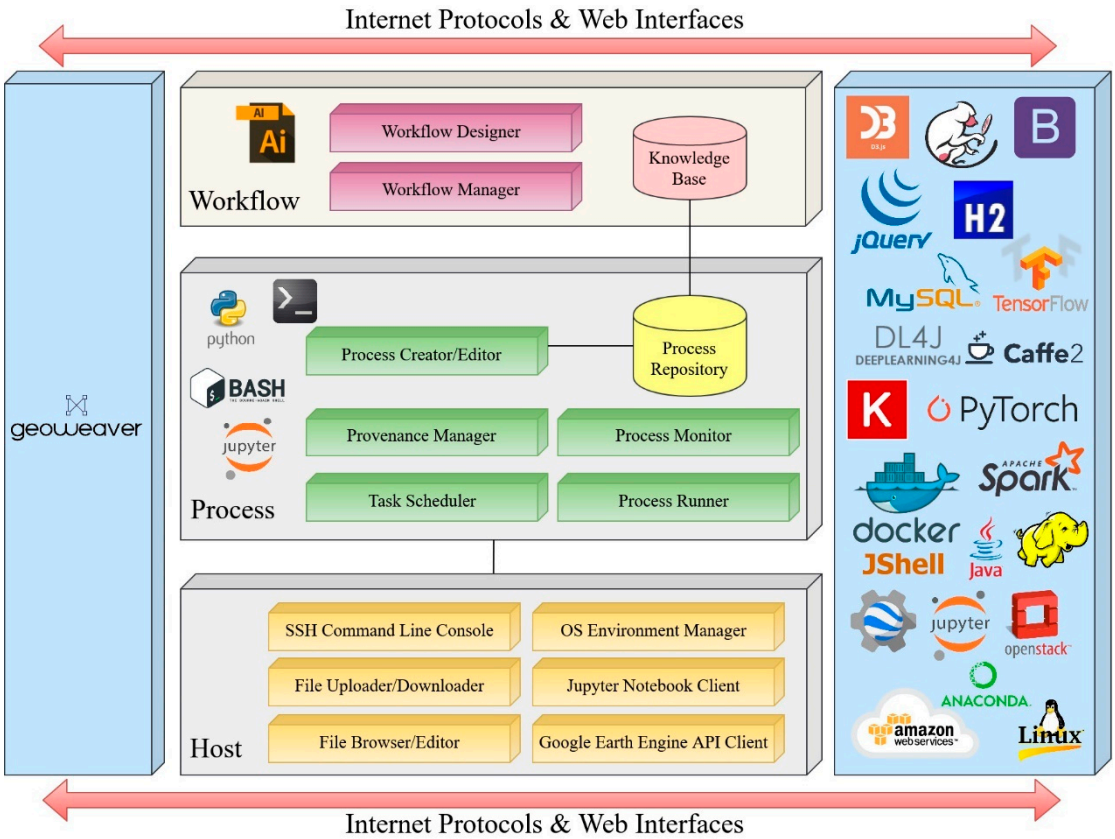

Figure 2. The proposed artificial intelligence (AI) workflow management framework. API, application programming interface. (OS: Operating System)

4.1. Host

This module is the foundation of the entire framework and the major difference from the other WfMS. It opens the entry to the existing resources like servers, virtual machines, Jupyter server instances, and third-party computing platforms such as Google Earth Engine and Google Colab. This module integrates the API (application programming interface) client to enable the manipulation of the targeted resources. To facilitate quick and easy adoption of the platform, an interface that is familiar to users accustomed to common existing tools is recommended. For example, this module can include an SSH (Secure Shell) console to access and type in command lines to remote servers (including physical servers or virtual machines provided by cloud platforms), a file browser, and code editing windows. The host module also provides file transfer services, allowing for file upload from local to remote servers, and file download from remote servers to the local computer.

Another important designed function is to take over the management of the system environment. The system environment is primarily the set of variables that define the current settings for the execution environment. System environments are very complicated owing to the variety and version compatibility of dependent libraries. Setting the path and environment variables will differ depending on the version of operating system of the machines. Environment variables often differ between systems owing to discrepancies in installation locations of dependencies, a major cause of problems and often a barrier to replicating an experiment on a new machine. To better manage environments, software like package managers, such as conda or venv for Python, commonly used with the distribution of packages, have been developed. A virtual environment is a directory that contains a specific collection of Python packages. Via these management tools, the environment can be restored by the host module so that the program can be successfully executed. An alternative to independently manage all the environment variables is using containerization technologies (Docker and Kubernetes). Containers can provide a production-ready environment for deploying applications in no time by eliminating the need to install, manage, and operate the environment dependencies.

The host module also should be able to interact with third-party platforms like Jupyter Notebook server and Google Earth Engine. Using their open API, host module can send messages to start processes on those platforms. The function allows this framework to integrate with powerful high-performance computing platforms in the public domain. Users can smoothly transfer their workflows from these platforms to this framework.

4.2. Process

The process module includes five submodules and one database. The supported atomic processes should include the widely used scripts, programs, commands, or code. As the current AI experiments most commonly employ Python, the process module should at a minimum support Python, Jupyter Notebook, shell scripts (i.e., Bash), and SSH for running system-level programs. The AI libraries like DeepLearning4j, Keras [4], PyTorch [98], Caffe, Tensorflow [99], and so on are directly accessible in the Python or Jupyter processes. A process creator/editor is a dialog in which users can create new processes or edit old processes. There should be a text area for rendering Shell scripts, Java/Python code snippets, and software commands with colors for intuitive reading. The newly created processes should be stored in a database (MySQL, or H2). A task scheduler is responsible for scheduling the process execution requests in the waiting list and assign the tasks with appropriate platforms from the resource pool. A process runner is responsible for binding via SSH channels and executing processes on assigned platforms. The execution needs to first transfer the latest code to the remote servers and trigger the code by sending commands. The process monitor can listen to all the events occurred during the execution and report the real-time status. Once the execution is over, the result messages together with the inputs and executed code shall be saved into the database for later inquiry. A provenance manager is responsible for querying and analyzing the stored history information of every processes execution to evaluate data quality or recover the process execution from failures.

4.3. Workflow

The workflow module provides two major functions: (1) composing workflows out of atomic processes from the process module; and (2) managing the query, edit, execution, monitoring, and retirement of workflows. Like other graphical workflow systems, the workflow creator should have a graphical panel as the work canvas for drag-n-drop processes and linking them into a workflow. An interface should be developed to allow users to compose workflow. There are many recent developments in JavaScript frameworks, namely React, Angular, and Vue.js, as well as JavaScript libraries such as D3.js and CodeMirror, that can aid in building websites to provide a rich user experience through fully feature user interfaces. Via the interfaces, people can drag and drop the atomic processes and draw lines to connect them into workflows. The created workflows represent the knowledge of large-scale data processing and are stored in the database. A toolbar should be developed to manage the workflows, including searching the existing workflows, running and monitoring the workflows, and checking the history of workflow execution. The workflow module provides real-time feedbacks of the member process statuses by indicating with different colors or showing progress bars. To make the workflow knowledge sharable, this module should also allow people to export and import (or upload and download) the workflows. User should be able to export their workflows from old instance and import their workflows into a new instance where they can directly work as usual.

5. A Prototype: Geoweaver

We have developed a prototype named Geoweaver [100,101] to validate the framework. The interface design is displayed in Figure 3. It simply consists of a workspace canvas, right-side menu panel, navigation toolbar, workflow toolbar, and log out window. On the right-side menu panel, there are three folders: host, process, and workflow. Each folder has child nodes. In the host folder, each child node is a machine, either a physical remote server or a virtual machine. Each machine node follows four buttons: SSH console button, file uploading button, file browser/downloading button, and host deleting button. In the process folder, there are four child folders: shell scripts, python code, Jupyter notebooks, and built-in processes. Each folder contains the processes in the same language or format. The execution mechanisms for different folders are different.

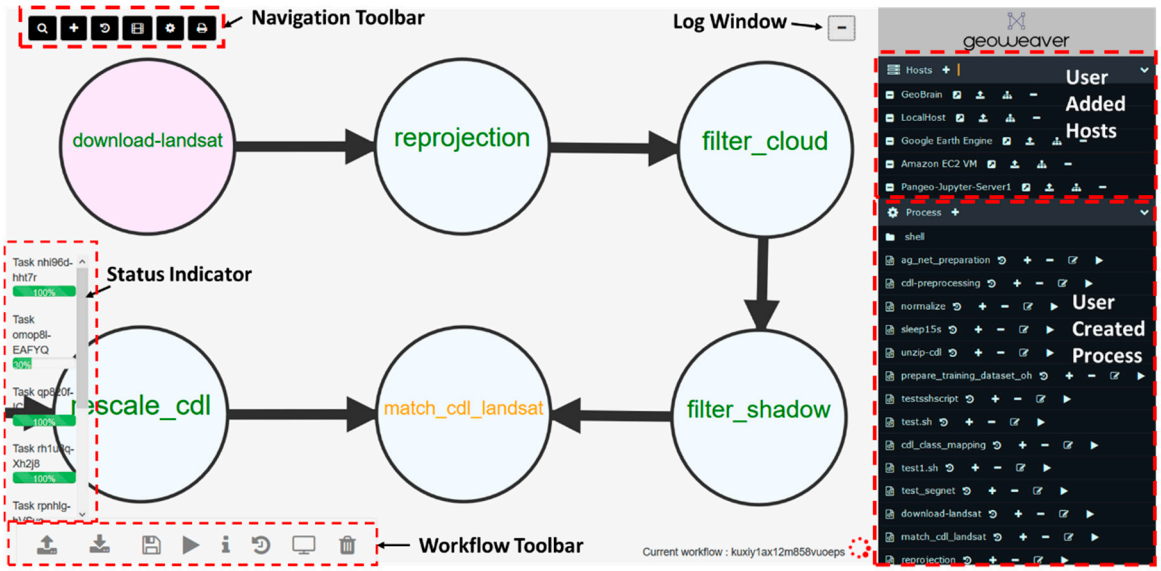

Figure 3. Geoweaver.

The Geoweaver software was initially funded by ESIP (Earth Science Information Partners) Lab Incubator program and has been released as an open source software on Github (https://esipfed.

github.io/Geoweaver). It uses an MIT license, which is very permissive for both commercial and non-commercial reuse.

6. Use Case: AI-Based Agriculture Mapping

We experimented Geoweaver by our current research of using Landsat images and deep learning classification algorithms to study land cover changes and corresponding socio-economic influences [3,5,102–107]. The AI workflow has already been published in the work of [5]. We replicate the workflow in GeoWeaver and run it on new test data and successfully reproduced new crop maps [101] (Figures 4 and 5).

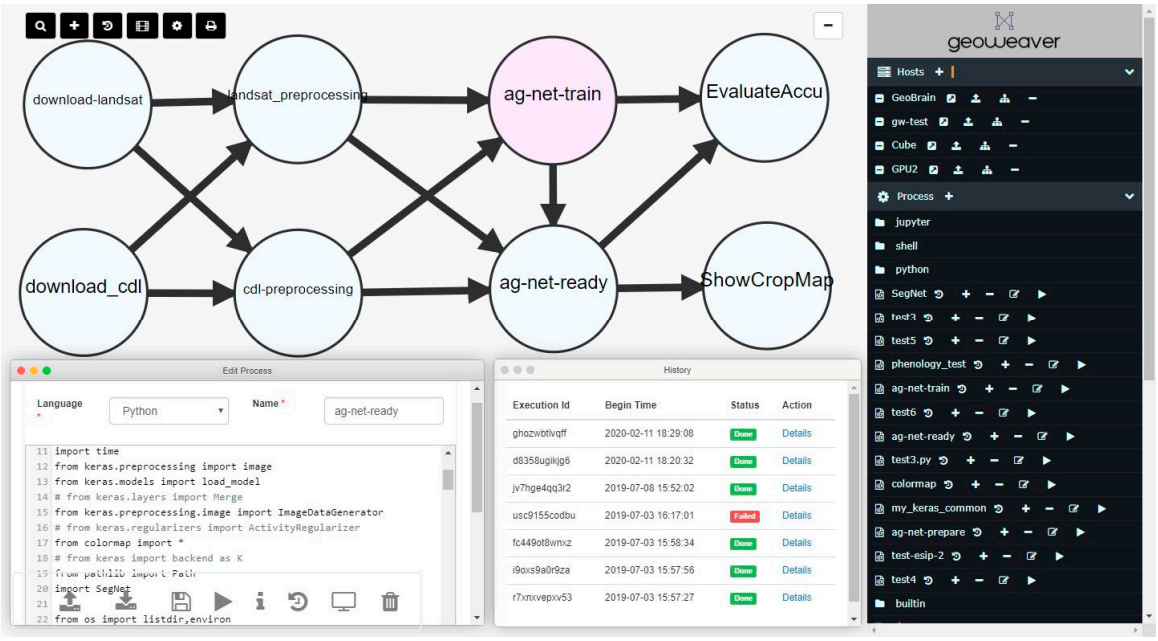

Figure 4. The created GeoWeaver workflow for crop mapping.

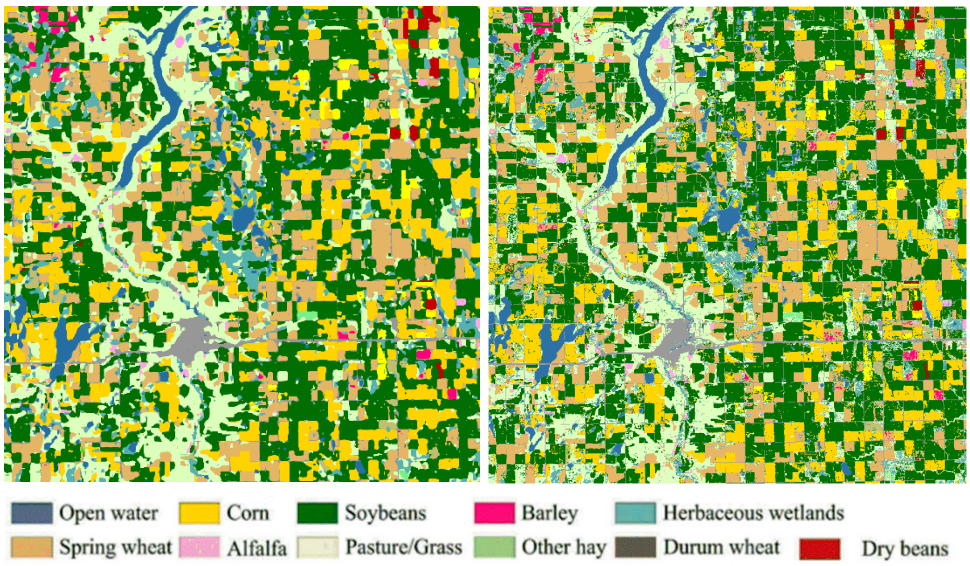

Figure 5. GeoWeaver-reproduced crop map (left) compared with United States Department of Agriculture (USDA) map (right).

First, we downloaded the cropland data layer (CDL) from USDA (United States Department of Agriculture) NASS (National Agricultural Statistics Service) [108] as reference data to predict the unknown areas and periods. In this workflow, LSTM (long short term memory) [109] is utilized as the classification method. The Landsat and CDL are first preprocessed into small tiles of images (example data could be found at https://github.com/ZihengSun/Ag-Net-Dataset). There are several tutorial Ag-Net Jupyter notebooks (https://github.com/ESIPFed/Ag-Net). We leveraged cloud computing (GeoBrain Cloud) [110–112] and parallel processing (NVIDIA CUDA GPU) [93,113] to meet the challenge of processing tremendous number of pixels in remote sensing images (a single Landsat 8 scene contains more than 34 million pixels) [114]. The classified maps will help agriculture monitoring, prediction, and decision making [107,115]. The entire experiment poses too many management issues for scientists to handle. We must contend with difficulties in maintaining organization and personal communication, hardware communication constraints, and vexing configuration problems. We keenly need a management software solution to help sort out the steps and processes, and provide us with an overview dashboard to operate, manage the underlying facilities via the Internet, and track down issues and provenance. In our case, we need it to serve functions to create an intuitive compound workflow for the LSTM-based classification from raw images to land cover maps, run the workflow on GeoBrain Cloud with Tensorflow and CUDA [4,99], track the provenance of each map [116], and share the results with other scientists via Email or social media. The results show the workflow in GeoWeaver can successfully classify the Landsat images accurately compared with the USDA crop maps (Figure 5).

In the experiment, Geoweaver helped us to flexibly adjust the granularity of atomic processes in the workflow. We redesigned the workflow for better management by combining all the processing steps of Landsat images together into one atomic process “landsat-preprocessing”, and combining all the preprocessing steps of CDL into “cdl-preprocessing”. We merged the code into one Shell process that has larger granularity. The larger process will make it easier to cope with the situations like data source change, for example, changing the input images from Landsat 8 to Landsat 5 (band specifications are different). Via GeoWeaver, the merge/decomposition of code could be easily done by creating new processes and splitting a large step into snippets or merging tedious steps into a large process. Compared with other WfMS, Geoweaver provides more flexibility on allowing users to resize the granularity of atomic processes.

Geoweaver also helped us in migrating the workflow around without rewriting anything. We have deployed multiple instances of Geoweaver on GeoBrain Cloud in the case of network limitations or accidental situations like power outage and server collapse. One Geoweaver instance has no access to GPU server and one Geoweaver instance does. Initially we created the workflow on the non-GPU Geoweaver instance. To leverage GPU, we download the workflow together with the process code as a package and upload it into the Geoweaver instance with GPU access. It works smoothly on the new instance and the transition is very time-saving compared with manually moving the workflow from one environment to another. This also reflects the fact that Geoweaver reduces the barriers of reproducing the experiment in a new environment.

During the workflow recreation, it was of great convenience that Geoweaver was able to chain hybrid processes written in multiple languages and run them on multiple remote servers. The downloading scripts and preprocessing scripts are written in Shell. The ag-net-train and ag-net-ready and evaluate-results are in Python. The ShowCropMap process is a built-in process of Geoweaver, which retrieves the result files from the remote servers and displays them in visual form in the Geoweaver interface.

The training of neural networks was repeated dozens of times owing to the frequent adjustments in the preprocessing steps and the neural network configuration. Each execution of the workflow triggers the process monitor and provenance manager to start collecting and recording the provenance information. All the successes and failures are saved and available for query via the history button. The history table has a column “Status”, which has four options: *Done*, *Failed*, *Pending*, *Stopped*. Errors in

Geoweaver workflow are properly detected and thoroughly recorded to inform the users. Geoweaver makes them available in a consistent one-stop manner, which saves a lot of wasteful execution of the workflow by analyzing the recorded provenance and tuning the neural network models to optimal status quickly.

The overall data volume in the experiment is 9.5 terabytes (including the original Landsat/CDL data and semi-finished data products). It was boring and a challenge before for the researchers to manually monitor the process running on such a dataset. Geoweaver coordinates between the file storage system and the users via its host module. It allows users to access and manage the files in one place and guides users to focus on the algorithms without being lost in the sea of data files. Big data workflow could be monitored via web browser and becomes more controllable. Geoweaver uses its flexible running module and real-time monitoring module to help us get rid of constant standbys and the fear of GPU memory leak or missing the overfitting/underfitting turning point. In the foreseeable future, high performance computing services (e.g., Google Earth Engine, Hadoop/Spark, Pangeo) will be supported in Geoweaver and further shorten the waiting time in big data processing tasks.

7. Discussion

There are several reasons Geoweaver is a capable and comprehensive scientific workflow management system. Geoweaver is targeting several key problems in the current landscape of AI scientific research. Scientists often find themselves asking the following questions: Where is my code? Where is my data? How can I find my historical results? How do I manage multiple servers, HPCs, and laptops simultaneously? How can I link my experiment into an automatic workflow? AI workflow is complicated and huge. It involves many libraries, tools, systems, platforms, machines, or even super computers. Existing workflow systems fall short on managing multiple programming languages, multiple servers, and multiple datasets. In most scenarios, small group of scientists have very limited choices in research management tools. Geoweaver has a unique combination of five very useful features, which make its experiences more complete than other current WfMS for AI practitioners.

(1) ***Hybrid Workflow***: Geoweaver can help AI practitioners to take advantage of both public resources and private resources and combine them together into one workflow. The training dataset preparation needs a lot of legacy datasets and the programs transforming them into AI-ready format. However, the AI model training facilities are mostly in the public domain, such as Amazon EC2 GPU instances. It is hard to connect the legacy data processing with the AI model training using other WfMS. Geoweaver uses the host module and dynamic computing binding to allow scientists to combine the processes executed on private servers and the public platforms into one workflow and enable the hybrid workflow management in one place.

(2) ***Full Access of Remote Files***: As mentioned above, most files associated with AI workflow are stored on remote servers/virtual machines. Users always appreciate the tools that allow them to have full and convenient control over the actual files, including creating new files, browsing file folder structure, downloading files, and editing files in place. Geoweaver is not only a workflow system, but also a file management system of multiple remote servers.

(3) ***Hidden Data Flow***: Business workflows such as BPEL usually separate the content of workflow into two divisions: control flow and data flow. The former defines the sequence of involved processes, and the latter defines the data transfer among input and output variables. It takes a lot of attention to maintain the data flow once the data are big and the file count is dynamic. Geoweaver can create a familiar environment for people to create the workflows without concern about the data flow. Each process is independent and data flow is taken care of by the process content logic.

(4) ***Code-Machine Separation***: Another feature of Geoweaver is that it separates the code from the execution machine. There are couples of benefits by doing this. The code will be managed in

one place and version control for better code integrity would be much easier. Geoweaver will dynamically write code into a file on the remote servers and execute the code. Once the process is over, Geoweaver will remove the code from the remote servers. Regarding the fact that the GPU servers are usually shared by multiple users, the mechanism will better protect the code privacy from other users on the same machine.

(5) ***Process-Oriented Provenance***: Distinct from data-centric provenance architecture, Geoweaver uses process as major objects to record provenance. The recorded information is also different. In Geoweaver, the inputs are the executed code content, and the outputs are the execution log. Rather than storing partially completed data products, process-oriented provenance can save disk space and enrich the history information of the final data products. Process-oriented provenance can prevent barriers to reproduction of the workflow that would otherwise be caused by changes to the code.

8. Conclusions

This paper proposes a scientific workflow framework to address these AI workflow-related issues. A prototype system named Geoweaver is implemented for validation. We tested the system by a use case of AI-based agricultural land cover classification. The task involves several distributed servers, GPUs, three programming languages, dozens of command tools, geographical information system, deep learning toolkit, and multiple public data sources. Geoweaver makes all these entities manageable in one place and record the provenance of every execution in a separate database for future examine. This study proves that the proposed framework can bring great conveniences to the AI community to build, run, monitor, share, track, modify, replicate, and reuse their AI workflows in either a single-machine environment or distributed environment. The concepts demonstrated through Geoweaver serve as an important building block in the future of cyberinfrastructure for AI research.

Author Contributions: Conceptualization, Ziheng Sun, Liping Di, Annie Burgess; Methodology, Ziheng Sun, Liping Di; Software, Ziheng Sun, Andrew B. Magill; Validation, Ziheng Sun, Annie Burgess, Jason A. Tullis; Formal Analysis, Ziheng Sun, Jason A. Tullis; Investigation, Ziheng Sun, Liping Di, Annie Burgess; Resources, Ziheng Sun, Liping Di; Data Curation, Ziheng Sun; Writing—Original Draft Preparation, Ziheng Sun; Writing—Review & Editing, Liping Di, Annie Burgess, Jason A. Tullis, Andrew B. Magill; Visualization, Ziheng Sun; Supervision, Ziheng Sun, Liping Di; Project Administration, Ziheng Sun, Annie Burgess; Funding Acquisition, Ziheng Sun, Liping Di, Annie Burgess. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by ESIP Lab and National Science Foundation, grant number AGS#1740693 and CNS#1739705.

Acknowledgments: Thanks to the authors of all the open source libraries and software we used in developing Geoweaver. Thanks to our colleagues in George Mason University and many other institutes who gave kind advice on the project development. Thanks to SGCI (Science Gateways Community Institute) for providing consulting services to this project. We greatly appreciate the anonymous reviewers for their constructive suggestions.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
2. Bengio, Y. Deep learning of representations: Looking forward. In Proceedings of the International Conference on Statistical Language and Speech Processing, Ljubljana, Slovenias, 14–16 October 2019.
3. Sun, Z. Some Basics of Deep Learning in Agriculture. **2019**. [CrossRef]
4. Sun, Z. Automatically Recognize Crops from Landsat by U-Net, Keras and Tensorflow. Available online: https://medium.com/artificial-intelligence-in-geoscience/automatically-recognize-crops-from-landsat-by-u-net-keras-and-tensorflow-7c5f4f666231 (accessed on 26 January 2020).
5. Sun, Z.; Di, L.; Fang, H. Using long short-term memory recurrent neural network in land cover classification on Landsat and Cropland data layer time series. *Int. J. Remote Sens.* **2018**, *40*, 593–614. [CrossRef]
6. Yaseen, Z.M.; El-Shafie, A.; Jaafar, O.; Afan, H.A.; Sayl, K.N. Artificial intelligence based models for stream-flow forecasting: 2000–2015. *J. Hydrol.* **2015**, *530*, 829–844. [CrossRef]

7. Reichstein, M.; Camps-Valls, G.; Stevens, B.; Jung, M.; Denzler, J.; Carvalhais, N. Deep learning and process understanding for data-driven Earth system science. *Nature* **2019**, *566*, 195. [CrossRef] [PubMed]
8. Ghorbanzadeh, O.; Blaschke, T.; Gholamnia, K.; Meena, S.R.; Tiede, D.; Aryal, J. Evaluation of different machine learning methods and deep-learning convolutional neural networks for landslide detection. *Remote Sens.* **2019**, *11*, 196. [CrossRef]
9. Heermann, P.D.; Khazenie, N. Classification of multispectral remote sensing data using a back-propagation neural network. *Geosci. Remote Sens.* **1992**, *30*, 81–88. [CrossRef]
10. Britt, A. Kohonen neural networks and language. *Brain Lang.* **1999**, *70*, 86–94. [CrossRef]
11. Pao, Y. *Adaptive Pattern Recognition and Neural Networks*; Addison-Wesley: Boston, MA, USA, 1989.
12. Gurney, K. *An Introduction to Neural Networks*; CRC Press: Boca Raton, FL, USA, 2014.
13. Frankish, K.; Ramsey, W.M. *The Cambridge Handbook of Artificial Intelligence*; Cambridge University Press: Cambridge, UK, 2014.
14. Graves, A.; Mohamed, A.-R.; Hinton, G. Speech recognition with deep recurrent neural networks. In Proceedings of the 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Vancouver, BC, Canada, 26–31 May 2013.
15. Sallab, A.E.; Abdou, M.; Perot, E.; Yogamani, S. Deep reinforcement learning framework for autonomous driving. *Electron. Imaging* **2017**, *2017*, 70–76. [CrossRef]
16. Kök, İ.; Şimşek, M.U.; Özdemir, S. A deep learning model for air quality prediction in smart cities. In Proceedings of the 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 11–14 December 2017.
17. Cook, D.J. How smart is your home? *Science* **2012**, *335*, 1579–1581. [CrossRef]
18. Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep learning for computer vision: A brief review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef] [PubMed]
19. Najafabadi, M.M.; Villanustre, F.; Khoshgoftaar, T.M.; Seliya, N.; Wald, R.; Muharemagic, E. Deep learning applications and challenges in big data analytics. *J. Big Data* **2015**, *2*, 1. [CrossRef]
20. Dilek, S.; Çakır, H.; Aydın, M. Applications of artificial intelligence techniques to combating cyber crimes: A review. *arXiv* **2015**, arXiv:1502.03552. [CrossRef]
21. Tsipis, K. 3Q: Machine Learning and Climate Modeling. Available online: http://news.mit.edu/2019/mit-3q-paul-o-gorman-machine-learning-for-climate-modeling-0213 (accessed on 7 June 2019).
22. Sattar, A.M.; Ertuğrul, Ö.F.; Gharabaghi, B.; McBean, E.A.; Cao, J. Extreme learning machine model for water network management. *Neural Comput. Appl.* **2019**, *31*, 157–169. [CrossRef]
23. Bergen, K.J.; Johnson, P.A.; Maarten, V.; Beroza, G.C. Machine learning for data-driven discovery in solid Earth geoscience. *Science* **2019**, *363*, eaau0323. [CrossRef]
24. Watson, G.L.; Telesca, D.; Reid, C.E.; Pfister, G.G.; Jerrett, M. Machine learning models accurately predict ozone exposure during wildfire events. *Environ. Pollut.* **2019**, *254*, 112792. [CrossRef]
25. Sayad, Y.O.; Mousannif, H.; Al Moatassime, H. Predictive modeling of wildfires: A new dataset and machine learning approach. *Fire Saf. J.* **2019**, *104*, 130–146. [CrossRef]
26. Spina, R. Big Data and Artificial Intelligence Analytics in Geosciences: Promises and Potential. *GSA Today* **2019**, *29*, 42–43. [CrossRef]
27. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010.
28. LeCun, Y. LeNet-5, Convolutional Neural Networks. Available online: http://yann.lecun.com/exdb/lenet (accessed on 21 February 2020).
29. Li, X.; Peng, L.; Hu, Y.; Shao, J.; Chi, T. Deep learning architecture for air quality predictions. *Environ. Sci. Pollut. Res.* **2016**, *23*, 22408–22417. [CrossRef]
30. Rasp, S.; Pritchard, M.S.; Gentine, P. Deep learning to represent subgrid processes in climate models. *Proc. Natl. Acad. Sci. USA* **2018**, *115*, 9684–9689. [CrossRef] [PubMed]
31. Sun, Z.; Di, L.; Huang, H.; Wu, X.; Tong, D.Q.; Zhang, C.; Virgei, C.; Fang, H.; Yu, E.; Tan, X. CyberConnector: A service-oriented system for automatically tailoring multisource Earth observation data to feed Earth science models. *Earth Sci. Inform.* **2017**, *11*, 1–17. [CrossRef]
32. Sun, Z.; Di, L.; Cash, B.; Gaigalas, J. Advanced cyberinfrastructure for intercomparison and validation of climate models. *Environ. Model. Softw.* **2019**, *123*, 104559. [CrossRef]

33. Sun, Z.; Di, L. CyberConnector COVALI: Enabling inter-comparison and validation of Earth science models. In Proceedings of the AGU Fall Meeting Abstracts, San Francisco, CA, USA, 31 July 2019.
34. O'Leary, D.E. Artificial intelligence and big data. *IEEE Intell. Syst.* **2013**, *28*, 96–99. [CrossRef]
35. Lee, J.; Kao, H.-A.; Yang, S. Service innovation and smart analytics for industry 4.0 and big data environment. *Procedia Cirp* **2014**, *16*, 3–8. [CrossRef]
36. Wikipedia. Big Data. Available online: http://en.wikipedia.org/wiki/Big_data (accessed on 21 September 2014).
37. Manyika, J. Big Data: The Next Frontier for Innovation, Competition, and Productivity. Available online: http://www.mckinsey.com/Insights/MGI/Research/Technology_and_Innovation/Big_data_The_next_frontier_for_innovation (accessed on 26 January 2020).
38. Weiss, R.; Zgorski, L.-J. Obama administration unveils "big data" initiative: Announces $200 million in new R&D investments. *Off. Sci. Technol. Policy Exec. Off. Pres.* **2012**.
39. Yue, P.; Jiang, L. BigGIS: How big data can shape next-generation GIS. In Proceedings of the 2014 the Third International conference on Agro-Geoinformatics, Beijing, China, 11–14 August 2014; pp. 1–6.
40. Borthakur, D. The hadoop distributed file system: Architecture and design. *Hadoop Proj. Website* **2007**, *11*, 21.
41. Zaharia, M.; Xin, R.S.; Wendell, P.; Das, T.; Armbrust, M.; Dave, A.; Meng, X.; Rosen, J.; Venkataraman, S.; Franklin, M.J. Apache spark: A unified engine for big data processing. *Commun. ACM* **2016**, *59*, 56–65. [CrossRef]
42. George, L. *HBase: The Definitive Guide: Random Access to Your Planet-Size Data*; O'Reilly Media, Inc.: Sevastopol, CA, USA, 2011.
43. Thusoo, A.; Sarma, J.S.; Jain, N.; Shao, Z.; Chakka, P.; Anthony, S.; Liu, H.; Wyckoff, P.; Murthy, R. Hive: A warehousing solution over a map-reduce framework. *Proc. VLDB Endow.* **2009**, *2*, 1626–1629. [CrossRef]
44. Chodorow, K. *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*; O'Reilly Media, Inc.: Sevastopol, CA, USA, 2013.
45. Gorelick, N.; Hancher, M.; Dixon, M.; Ilyushchenko, S.; Thau, D.; Moore, R. Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sens. Environ.* **2017**, *202*, 18–27. [CrossRef]
46. Varia, J.; Mathew, S. Overview of Amazon web Services. Available online: http://cabibbo.dia.uniroma3.it/asw-2014-2015/altrui/AWS_Overview.pdf (accessed on 26 January 2020).
47. Kluyver, T.; Ragan-Kelley, B.; Pérez, F.; Granger, B.E.; Bussonnier, M.; Frederic, J.; Kelley, K.; Hamrick, J.B.; Grout, J.; Corlay, S. Jupyter Notebooks-a publishing format for reproducible computational workflows. In Proceedings of the 20th International Conference on Electronic Publishing, Göttingen, Germany, June 2016; pp. 87–90.
48. Hashem, I.A.T.; Yaqoob, I.; Anuar, N.B.; Mokhtar, S.; Gani, A.; Khan, S.U. The rise of "big data" on cloud computing: Review and open research issues. *Inf. Syst.* **2015**, *47*, 98–115. [CrossRef]
49. Ranjan, R. Streaming big data processing in datacenter clouds. *IEEE Cloud Comput.* **2014**, *1*, 78–83. [CrossRef]
50. Ma, Y.; Wu, H.; Wang, L.; Huang, B.; Ranjan, R.; Zomaya, A.; Jie, W. Remote sensing big data computing: Challenges and opportunities. *Future Gener. Comput. Syst.* **2015**, *51*, 47–60. [CrossRef]
51. Rathore, M.M.U.; Paul, A.; Ahmad, A.; Chen, B.W.; Huang, B.; Ji, W. Real-Time Big Data Analytical Architecture for Remote Sensing Application. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2015**, *8*, 4610–4621. [CrossRef]
52. Sun, Z.; Yue, P.; Lu, X.; Zhai, X.; Hu, L. A Task Ontology Driven Approach for Live Geoprocessing in a Service Oriented Environment. *Trans. GIS* **2012**, *16*, 867–884. [CrossRef]
53. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
54. Schneider, M. Uncertainty management for spatial datain databases: Fuzzy spatial data types. In Proceedings of the International Symposium on Spatial Databases, Hong Kong, China, 26–28 August 2013.
55. Camponovo, M.E.; Freundschuh, S.M. Assessing uncertainty in VGI for emergency response. *Cartogr. Geogr. Inf. Sci.* **2014**, *41*, 440–455. [CrossRef]
56. Vatsavai, R.R.; Ganguly, A.; Chandola, V.; Stefanidis, A.; Klasky, S.; Shekhar, S. Spatiotemporal data mining in the era of big spatial data: Algorithms and applications. In Proceedings of the 1st ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data, Redondo Beach, CA, USA, 6 November 2012.
57. Couso, I.; Borgelt, C.; Hullermeier, E.; Kruse, R. Fuzzy sets in data analysis: From statistical foundations to machine learning. *IEEE Comput. Intell. Mag.* **2019**, *14*, 31–44. [CrossRef]

58. Sun, Z.; Di, L.; Chen, A.; Yue, P.; Gong, J. The use of geospatial workflows to support automatic detection of complex geospatial features from high resolution images. In Proceedings of the 2013 Second International Conference on Agro-Geoinformatics (Agro-Geoinformatics), Fairfax, VA, USA, 12–16 August 2013.
59. Sun, Z.; Yue, P. The use of Web 2.0 and geoprocessing services to support geoscientific workflows. In Proceedings of the 2010 18th International Conference on Geoinformatics, Beijing, China, 18–20 June 2010.
60. Sun, Z.; Yue, P.; Di, L. GeoPWTManager: A task-oriented web geoprocessing system. *Comput. Geosci.* **2012**, *47*, 34–45. [CrossRef]
61. Cohen-Boulakia, S.; Belhajjame, K.; Collin, O.; Chopard, J.; Froidevaux, C.; Gaignard, A.; Hinsen, K.; Larmande, P.; Le Bras, Y.; Lemoine, F. Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities. *Future Gener. Comput. Syst.* **2017**, *75*, 284–298. [CrossRef]
62. Taylor, I.; Deelman, E.; Gannon, D. *Workflows for e-Science: Scientific Workflows for Grids*; Springer: Berlin/Heidelberg, Germany, 2006.
63. Allen, D.W. *Getting to Know ArcGIS ModelBuilder*; Esri Press: Redlands, CA, USA, 2011.
64. Heloisa Martins, S.; Tseng, M.M. Workflow technology-based monitoring and control for business process and project management. *Int. J. Proj. Manag.* **1996**, *14*, 373–378. [CrossRef]
65. Yue, P.; Gong, J.Y.; Di, L.P. Automatic Transformation from Semantic Description to Syntactic Specification for Geo-Processing Service Chains. In Proceedings of the Web and Wireless Geographical Information Systems, Naples, Italy, 12–13 April 2012.
66. Sun, Z.; Di, L.; Gaigalas, J. SUIS: Simplify the use of geospatial web services in environmental modelling. *Environ. Model. Softw.* **2019**, *119*, 228–241. [CrossRef]
67. Juric, M.B.; Krizevnik, M. *WS-BPEL 2.0 for SOA Composite Applications with Oracle SOA Suite 11g*; Packt Publishing Ltd.: Birmingham, UK, 2010.
68. Raschka, S. *Python Machine Learning*; Packt Publishing Ltd.: Birmingham, UK, 2015.
69. Sun, Z.; Peng, C.; Deng, M.; Chen, A.; Yue, P.; Fang, H.; Di, L. Automation of Customized and Near-Real-Time Vegetation Condition Index Generation Through Cyberinfrastructure-Based Geoprocessing Workflows. *Sel. Top. Appl. Earth Obs. Remote Sens. IEEE J.* **2014**, *7*, 4512–4522. [CrossRef]
70. WfMC, W.P.D.I.X. *Process Definition Language (XPDL), WfMC Standards*; WFMC: Washington, DC, USA, 2001.
71. Moreau, L.; Missier, P.; Belhajjame, K.; B'Far, R.; Cheney, J.; Coppens, S.; Cresswell, S.; Gil, Y.; Groth, P.; Klyne, G. Prov-dm: The prov data model. *Retrieved July* **2013**, *30*, W3C.
72. Goble, C.A.; Bhagat, J.; Aleksejevs, S.; Cruickshank, D.; Michaelides, D.; Newman, D.; Borkum, M.; Bechhofer, S.; Roos, M.; Li, P. myExperiment: A repository and social network for the sharing of bioinformatics workflows. *Nucleic Acids Res.* **2010**, *38*, W677–W682. [CrossRef] [PubMed]
73. Goecks, J.; Nekrutenko, A.; Taylor, J. Galaxy: A comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences. *Genome Biol.* **2010**, *11*, R86. [CrossRef] [PubMed]
74. Ludäscher, B.; Altintas, I.; Berkley, C.; Higgins, D.; Jaeger, E.; Jones, M.; Lee, E.A.; Tao, J.; Zhao, Y. Scientific workflow management and the Kepler system. *Concurr. Comput. Pract. Exp.* **2006**, *18*, 1039–1065. [CrossRef]
75. de Carvalho Silva, J.; de Oliveira Dantas, A.B.; de Carvalho Junior, F.H. A Scientific Workflow Management System for orchestration of parallel components in a cloud of large-scale parallel processing services. *Sci. Comput. Program.* **2019**, *173*, 95–127. [CrossRef]
76. ACM, A. Artifact Review and Badging. Available online: https://www.acm.org/publications/policies/artifact-review-badging (accessed on 19 February 2020).
77. Moreau, L. *The Foundations for Provenance on the Web*; Now Publishers: Hanover, MA, USA, 2010.
78. McCaney, K. Machine Learning is Creating a Crisis in Science. Available online: https://www.governmentciomedia.com/machine-learning-creating-crisis-science (accessed on 26 January 2020).
79. National Academies of Sciences, Engineering and Medicine. *Reproducibility and Replicability in Science*; The National Academies Press: Washington, DC, USA, 2019. [CrossRef]
80. Di, L.; Yue, P.; Sun, Z. Ontology-supported complex feature discovery in a web service environment. In Proceedings of the 2012 IEEE International, Geoscience and Remote Sensing Symposium (IGARSS), Munich, Germany, 22–27 July 2012; pp. 2887–2890.
81. Miller, D.D. The medical AI insurgency: What physicians must know about data to practice with intelligent machines. *NPJ Digit. Med.* **2019**, *2*, 1–5. [CrossRef]

82. Tullis, J.A.; Cothren, J.D.; Lanter, D.P.; Shi, X.; Limp, W.F.; Linck, R.F.; Young, S.G.; Alsumaiti, T. Geoprocessing, Workflows, and Provenance. In *Remotely Sensed Data Characterization, Classification, and Accuracies, Remote Sens. Handbook*; Thenkabail, P., Ed.; CRC Press: Boca Raton, FL, USA, 2015; pp. 401–421.
83. Tullis, J.A.; Corcoran, K.; Ham, R.; Kar, B.; Williamson, M. Multiuser Concepts and Workflow Replicability in sUAS Applications. In *Applications of Small Unmanned Aircraft Systems*; Sharma, J.B., Ed.; CRC Press: Boca Raton, FL, USA, 2019.
84. Yue, P.; Sun, Z.; Gong, J.; Di, L.; Lu, X. A provenance framework for Web geoprocessing workflows. In Proceedings of the 2011 IEEE International Geoscience and Remote Sensing Symposium (IGARSS11), Vancouver, BC, Canada, 24–29 July 2011; pp. 3811–3814.
85. Goodchild, M.; Fotheringham, S.; Li, W.; Kedron, P. Replicability and Reproducibility in Geospatial Research: A SPARC Workshop. Available online: https://sgsup.asu.edu/sparc/RRWorkshop (accessed on 26 January 2020).
86. Naseri, M.; Ludwig, S.A. Evaluating workflow trust using hidden markov modeling and provenance data. In *Data Provenance and Data Management in eScience*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 35–58.
87. Roemerman, S. Four Reasons Data Provenance is Vital for Analytics and AI. Available online: https://www.forbes.com/sites/forbestechcouncil/2019/05/22/four-reasons-data-provenance-is-vital-for-analytics-and-ai/ (accessed on 23 December 2019).
88. Sun, Z.; Di, L.; Tong, D.; Burgess, A.B. Advanced Geospatial Cyberinfrastructure for Deep Learning Posters. In Proceedings of the AGU Fall Meeting, San Francisco, CA, USA, 9–13 December 2019.
89. Caíno-Lores, S.; Lapin, A.; Carretero, J.; Kropf, P. Applying big data paradigms to a large scale scientific workflow: Lessons learned and future directions. *Future Gener. Comput. Syst.* **2018**. [CrossRef]
90. Oliver, H.J.; Shin, M.; Sanders, O. Cylc: A Workflow Engine for Cycling Systems. *J. Open Source Softw.* **2018**, *3*, 737. [CrossRef]
91. Armbrust, M.; Fox, A.; Griffith, R.; Joseph, A.D.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I. A view of cloud computing. *Commun. ACM* **2010**, *53*, 50–58. [CrossRef]
92. Sun, Z.; Di, L.; Heo, G.; Zhang, C.; Fang, H.; Yue, P.; Jiang, L.; Tan, X.; Guo, L.; Lin, L. GeoFairy: Towards a one-stop and location based Service for Geospatial Information Retrieval. *Comput. Environ. Urban Syst.* **2017**, *62*, 156–167. [CrossRef]
93. Tan, X.; Di, L.; Deng, M.; Huang, F.; Ye, X.; Sha, Z.; Sun, Z.; Gong, W.; Shao, Y.; Huang, C. Agent-as-a-service-based geospatial service aggregation in the cloud: A case study of flood response. *Environ. Model. Softw.* **2016**, *84*, 210–225. [CrossRef]
94. Bhardwaj, S.; Jain, L.; Jain, S. Cloud computing: A study of infrastructure as a service (IAAS). *Int. J. Eng. Inf. Technol.* **2010**, *2*, 60–63.
95. Reed, D.A.; Dongarra, J. Exascale computing and big data. *Commun. ACM* **2015**, *58*, 56–68. [CrossRef]
96. Contributors, W. Big Data. Available online: https://en.wikipedia.org/w/index.php?title=Big_data&oldid=925811014 (accessed on 14 November 2019).
97. Arendt, A.A.; Hamman, J.; Rocklin, M.; Tan, A.; Fatland, D.R.; Joughin, J.; Gutmann, E.D.; Setiawan, L.; Henderson, S.T. Pangeo: Community tools for analysis of Earth Science Data in the Cloud. In Proceedings of the AGU Fall Meeting Abstracts, San Francisco, CA, USA, 31 July 2019.
98. Ketkar, N. Introduction to pytorch. In *Deep Learning with Python*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 195–208.
99. Abadi, M.; Barham, P.; Chen, J.; Chen, Z.; Davis, A.; Dean, J.; Devin, M.; Ghemawat, S.; Irving, G.; Isard, M. TensorFlow: A System for Large-Scale Machine Learning. In Proceedings of the OSDI, Savannah, GA, USA, 2–4 November 2016; pp. 265–283.
100. Sun, Z.; Di, L. Geoweaver: A Web-Based Prototype System for Managing Compound Geospatial Workflows of Large-Scale Distributed Deep Networks. **2019**. [CrossRef]
101. Sun, Z.; Di, L.; Fang, H.; Burgess, A.B.; Singh, N. Deep Learning Cyberinfrastructure for Crop Semantic Segmentation. In Proceedings of the AGU Fall Meetin, San Francisco, CA, USA, 31 July 2019.
102. Sun, Z.; Fang, H.; Di, L.; Yue, P.; Tan, X.; Bai, Y. Developing a web-based system for supervised classification of remote sensing images. *GeoInformatica* **2016**, *20*, 629–649. [CrossRef]
103. Sun, Z.; Fang, H.; Di, L.; Yue, P. Realizing parameterless automatic classification of remote sensing imagery using ontology engineering and cyberinfrastructure techniques. *Comput. Geosci.* **2016**, *94*, 56–67. [CrossRef]

104. Sun, Z.; Fang, H.; Deng, M.; Chen, A.; Yue, P.; Di, L. Regular Shape Similarity Index: A Novel Index for Accurate Extraction of Regular Objects from Remote Sensing Images. *Geosci. Remote Sens.* **2015**, *53*, 3737–3748. [CrossRef]
105. You, M.C.; Sun, Z.; Di, L.; Guo, Z. A web-based semi-automated method for semantic annotation of high schools in remote sensing images. In Proceedings of the Third International Conference on Agro-geoinformatics (Agro-geoinformatics 2014), Beijing, China, 11–14 August 2014; pp. 1–4.
106. Sun, J.; Di, L.; Sun, Z.; Shen, Y.; Lai, Z. County-Level Soybean Yield Prediction Using Deep CNN-LSTM Model. *Sensors* **2019**, *19*, 4363. [CrossRef] [PubMed]
107. Sun, Z.; Di, L.; Fang, H.; Guo, L.; Yu, E.; Tang, J.; Zhao, H.; Gaigalas, J.; Zhang, C.; Lin, L. Advanced Cyberinfrastructure for Agricultural Drought Monitoring. In Proceedings of the 2019 8th International Conference on Agro-Geoinformatics (Agro-Geoinformatics), Istanbul, Turkey, 16–19 July 2019; pp. 1–5.
108. Han, W.; Yang, Z.; Di, L.; Mueller, R. CropScape: A Web service based application for exploring and disseminating US conterminous geospatial cropland data products for decision support. *Comput. Electron. Agric.* **2012**, *84*, 111–123. [CrossRef]
109. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef]
110. Zhang, C.; Di, L.; Sun, Z.; Lin, L.; Eugene, G.Y.; Gaigalas, J. Exploring cloud-based Web Processing Service: A case study on the implementation of CMAQ as a service. *Environ. Model. Softw.* **2019**, *113*, 29–41. [CrossRef]
111. Gaigalas, J.; Di, L.; Sun, Z. Advanced Cyberinfrastructure to Enable Search of Big Climate Datasets in THREDDS. *ISPRS Int. J. Geo Inf.* **2019**, *8*, 494. [CrossRef]
112. Zhang, C.; Di, L.; Sun, Z.; Eugene, G.Y.; Hu, L.; Lin, L.; Tang, J.; Rahman, M.S. Integrating OGC Web Processing Service with cloud computing environment for Earth Observation data. In Proceedings of the 2017 6th International Conference on Agro-Geoinformatics, Fairfax, VA, USA, 7–10 August 2017; pp. 1–4.
113. Tan, X.; Guo, S.; Di, L.; Deng, M.; Huang, F.; Ye, X.; Sun, Z.; Gong, W.; Sha, Z.; Pan, S. Parallel Agent-as-a-Service (P-AaaS) Based Geospatial Service in the Cloud. *Remote Sens.* **2017**, *9*, 382. [CrossRef]
114. Roy, D.P.; Wulder, M.; Loveland, T.; Woodcock, C.; Allen, R.; Anderson, M.; Helder, D.; Irons, J.; Johnson, D.; Kennedy, R. Landsat-8: Science and product vision for terrestrial global change research. *Remote Sens. Environ.* **2014**, *145*, 154–172. [CrossRef]
115. Sun, Z.; Di, L.; Zhang, C.; Fang, H.; Yu, E.; Lin, L.; Tan, X.; Guo, L.; Chen, Z.; Yue, P. Establish cyberinfrastructure to facilitate agricultural drought monitoring. In Proceedings of the 2017 6th International Conference on Agro-Geoinformatics, Fairfax, VA, USA, 7–10 August 2017; pp. 1–4.
116. Sun, Z.; Yue, P.; Hu, L.; Gong, J.; Zhang, L.; Lu, X. GeoPWProv: Interleaving Map and Faceted Metadata for Provenance Visualization and Navigation. *Geosci. Remote Sens.* **2013**, *51*, 5131–5136.