GeoEngine: A Platform for Production-Ready Geospatial Research

Sagar Verma[1,2], Siddharth Gupta[2], Hal Shin[2], Akash Panigrahi[2], Shubham Goswami[2], Shweta Pardeshi[2], Natanael Exe[2], Ujwal Dutta[2], Tanka Raj Joshi[2], Nitin Bhojwani[2]

[1] Université Paris-Saclay, CentraleSupélec, Inria, Centre de Vision Numérique

[2] Granular AI

{sagar,akash,sid}@granular.ai

Abstract

Geospatial machine learning has seen tremendous academic advancement, but its practical application has been constrained by difficulties with operationalizing performant and reliable solutions. Sourcing satellite imagery in real-world settings, handling terabytes of training data, and managing machine learning artifacts are a few of the challenges that have severely limited downstream innovation. In this paper we introduce the GeoEngine[1] platform for reproducible and production-ready geospatial machine learning research. GeoEngine removes key technical hurdles to adopting computer vision and deep learning-based geospatial solutions at scale. It is the first end-to-end geospatial machine learning platform, simplifying access to insights locked behind petabytes of imagery. Backed by a rigorous research methodology, this geospatial framework empowers researchers with powerful abstractions for image sourcing, dataset development, model development, large scale training, and model deployment. In this paper we provide the GeoEngine architecture explaining our design rationale in detail. We provide several real-world use cases of image sourcing, dataset development, and model building that have helped different organisations build and deploy geospatial solutions.

1. Introduction

In order to address many of humanity's biggest challenges it is imperative that we develop a thorough understanding of our planet and how it is evolving. Satellite and aerial imagery combined with geospatial machine learning offer an unparalleled source of objective global-scale data. The past decade has seen a near eight-fold increase in the number of earth observation satellites deployed to orbit, with similar growth in the availability of commercial aerial images obtained via fixed wing planes and balloons. Moreover, small unmanned aerial vehicles (UAVs) in both the retail and commercial markets are generating a plethora of geospatial data. This recent increase in aerial data (e.g. multispectral, SAR, LIDAR) as well as non-image geospatial data (e.g. GPS) has radically changed the potential for downstream applications. As such data becomes cheaper and more readily available, corporations, governments and

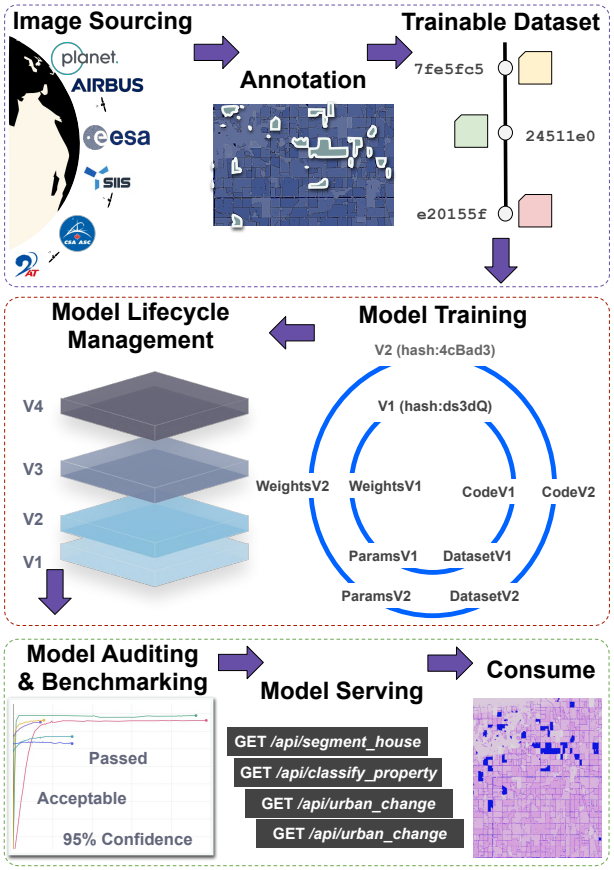

Figure 1. MLOps pipeline for Geospatial ML.

[1] https://apps.granular.ai/apps

citizens alike seek to leverage it to better understand our planet and its inhabitants.

Along with this increase in raw image availability, methods in computer vision have seen remarkable development driven in large part by the advancement of machine learning. Yet, despite this progress, tooling to support earth observation research is lacking. Researchers and developers working on applying geospatial machine learning methods on real world solutions are actively looking for an end-to-end platform that can manage every aspect of geospatial research. Much of the focus in geospatial research has been solving problems and building novel solutions, while less attention has been given to the underlying supporting infrastructure and protocols. While it is possible to achieve decent performance and accuracy in research environments with an ad-hoc methodology and environment, the challenge is building an integrated geospatial machine learning solution that can support production-scale data throughput.

The machine learning development lifecycle mirrors that of software development, with four key stages: planning, development, testing and deployment. Similarly, maintaining ML systems requires similar ongoing care to that of conventional software systems, from identifying faulty logic and applying appropriate fixes to improving features and performance. To carry out these processes, software development relies on DevOps to streamline development while continuously delivering new releases and maintaining quality. The workflow for machine vision models follows a similar pattern. Where the two practices differ is the from traditional software development is in the environment where it operates. These two development practices diverge when it comes to the deterministic nature of software development juxtaposed to the inherently probabilistic nature of computer vision development. Further, as recent global events have demonstrated, this planet is constantly changing, so CV practitioners must expect that the real-world data that powers their models will inevitably change as well.

The apparent similarities of these two development practices has led to an increasing drive to build development operations (DevOps) systems for machine learning. Machine learning operations (MLOps) platforms systematize the process of building and training experimental machine learning models and translating them into production. MLOps platforms for computer vision can serve multiple purposes, with some common functions including: data sourcing and broadcasting, image data management, dataset development, model metadata and artifact management, model validation and operationalization. Unfortunately, of the platforms that support machine vision problems today, none fully support geospatial research. Along with complexities specific to geospatial data, such as sensor and image qualities and map projection, problems around data representation, data access limitations, and performance issues with working with large geospatial images make these platforms ill-suited for geospatial research.

In this paper we introduce a new MLOps tool focused solely on geospatial machine learning. GeoEngine creates and manages large annotated datasets, converts them into analysis-ready datasets, launches efficient training pipelines, stores and versions training artifacts, and deploys geospatial model APIs for real-world consumption. There are additional features that are oriented towards geospatial and remote sensing research problems as well as in-production solutions, such as: live sourcing satellite imagery, rendering analytical data to maps and managing vector data for time-series datasets.

2. Existing MLOps Tools

There are fewer computer vision machine learning platforms that support end-to-end machine learning research from dataset sourcing to end products. Researchers and academics widely use different tools for different tasks in a machine learning pipeline. In case of geospatial machine learning this is widely the case due to lack of platforms that support tighter integration with geospatial data and downstream GIS and analysis tools. In this section we discuss the different tools used to create and manage datasets, train models, manage artifacts and life-cycles, and deploy models. We also discuss some of the tools that do in fact support end-to-end automation of machine learning research.

2.1. Dataset Creation and Management

While there exist several tools designed for image annotation, few fully support geospatial data. Tools like LabelMe [33], VIA [8], VoTT [24], and CVAT [34] allow annotation of non-georegistered images. These tools have empowered computer vision community with large datasets that have helped accelerate computer vision research. LabelBox [35] and Scale AI [18] are two widely used commercial tool that have limited geospatial support, with basic support for web map service (WMS) images. While this presents a challenge when labelling multi-temporal data, these tools do have a fantastic set of active learning features for AI-assisted data annotation.

QGIS [29] is a widely used open source app that is mostly used to analyze geospatial data. The tooling does support annotating shapes but is very limited in managing complex annotation workflows. ArcGIS [31] is a widely used commercial tool similar to QGIS in terms of features and limitations. PulseSatellite [23] is a web based collaborative tool where analysts can annotate few images and then use Mask R-CNN [13] to automate annotation of large areas. This tool is very limited in terms of importing images and types of problems supported. V-RSIS [14] is collaborative tool which allows users to annotate WMS images from Google Maps.

Tool	Goespatial Data	Model Training	Model Lifecycle	Auditing & Benchmarking	Serving
QGIS [29]	Yes	Limited	No	No	Limited
ArcGIS [31]	Yes	Yes	No	No	Limited
Descartes Labs [17]	Yes	Yes	No	No	No
Orbital Insight [19]	Yes	No	No	No	No
UP42 [11]	Yes	Yes	No	No	Yes
GeoEngine	Yes	Yes	Yes	Yes	Yes
CVS [3]	No	Yes	No	No	Yes
Aoto ML Vision [5]	No	Yes	Yes	Limited	Yes
TPOT [26]	No	Yes	No	Limited	No
Auto-Sklearn [9, 10]	No	Limited	No	Limited	No
MLBox [32]	No	Yes	No	Limited	No
Algorithmia [16]	No	Yes	Yes	Yes	Yes
Kubeflow [12]	Limited	Yes	Yes	Yes	Yes
Azure ML [2]	No	Yes	Yes	Yes	Yes
Gradient [28]	No	Yes	Yes	Yes	Yes

Table 1. Comparison of GeoEngine with other geospatial, Auto ML, and MLOps platforms.

2.2. Model Training and Management

Most MLOps platforms principally operate downstream of dataset creation tools. With such platforms, one can begin to achieve continuous training. Paradigmatically similar to continuous integration, continuous training enables users to systematically iterate and improve on models without impacting system stability. The model registry helps users achieve this task by maintaining model images, source code versioning, and model artifacts.

Polyaxon [25] addresses many challenges related to ML training, from data versioning to efficient resource provisioning and experiment management. It is a kubernetes-based platform that ships with a user interface for accessing experimentation logs and metrics. It supports team-level collaboration and integrates with many tools, from data management to notifications. MLFlow [22] offers similar functionality, with a focus on creating reproducible "Projects" that can be migrated from development to production environments. It also creates abstractions to support native model deployment for most popular frameworks.

While Weights and Biases (WandB) [38] is designed with a strong developer-first focus, it is strongest in an academic context. WandB focuses on ease of integration and setup, and its strong experiment observability features make it optimal for training. Similar to Weights and Biases, NeptuneAI [20] provides experiment tracking functionality, but stops short of data storage. It does offer a comprehensive set of charting features along with support for major charting libraries, enabling users to integrate it into existing workflows, rather than serving as a replacement. It also supports experiment optimization such as hyper-parameter tuning.

2.3. Model Deployment and Life Cycle Management

Once a vision model has been created, the final steps in operationalizing that model are to test and deploy it to a dedicated serving environment. Certain serving platforms also attempt to support model development and training, while others focus exclusively on model hosting. While some systems can be hosted on-premise, this can be costly and difficult to manage at scale. As such, most serving platforms focus on cloud-based hosting, with kubernetes-based hosting being a natural choice for its scaling properties. Tools like BentoML [4] and Seldon [15] can facilitate deployment, while other tools serve to manage the underlying requirements, from hardware provisioning to data management. BentoML provides a unified deployment framework that serves as a connector between machine learning frameworks and serving platforms. Seldon provides powerful data structures to let users package their models for cloud deployment. Seldon also ships with Kubeflow and is supported by any Kubernetes environment.

Model monitoring is not a commonly found feature in such platforms, but would be beneficial as it would let one tweak and improve a model in production continuously. In a highly developed MLOps workflow, this should be an active process. There are three aspects to monitoring:

- **Technical/system monitoring checks** if the model infrastructure is served correctly or not.
- **Model monitoring** validates the predictions' accuracy.
- **Business performance monitoring** comes down to whether the model is helping the business or not.

2.4. Automation

Tools discussed above can be used for a single component or a few components of a ML pipeline. Often it is required to automate the entire process right from collecting and managing data to deploying it and monitoring it in production. Such tools can be categories into AutoML and MLOps automation. Various AutoML frameworks automate various steps of the machine learning lifecycle. SageMaker [1], VisionAI [6], Azure Machine Learning [2], Kubeflow [12], Algorithmia [16], and Gradient [28] are widely used in academia and industry. It can be observed in Table 1 that tools that support geospatial data are not MLOps oriented, most of them do not even support model training. Auto ML and MLOps tools are very good for model training and serving part. All of them lack geospatial data integration with limited exception of Kubeflow.

SageMaker serves many of the most commonly used models along with data management built on Amazon cloud infrastructure. It also offers ancillary human-in-the-loop services like data labeling. It's general purpose nature and managed approach does make it challenging to employ with certain workflows, particularly when there exist complex follow-on processes and/or visualization requirements.

VisionAI is another Google hosted machine vision platform slightly more focused in scope than SageMaker, but still designed for a broad spectrum of machine vision applications. Similar to SageMaker, it can benefit from a tight integration with Google Cloud Platform's data management and computing resources. The hosted nature of VisionAI might limit its viability for sophisticated operators requiring more control.

AzureML much like SageMaker, Azure's hosted offering for ML management is a general-purpose artifact management and training platform, with similar limitations and benefits.

Kubeflow is an effort to make it simpler to connect compute-intensive operations in machine learning with cloud resources managed within Kubernetes. Kubeflow, supports ML pipelines through Argo, a Kubernetes workflow manager, and supports model deployment as packaged APIs across a broad range of native and third-party systems. Kubeflow in limited capacity can interact with Google EarthEngine to support geospatial machine learning.

3. GeoEngine Architecture

This section explains different components of GeoEngine that enables its user to create large datasets from scratch, manage model training and deploy real world solutions.

3.1. Europa: Data Annotation Tool

Europa was conceived to satisfy a more niche community of users within the annotation tooling landscape. As such, while Europa shares many features common to alternative applications, the features of Europa have primarily evolved around usages more pertinent to geospatial problems. At a high level, Europa aims at solving the following goals:

- To readily ingest image sources, public or private.
- To support a broad array of remote sensing and vision problems.
- To deliver high-quality model training data.
- To share datasets adhering to open and reproducible standards.

Of the various remote sensing and vision problems, change detection is a notably challenging requirement. It requires the ability to track changes in a given area-of-interest across multiple dates, which requires the ability to swap the image context in which the annotation is situated.

To create high quality annotations, we must ensure that the application can support several key features. Firstly, it must allow for sufficiently large datasets. Secondly, it must support a multitude of image types, different bands, and as mentioned above, multi-date image sets. Lastly, a systematic validation process is essential in both training annotators and correcting mistakes during the annotations.

In addition to the annotation process, the application should allow for easy sharing and importing of datasets. This feature would allow users to quickly begin testing hypotheses and further improve existing imported or annotated datasets.

3.2. Neso: Image Sourcing Tool

Neso is a microservice which provides capability through REST endpoints to query and download satellite and aerial images. It interacts with several geospatial image providers to automate the retrieval of geospatial images in a structured and minimal way. Neso also provides the functionality of adding private images via custom AWS and Google Storage buckets. Europa uses Neso internally to allow its users to search for images for annotations tasks. Dione and Titan use Neso to get historical and future tasked images for inference workflows and dashboard visualizations.

3.3. Atlas: Training Data Management Tool

When an annotation task is created on Europa, all the images are available in our platform in two different formats. One format is Cloud Optimized GeoTiffs (COGs) which is

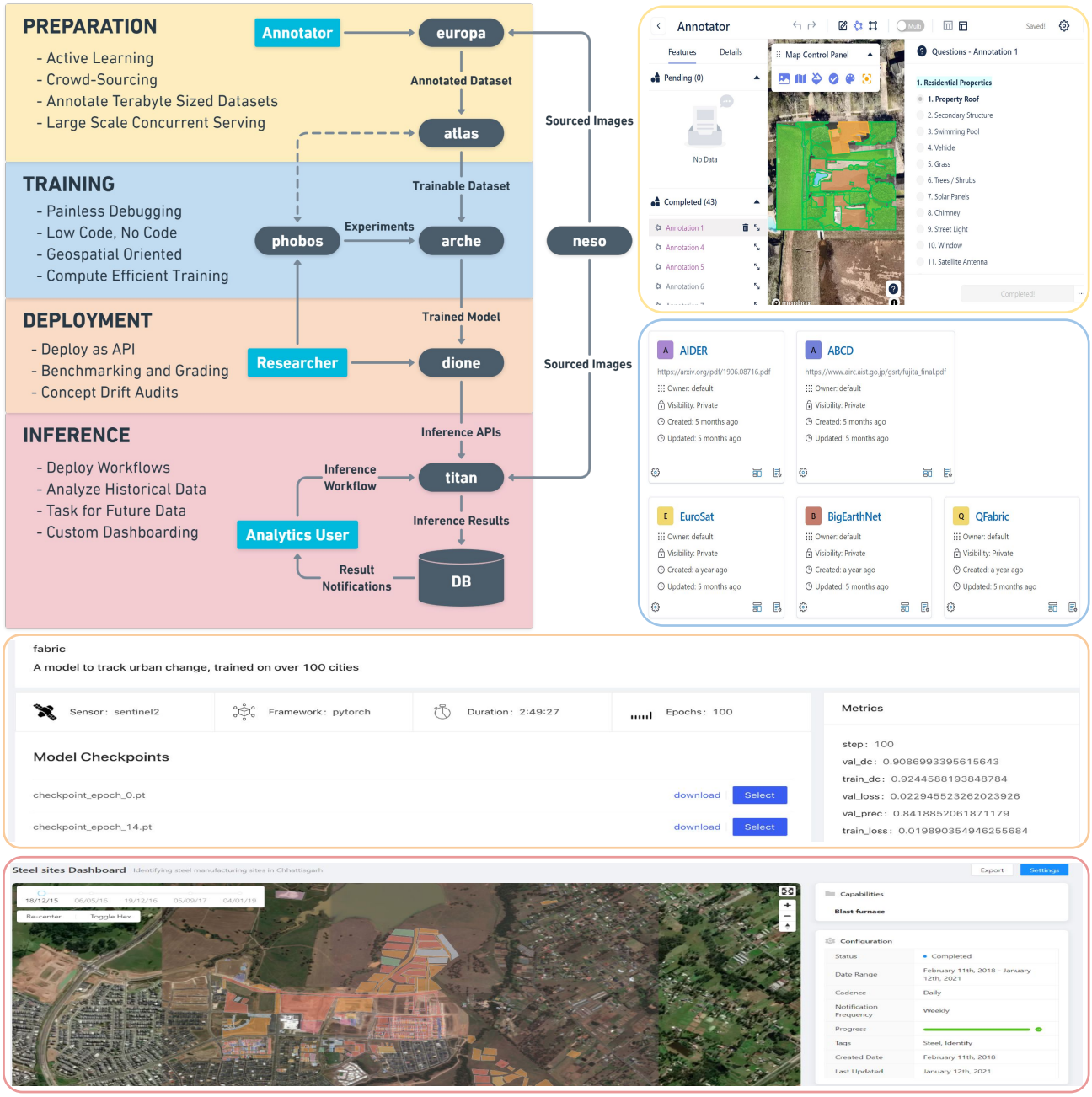

Figure 2. A typical geospatial research-to-production workflow on GeoEngine.

used by Europa to render images as maps. Other is the original raw format that Neso acquires. After annotations are complete, we need to use the annotations and raw images to make a trainable dataset. Atlas can be invoked by Phobos CLI to create trainable datasets in an asynchronous manner. For very large datasets it may take several hours depending upon how much resource is allowed to be used. Atlas creates and serves the trainable datasets in an efficient manner. Atlas has been designed to scale linearly with dataset size while using cheap compute instances. A single trainable dataset can be simultaneously used by hundreds of models under training. Atlas can serve thousands of users and petabytes of training data at any given time. Atlas enables researchers to share their datasets in trainable format openly as POSIX file URLs. This is very much required for reproducible machine learning.

3.4. Arche: Experiment Management Tool

Arche enables users to deploy geospatial machine learning development workflows to the cloud. Working in concert with Phobos, Arche packages and deploys user-defined models to the cloud for training. Further, it fetches relevant training data efficiently, piping analysis-ready image and training data to experiments without the need for manual data management.

With Arche, each deployment- and experiment-related event is logged, including run-time details, experiment hyper-parameters, and performance metrics. This enables maximal user control and observability into the model training life-cycle, which is particularly important when models have hardware requirements the necessitate training in the cloud. Arche also versions each data and machine learning asset, facilitating model reproduction in production environments and allowing for iterative model improvement. Through Arche, model training overhead and complexity is significantly reduced. By maximizing utilization and control of our owned or cloud-leased hardware, cost can further be reduced.

3.5. Phobos: Geospatial AI Python Library

Phobos is a utility library that serves multiple functions during the model discovery and development phases of research. First, it assists in exporting annotations from Europa and pre-processing raw images into trainable datasets. It also generates boilerplate code for new geospatial machine learning projects. This boilerplate code enables researchers to write minimal code and focus on actual model development. This boilerplate is highly configurable and allows training locally as well as in the cloud using multiple GPUs and multiple nodes. This means that experimentation can be prepared and debugged locally prior to complete training in the cloud. This functionality is readily accessible within python as well as from the Phobos command-line interface (CLI).

Phobos library provides the necessary training components to populate the project:

- **grain** allows users to configure their project hyperparameters and other information using a YAML file.
- **transforms** can be used to do image preprocessing tasks like augmentations and satellite image specific processing.
- **loss** module contains widely used losses. Currently Phobos supports 43 different losses for image classification, segmentation and detection tasks. It also allows users to easily use custom defined loss functions.
- **metrics** module contains widely used metrics. Currently Phobos supports 21 different metrics for image classification, segmentation and detection tasks. It also allows users to easily use custom defined metrics.
- **io** provides representation for model/network input and output structure. It supports multiple input structure for tasks like multi-sensor fusion and multiple output structure for multi-task like problems. It also supports processing of raw images and annotations into trainable datasets.
- **runner** is the core module that manages training iterations/epochs. It has all the functionalities around multiple node and multiple GPU training and metrics computation and logging of metrics into Arche.

3.6. Dione: Model Deployment Tool

In order to rely on a mission-critical model there must be a clear demonstration that the model works under real-world conditions. Dione exposes a series of tests that can be conducted to evaluate a model's performance under a broad range of conditions. Through this "test-bay" users can benchmark their model's performance *before* using it in mission-critical environments. Dione enables model auditing by providing a sandbox test-bay where models developed under lab conditions can be performance benchmarked and tested against real-world conditions. For instance, a model designed to track infrastructure development in Pakistan may not be well suited for the environment and geography of Sudan. With Dione, users see what a model is "graded" for and where there might be blind spots that the model is not equipped to address. Dione supports following

- **Benchmarking** provides a means for users to understand the context in which a model will perform well, and critically, when it will not.
- **Deployment** of trained models as production APIs so that they can be consumed by user for real time analysis on historical and future satellite image streams. Model pruning [37] and quantization are used to make inference efficient.
- **Grading** allows users to understand and generate reports on how well a model performs with different inputs (the spatial and spectral properties of the imagery),
- **Ongoing Validation** allows users to preemptively address issues of model performance degradation over time.

3.7. Titan: Analysis Dashboard

Titan attempts to take geospatial models one step further, enabling non-technical users to deploy inference workflows and investigate model outputs. By mapping geospatial

machine learning operations to business needs, Titan effectively bridges the gap between domain awareness and technical requirement. In the Titan platform, users can select and deploy model APIs based on model performance and applicability to the user's query. Once the model API is available, the user can interrogate imagery. Thanks to GeoEngine's cluster management, models are capable of autoscaling to support global-scale analytical requirements.

Titan workflows supports Region-of-Interest (RoI) subscriptions, enabling users to run models against dynamically fetched imagery for their desired location at a regular interval (daily, weekly, monthly). Imagery is fetched via Neso based on model and user requirements, obviating the need for manual data acquisition for one-off or recurring analysis. One can execute these workflows through both the Titan UI and Titan API, while associated client libraries enable the downstream consumption of the resulting output data, making it trivial to bring resultant model data into external environments for visualization, analysis and decision-making.

4. GeoEngine in Production

Metrics	Value
Satellite Sources	21
Aerial and UAV Sources	5
Annotated Datasets	16
Annotated Pixels	95 billions
Annotated Area	695 thousands
Annotated Polygons	1.1 million
Imported Open Datasets	56
Enriched Open Datasets	6
Total Pixels	2.3 trillions
Total Area	6.95 millions
Total Polygons	120 millions

Table 2. Europa Usage Statistics.

Metrics	Value
Vision Backbones	372
Loss Functions	43
Metrics	21
Problems Types	17
Total Projects	79
Total Experiments	7318
Average Experiments Per Project	43

Table 3. Phobos and Arche Statistics.

GeoEngine has been evolving steadily since its inception back in January 2021. It started with development of minimal version of all components. Then the focus shifted on maturing Europa, Phobos and Arche. In this section we discuss the current status of GeoEngine and provide some quantitative details on usage statistics of the various components. We also discuss some projects from dataset development to model training and deployment.

4.1. Current Status

Table 2 provides statistics of Europa. Through the efforts of our annotation team we have been able to build out 16 datasets by sourcing satellite and aerial imagery from 26 different sources. In the past year we have annotated 659K square-km of area resulting in 1.1 million polygons. This has given us 95 billions pixels worth of data across 16 different use cases involving segmentation, object detection, 3D registration, multi-sensor fusion, multi-task learning, change detection, and other non-conventional modeling operations.

We have performed experiments on all the datasets available in our platform. Table 3 shows feature and usage statistics of Phobos and Arche. All projects have been trained and managed on Arche and were created using Phobos library. Phobos has enabled us to rapidly iterate through catalogued models using available loss functions and metrics to train and in doing so judge thousands of experiments.

4.2. Open Source Datasets

We have imported 56 open source geospatial datasets in Europa. This gave us an opportunity to validate some of the open source datasets. We identified several mistakes in annotations in 6 imported open source datasets namely XView [21], OSCD [7], OSCD MultiDate [27], QFabric [36], and FloodNet [30]. We then fixed and enriched these datasets with more labels and better polygon boundaries. Figure 3 shows QFabric dataset imported in Europa for validation and debugging. These datasets along with other imported open source datasets are freely available on our platform. We plan to import all open source geospatial datasets and keep validating and extending them for public usage. We believe this practice will lead to higher quality densely rich datasets that will dramatically accelerate geospatial machine learning research.

4.3. Open Source Projects

We have 79 geospatial machine learning projects that area publicly available with all the experiments. All version information, license and authorship data, code, datasets, and model artifacts are available publicly. Figure 4 shows list of experiments performed on the original version of QFabric dataset. This combined with openly available datasets

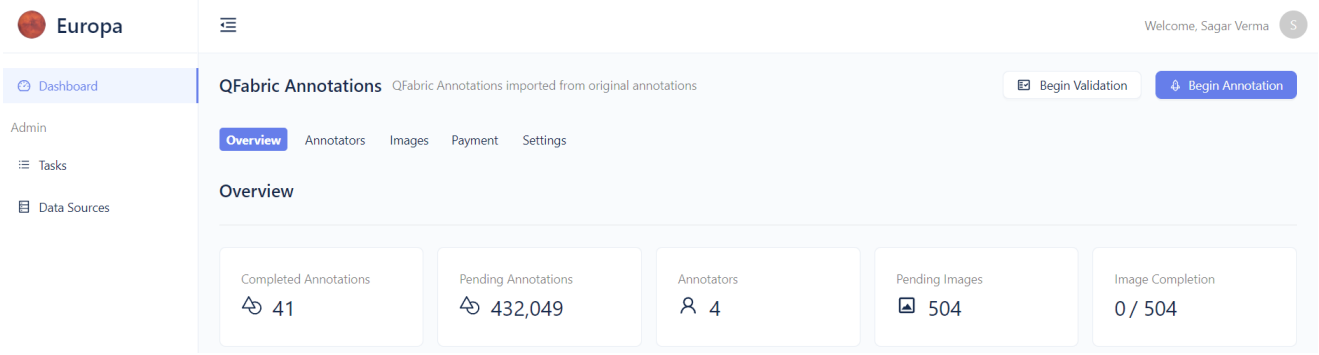

Figure 3. Fixing and enriching QFabric annotations on Europa.

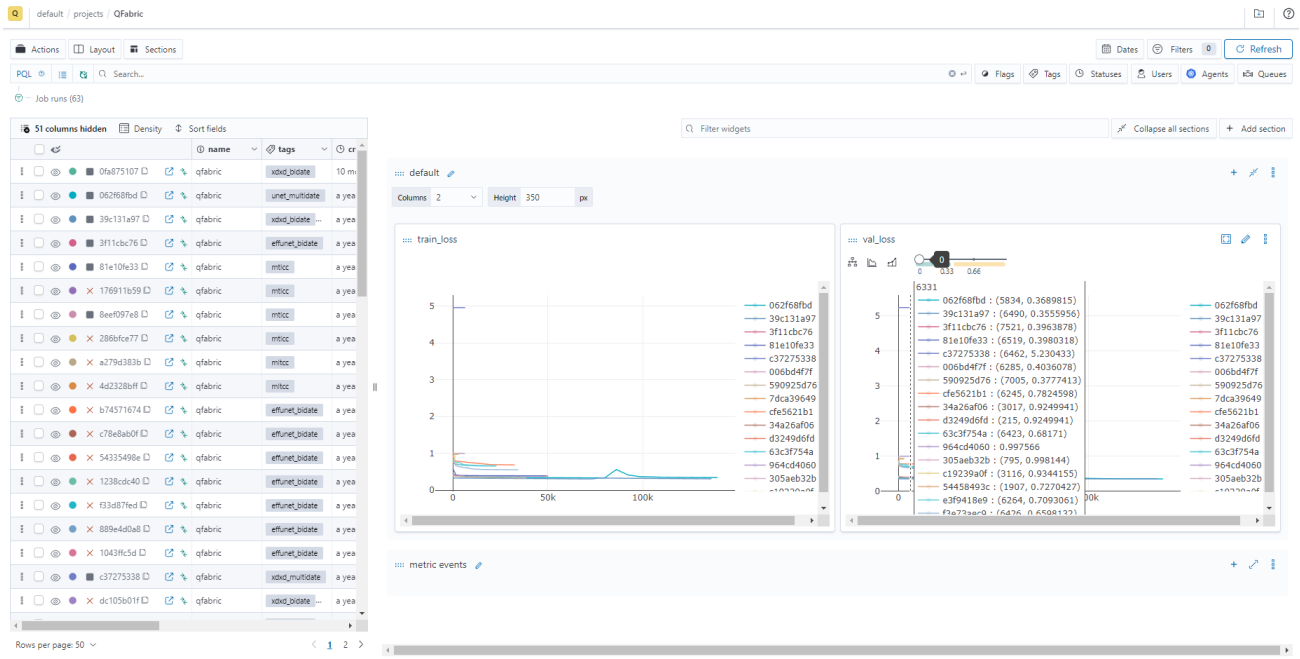

Figure 4. Experiments on QFabric dataset performed on Arche.

make GeoEngine a very powerful and beneficial platform for industry and academia alike.

5. Expectations and Future Goals

This paper presents GeoEngine, a geospatial machine learning platform that enables researchers and developers to manage large geospatial projects. GeoEngine is qualitatively compared with existing MLOps platforms. Design choices are considered based on what is lacking in existing tools and what is required by remote sensing researchers and developers. We also provide a walk-through example and the current status of different components of GeoEngine.

The goal of this paper and the associated demo is to introduce GeoEngine to geospatial researchers and developers in academia and industry. We are actively working on importing all open source datasets into GeoEngine and then validating and enriching those datsets. We are continuously building bigger datasets on varied problems and making them open source.

On the modeling front we are introducing wide array of backbones trained on bigger geospatial datasets. From the feature perspective a lot of novel research is going on active learning, concept drift, model certification, etc which might result into new exciting features. User experience research is being done to continuously improve ease of use. We hope that the geospatial and computer vision community gets to use our platform under free tier and help us evolve it into a state-of-the-art geospatial research platform by providing valuable feedback and critique.

References

[1] Amazon. Sagemaker. https://aws.amazon.com/sagemaker/. 4

[2] Microsoft Azure. Azure ml. https://azure.microsoft.com/en-in/services/machine-learning/. 3, 4

[3] Microsoft Azure. Custom vision service. https://azure.microsoft.com/en-in/services/cognitive-services/custom-vision-service/. 3

[4] BentoML. Bentoml. https://www.bentoml.ai/. 3

[5] Google Cloud. Automl vision. https://cloud.google.com/vision/automl. 3

[6] Google Cloud. Vision ai. https://cloud.google.com/vision. 4

[7] Rodrigo Caye Daudt, Bertrand Le Saux, Alexandre Boulch, and Yann Gousseau. Urban change detection for multispectral earth observation using convolutional neural networks. In *IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018. 7

[8] A. Dutta, A. Gupta, and A. Zissermann. VGG image annotator (VIA). http://www.robots.ox.ac.uk/ vgg/software/via/, 2016. 2

[9] Matthias Feurer, Katharina Eggensperger, Stefan Falkner, Marius Lindauer, and Frank Hutter. Auto-sklearn 2.0: Hands-free automl via meta-learning. 2020. 3

[10] Matthias Feurer, Aaron Klein, Jost Eggensperger, Katharina Springenberg, Manuel Blum, and Frank Hutter. Efficient and robust automated machine learning. In *Advances in Neural Information Processing Systems 28 (2015)*, pages 2962–2970, 2015. 3

[11] UP42 GmbH. Up42. https://up42.com/. 3

[12] Google and Community. Kubeflow. https://www.kubeflow.org/. 3, 4

[13] Kaiming He et al. Mask r-cnn. *ICCV*, pages 2980–2988, 2017. 2

[14] Dongyang Hou et al. V-rsir: An open access web-based image annotation tool for remote sensing image retrieval. *IEEE Access*, 7:83852–83862, 2019. 2

[15] Alex Housley et al. Seldon. https://www.seldon.io/. 3

[16] Algorithmia Inc. Algorithmia. https://algorithmia.com/. 3, 4

[17] Descartes Labs Inc. Descartes labs. https://descarteslabs.com/. 3

[18] Scale AI Inc. Scale ai. https://scale.com/. 2

[19] Orbital Insight. Orbital insight. https://orbitalinsight.com/. 3

[20] Neptune Labs. Neptune ai. https://neptune.ai/. 3

[21] Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xview: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018. 7

[22] LLC LF Projects. Mlflow. https://www.mlflow.org/. 3

[23] Tomaz Logar et al. Pulsesatellite: A tool using human-ai feedback loops for satellite image analysis in humanitarian contexts. In *AAAI*, pages 13628–13629, 2020. 2

[24] Microsoft and community. Visual object tagging tool: An electron app for building end to end object detection models from images and videos, 2018. 2

[25] Mourad Mourafiq et al. Polyaxon. https://github.com/polyaxon/polyaxon. 3

[26] Randal S. Olson et al. Tpot. http://epistasislab.github.io/tpot/. 3

[27] M. Papadomanolaki, S. Verma, M. Vakalopoulou, S. Gupta, and K. Karantzalos. Detecting urban changes with recurrent neural networks from multitemporal sentinel-2 data. In *IEEE International Geoscience and Remote Sensing Symposium*, pages 214–217, 2019. 7

[28] Paperspace. Gradient. https://gradient.run/. 3, 4

[29] QGIS Development Team. *QGIS Geographic Information System*. Open Source Geospatial Foundation, 2009. 2, 3

[30] Maryam Rahnemoonfar, Tashnim Chowdhury, Argho Sarkar, Debvrat Varshney, Masoud Yari, and Robin Murphy. Floodnet: A high resolution aerial imagery dataset for post flood scene understanding. *arXiv preprint arXiv:2012.02951*, 2020. 7

[31] CA: Environmental Systems Research Institute Redlands. Arcgis desktop: Release 10, 2011. 2, 3

[32] Axel Aronio De Romblay. Mlbox. https://mlbox.readthedocs.io/. 3

[33] Bryan C. Russell et al. Labelme: A database and web-based tool for image annotation. In *IJCV*, 2007. 2

[34] Boris Sekachev et al. opencv/cvat: v1.1.0, 2020. 2

[35] Labelbox Development Team. Labelbox, 2022. 2

[36] Sagar Verma, Akash Panigrahi, and Siddharth Gupta. Qfabric: Multi-task change detection dataset. In *Earthvision Workshop Computer Vision and Pattern Recognition (CVPR 2021)*, page 10, 2021. 7

[37] Sagar Verma and Jean-Christophe Pesquet. Sparsifying networks via subdifferential inclusion. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10542–10552, 2021. 6

[38] Weights and Inc. Biases. Weights and biases. https://wandb.ai/. 3