Technical Note

The use of multi-modal models and machine learning techniques to improve the efficiency and accuracy of geospatial data analysis

Matthew Gaskins

[1] mgaskins@captechu.edu

Featured Application: This research demonstrates potential improvements in the efficiency and accuracy of geospatial data analytics, which may enhance national security (e.g., surveillance, border control, and threat detection), disaster response, environmental conservation, urban planning, and resource management.

Abstract: Geospatial data analysis is heavily reliant on human interpretation of large-scale imagery which leads to constraints in scalability. This study evaluates whether multi-modal models can assist in overhead image understanding by accurately interpreting imagery and automating workflows. A hybrid machine learning solution using OversightML (OSML)—an open-source, cloud-based framework—is assessed for its ability to improve geospatial workflows. OSML integrates state-of-the-art computer vision with generative AI capabilities and streamlines preprocessing and detection aggregation. Results indicate that combining domain-specific CV models with foundation models offers a scalable and efficient alternative to manual analysis workflows [5, 9, 10, 12–21].

Keywords: Large language models, retrieval-augmented generation, computer vision, geospatial analysis, automated imagery processing, contextual data enrichment.

1. Introduction

Geospatial data analysis is heavily dependent on manual workflows that are both inefficient and resource intensive. The rapidly expanding volume of remote sensing imagery far exceeds the capacity for timely human analysis [8, 14]. Traditional methods primarily depend on analysts to manually interpret large-scale imagery, leading to bottlenecks, increased processing time, and higher operational costs. This reliance underscores the need for automated solutions capable of processing imagery quickly, accurately, and at scale.

Recent advancements in artificial intelligence (AI) and machine learning (ML) technologies offer promising solutions to these challenges. However, early automation efforts often involved rigid, rule-based systems requiring significant engineering effort and manual intervention for adjustments. These limitations have driven researchers and practitioners to explore more flexible, intelligent approaches.

This study addresses whether recent innovations in multi-modal and foundational ML models can enhance the efficiency and accuracy of geospatial data analysis. Specifically, it evaluates OSML, an open-source, cloud-based platform designed to automate and streamline remote sensing imagery analysis. OSML integrates advanced computer vision

Academic Editor: Firstname Lastname

Received: date
Revised: date
Accepted: date
Published: date

Citation: To be added by editorial staff during production.

(CV) technologies deployed via Amazon SageMaker and generative AI (GenAI) capabilities through Amazon Bedrock agents. This hybrid approach leverages pre-processing automation, including image decomposition, object detection aggregation, and contextual analysis of detected features [6, 12].

By combining specialized CV models with foundational GenAI resources, OSML aims to reduce the analytical burden on human experts significantly. This research investigates the potential for OSML to serve as a scalable, efficient alternative to manual workflows, with substantial positive implications for national security, disaster response, environmental monitoring, urban planning, and resource management [5, 9, 10, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21].

1.1. Details of OSML

OSML is an AI-powered, open-source geospatial toolkit designed to rapidly transform satellite and airborne sensor imagery into actionable insights. Built upon AWS's SageMaker and Bedrock platforms, OSML offers comprehensive tools to enhance efficiency and reduce costs associated with analyzing extensive geospatial datasets.

OSML includes several integrated components. The OSML Tile Server and Data Catalog provide efficient organization and retrieval of large imagery datasets stored in Amazon S3, facilitating quick access and processing of extensive imagery archives. Specialized pre-trained ML models, hosted on Amazon SageMaker, enable precise object detection and scene segmentation tasks tailored specifically for overhead imagery analysis.

The Distributed Computing Workflow, known as OSML Model Runner, manages the preprocessing of imagery data at scale. This component efficiently orchestrates thousands of SageMaker endpoint invocations, enabling comprehensive and rapid imagery analysis. Additionally, OSML Jupyter Extensions integrated within SageMaker Unified Studio offers intuitive tools for visualizing and interacting with geospatial imagery and data, streamlining user workflows and enhancing data interpretation.

Leveraging Amazon Bedrock agents, OSML Geo-Agents enhance contextual interpretation by using GenAI to analyze detected objects within the context of historical records, past analyses, geospatial metadata, and real-time data streams. This capability enables more precise and meaningful interpretations, significantly enriching the quality and depth of analytical outcomes.

The open-source nature of OSML supports customization and adaptability, allowing users from various sectors such as national security, disaster response, environmental conservation, urban planning, and agriculture, to adapt and extend the toolkit according to their specific mission requirements. Users can easily integrate additional models and tools developed by academia, industry, or public sector labs, aided by comprehensive documentation, reference architectures, and benchmarks provided by AWS. This flexibility ensures OSML remains scalable and responsive to diverse analytical needs. By utilizing OSML, this study aims to demonstrate how an open-source, well-supported infrastructure can significantly enhance the efficiency and accuracy of geospatial data analysis.

1.2. Past and Current Efforts in CV and ML

An ever-growing number of commercial and government-owned platforms have driven the geospatial community to adopt ML to help mine actionable information from the vast quantities of earth observation data. Past efforts have been primarily focused on training CV models to convert unstructured imagery into geographic features that represent objects of interest. Human analysts then combine that information with additional data sources to answer key intelligence questions. The acceleration of Bedrock's AI-powered agents provides an opportunity to expand the role of ML in this community.

Initial experiments show that multi-modal foundation models (FMs) like Anthropic's Claude 3.5 Sonnet v2 are not performing well on the core object detection and spatial reasoning tasks used to analyze remote sensing imagery, and they are not ready to be applied at full scale. Bedrock currently has a non-adjustable limit of 250 invocations per minute for this model so even a single small 50Kx50K satellite image broken into 2048 tiles would occupy a full account's regional capacity for several minutes [9, 10].

Image generators are too creative to produce maps with specific spatial/scale constraints. Instead of invoking the Bedrock model directly, experts proposed a hybrid approach where state-of-the-art CV is applied and hosted on SageMaker and Geographic Information System (GIS) technologies to augment the core GenAI capabilities.

For this study, OSML performs the requisite preprocessing work by decomposing the image into chunks, orchestrating the thousands of SageMaker Endpoint invocations, then geolocating detections and aggregating results. The resulting detections are then clustered, and the areas of the image found to contain objects of interest are cropped and sent to Bedrock for additional analysis. This approach uses the CV model to focus the attention of the more expensive GenAI resources which are then used to analyze the meaning of multiple objects in context.

2. Methods

2.1. Study Design

This study employs a hybrid approach to geospatial data analysis, integrating state-of-the-art CV alongside GIS technologies. The primary objective is to improve the efficiency and accuracy of processing and analyzing remote sensing imagery.

2.2. Data Collection

The data used in this study consists of imagery collected by satellites and unmanned aerial vehicles. These images are stored in Amazon S3 and are processed using new pre-trained ML models designed for object detection and scene segmentation.

2.3. Image Preprocessing

The preprocessing workflow involves decomposing the raw images into smaller tiles suitable for analysis. These tiles are then subjected to dynamic range adjustments, color balancing, and other operations to make them suitable for human review and visualization. The preprocessed tiles are stored in a common commercial format (PNG) and are used as input for the CV models.

2.4. Model Training and Deployment

The CV models are trained to take advantage of specific sensor features and incorporate details from the geospatial metadata available with the tiles. These models are deployed on Amazon SageMaker, where they are used to perform object detection and scene segmentation tasks.

2.5. Hybrid Approach

This hybrid approach involves using CV models to focus the attention of more expensive GenAI resources [5, 6, 12]. The CV models perform initial object detection and scene segmentation, while the GenAI models analyze the meaning of multiple objects in context. This approach leverages the strengths of both CV and GIS technologies to provide a comprehensive solution for geospatial data analysis.

2.6. *Workflow Orchestration*

The OSML solution orchestrates the entire workflow, from image preprocessing to model inference and result aggregation. This includes the use of new OSML-agent-tools containers, integration with Bedrock Agents, and extensions to the SageMaker Unified Studio managed Jupyter environment.

2.7. *Evaluation*

The performance of the hybrid approach is evaluated through a series of experiments [12, 13, 14]. These experiments assess the accuracy and efficiency of the CV and GenAI models in detecting and analyzing objects in remote sensing imagery. The results are compared with traditional methods to determine the effectiveness of the proposed solution.

2.8. *New OSML Features*

OSML supports a range of opportunities through the delivery of robust reusable components versus being focused on a single customer end-to-end workflow. The roadmap below illustrates the components supporting multiple areas of a typical image analysis enterprise.

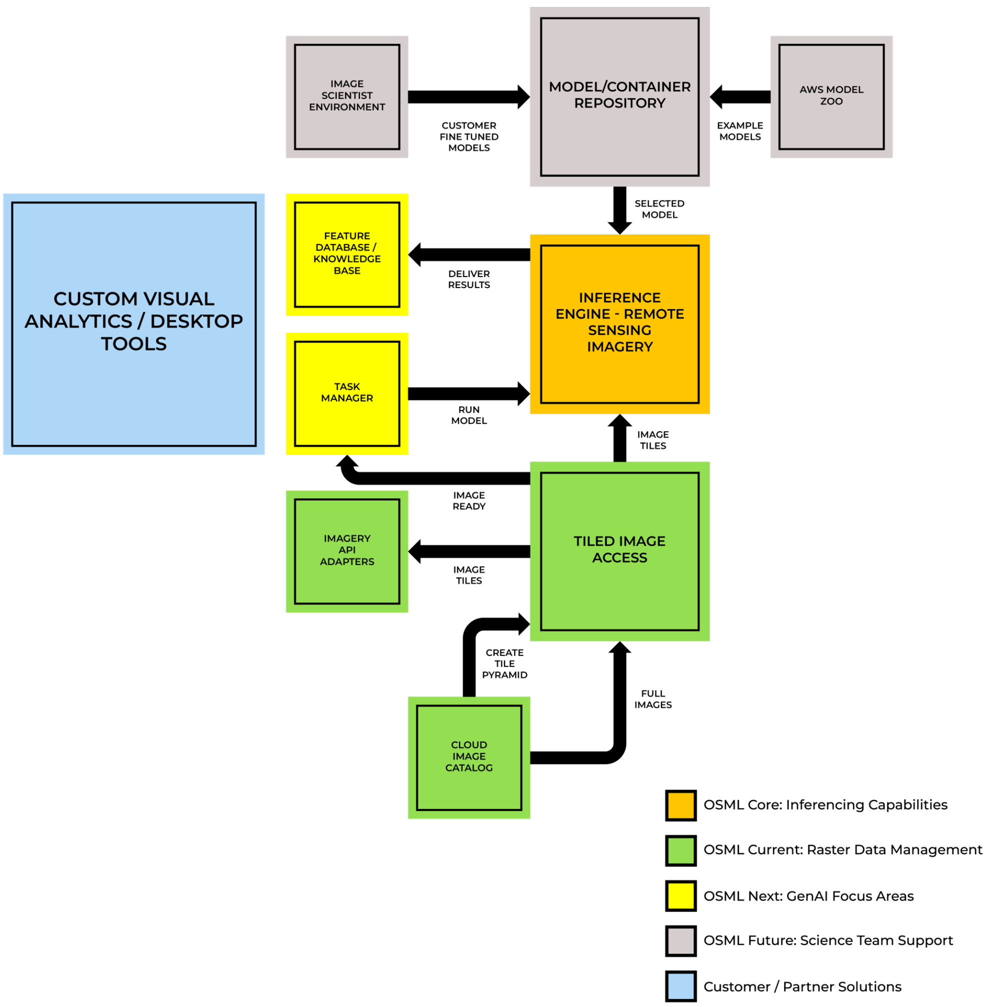

Figure 1. OSML Roadmap

2.9. Steps Used To Integrate System Into Workflow

The integration of OSML into the geospatial workflow follows a structured process designed to streamline remote sensing imagery analysis:

Initially, imagery data is managed through cloud-based image caching, creating optimized tiled image pyramids for efficient processing. These image tiles are accessed through a Tiled Image Access component, facilitating rapid retrieval and management.

Next, imagery Application Programming Interface (API) adapters are employed, which, together with a Task Manager and a Feature Database or Knowledge Base, interface with custom visual analytics or desktop tools. These steps ensure the imagery is systematically prepared for advanced analytical tasks.

The Task Manager directs image tiles to an Inference Engine designed specifically for remote sensing imagery analysis. This engine orchestrates the execution of pre-trained CV models hosted within a Model/Container Repository.

Upon execution, CV models detect and cluster relevant features within the imagery. The system then generates cropped image segments representing clusters of detected features. These segments are subsequently processed by multi-modal FMs which provide advanced analysis of object appearance and spatial arrangements.

Throughout the workflow, an ML Operations Engineer or Research Scientist facilitates iterative development and model refinement, allowing scientists and researchers to enhance the analytical capability continuously. This workflow ensures efficient integration and utilization of advanced machine learning techniques, significantly improving geospatial imagery analysis.

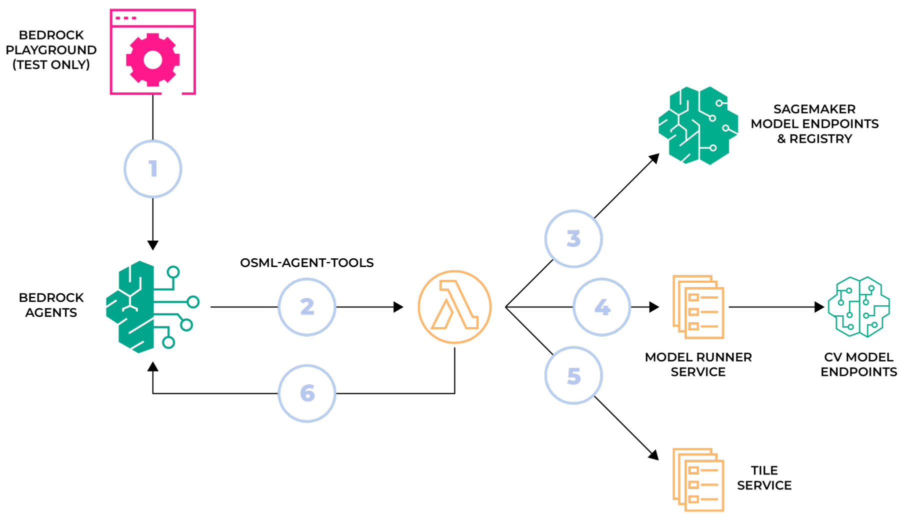

Figure 2. Steps Used for Integration

2.10. New OSML Agent Tools

New OSML-agent-tools-containers were created which host a Lambda runtime environment. The container utilizes a Python/GDAL environment and implements a single handler capable of responding to Bedrock Agent Events. Each event message contains agent, action group, and function name attributes, which are used to route the event to a specific tool handler. This architecture allows a single Lambda deployment to provide multiple tools for use. The implementation includes partial development of multiple tools required for an end-to-end image analysis workflow.

1. List-models: This tool lets the agent discover what SageMaker endpoints are available to analyze imagery. Perishable/account specific information of this kind cannot be trained into a model so it must be provided by a dynamic external source. For this study, a list of models was hard coded with each result containing a model name and a short one-sentence description of the model capabilities. (An actual implementation would query SM Endpoints or a model registry to find active endpoints that match selection criteria.) This part of the solution was judged to be low risk so there was no further effort applied.

2. Run-model: This tool allows an agent to run a CV model on an overhead image. Bedrock Agents have a limit of 25K characters on the Lambda response so this tool must summarize the raw GeoJSON results into a compact format that is suitable for analysis by the FMs. This effort prototyped an approach using density-based clustering of objects of like-type and then returning the count, type, and bboxes (both geo and image) as the result. For this study, the actual invocation of the model using OSML was skipped in favor of reading an example GeoJSON result from an external run. There is still some investigation to be done to understand how the ModelRunner invoke timelines align with these agents. As a fallback this processing might be a query into a feature store of existing results [9, 10].

3. Enrich-detections: This tool takes in the clustered detection results and invokes a multi-modal FM to analyze each group of detections. The invoke message for each group contains a crop of the image from the tile server along with the summary detections and prompt information used before. The call can be enhanced to include filters for a geospatial knowledge base (geospatial and temporal bounds taken from the image and cluster bounds).

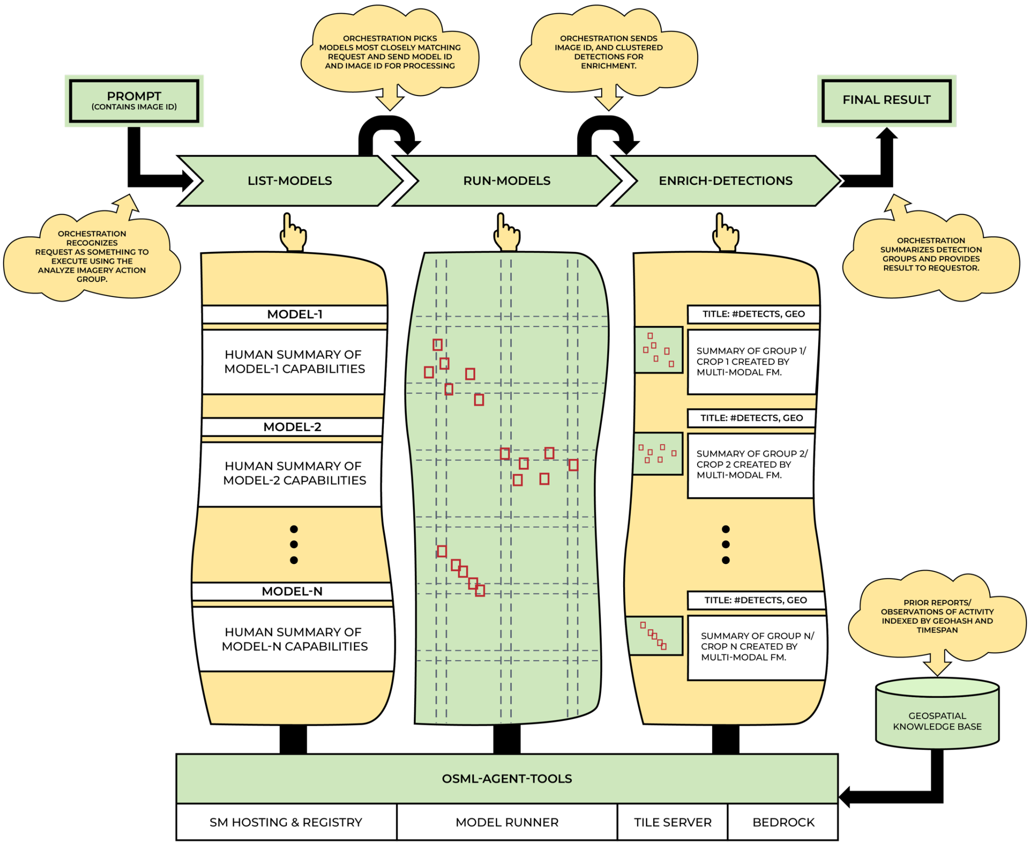

Figure 3: OSML Agent Tools

2.11. Image Preprocessing and Model Expectations

An important aspect of this study's solution is that the image pixels analyzed by the CV model and the image pixels processed by the multi-modal FM may have very different preprocessing steps. OSML sends raw unprocessed tiles from the original image to the CV model and these tiles are often of different types and bit depths (e.g., 11-bit per pixel panchromatic imagery, 32-bit 2-band floating point complex SAR data, 8-band multi-spectral imagery, etc.) This means that CV models can be trained to take advantage of specific sensor features and incorporate details from the geospatial metadata available with the NITF/SICD/SIDD tiles.

The tiles created by the TileServer have been preprocessed to either an 8-bit per pixel grayscale or a 3-band RGB image and they use a common commercial format (PNG). Part of this preprocessing involves operations such as dynamic range adjustments, and color balancing, necessary to make the original raw pixels suitable for human review and visualization. These tiles are similar to what the multi-modal foundation models may have seen in their generalized training sets, so they are provided instead of the scientific imagery fed to the CV model [9, 10].

2.12. Additional Steps in Integration

For this study, the following steps were taken to integrate the multimodal model.

1. CM control of the OSML-agent-tools hackathon prototype was established and brought up to the quality standards of our code baseline. An internal git repository was created to store the code, review the module structure, add comments, unit tests, and more.
2. List-models query to SageMaker Endpoints was implemented to identify models available in the account. A very simple Model Registry was investigated using tags on the endpoint for resources to be built. For this step, tag values were limited to 256 characters, so content had to be very terse. Longer values may need to be stored in S3 or some other external knowledge store to provide enough context for an FM to match a model to a task.
3. Run-model invocation of ModelRunner was implemented. CLIP/endpoint-based pull epics was prioritized and an option to send JSON requests was added instead of full tiles to SM endpoints. This was to make the ModelRunner process lightweight enough to be run within the tool Lambda. The GPU-based J2K decoding and GDAL-free NITF metadata parsing epic was implemented to help accelerate processing of the endpoints which already run on a graphics processing unit (GPU) and make it more efficient for the ModelRunner to parse the metadata necessary to calculate tiles/geolocate results.
4. Enrich-detections were improved by linking in the geospatial knowledge base to help interpret results in the last step. This was completed because the current workflow appeared to be using information trained directly into FM which was not current.
5. Size checks and prioritization were implemented and the number of groups processed during enrich-detections were limited. The flow is successful only if a relatively small number of clusters is of interest on any given image. Uninteresting types or low scoring detections were filtered out.
6. A geocoder was implemented, along with find-image tools, to convert well known place names to geographic coordinates and to find images in a Spatial Temporal Asset Catalog (STAC) matching geo-temporal criteria. Adding these tools allows analysts to start with prompts that use place names and times instead of specific image IDs. Utilities should be converted into image IDs to the S3 image URLs needed to task ModelRunner, but the prototype in this study did not require this because the results were hardcoded. A real system would need to determine the physical location of the image asset and may also need to create viewpoints for the data so it can be accessed via TileServer APIs.
7. Normally, the detection Internalized Resource Identifier (IRI) for each object type would have been mapped to a more human readable format with the run-model tool retrieving this information from the model registry. However, this step wasn't necessary in our prototype because the IRIs produced by our test model were human readable (e.g., "airplane") whereas real IRIs would not be suitable for use as-is and would need to be converted.
8. A demo-UI combining a geospatial viewer with a chat bot UI was created. This step presents an opportunity to visualize the geospatial results of these workflows.
9. Bedrock agents advanced prompting support was investigated to better relay the individual group summaries/URLs in the final output.

2.13. Evaluation via Example Execution of a Single Image

Shown below are the example results of asking Anthropic Claude 3.5 Sonnet v2 to analyze the following image.

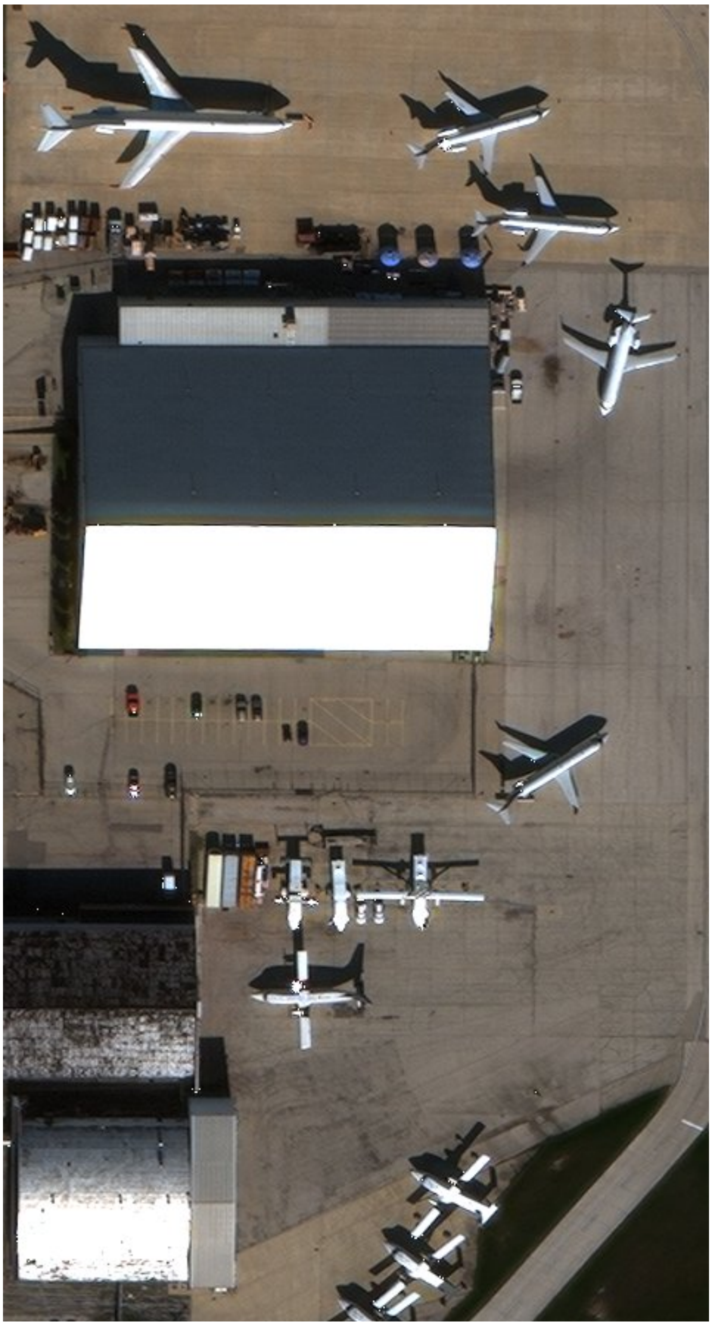

Figure 4: Sample Aerial Image

Result for Bedrock Agent Using OSML-agent-tools:

"This image appears to be an aerial view of what looks like an aircraft storage or maintenance facility. There are multiple commercial aircraft visible in the image, appearing to be in a white and dark color scheme. The aircraft are parked at various angles around a large rectangular building structure with what appears to be a partially white roof.

The facility includes:
A main building/hangar structure
A paved tarmac area
Several parked vehicles in what appears to be a parking lot
Multiple commercial aircraft (approximately 7-8 visible in the frame)
Some smaller structures or equipment on the ground

> The image is taken from directly above (likely by drone or aircraft), giving a clear bird's eye view of the entire facility. The ground appears to be concrete or asphalt in a grayish tone, and there's some green vegetation visible at the edges of the frame.
>
> The aircraft appear to be of similar size and model, suggesting this might be a maintenance or storage facility for a specific airline or aircraft type. The layout suggests this is likely part of an airport or aviation facility, though the image only shows this particular section"

Note that while the URLs mentioned were not actually included in the result, they were generated and are visible in the conversational history. Going forward, it will be necessary to adjust the prompts and/or agent guidance to ensure they are included in the output along with specific summaries of each area.

2.14. Evaluation via Prompt

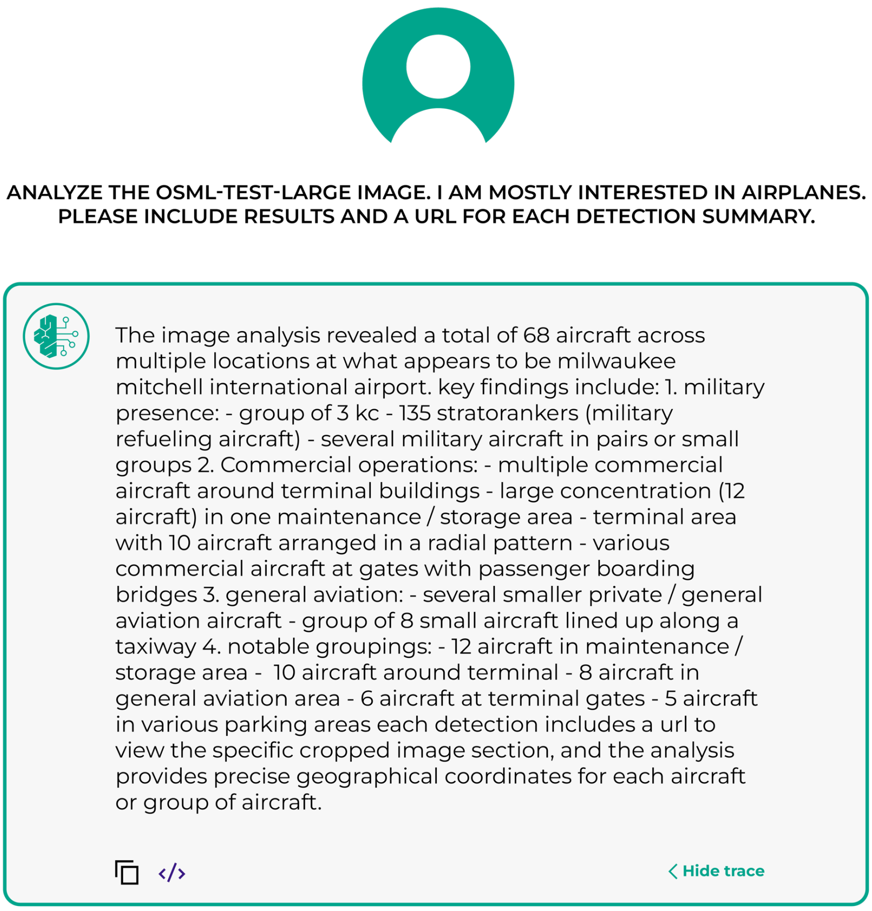

Figure 5: Bedrock Agent Prompt

The trace information below is from the Bedrock Agent orchestrated tasks in response to the prompt shown above.

Step 1: Execute List-Models


```
"rationale": {
  "text": "I'll help analyze the image focusing on airplanes. Let me first check what models are
    available and then run the appropriate analysis.\nFirst, I'll list the available models.",
  "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-0"
},
```



```
"invocationInput": [
  {
    "actionGroupInvocationInput": {
      "actionGroupName": "FindObjects",
      "executionType": "LAMBDA",
      "function": "ListModels"
    },
    "invocationType": "ACTION_GROUP",
    "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-0"
  }
],
```



```
"observation": [
  {
    "actionGroupInvocationOutput": {
      "text": "{\"result\": \"osml-airoob: Model for detecting aircraft and other air order of
        battle objects in EO imagery\\nosml-sar-ships: Model for detecting maritime vessels in SAR
        imagery\\nosml-generic-objects: Model for detecting ground motor vehicles and other
        uncategorized objects in EO imagery.\\n\"}"
    },
    "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-0",
    "type": "ACTION_GROUP"
  }
]
```


Step 2: Execute Run-Model

The invocation input of this step demonstrates that the orchestration model picked a reasonable model from the list of results based on the original prompt. "I am mostly interested in airplanes." → "osml-airoob: Model for detecting aircraft and other air order of battle objects in EO imagery." The model successfully pulled the model name and image ID parameters from the text and passed those to RunImage.


```
"rationale": {
  "text": "I see that the \"osml-airoob\" model is most appropriate for detecting aircraft. I'll
    run this model on the specified image.",
  "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-1"
},
```



```
"invocationInput": [
  {
    "actionGroupInvocationInput": {
      "actionGroupName": "FindObjects",
      "executionType": "LAMBDA",
      "function": "RunImage",
      "parameters": [
        {
          "name": "model_name",
          "type": "string",
          "value": "osml-airoob"
        },
        {
          "name": "image_id",
          "type": "string",
          "value": "OSML-test-large"
        }
      ]
    },
    "invocationType": "ACTION_GROUP",
    "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-1"
  }
],
```

These are the summary results generated by the run-model agent. Each line represents a cluster of like-type objects with the following fields. It may be more appropriate to return these as a structured values since the FM seems to be well trained on XML.

COUNT: <# in cluster>
TYPE: <obj type>
GEOBBOX: <min_lon, min_lat, max_lon, max_lat>
IMAGEBBOX: <min_x, min_y, max_x, max_y>


```
"observation": [
  {
    "actionGroupInvocationOutput": {
      "text": "{\"result\": \"COUNT: 12 TYPE: aircraft GEOBBOX: (-87.908425, 42.953095, -87.907038,
        42.955704) IMAGEBBOX: (5033.0, 3310.0, 5462.0, 4117.0)\\nCOUNT: 2 TYPE: aircraft GEOBBOX:
        (-87.913296, 42.942895, -87.912862, 42.943338) IMAGEBBOX: (3527.0, 7134.0, 3661.0, 7271.0
        )\\nCOUNT: 4 TYPE: aircraft GEOBBOX: (-87.88953, 42.941702, -87.887751, 42.942552)
        IMAGEBBOX: (10876.0, 7377.0, 11426.0, 7640.0)\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87
        .905997, 42.939173, -87.905861, 42.939286) IMAGEBBOX: (5784.0, 8387.0, 5826.0, 8422.0
        )\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.900932, 42.949369, -87.900716, 42.949579)
        IMAGEBBOX: (7350.0, 5204.0, 7417.0, 5269.0)\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.907682,
        42.95704, -87.907465, 42.957231) IMAGEBBOX: (5263.0, 2838.0, 5330.0, 2897.0)\\nCOUNT: 1
        TYPE: aircraft GEOBBOX: (-87.906239, 42.957438, -87.90611, 42.957538) IMAGEBBOX: (5709.0,
        2743.0, 5749.0, 2774.0)\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.895195, 42.958292, -87
        .895037, 42.958408) IMAGEBBOX: (9124.0, 2474.0, 9173.0, 2510.0)\\nCOUNT: 1 TYPE: aircraft
        GEOBBOX: (-87.888611, 42.955019, -87.888527, 42.95508) IMAGEBBOX: (11160.0, 3503.0, 11186.0
        , 3522.0)\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.903584, 42.950776, -87.903322, 42.950931)
        IMAGEBBOX: (6530.0, 4786.0, 6611.0, 4834.0)\\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.910333,
        42.94805, -87.910262, 42.948121) IMAGEBBOX: (4443.0, 5655.0, 4465.0, 5677.0)\\nCOUNT: 1
        TYPE: aircraft GEOBBOX: (-87.888314, 42.939994, -87.888242, 42.940068) IMAGEBBOX: (11252.0,
        8145.0, 11274.0, 8168.0)\\nCOUNT: 6 TYPE: aircraft GEOBBOX: (-87.901385, 42.950019, -87
        .899196, 42.951277) IMAGEBBOX: (7210.0, 4679.0, 7887.0, 5068.0)\\nCOUNT: 4 TYPE: aircraft
        GEOBBOX: (-87.91511, 42.942407, -87.914496, 42.943005) IMAGEBBOX: (2966.0, 7237.0, 3156.0,
```


Step 3: Execute Enrich-Detections


```
"rationale": {
  "text": "Let me now enrich these detections to get more detailed information about the aircraft
    configurations and relationships.",
  "traceId": "5d14653e-5c9a-43e5-a094-9a719307fbf9-2"
},
```



```
"invocationInput": [
  {
    "actionGroupInvocationInput": {
      "actionGroupName": "FindObjects",
      "executionType": "LAMBDA",
      "function": "EnrichDetections",
      "parameters": [
        {
          "name": "image_id",
          "type": "string",
          "value": "OSML-test-large"
        },
        {
          "name": "detections",
          "type": "string",
          "value": "COUNT: 12 TYPE: aircraft GEOBBOX: (-87.908425, 42.953095, -87.907038, 42.955704
            ) IMAGEBBOX: (5033.0, 3310.0, 5462.0, 4117.0)\nCOUNT: 2 TYPE: aircraft GEOBBOX: (-87
            .913296, 42.942895, -87.912862, 42.943338) IMAGEBBOX: (3527.0, 7134.0, 3661.0, 7271.0
            )\nCOUNT: 4 TYPE: aircraft GEOBBOX: (-87.88953, 42.941702, -87.887751, 42.942552)
            IMAGEBBOX: (10876.0, 7377.0, 11426.0, 7640.0)\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87
            .905997, 42.939173, -87.905861, 42.939286) IMAGEBBOX: (5784.0, 8387.0, 5826.0, 8422.0
            )\nCOUNT: 1 TYPE: aircraft GEOBBOX: (-87.900932, 42.949369, -87.900716, 42.949579)
```


In this example, experiments with the structured outputs and each cluster have been enriched. They included the following:

<detectionSummary>
<tagline> COUNT: ## TYPE: ... </tagline>

<url>https://tile-server.com/viewpoints/{image_id}/image/crop/{imagebbox}.PNG</url>

<summary> Text from the FM analysis of the cropped region. </summary>
</detectionSummary>


```
"observation": [
  {
    "actionGroupInvocationOutput": {
      "text": "{\"result\": \"<detectionSummary><tagline>COUNT: 12 TYPE: aircraft GEOBBOX: (-87
        .908425, 42.953095, -87.907038, 42.955704) IMAGEBBOX: (5033.0, 3310.0, 5462.0, 4117.0
        )</tagline><url>https://ie7utuk2tl.execute-api.us-west-2.amazonaws.com/viewpoints/OSML-test
        -large/image/crop/5008,3285,5487,4142.PNG</url><summary>This image appears to be an aerial
        view of what looks like an aircraft storage or maintenance facility. There are multiple
        commercial aircraft visible in the image, appearing to be in a white and dark color scheme.
        The aircraft are parked at various angles around a large rectangular building structure
        with what appears to be a partially white roof.\\n\\nThe facility includes:\\n- A main
        building/hangar structure\\n- A paved tarmac area\\n- Several parked vehicles in what
        appears to be a parking lot\\n- Multiple commercial aircraft (approximately 7-8 visible in
        the frame)\\n- Some smaller structures or equipment on the ground\\n\\nThe image is taken
        from directly above (likely by drone or aircraft), giving a clear bird's-eye view of the
        entire facility. The ground appears to be concrete or asphalt in a grayish tone, and
        there's some green vegetation visible at the edges of the frame.\\n\\nThe aircraft appear
        to be of similar size and model, suggesting this might be a maintenance or storage facility
        for a specific airline or aircraft type. The layout suggests this is likely part of an
        airport or aviation facility, though the image only shows this particular section.</summary
        ></detectionSummary\\n<detectionSummary><tagline>COUNT: 2 TYPE: aircraft GEOBBOX: (-87
        .913296, 42.942895, -87.912862, 42.943338) IMAGEBBOX: (3527.0, 7134.0, 3661.0, 7271.0
        )</tagline><url>https://ie7utuk2tl.execute-api.us-west-2.amazonaws.com/viewpoints/OSML-test
```


2.15. Preprocessor to Enable Spatio-Temporal Knowledge Bases

Retrieval-augmented generation (RAG) solutions are a common way to use information from data stores to augment the embedded knowledge of FMs trained on public information. Bedrock provides a Knowledge Base feature that helps manage the chunking and indexing of documents that is fully integrated with Bedrock hosted models and agents through API calls like RetrieveAndGenerate. In this example, each document portion is indexed by a combination of an embedding vector and an optional set of metadata properties. The embedding vector will handle the generic text matching, while the metadata attributes allow users to filter requests down to a specific geo-temporal region of interest.

Adding a containerized application can run as a SageMaker Processing Job to OSML-data-intake to extract geospatial and temporal metadata from documents and output the metadata.json files needed to feed a Bedrock Knowledge Store.

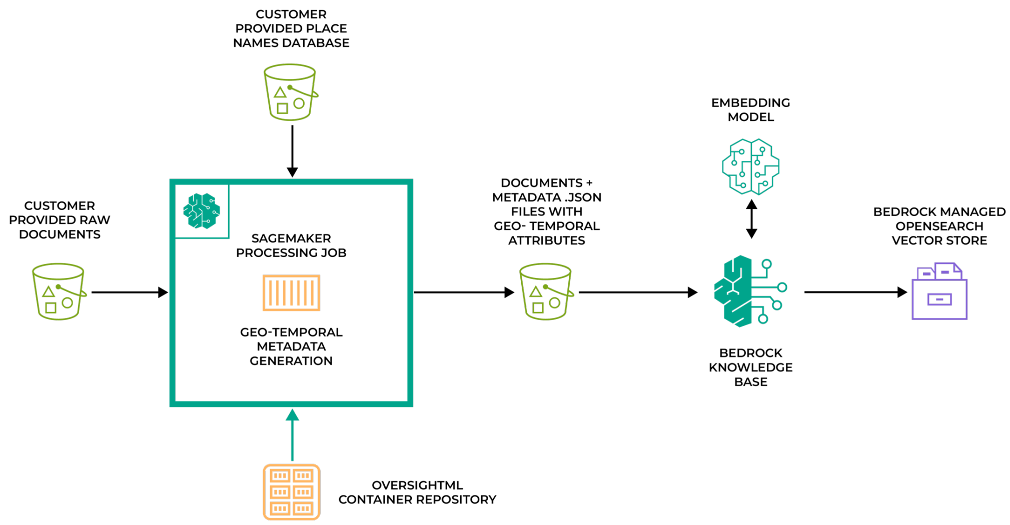

Figure 6: Retrieval-augmented generation (RAG)

These metadata files will contain new attributes such as geoCellIndex, startTime, and endTime, for each document allows the use of Bedrock, startsWith, in, greaterThanOrEquals, and lessThanOrEquals filter operators to limit the retrieved context to a specific geo-temporal region. This makes explicit use of the hierarchical nature of many geospatial cell indexing schemes (e.g., H3) which allows the use of a prefix search to identify documents linked to a parent and all child cells. In these schemes, the length of the prefix determines the requested resolution level (i.e., a longer prefix defines a narrower geospatial region) so documents and searches can be executed at a zoom level appropriate to the information.

```
{
  "metadataAttributes": {
    "geoCellIndex": "1210340...",
    "startTime": 1719329636,
    "endTime": 9999999999
  }
}
```

Above: The provided JSON snippet represents metadata attributes typically stored in a Bedrock Knowledge Base for use in geo-temporal retrieval tasks:

- geoCellIndex ("1210340..."): This is a reference to the H3 hexagonal grid cell identifier. It specifies a particular geographic region associated with stored data. Each cell index uniquely identifies a discrete area on Earth's surface.
- startTime (1719329636): This is a Unix timestamp indicating the beginning of the temporal range during which the referenced data or event is valid or relevant.
- endTime (9999999999): Another Unix timestamp, marking the end of the valid temporal range for the data. The provided large number (9999999999) generally indicates a far-future date, meaning the data has indefinite or ongoing validity.

Together, these attributes enable the retrieval of contextually relevant documents or data based on specific geographic and temporal parameters. The structured nature of this metadata facilitates precise, efficient queries within Bedrock Knowledge Base, allowing users to rapidly access data relevant to specific locations and time periods.

Below: The H3 hexagonal grid index is utilized within OSML to efficiently manage and query geo-temporal metadata. Each hexagonal cell represents a discrete geographic area with a unique identifier, enabling structured and scalable data indexing. In practice, a geographic query region—indicated by a red boundary—intersects multiple H3 cells, each containing geo-temporally indexed documents or datasets. Bedrock RetrievalFilters can then leverage these H3 cell identifiers (e.g., 12101, 12102, 12103) to precisely retrieve documents relevant to specific geographic areas and timeframes. This structured approach enhances the speed and accuracy of spatial queries and data analysis within geospatial workflows.

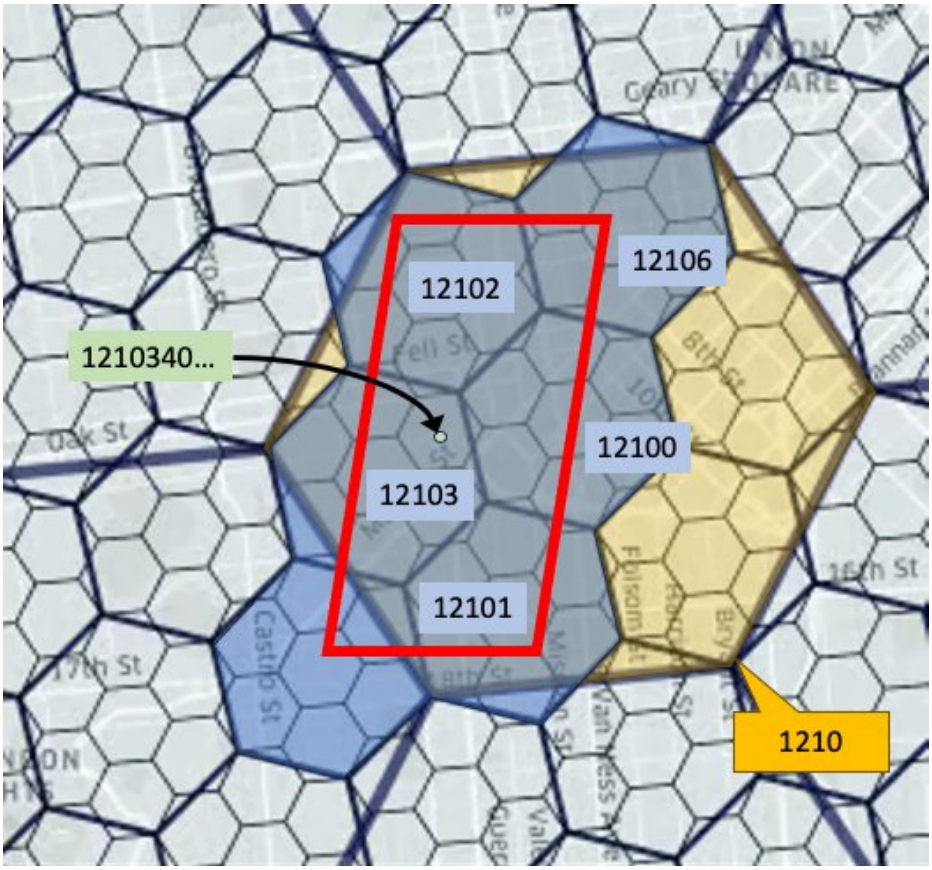

Figure 7: Example of H3 hexagonal grid index showing document location and the cells intersecting a sample query region.

```
{
  "andAll": [ {
    "orAll": [ {
      "startsWith": {
        "key": "geoCellIndex",
        "value": "12103"
      }}, {
      "startsWith": {
        "key": "geoCellIndex",
        "value": "12101"
```

```
        }}, ... # Remaining cell IDs omitted for brevity
      ], {
      "greaterThanOrEquals": {
        "key": "startTime",
        "value": 1719329636
      }}, {
      "lessThanOrEquals": {
        "key": "endTime",
        "value": 1719400000
      }}
    ]
  }
```

2.16. Library of Geospatial Agents

Augmenting geospatial workflows with ML actions is essential to automate routine processing steps, significantly advancing geospatial data analytics. The OSML-geospatial-agents library implements a suite of geospatial tools accessible through Bedrock agents.

This library includes the core software implementations and Lambda function handlers required for the seamless operation and interaction of geospatial analysis tools.

- The Gazetteer Tool translates textual place names into precise geographic coordinates, enabling seamless transitions between descriptive text and spatial data representations.
- The Atlas Tool allows users to execute advanced queries against Spatio-Temporal Asset Catalogs, retrieving information based on specific spatial and temporal constraints to facilitate precise and contextually accurate analyses.
- The Detection Tool facilitates the selection and execution of computer vision models hosted on the OSML platform, accurately identifying and localizing objects of interest within remote sensing imagery.
- The Enrichment Tool enhances the value of detected geospatial features by integrating supplementary properties extracted from structured GIS datasets and unstructured textual information sources, greatly enriching the contextual depth and analytical relevance.
- The Spatial Reasoning Tool provides robust capabilities for organizing, grouping, and analyzing geospatial features based on their spatial relationships and contextual attributes, supporting basic spatial comparisons and potentially extending to dynamic motion modeling.
- Finally, the Map and Image Annotation Tool generates visual annotations and custom maps, making it straightforward for users to visualize, interpret, and report analytical findings through geospatial graphics and annotations.

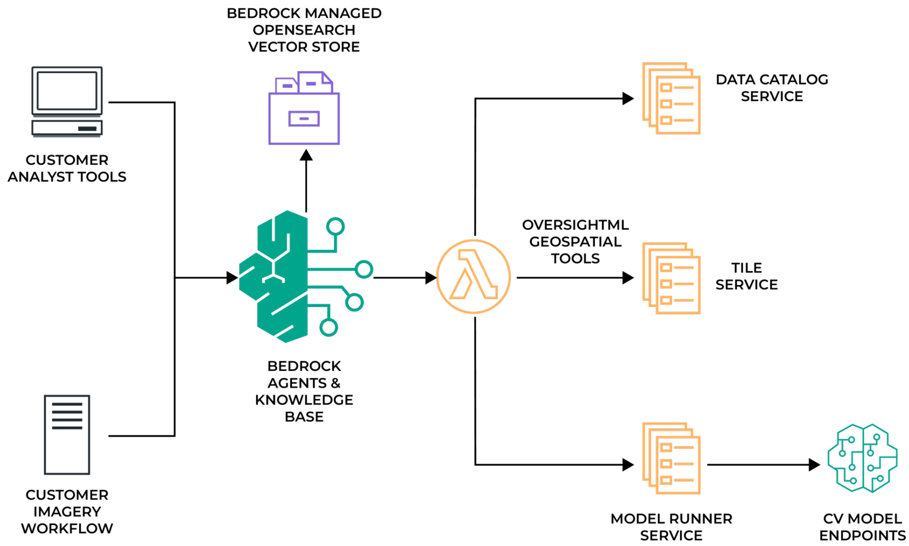

Figure 8: OversightML Architecture for Automated Geospatial Analysis.

The diagram illustrates the high-level system architecture of OSML, an open-source toolkit developed by AWS to automate geospatial imagery analysis using a hybrid of CV and GenAI tools. On the left, analysts and imagery ingestion workflows interact with OSML via analyst tools and automated pipelines. Central to the system is a Bedrock-managed knowledge base and agent layer, which enables interaction with downstream services and tools. The system uses a Bedrock Agent to mediate access to a suite of geospatial processing tools deployed as AWS Lambda functions. These tools interface with a data catalog and tile service to provide indexed access to imagery stored in S3, as well as a model runner service responsible for orchestrating calls to hosted CV models. The outputs from these services are made available to CV model endpoints, allowing for scalable execution of object detection and scene segmentation tasks. This architecture enables end-to-end orchestration of geospatial workflows, reducing analyst burden while allowing flexible integration of custom models and knowledge sources.

2.17. Map Generation Using Q

Many wrongly assume that image generation models are capable of being used in the generation of maps. This assumption is incorrect because most generators are too imaginative and imprecise to generate an accurate spatial representation. Instead, it is necessary to reframe the problem as a code generation task, using Amazon Q to generate XML/JSON/SVG that can be rendered as a map from descriptive prompts and feature sets. Some of these cases may already be supported as-is; the example below is a real result generated from an Amazon Q service without any special tuning.

Example Prompt:

To render object detection results over base imagery, this study uses Scalable Vector Graphics (SVG) as a lightweight overlay format. SVG offers native support for resolution-independent graphics, semantic tagging, and DOM-level interactivity, making it suitable for integration with browser-based and geospatial rendering engines.

In the proposed OSML workflow, each detection is represented as a <circle> element positioned by its pixel coordinates in the tile reference frame. These graphical markers are grouped by object type using the <g> tag to facilitate interactive rendering and styling (e.g., toggling object layers). To support geospatial integration, each element may include custom attributes (e.g., data-lat, data-lon, data-type) that encode real-world metadata extracted from the model outputs or geospatial transformation layers. This metadata is useful for downstream services that generate tooltips, link to external records, or filter detections based on spatial queries.

The example below demonstrates a minimal SVG output used to visualize three detection points over a 512×512 image tile. The red markers represent detected aircraft with metadata included for integration into the rendering stack:

```
<svg width="512" height="512" viewBox="0 0 512 512"
     xmlns="http://www.w3.org/2000/svg">
  <defs>
    <pattern id="background"
             patternUnits="userSpaceOnUse"
             width="512" height="512">
      <image
        href="https://dummyimage.com/512x512/000/fff.png"
        x="0" y="0" width="512" height="512" />
    </pattern>
  </defs>
  <rect width="512" height="512" fill="url(#background)" />
  <circle cx="20" cy="45" r="5" fill="red" />
  <circle cx="249" cy="250" r="5" fill="red" />
  <circle cx="145" cy="42" r="5" fill="red" />
</svg>
```

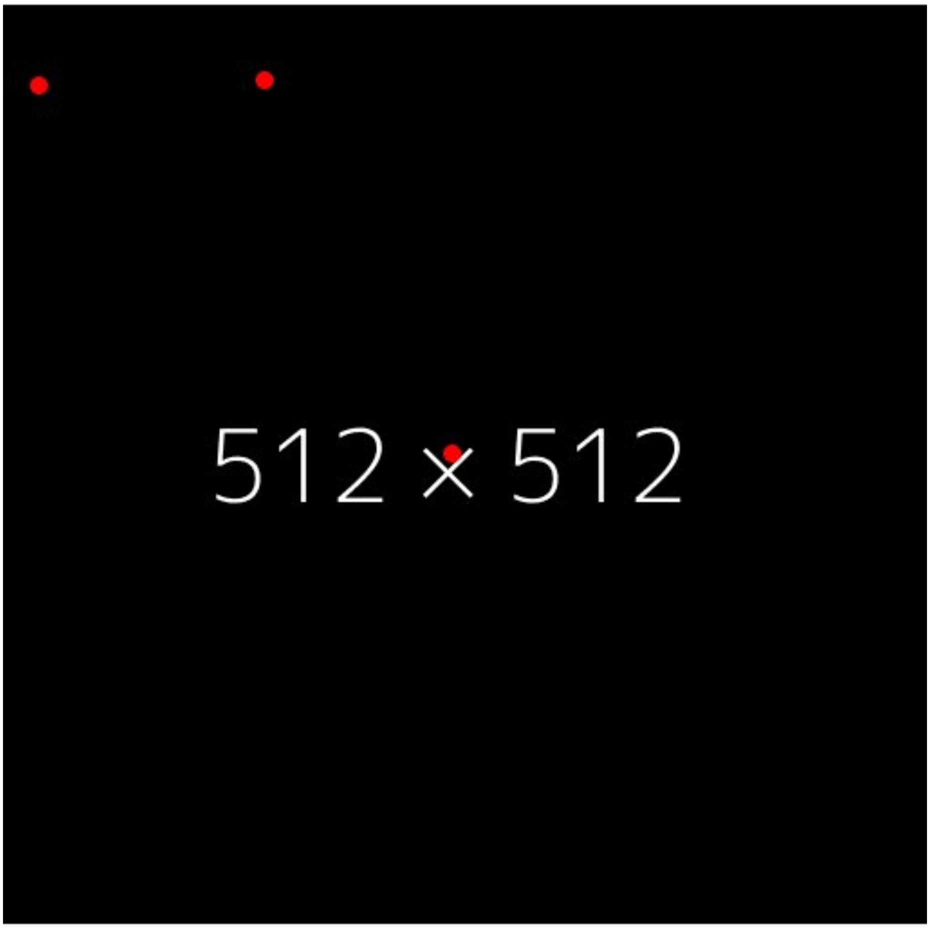

Figure 9: Example image result generated by sending the response through an SVG image renderer.

Combining these capabilities with existing OSML tile and object detection services should yield some demonstrable level of map generation capabilities built on top of Q. Some engineering capacity should be reserved to evaluate and explore the limits of this approach and recommend future fine tuning and customization efforts that can be undertaken later in the roadmap.

3. Results

The results of this study suggest the latest generation of multi-modal FMs is showing promise and, as a result, hybrid solutions that combine purpose-built CV models with GenAI technologies are likely to become more widely studied and adopted. While some engineering effort will be required to move beyond prototype systems to quantify the cost and infrastructure for operational scale deployments, the new OSML prototype allows GenAI, RAG's and Large Language Models (LLMs) to be added to a geospatial imagery processing workflow [9, 10].

3.1. Proposed Capability Demonstrations

The overall goal is to increase the efficiency of end user geospatial analysts, allowing them to assess a larger volume of data and make decisions more quickly than they can using traditional methods. Demonstrations should illustrate the following features.

1. Improved Detections: Today's leading models are only considering pixels and metadata and are therefore missing a key opportunity to include information from past observations when analyzing a new image. The proposed Spatio-Temporal Knowledge Base solutions provide an opportunity to tap into existing knowledge stores to enrich detections produced by state-of-the-art CV algorithms. Traditional object detection models will be used to localize the objects of interest (i.e., the where) while new GenAI-based components will provide additional details (i.e., the what and why). Overall, CV detections will demonstrate improvement in quality and accuracy, which may alter the plans for future model training efforts. Specifically, this may allow the community to use more generic object detectors instead of retraining new models for deep object type hierarchies.

2. Automated Reporting: Current ML workflows convert imagery into large GIS feature layers that analysts need to consume. ML is not actually helping analysts analyze the content or generate reports. New Geospatial Agents and Map Generation tools integrated into an analytic front end will allow analysts to automate steps in the report generation process. The analyst will guide the overall reasoning process while letting ML provide automation for repetitive steps. Overall, this should result in a reduction in the manual effort required from human analysts, resulting in a direct time savings when compared to current tradecraft.

3. Workflow Optimization: Current systems apply CV models broadly to imagery using simple rules-based orchestration to task inference engines. Using LLMs to work backwards from a key analysis question, experts will determine which datasets and models to run to answer key intelligence questions. This will demonstrate a more efficient allocation of computing resources through as-needed execution of models based on mission needs.

3.2. Key Technical Risks

Geocoding Service: This approach builds on an assumption that the geo-temporal context cab be extracted from a document necessary to populate attributes in the RAG metadata.json file. In some cases, this information is readily available as structured data in an existing knowledge base. In these cases, integration partners (e.g., ProServe or 3rd party LSIs) should be able to write an ETL process to produce the documents and metadata.json files that will match proposed conventions. In situations where larger collections of raw documents are presented, a geocoding service that can convert place names and coordinate references mentioned in the unstructured text to geographic coordinates will be required.

This geocoding problem is an area where current generations of generic LLMs are still behind their more traditional counterparts. Amazon Location Service provides basic lookup, but their SearchPlaceIndexForText API only accepts 200 characters of text, and the expectation is that this is an already extracted place name or address. The limitations of both traditional service offerings and current GenAI solutions leaves this as an opportunity for OSML to provide additional geospatial solutions.

Current state-of-the-art solutions are using smaller task-specific NLP models in conjunction with customer supplied gazetteers of canonical place names. One should be prepared to take contributions from the open-source community, adapt them to the data formats and conventions, then deliver a pre-packaged solution capable of filling this gap. This is a traditionally complex problem requiring incremental refinement of a solution over an extended period.

Model Fine-Tuning / Extensive Prompt Optimization: This approach attempts to work within any limitations imposed by the current generation of FMs provided by Bedrock and Amazon Q. As these capabilities are developed, there may be limits of the generic models which would make further progress dependent on model fine-tuning and more science driven prompt analysis tasking.

Alternatives Considered: This proposal was deliberately built atop the existing Bedrock service APIs instead of pursuing a more generic solution based on completely open-source alternatives (e.g., LiteLLM Proxy, LangChain Agents, OpenAI Model APIs, etc.).

4. Discussion

The findings of this study highlight the potential of integrating state-of-the-art CV and GIS technologies to enhance geospatial data analysis. The hybrid approach proposed in this research, which combines CV models with GenAI resources, has shown promise in addressing the inefficiencies of traditional methods.

4.1. Interpretation of Results

The initial experiments demonstrated that while multi-modal foundation models like Anthropic's Claude 3.5 Sonnet v2 are not yet performing well on core object detection tasks, the hybrid approach that integrates CV and GIS technologies holds significant potential1. By leveraging CV models to perform initial object detection and scene segmentation and then using GenAI models to analyze the meaning of multiple objects in context, the proposed solution offers a more efficient and accurate method for geospatial data analysis [9, 10].

4.2. Implications for Geospatial Workflows

The implementation of the OSML solution has the potential to revolutionize geospatial workflows. By automating routine processing steps, human analysts can focus on more complex tasks, thereby increasing overall efficiency. This approach also allows for the integration of new pre-trained ML models for object detection and scene segmentation, which can be deployed on Amazon SageMaker. The fully managed, distributed computing workflow and extensions to the SageMaker Unified Studio managed Jupyter environment further enhance the capabilities of the solution.

4.3. Limitations

Despite the promising results, there are limitations to the current approach. The performance of multi-modal foundation models is still not at the level required for full-scale deployment. Additionally, the Bedrock model has a non-adjustable limit of 250 invocations per minute, which can be a bottleneck when processing large satellite images. Further research and development are needed to address these limitations and improve the scalability of the solution [9, 10].

4.4. Future Research

The integration of multi-modal ML techniques into geospatial data analysis presents a multitude of avenues for future exploration. Building upon the current study, several key research directions are proposed:

1. Development of Scalable FMs for Geospatial Data

The advent of FMs—large-scale, pre-trained models adaptable to various tasks—has revolutionized natural language processing and CV. Extending this paradigm to geospatial data involves creating models capable of understanding and processing diverse data types, including satellite imagery, hyperspectral data, and spatial-temporal datasets. Recent work has introduced architectures such as the Low-rank Efficient Spatial-Spectral Vision Transformer (LESS ViT) [10, 11] which is designed to handle the unique challenges of multi-modal and hyperspectral geospatial data. LESS ViT approximates high-dimensional spatial-spectral attention through low-dimensional components, offering a promising direction for scalable geospatial analysis.

2. Enhancing Multi-Modal Alignment and Fusion Techniques

Effectively integrating heterogeneous data sources—such as combining satellite imagery with textual reports or sensor data—remains a critical challenge. Advancements in multi-modal alignment and fusion, including the development of sophisticated attention mechanisms and transformer architectures, have shown potential in improving the accuracy and applicability of geospatial models. Future research should focus on refining these techniques to better capture the complex relationships inherent in geospatial data.

3. Addressing Data Scarcity and Bias in Geospatial AI

The effectiveness of AI models is often hindered by limited or biased geospatial datasets. Emerging frameworks, such as graph neural networks and transformers, offer opportunities to learn from non-Euclidean relationships and perform parallel computations at scale. Future studies should prioritize the development of methods to mitigate data scarcity and bias, ensuring more robust and generalizable geospatial AI applications.

4. Advancements in Deep Learning for Multi-Modal Remote Sensing Data Fusion

Deep learning has significantly impacted the fusion of multi-modal remote sensing data, enabling more comprehensive analysis of complex geospatial phenomena. A comprehensive review highlights the versatility of ML methods in addressing a wide range of geospatial analysis challenges, including traffic anomaly detection, image fusion, and semantic segmentation. Future research should continue to explore and enhance deep learning techniques for integrating diverse remote sensing data sources.

5. Integration of AI in Disaster Response and Urban Planning

Artificial intelligence is increasingly utilized to enhance response and monitoring systems for natural disasters, particularly in urban areas. Applications range from improving forecasting accuracy and data collection to real-time disaster response and public alert systems. Future research should focus on integrating AI technologies into disaster preparedness and urban planning to improve resilience and response strategies.

6. Development of Geospatial FMs

The creation of large-scale, pre-trained models specifically designed for geospatial data, known as geospatial FMs, represents a promising research direction. These models aim to interpret complex patterns in location data, facilitating applications in environmental monitoring, urban planning, and disaster response. Future research should focus on developing such models to fully harness their potential.

7. Enhancing Explainability and Human-AI Interaction

As AI systems become more complex, ensuring their decisions are interpretable is crucial, especially in critical applications such as disaster response and urban planning. Research into explainable AI aims to improve AI reasoning and allow human guidance to correct AI decisions, enhancing trust and collaboration between humans and AI systems. Future research should focus on developing methods to enhance the explainability of AI systems and improve human-AI interaction.

8. Advancements in Spatial Embedding Techniques

Spatial embedding techniques, which involve representing spatial data in a continuous vector space, have shown promise in improving the performance of geospatial analyses. These techniques can effectively handle various data types, including text, images, and graphs, facilitating more accurate and efficient analyses. Future research should focus on advancing spatial embedding techniques to enhance geospatial data analysis.

9. Integration of AI in Earth Sciences

ML has been increasingly applied in earth sciences, enabling more accurate mapping, prediction, and analysis of geological phenomena. Applications include geological mapping, mineral prospectivity mapping, and environmental monitoring. Future research should focus on integrating AI techniques into earth sciences to enhance our understanding and management of geological processes.

5. Conclusion

This paper addresses the growing challenge of analyzing large-scale geospatial imagery by evaluating the use of multi-modal ML techniques in combination with domain-specific CV models. Through the implementation and assessment of OSML, an open-source, cloud-native toolkit developed on AWS, the study demonstrates a scalable and efficient alternative to traditional, labor-intensive geospatial workflows.

OSML integrates pre-trained CV models deployed on Amazon SageMaker with GenAI capabilities from Bedrock agents to automate and augment the image analysis pipeline.

The system performs image tiling, object detection, and feature clustering before passing relevant regions to large foundation models for contextual enrichment. New geospatial agent tools and orchestration layers were developed to coordinate this hybrid workflow and interface with structured knowledge sources, tile servers, and inference engines.

The study demonstrates that using CV to focus the attention of more resource-intensive foundation models can significantly reduce costs and improve analytical throughput. It also shows how RAG and agent-guided orchestration can add interpretability and context to raw object detections, increasing the utility of results for human analysts.

This study develops and tests a prototype pipeline that successfully:

- Decomposes remote sensing images into manageable tiles;
- Executes detection models at scale using distributed orchestration;
- Routes clustered detections into a generative analysis step;
- Enables human-readable visualization and reporting via structured outputs and SVG overlays; and
- Supports integration with spatio-temporal knowledge stores.

These findings support the broader hypothesis that hybrid AI systems combining CV and GenAI components offer a practical path forward for high-volume geospatial workflows. This approach balances performance and cost, while preserving interpretability and extensibility. As data volumes continue to increase, tools like OSML may offer the geospatial community a more sustainable way to process, analyze, and act on remote sensing imagery at operational scale.

Future research should focus on improving foundation model alignment with geospatial tasks, expanding model coverage for multi-sensor imagery, and refining orchestration tools for real-time and disconnected environments.

References

1. Janowicz, K.; Gao, S.; McKenzie, G.; Yingjie, H.; Bhaduri, L. GeoAI: Spatially Explicit Artificial Intelligence Techniques for Geographic Knowledge Discovery and Beyond. *International Journal of Geographical Information Science*. **2020**, *April*, 1-13
2. Li, Z.; Ning, H. Autonomous GIS: The Next-Generation AI-Powered GIS. *Autonomous Intell. Syst.* **2021**, *1*, 3. https://doi.org/10.1007/s43684-021-00003-0
3. Yang, H.; Liu, Y.; Han, Q.; Xu, L.; Zhang, T.; Wang, Z.; Yan, A.; Zhao, S.; Han, J.; Wang, Y. Improved Landslide Deformation Prediction Using Convolutional Neural Network–Gated Recurrent Unit and Spatial–Temporal Data. *Remote Sens.* **2025**, *17*, 727. https://doi.org/10.3390/rs17040727
4. Li, H.; Zhang, X.; Qu, H. DDFAV: Remote Sensing Large Vision Language Models Dataset and Evaluation Benchmark. *Remote Sens.* **2025**, *17*, 719. https://doi.org/10.3390/rs17040719
5. Lee, S.; Han, J.; Kim, Y. Unlocking the Power of Multi-Modal Learning for Geospatial Images. *DeepBlock* [Online], 2023. https://www.deepblock.net/blog/unlocking-the-power-of-multi-modal-learning-for-geospatial-images (accessed Apr 5, 2025).
6. Zhou, M.; Lu, X.; Chen, M.; Chen, T. Geospatial Mapping of Distribution Grid with Machine Learning and Multi-Modal Data. *Nat. Commun.* **2023**, *14*, 4294. https://doi.org/10.1038/s41467-023-39647-3

7. Zhao, P.; Li, Y.; Xu, H.; Zhang, Z.; Hu, Y.; Cao, X.; Li, Y. A Multi-Modal Geospatial–Temporal LSTM Based Deep Learning Framework for Urban Mobility Prediction. *Sci. Rep.* **2024**, *14*, 5584. https://doi.org/10.1038/s41598-024-74237-3
8. Kang, Y.; Liu, Y.; Li, W.; Zhang, H. Multimodal Deep Learning for GeoAI. In *GeoAI: Machine Learning and AI for Spatial Analysis*; Elsevier, **2023**. https://www.sciencedirect.com/science/article/abs/pii/B9780443190773000146
9. Zhou, T.; Liu, J.; Shen, H.; Li, Z. Towards Scalable Foundation Models for Multi-Modal and Hyperspectral Geospatial Data. *arXiv* Preprint arXiv:2503.12843, **2025**. https://arxiv.org/abs/2503.12843
10. Li, W.; Kang, Y.; Janowicz, K.; Xu, B. On the Opportunities and Challenges of Foundation Models for Geospatial Artificial Intelligence. *arXiv* Preprint arXiv:2304.06798, 2023. https://arxiv.org/abs/2304.06798
11. Huang, Q.; Zhang, Y.; Tu, W.; Shen, Y. Urban2Vec: Incorporating Street View Imagery and POIs for Multi-Modal Urban Neighborhood Embedding. *arXiv* Preprint arXiv:2001.11101, **2020**. https://arxiv.org/abs/2001.11101
12. Guiney, R.; Santucci, E.; Valman, S.; Booth, A.; Birley, A.; Haynes, I.; Marsh, S.; Mills, J. Integration and Analysis of Multi-Modal Geospatial Secondary Data to Inform Management of at-Risk Archaeological Sites. *ISPRS Int. J. Geo-Inf.* **2021**, *10*, 575. https://doi.org/10.3390/ijgi10090575. Multi-Modal Geospatial and Thematic Data to Foster Green Deal Applications. *ISPRS Arch.* **2023**, *XLVIII-1-W3*, 9. https://isprs-archives.copernicus.org/articles/XLVIII-1-W3-2023/9/2023/
13. Su, C.; Hu, X.; Meng, Q.; Zhang, L.; Shi, W.; Zhao, M. A Multimodal Fusion Framework for Urban Scene Understanding and Functional Identification Using Geospatial Data. *Int. J. Appl. Earth Obs. Geoinf.* **2024**, *127*, 103696. https://doi.org/10.1016/j.jag.2024.103696. Geospatial Big Data: Theory, Methods, and Applications. *Ann. GIS* **2024**, *30*, 1–12. https://www.tandfonline.com/doi/full/10.1080/19475683.2024.2419749
14. Huang, T.; Wang, Z.; Sheng, H.; Ng, A. Y.; Rajagopal, R. Learning Neighborhood Representation from Multi-Modal Multi-Graph: Image, Text, Mobility Graph and Beyond. *arXiv* Preprint arXiv:2105.02489, **2021**. https://arxiv.org/abs/2105.02489
15. Afroosheh, S.; Askari, M. Geospatial Data Fusion: Combining Lidar, SAR, and Optical Imagery with AI for Enhanced Urban Mapping. *arXiv* Preprint arXiv:2412.18994, **2024**. https://arxiv.org/abs/2412.18994
16. Mateu, J.; Montes, F., Eds. *Spatial Statistics Through Applications*; WIT Press / Computational Mechanics: Southampton, UK, **2002**.
17. Montero, J.-M.; Fernández-Avilés, G.; Mateu, J. Spatial and Spatio-Temporal Geostatistical Modeling and Kriging; Wiley: Hoboken, NJ, **2015**.
18. Mateu, J.; Giraldo, R., Eds. *Geostatistical Functional Data Analysis*; Wiley: Hoboken, NJ, **2021**.
19. Mitchell, H. B. *Data Fusion: Concepts and Ideas*; Springer: Berlin, **2012**. https://en.wikipedia.org/wiki/Data_fusion