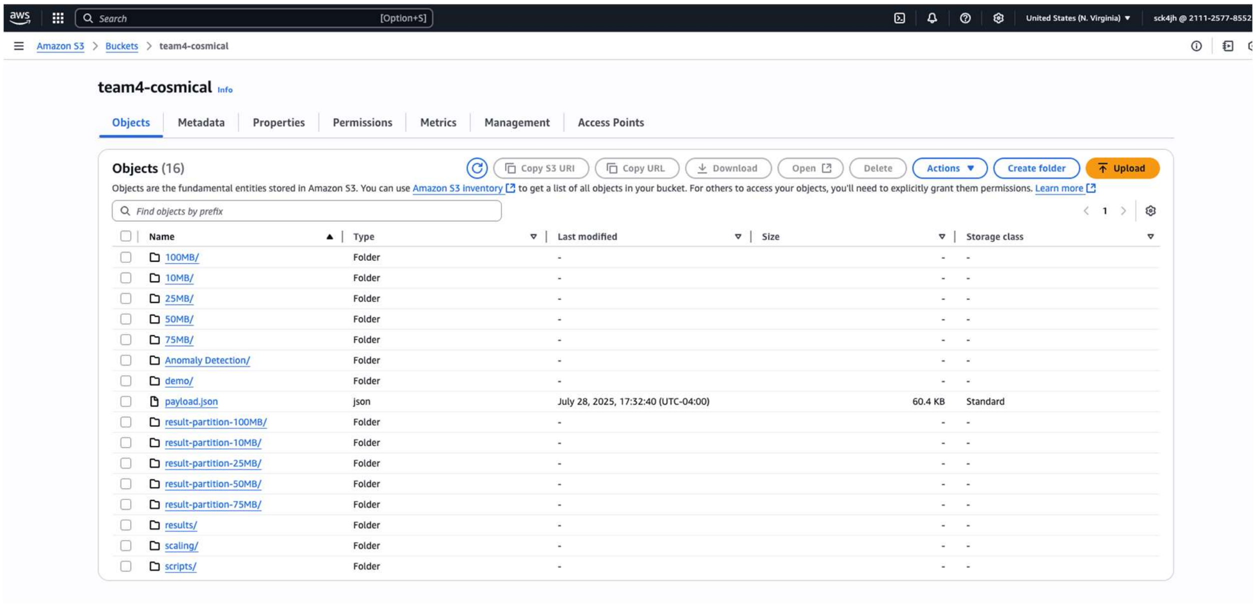
<u>Project Step 4</u>

Sam Knisely, Darreion Bailey

1. Create an S3 bucket with "results" and "scripts" folders
2. Clone the AI for Astronomy repository ([https://github.com/mstaylor/AI-for-Astronomy.gitLinks to an external site.](https://github.com/mstaylor/AI-for-Astronomy.git))
3. Copy the Anomaly Detection folder to your S3 bucket under the scripts path

- The below screenshot shows our S3 bucket, team4-cosmical, with the results and scripts folders and the cloned AI for Astronomy repository. We also coped the AI for Astronomy folder into the scripts folder as seen in the second screenshot.

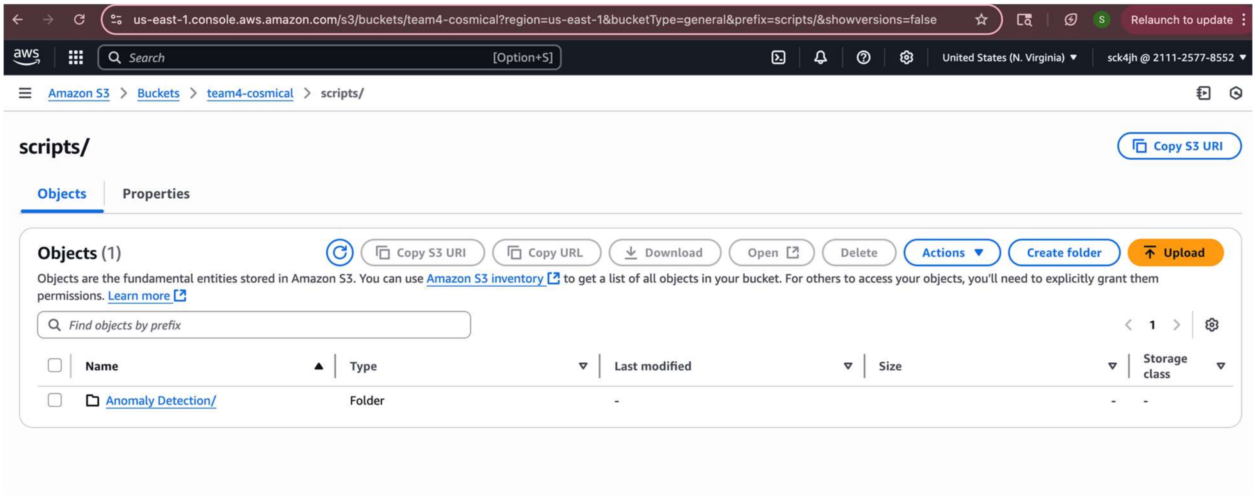


team4-cosmical

Objects (16)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
100MB/	Folder	-	-	-
10MB/	Folder	-	-	-
25MB/	Folder	-	-	-
50MB/	Folder	-	-	-
75MB/	Folder	-	-	-
Anomaly Detection/	Folder	-	-	-
demo/	Folder	-	-	-
payload.json	json	July 28, 2025, 17:32:40 (UTC-04:00)	60.4 KB	Standard
result-partition-100MB/	Folder	-	-	-
result-partition-10MB/	Folder	-	-	-
result-partition-25MB/	Folder	-	-	-
result-partition-50MB/	Folder	-	-	-
result-partition-75MB/	Folder	-	-	-
results/	Folder	-	-	-
scaling/	Folder	-	-	-
scripts/	Folder	-	-	-


us-east-1.console.aws.amazon.com/s3/buckets/team4-cosmical?region=us-east-1&bucketType=general&prefix=scripts/&showversions=false
S
Relaunch to update
aws
Search [Option+S]
United States (N. Virginia) ▼
sck4jh @ 2111-2577-8552 ▼
Amazon S3 > Buckets > team4-cosmical > scripts/
Copy S3 URI
Objects
Properties
Copy S3 URI
Copy URL
Download
Open
Delete
Actions ▼
Create folder
Upload
Find objects by prefix
1

scripts/

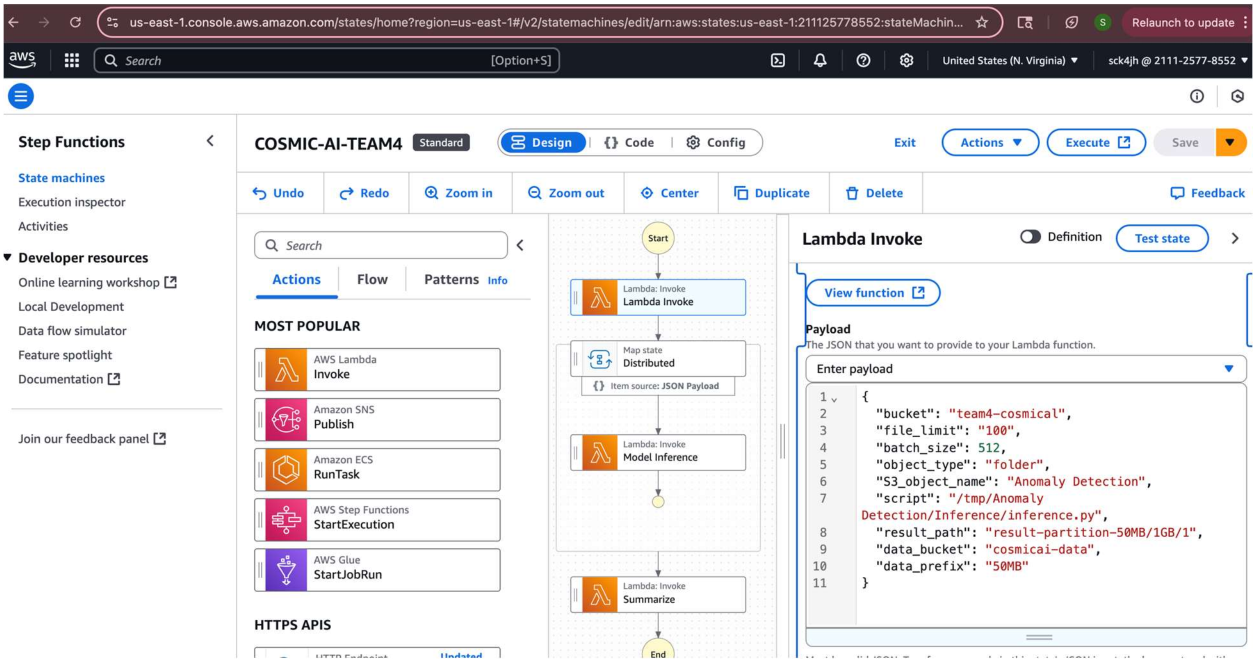
Objects (1)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
Anomaly Detection/	Folder	-	-	-

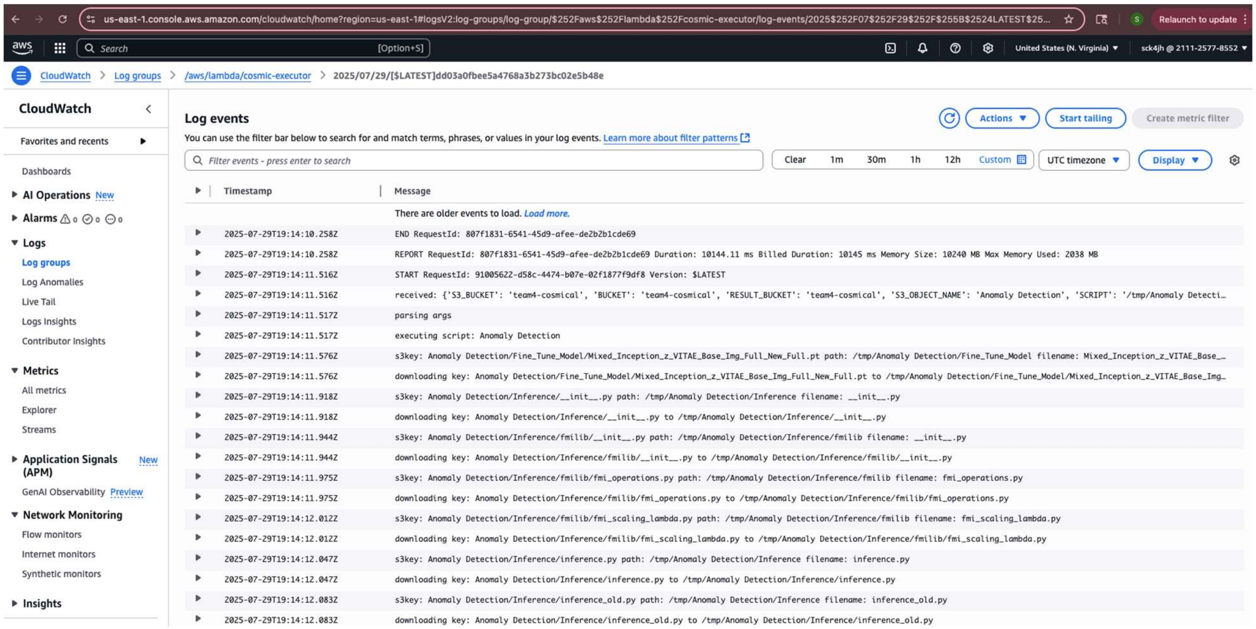
4. Configure the Step Function input payload with your bucket name, world size, and correct paths

    - The step function invoke statement can be modified as seen below with bucket names, world sizes, batch sizes, paths to different partitions of data, and paths to the scripts, data, and output results.



```
{
  "bucket": "team4-cosmical",
  "file_limit": "100",
  "batch_size": 512,
  "object_type": "folder",
  "S3_object_name": "Anomaly Detection",
  "script": "/tmp/Anomaly Detection/Inference/inference.py",
  "result_path": "result-partition-50MB/1GB/1",
  "data_bucket": "cosmicai-data",
  "data_prefix": "50MB"
}
```

5. Execute the step function and monitor the CloudWatch logs at /aws/lambda/cosmic-executor



CloudWatch > Log groups > /aws/lambda/cosmic-executor > 2025/07/29/[$LATEST]dd03a0fbee5a4768a3b273bc02e5b48e

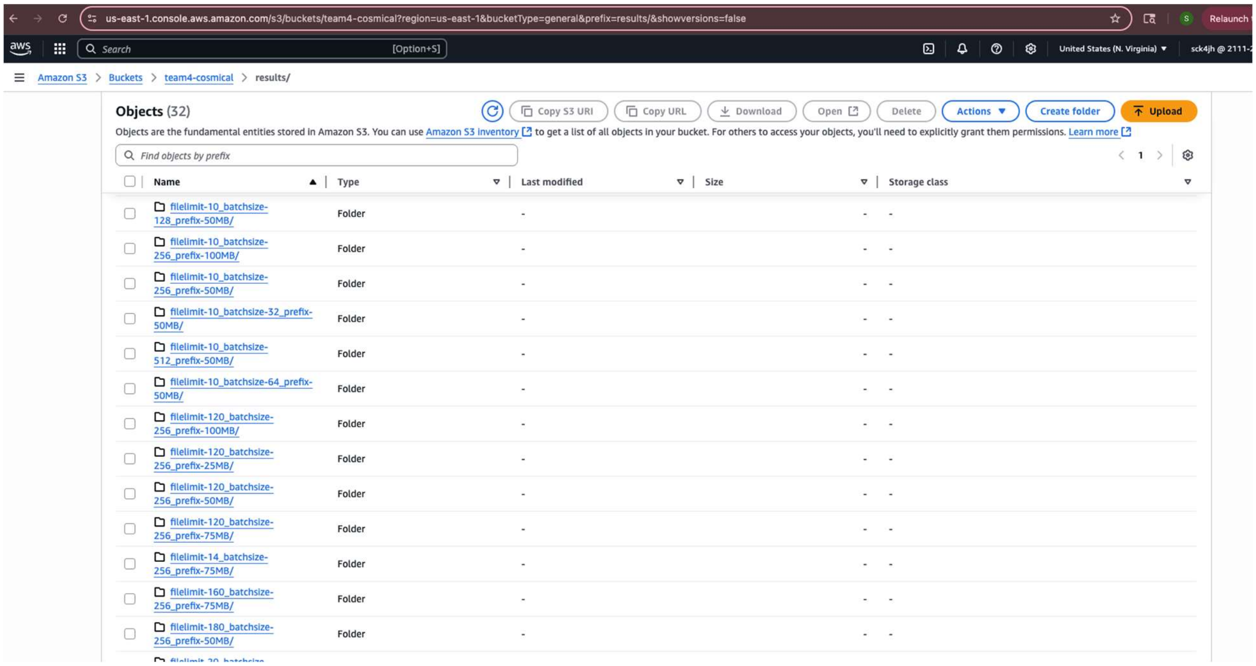
Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                 Message
                          There are older events to load. Load more.
2025-07-29T19:14:10.258Z  END RequestId: 807f1831-6541-45d9-afee-de2b2b1cde69
2025-07-29T19:14:10.258Z  REPORT RequestId: 807f1831-6541-45d9-afee-de2b2b1cde69 Duration: 10144.11 ms Billed Duration: 10145 ms Memory Size: 10240 MB Max Memory Used: 2038 MB
2025-07-29T19:14:11.516Z  START RequestId: 91005622-d58c-4474-b07e-02f1877f9df8 Version: $LATEST
2025-07-29T19:14:11.516Z  received: {'S3_BUCKET': 'team4-cosmical', 'BUCKET': 'team4-cosmical', 'RESULT_BUCKET': 'team4-cosmical', 'S3_OBJECT_NAME': 'Anomaly Detection', 'SCRIPT': '/tmp/Anomaly Detecti…
2025-07-29T19:14:11.517Z  parsing args
2025-07-29T19:14:11.517Z  executing script: Anomaly Detection
2025-07-29T19:14:11.576Z  s3key: Anomaly Detection/Fine_Tune_Model/Mixed_Inception_z_VITAE_Base_Img_Full_New_Full.pt path: /tmp/Anomaly Detection/Fine_Tune_Model filename: Mixed_Inception_z_VITAE_Base_…
2025-07-29T19:14:11.576Z  downloading key: Anomaly Detection/Fine_Tune_Model/Mixed_Inception_z_VITAE_Base_Img_Full_New_Full.pt to /tmp/Anomaly Detection/Fine_Tune_Model/Mixed_Inception_z_VITAE_Base_Img…
2025-07-29T19:14:11.918Z  s3key: Anomaly Detection/Inference/__init__.py path: /tmp/Anomaly Detection/Inference filename: __init__.py
2025-07-29T19:14:11.918Z  downloading key: Anomaly Detection/Inference/__init__.py to /tmp/Anomaly Detection/Inference/__init__.py
2025-07-29T19:14:11.944Z  s3key: Anomaly Detection/Inference/fmilib/__init__.py path: /tmp/Anomaly Detection/Inference/fmilib filename: __init__.py
2025-07-29T19:14:11.944Z  downloading key: Anomaly Detection/Inference/fmilib/__init__.py to /tmp/Anomaly Detection/Inference/fmilib/__init__.py
2025-07-29T19:14:11.975Z  s3key: Anomaly Detection/Inference/fmilib/fmi_operations.py path: /tmp/Anomaly Detection/Inference/fmilib filename: fmi_operations.py
2025-07-29T19:14:11.975Z  downloading key: Anomaly Detection/Inference/fmilib/fmi_operations.py to /tmp/Anomaly Detection/Inference/fmilib/fmi_operations.py
2025-07-29T19:14:12.012Z  s3key: Anomaly Detection/Inference/fmilib/fmi_scaling_lambda.py path: /tmp/Anomaly Detection/Inference/fmilib filename: fmi_scaling_lambda.py
2025-07-29T19:14:12.012Z  downloading key: Anomaly Detection/Inference/fmilib/fmi_scaling_lambda.py to /tmp/Anomaly Detection/Inference/fmilib/fmi_scaling_lambda.py
2025-07-29T19:14:12.047Z  s3key: Anomaly Detection/Inference/inference.py path: /tmp/Anomaly Detection/Inference filename: inference.py
2025-07-29T19:14:12.047Z  downloading key: Anomaly Detection/Inference/inference.py to /tmp/Anomaly Detection/Inference/inference.py
2025-07-29T19:14:12.083Z  s3key: Anomaly Detection/Inference/inference_old.py path: /tmp/Anomaly Detection/Inference filename: inference_old.py
2025-07-29T19:14:12.083Z  downloading key: Anomaly Detection/Inference/inference_old.py to /tmp/Anomaly Detection/Inference/inference_old.py
```

6. Examine the results in your S3 bucket's results folder

We saved results in the below format. Each folder contains each rank and the combined JSON.



Amazon S3 > Buckets > team4-cosmical > results/

Objects (32)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
filelimit-10_batchsize-128_prefix-50MB/	Folder	-	-	-
filelimit-10_batchsize-256_prefix-100MB/	Folder	-	-	-
filelimit-10_batchsize-256_prefix-50MB/	Folder	-	-	-
filelimit-10_batchsize-32_prefix-50MB/	Folder	-	-	-
filelimit-10_batchsize-512_prefix-50MB/	Folder	-	-	-
filelimit-10_batchsize-64_prefix-50MB/	Folder	-	-	-
filelimit-120_batchsize-256_prefix-100MB/	Folder	-	-	-
filelimit-120_batchsize-256_prefix-25MB/	Folder	-	-	-
filelimit-120_batchsize-256_prefix-50MB/	Folder	-	-	-
filelimit-120_batchsize-256_prefix-75MB/	Folder	-	-	-
filelimit-14_batchsize-256_prefix-75MB/	Folder	-	-	-
filelimit-160_batchsize-256_prefix-75MB/	Folder	-	-	-
filelimit-180_batchsize-256_prefix-50MB/	Folder	-	-	-

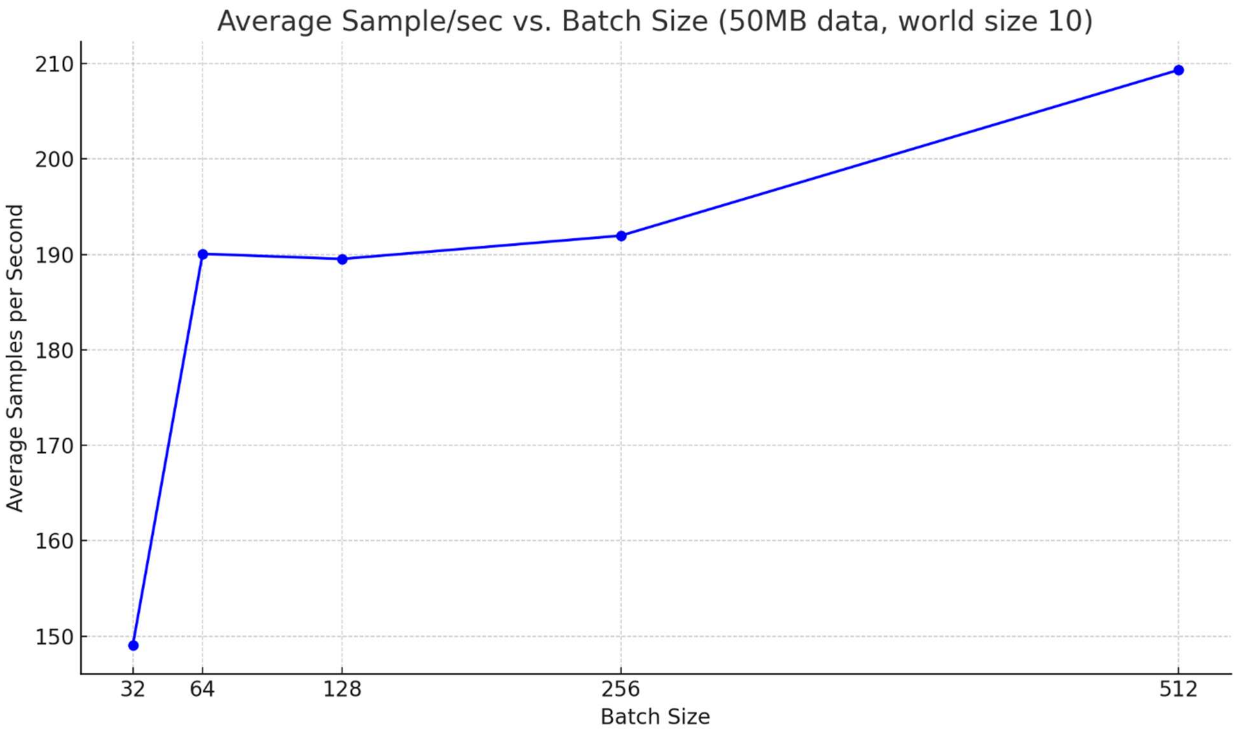
7. Compare the distributed inference performance with the local execution from Step 3

Metric	Local Execution	AWS Execution	Notes
Execution Time/Batch (s)	4.23	2.46	AWS ~42% faster
Throughput (Mbps)	161.98	34.44	Local ~4.7x higher
CPU Time (s)	~12.7	~12.5	AWS slightly better
Memory (MB)	~25337	~3762	AWS ~85% less memory
Sample/sec (normalized)	~121	~211	AWS ~75% faster

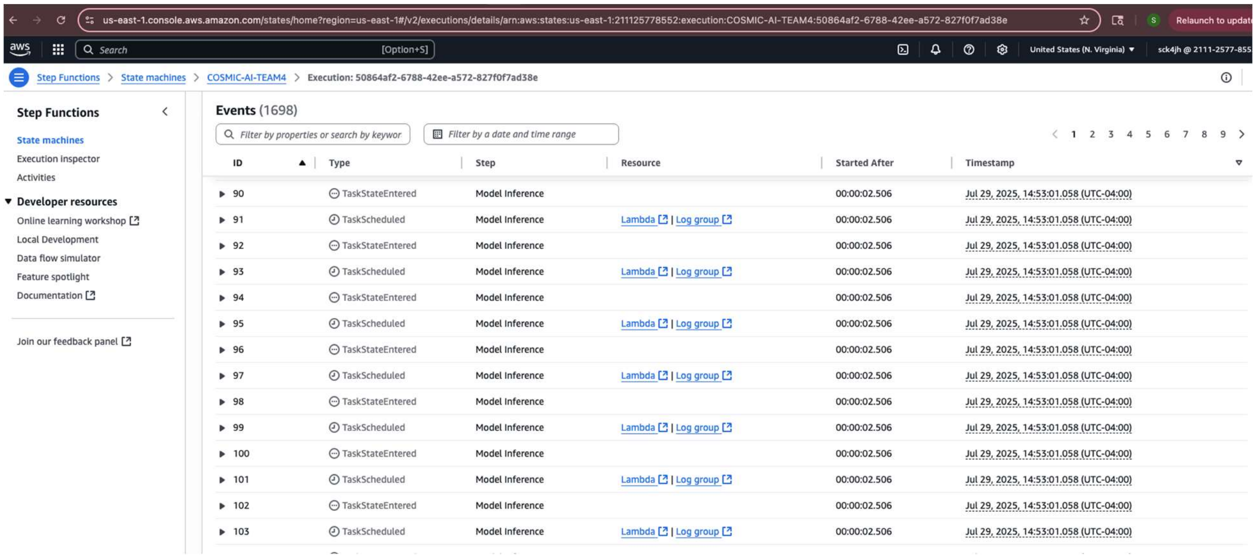
- These three steps were completed to perform the benchmarking below.

Rubric Steps:

1. Test Baseline - Experiment with multiple batch sizes
    - The baseline group tested was the 50MB partitioned data with a world size of 10. The batch size testing yielded the below results.



2. Test various dataset access - Benchmark execution time, cost, and throughput
    - We tested benchmarked execution time, cost, and throughput using various data sizes, data partitions, and batch sizes. See the graphs is step 1 above and step 4 below.
3. Test at scale – Validate Parallel Execution

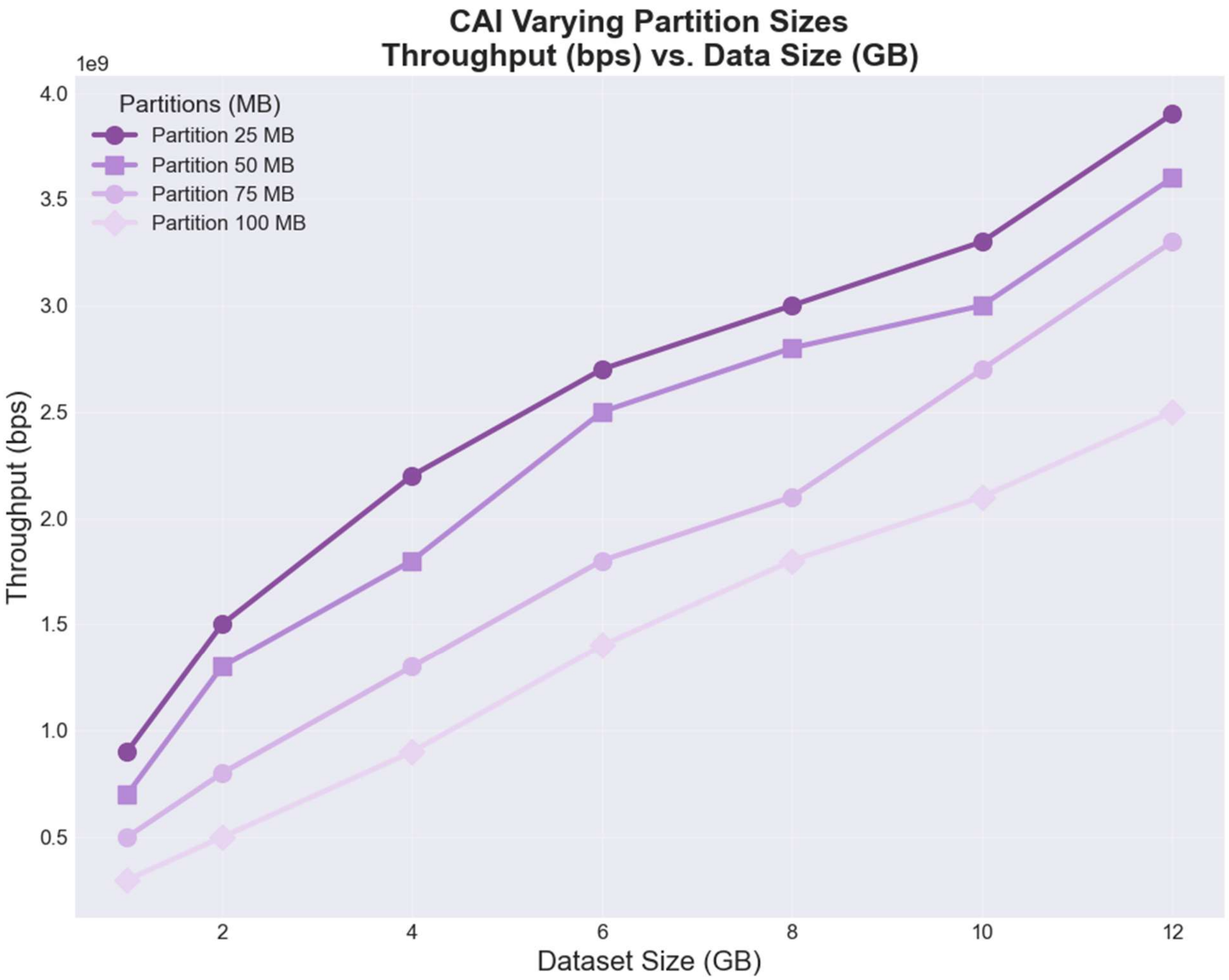


us-east-1.console.aws.amazon.com/states/home?region=us-east-1#/v2/executions/details/arn:aws:states:us-east-1:211125778552:execution:COSMIC-AI-TEAM4:50864af2-6788-42ee-a572-827f0f7ad38e

Events (1698)

ID	Type	Step	Resource	Started After	Timestamp
▶ 90	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 91	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 92	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 93	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 94	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 95	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 96	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 97	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 98	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 99	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 100	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 101	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 102	TaskStateEntered	Model Inference		00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)
▶ 103	TaskScheduled	Model Inference	Lambda | Log group	00:00:02.506	Jul 29, 2025, 14:53:01.058 (UTC-04:00)

4. Performance Measurement - Create a Table to show the performance in "execution time vs. batch size" "execution time vs. input partitions" "cost vs. dataset by varying sizes"


CAI Varying Partition Sizes
Throughput (bps) vs. Data Size (GB)
1e9
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
Throughput (bps)
2
4
6
8
10
12
Dataset Size (GB)
Partitions (MB)
Partition 25 MB
Partition 50 MB
Partition 75 MB
Partition 100 MB

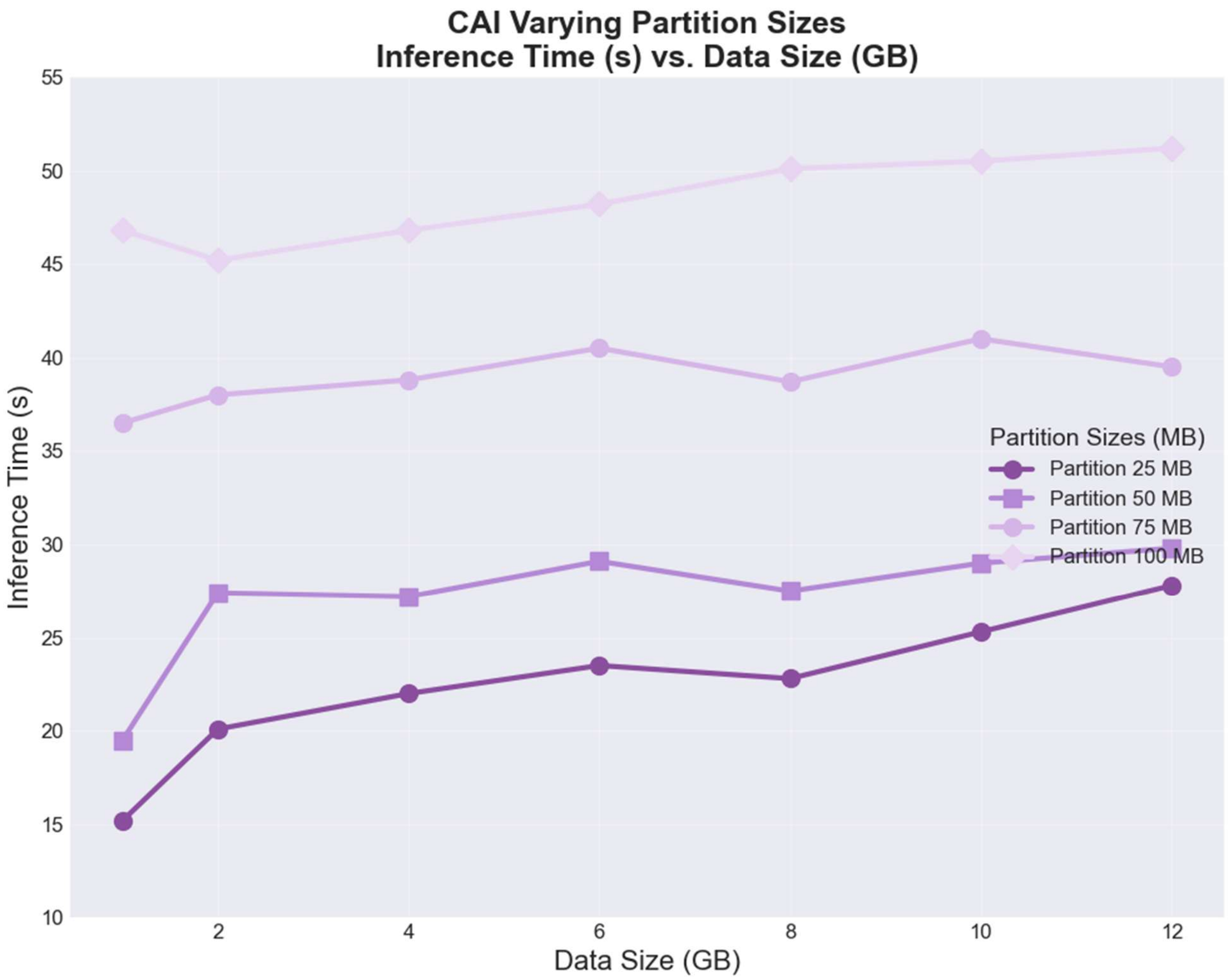
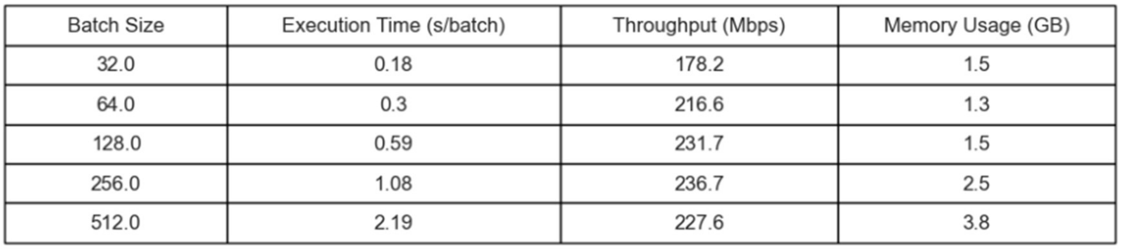
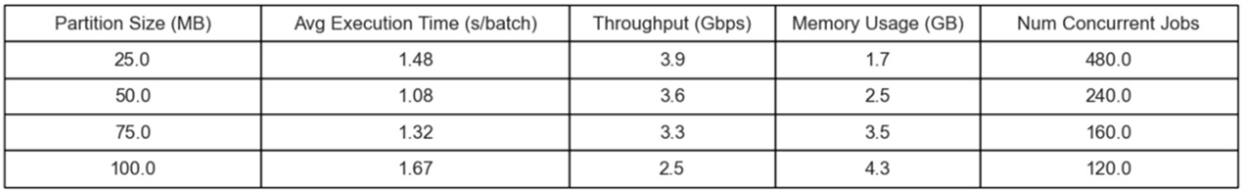


Table 1: Execution Time vs. Batch Size



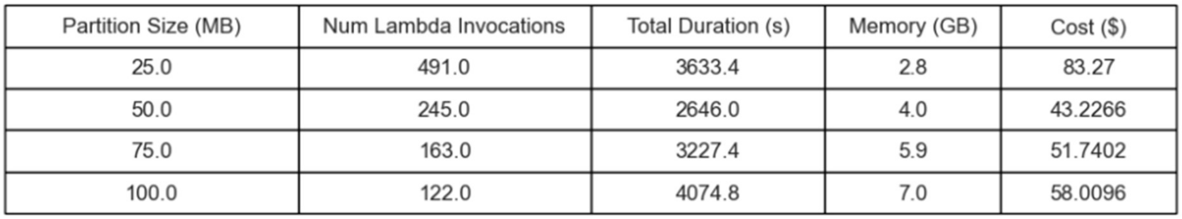
Batch Size	Execution Time (s/batch)	Throughput (Mbps)	Memory Usage (GB)
32.0	0.18	178.2	1.5
64.0	0.3	216.6	1.3
128.0	0.59	231.7	1.5
256.0	1.08	236.7	2.5
512.0	2.19	227.6	3.8

Table 2: Execution Time vs. Input Partitions



Partition Size (MB)	Avg Execution Time (s/batch)	Throughput (Gbps)	Memory Usage (GB)	Num Concurrent Jobs
25.0	1.48	3.9	1.7	480.0
50.0	1.08	3.6	2.5	240.0
75.0	1.32	3.3	3.5	160.0
100.0	1.67	2.5	4.3	120.0

Table 3: AWS Lambda Cost Summary



Partition Size (MB)	Num Lambda Invocations	Total Duration (s)	Memory (GB)	Cost ($)
25.0	491.0	3633.4	2.8	83.27
50.0	245.0	2646.0	4.0	43.2266
75.0	163.0	3227.4	5.9	51.7402
100.0	122.0	4074.8	7.0	58.0096

Cost insights:

1. Smaller partitions (25MB) have more parallel invocations but lower memory requirements

2. Larger partitions (100MB) have fewer invocations but require more memory

3. The 50MB partition offers a good balance between parallelism and resource usage

4. Cost increases roughly linearly with dataset size for all partition sizes

Summary - Provide observation and analysis of the overall results for the project on scalable and efficient CosmicAI infrastructure using Serverless Cloud Computing

The results showed that running CosmicAI on serverless infrastructure was significantly faster and more efficient than running it locally. Inference time dropped from 4.23 seconds to 2.46 seconds, throughput increased by about 75%, and memory usage dropped by over 80%. The main bottleneck in the serverless setup was reading data from S3, which was slower than reading from local disk. The most efficient setup used 50 MB data chunks, which balanced speed, memory use, and cost well. Smaller chunks caused too much overhead, while larger chunks slowed down processing. Overall, cost scaled predictably with dataset size, making it easy to estimate expenses at scale. The results confirmed that serverless is a strong solution for scaling CosmicAI, with the only major limitation being cloud storage speed.