IMAGE CLASSIFICATION USING AWS SAGEMAKER AND S3

TEAM CLOUD ALLIANCE

Jagadish Arepalli
College of Natural Science & Mathematics
Department of computer science
University of Houston
Houston, Texas
jarepall@cougarnet.uh.edu

Vamshi Teja Yellenki
College of Natural Science & Mathematics
Department of computer science
University of Houston
Houston, Texas
vyellenk@cougarnet.uh.edu

Yaswanth Chowdary Pavuluri
College of Natural Science & Mathematics
Department of computer science
University of Houston
Houston, Texas
ypavulur@cougarnet.uh.edu

Under the guidance of - Dr. Weidong Shi
Rabimba Karanjai
Nour Diallo

PROPOSED ABSTRACT

Identifying natural scenes from all around the world is an interesting computer vision problem. In this project, we are going to classify six different categories of images (buildings, forest, glacier, mountain, sea, street). A simple Machine learning model is going to classify the images in the dataset. We are going to do performance analysis by deploying the same machine learning algorithm on various EC2 instances and compare their performances with respect to the training time and s3 upload time.

IMPROVEMENTS PROPOSED

1. Use EC2 for comparison of the performance of the image classification algorithm.
2. Use Lambda for inference from the trained model and benchmark the results.

ABSTRACT

In this project, we are going to implement various concepts of Amazon Web Services to build a model classifying six different categories of images buildings, forest, glacier, mountain, sea and street. A simple architecture of our project is to develop an image classification algorithm and compare its performance in different EC2 instances with respect to the training time and s3 upload time along with transfer learning on the trained model. The goal is to build a machine learning model with Amazon SageMaker and deploy it to a sagemaker endpoint and perform transfer learning using the intel-image-classification dataset, and expose the inference engine via an API. By using Amazon SageMaker, we can easily build and train machine learning models, and then directly deploy them into a production-ready hosted environment. It provides common machine learning algorithms that are optimized to run efficiently against extremely large data in a distributed environment. Amazon SageMaker and EC2 instances offer flexible distributed training options to deploy and analyze the model in secure and scalable environment. To meet the reliability and scaling needs, we are using Amazon Simple Storage Service(S3). For deploying and testing our results in real-time use case, we utilise the Lambda and API Gateway resources.

INTRODUCTION

Software systems that learn from data are being deployed in increasing numbers in industrial application scenarios. Managing these machine learning (ML) systems and the models which they apply imposes additional challenges beyond those of traditional software systems. Machine learning was once out of the reach of most enterprise budgets, but today, public cloud providers ability to offer machine- learning services makes this technology affordable.

A select set of applications requires a storage technology that is flexible enough to let application designers configure their data store appropriately based on these tradeoffs to achieve high availability and guaranteed performance in the most cost effective manner. AWS SageMaker is one such cloud machine learning SDK designed for speed of iteration, and it's one of the fastest- growing toys in the Amazon AWS Ecosystem.

Machine learning algorithm included in sagemaker instance is based on the concept of inference as a model. Since, data is fed through the model in both the training and testing phases; they may appear to be comparable. In contrast, a model will iteratively reduce the loss function while training by processing the data numerous times. Results are frequently compared to outputs while evaluating and reevaluating them.

SageMaker

Amazon SageMaker is a fully managed machine learning service. SageMaker allows to create, train and deploy machine learning models in cloud. Here in this project we use SageMaker notebook instances to deploy our code and train a machine learning model on our dataset.

S3 Bucket

An Amazon S3 bucket is a public cloud storage resource available in Amazon Web Services (AWS) Simple Storage Service (S3), an object storage offering. Objects that are stored in Amazon S3 buckets which resemble file folders are made up of data and the metadata. The user then chooses an S3 tier for the data after creating the bucket, with each tier having a different level of redundancy, cost, and accessibility. Different S3 storage tiers of objects can be stored in a same bucket. By using tools like the AWS IAM roles (Identity and Access Management service), bucket rules, and access control lists, the user can define access privileges for the objects kept in a bucket.

Lambda Function

One of the simple solutions for serverless and event-driven computing is AWS's Lambda. The automatic execution of codes without any need for servers or clusters is very simplified. Simply put, without having to worry about installing or managing infrastructure, codes may be uploaded and immediately run. Therefore, regardless of their size, "code execution requests" are automatically accepted by this service. Additionally, you can only pay for computed time with AWS Lambda, which makes it an efficient cost-control tool.

Identity and Access Management

IAM helps us to specify who and what can access resources in AWS. With the help of IAM role we can restrict users with the level of access that they have and the resources that they can access. We can also restrict the communication between different services by creating a role and attaching policies to it. For example in this project we have given the role AWSLambdaExecute such that the lambda function has access to execute the deployed sagemaker endpoint.

Dataset

The intel-image-classification dataset that will be used for anlaysis and training the image classification model. This dataset consists of different types of images that we see around us. This dataset has around 25 thousand images of size 150x150 distributed under six categories. Each of the category has a specific label 'buildings' - 0, 'forest' - 1, 'glacier' - 2, 'mountain' - 3, 'sea' - 4, 'street' - 5 .

Lambda Layers

A Lambda Layer is same as a directory containing a library in a function code. The difference is that, instead of creating a package to this library within the function code, it can be packaged separately. Lambda will build a layer together with the function when it's triggered with endpoint.

Endpoint

In the next step, we use trained model artifacts to make real-time inferences or batch predictions. An Amazon SageMaker endpoint is a fully managed service that enables you to draw conclusions in real-time using a REST API. It removes the hassle of managing your own EC2 instances, loading S3 artifacts, encasing the model in REST application and connecting.

API Gateway

AWS API Gateway is a fully managed service for creating, monitoring, and securing APIs at scale. It serves as a "front door" for backend service-using REST and Web socket applications and manages all the operations required to accept and process up to hundreds of thousands of API calls concurrently, including traffic management, authorization and access control, monitoring, and API version management. You only pay for the API calls you make and the data that is sent out using API Gateway, which is affordable and does not have any minimum fees. REST, HTTP, and Web socket APIs can be created, published, maintained, monitored, and secured at any scale with Amazon API Gateway. In order to construct APIs that access AWS and data kept in the AWS Cloud, API Gateway is used. API Gateway creates RESTful APIs that:

- Are HTTP-based.
- Enable stateless client-server communication.
- Implement standard HTTP methods such as GET POST, PUT and DELETE.

Postman

Postman supports API-first development with the API Builder. Use the API Builder to design your API in Postman. Your API definition can then act as the single source of truth for your API project. You can connect various elements of your API development and testing process to your API definition, such as collections, documentation, tests, and monitors.and scalability of the software systems.

PROCEDURE

Overall Architecture:

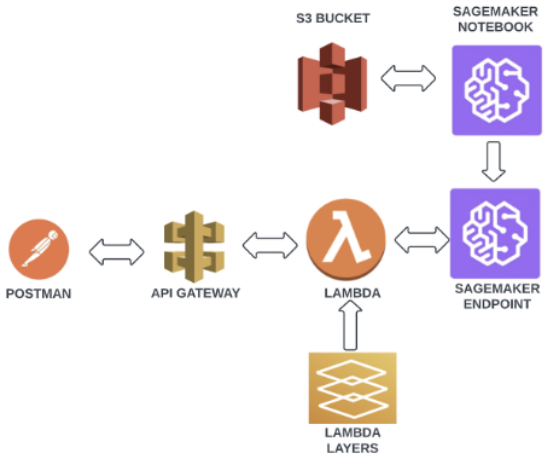

Figure 1: Project Architecture

S3 Upload:

We will upload the Intel image classification dataset into the Amazon Simple Storage Services(S3) buckets which is designed to make web scale computing easier and store and retrieve any amount of data at anytime from anywhere on the web.

We can implement algorithm using Amazon SageMaker. In machine learning, you "teach" a computer to make predictions, or inferences. First, you use an algorithm and example data to train a model. Then you integrate your model into your application to generate inferences in real time and at scale. In a production environment, a model typically learns from millions of example data items and produces inferences in hundreds to less than 20 milliseconds.

In Amazon SageMaker, we preprocess data in a Jupyter notebook on your notebook instance. We use notebook to fetch the dataset, explore it, and prepare it for model training.

Model Training

Model training includes both training and evaluating the model, as follows:

Training:

To train a model, you need an algorithm. We can use the previously specified algorithms or one of the algorithms that Amazon SageMaker provides. Depending on the size of your training dataset and how quickly you need the results, you can use resources ranging from a single general-purpose instance to a distributed cluster of GPU instances.

Evaluating:

After we have trained the model, evaluate it to determine whether the accuracy of the inferences is acceptable. In Amazon SageMaker, we use either the AWS SDK for Python (Boto) or the high-level Python library that Amazon SageMaker provides to send requests to the model for inferences. We can also use a Jupyter notebook in our Amazon SageMaker notebook instance to train and evaluate the model.

Validating the model

After training a model, evaluate the trained model to determine its performance and accuracy. We can generate multiple models using different methods and evaluate each of it. For example, you could apply different business rules for each model, and then apply various Amazon SageMaker Developer Guide Validating Models measures to determine each model's suitability.

Model Deploying

We traditionally re-engineer a model before integrating it with the application and deploy it. With Amazon SageMaker hosting services, we can deploy the model independently, decoupling it from the application code. In this project we deploy the model to a sagemaker endpoint and expose it to an API.

Testing the model

By creating a POST endpoint we can use it to trigger the lambda function which in turn would call the model endpoint with the parameters passed and give us the prediction. Here in this project we are passing the input as a JSON with key - 'url' and value is the image url.

Performance Evaluation

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. EC2 is designed to make web-scale cloud computing easier with desirable configurations.

The objective of our study is to evaluate the performance in different EC2 instances which are created during setup of notebook instance in sagemaker. We are analyzing the performance of 3 CPU instances which are mentioned below-

- ml.m5.xlarge
- ml.m5.4xlarge
- ml.c5.4xlarge

M5 Instances:

- M5 instances offer a balance of compute, memory, and networking resources for a broad range of workloads.
- Used mainly in web and application servers, small and mid-sized databases cluster computing, gaming servers, caching fleets, and app development environments.

C5 Instances:

- Amazon EC2 C5 instances deliver cost-effective high performance at a low price per compute ratio for running advanced compute-intensive workloads.
- Used mainly in high-performance web servers, high-performance computing (HPC), highly scalable multiplayer gaming, ad serving, scientific modeling, distributed analytics and machine/deep learning inference.

By deploying our machine learning algorithm in the above instances we could see a significant change in the time taken to upload the data to s3 bucket and train the model.

By using ml.m5.xlarge instance we could reduce the execution time by 18.7% when compared with PC and by replacing the above instance with a better performing instance like ml.m5.4xlarge we could further reduce the execution time by 64.8% as compared to using PC and 56.7% as compared to ml.m5.xlarge instance.Further replacing the above instances with compute optimized instance ml.c5.4xlarge reduced the training time by 74.6% as compared to PC and 27% reduce as compared to ml.m5.4xlarge.

After a thorough examination of all the data obtained post experimentation, we conclude that C5 instance is a better choice to train a model that predicts image classifying it to the definite category using the compute optimized instance on AWS that helps to train and execute the model faster.

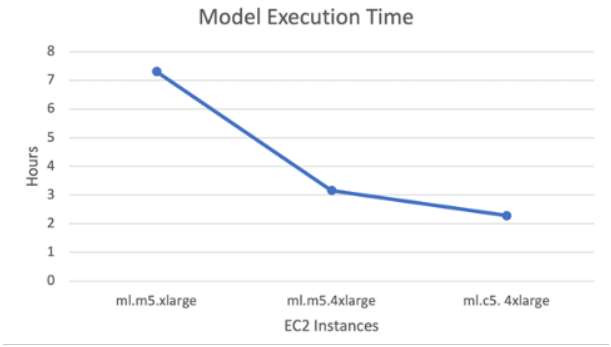

Figure 2: Model Execution Time

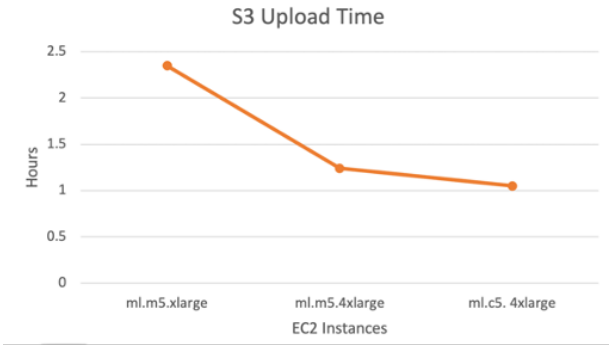

Figure 3: S3 Upload Time

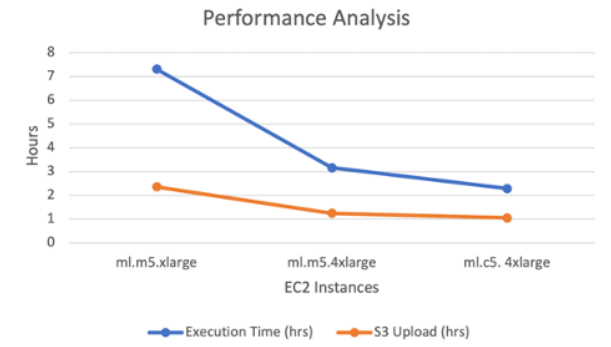

Figure 4: Combined Chart of S3 upload time and Execution time

EXPERIMENTS AND RESULTS

After deploying the changes in production environment, we tested the classification algorithm using Postman and Curl as our execution stages.

Postman is an API platform that allows developers to design, build and test their API's. Here in this project we are using Postman to test the API that we created using the AWS API Gateway service. This helps us to call our API by passing the input parameters in the raw data.

cURL is a command line tool for transfering data. The name stands for 'Client URL'. In this project we used cURL to call our API, this served as an alternative to postman while testing our predictions done by the sagemaker endpoint.

- In Postman we provide the invoke url by selecting the post method and pass image url as parameter in raw data of POST method.

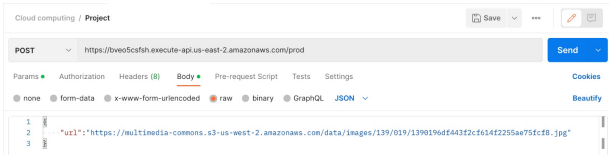

Figure 5: Invoke url in postman

- cURL request is used to invoke the API using command line. The default curl request is a GET, hence we do -X POST to make it a POST request. To send the input data as a parameter we use -d.

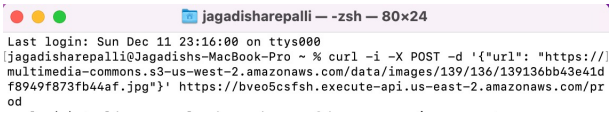

Figure 6: cURL request in command line

We have tested the model using five different images building, forest, sea, mountain and glacier.

Following are the figures displaying input and outputs using Postman and cURL.

Prediction for Building image:

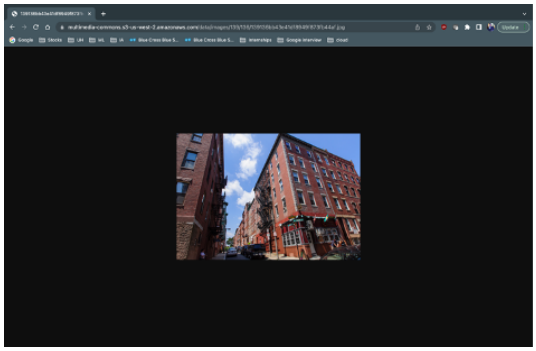

Figure 7: Input image given as building

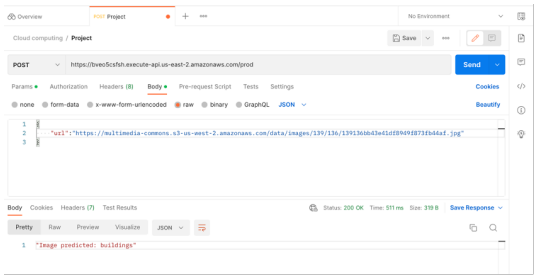

Figure 8: Postman Output predicted as building

```
jagadisharepalli@Jagadishs-MacBook-Pro ~ % curl -i -X POST -d '{"url": "https://
multimedia-commons.s3-us-west-2.amazonaws.com/data/images/139/136/139136bb43e41d
f8949f873fb44af.jpg"}' https://bveo5csfsh.execute-api.us-east-2.amazonaws.com/pr
od
HTTP/2 200
date: Tue, 06 Dec 2022 03:48:22 GMT
content-type: application/json
content-length: 28
x-amzn-requestid: 20079e8c-384f-4a18-8553-1177e08e9fd3
x-amz-apigw-id: ctI8mGV7CYcFv6Q=
x-amzn-trace-id: Root=1-638ebb83-7b27e2b725cca0d26736bb67;Sampled=0

["Image predicted: buildings"]
```

Figure 9: cURL Output predicted as building

Prediction for Forest image:

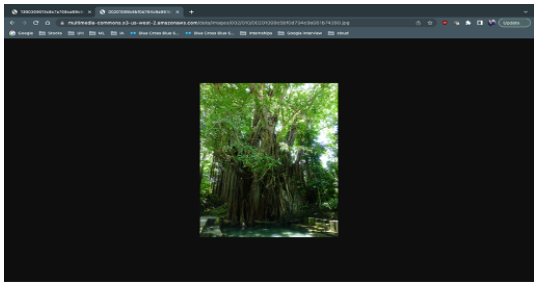

Figure 10: Input image given as forest

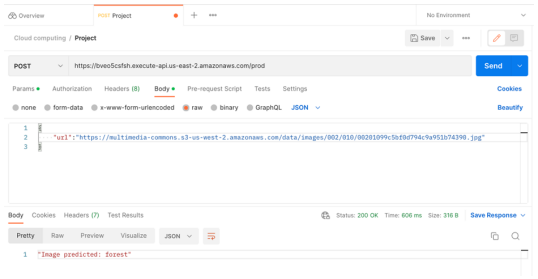

Figure 11: Postman Output predicted as forest

```
jagadisharepalli@Jagadishs-MacBook-Pro ~ % curl -i -X POST -d '{"url": "https://
multimedia-commons.s3-us-west-2.amazonaws.com/data/images/002/010/00201099c5bf0d
794c9a951b74390.jpg"}' https://bveo5csfsh.execute-api.us-east-2.amazonaws.com/pr
od
HTTP/2 200
date: Tue, 06 Dec 2022 04:14:55 GMT
content-type: application/json
content-length: 25
x-amzn-requestid: 19712f44-d15d-46f4-8b79-32fc781776fc
x-amz-apigw-id: ctM1zE1QiYcF4Pg=
x-amzn-trace-id: Root=1-638ec1be-574d79220d0d20df18104707;Sampled=0

"Image predicted: forest"%
```

Figure 12: cURL Output predicted as forest

Prediction for Sea image:

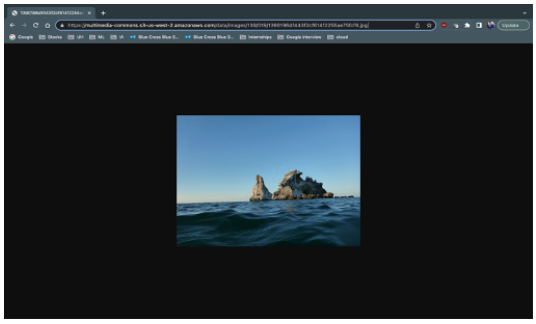

Figure 13: Input image given as sea

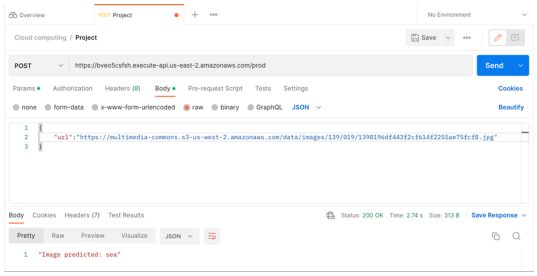

Figure 14: Postman Output predicted as sea

```
jagadisharepalli@Jagadishs-MacBook-Pro ~ % curl -i -X POST -d '{"url": "https://
multimedia-commons.s3-us-west-2.amazonaws.com/data/images/139/019/1390196df443f2
cf614f2255ae75fcf8.jpg"}' https://bveo5csfsh.execute-api.us-east-2.amazonaws.com
/prod
HTTP/2 200
date: Tue, 06 Dec 2022 03:51:04 GMT
content-type: application/json
content-length: 22
x-amzn-requestid: fb506944-2108-4e97-ac36-8fda4ceb9a02
x-amz-apigw-id: ctJWQHXvCYcFr-w=
x-amzn-trace-id: Root=1-638ebc27-61071d1c1d6c4e7052e26b48;Sampled=0

"Image predicted: sea"%
```

Figure 15: cURL Output predicted as sea

Prediction for Mountain image:

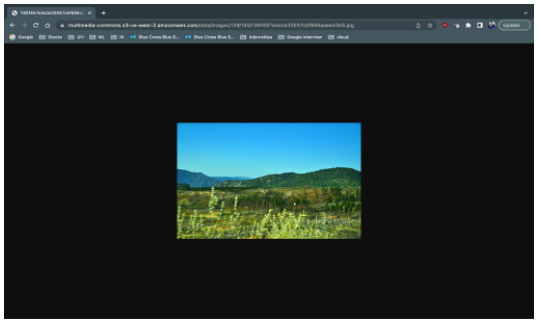

Figure 16: Input image given as mountain

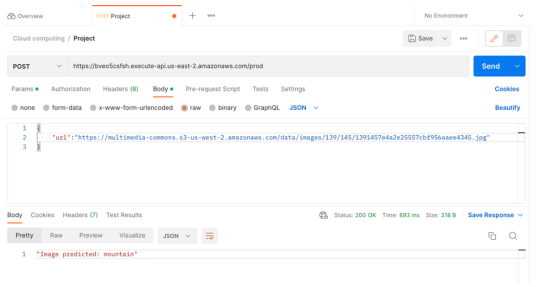

Figure 17: Postman Output predicted as mountain

```
jagadisharepalli@Jagadishs-MacBook-Pro ~ % curl -i -X POST -d '{"url": "https://
multimedia-commons.s3-us-west-2.amazonaws.com/data/images/139/145/1391457e4a2e25
557cbf956aaee4345.jpg"}' https://bveo5csfsh.execute-api.us-east-2.amazonaws.com/
prod
HTTP/2 200
date: Tue, 06 Dec 2022 04:16:58 GMT
content-type: application/json
content-length: 27
x-amzn-requestid: 6642fbc4-9c3f-46a7-904b-0d6e383c363e
x-amz-apigw-id: ctNJEG5MiYcFznA=
x-amzn-trace-id: Root=1-638ec239-686c1f547d8cf73d6688282d;Sampled=0

"Image predicted: mountain"%
```

Figure 18: cURL Output predicted as mountain

Prediction for Glacier image:

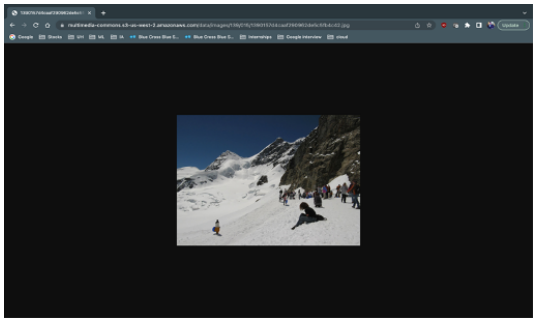

Figure 19: Input image given as glacier

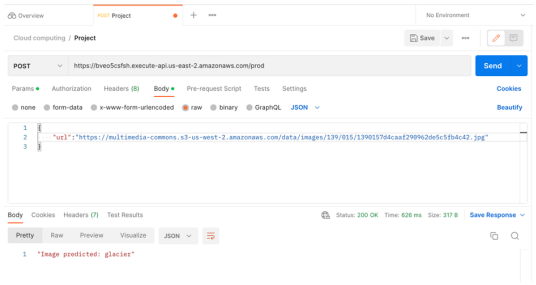

Figure 20: Postman Output predicted as glacier

```
jagadisharepalli@Jagadishs-MacBook-Pro ~ % curl -i -X POST -d '{"url": "https://
multimedia-commons.s3-us-west-2.amazonaws.com/data/images/139/015/1390157d4caaf2
90962de5c5fb4c42.jpg"}' https://bveo5csfsh.execute-api.us-east-2.amazonaws.com/p
rod
HTTP/2 200
date: Tue, 06 Dec 2022 04:12:41 GMT
content-type: application/json
content-length: 26
x-amzn-requestid: f66b752c-ae7d-41e8-9f1e-21c6a666c539
x-amz-apigw-id: ctMgnHEGCYcFm9A=
x-amzn-trace-id: Root=1-638ec136-1a5c02b42ad7dad174a396b1;Sampled=0

"Image predicted: glacier"%
```

Figure 21: cURL Output predicted as glacier

TEAM MEMBER AND CONTRIBUTIONS

JAGADISH AREPALLI

- Refer different types of datasets required for project.
- Creation of the inference model by integrating the endpoint with lambda and API Gateway.

VAMSHI TEJA YELLENKI

- Create and integrate the build module for training the data using different instances in EC2.
- Validate and compare the necessary performance results in different instances.

YASWANTH CHOWDARY PAVULURI

- Search and analyze the various CNN models that can be used for project development.
- Create the required responses for data input and perform necessary validation.

References

[1] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks[J] Advances in Neural Information Processing Systems, 2012, 25(2):2012.

[2] Tianmei Guo, Jiwen Dong ,Henjian Li Yunxing Gao Simple Convolutional Neural Network on Image Classification. 2017 IEEE 2nd International Conference on Big Data Analysis.

[3] Source code and development

- https://aws.amazon.com/getting-started/hands-on/build-train-deploy-machine-learning-model-sagemaker/
- https://github.com/aws-samples/amazon-sagemaker-visual-transformer
- https://aws.amazon.com/blogs/machine-learning/machine-learning-inference-at-scale-using-aws-serverless/
- https://medium.com/thecyphy/train-cnn-model-with-pytorch-21dafb918f48

[4] Dataset

- https://www.kaggle.com/datasets/puneet6060/intel-image-classification

[5] Lambda

- https://www.youtube.com/watch?v=xe9-GZ1tX28t=4s
- https://medium.com/analytics-vidhya/invoke-an-amazon-sagemaker-endpoint-using-aws-lambda-83ff1a9f5443

[6] API Gateway

- https://medium.com/@xiaolancara/using-aws-api-gateway-to-build-an-end-to-end-http-api-efd8fbd917c6
- https://www.youtube.com/watch?v=vgl9q5Ox5LEt=720s