

# 1. introduction

In this assignment, we will deploy a cluster on the Google Kubernetes Engine(GKE) to do the training job and inference job for a CNN model.

## 2. prerequisites

### 2.1 Prepare the images

In order to deploy the cluster and enable the inference service, we need to build the images and push the images into docker hub, so that we can easily use the images to create containers in K8S. Here attached the structure of total repo.

```
.
├── inference
│   ├── Dockerfile
│   ├── inference.py
│   └── requirements.txt
├── inference.yaml
├── pvc.yaml
├── service.yaml
├── train
│   ├── Dockerfile
│   └── train.py
└── train.yaml
```

After we finished the code writing, we can use docker to build the image and push it to docker hub.

```
docker login -u <username>
```

```
docker build -t inferencejob:1.0 inference/
```

```
docker build -t trainjob:1.0 train/
```

```
docker tag inferencejob:1.0 yp2141/inferencejob:1.0
```

```
docker tag trainjob:1.0 yp2141/trainjob:1.0
```

```
docker push yp2141/inferencejob:1.0
```

```
docker push yp2141/trainjob:1.0
```

### 2.2 Create a cluster on GKE

First, we need to create a cluster on google cloud, and after we successfully created the cluster, we can see a cluster in **OVERVIEW** part.

Google Cloud nyu-cloudML2024 Search (/) for resources, docs, products, and more

Kubernetes Engine Kubernetes clusters **CREATE** **DEPLOY** **REFRESH** **ONBOARDING** **LEARN**

**Eight steps to set up GKE**  
Now it's even easier to set up GKE. Learn the best practices and Google recommendations on how to run production-grade GKE clusters. **START**

**Run your business critical workloads faster, safer, and easier at enterprise scale**  
GKE Enterprise combines multi-cluster and multi-team operations with fully managed security, governance, and service networking components. Enjoy all the benefits of GKE Standard along with the tools that secure workloads, enforce compliance policies, and provide application visibility with actionable insights and an application-aware network for resiliency.  
When you're ready to scale beyond a single team or cluster, GKE Enterprise delivers an integrated and consistent way to configure, secure, protect, and monitor container workloads. **LEARN AND ENABLE**

**OVERVIEW** OBSERVABILITY COST OPTIMIZATION

**Filter** Enter property name or value

<input type="checkbox"/> Status	Name ↑	Location	Tier ?	Number of nodes	Total vCPUs	Total memory	Notifications	Labels
<input checked="" type="checkbox"/>	autopilot-cluster-1	us-central1	Standard		0	0 GB		—

Then clicking the cluster and entering into **Clusters** page, we can connect to it under cloud shell.

Kubernetes Engine Clusters **DELETE** **DEPLOY** **CONNECT** **DUPLICATE**

**Connect to the cluster**

You can connect to your cluster via command-line or using a dashboard.

**Command-line access**  
Configure [kubectl](#) command line access by running the following command:

```
$ gcloud container clusters get-credentials autopilot-cluster-1 --region us-central1 --project nyu-cloudml2024
```

**RUN IN CLOUD SHELL**

**Cloud Console dashboard**  
You can view the workloads running in your cluster in the Cloud Console [Workloads dashboard](#).

**OPEN WORKLOADS DASHBOARD**

**OK**

Then, use `git clone` to clone the repository and deploy the pods

```
Welcome to Cloud Shell! Type "help" to get started.
Your Cloud Platform project in this session is set to nyu-cloudml2024.
Use "gcloud config set project [PROJECT_ID]" to change to a different project.
yp2141@cloudshell:~ (nyu-cloudml2024) $ gcloud container clusters get-credentials autopilot-cluster-1 --region us-central1 --project nyu-cloudml2024
Fetching cluster endpoint and auth data.
kubeconfig entry generated for autopilot-cluster-1.
yp2141@cloudshell:~ (nyu-cloudml2024) $ git clone https://github.com/yipeng0016/mnistK8S.git
Cloning into 'mnistK8S'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (13/13), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 13 (delta 0), reused 13 (delta 0), pack-reused 0 (from 0)
yp2141@cloudshell:~ (nyu-cloudml2024) $ ls
main.tf  mnistK8S  README-cloudshell.txt  terraform.tfstate  terraform.tfstate.backup
```

## 2.3 Deploy service on the cluster

First, we need to use the command `kubectl apply -f pvc.yaml` to deploy the persistence volume claim. It should be noted that pvc will remain in the pending state until an application uses it, at which time the state of pvc will change from pending to bounded.

Kubernetes Engine

Learn about Enterprise

All Fleets

Resource Management

Overview

Clusters

Workloads

Teams

Applications

Secrets & ConfigMaps

Storage

Object Browser

Rollout Sequencing

Backup for GKE

Storage

Cluster

Namespace

RESET

SAVE

PERSISTENT VOLUME CLAIMS

STORAGE CLASSES

Persistent volume claims are requests for storage of specific size and access mode.

Learn more

Filter

Filter persistent volume claims

	Name	Phase	Volume	Storage class	Namespace	Cluster
	pvc	Bound	pvc-df1327a6-1464-41db-b279-ea9f71f833d3	standard-rwo	default	autopilot-cluster-1

Then, we need to apply the training, inference and service.yaml.

```
kubectl apply -f training.yaml
kubectl apply -f inference.yaml
kubectl apply -f service.yaml
```

Kubernetes Engine

Learn about Enterprise

All Fleets

Resource Management

Overview

Clusters

Workloads

Teams

Applications

Secrets & ConfigMaps

Storage

Object Browser

Rollout Sequencing

Backup for GKE

Posture Management

Marketplace

Release Notes

Deployment details

REFRESH

EDIT

DELETE

ACTIONS

KUBECTL

SHOW INFO PANEL

UTC-5

8:00 PM

0

UTC-5

8:00 PM

0

UTC-5

8:00 PM

0

Cluster

autopilot-cluster-1

Namespace

default

Labels

app: serviceapi

tier: backend

Logs

Container logs, Audit logs

Replicas

1 updated, 1 ready, 1 available, 0 unavailable

Pod specification

Revision 1, containers: inferencejob, volumes: pvc

Horizontal Pod

Not configured

Autoscaler

Not configured

Vertical Pod Autoscaler

Not configured

Active revisions

Revision	Name	Status	Summary	Created on	Pods running/Pods total
1	inferencejob-644bf6cc44	OK	inferencejob: yp2141/inferencejob:5.0	Dec 1, 2024, 5:18:00 PM	1/1

Managed pods

Revision	Name	Status	Restarts	Created on
1	inferencejob-644bf6cc44-wbzvg	Running	0	Dec 1, 2024, 5:18:00 PM

Exposing services

Name	Type	Endpoints
inferenceclb	Load balancer	34.134.245.100:8000

### 3. Access the inference service

After the deployment, we can use the exposed ip to access the service. In this project, I provided a restful api and you can use curl to access to it.

```
curl -X POST -F "file=@test.png" http://34.134.245.100:8000/predict
```

The final github repo: <https://github.com/yipeng0016/mnistK8S>

## Reference

The most scalable and fully automated Kubernetes service (<https://cloud.google.com/kubernetes-engine?hl=en>)

Pytorch MNIST example (<https://github.com/pytorch/examples/tree/main/mnist>)

Deploy an app to a GKE cluster (<https://cloud.google.com/kubernetes-engine/docs/deploy-app-cluster>)