**PyTorch fserving (flask serving)**

1. **Creating PV**

* **PyTorch-training-V1.0.docx 참조**

1. **Creating PVC**

* **PyTorch-training-V1.0.docx 참조**

1. **Build Image**

Pytorch serving POD Example에서 실행될 Docker Image는 손글씨 인식 모델인 MNIST 모델을 사용한다.

제작된 Docker Image는 Docker Hub에 Push되어야 한다.

Docker image 제작을 위한 명령어는 다음과 같다.

1. Docker Image를 Build한다.

**sudo docker build -f Dockerfile\_fserving -t pytorchjob-fserving-mnist .**

1. Docker Image를 Tagging한다.

**sudo docker tag pytorchjob-fserving-mnist joleedocker/pytorchjob-fserving-mnist:1.0**

1. Docker Image를 Docker Hub에 Push한다.

**sudo docker push joleedocker/pytorchjob-fserving-mnist:1.0**

1. 생성된 Docker image를 확인한다.

**sudo docker images**

Docker image 제작을 위한 Dockerfile은 다음과 같다.

## ****Dockerfile\_fserving****

|  |
| --- |
| FROM pytorch/pytorch:1.0-cuda10.0-cudnn7-runtime  RUN apt update  RUN apt install software-properties-common -y  RUN add-apt-repository ppa:deadsnakes/ppa -y  RUN apt install python3.7 --version -y  RUN python -m pip install --upgrade pip  RUN pip install tensorboardX==1.6.0  RUN pip install flask  RUN pip install pandas  RUN mkdir -p /opt/mnist  RUN mkdir -p /opt/mnist/models  WORKDIR /opt/mnist/src  ADD mnist\_fserving.py /opt/mnist/src/mnist\_fserving.py  RUN chgrp -R 0 /opt/mnist \  && chmod -R g+rwX /opt/mnist  EXPOSE 80  ENTRYPOINT ["python3", "/opt/mnist/src/mnist\_fserving.py"] |

## ****mnist\_fserving.py****

|  |
| --- |
| from \_\_future\_\_ import print\_function  from flask import Flask, request, jsonify  import json  import csv  import pandas as pd  from datetime import datetime, timedelta  import argparse  import os  #from tensorboardX import SummaryWriter  from torchvision import datasets, transforms  import torch  import torch.distributed as dist  import torch.nn as nn  import torch.nn.functional as F  import torch.optim as optim  from torch import tensor  WORLD\_SIZE = int(os.environ.get('WORLD\_SIZE', 1))  app = Flask(\_\_name\_\_)  global\_variable = 111  class Net(nn.Module):  def \_\_init\_\_(self):  super(Net, self).\_\_init\_\_()  self.conv1 = nn.Conv2d(1, 20, 5, 1)  self.conv2 = nn.Conv2d(20, 50, 5, 1)  self.fc1 = nn.Linear(4\*4\*50, 500)  self.fc2 = nn.Linear(500, 10)  def forward(self, x):  x = F.relu(self.conv1(x))  x = F.max\_pool2d(x, 2, 2)  x = F.relu(self.conv2(x))  x = F.max\_pool2d(x, 2, 2)  x = x.view(-1, 4\*4\*50)  x = F.relu(self.fc1(x))  x = self.fc2(x)  return F.log\_softmax(x, dim=1)  MODEL = Net()  def load\_model(model\_name):  model\_export\_dir = "/opt/mnist/models"  global MODEL  MODEL.load\_state\_dict(torch.load(model\_export\_dir+model\_name))  print("model load ok")  @app.route('/')  def home():  return "Hello !!!\n"  @app.route('/predict', methods=['POST'])  def predict\_json():  params = request.get\_json()  data = params['data']  data = eval(data)  global MODEL  output = MODEL(data)  print("output:", output)  output\_str = str(output).replace("\n", "")  json\_str = jsonify(output\_str)  return json\_str  if \_\_name\_\_=='\_\_main\_\_':  load\_model("mnist\_cnn.pt")  app.run(host="0.0.0.0",port=80) |

1. **Creating a PyTorch serving service**

Pytorch serving pod를 외부에서 호출하기 위해서는 service 배포가 필요하다.

1. Pytorch serving service를 Kubernetes namespace kubeflow에 배포한다.

**kubectl create -f traindb-fserving-mnist-service.yaml -n kubeflow**

|  |
| --- |
| service/traindb-fserving-mnist-service created |

YAML파일은 다음과 같다.

## traindb-fserving-mnist-service****.yaml****

|  |
| --- |
| apiVersion: v1  kind: Service  metadata:  name: traindb-fserving-mnist-service  spec:  type: ClusterIP  #type: LoadBalancer  #type: NodePort  ports:  - name: "traindb-fserving-mnist-port"  protocol: "TCP"  port: 8282  targetPort: 80  selector:  name: traindb-fserving-mnist-pod |

1. 배포된 Pytorch serving service를 확인한다.

**kubectl get svc traindb-fserving-mnist-service -n kubeflow**

|  |
| --- |
| NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  traindb-fserving-mnist-service ClusterIP 10.152.183.106 <none> 8282/TCP 27s |

1. Pytorch serving service를 삭제할 경우 아래 명령어를 실행한다.

**kubectl delete svc traindb-fserving-mnist-service -n kubeflow**

|  |
| --- |
| pod " traindb-fserving-mnist-pod-796b4d7d57-qmnbf " deleted |

1. Pytorch serving service가 삭제된 것을 확인한다.

**kubectl get svc traindb-fserving-mnist-service -n kubeflow**

|  |
| --- |
| No resources found in kubeflow namespace. |

1. **Creating a PyTorch serving pod**
2. Pytorch serving POD를 Kubernetes namespace kubeflow에 배포한다.

**kubectl create -f traindb-fserving-mnist-pod.yaml -n kubeflow**

YAML파일은 다음과 같다.

## traindb-fserving-mnist-pod****.yaml****

|  |
| --- |
| apiVersion: apps/v1  kind: Deployment  metadata:  name: traindb-fserving-mnist-pod  spec:  replicas: 3  selector:  matchLabels:  name: traindb-fserving-mnist-pod  template:  metadata:  labels:  name: traindb-fserving-mnist-pod  spec:  volumes:  - name: task-pv-storage  persistentVolumeClaim:  claimName: traindb-models-claim  containers:  - name: traindb-mnist-serving-container  image: joleedocker/**pytorchjob-fserving-mnist**:1.0  ports:  - containerPort: 80  volumeMounts:  - mountPath: "/opt/mnist/models"  name: task-pv-storage  command:  - "python3"  - "/opt/mnist/src/mnist\_fserving.py" |

1. 배포된 Pytorch serving POD를 확인한다.

**kubectl get pods traindb-fserving-mnist-pod -n kubeflow**

|  |
| --- |
| NAME READY STATUS RESTARTS AGE  traindb-fserving-mnist-pod-796b4d7d57-qmnbf 1/1 Running 0 19m |

1. 배포된 Pytorch serving POD의 실행 로그를 확인한다.

**kubectl logs -f pod traindb-fserving-mnist-pod -n kubeflow**

|  |
| --- |
| \* Serving Flask app 'mnist\_fserving' (lazy loading)  \* Environment: production  WARNING: This is a development server. Do not use it in a production deployment.  Use a production WSGI server instead.  \* Debug mode: off  \* Running on all addresses.  WARNING: This is a development server. Do not use it in a production deployment.  \* Running on http://10.1.165.119:80/ (Press CTRL+C to quit)  127.0.0.1 - - [17/Jun/2022 11:56:33] "POST /predict HTTP/1.1" 200 - |

1. Pytorch serving POD를 삭제할 경우 다음 명령을 실행한다.

**kubectl delete pods traindb-fserving-mnist-pod -n kubeflow**

|  |
| --- |
| pod " traindb-fserving-mnist-pod-796b4d7d57-qmnbf " deleted |

1. Pytorch serving POD가 삭제된 것을 확인한다.

**kubectl get pods traindb-fserving-mnist-pod -n kubeflow**

|  |
| --- |
| No resources found in kubeflow namespace. |

1. **port forwarding**

Pytorch serving pod를 외부에서 호출하기 위한 service에 대한 port forward를 수행한다.

**kubectl port-forward -n kubeflow svc/traindb-fserving-mnist-service 12345:8282 &**

|  |
| --- |
| Forwarding from 127.0.0.1:12345 -> 8282  Forwarding from [::1]:12345 -> 8282 |

1. **predict**

host에서 curl 명령을 수행하여 Pytorch serving pod를 호출한다.

**curl --header "Content-Type: application/json" \**

**--request POST \**

**--data '{"data": "tensor([[[-0.4242, -0.4242, .… -0.4242, -0.4242]]])", "target": 0}' \**

**http://127.0.0.1:12345/predict**

해당 curl명령은 파일(curl\_client)로 제공된다.

아래와 같은 결과를 반환 받는다.

## ****결과****

|  |
| --- |
| "tensor([[ -9.3838, -11.5968, -10.1334, -10.4617, -9.8735, -2.7170, -9.0913, -1.5140, -5.0207, -0.3468]], grad\_fn=<LogSoftmaxBackward0>)" |