

تمرین دو (محمدباربد امیرمزلقانی) - ۸۱۰۱۰۲۳۴۸

سوال یک:

(الف)

در این سوال بر مبنای کد پیاده سازی شده توسط TA یک سری تغییرات لازم را اعمال میکنیم. یک لایه کانولوشن اضافه میکنیم، تعداد epoch را به ۱۰ افزایش میدهیم و جهت جلوگیری از بیشبرازش rate را به ۰.۵ افزایش دادیم)

The terminal window displays the following code:

```
GNU nano 7.2
classifier.py

import torch
import torch.nn as nn
import torchvision
import torchvision.transforms as transforms
import time

device = "cuda:0"
data_path = "/home/dmels/amirmazlaghani/fashion_mnist_data"
train_batch_size = 32
test_batch_size = 32
learning_rate = 0.001
n_epochs = 10

train_set = torchvision.datasets.FashionMNIST(data_path, download=True, transform=transforms.Compose([transforms.ToTensor()]))
test_set = torchvision.datasets.FashionMNIST(data_path, download=True, train=False, transform=transforms.Compose([transforms.ToTensor()]))
train_loader = torch.utils.data.DataLoader(train_set, batch_size=train_batch_size, shuffle=True)
test_loader = torch.utils.data.DataLoader(test_set, batch_size=test_batch_size, shuffle=False)

class FashionMNISTCNN(nn.Module):
    def __init__(self):
        super(FashionMNISTCNN, self).__init__()
        self.layer1 = nn.Sequential(
            nn.Conv2d(in_channels=1, out_channels=32, kernel_size=3, padding=1),
            nn.BatchNorm2d(32),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.layer2 = nn.Sequential(
            nn.Conv2d(in_channels=32, out_channels=64, kernel_size=3, padding=1),
            nn.BatchNorm2d(64),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.layer3 = nn.Sequential(
            nn.Conv2d(in_channels=64, out_channels=128, kernel_size=3, padding=1),
            nn.BatchNorm2d(128),
            nn.ReLU(),
            nn.MaxPool2d(kernel_size=2, stride=2)
        )
        self.fc1 = nn.Linear(in_features=128*3*3, out_features=1024)
        self.drop = nn.Dropout(0.5)
        self.fc2 = nn.Linear(in_features=1024, out_features=512)
        self.fc3 = nn.Linear(in_features=512, out_features=10)

    def forward(self, x):
        return self.fc3(self.drop(self.fc2(self.fc1(self.layer3(self.layer2(self.layer1(x)))))))

if __name__ == '__main__':
    model = FashionMNISTCNN().to(device)
    criterion = nn.CrossEntropyLoss()
    optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)
    start_time = time.time()

    for epoch in range(n_epochs):
        for i, (inputs, targets) in enumerate(train_loader):
            inputs, targets = inputs.to(device), targets.to(device)
            optimizer.zero_grad()
            outputs = model(inputs)
            loss = criterion(outputs, targets)
            loss.backward()
            optimizer.step()

            if i % 100 == 0:
                print(f'Epoch {epoch+1}/{n_epochs}, Step {i}, Loss: {loss.item():.4f}')

    end_time = time.time()
    print(f'Training completed in {end_time - start_time:.2f} seconds')

    model.eval()
    with torch.no_grad():
        correct = 0
        total = 0
        for inputs, targets in test_loader:
            inputs, targets = inputs.to(device), targets.to(device)
            outputs = model(inputs)
            _, predicted = torch.max(outputs, 1)
            total += targets.size(0)
            correct += (predicted == targets).sum().item()

        accuracy = 100 * correct / total
        print(f'Test accuracy: {accuracy:.2f}%')
```

Three GPU monitoring plots are displayed on the right side of the terminal window:

- Device 0 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 KiB/s TX: 0.000 GPU 0MHz MEM 405MHz TEMP 35°C FAN 0% POM 27 / 350 W GPU[0%] MEM[0.311Gi / 24.0 GiB]
- Device 1 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 KiB/s TX: 0.000 GPU 0MHz MEM 405MHz TEMP 34°C FAN 0% POM 22 / 350 W GPU[0%] MEM[0.311Gi / 24.0 GiB]
- GPU0 % GPU0 mem%
- GPU1 % GPU1 mem%
- TID USER DEV TYPE GPU GPU MEM CPU HOST MEM Command

```
GNU nano 7.2 classifier.py
def forward(self, x):
    out = self.layer1(x)
    out = self.layer2(out)
    out = self.layer3(out)
    out = out.view(out.size(0), -1)
    out = self.fc1(out)
    out = self.dropout(out)
    out = self.fc2(out)
    out = self.fc3(out)
    return out

if __name__ == "__main__":
    model = FashionMNISTCNN()
    model.to(device)

criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)
start_time = time.time()
for epoch in range(num_epochs):
    start_time = time.time()
    model.train()
    running_loss = 0.0
    for i, (images, labels) in enumerate(train_loader):
        images, labels = images.to(device), labels.to(device)

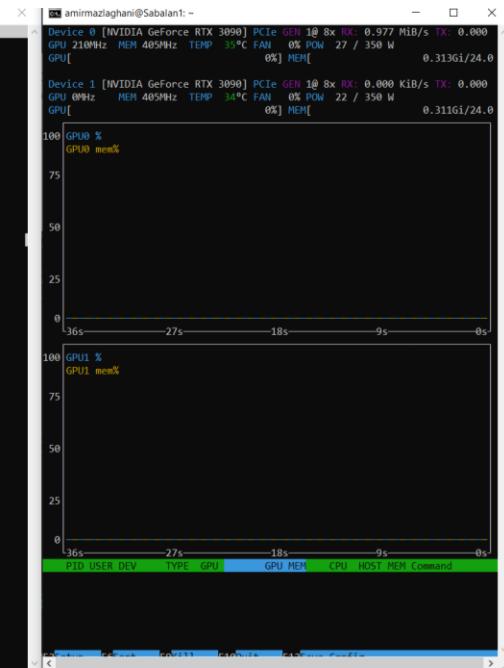
        optimizer.zero_grad()
        outputs = model(images)
        loss = criterion(outputs, labels)
        loss.backward()
        optimizer.step()
        cuda_mem = torch.cuda.max_memory_allocated(device=device)

        running_loss += loss.item()

    if i % 100 == 99:
        print(f"[{epoch + 1}, {i + 1}] loss: {running_loss / 100:.3f}")
        running_loss = 0.0

    model.eval()
    total = 0
    correct = 0
    with torch.no_grad():
        for images, labels in test_loader:
            images, labels = images.to(device), labels.to(device)
            outputs = model(images)
            _, predicted = torch.max(outputs.data, 1)

    Help   Write Out   Where Is   Cut   Execute   Location   Undo
    Exit   Read File   Replace   Paste   Justify   Go To Line   Ctrl-U   Ctrl-Z
    Ctrl-G   Ctrl-S   Ctrl-F   Ctrl-C   Ctrl-V   Ctrl-L   Ctrl-Y   Ctrl-R
    Ctrl-H   Ctrl-D   Ctrl-B   Ctrl-P   Ctrl-N   Ctrl-F   Ctrl-U   Ctrl-Z
```



```
GNU nano 7.2                         classifier.py
if __name__ == "__main__":
    model = FashionMNISTCNN()
    model.to(device)

    criterion = nn.CrossEntropyLoss()
    optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)
    start_time = time.time()
    for epoch in range(num_epochs):
        model.train()
        running_loss = 0.0
        for i, (images, labels) in enumerate(train_loader):
            images, labels = images.to(device), labels.to(device)

            optimizer.zero_grad()
            outputs = model(images)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()
            cuda_mem = torch.cuda.max_memory_allocated(device=device)

            running_loss += loss.item()

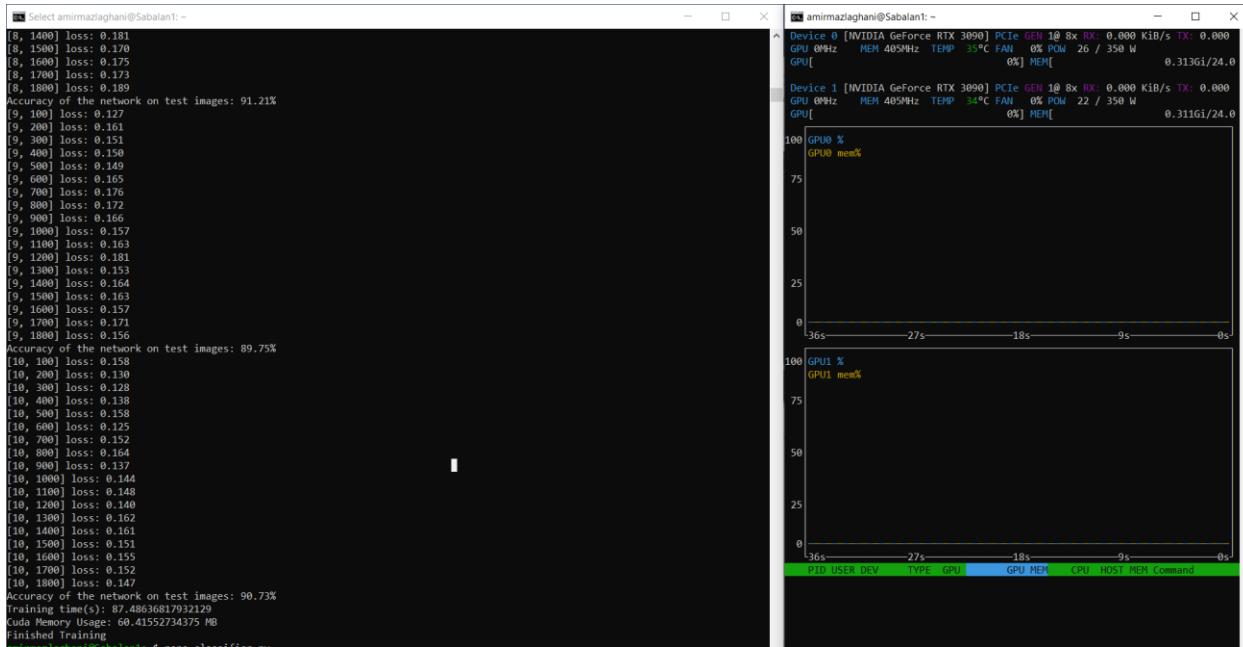
        if i % 100 == 99:
            print(f"[{epoch + 1}, {i + 1}] loss: {running_loss / 100:.3f}")
        running_loss = 0.0

    model.eval()
    total = 0
    correct = 0
    with torch.no_grad():
        for images, labels in test_loader:
            images, labels = images.to(device), labels.to(device)
            outputs = model(images)
            _, predicted = torch.max(outputs.data, 1)
            total += labels.size(0)
            correct += (predicted == labels).sum().item()

    accuracy = 100 * correct / total
    print(f"Accuracy of the network on test images: {accuracy:.2f}%")
end_time = time.time()
print(f"Training time: {end_time - start_time}s")
print(f"Cuda Memory Usage: [{cuda_mem / (1024 ** 2)} MB]")
print("Finished Training")
```

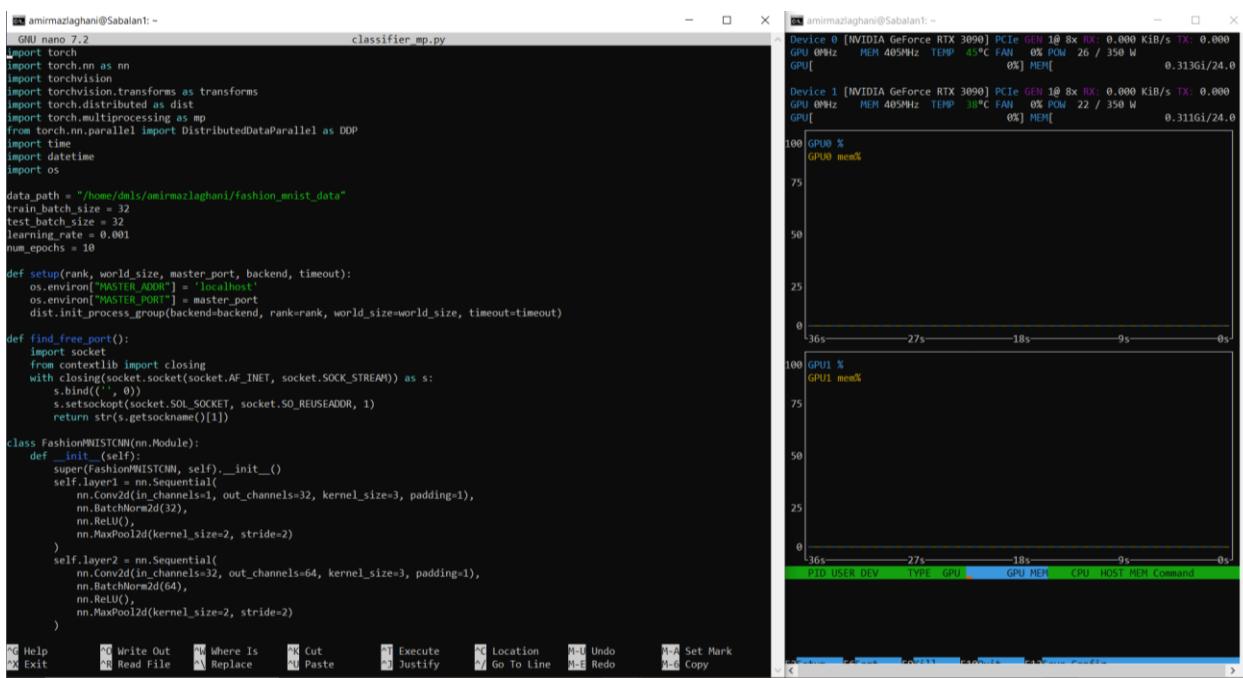


در ادامه دقت و زمان و حافظه را مشاهده میکنیم:



ب)

در قسمت دوم باز مینا را کد TA قرار میدهیم مدل را هم ، همان مدل مرحله الف میگذاریم در ادامه ابتدا کد این بخش و سپس خروجی آن که نشان دهنده کاهش محسوس زمان است را مشاهده میکنیم.



```

amirmazlaghani@Sabalan1: ~
GNU nano 7.2                                classifier mp.py
    nn.MaxPool2d(kernel_size=2, stride=2)
)
self.layer3 = nn.Sequential(
    nn.Conv2d(in_channels=64, out_channels=128, kernel_size=3, padding=1),
    nn.BatchNorm2d(128),
    nn.ReLU(),
    nn.MaxPool2d(kernel_size=2, stride=2)
)
self.fc1 = nn.Linear(in_features=128*3*3, out_features=1024)
self.drop = nn.Dropout(0.5)
self.fc2 = nn.Linear(in_features=1024, out_features=512)
self.fc3 = nn.Linear(in_features=512, out_features=10)

def forward(self, x):
    out = self.layer1(x)
    out = self.layer2(out)
    out = self.layer3(out)
    out = out.view(-1, -1)
    out = self.fc1(out)
    out = self.drop(out)
    out = self.fc2(out)
    out = self.fc3(out)
    return out

def load_data(rank, world_size):
    train_set = torchvision.datasets.FashionMNIST(data_path, download=True, transform=transforms.Compose([transforms.ToTensor()]))
    train_sampler = torch.utils.data.distributed.DistributedSampler(train_set, num_replicas=world_size, rank=rank)
    train_loader = torch.utils.data.DataLoader(dataset=train_set, sampler=train_sampler, batch_size=train_batch_size, shuffle=False)

    test_set = torchvision.datasets.FashionMNIST(data_path, download=True, train=False, transform=transforms.Compose([transforms.ToTensor()]))
    test_sampler = torch.utils.data.distributed.DistributedSampler(test_set, num_replicas=world_size, rank=rank)
    test_loader = torch.utils.data.DataLoader(dataset=test_set, sampler=test_sampler, batch_size=test_batch_size, shuffle=False)

    return train_loader, test_loader

def train(rank, world_size, master_port, backend, timeout):
    setup(rank, world_size, master_port, backend, timeout)
    torch.cuda.set_device(rank)
    train_loader, test_loader = load_data(rank, world_size)

    model = FashionMNISTNN().to(rank)
    ddp_model = DDP(model, device_ids=[rank])
    criterion = nn.CrossEntropyLoss()
    optimizer = torch.optim.Adam(ddp_model.parameters(), lr=learning_rate)

    start_time = time.time()
    for epoch in range(num_epochs):
        model.train()
        for i, (images, labels) in enumerate(train_loader):
            images, labels = images.to(rank), labels.to(rank)
            optimizer.zero_grad()
            outputs = ddp_model(images)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            if i % 100 == 99:
                print(f"Rank: {rank}, Epoch: {epoch + 1}, Batch: {i + 1}, Loss: {loss.item():.3f}")

        model.eval()
        total = 0
        correct = 0
        with torch.no_grad():
            for images, labels in test_loader:
                images, labels = images.to(rank), labels.to(rank)
                outputs = ddp_model(images)
                _, predicted = torch.max(outputs.data, 1)
                total += labels.size(0)
                correct += (predicted == labels).sum().item()

        accuracy = 100 * correct / total
        print(f"Rank: {rank}, Epoch: {epoch + 1}, Accuracy: {accuracy:.2f}%")

    end_time = time.time()
    cuda_mem = torch.cuda.max_memory_allocated(device=rank) / (1024 ** 2)
    print(f"Rank: {rank}, Training Time: {(end_time - start_time)} seconds, Max Memory Allocated: {(cuda_mem)} MB")

    if rank == 0:
        print("Finished Training")

if __name__ == "__main__":
    world_size = torch.cuda.device_count()
    master_port = find_free_port()
    backend = 'nccl'
    timeout = datetime.timedelta(seconds=10)
    mp.spawn(train, args=(world_size, master_port, backend, timeout), join=True)

```

Device 0 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX 0.000 Kib/s TX 0.000
GPU 0MHz MEM 405MHz TEMP 45°C FAN 0% POW 26 / 350 W GPU 0% MEM 0.313Gi/24.0
Device 1 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX 0.000 Kib/s TX 0.000
GPU 0MHz MEM 405MHz TEMP 38°C FAN 0% POW 22 / 350 W GPU 0% MEM 0.311Gi/24.0

```

amirmazlaghani@Sabalan1: ~
rank: 0, Epoch: 8, Batch: 800, Loss: 0.063
rank: 1, Epoch: 8, Batch: 800, Loss: 0.118
rank: 0, Epoch: 8, Batch: 900, Loss: 0.267
rank: 1, Epoch: 8, Batch: 900, Loss: 0.450
rank: 0, Epoch: 8, Accuracy: 89.26%
rank: 1, Epoch: 8, Accuracy: 89.10%
rank: 0, Epoch: 9, Batch: 100, Loss: 0.055
rank: 1, Epoch: 9, Batch: 100, Loss: 0.191
rank: 0, Epoch: 9, Batch: 200, Loss: 0.319
rank: 1, Epoch: 9, Batch: 200, Loss: 0.424
rank: 0, Epoch: 9, Batch: 300, Loss: 0.121
rank: 1, Epoch: 9, Batch: 300, Loss: 0.231
rank: 0, Epoch: 9, Batch: 400, Loss: 0.176
rank: 1, Epoch: 9, Batch: 400, Loss: 0.113
rank: 0, Epoch: 9, Batch: 500, Loss: 0.199
rank: 1, Epoch: 9, Batch: 500, Loss: 0.109
rank: 0, Epoch: 9, Batch: 600, Loss: 0.133
rank: 1, Epoch: 9, Batch: 600, Loss: 0.324
rank: 0, Epoch: 9, Batch: 700, Loss: 0.066
rank: 1, Epoch: 9, Batch: 700, Loss: 0.028
rank: 0, Epoch: 9, Batch: 800, Loss: 0.019
rank: 1, Epoch: 9, Batch: 800, Loss: 0.050
rank: 0, Epoch: 9, Batch: 900, Loss: 0.088
rank: 1, Epoch: 9, Batch: 900, Loss: 0.296
rank: 1, Epoch: 9, Accuracy: 89.54%
rank: 0, Epoch: 9, Accuracy: 89.44%
rank: 0, Epoch: 10, Batch: 100, Loss: 0.140
rank: 1, Epoch: 10, Batch: 100, Loss: 0.040
rank: 0, Epoch: 10, Batch: 200, Loss: 0.180
rank: 1, Epoch: 10, Batch: 200, Loss: 0.010
rank: 0, Epoch: 10, Batch: 300, Loss: 0.137
rank: 1, Epoch: 10, Batch: 300, Loss: 0.097
rank: 0, Epoch: 10, Batch: 400, Loss: 0.247
rank: 1, Epoch: 10, Batch: 400, Loss: 0.104
rank: 0, Epoch: 10, Batch: 500, Loss: 0.129
rank: 1, Epoch: 10, Batch: 500, Loss: 0.162
rank: 0, Epoch: 10, Batch: 600, Loss: 0.062
rank: 1, Epoch: 10, Batch: 600, Loss: 0.092
rank: 0, Epoch: 10, Batch: 700, Loss: 0.046
rank: 1, Epoch: 10, Batch: 700, Loss: 0.022
rank: 0, Epoch: 10, Batch: 800, Loss: 0.031
rank: 1, Epoch: 10, Batch: 800, Loss: 0.083
rank: 0, Epoch: 10, Batch: 900, Loss: 0.196
rank: 1, Epoch: 10, Batch: 900, Loss: 0.311
rank: 0, Training Time: 46.2173123596802 seconds, Max Memory Allocated: 68.517578125 MB
rank: 1, Training Time: 46.2173123596802 seconds, Max Memory Allocated: 68.517578125 MB
rank: 0, Epoch: 10, Accuracy: 89.88%
rank: 0, Training Time: 46.2220799229431 seconds, Max Memory Allocated: 68.517578125 MB
Finished Training

```

(ج)

طبق ویدیویی که کمکمدرس در اختیارمان گذاشت، یک سری تغییرات شامل حذف متغیرهای محیطی و هندل شدن آن‌ها توسط `torchrun` صورت گرفت همچنین اضافه شدن اسنپ شات برای `fault tolerance` و تغییراتی دیگر در همین موارد. در ادامه کد و سپس خروجی آن را مشاهده می‌کنیم که شبیه به قسمت دو شد

```

amirmazlaghani@Sabalan1: ~
GNU nano 7.2          classifier_torchrun.py
  images, labels = images.to(rank), labels.to(rank)
  optimizer.zero_grad()
  outputs = ddp_model(images)
  loss = criterion(outputs, labels)
  loss.backward()
  optimizer.step()

  if i % 100 == 99:
    print(f"Rank: {rank}, Epoch: {epoch + 1}, Batch: {i + 1}, Loss: {loss.item():.3f}")

  model.eval()
  total = 0
  correct = 0
  with torch.no_grad():
    for images, labels in test_loader:
      images, labels = images.to(rank), labels.to(rank)
      outputs = model(images)
      _, predicted = torch.max(outputs.data, 1)
      total += labels.size(0)
      correct += (predicted == labels).sum().item()

  accuracy = 100 * correct / total
  print(f"Rank: {rank}, Epoch: {epoch + 1}, Accuracy: {accuracy:.2f}%")

end_time = time.time()
print(f"Rank: {rank}, Training Time: {(end_time - start_time)} seconds")

if rank == 0:
  torch.save(model.state_dict(), 'model_final.pt')
  torch.save(optimizer.state_dict(), 'optimizer_final.pt')
  with open('final_snapshot.json', 'w') as file:
    json.dump({'epoch': num_epochs, 'model_state': 'model_final.pt', 'optimizer_state': 'optimizer_final.pt'})

if rank == 0:
  print("Finished training")

if __name__ == "__main__":
  rank = int(os.environ.get("LOCAL_RANK", 0))
  world_size = torch.cuda.device_count()
  master_port = find_free_port()
  backend = 'nccl'
  timeout = datetime.timedelta(seconds=10)
  train(rank, world_size, master_port, backend, timeout)

Help   Write Out  Where Is  Cut  Execute  Location  Undo
Exit  Read File  Replace  Paste  Justify  Go To Line  Redo
Type here to search
F2Setup F6Sort F9Kill F10Quit F12Save Config

```

Device 0 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 MiB/s TX: 0.000 MiB/s
GPU 0MHz MEM 405MHz TEMP 37°C FAN 0% POW 26 / 350 W GPU[0%] MEM[0%] 0.589Gi/24.000Gi

Device 1 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 MiB/s TX: 0.000 MiB/s
GPU 0MHz MEM 405MHz TEMP 37°C FAN 0% POW 22 / 350 W GPU[0%] MEM[0%] 0.314Gi/24.000Gi

PID	User	Dev	Type	GPU	GPU Mem	CPU	Host Mem	Command
882430	attar	0	Compute	0%	276MiB	1%	0%	489MiB python classifier.py

1007 PM 12/9/2023

```

amirmazlaghani@Sabalan1: ~
GNU nano 7.2          classifier_torchrun.py
  out = self.drop(out)
  out = self.fc2(out)
  return self.fc3(out)

def setup(backend):
  dist.init_process_group(backend=backend)

def find_free_port():
  import socket
  from contextlib import closing
  with closing(socket.socket(socket.AF_INET, socket.SOCK_STREAM)) as s:
    s.bind(('', 0))
    s.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
  return str(s.getsockname()[1])

def load_data(rank, world_size):
  train_set = torchvision.datasets.FashionMNIST(data_path, download=True, train=True, transform=transforms.ToTensor())
  train_sampler = torch.utils.data.distributed.DistributedSampler(train_set, num_replicas=world_size, rank=rank)
  train_loader = torch.utils.data.DataLoader(dataset=train_set, sampler=train_sampler, batch_size=train_set.train_labels.size(0))

  test_set = torchvision.datasets.FashionMNIST(data_path, download=True, train=False, transform=transforms.ToTensor())
  test_sampler = torch.utils.data.distributed.DistributedSampler(test_set, num_replicas=world_size, rank=rank)
  test_loader = torch.utils.data.DataLoader(dataset=test_set, sampler=test_sampler, batch_size=test_set.test_labels.size(0))

  return train_loader, test_loader

def train(rank, world_size, master_port, backend, timeout):
  setup(backend)
  torch.cuda.set_device(rank)
  train_loader, test_loader = load_data(rank, world_size)

  model = FashionMNISTNN().to(rank)
  ddp_model = DDP(model, device_ids=[rank])
  criterion = nn.CrossEntropyLoss()
  optimizer = torch.optim.Adam(ddp_model.parameters(), lr=learning_rate)

  start_epoch = 0
  snapshot_path = "training_snapshot.json"
  if os.path.exists(snapshot_path):
    with open(snapshot_path) as file:
      snapshot_json.load(file)
      start_epoch = snapshot['epoch']
    model.load_state_dict(torch.load(snapshot['model_state']))
    optimizer.load_state_dict(torch.load(snapshot['optimizer_state']))

Help   Write Out  Where Is  Cut  Execute  Location  Undo
Exit  Read File  Replace  Paste  Justify  Go To Line  Redo
Type here to search
F2Setup F6Sort F9Kill F10Quit F12Save Config

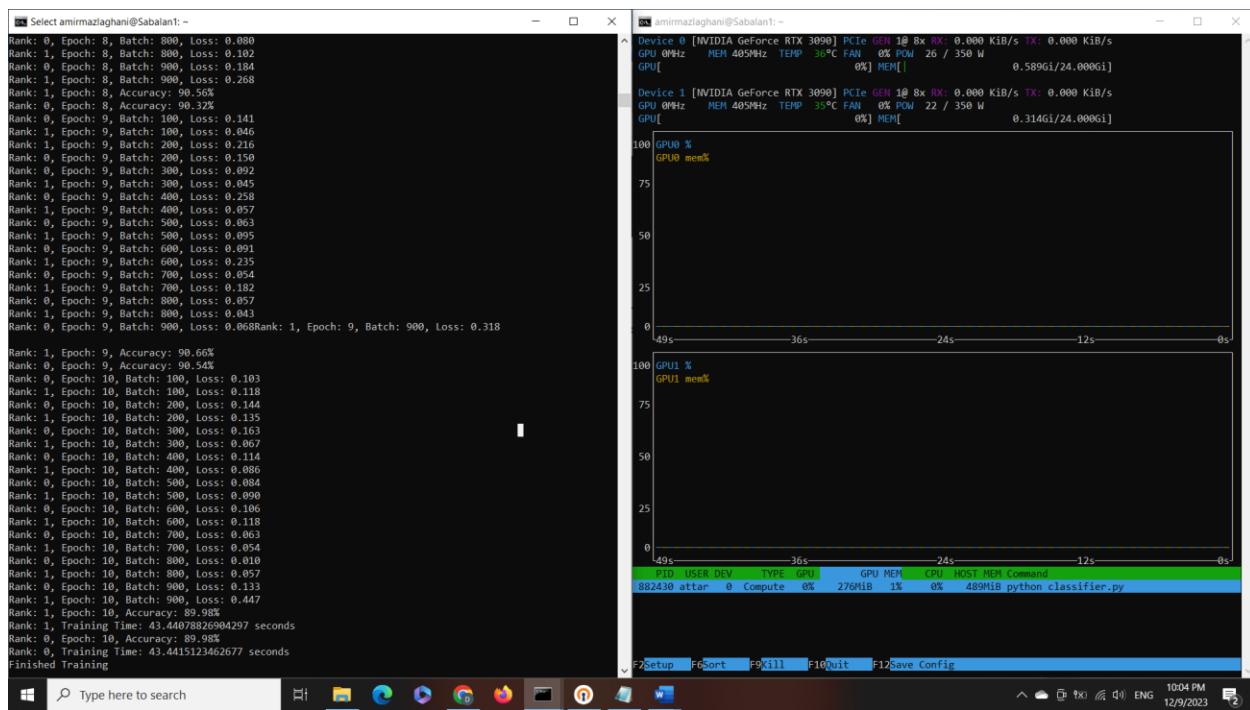
```

Device 0 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 1.953 MiB/s TX: 0.000 MiB/s
GPU 0MHz MEM 405MHz TEMP 37°C FAN 0% POW 26 / 350 W GPU[0%] MEM[0%] 0.589Gi/24.000Gi

Device 1 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.977 MiB/s TX: 0.000 MiB/s
GPU 0MHz MEM 405MHz TEMP 37°C FAN 0% POW 23 / 350 W GPU[0%] MEM[0%] 0.314Gi/24.000Gi

PID	User	Dev	Type	GPU	GPU Mem	CPU	Host Mem	Command
882430	attar	0	Compute	0%	276MiB	1%	0%	489MiB python classifier.py

1007 PM 12/9/2023



سوال دوم:

سایز ۱۶ :

```

amirmazlaghani@Sabalan1: ~
Rank: 1, Epoch: 9, Batch: 160, Loss: 0.045
Rank: 0, Epoch: 9, Batch: 1700, Loss: 0.085
Rank: 1, Epoch: 9, Batch: 1700, Loss: 0.012
Rank: 0, Epoch: 9, Batch: 1800, Loss: 0.066
Rank: 1, Epoch: 9, Batch: 1800, Loss: 0.200
Rank: 1, Epoch: 9, Accuracy: 91.18%
Rank: 0, Epoch: 9, Accuracy: 90.48%
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.015
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.390
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.091
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.039
Rank: 0, Epoch: 10, Batch: 300, Loss: 0.047
Rank: 1, Epoch: 10, Batch: 300, Loss: 0.091
Rank: 0, Epoch: 10, Batch: 400, Loss: 0.130
Rank: 1, Epoch: 10, Batch: 400, Loss: 0.269
Rank: 0, Epoch: 10, Batch: 500, Loss: 0.205
Rank: 1, Epoch: 10, Batch: 500, Loss: 0.041
Rank: 0, Epoch: 10, Batch: 600, Loss: 0.233
Rank: 1, Epoch: 10, Batch: 600, Loss: 0.151
Rank: 1, Epoch: 10, Batch: 700, Loss: 0.004
Rank: 0, Epoch: 10, Batch: 700, Loss: 0.093
Rank: 0, Epoch: 10, Batch: 800, Loss: 0.001
Rank: 1, Epoch: 10, Batch: 800, Loss: 0.032
Rank: 1, Epoch: 10, Batch: 900, Loss: 0.067
Rank: 0, Epoch: 10, Batch: 900, Loss: 0.180
Rank: 0, Epoch: 10, Batch: 1000, Loss: 0.266
Rank: 1, Epoch: 10, Batch: 1000, Loss: 0.026
Rank: 0, Epoch: 10, Batch: 1100, Loss: 0.102
Rank: 1, Epoch: 10, Batch: 1100, Loss: 0.080
Rank: 0, Epoch: 10, Batch: 1200, Loss: 0.079
Rank: 1, Epoch: 10, Batch: 1200, Loss: 0.551
Rank: 0, Epoch: 10, Batch: 1300, Loss: 0.055
Rank: 1, Epoch: 10, Batch: 1300, Loss: 0.334
Rank: 0, Epoch: 10, Batch: 1400, Loss: 0.208
Rank: 1, Epoch: 10, Batch: 1400, Loss: 0.223
Rank: 0, Epoch: 10, Batch: 1500, Loss: 0.042
Rank: 1, Epoch: 10, Batch: 1500, Loss: 0.227
Rank: 0, Epoch: 10, Batch: 1600, Loss: 0.049
Rank: 1, Epoch: 10, Batch: 1700, Loss: 0.035
Rank: 1, Epoch: 10, Batch: 1700, Loss: 0.015
Rank: 0, Epoch: 10, Batch: 1800, Loss: 0.055
Rank: 1, Epoch: 10, Batch: 1800, Loss: 0.328
Rank: 1, Epoch: 10, Accuracy: 90.60%
Rank: 1, Training Time: 82.99076890945435 seconds
Rank: 0, Epoch: 10, Accuracy: 90.40%
Rank: 0, Training Time: 82.9952712059021 seconds
Finished Training
amirmazlaghani@Sabalan1: ~
```

F2Setup F6Sort F9Kill F10Quit F12Save Config

1025 PM 12/9/2023

سایز: ۶۴

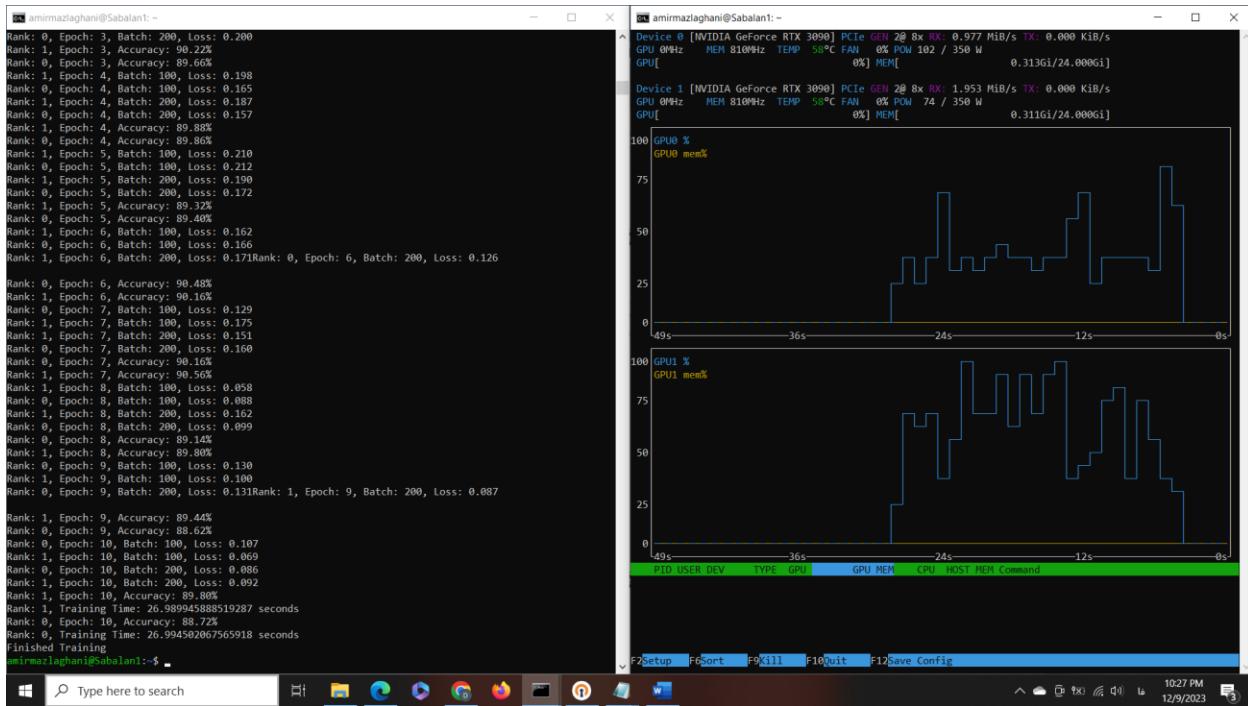
```

amirmazlaghani@Sabalan1: ~
Rank: 0, Epoch: 6, Batch: 400, Loss: 0.064
Rank: 1, Epoch: 6, Batch: 400, Loss: 0.134
Rank: 1, Epoch: 6, Accuracy: 90.54%
Rank: 0, Epoch: 6, Accuracy: 90.30%
Rank: 0, Epoch: 7, Batch: 100, Loss: 0.064
Rank: 1, Epoch: 7, Batch: 100, Loss: 0.179
Rank: 0, Epoch: 7, Batch: 200, Loss: 0.112
Rank: 1, Epoch: 7, Batch: 200, Loss: 0.225
Rank: 1, Epoch: 7, Batch: 300, Loss: 0.141
Rank: 0, Epoch: 7, Batch: 300, Loss: 0.141
Rank: 1, Epoch: 7, Batch: 400, Loss: 0.114
Rank: 0, Epoch: 7, Batch: 400, Loss: 0.066
Rank: 1, Epoch: 7, Accuracy: 91.14%
Rank: 0, Epoch: 7, Accuracy: 90.98%
Rank: 1, Epoch: 8, Batch: 100, Loss: 0.089
Rank: 0, Epoch: 8, Batch: 100, Loss: 0.114
Rank: 0, Epoch: 8, Batch: 200, Loss: 0.306
Rank: 1, Epoch: 8, Batch: 200, Loss: 0.110
Rank: 1, Epoch: 8, Batch: 300, Loss: 0.170
Rank: 0, Epoch: 8, Batch: 300, Loss: 0.197
Rank: 1, Epoch: 8, Batch: 400, Loss: 0.115
Rank: 0, Epoch: 8, Batch: 400, Loss: 0.024
Rank: 1, Epoch: 8, Accuracy: 91.00%
Rank: 1, Epoch: 8, Accuracy: 90.96%
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.126
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.054
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.098
Rank: 0, Epoch: 9, Batch: 200, Loss: 0.187
Rank: 1, Epoch: 9, Batch: 300, Loss: 0.087
Rank: 0, Epoch: 9, Batch: 300, Loss: 0.099
Rank: 1, Epoch: 9, Batch: 400, Loss: 0.070
Rank: 0, Epoch: 9, Batch: 400, Loss: 0.014
Rank: 1, Epoch: 9, Accuracy: 90.88%
Rank: 0, Epoch: 9, Accuracy: 90.78%
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.072
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.073
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.156
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.044
Rank: 0, Epoch: 10, Batch: 300, Loss: 0.228
Rank: 1, Epoch: 10, Batch: 300, Loss: 0.167
Rank: 1, Epoch: 10, Batch: 400, Loss: 0.158
Rank: 0, Epoch: 10, Batch: 400, Loss: 0.063
Rank: 1, Epoch: 10, Accuracy: 90.54%
Rank: 1, Training Time: 29.478872537612915 seconds
Rank: 0, Epoch: 10, Accuracy: 90.76%
Rank: 0, Training Time: 29.482677936553955 seconds
Finished Training
amirmazlaghani@Sabalan1: ~
```

F2Setup F6Sort F9Kill F10Quit F12Save Config

1026 PM 12/9/2023

سایز: ۱۲۸



تغییرات per epoch هم در لاغ تصاویر موجود است.(وقت برای نمودار کشیدن نبود) 😊

سوال چهار :

SGD: کد در ضمیمه موجود است

```

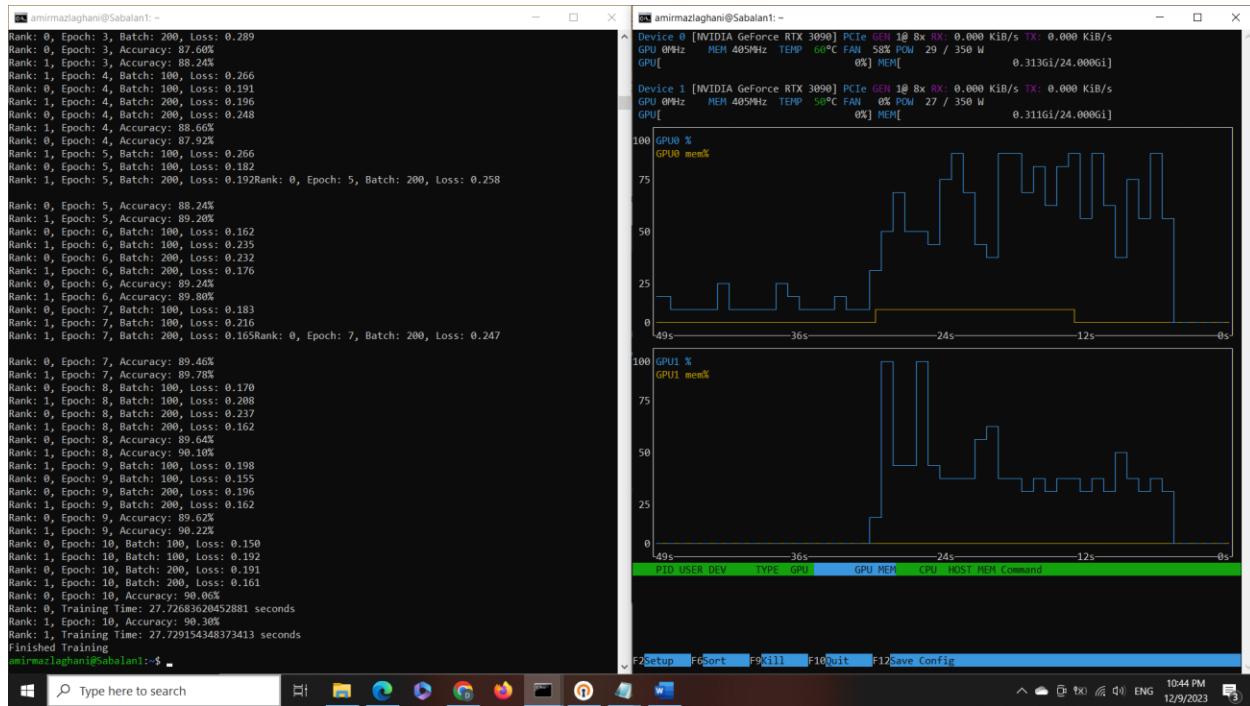
amirmazlaghani@Sabalan: ~
Rank: 1, Epoch: 3, Batch: 200, Loss: 0.300
Rank: 1, Epoch: 3, Accuracy: 86.52%
Rank: 0, Epoch: 3, Accuracy: 86.02%
Rank: 0, Epoch: 4, Batch: 100, Loss: 0.274
Rank: 1, Epoch: 4, Batch: 100, Loss: 0.353
Rank: 1, Epoch: 4, Batch: 200, Loss: 0.262
Rank: 0, Epoch: 4, Batch: 200, Loss: 0.352
Rank: 1, Epoch: 4, Accuracy: 87.64%
Rank: 0, Epoch: 4, Accuracy: 86.90%
Rank: 0, Epoch: 5, Batch: 100, Loss: 0.256
Rank: 0, Epoch: 5, Batch: 100, Loss: 0.325
Rank: 0, Epoch: 5, Batch: 200, Loss: 0.316
Rank: 1, Epoch: 5, Batch: 200, Loss: 0.267
Rank: 1, Epoch: 5, Accuracy: 88.54%
Rank: 0, Epoch: 5, Accuracy: 87.54%
Rank: 1, Epoch: 6, Batch: 100, Loss: 0.324
Rank: 0, Epoch: 6, Batch: 100, Loss: 0.217
Rank: 1, Epoch: 6, Batch: 200, Loss: 0.255
Rank: 0, Epoch: 6, Batch: 200, Loss: 0.290
Rank: 1, Epoch: 6, Accuracy: 89.12%
Rank: 0, Epoch: 6, Accuracy: 88.24%
Rank: 1, Epoch: 7, Batch: 100, Loss: 0.286
Rank: 0, Epoch: 7, Batch: 100, Loss: 0.208
Rank: 0, Epoch: 7, Batch: 200, Loss: 0.223
Rank: 1, Epoch: 7, Accuracy: 89.70%
Rank: 0, Epoch: 7, Accuracy: 88.96%
Rank: 1, Epoch: 8, Batch: 100, Loss: 0.288
Rank: 0, Epoch: 8, Batch: 100, Loss: 0.204
Rank: 1, Epoch: 8, Batch: 200, Loss: 0.218
Rank: 0, Epoch: 8, Accuracy: 89.46%
Rank: 1, Epoch: 8, Accuracy: 90.20%
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.282
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.226
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.285
Rank: 0, Epoch: 9, Accuracy: 89.72%
Rank: 1, Epoch: 9, Accuracy: 90.18%
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.262
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.192
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.251
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.219
Rank: 0, Epoch: 10, Accuracy: 89.82%
Rank: 0, Training Time: 26.957414150238037 seconds
Rank: 1, Epoch: 10, Accuracy: 90.52%
Rank: 1, Training Time: 26.969444036483765 seconds
Finished Training
amirmazlaghani@Sabalan: ~
```

:RMSProp

```

amirmazlaghani@Sabalan: ~
Rank: 0, Epoch: 3, Batch: 200, Loss: 0.316
Rank: 1, Epoch: 3, Accuracy: 85.36%
Rank: 0, Epoch: 3, Accuracy: 84.82%
Rank: 1, Epoch: 4, Batch: 100, Loss: 0.484
Rank: 0, Epoch: 4, Batch: 100, Loss: 0.357
Rank: 0, Epoch: 4, Batch: 200, Loss: 0.257
Rank: 1, Epoch: 4, Batch: 200, Loss: 0.214
Rank: 1, Epoch: 4, Accuracy: 87.80%
Rank: 0, Epoch: 4, Accuracy: 86.64%
Rank: 0, Epoch: 5, Batch: 100, Loss: 0.302
Rank: 1, Epoch: 5, Batch: 100, Loss: 0.414
Rank: 0, Epoch: 5, Batch: 200, Loss: 0.263
Rank: 1, Epoch: 5, Batch: 200, Loss: 0.246
Rank: 1, Epoch: 5, Accuracy: 88.92%
Rank: 0, Epoch: 5, Accuracy: 88.44%
Rank: 0, Epoch: 6, Batch: 100, Loss: 0.287
Rank: 1, Epoch: 6, Batch: 100, Loss: 0.334
Rank: 0, Epoch: 6, Batch: 200, Loss: 0.237
Rank: 1, Epoch: 6, Batch: 200, Loss: 0.246
Rank: 1, Epoch: 6, Accuracy: 90.12%
Rank: 0, Epoch: 6, Accuracy: 89.86%
Rank: 0, Epoch: 7, Batch: 100, Loss: 0.155
Rank: 1, Epoch: 7, Batch: 100, Loss: 0.119
Rank: 0, Epoch: 7, Batch: 200, Loss: 0.202
Rank: 1, Epoch: 7, Batch: 200, Loss: 0.249
Rank: 1, Epoch: 7, Accuracy: 89.46%
Rank: 0, Epoch: 7, Accuracy: 88.84%
Rank: 0, Epoch: 8, Batch: 100, Loss: 0.152
Rank: 1, Epoch: 8, Batch: 100, Loss: 0.155
Rank: 0, Epoch: 8, Batch: 200, Loss: 0.185
Rank: 1, Epoch: 8, Batch: 200, Loss: 0.203
Rank: 0, Epoch: 8, Accuracy: 89.32%
Rank: 1, Epoch: 8, Accuracy: 90.14%
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.127
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.153
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.147
Rank: 0, Epoch: 9, Batch: 200, Loss: 0.178
Rank: 0, Epoch: 9, Accuracy: 88.36%
Rank: 1, Epoch: 9, Accuracy: 89.00%
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.129
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.089
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.122
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.157
Rank: 0, Epoch: 10, Accuracy: 90.24%
Rank: 0, Training Time: 27.1936109066000952 seconds
Rank: 1, Epoch: 10, Accuracy: 90.52%
Rank: 1, Training Time: 27.202853441238403 seconds
Finished Training
amirmazlaghani@Sabalan: ~
```

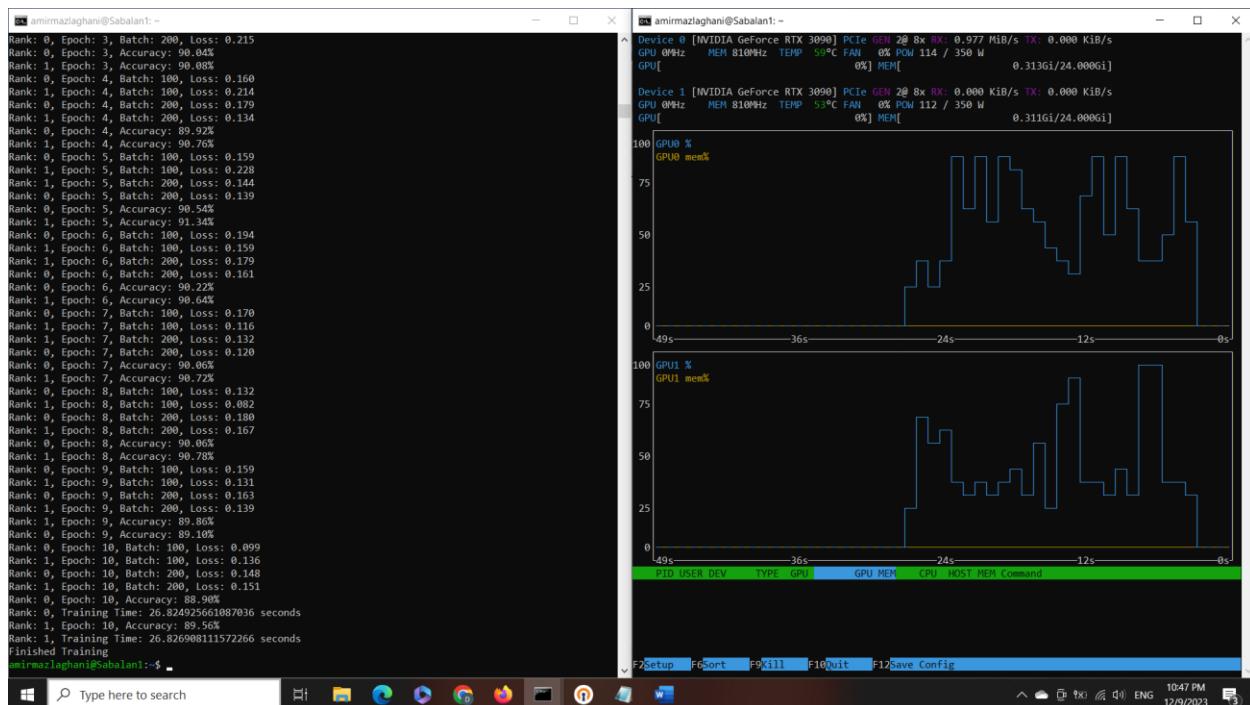
:Adagrad



کد تمامی قسمت ها به صورت تفکیک شده در ضمیمه موجود است.

سوال پنج:

:Nccl 128



Nccl 32 :

```
amirmazlaghani@Sabalant: ~
Rank: 1, Epoch: 8, Batch: 800, Loss: 0.111
Rank: 0, Epoch: 8, Batch: 900, Loss: 0.150
Rank: 1, Epoch: 8, Batch: 900, Loss: 0.066
Rank: 1, Epoch: 8, Accuracy: 89.42%
Rank: 0, Epoch: 8, Accuracy: 89.84%
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.168
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.090
Rank: 0, Epoch: 9, Batch: 200, Loss: 0.233
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.027
Rank: 0, Epoch: 9, Batch: 300, Loss: 0.144
Rank: 1, Epoch: 9, Batch: 300, Loss: 0.088
Rank: 0, Epoch: 9, Batch: 400, Loss: 0.220
Rank: 1, Epoch: 9, Batch: 400, Loss: 0.069
Rank: 0, Epoch: 9, Batch: 500, Loss: 0.113
Rank: 1, Epoch: 9, Batch: 500, Loss: 0.116
Rank: 0, Epoch: 9, Batch: 600, Loss: 0.166
Rank: 1, Epoch: 9, Batch: 600, Loss: 0.140
Rank: 0, Epoch: 9, Batch: 700, Loss: 0.055
Rank: 1, Epoch: 9, Batch: 700, Loss: 0.019
Rank: 0, Epoch: 9, Batch: 800, Loss: 0.036
Rank: 1, Epoch: 9, Batch: 800, Loss: 0.072
Rank: 0, Epoch: 9, Batch: 900, Loss: 0.139
Rank: 1, Epoch: 9, Batch: 900, Loss: 0.367
Rank: 0, Epoch: 9, Accuracy: 89.54%
Rank: 1, Epoch: 9, Accuracy: 89.2%
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.027
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.195
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.089
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.121
Rank: 0, Epoch: 10, Batch: 300, Loss: 0.135
Rank: 1, Epoch: 10, Batch: 300, Loss: 0.113
Rank: 0, Epoch: 10, Batch: 400, Loss: 0.224
Rank: 1, Epoch: 10, Batch: 400, Loss: 0.055
Rank: 0, Epoch: 10, Batch: 500, Loss: 0.091
Rank: 1, Epoch: 10, Batch: 500, Loss: 0.029
Rank: 0, Epoch: 10, Batch: 600, Loss: 0.136
Rank: 1, Epoch: 10, Batch: 600, Loss: 0.127
Rank: 0, Epoch: 10, Batch: 700, Loss: 0.013
Rank: 0, Epoch: 10, Batch: 800, Loss: 0.026
Rank: 1, Epoch: 10, Batch: 800, Loss: 0.024
Rank: 0, Epoch: 10, Batch: 900, Loss: 0.129
Rank: 1, Epoch: 10, Batch: 900, Loss: 0.455
Rank: 1, Epoch: 10, Accuracy: 89.84%
Rank: 1, Training Time: 43.62501668930054 seconds
Rank: 0, Epoch: 10, Accuracy: 89.26%
Rank: 0, Training Time: 43.640444809341431 seconds
Finished Training
amirmazlaghani@Sabalant: ~
```

GPU0 % GPU0 mem%
GPU1 % GPU1 mem%

PID USER DEV TYPE GPU GPU MEM CPU HOST MEM Command
888209 attar 0 Compute 11% 454MB 2% 100% 1103MB python classifier.py

:Gloo 128

```
amirmazlaghani@Sabalant: ~
Rank: 0, Epoch: 3, Batch: 200, Loss: 0.245
Rank: 1, Epoch: 3, Accuracy: 90.40%
Rank: 0, Epoch: 3, Accuracy: 90.40%
Rank: 0, Epoch: 4, Batch: 100, Loss: 0.220
Rank: 1, Epoch: 4, Batch: 100, Loss: 0.160
Rank: 0, Epoch: 4, Batch: 200, Loss: 0.177
Rank: 1, Epoch: 4, Batch: 200, Loss: 0.182
Rank: 1, Epoch: 4, Accuracy: 89.92%
Rank: 0, Epoch: 4, Accuracy: 89.48%
Rank: 0, Epoch: 5, Batch: 100, Loss: 0.207
Rank: 1, Epoch: 5, Batch: 100, Loss: 0.179
Rank: 1, Epoch: 5, Batch: 200, Loss: 0.139
Rank: 0, Epoch: 5, Batch: 200, Loss: 0.135
Rank: 1, Epoch: 5, Accuracy: 88.30%
Rank: 0, Epoch: 5, Accuracy: 88.06%
Rank: 0, Epoch: 6, Batch: 100, Loss: 0.165
Rank: 1, Epoch: 6, Batch: 100, Loss: 0.150
Rank: 0, Epoch: 6, Batch: 200, Loss: 0.099
Rank: 1, Epoch: 6, Batch: 200, Loss: 0.156
Rank: 1, Epoch: 6, Accuracy: 89.20%
Rank: 0, Epoch: 6, Accuracy: 89.20%
Rank: 0, Epoch: 7, Batch: 100, Loss: 0.140
Rank: 1, Epoch: 7, Batch: 100, Loss: 0.183
Rank: 0, Epoch: 7, Batch: 200, Loss: 0.130
Rank: 1, Epoch: 7, Batch: 200, Loss: 0.137
Rank: 1, Epoch: 7, Accuracy: 89.78%
Rank: 0, Epoch: 7, Accuracy: 88.92%
Rank: 1, Epoch: 8, Batch: 100, Loss: 0.166
Rank: 0, Epoch: 8, Batch: 100, Loss: 0.217
Rank: 0, Epoch: 8, Batch: 200, Loss: 0.146
Rank: 1, Epoch: 8, Batch: 200, Loss: 0.168
Rank: 1, Epoch: 8, Accuracy: 90.88%
Rank: 0, Epoch: 8, Accuracy: 90.22%
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.195
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.124
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.141
Rank: 1, Epoch: 9, Accuracy: 89.98%
Rank: 0, Epoch: 9, Accuracy: 89.98%
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.166
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.119
Rank: 1, Epoch: 10, Batch: 200, Loss: 0.127
Rank: 1, Epoch: 10, Accuracy: 90.18%
Rank: 1, Training Time: 29.53988572921753 seconds
Rank: 0, Epoch: 10, Accuracy: 89.42%
Rank: 0, Training Time: 29.555269956588745 seconds
Finished Training
amirmazlaghani@Sabalant: ~
```

GPU0 % GPU0 mem%
GPU1 % GPU1 mem%

PID USER DEV TYPE GPU GPU MEM CPU HOST MEM Command
888209 attar 0 Compute 11% 454MB 2% 100% 1103MB python classifier.py

:Gloo 32

```

amirmazlaghani@Sabalan1: ~
Rank: 0, Epoch: 8, Batch: 800, Loss: 0.089
Rank: 0, Epoch: 8, Batch: 900, Loss: 0.078Rank: 1, Epoch: 8, Batch: 900, Loss: 0.213
Rank: 0, Epoch: 8, Accuracy: 89.50%
Rank: 1, Epoch: 8, Accuracy: 90.60%
Rank: 0, Epoch: 9, Batch: 100, Loss: 0.201
Rank: 1, Epoch: 9, Batch: 100, Loss: 0.179
Rank: 0, Epoch: 9, Batch: 200, Loss: 0.206
Rank: 1, Epoch: 9, Batch: 200, Loss: 0.102
Rank: 0, Epoch: 9, Batch: 300, Loss: 0.174
Rank: 1, Epoch: 9, Batch: 300, Loss: 0.117
Rank: 0, Epoch: 9, Batch: 400, Loss: 0.120
Rank: 1, Epoch: 9, Batch: 400, Loss: 0.038
Rank: 0, Epoch: 9, Batch: 500, Loss: 0.172
Rank: 1, Epoch: 9, Batch: 500, Loss: 0.114
Rank: 1, Epoch: 9, Batch: 600, Loss: 0.139Rank: 0, Epoch: 9, Batch: 600, Loss: 0.215
Rank: 0, Epoch: 9, Batch: 700, Loss: 0.060
Rank: 1, Epoch: 9, Batch: 700, Loss: 0.037
Rank: 0, Epoch: 9, Batch: 800, Loss: 0.092Rank: 1, Epoch: 9, Batch: 800, Loss: 0.093
Rank: 0, Epoch: 9, Batch: 900, Loss: 0.160
Rank: 1, Epoch: 9, Batch: 900, Loss: 0.402
Rank: 0, Epoch: 9, Accuracy: 89.72%
Rank: 0, Epoch: 10, Batch: 100, Loss: 0.058
Rank: 1, Epoch: 10, Batch: 100, Loss: 0.029
Rank: 0, Epoch: 10, Batch: 200, Loss: 0.145Rank: 1, Epoch: 10, Batch: 200, Loss: 0.029
Rank: 0, Epoch: 10, Batch: 300, Loss: 0.082Rank: 1, Epoch: 10, Batch: 300, Loss: 0.082
Rank: 0, Epoch: 10, Batch: 400, Loss: 0.191Rank: 1, Epoch: 10, Batch: 400, Loss: 0.104
Rank: 0, Epoch: 10, Batch: 500, Loss: 0.168
Rank: 1, Epoch: 10, Batch: 500, Loss: 0.024
Rank: 0, Epoch: 10, Batch: 600, Loss: 0.198Rank: 1, Epoch: 10, Batch: 600, Loss: 0.162
Rank: 0, Epoch: 10, Batch: 700, Loss: 0.051
Rank: 1, Epoch: 10, Batch: 700, Loss: 0.066
Rank: 0, Epoch: 10, Batch: 800, Loss: 0.037
Rank: 1, Epoch: 10, Batch: 800, Loss: 0.044
Rank: 0, Epoch: 10, Batch: 900, Loss: 0.070
Rank: 1, Epoch: 10, Batch: 900, Loss: 0.266
Rank: 0, Epoch: 10, Accuracy: 89.96%
Rank: 0, Training Time: 70.9379186630249 seconds
Rank: 1, Epoch: 10, Accuracy: 90.06%
Rank: 1, Training Time: 70.93828058242798 seconds
Finished Training
amirmazlaghani@Sabalan1: ~
```

Device 0 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 Kib/s TX: 0.000 Kib/s GPU 0MHz MEM 405MHz TEMP 55°C FAN 58% POW 27 / 350 W GPU[0%] MEM[0.313Gi/24.000Gi]

Device 1 [NVIDIA GeForce RTX 3090] PCIe GEN 1@ 8x RX: 0.000 Kib/s TX: 0.000 Kib/s GPU 0MHz MEM 405MHz TEMP 55°C FAN 37% POW 24 / 350 W GPU[0%] MEM[0.311Gi/24.000Gi]

F2Setup F4Sort F9Kill F10Quit F12Save Config

1053 PM 12/9/2023