Exploration on Behaviour of Neural Network Model using PyTorch on Graphics Processing Unit

18 DEC 2023

PyTorch

Step 1

https://pytorch.org/get-started/locally/

PyTorch

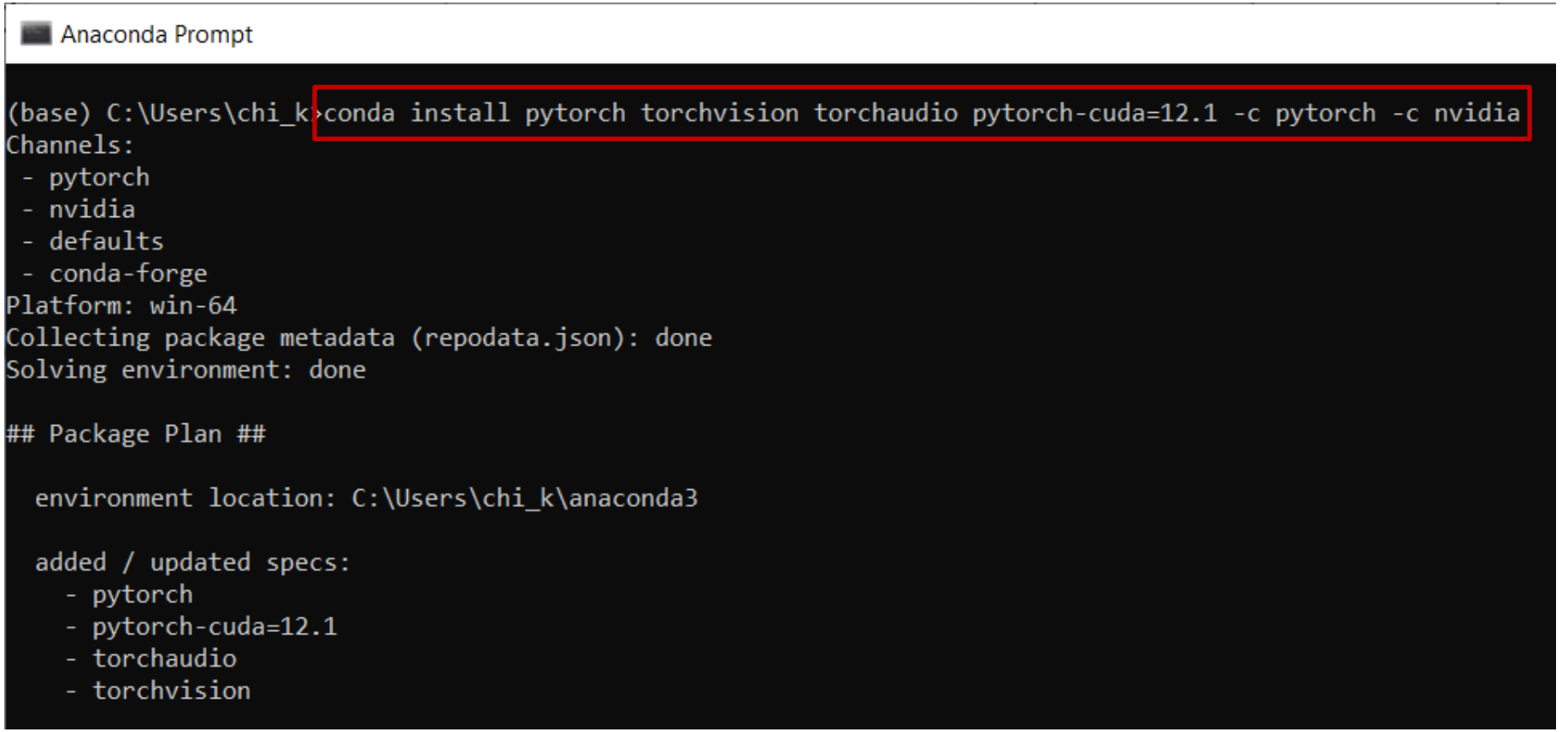
Step 2

NOTE: Latest PyTorch requires Python 3.8 or later.

PyTorch Build	**Stable (2.1.1)**		Preview (Nightly)	
Your OS	Linux	Mac	**Windows**	
Package	**Conda**	Pip	LibTorch	Source
Language	**Python**		C++ / Java	
Compute Platform	CUDA 11.8	**CUDA 12.1**	~~ROCm 5.6~~	CPU
Run this Command:	`conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c nvidia`			

PyTorch

Step 3



PyTorch

Step 4

Success if you can get this

[1]:
```
import torch

print('GPU device available for use: ', torch.cuda.is_available())
print()

print('Number of GPU devices: ', torch.cuda.device_count())
print()

print('GPU device number: ', torch.cuda.current_device())
print()

print('Model of GPU device: ', torch.cuda.get_device_name(torch.cuda.current_device()))
print()
```

```
GPU device available for use:  True

Number of GPU devices:  1

GPU device number:  0

Model of GPU device:  NVIDIA GeForce GTX 1660 Ti
```

GPU

To train NN model in GPU, tensor and NN model must be pushed to GPU before training

Step 6: Set up GPU

[10]:
```
if torch.cuda.is_available():
    device = torch.device("cuda:0")
    print("Running on the GPU")
else:
    device = torch.device("cpu")
    print("Running on the CPU")
```

```
Running on the GPU
```

Step 7: Push to GPU

[11]:
```
X = X.to(device)
y = y.to(device)
```

[12]:
```
X[:5]
```

[12]:
```
tensor([[6.0000e+00, 1.4800e+02, 7.2000e+01, 3.5000e+01, 0.0000e+00, 3.3600e+01,
         6.2700e-01, 5.0000e+01],
        [1.0000e+00, 8.5000e+01, 6.6000e+01, 2.9000e+01, 0.0000e+00, 2.6600e+01,
         3.5100e-01, 3.1000e+01],
        [8.0000e+00, 1.8300e+02, 6.4000e+01, 0.0000e+00, 0.0000e+00, 2.3300e+01,
         6.7200e-01, 3.2000e+01],
        [1.0000e+00, 8.9000e+01, 6.6000e+01, 2.3000e+01, 9.4000e+01, 2.8100e+01,
         1.6700e-01, 2.1000e+01],
        [0.0000e+00, 1.3700e+02, 4.0000e+01, 3.5000e+01, 1.6800e+02, 4.3100e+01,
         2.2880e+00, 3.3000e+01]], device='cuda:0')
```

Step 10: Set up model and push to GPU

[19]:
```
# Hidden layer
#    Multi-Layer Perceptron = ReLU
#    Convolutional Neural Network = ReLU
#    Recurrent Neural Network = Sigmoid or Tanh

# Output layer
#    Regression = Linear
#    Binary = Sigmoid
#    Multi-class = Softmax
#    Multi-label = Sigmoid

model = nn.Sequential(
    nn.Linear(8, 12),
    nn.ReLU(),
    nn.Linear(12, 8),
    nn.ReLU(),
    nn.Linear(8, 1),
    nn.Sigmoid()
).to(device)
print(model)
```

```
Sequential(
  (0): Linear(in_features=8, out_features=12, bias=True)
  (1): ReLU()
  (2): Linear(in_features=12, out_features=8, bias=True)
  (3): ReLU()
  (4): Linear(in_features=8, out_features=1, bias=True)
  (5): Sigmoid()
)
```

[20]:
```
next(model.parameters()).is_cuda
```

[20]:
```
True
```

Context

I want to know if GPU can train NN model faster than GPU

Parameters for Investigation

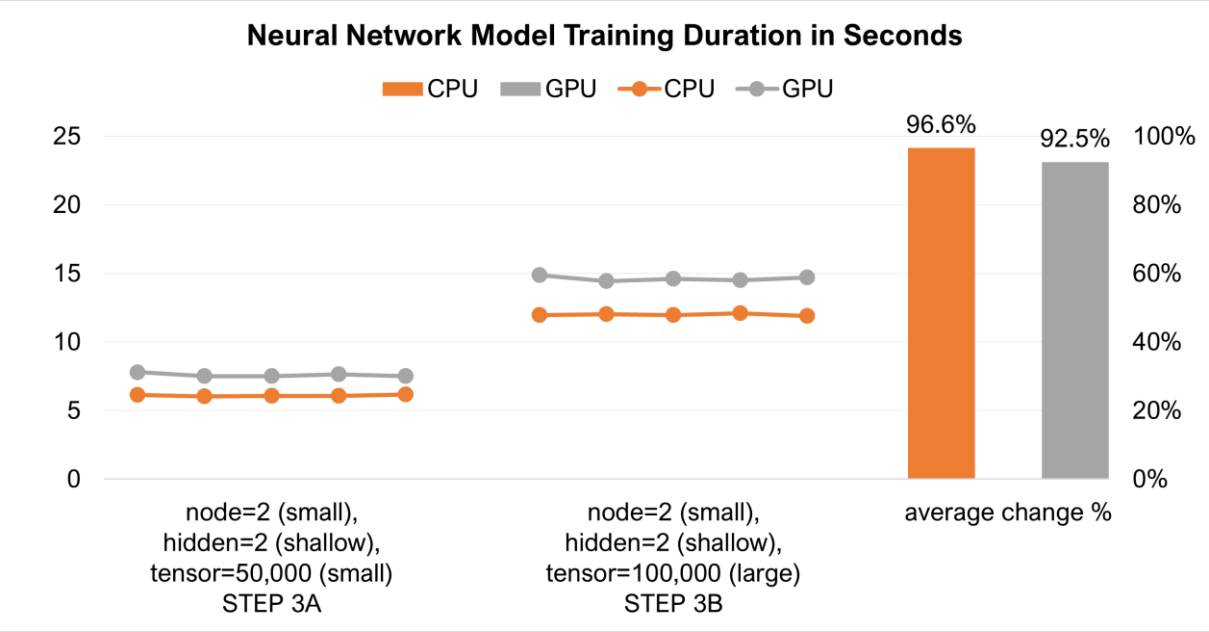
Tensor size

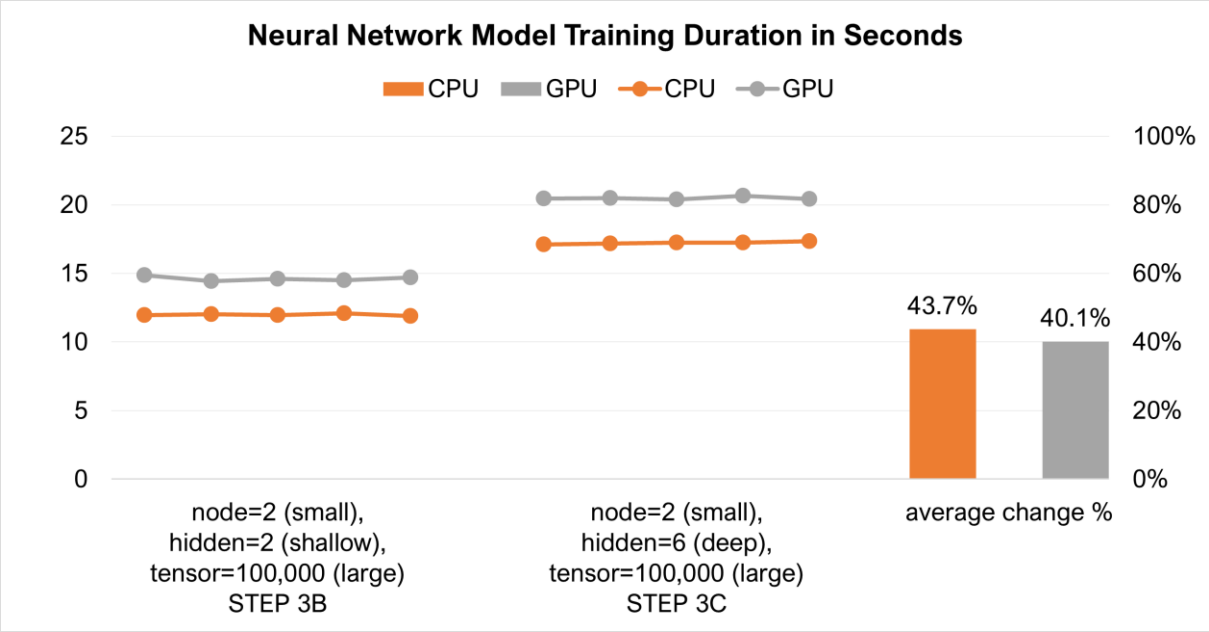
Hidden layers

Node size

Result

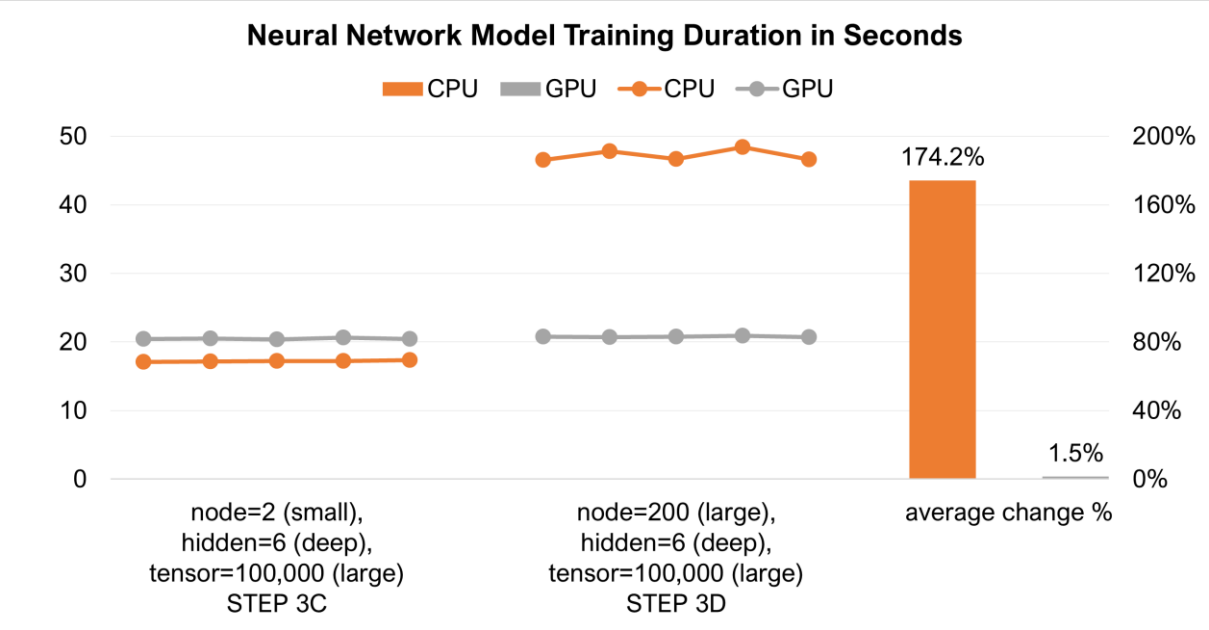
Difference in training duration change due to tensor size alone and hidden layers alone **IS NOT** significant

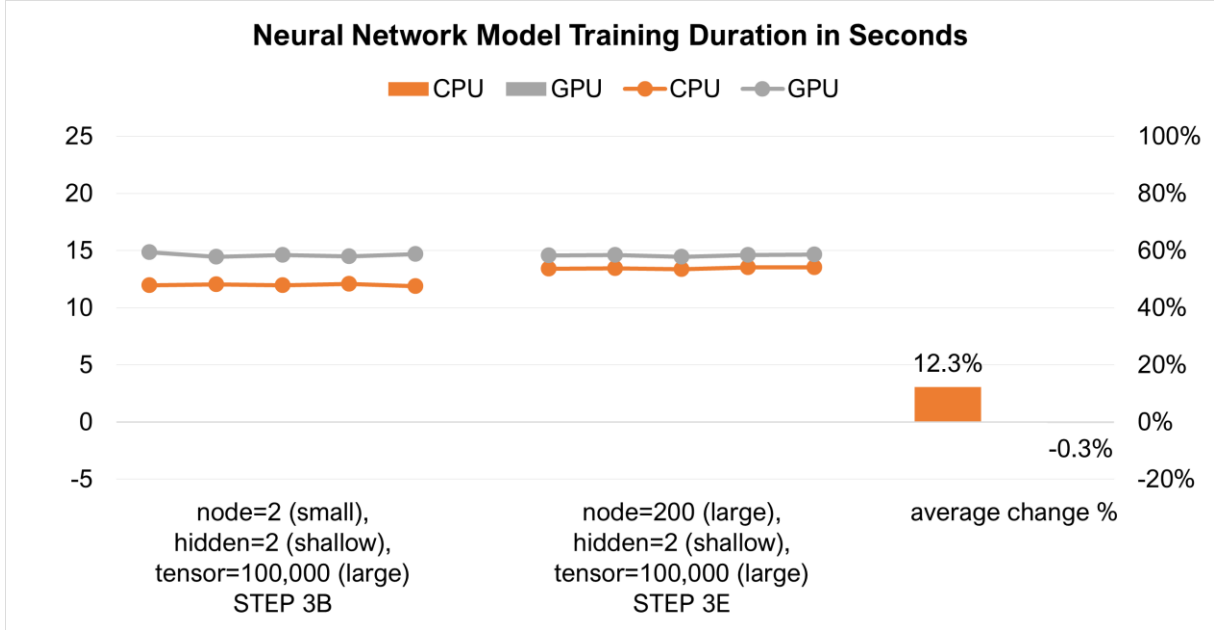




Result

Difference in training duration change due to interaction between node size and hidden layers **IS** significant





Conclusion

Deep NN models with large node size can be trained faster on GPU compared to on CPU

The End

Hope you find this useful

Visit my GitHub for the codes

https://github.com/johnwck/my_da_ds_work/tree/master/my_projects_github_pages/pytorch_neural_network_model_on_gpu