Is my neural network too big to fit into GPU?

Weiguang Guan

SHARCNet/CC
guanw@sharcnet.ca

Is my neural network too big to fit into GPU?

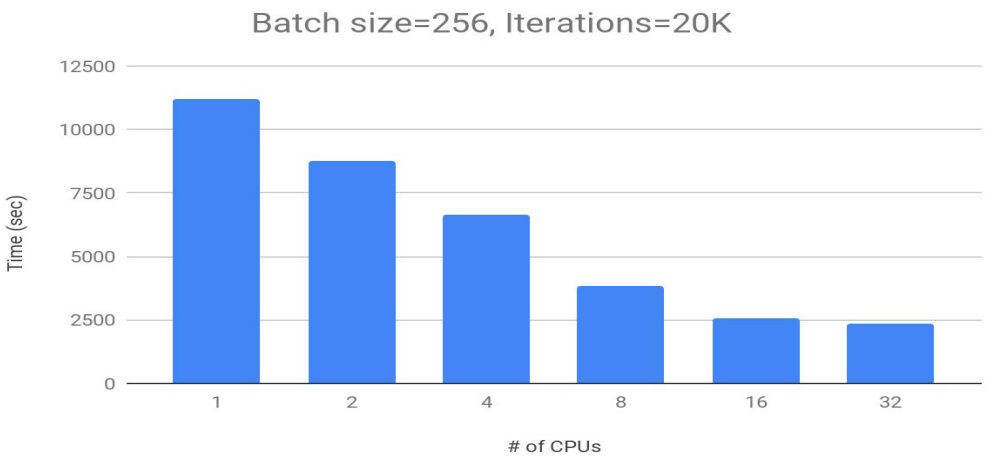
Abstract: It is well known that GPU can significantly accelerate neural network training. However, there are lots of questions around the use of GPU(s), especially for beginners. In this talk, we will dissect a particular convolutional NN and use it as an example to answer these frequently asked questions. We will illustrate how to summarize and visualize the architecture of a NN, from which we will make a coarse estimate of memory requirement. Then, we'll show how to accurately check the GPU memory usage at runtime and provide several advice in case it runs out of GPU memory. The live demo in the talk uses Keras interface on Graham cluster and source code will be provided after the talk.

Background

It’s well known

- AI and deep learning are revolutionary technology
- GPUs can significantly accelerate NN training

Benchmark of multiple CPUs vs single GPU



~150 seconds on nVidia Pascal p100 GPU

1 p100 GPU is ~15 times faster than 16 CPU cores (Intel E5-2683 v4 Broadwell)

Batch size = 256 Iterations = 20000	
# of CPUs	Time (sec)
1	11193.94
2	8757.93
4	6645.16
8	3838.28
16	2559.46
32	2336.07

Background

FAQs about using GPU for deep learning

- Can my NN fit into a single GPU?
- How to check the availability of GPUs?
- How to make sure a training is run on GPU?
- Can I benefit from using multiple GPUs?

Goals

- Summarize and visualize your NN
- Estimate the major factors that affect GPU memory usage
- Check GPU memory usage at runtime
- Suggestions when NN is too big for a GPU

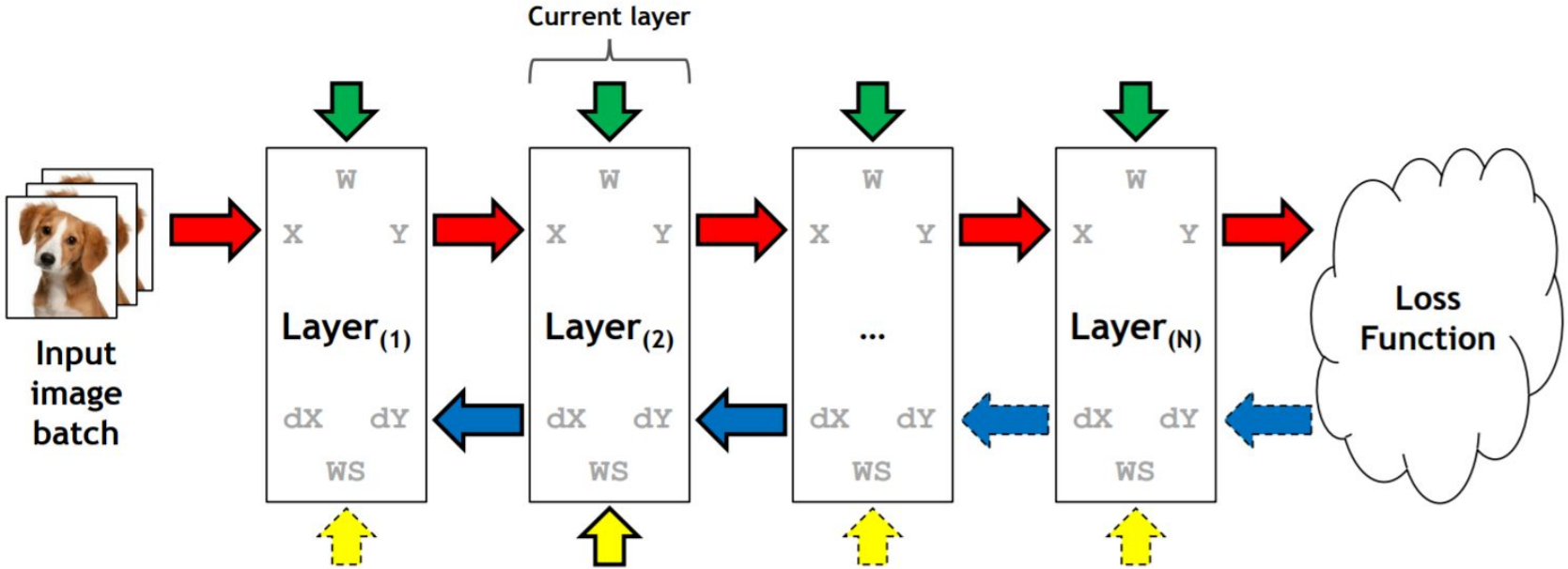
GPUs on CC clusters

Cluster	GPU model	# GPUs per node
Graham	P100-PCIE-12GB	2
Graham	V100-PCIE-16GB	8
Graham	Tesla T4 16GB	4
Cedar	P100-PCIE-12GB	4
Cedar	P100-PCIE-16GB	4
Cedar	V100-PCIE-32GB	4

Cluster	GPU model	# GPUs per node
Béluga	V100-SXM2-16GB	4
Hélios	K20 5GB	8
Hélios	K80 12GB	16
Mist	V100-SXM2-32GB	4

Ref: https://docs.computecanada.ca/wiki/Using_GPUs_with_Slurm

NN Training (feed forward and back propagation)



Source: [Minsoo Rhu et al. 2016]

Estimate GPU memory needed for a NN

Shape Inference

- Memory for NN's parameters (or weights)
- Memory for the outputs of intermediate layers in feed-forward pass
- Memory for derivatives of outputs of intermediate layers and NN's parameters in back-propagation pass
- Misc memory used for temporary variables or cudnn workspace, fragmentation, etc

Inaccurate estimation with average error ~30-50% [Yanjie Gao et al 2020]

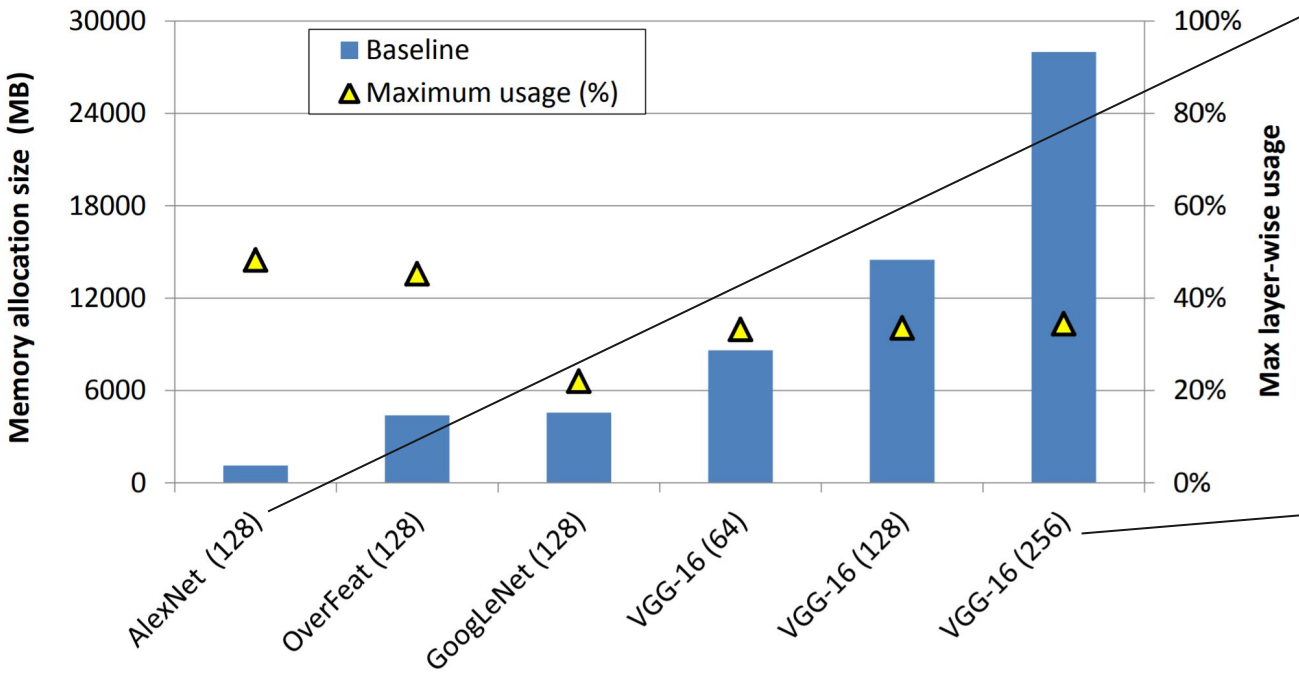
Estimate GPU memory needed for a NN (cont.)

Estimate by comparison with benchmarked NN models, such as

- AlexNet
- GoogleNet
- VGG-16 Net-D
- VGG-16 Net-E (VGG-19)

with different batch sizes in training.

ImageNet winners



AlexNet: 1.1GB

5 convolutional layers + 2 dense layers

VGG-16 Net-E: 28GB

16 convolutional layers + 3 dense layers

Source: [Minsoo Rhu et al. 2016]

Case study: Recognition of handwritten digits

Dissect a vanilla version of NN --- mnist CNN contains

- Convolution
- Pooling
- Drop-out
- Dense

Task 1: Summarize and visualize NN's Architecture

Use Keras APIs to summarize and plot NN's architecture

- Conv2d: 32x(5x5+1) = 832 parameters
- Conv2d: 64x(5x5x32+1) = 51264 parameters
- Dense: 1024x(1024+1) = 1049600 parameters
- Dense: 10x(1024+1) = 10250 parameters

Total = 1,111,946 parameters

Attempt: Estimate GPU memory usage

- **Shape inference alone is not accurate.**
- More sophisticated algorithms have shown promising results [Yanjie Gao et al, “Estimating GPU Memory Consumption of Deep Learning Models”, Microsoft Research, 2020.]

Attempt: Estimate GPU memory usage (cont.)

Memory of parameters:

4 bytes in float type

1111946 x 2 x 4 = 8895568 Bytes = **8.48MB**

For both forward and backward passes

Attempt: Estimate GPU memory usage (cont.)

Memory of intermediate outputs per **single image**:

28x28 + 24x24x32 + 12x12x32 + 8x8x64 + 4x4x64 + 1024 + 10 =

29978

4 bytes in float type

29978 x 2 x 4 = 239824 Bytes = **0.229MB**

For both forward and ~~backward~~ passes

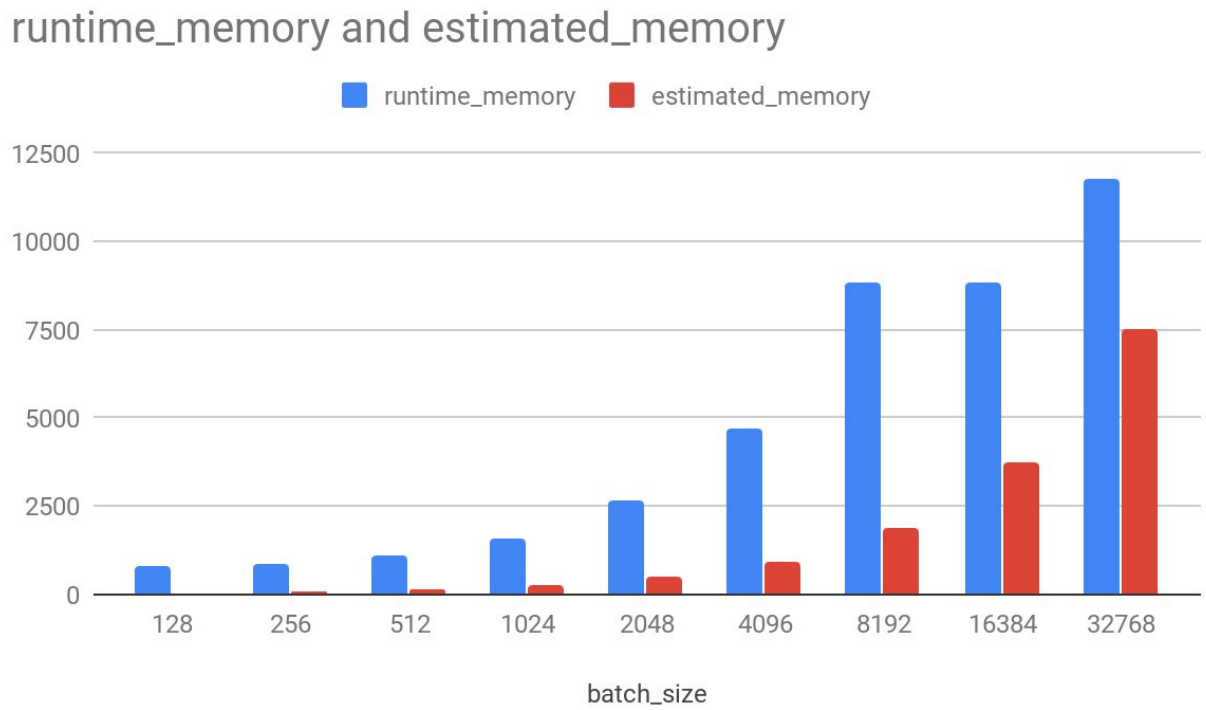
Attempt: Estimate GPU memory usage (cont.)

Total memory = Memory_of_parameters + batch_size x

Memory_of_intermediate_outputs_per_single_image

= 8.48MB + batch_size x 0.229MB

Attempt: Estimate GPU memory usage (cont.)



Task 2: Does my NN training run on GPU?

Need to:

- Pip install tensorflow_gpu
- Use --gres in sbatch or salloc (e.g. --gres=gpu:v100:1)

Ref: https://docs.computecanada.ca/wiki/Using_GPUs_with_Slurm

Task 2: Does my NN training run on GPU? (cont.)

Verify:

- In the output:
  - "... physical GPU (device: 0, name: Tesla V100-PCIE-16GB ..."
  - "... failed call to cuInit: CUDA_ERROR_NO_DEVICE: no CUDA-capable device is detected..."
- In your code, check the availability of GPUs and log the placement of operations (tf.py)
- Monitor GPU usage with command "watch -n1 nvidia-smi"

Task 3: Monitor GPU memory usage at runtime

- Issue command “watch -n1 nvidia-smi” on the same node
- Query inside your python code
  - Use subprocess to run “nvidia-smi” command
  - Use nvidia-ml-py3 package

Task 3: Monitor GPU memory usage at runtime (cont.)

By default, Tensorflow will allocate all GPU memory even if a training needs a small fraction of it.

To dynamically allocate GPU memory as needed, you need to set

tf.config.experimental.set_memory_growth(gpu, True)

Task 3: Monitor GPU memory usage at runtime (cont.)

Use nvidia-ml-py3 package

- pip install nvidia_ml_py3
- Insert the code snippet after one iteration of training

```
import nvidia_smi
nvidia_smi.nvmlInit()

handle = nvidia_smi.nvmlDeviceGetHandleByIndex(0) #GPU0
info = nvidia_smi.nvmlDeviceGetMemoryInfo(handle)

print("Total memory:", info.total, ", Free memory:", info.free, “, Used memory:",
info.used)

nvidia_smi.nvmlShutdown()
```

Task 4: How to tell if it runs out GPU memory?

Crash test by increasing the scale of NN. for example, change

- num_filters from [32, 64] to [320, 640]
- dense_layer_size from 1024 to 10240

You will see

"… Allocator (GPU_0_bfc) ran out of memory … "

in the output

What to do if my model is too big for GPU

- Reduce batch size
- Reduce the scale of NN
- Customize the data in/out of GPU memory
- Use package "TensorFlow large model support (IBM)"
- ~~Wait for new GPUs with larger memory to come~~

Slides/code:

www.sharcnet.ca/~guanw/GIS-2020-09-23/IsmyNNtoobigforGPU.pdf

www.sharcnet.ca/~guanw/GIS-2020-09-23/tf-gpu.tar.gz

Thank you!

Questions?