Parallelizing Optimization Algorithms for Machine Learning

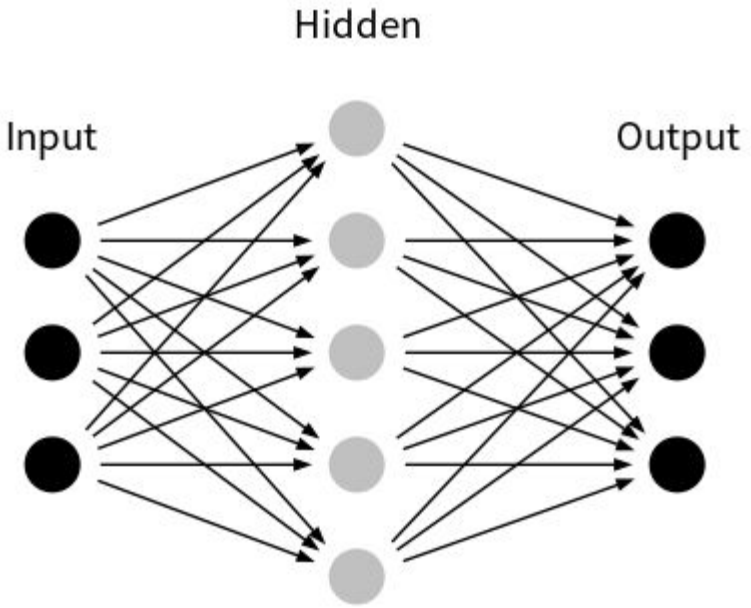
109550042 林律穎
109550065 陳重諺
109652039 林立倫

Introduction

- Neural network implementation with C++
- Parallelize with OpenMP and CUDA
- MNIST
- Back propagation
- Stochastic gradient descent

Problem statement

- Optimization of neural network
- Parallelize back propagation
- Stochastic gradient descent



Proposed solution - Parallelization

- Linear layer
  - Matrix calculation
  - Vector addition
  - $input \times Weight + Bias$
- Activation Function (Sigmoid)
  - Elementwise operation
- Gradient
  - Independant calculation

Challenges encountered

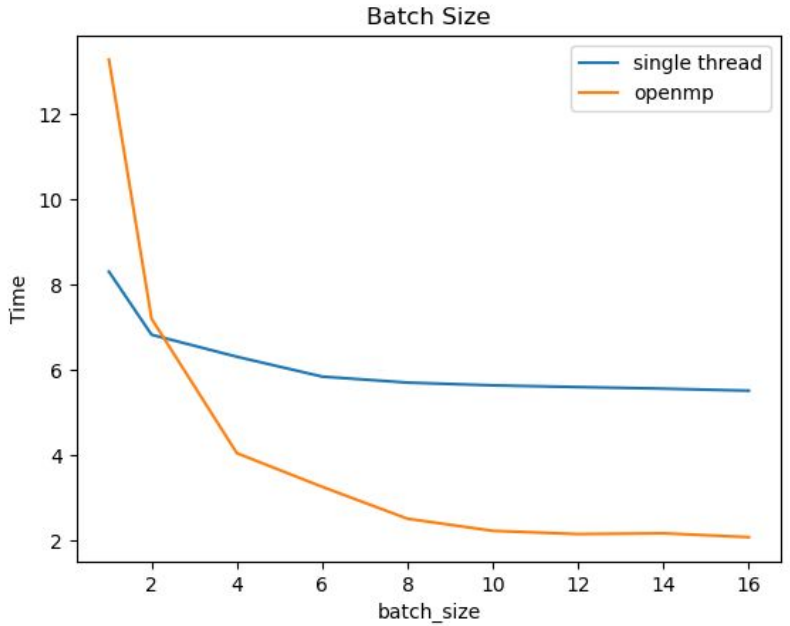
- Memory management
- Race condition
- Runtime Error
- Implementation detail
- Hard to debug
- Compilation

Evaluation Platform

- i7-13700K (limited to 8 hardware threads)
- GeForce RTX 3090
- Ubuntu 22.04.1 LTS
- GCC 11.3.0
- CUDA 11.8

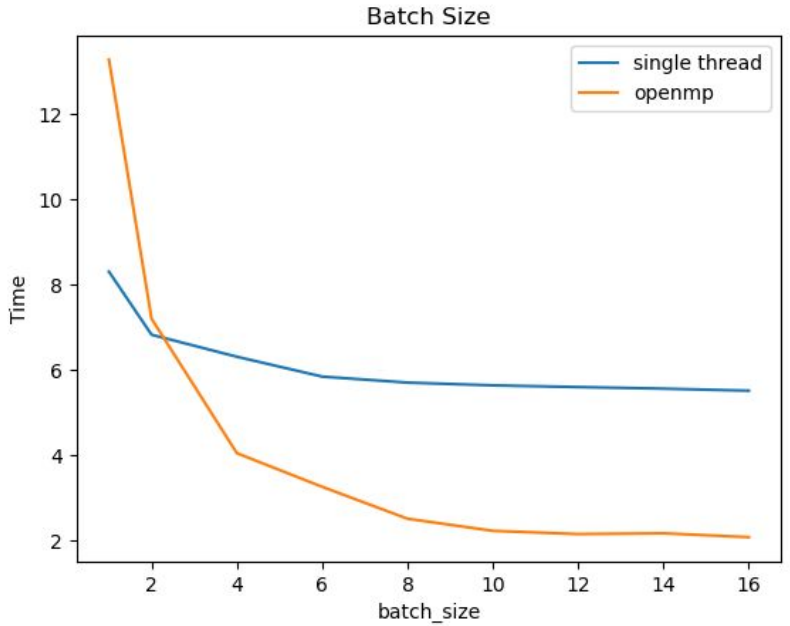
Evaluation

- Batch Size ↔ Time
  - Hidden Dimension = 300
- batch_size is bigger
  - SGD iteration is smaller
  - zero_gred, backward, update
  - single thread time get lower



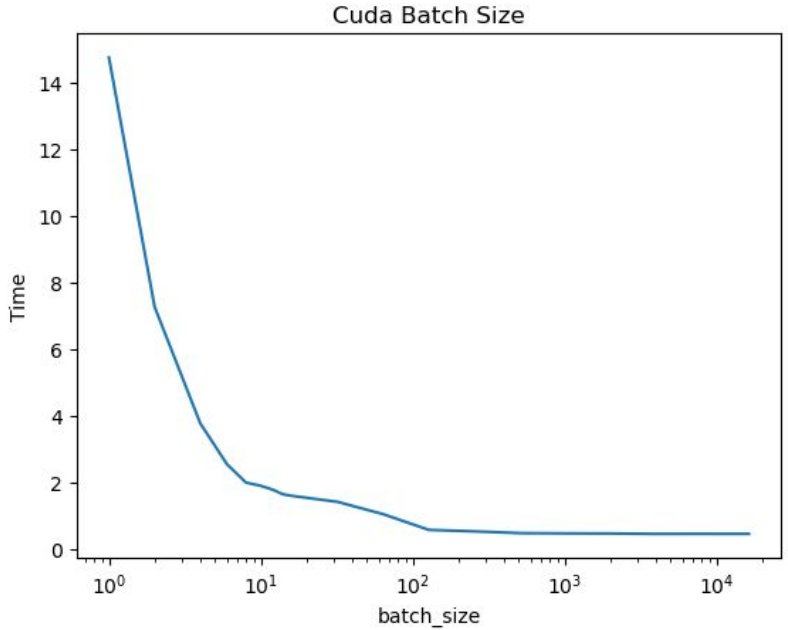
Evaluation

- Parallelized on batch_size
  - batch_size 1 to 8
  - multi thread get parallelized
- Thread creation overhead
  - batch_size 1 and 2
  - lower than single thread
- Limited to 8 hardware threads
  - batch_size > 8, as single thread



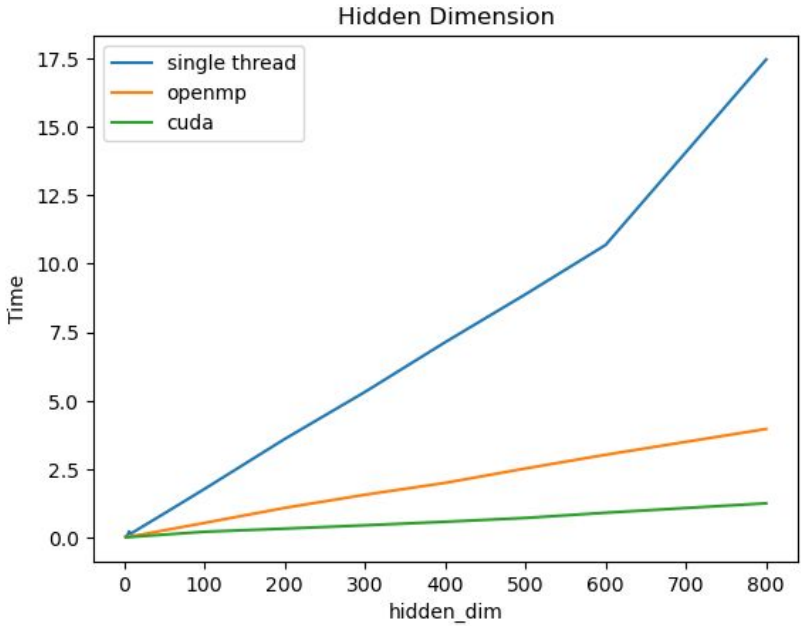
Evaluation

- Parallelize with CUDA
- 10496 CUDA CORES in RTX-3090
- Parallelized on batch_size
  - batch_size 1 to 1e4
  - multi thread get parallelized



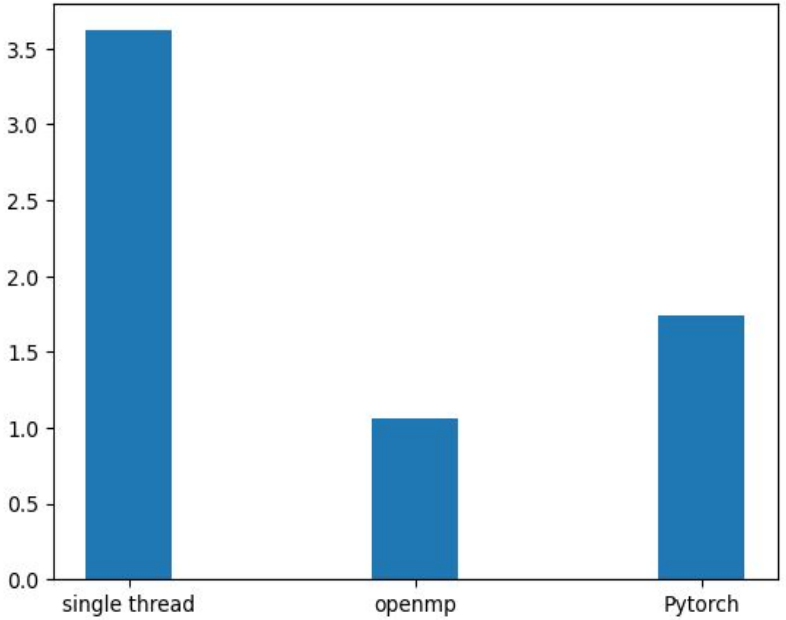
Evaluation

- Hidden Dimension ↔ Time
  - Batch size = 1024
- hidden_dim of NN is bigger
  - computation get larger
  - time get longer for all cases



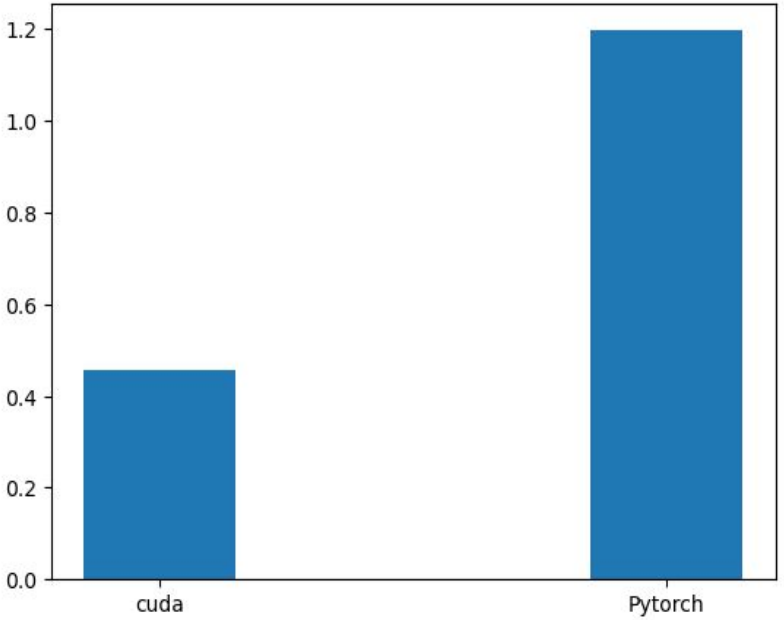
Comparison - PyTorch

- CPU
- Parameters:
  - batch_size = 1024
  - hidden_dim = 300
- 1.7392157264985144 sec



Comparison - PyTorch

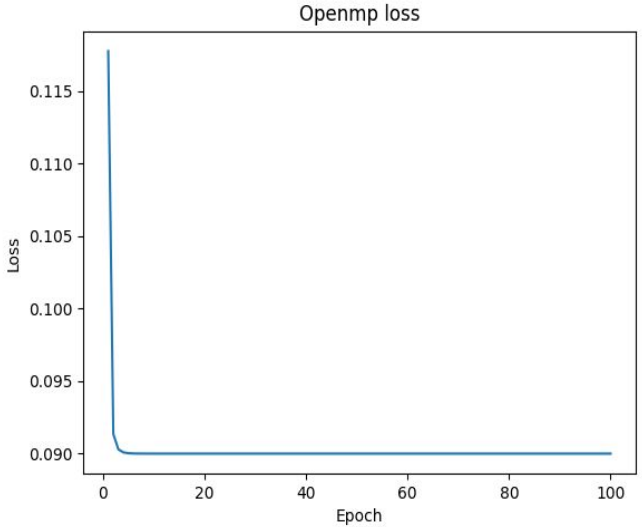
- CUDA
- Parameters:
  - batch_size = 1024
  - hidden_dim = 300
- 1.1973519088700413 sec

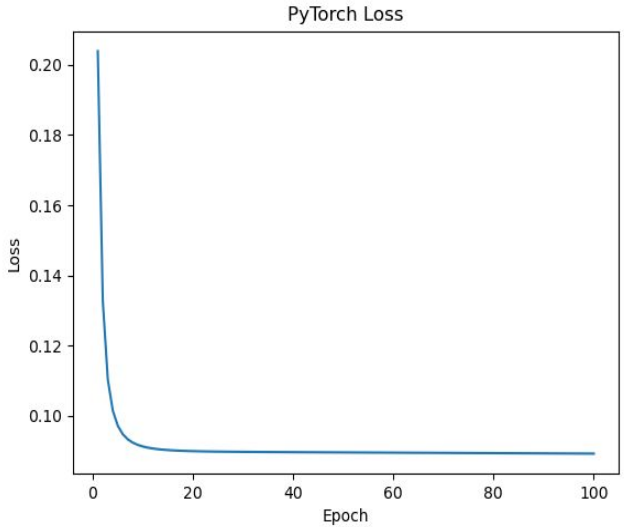


Related work

- PyTorch
- TensorFlow
- Keras

Related work - PyTorch Loss





Contributions of each member

- 林律穎
  - Coding(20%), Script, Design Experiment, Running Experiment, Presentation
- 陳重諺
  - Coding(60%), Environment Setup, Design Experiment, Testing Platform 😍
- 林立倫
  - Coding(20%), Project Structure, Environment Setup, Presentation

Conclusion

- The degree of parallelism highly depend on batch size in our implementation
- Because in our implementation, we mostly use block partition on batch size.
- Block partition on batch size has less chance to create race condition.
- Put all data in device memory to lessen the overhead of memory transmission.
- And let managing memory become easier.
- Use libraries if there are existed ones

Q & A

Thanks for listening