RESEARCH PROPOSAL

⟨ Parallelize Convolutional Neuron Network Layer For Image Classification ⟩

Group name: M team

List of members: 19127330 - Lê Tâm Anh
19127154 - Nguyễn Thế Hưng
19127281 - Trần Minh Thiện

Keywords: Image Classification, CNN, Parallel computing, Image processing kernel, Deep learning.

List of references:

- Documentations, tutorials, technical reports:
- Source codes, pretrained models, demonstrative applications:
  - https://github.com/TheIndependentCode/Neural-Network
  - https://github.com/andreoniriccardo/CNN-from-scratch
- List of books, papers, or web pages:
  - https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939
  - https://akshat0304.github.io/ParallelCNN/?fbclid=IwAR0hqHKLHRrKOv9ne58SImtZQEHrcvpOr29U_kgNSUIDUQJySPw1QIeb_KE
  - https://www.geeksforgeeks.org/introduction-convolution-neural-network/
  - https://medium.com/analytics-vidhya/convolutional-neural-network-cnn-and-its-application-all-u-need-to-know-f29c1d51b3e5

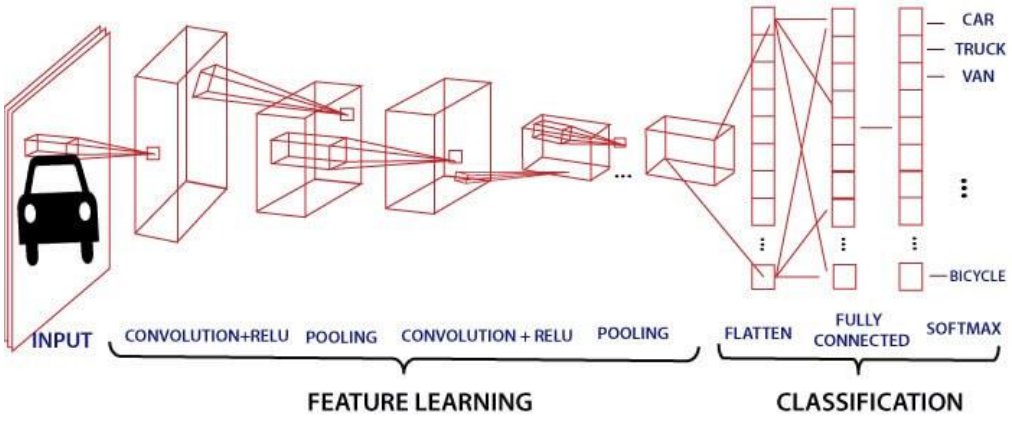
Content

1. **Summary:**

- We are going to build a basic building block for Convolutional neural network (CNN) from scratch and make it work in parallel to increase the speak of the network. First, we will start with a sequential implementation and check how much time it takes to run a sequential CNN, after that we will try to parallelize it by using Numba or some of the library like Cupy, CuDF, … to increase the performance of the network.

2. **Background**

- A Convolutional Neural Network is a type of deep neural network, most applied to image classification, CNN often used to extract the feature from the grid-like matrix dataset. CNN will consist of multiple layers like the input layer, convolutional layer, max-pooling layer and fully connected layer and in CNN we will have to perform multiple dot product between two matrices so it can be counted as a compute-intensive application.



CNN architecture

(nguồn: https://medium.com/analytics-vidhya/convolutional-neural-network-cnn-and-its-application-all-u-need-to-know-f29c1d51b3e5)

- The aspect of the problem might benefit from parallelism is that we can parallelize both the back and forward propagation phases along with improve the speed of some of the function that multiply two matrices together inside the class of CNN to reduce memory footprint and achieve a high speed up at the cost of the lowest possible accuracy drop.

3. The challenge

- Low Arithmetic Intensity: As the computation of the next layer depends on the results of the previous layer, the computational results of each layer need to be shared among threads. Similarly, the back-propagation step also requires results from previous layers. As a result, CNNs inherently have a high communication to computation ratio.
- Memory Limitation: The number of training examples that can be computed in parallel is significantly restricted by the amount of global memory available to the GPU.
- Data Dependencies: The next layer requires the result of the previous layer. That limits parallelism within a layer but not between layers.
- We could optimize the functions in each layer for better memory allocation and communication.

4. Resources

- We will use the code an existing CNN codebase made with Numpy and then optimize it using Numba: https://github.com/andreoniriccardo/CNN-from-scratch
- Google Colab seems to be a good place to for us to borrow the GPU power.

5. Goals And Deliverables

- In this project, we will speed up basis Convolutional Neuron Network layers like Convolution, Max-pooling, SoftMax. By speeding up layers, our goal is making the functions in each layer run faster at least 2 times (>= 2x faster). If the project goes well, our team hopes to apply many methods to speed up the layers as much as possible (maybe about 5 times faster). However, in case our work goes more slowly, we will try to parallelize functions in each layer for better performance time than the basis CNN version (< 2x faster).

- At the seminar presentation, we are looking forward to showing three main parts of our project that will prove our effort to do a good job and the knowledge we gained during this project. Firstly, pseudo code will show the way we parallelize each function and how we handle the memory. Secondly, we would love to run a few parts of the code to make the demo more interactive and prove that our code is right and faster. Finally, it will be the speedup graphs to compare between versions.

- As the basis CNN for image classification takes too much time executing, we hope that the new system can run faster than the basis version to speed up the image classification model using CNN at least 2x faster. With the improvement of the system, each layer of the CNN can parallelize the matrix computing, which we think is the most time-consuming function in the CNN structure. We will try our best to improve the algorithms and deliver three versions of the new system with parallel computing functions for each layer.

Weekly schedule:

	Week 01 (from 23/6 to 29/6)	Week 02, 03 (from 30/6 to 13/7)	Week 04, 05 (from 14/7 to 27/7)	Week 06 (from 28/7 to 4/8)
Lê Tâm Anh	Researching proposal	Parallelize Max-Pooling Layer (Version 1)	Parallelize Max-Pooling Layer (Version 2)	Parallelize Max-Pooling Layer (Version 3)
Trần Minh Thiện	Researching proposal	Parallelize SoftMax Layer (Version 1)	Parallelize SoftMax Layer (Version 2)	Parallelize SoftMax Layer (Version 3)
Nguyễn Thế Hưng	Researching proposal	Parallelize Convolution Layer (Version 1)	Parallelize Convolution Layer (Version 2)	Parallelize Convolution Layer (Version 3)