Setting up Deep Learning Analysis on GPU-based Virtual cluster for Traffic Monitoring Project

UCSD
Nadya Williams
Phil Papadopoulos
Shava Smallen

NCTU, Taiwan
Jimmy Liu
Yu-Chuan Huang

Deep Learning

"**Deep learning** (a.k.a. **deep structured learning** or **hierarchical learning** is a part of broader family of machine learning methods based on learning data representations, as opposed to task specific algorithms."

-- Wikipedia

"**Deep learning** is the fastest-growing field in machine learning. It uses **many-layered Deep Neural Networks (DNNs)** to learn levels of representation and abstraction that make sense of data such as images, sound, and text."

https://developer.nvidia.com/deep-learning

Ignore it and you are left behind...

Deep Learning sides

- Deep learning is present every day in global headlines

- Fastest growing field of machine learning
- NVIDIA GPUs – engine of deep learning
- Used in research and industry to help to solve big data problems:
  - Image recognition
  - Speech recognition
  - Vehicle, person or landmark identification
  - Many others

- Complex to build, impractical to run in a large number of real world scenarios
- Cloud computing remains too unreliable and insecure for many scenarios.
  - Human safety
  - Well fare and medical applications
- Cloud computing costs of deep learning are too high to scale
  - Bandwidth -10's of Pb per day
  - Latency: safety-critical services
  - Confidentiality: private cloud/ storage
  - Connectivity: 50% of populated world < 8mbps

GPU-based virtual cluster

- GPU virtualization can greatly improve application performance
- Suitable for deep learning, simulations molecular dynamics codes, visualization, etc.
- Automate building powerful deep learning AI platform with an access to a dedicated GPU:
  - Build a ROCKS virtual cluster
  - Connect GPU to a virtual machine through the hypervisor
  - Allocate full GPU capability to a single virtual machine
  - Add required software (rolls)

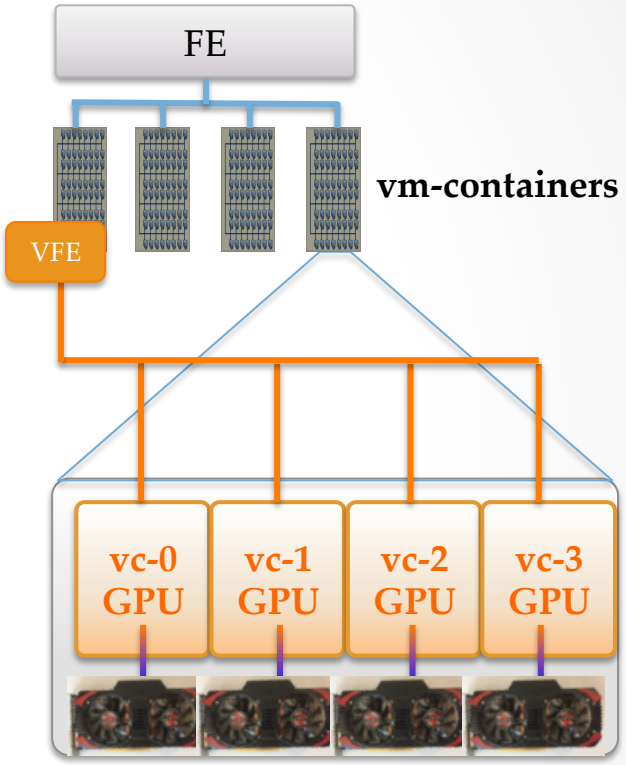
Step by Step on a physical FE

- Add rolls:
  - **cuda**: NVIDIA toolkit and driver
  - **gpupt**:
    - Contains commands to manipulate GPU assignment (attach, detach, list info, status):

    rocks run host vm-container-0-1 "gpupci –d all"

    rocks add host gpu compute-0-1-0 gpupci pci_0000_02_00_0

    rocks list host gpu
    - Provide correct parameters from PCI bus to the hypervisor (address, name)
    - Enables creation of a portion of VM xml description for the GPU
    - Recompile qemu-kvm that understands **–enable-kvm** flag
- Add GPU cards
  - GeForce GTX 1070 (consumer grade)
- Prepare physical host for pass-though
  - Enable VT-D extensions in BIOS
  - Activate Vt-d extensions in the kernel
  - Verify GPU cards work with cuda roll,
  - Detach GPUs
- Create a virtual cluster
  - Run virtual FE anywhere
  - Run virtual compute nodes on GPU-enabled vm-container
  - A Single GPU card is assigned to a virtual host:



Virtual cluster

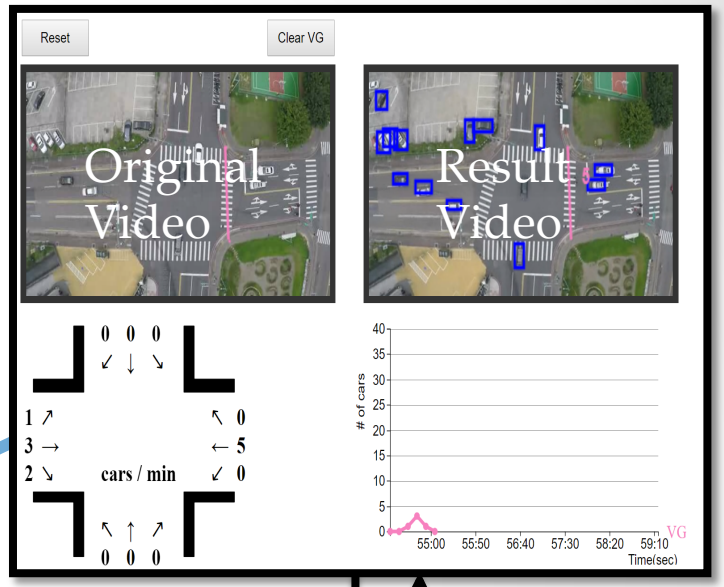
- Rocks 7.0 (based on CentOS Linux 7.4.1708)
- NVIDIA drivers 384.90 and cuda toolkit 8.0.61
  https://github.com/nbcrrolls/cuda
- GPU cards
  - GeForce GTX 1060
  - GeForce GTX 1070
  - Tesla K20x
  - Tesla C2075
- Add software
  - **Cuda roll**
  - **AI roll**: OpenBLAS setting modules
  - **Conda3** provides (~3Gb)
    - Python, ipython
    - Caffe and pycaffe
    - Jupyter
    - Pandas …
  - **Tensorflow roll** (20+ packages)
    - cuDNN
    - Tensorflow …
- Create a user and send login info

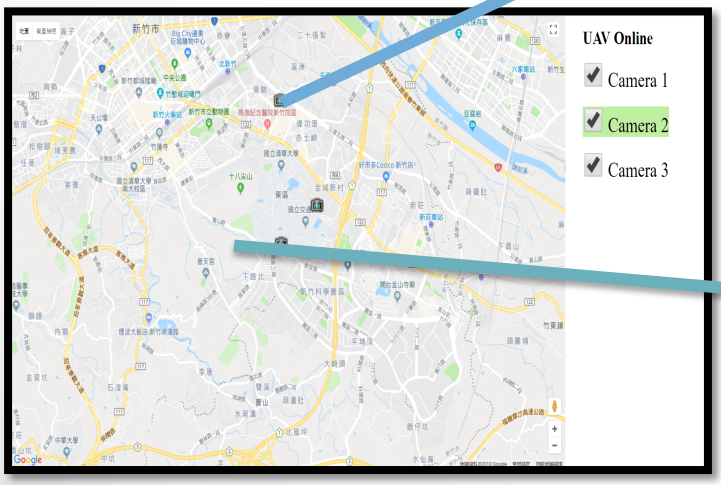
NVIDIA inserted a "bug" into its drivers ~340+ version

- Detects hypervisor and disallows a GPU card use
- Your mileage is different depending on the OS and the specific card used
  - driver 384.90 ✔
  - driver 390.25 ✘

How it works

- When user clicks the camera icon on map, it will pop-up new page
- This new page will play the record video from the streaming server
- It will send the streaming URL to deep learning machine to request DL analysis and get the result back
- After get the result, it draw the tracking and statistic graph





DL machine 1

DL machine 2

Issues

- Main issue: Synchronize problem
  - I hope to let whole process run in real time so that user can easily compare the original video and result video, but my deep learning program (I use Faster R-CNN) takes too much time. It can't play the original video and result video at the same time.
- Resource problem
  - Each page have to use one dedicated computer to support.
  - I think it can be solved by using Virtual GPU cluster :)

TrackNet demo

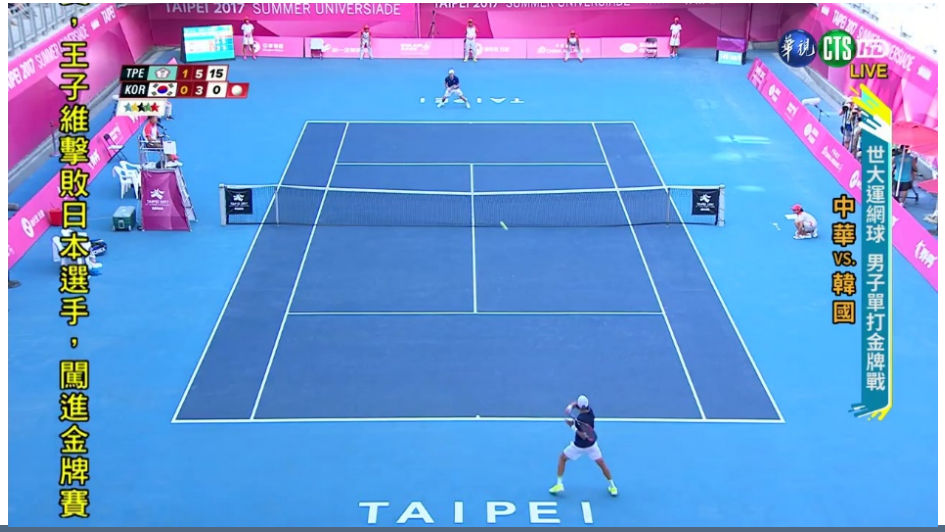
Input Data

- The input data is video mp4 file with movie frames and a model

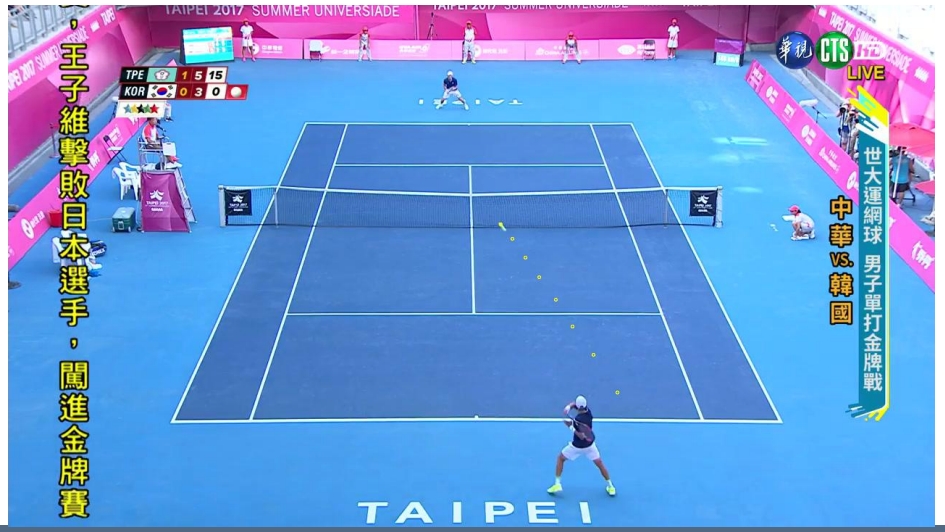
Output Data

- The deep learning program computes the moving path of the ball and generates new image for each frame
- These new images are used to make the result output video mp4 file.

Input



Output



Simulation Result

	Frame 1 (748k)	Frame 2 (284k)
run time on GPU node	real 3m1.054s user 4m12.571s sys 0m38.955s	real 0m39.972s user 0m40.167s sys 0m7.135s
scp time GPU node -> remote server	real 0m4.779s user 0m0.078s sys 0m0.029s	real 0m3.841s user 0m0.045s sys 0m0.022s
rendering time on a remote server	real 0m5.005s user 0m0.036s sys 0m0.012s	real 0m5.926s user 0m0.035s sys 0m0.011s

Links

- Simulation result
  http://140.113.215.114/results/TrackNet/CompareTrackNet/

- cuda roll https://github.com/nbcrrolls/cuda
- gpupt roll https://github.com/pragmagrid/gpupt
- tensorflow roll https://github.com/pragmagrid/tensorflow
- NVIDIA Deep learning AI
  https://www.nvidia.com/en-us/deep-learning-ai/

Thank you!

Questions?