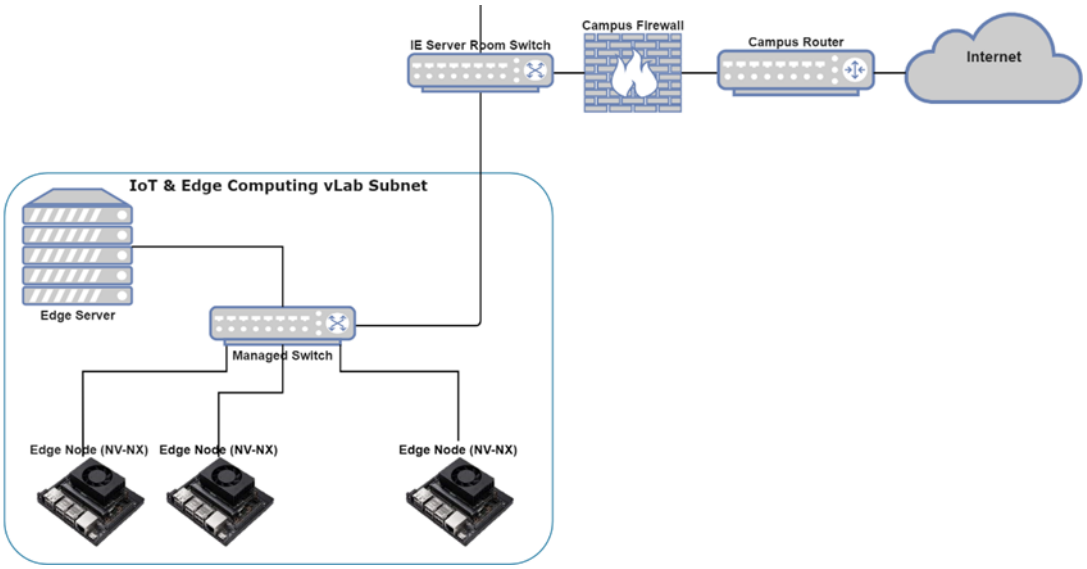
IEMS 5709 IoT & Edge Computing Term Project

Platform



This term project shall be performed on the Edge Nodes (NVidia Jetson NX Developer Kits) and the Edge Server (SuperMicro 4124GS-TNR with dual AMD EPYC 7313 CPUs) of the AIoT & Edge Computing Virtual Laboratory of the CUHK Information Engineering Department. Each team will be assigned an Edge Node and a virtual machine with 4 CPU cores, 16GB RAM, 64 GB disk space and one network port.

Unfortunately, the INC Smart Lamppost subnet has not been fully deployed; hence, we cannot connect our Edge Nodes to those lampposts.

Software

In your projects, students are encouraged to use the following computer vision and deep learning (DL) analytics programs introduced in this course:

1. NVidia DeepStream
2. OpenDataCam

You may also use pre-trained DL models available on NVidia NGC Catalog.

For students with advanced background, you may consider using Ultralytics YOLOv5 or other state-of-art DL inference programs. However, please note that this course focuses on software containerization and orchestration for Edge Computing instead of deep learning.

You are also encouraged to use the following software containerization and orchestration tools to develop and manage your software containers:

1. Docker Build, Docker Run, Docker Login, Docker Push and Docker Save with your Docker Hub account
2. Kubernetes Lens IDE

Data Sets

The Head TA, ZHAO Zhihe (赵之赫), has uploaded the following five data sets onto the Edge Server and stored them under ~/Videos directory.

1. **The Sherbrooke Video** was filmed at the Sherbrooke/Amherst intersection in Montreal and the camera was located a couple of meters above ground. Since the camera is not very high, there are numerous occlusions between the road users. The video resolution is 800x600. For the evaluation, a 1001 frames (30 fps) part of the video was chosen with 15 cars and 5 pedestrians. The scene contains 7 objects simultaneously. Video was kindly provided by Marilyne Brosseau, Jean-Simon Bourdeau and Nicolas Saunier. The video was annotated by Jean-Philippe Jodoin

2. **The Atrium Video** was filmed at École Polytechnique Montréal. It offer a view from the inside of the building and we can see pedestrians moving around, dropping objects et crossing each others. This movie is intersecting since it allows to evaluate the performance on pedestrians. The movie resolution is 800x600. For the evaluation, the 4540 frames (30 fps) of the movie were annotated. The video was kindly provided by Atousa Torabi and Guillaume-Alexandre-Bilodeau. The video was annotated by Jean-Philippe Jodoin.

3. **The St-Marc Video** was filmed at the intersection of Saint-Marc and Maisonneuve in Montréal. Numerous pedestrians move as a group for the whole sequence. The video resolution is 1280x720. For the evaluation, as sequence of 1000 frames was selected (30 fps) with 7 vehicles, 2 bicycles and 19 pedestrians. The scene contains up to 14 objects simultaneously. The video was kindly provided by Mohamed Gomaa Mohamed and Nicolas Saunier. The video was annotated by Jean-Philippe Jodoin.

4. **SPHAR: Surveillance Perspective Human Action Recognition Dataset**

   SPHAR is a video dataset for human action recognition. Its main purpose is to support research in the application area of analyzing activities on public places. In this domain, most cameras will share a similar mounting angle and perspective, which we will call the surveillance perspective from now on. In SPHAR, all videos are shot from this or a similar perspective. The videos have been aggregated from multiple sources, converted to a consistent file type (H265 HEVC .mp4), cutted and cropped (spatio-temporally) to contain only one action at a time and last but not least sorted into 14 action classes.

5. **MIT-BIH Arrhythmia (ECG) Database**

   The MIT-BIH Arrhythmia Database contains 48 half-hour excerpts of two-channel ambulatory ECG recordings, obtained from 47 subjects studied by the BIH Arrhythmia Laboratory between 1975 and 1979. Twenty-three recordings were chosen at random from a set of 4000 24-hour ambulatory ECG recordings collected from a mixed population of inpatients (about 60%) and outpatients (about 40%) at Boston's Beth Israel Hospital; the remaining 25 recordings were selected from the same set to include less common but clinically significant arrhythmias that would not be well-represented in a small random sample.

   The recordings were digitized at 360 samples per second per channel with 11-bit resolution over a 10mV range. Two or more cardiologists independently annotated each record; disagreements were resolved to obtain the computer-readable reference annotations for each beat (approximately 110,000 annotations in all) included with the database.

This directory contains the entire MIT-BIH Arrhythmia Database. About half (25 of 48 complete records, and reference annotation files for all 48 records) of this database has been freely available here since PhysioNet's inception in September 1999. The 23 remaining signal files, which had been available only on the MIT-BIH Arrhythmia Database CD-ROM, were posted here in February 2005.

You may choose to use one of these data sets in your experiment. *You can also use your own data sets found on the web*. Please clearly describe the data sets you used.

Unfortunately, due to privacy concerns, we are not allowed to use the thermal images captured by the Smart Lampposts. Those files have been deleted from the Edge Server.

Procedures

This project will be a group effort. Each team only needs to submit *one* short project report and demonstrate their work during the regular class session on Saturday, April 30th.

In this project, each team needs to perform the following functions:

1. Deploy a containerized deep learning (DL) application from your virtual machine (VM) in the Edge Server onto your Edge Node.

   You may use OpenDataCam, YOLOv5 or other applications. Please wrap them into containers if they were not yet containerized.

2. Use Kubernetes to manage the containerized applications and run it on your Edge Node.

   You may use Kubernetes Lens to create and edit the Kubernetes manifest in YAML.

3. Choose appropriate data set(s) and feed them to your application to perform inference.

   Since this is a project, please perform useful functions using your application e.g., counting the cars crossing a certain intersection or identify different object types. Try to make it a useful application.

4. Debug the execution of the application and monitor its performance including CPU / memory usage and other key parameters.

5. Demonstrate the proper execution of your application during the project demonstration session on Saturday, April 30th.

Report

Submit a short project report including a brief description of the purpose of your applications, the program, tools, and data sets you used.

Please include the following records in this report:

- The Kubernetes manifests for configuring and managing your containerized application.
- The screen shots of the proper execution of your application.
- The resource usage and performance of your application.
- The output of your application.