Demo AI on Chips: Real-time Object Detection on Jetson Nano

12110916 李浩宇 12110644 周思呈 12110649 毛晨羊 12112328 郑祖彬

2024.12.26

Content

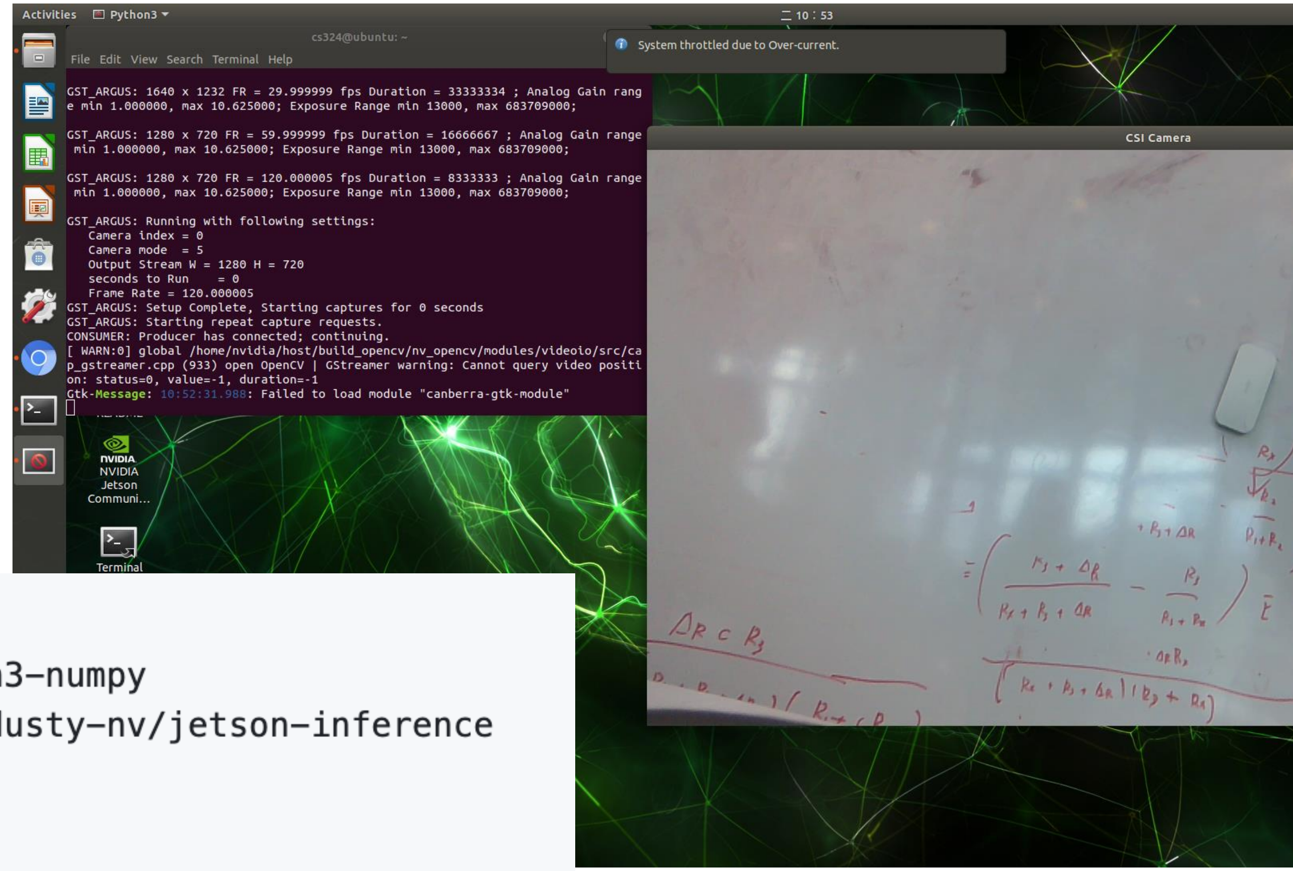
- Introduction
- Demonstration
- Analysis

Introduction

- Target：Real-time Object Detection
- Code Repo：https://github.com/dusty-nv/jetson-inference
- Model：DetectNet, MobileNetv2-SSD

Setup

- Camera setup
- Build the project from source



```
GST_ARGUS: 1640 x 1232 FR = 29.999999 fps Duration = 33333334 ; Analog Gain rang
e min 1.000000, max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 59.999999 fps Duration = 16666667 ; Analog Gain range
 min 1.000000, max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: 1280 x 720 FR = 120.000005 fps Duration = 8333333 ; Analog Gain range
 min 1.000000, max 10.625000; Exposure Range min 13000, max 683709000;

GST_ARGUS: Running with following settings:
   Camera index = 0
   Camera mode  = 5
   Output Stream W = 1280 H = 720
   seconds to Run    = 0
   Frame Rate = 120.000005
GST_ARGUS: Setup Complete, Starting captures for 0 seconds
GST_ARGUS: Starting repeat capture requests.
CONSUMER: Producer has connected; continuing.
[ WARN:0] global /home/nvidia/host/build_opencv/nv_opencv/modules/videoio/src/ca
p_gstreamer.cpp (933) open OpenCV | GStreamer warning: Cannot query video positi
on: status=0, value=-1, duration=-1
Gtk-Message: 10:52:31.988: Failed to load module "canberra-gtk-module"
```

```
sudo apt-get update
sudo apt-get install git cmake libpython3-dev python3-numpy
git clone --recursive --depth=1 https://github.com/dusty-nv/jetson-inference
cd jetson-inference
mkdir build
cd build
cmake ../
make -j$(nproc)
sudo make install
sudo ldconfig
```

DetectNet

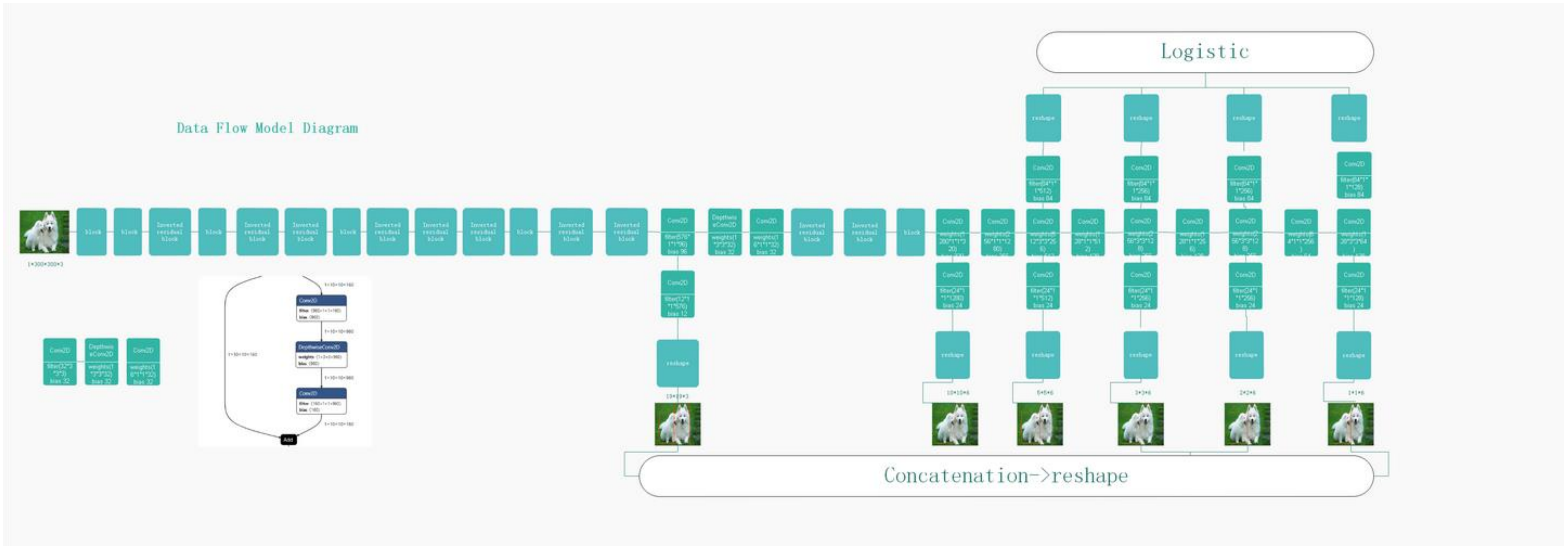
- A deep learning-based object detection framework.
- Specifically designed for efficient and real-time detection of objects within images or video streams.
- A part of NVIDIA's Deep Learning framework ecosystem and works seamlessly with NVIDIA GPUs, including the Jetson platform.
- Leverages convolutional neural networks (CNNs) to perform both object localization and classification in one unified architecture.
- Effective for applications like autonomous vehicles, robotics, and video surveillance.

DetectNet

- **Efficiency on Edge Devices** Jetson Nano is a small, low-power device designed for edge AI applica-
- tions. DetectNet's lightweight and efficient architecture is well-suited for this platform, as it provides
- real-time performance without requiring extensive computational resources.
- **Real-Time Object Detection** Many applications of Jetson Nano, such as robotics, smart cameras, and
- IoT devices, require real-time object detection. DetectNet's single-shot detection approach ensures
- fast and accurate predictions, enabling real-time response.
- **Optimized for NVIDIA GPUs** DetectNet is tailored to leverage NVIDIA GPUs, including the em-
- bedded GPU on Jetson Nano. This ensures maximum performance by utilizing CUDA cores for
- parallel processing and accelerating inference tasks.
- **Ease of Training and Deployment** DetectNet can be retrained using NVIDIA's Transfer Learning
- Toolkit, allowing developers to fine-tune the model on specific datasets. It also integrates seamlessly
- with NVIDIA′ s TensorRT for optimized inference performance on Jetson Nano.

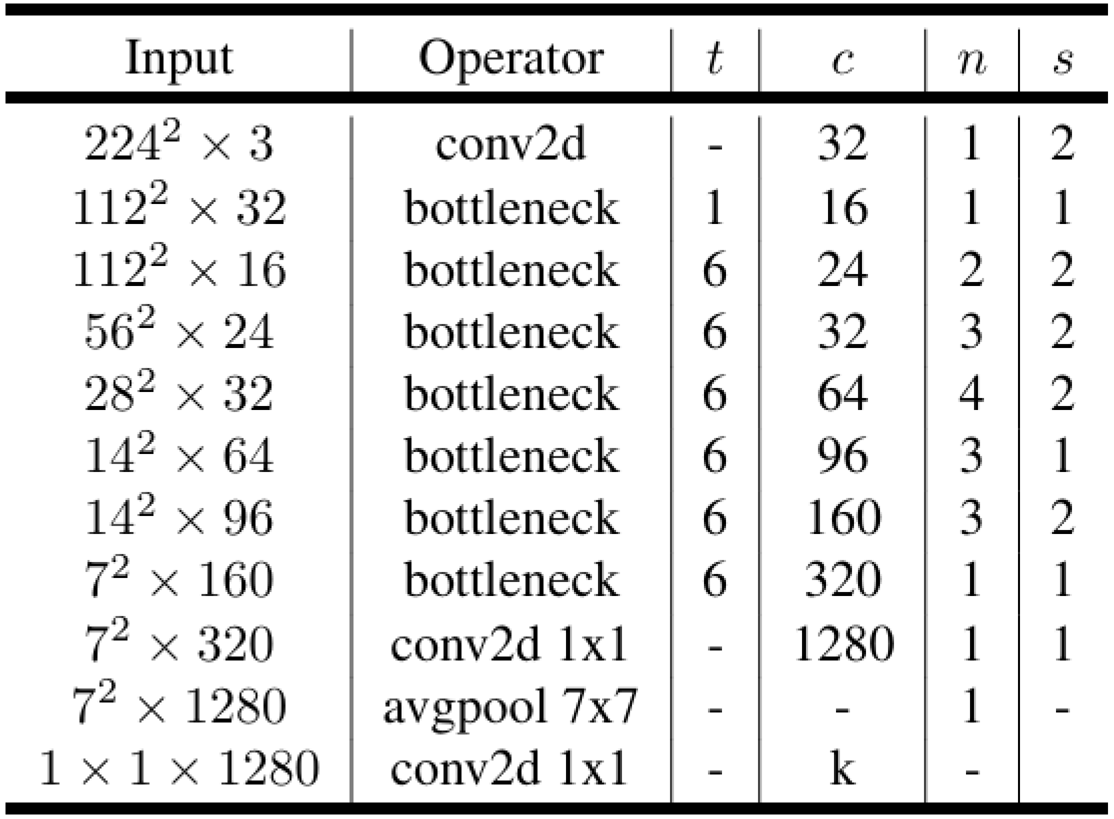
MobileNetv2-SSD

- Based on MobileNetV2, taking the idea of SSD.

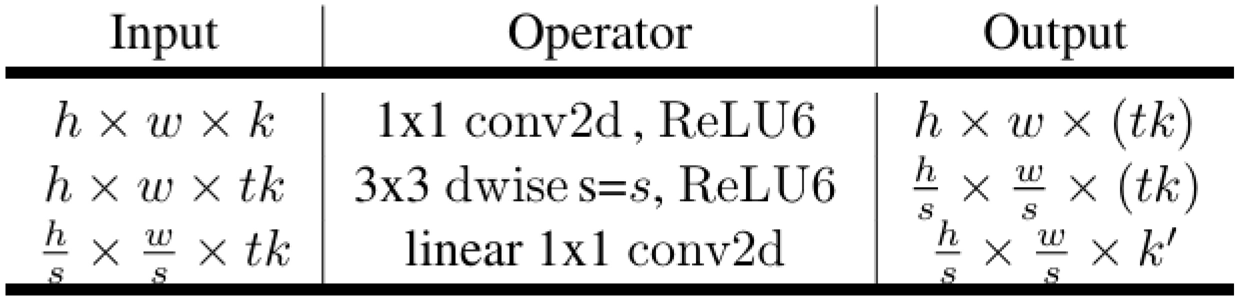


MobileNetv2-SSD

MobileNetv2-SSD



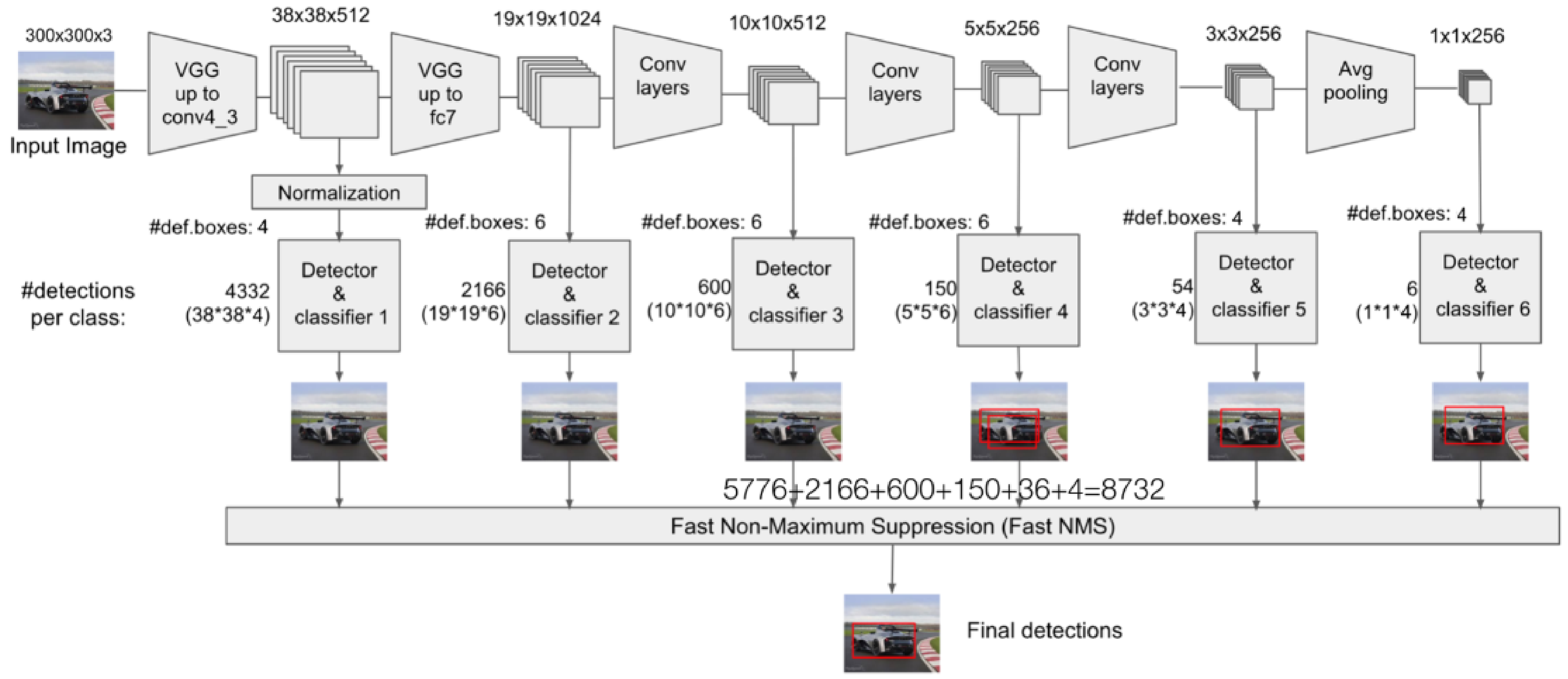
Input	Operator	t	c	n	s
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	



Input	Operator	Output
$h \times w \times k$	1x1 conv2d , ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3x3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1x1 conv2d	$\frac{h}{s} \times \frac{w}{s} \times k'$

Structure of MobileNetv2

MobileNetv2-SSD

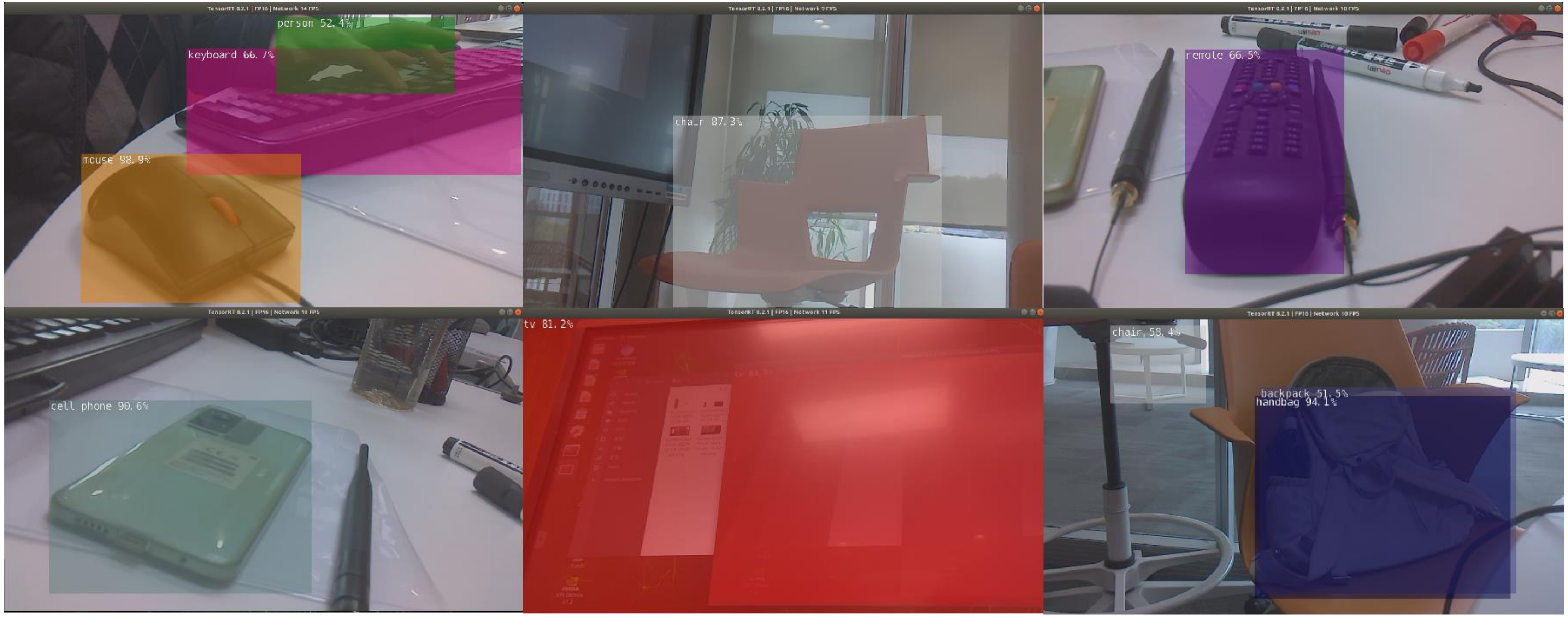


Orginal SSD based on VGG

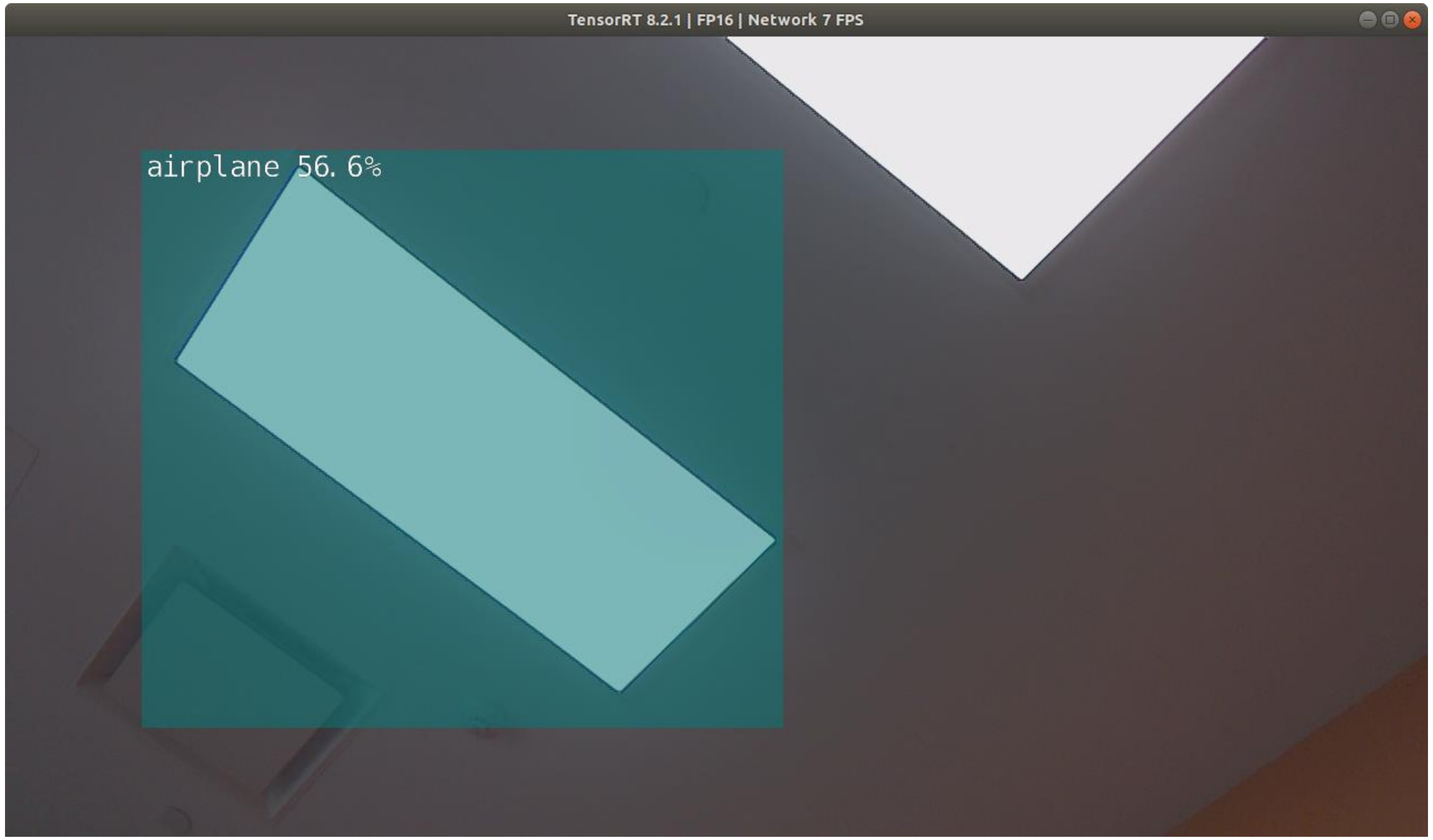
MS COCO (Microsoft Common Objects in Context) dataset

- A large-scale object detection, segmentation, key-point detection, and captioning dataset.
- Consists of 328K images of 91 categories.

Demonstration



Demonstration

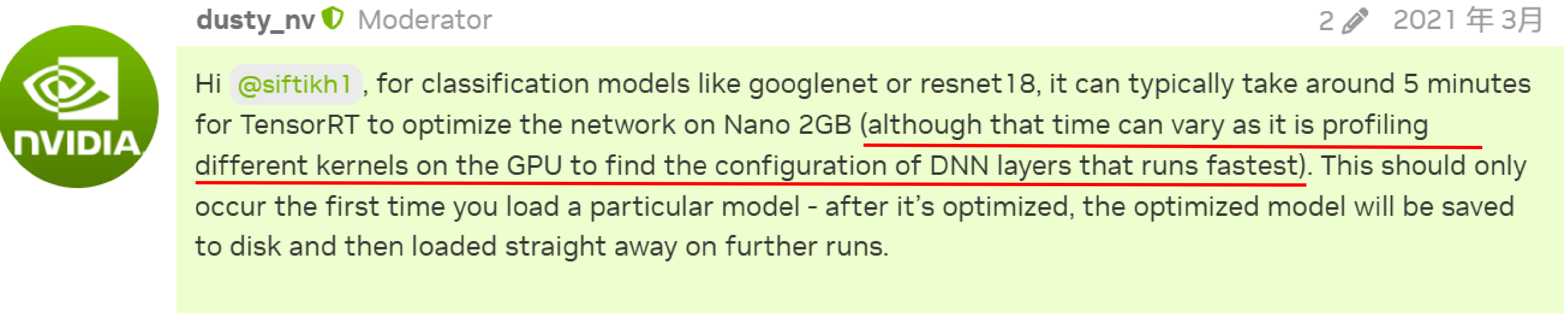


Analysis

- When deploying the model for inference, we noticed an interesting phenomenon:
  - the first time we ran the model MobileNetv2-SSD, it seemed that TensorRT took a few minutes (i.e., 4 ~ 6 minutes) to optimize the network.
  - Fortunately, this optimized network file was then cached to disk, so later runs using the model loaded faster (i.e., 20 ~ 30 seconds).

Analysis

- Official Response



dusty_nv Moderator 2 2021 年 3月

Hi @siftikh1 , for classification models like googlenet or resnet18, it can typically take around 5 minutes for TensorRT to optimize the network on Nano 2GB (although that time can vary as it is profiling different kernels on the GPU to find the configuration of DNN layers that runs fastest). This should only occur the first time you load a particular model - after it's optimized, the optimized model will be saved to disk and then loaded straight away on further runs.

Analysis

- TensorRT
  - includes an inference runtime and model optimizations that deliver low latency and high throughput for production applications.
  - benefits:
    - Speed Up Inference by 36X
    - Optimize Inference Performance

Analysis

- TensorRT Model Optimizer
    - It compresses deep learning models for downstream deployment frameworks like TensorRT to efficiently optimize inference on NVIDIA devices.
    - Post-training quantization (PTQ)
    - Sparsity

Thanks!