Image classification inference

1.ImageNet classifies images

There are various types of deep learning networks available, including recognition, detection/localization, and semantic segmentation. The first deep learning feature we emphasize in this tutorial is image recognition, which uses classification networks trained on large datasets to recognize scenes and objects.

The imageNet object accepts input images and outputs probabilities for each category. After training on the ImageNet ILSVRC dataset of 1000 objects, the Google Net and ResNet-18 models were automatically downloaded during the construction step. For other classification models that can be downloaded and used, we provide sample programs in C++and Python

Using the ImageNet program on Jetson:

Firstly, let's try using the imagenet program to test imagenet recognition on some sample images. It loads one or more images, uses TensorRT and imageNet classes for inference, then overlays the classification results and saves the output image. This project provides sample images for your use in the images/directory.

After building the project, please ensure that your terminal is located in the aarch64/bin directory:

```
cd jetson-inference/build/aarch64/bin
```

Next, let's use the imagenet program to classify the sample images, using C++or Python variants. If you are using a Docker container, it is recommended to save the classified output image to the directory mounted on images/test. Then, these images will be easily viewed from the Jetson inference/data/images/test directory of the host device

```
# C++
$ ./imagenet images/orange_0.jpg images/test/output_0.jpg     # (default network
is googlenet)

# Python
$ ./imagenet.py images/orange_0.jpg images/test/output_0.jpg  # (default network
is googlenet)
```

Note: TensorRT will take a few minutes to optimize the network when running each model for the first time. This optimized network file is then cached on disk, so running this model in the future will load fasterIn addition to loading a single image, you can also load directories or video files for other images.

2.Download other classification models

The following pre trained image classification models are available for use and will be automatically downloaded:

Network	CLI argument	NetworkType enum
AlexNet	`alexnet`	`ALEXNET`
GoogleNet	`googlenet`	`GOOGLENET`
GoogleNet-12	`googlenet-12`	`GOOGLENET_12`
ResNet-18	`resnet-18`	`RESNET_18`
ResNet-50	`resnet-50`	`RESNET_50`
ResNet-101	`resnet-101`	`RESNET_101`
ResNet-152	`resnet-152`	`RESNET_152`
VGG-16	`vgg-16`	`VGG-16`
VGG-19	`vgg-19`	`VGG-19`
Inception-v4	`inception-v4`	`INCEPTION_V4`

Usually, more complex networks can have higher classification accuracy and increase runtime.

Using different classification models:

You can specify the model to load by setting the -- network flag on the command line to one of the corresponding CLI parameters in the table above. By default, if the optional -- network flag is not specified, GoogleNet will be loaded.

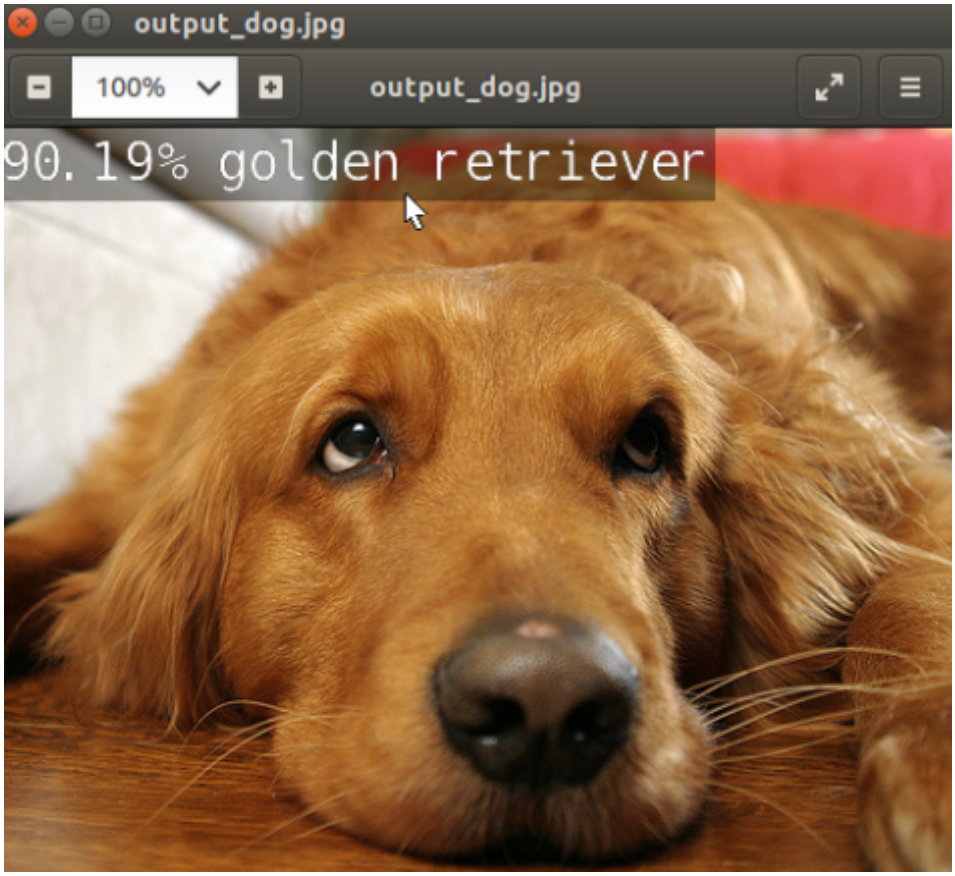
After building the project, please ensure that your terminal is located in the aarch64/bin directory:

```
cd jetson-inference/build/aarch64/bin
```

Here are some examples of using the ResNet-18 model:

```
# C++
$ ./imagenet --network=resnet-18 images/dog_2.jpg images/test/output_dog.jpg

# Python
$ ./imagenet.py --network=resnet-18 images/dog_2.jpg images/test/output_dog.jpg
```



Note: If you are building your own environment, you must download the resnet-18.tar.gz model file to the network folder and unzip it before running the above program. By using the image we provide, you can directly input the above program

3.Run real-time camera recognition demonstration

The imagenet.cpp/imagent.py sample we previously used can also be used for real-time camera streaming. The supported camera types include:

- MIPI CSI cameras (`csi://0`)
- V4L2 cameras (`/dev/video0`)
- RTP/RTSP streams (`rtsp://username:password@ip:port`)

Here are some typical scenarios for launching programs on camera feeds.

C++

```
$ ./imagenet csi://0                    # MIPI CSI camera
$ ./imagenet /dev/video0                # V4L2 camera
$ ./imagenet /dev/video0 output.mp4     # save to video file
```

python

```
$ ./imagenet.py csi://0                 # MIPI CSI camera
$ ./imagenet.py /dev/video0             # V4L2 camera
$ ./imagenet.py /dev/video0 output.mp4  # save to video file
```

The OpenGL window displays the real-time camera stream, classification object name, confidence level of the classification object, and frame rate of the network. On Jetson orin nano, the frame rate of GoogleNet and ResNet-18 should be as high as approximately 75 FPS (faster than other Jetson models).

This application can recognize up to 1000 different types of objects, as the classification model is trained on an ILSVRC ImageNet dataset containing 1000 types of objects. Name mapping for 1000 types of objects, available in data/networks/ilsvrc12_ synset_ Found in repo under words.txtThis concludes the tutorial on image classification for Hello AI World. Next, we will start using an object detection network, which provides us with bounding box coordinates for multiple objects per frame.