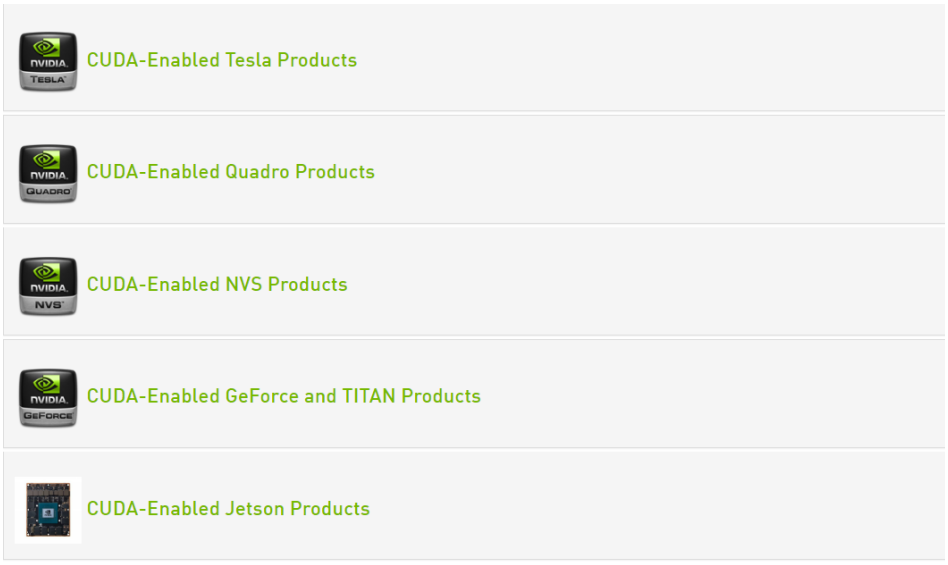
3.6 Tools of NVIDIA for AI field

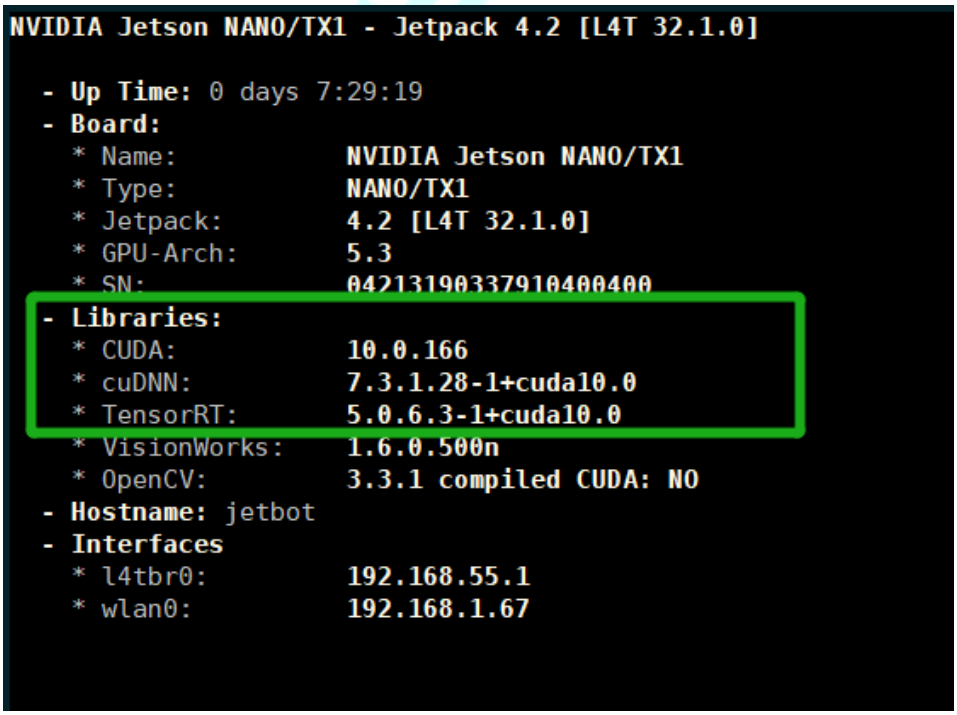
The Jetson nano used by Jetbot is a CUDA-enabled product of the Jetson series as shown below, so these features are available on the Jetson nano main control board used by Jetbot for immediate use.



Jetson Products

GPU	Compute Capability
Jetson AGX Xavier	7.2
Jetson Nano	5.3
Jetson TX2	6.2
Jetson TX1	5.3
Tegra X1	5.3

The version information for these components is shown below:



1. CUDA NVIDIA's Universal Parallel Computing Architecture

CUDA (Compute Unified Device Architecture) is a computing platform introduced by graphics card manufacturer NVIDIA. CUDA is a general-purpose parallel computing architecture introduced by NVIDIA that enables GPUs to solve complex computing problems.

With CUDA, GPUs can be easily used for general purpose computing (somewhat like numerical calculations in the CPU, etc.). Before CUDA, GPUs were generally only used for graphics rendering (eg via OpenGL, DirectX). Developers can perform parallel programming by calling CUDA's API for high performance computing purposes.

```
device = torch.device('cuda')
model = model.to(device)
model = model.eval().half()
```

The above code is where CUDA is often used in future projects. Generally, after the AI model is loaded, the weight of the current model is located in the CPU memory, and the above code is executed to be transmitted to the GPU device.

If you are interested in CUDA, if you want to know more about CUDA, you can go to the NVIDIA developer website to study:
https://developer.nvidia.com/cuda-downloads

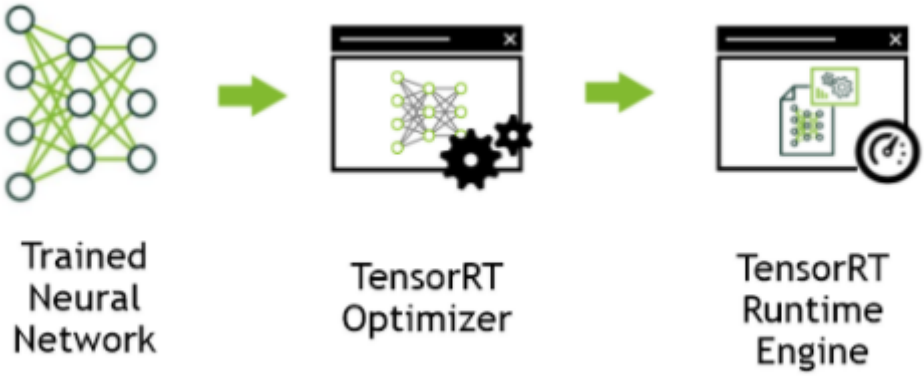
2.cuDNN deep neural network GPU acceleration primitive library

NVIDIA cuDNN is a GPU-accelerated library for deep neural networks. It emphasizes performance, ease of use, and low memory overhead. NVIDIA cuDNN can be integrated into higher-level machine learning frameworks such as Google's Tensorflow and the popular caffe software from the University of California at Berkeley.

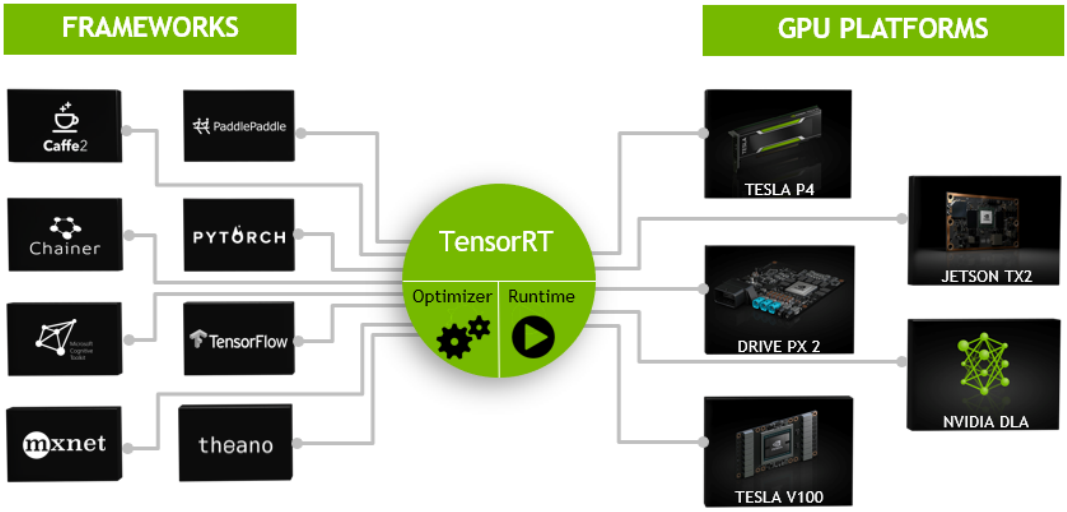
The simple plug-in design allows developers to focus on designing and implementing neural network models rather than simply adjusting performance while also enabling high-performance modern parallel computing on the GPU.

The relationship between CUDA and CUDNN:
CUDA is seen as a workbench with many tools such as hammers and screwdrivers. cuDNN is a CUDA-based deep learning GPU acceleration library that enables deep learning calculations on the GPU. It is equivalent to a working tool, such as a wrench. However, when the CUDA workbench was bought, no wrench was sent. To run a deep neural network on CUDA, install cuDNN, just like if you want to screw a nut, you need to buy the wrench. This will enable the GPU to work on deep neural networks, which is much faster than the CPU.

3.TensorRT high performance deep learning reasoning optimization accelerator



As shown above: TensorRT is a high-performance neural network inference optimizer and runtime engine for production deployment.



As shown above, TensorRT is defined as part of the high-performance inference optimizer and component runtime engine. It can use neural networks trained on these popular frameworks to optimize neural network calculations, generate a lightweight runtime engine (which is the only thing you need to deploy into a production environment), and then it will maximize throughput, latency, And the performance on these GPU platforms.

NVIDIA TensorRT is a high performance deep learning inference optimizer and runtime application that provides low latency and high throughput deep learning inference. With TensorRT, you can optimize your neural network model, accurately calibrate low-precision, and ultimately deploy your model to very large data center, embedded or automotive product platforms.

At the heart of TensorRT is a C++ library that facilitates high-performance inference for NVIDIA graphics processing units (GPUs).
It is designed to work in complementary ways with training frameworks such as TensorFlow, Caffe, PyTorch, and MXNet. It focuses on running a trained network quickly and efficiently on the GPU to produce results (a process that is called scoring, detection, regression or inference everywhere).

TensorRT-based gpu applications execute 100 times faster than CPUs in the reasoning of models that train all major frameworks.

TensorRT provides INT8 and FP16 optimization for production deployment of deep learning inference applications such as video streaming, speech recognition, push and natural language processing. Reducing inference accuracy can greatly reduce application latency, which is a requirement for many real-time services as well as automated and embedded applications.

You can import trained models from each deep learning framework to TensorRT. After application optimization, TensorRT selects a platform-specific core and maximizes performance on Jetbot's GPU. Developers using TensorRT can then focus on creating new ai-based applications instead of performance tuning for inferential deployment.