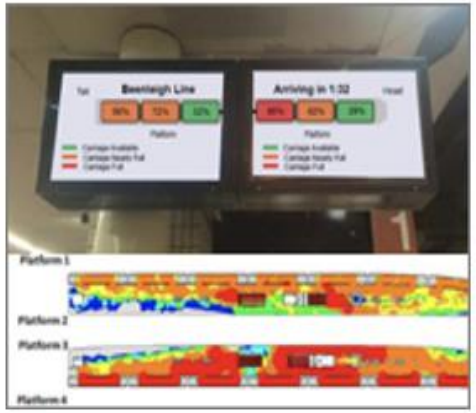
NVIDIA TRANSFER LEARNING TOOLKIT

NVIDIA 开发者社区 何琨

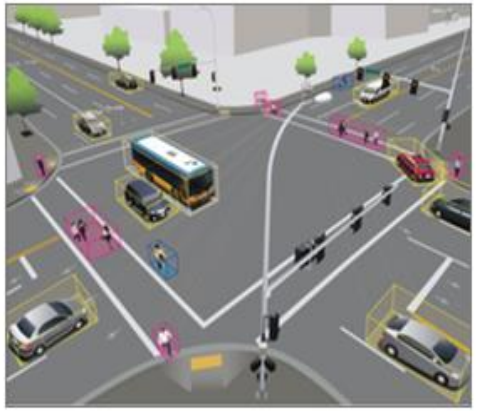
CV IN DEEPLEARNING

Access Control

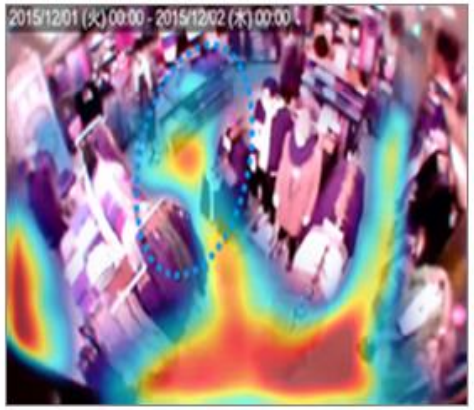


Public Transit

Parking Management



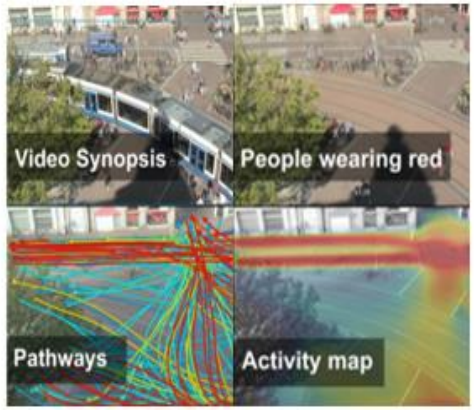
Traffic Engineering



Retail Analytics

Securing Critical Infrastructure

Managing Logistics



Forensic Analysis

Deep Learning Workflow Management

Deep Learning Challenges

Third Party Pre-Trained Models
- Lack accuracy
- Use case limitations
- Model size limitations
- Unoptimized for GPUs

Deep Learning Training
- Compute Resources
- Time spent training from scratch
- Learning DL frameworks

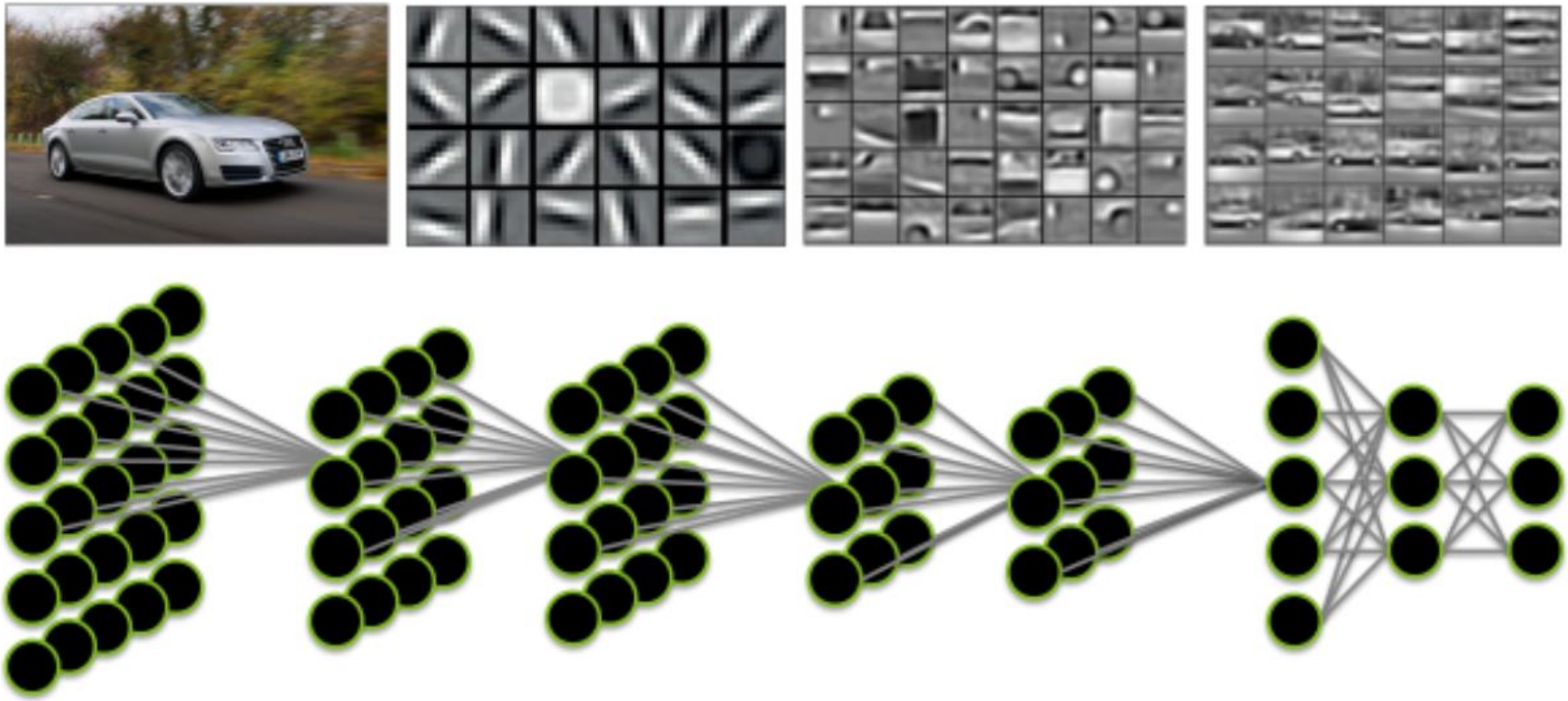
Deep Learning Inference
- Unclear workflows for production ready models
- Complex application pipeline

NVIDIA Deep Learning Solution

Transfer Learning Toolkit
- GPU accelerated pre trained models
- Incremental Training
- Pruning
- Easy to use
- Abstraction from learning DL frameworks

DeepStream SDK
- Faster intelligent insights
- Track inference
- End to end- easy AI deployment

How Deep Learning Network Works

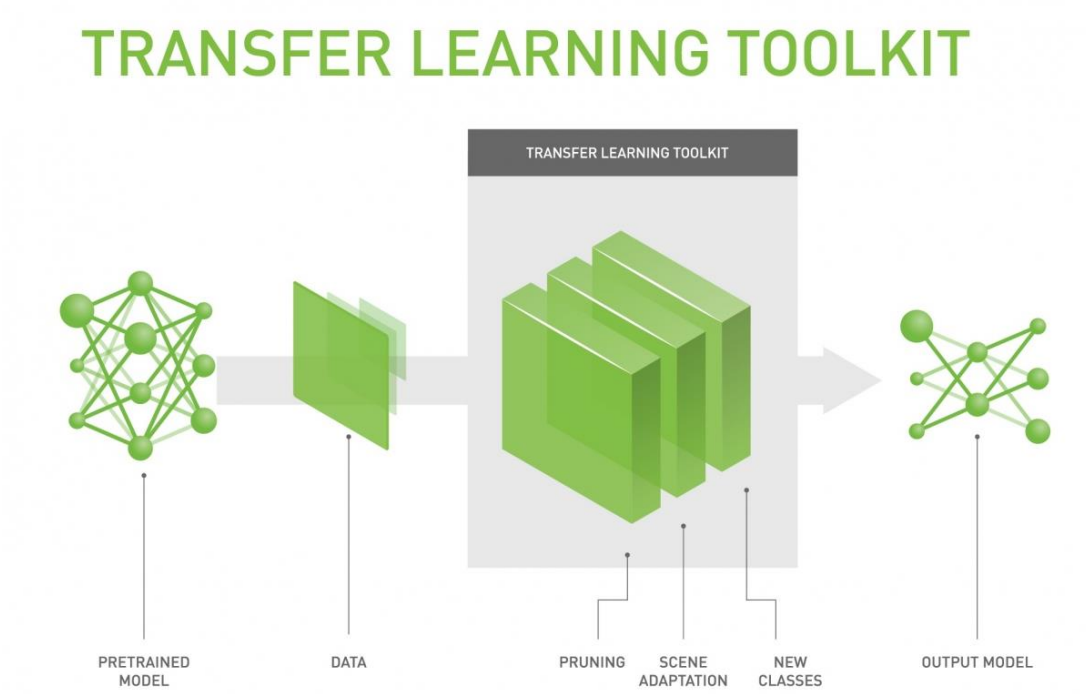


Transfer Learning is a process of transferring learned features from one model to another

NVIDIA TRANSFER LEARNING TOOLKIT

Transfer Learning Toolkit是一个基于python的工具包，它使开发人员能够利用NVIDIA预先训练的模型，并为开发人员提供一系列的工具，使流行的网络架构适应他们自己的数据，并且能够训练、调整、修剪和导出模型以进行部署。它还拥有简单的接口和抽象API，提高了深度学习训练工作流的效率。

- GPU优化的预训练砝码，可用于计算机视觉任务
- 轻松修改配置文件以添加新类并使用自定义数据重新训练模型
- 在异构的多GPU环境中执行模型调整和重新训练
- 使用修剪功能缩小模型尺寸
- 模型导出API，可在具有NVIDIA Tesla和Jetson产品的NVIDIA DeepStream SDK上部署



NVIDIA TRANSFER LEARNING TOOLKIT

Efficient Pre-trained Models

GPU-accelerated high performance models trained on large scale datasets.

Faster Inference with Model Pruning

Model pruning reduces size of the model resulting in faster inference

Training with Multiple GPUs

Re-training models, adding custom data for multi GPU training using an easy to use tool

Abstraction

Abstraction from having deep knowledge of frameworks, simple intuitive interface to the features

Containerization

Packaged in a container easily accessible from NVIDIA GPU Cloud website. All code dependencies are managed automatically

Integration

Models exported using TLT are easily consumable for inference with **Deep Stream SDK**

NVIDIA TRANSFER LEARNING TOOLKIT

在指定的公共数据集上训练的图像分类和目标检测模型，可与Transfer Learning Toolkit一起使用。

Image Classification

- ResNet10/18/50
- VGG16/19
- MobileNet V1/V2
- AlexNet
- SqueezeNet
- GoogLeNet

Faster RCNN supporting backbones:

- ResNet10/18/50
- VGG16/19
- GoogLeNet
- MobileNet V1/V2

Object Detection

DetectNet_v2 supporting backbones:

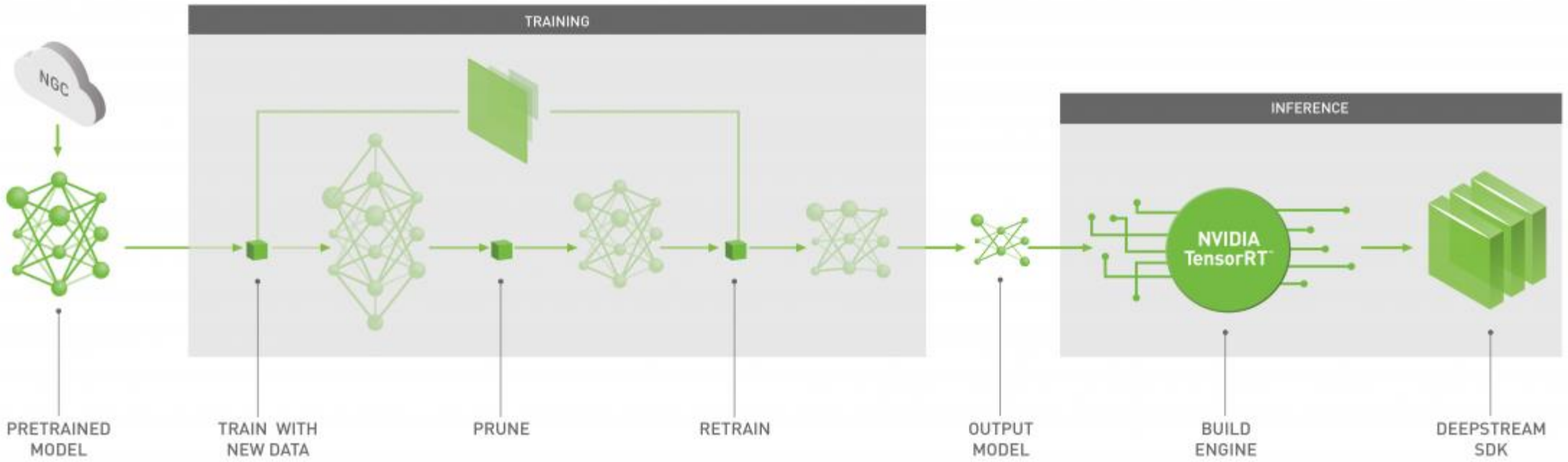
- ResNet10/18/50
- VGG 16/19
- GoogLeNet
- MobileNet V1/V2

SSD:

- ResNet10/18

NVIDIA TRANSFER LEARNING TOOLKIT

为应用在计算机视觉领域的深度学习工作流程，提供了全方位的便利工具



在服务器上部署TRANSFER LEARNING TOOLKIT

Hardware Requirements

Minimum

- 4 GB system RAM
- 4 GB of GPU RAM
- Single core CPU
- 1 GPU
- 50 GB of HDD space

Recommended

- 32 GB system RAM
- 32 GB of GPU RAM
- 8 core CPU
- 4 GPUs
- 100 GB of SSD space

Software Requirements

- Ubuntu 18.04 LTS/16.04 LTS
- NVIDIA GPU Cloud account and API key - https://ngc.nvidia.com/
- docker-ce installed, https://docs.docker.com/install/linux/docker-ce/ubuntu/
- nvidia-docker2 installed, instructions: https://github.com/nvidia/nvidia-docker/wiki/Installation-(version-2.0)
- NVIDIA GPU driver v410.xx or above

Note: DeepStream 4.0 - NVIDIA SDK inference https://developer.nvidia.com/deepstream-sdk is recommended.

在服务器上部署TRANSFER LEARNING TOOLKIT

Installation Prerequisites

•Install Docker. See: https://www.docker.com/.
•NVIDIA GPU driver v410.xx or above. Download from https://www.nvidia.com/Download/index.aspx?lang=en-us.
•Install NVIDIA Docker 2 from: https://github.com/NVIDIA/nvidia-docker.

Get an NGC API key

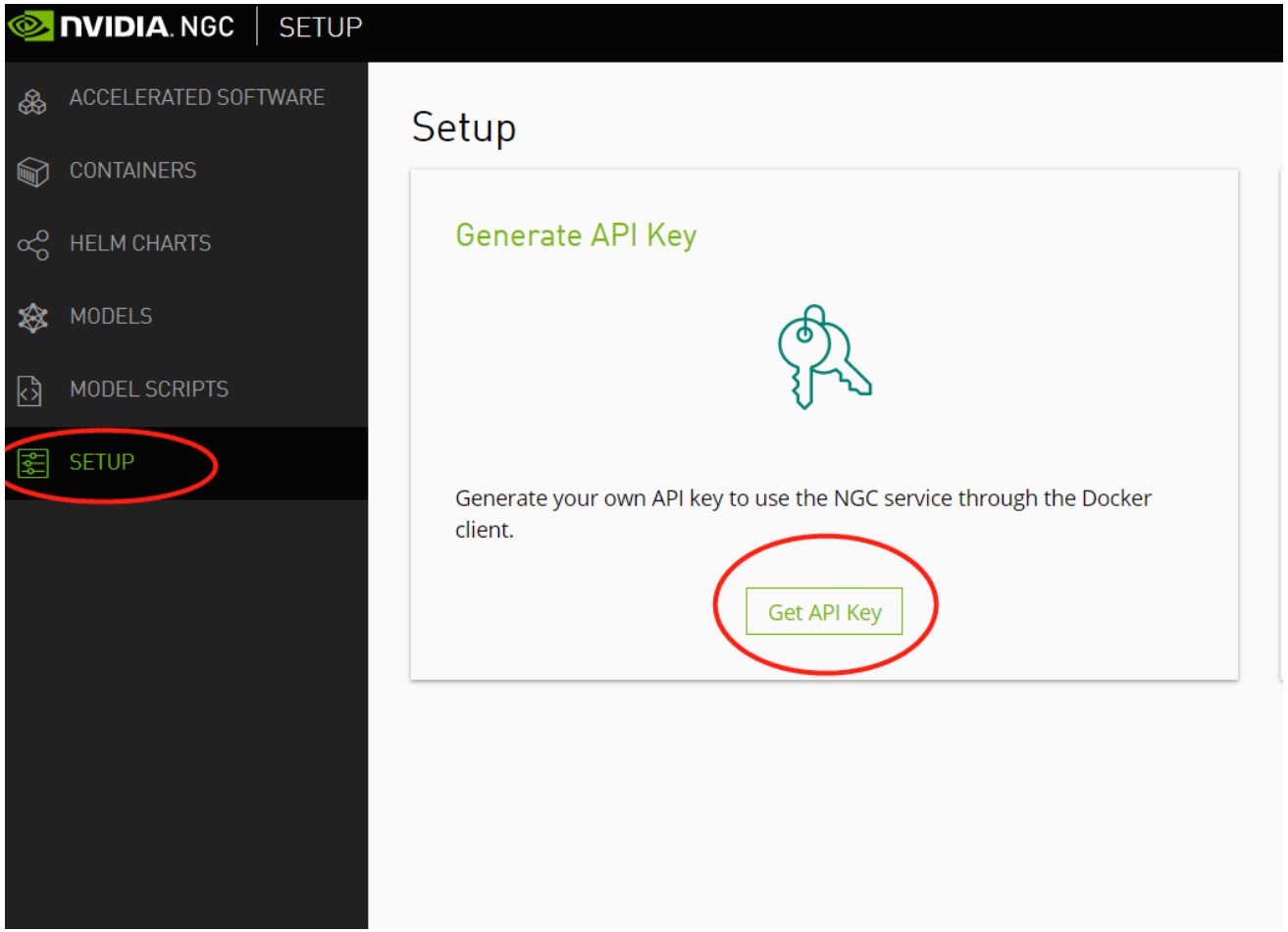
•NVIDIA GPU Cloud account and API key - https://ngc.nvidia.com/

1. Go to NGC and click the **Transfer Learning Toolkit** container in the **Catalog** tab. This message is displayed, **Sign in to access the PULL feature of this repository**.
2. Enter your email address and click **Next** or click **Create an Account**.
3. Choose your **organization** when prompted for Organization/Team.
4. Click **Sign In**.
5. Select the **Containers** tab on the left navigation pane and click the **Transfer Learning Toolkit** tile.

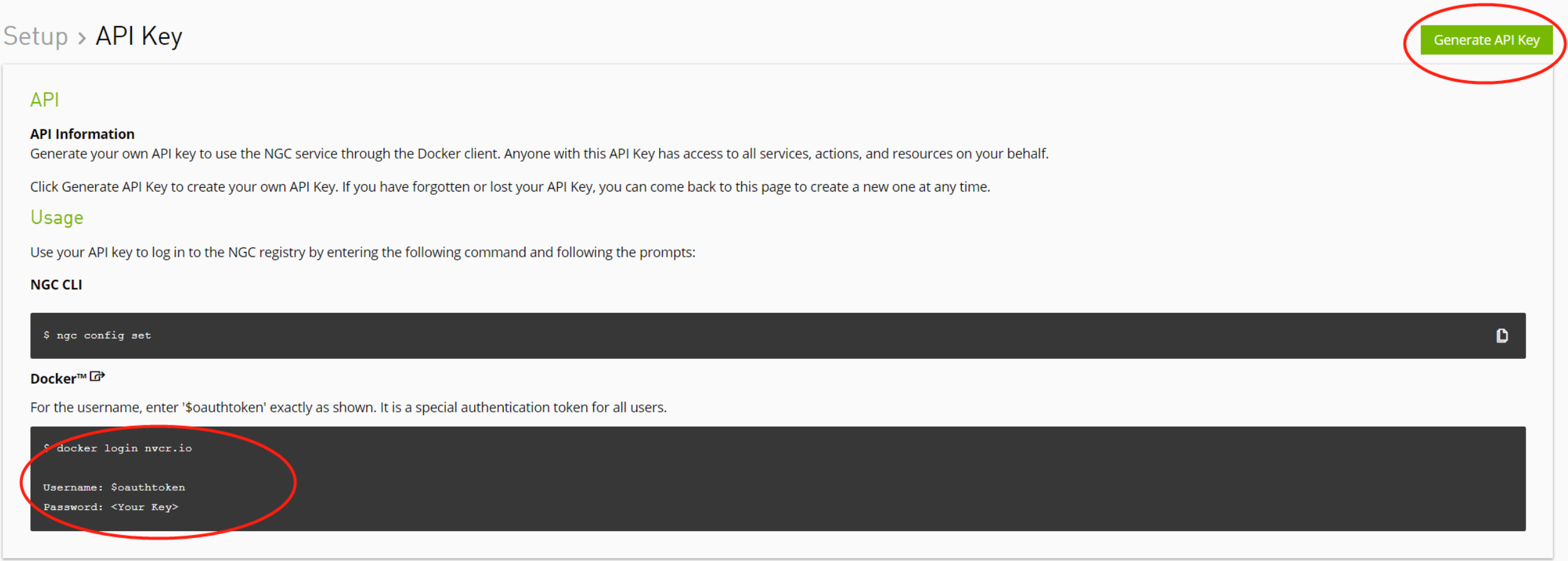
Download the docker container

•Execute docker login nvcr.io from the command line and enter your username and password.
  •Username: $oauthtoken
  •Password: API_KEY
•Execute docker pull nvcr.io/nvidia/tlt-streamanalytics:<version>

在服务器上部署TRANSFER LEARNING TOOLKIT



在服务器上部署TRANSFER LEARNING TOOLKIT



在服务器上部署TRANSFER LEARNING TOOLKIT

在服务器上运行TLT的镜像

1.Run the image using this command.

```
docker run --runtime=nvidia -it nvcr.io/nvidia/tlt-
streamanalytics:v1.0.1_py2 /bin/bash
```

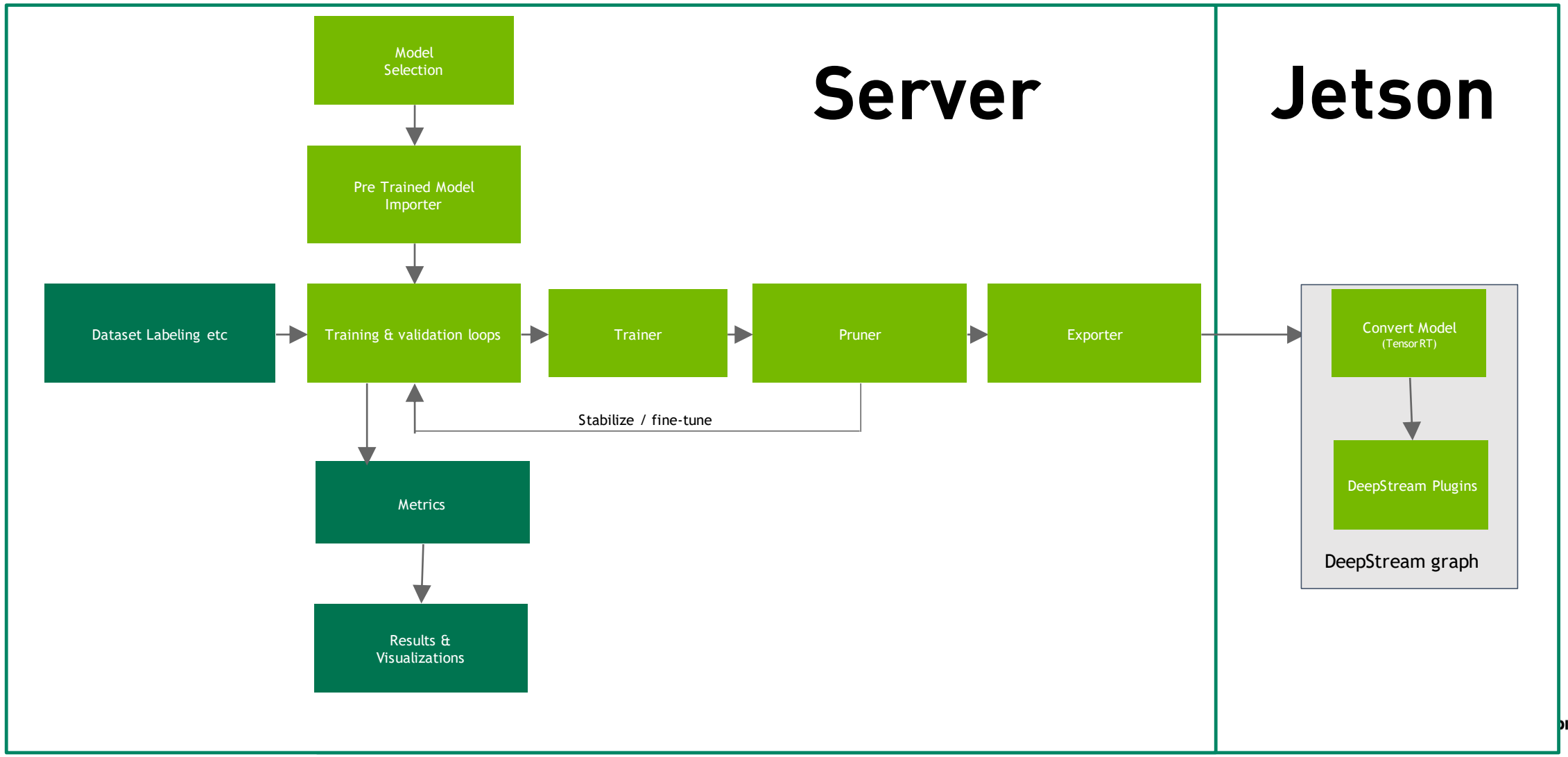
2.Mount local directories using -v and expose the docker ports to the host using -p

```
docker run --runtime=nvidia -it \
-v "/path/to/dir/on/host":"/path/to/dir/in/docker" \ -p 8888:8888 \
nvcr.io/nvidia/tlt-streamanalytics:v1.0_py2 /bin/bash
```

3.Invoke the jupyter notebook using the following command

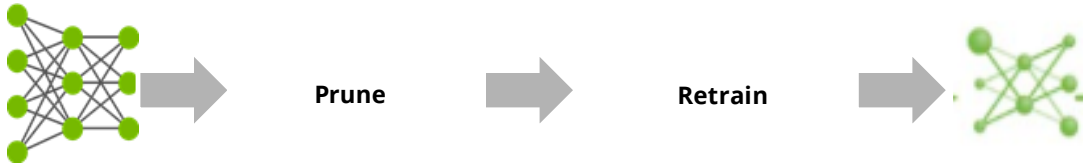
```
jupyter notebook --ip 0.0.0.0 --port 8888 --allow-root
```

TRANSFER LEARNING TOOLKIT的工作流程



TRANSFER LEARNING TOOLKIT的工作流程

Pruning



Network compression

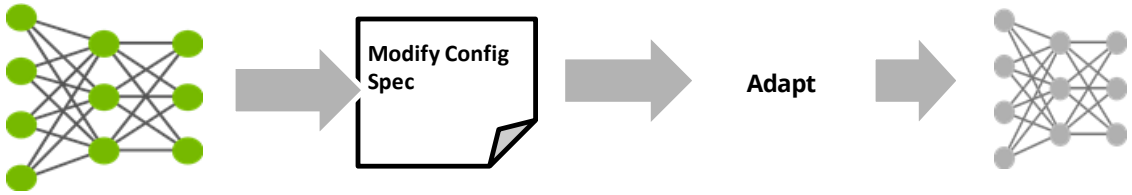
Scene Adaptation

Same network adapting to different angles and vantage points

Same network adapting to new data

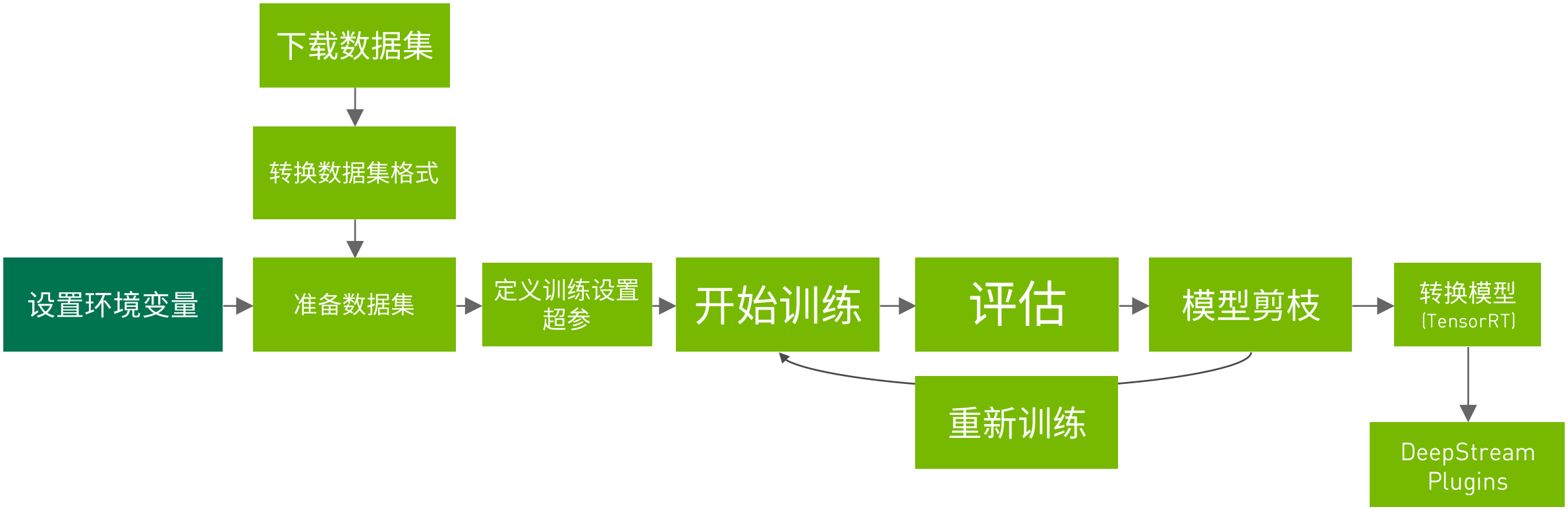
Train with new data from another vantage point, camera location, or added attribute

New Classes



Edited network detecting new class defined in config spec

TRANSFER LEARNING TOOLKIT的工作流程



TRANSFER LEARNING TOOLKIT的工作流程

TLT SSD example usecase

0. 设置环境变量
1. 准备数据集和预训练模型
   1.1 将数据集从kitti格式转换成tfrecords的格式
   1.2 下载预训练模型
2. 定义训练设置
3. 开始TLT训练
4. 评估训练结果
5. 模型剪枝
6. 训练剪枝模型
7. 评估训练好的模型
8. 可视化推理结果
9. 模型部署

TRANSFER LEARNING TOOLKIT的工作流程

0. 设置环境变量 ¶

```
# 设置环境变量.
print("Please replace the variable with your key.")
# 请从NGC官网生成API_KEY
%set_env KEY=cTJhcms3ODdvbHRsOWwxMTNvYW0yN3NuaHA6MTVjN2EzZjEtMDlhMi00YTZjLWFkZDgtMWY1ZmI5MGM0N2Qy
%set_env USER_EXPERIMENT_DIR=/workspace/tlt-experiments
%set_env DATA_DOWNLOAD_DIR=/workspace/tlt-experiments/data
%set_env SPECS_DIR=/workspace/examples/ssd/specs
!mkdir -p $DATA_DOWNLOAD_DIR
```

```
Please replace the variable with your key.
env: KEY=cTJhcms3ODdvbHRsOWwxMTNvYW0yN3NuaHA6MTVjN2EzZjEtMDlhMi00YTZjLWFkZDgtMWY1ZmI5MGM0N2Qy
env: USER_EXPERIMENT_DIR=/workspace/tlt-experiments
env: DATA_DOWNLOAD_DIR=/workspace/tlt-experiments/data
env: SPECS_DIR=/workspace/examples/ssd/specs
```

TRANSFER LEARNING TOOLKIT的工作流程

将数据集地址链接到$DATA_DOWNLOAD_DIR

```
!ln -sf /workspace-hekun/mydata/kitti/kitti_single/training $DATA_DOWNLOAD_DIR
```

1. 准备数据集和预训练模型

这里将使用kitti数据集。 可以访问下面链接查看详情： http://www.cvlibs.net/datasets/kitti/eval_object.php?obj_benchmark=2d.

从这里下载图片数据集 (http://www.cvlibs.net/download.php?file=data_object_image_2.zip)

从这里下载标签(http://www.cvlibs.net/download.php?file=data_object_label_2.zip)

并把他们保存在$DATA_DOWNLOAD_DIR.

```
# Check the dataset is present
!mkdir -p $DATA_DOWNLOAD_DIR
!if [ ! -f $DATA_DOWNLOAD_DIR/data_object_image_2.zip ]; then echo 'Image zip file not found, please download.'; else echo 'Found Ima
!if [ ! -f $DATA_DOWNLOAD_DIR/data_object_label_2.zip ]; then echo 'Label zip file not found, please download.'; else echo 'Found Lab
```

```
Image zip file not found, please download.
Label zip file not found, please download.
```

```
# unpack
!unzip -u $DATA_DOWNLOAD_DIR/data_object_image_2.zip -d $DATA_DOWNLOAD_DIR
!unzip -u $DATA_DOWNLOAD_DIR/data_object_label_2.zip -d $DATA_DOWNLOAD_DIR
```

```
# verify
!ls -l /workspace-hekun/mydata/kitti/kitti_single/training #$DATA_DOWNLOAD_DIR/
```

```
total 472
drwxrwxr-x 2 1000 1000 225280 Jan  4 15:36 image_2
drwxrwxr-x 2 1000 1000 258048 Jan  4 15:36 label_2
```

TRANSFER LEARNING TOOLKIT的工作流程

1.1 将数据集从KITTI格式转换成TFrecords

- 更新tfrecords定义文件
- 使用tlt-dataset-convert 创建tfrecords
- TFRecords 只需要被创建一次

```
print("TFrecords conversion spec file for training")
!cat $SPECS_DIR/ssd_tfrecords_kitti_trainval.txt
```

```
TFrecords conversion spec file for training
kitti_config {
  root_directory_path: "/workspace/tlt-experiments/data/training"
  image_dir_name: "image_2"
  label_dir_name: "label_2"
  image_extension: ".png"
  partition_mode: "random"
  num_partitions: 2
  val_split: 14
  num_shards: 10
}
image_directory_path: "/workspace/tlt-experiments/data/training"
```

```
# Creating a new directory for the output tfrecords dump.
!mkdir -p $USER_EXPERIMENT_DIR/tfrecords
#KITTI trainval
!tlt-dataset-convert -d $SPECS_DIR/ssd_tfrecords_kitti_trainval.txt \
                     -o $USER_EXPERIMENT_DIR/tfrecords/kitti_trainval/kitti_trainval
```

TRANSFER LEARNING TOOLKIT的工作流程

1.2 下载模型

我们这里将使用NGC CLI来下载预训练模型. 访问 ngc.nvidia.com 查看详情，点击 SETUP 查看教程.

```
!ngc registry model list nvidia/iva/tlt_*_ssd
```

```
+---------+---------+---------+---------+---------+---------+---------+---------+
| Name    | Reposit | Latest  | Applica | Framewo | Precisi | Last Mo | Permiss |
|         | ory     | Version | tion    | rk      | on      | dified  | ion     |
+---------+---------+---------+---------+---------+---------+---------+---------+
| TLT Res | nvidia/ | 1       | Object  | Transfe | FP32    | Oct 18, | unlocke |
| Net10   | iva/tlt |         | Detecti | r Learn |         | 2019    | d       |
| SSD     | _resnet |         | on      | ing     |         |         |         |
|         | 10_ssd  |         |         | Toolkit |         |         |         |
| TLT Res | nvidia/ | 1       | Object  | Transfe | FP32    | Oct 18, | unlocke |
| Net18   | iva/tlt |         | Detecti | r Learn |         | 2019    | d       |
| SSD     | _resnet |         | on      | ing     |         |         |         |
|         | 18_ssd  |         |         | Toolkit |         |         |         |
+---------+---------+---------+---------+---------+---------+---------+---------+
```

```
!mkdir -p $USER_EXPERIMENT_DIR/pretrained_resnet18/
```

```
# Pull pretrained model from NGC
!ngc registry model download-version nvidia/iva/tlt_resnet18_ssd:1 --dest $USER_EXPERIMENT_DIR/pretrained_resnet18
```

```
Downloaded 82.41 MB in 42m 3s, Download speed: 33.44 KB/s
------------------------------------------------------
Transfer id: tlt_resnet18_ssd_v1 Download status: Completed.
Downloaded local path: /workspace/tlt-experiments/pretrained_resnet18/tlt_resnet18_ssd_v1
Total files downloaded: 2
Total downloaded size: 82.41 MB
Started at: 2020-01-04 16:32:19.183090
Completed at: 2020-01-04 17:14:22.650753
Duration taken: 42m 3s
------------------------------------------------------
```

```
print("Check that model is downloaded into dir.")
!ls -l $USER_EXPERIMENT_DIR/pretrained_resnet18/tlt_resnet18_ssd_v1
```

TRANSFER LEARNING TOOLKIT的工作流程

2. 提供训练设置定义 ¶

- Tfrecords for the train datasets
  - 为了使用新生成的tfrecords，请将spec文件中的dataset_config参数更新为' $SPECS_DIR/ssd_train_resnet18_kit .txt '
- 预训练模型
- 为动态数据增加参数
- 设置一些训练的超参数， batch size, number of epochs, learning rate etc.

```
!cat $SPECS_DIR/ssd_train_resnet18_kitti.txt
```

TRANSFER LEARNING TOOLKIT的工作流程

3. 开始TLT训练

- 设置样本地址和输出模型地址
- WARNING: 这里训练的时间会很长，经过我实际测试单个V100要运行8个小时

```
!mkdir -p $USER_EXPERIMENT_DIR/experiment_dir_unpruned
```

```
!tlt-train ssd -e $SPECS_DIR/ssd_train_resnet18_kitti.txt \
               -r $USER_EXPERIMENT_DIR/experiment_dir_unpruned \
               -k $KEY \
               -m $USER_EXPERIMENT_DIR/pretrained_resnet18/tlt_resnet18_ssd_v1/resnet18.hdf5
```

```
Epoch 00180: saving model to /workspace/tlt-experiments/experiment_dir_unpruned/weights/ssd_resnet18_epoch_180.tlt
Number of images in the evaluation dataset: 1047
()
Producing predictions batch-wise: 100%|#########| 33/33 [00:33<00:00,  1.03s/it]
Matching predictions to ground truth, class 1/3.: 100%|#| 131853/131853 [00:10<00:00, 12659.37it/s]
Matching predictions to ground truth, class 2/3.: 100%|#| 22725/22725 [00:00<00:00, 33747.36it/s]
Matching predictions to ground truth, class 3/3.: 100%|#| 56622/56622 [00:02<00:00, 24782.92it/s]
Computing precisions and recalls, class 1/3
Computing precisions and recalls, class 2/3
Computing precisions and recalls, class 3/3
Computing average precision, class 1/3
Computing average precision, class 2/3
Computing average precision, class 3/3
********************************
car           AP    0.888
cyclist       AP    0.803
pedestrian    AP    0.723
              mAP   0.805
********************************
```

```
print("For multi-GPU, please uncomment and run this instead. Change --gpus based on your machine.")
# !tlt-train ssd -e $SPECS_DIR/ssd_train_resnet18_kitti.txt |
#                -r $USER_EXPERIMENT_DIR/experiment_dir_unpruned |
#                -k $KEY |
#                -m $USER_EXPERIMENT_DIR/pretrained_resnet18/tlt_resnet18_ssd_v1/resnet18.hdf5 |
#                --gpus 2
```

TRANSFER LEARNING TOOLKIT的工作流程

4. 评估模型

```
!tlt-evaluate ssd -e $SPECS_DIR/ssd_train_resnet18_kitti.txt \
                  -m $USER_EXPERIMENT_DIR/experiment_dir_unpruned/weights/ssd_resnet18_epoch_180.tlt \
                  -k $KEY
```

```
Number of images in the evaluation dataset: 1047
()
Producing predictions batch-wise:   0%|                  | 0/33 [00:00<?, ?it/s]2020-01-05 07:38:37.765287: I tensorflow/stream_exe
cutor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
Producing predictions batch-wise: 100%|#########| 33/33 [00:30<00:00,  1.07it/s]
Matching predictions to ground truth, class 1/3.: 100%|#| 131812/131812 [00:09<00:00, 14482.08it/s]
Matching predictions to ground truth, class 2/3.: 100%|#| 22801/22801 [00:00<00:00, 40236.74it/s]
Matching predictions to ground truth, class 3/3.: 100%|#| 56587/56587 [00:01<00:00, 29452.24it/s]
Computing precisions and recalls, class 1/3
Computing precisions and recalls, class 2/3
Computing precisions and recalls, class 3/3
Computing average precision, class 1/3
Computing average precision, class 2/3
Computing average precision, class 3/3
********************************
car            AP    0.888
cyclist        AP    0.805
pedestrian     AP    0.724
               mAP   0.806
********************************
```

TRANSFER LEARNING TOOLKIT的工作流程

5. 模型剪枝 ¶

- 定义预训练模型
- 定义threshold
- 定义API_KEY
- 定义输出模型位置

通常，你只需要调整'-pth '(阈值)来改变模型的准确性和模型大小. 更高的 `pth` 会获得更小的模型 (以及更快的速度) 但是比较低的精度. 阈值取决于数据集和模型。

```
!tlt-prune -pm $USER_EXPERIMENT_DIR/experiment_dir_unpruned/weights/ssd_resnet18_epoch_180.tlt \
           -o $USER_EXPERIMENT_DIR/experiment_dir_pruned/ \
           -eq intersection \
           -pth 0.6 \
           -k $KEY
```

```
Using TensorFlow backend.
WARNING:tensorflow:From /usr/local/lib/python2.7/dist-packages/tensorflow/python/framework/op_def_library.py:263: colocate_with (fr
om tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
2020-01-05 07:39:33,538 [WARNING] tensorflow: From /usr/local/lib/python2.7/dist-packages/tensorflow/python/framework/op_def_librar
y.py:263: colocate_with (from tensorflow.python.framework.ops) is deprecated and will be removed in a future version.
Instructions for updating:
Colocations handled automatically by placer.
2020-01-05 07:39:35.844711: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlo
w binary was not compiled to use: AVX2 FMA
2020-01-05 07:39:36.282324: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:998] successful NUMA node read from SysFS had ne
gative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-01-05 07:39:36.283978: I tensorflow/compiler/xla/service/service.cc:150] XLA service 0x8e9b5a0 executing computations on platf
orm CUDA. Devices:
2020-01-05 07:39:36.284024: I tensorflow/compiler/xla/service/service.cc:158]   StreamExecutor device (0): Tesla P100-PCIE-16GB, Co
mpute Capability 6.0
2020-01-05 07:39:36.287021: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 2499995000 Hz
2020-01-05 07:39:36.288119: I tensorflow/compiler/xla/service/service.cc:150] XLA service 0x8fb6a70 executing computations on platf
orm Host. Devices:
```

```
!ls -rlt $USER_EXPERIMENT_DIR/experiment_dir_pruned/
```

```
total 24236
-rw-r--r-- 1 root root 24816152 Jan  5 07:40 ssd_resnet18_pruned.tlt
```

TRANSFER LEARNING TOOLKIT的工作流程

6. 重新训练剪枝后的模型

- 修剪后的模型需要重新训练以恢复精度
- 设置训练的设置
- WARNING: 训练会花费比较长的时间，您可以减少epochs来少训练几圈

```
# Printing the retrain spec file.
# Here we have updated the spec file to include the newly pruned model as a pretrained weights.
!cat $SPECS_DIR/ssd_retrain_resnet18_kitti.txt
```

```
        value: "car"
    }
    target_class_mapping {
        key: "pedestrian"
        value: "pedestrian"
    }
    target_class_mapping {
        key: "cyclist"
        value: "cyclist"
    }
    target_class_mapping {
        key: "van"
        value: "car"
    }
    target_class_mapping {
        key: "person_sitting"
        value: "pedestrian"
    }
validation_fold: 0
}
```

```
!mkdir -p $USER_EXPERIMENT_DIR/experiment_dir_retrain
```

```
# Retraining using the pruned model as pretrained weights
!tlt-train ssd -e $SPECS_DIR/ssd_retrain_resnet18_kitti.txt \
               -r $USER_EXPERIMENT_DIR/experiment_dir_retrain \
               -m $USER_EXPERIMENT_DIR/experiment_dir_pruned/ssd_resnet18_pruned.tlt \
               -k $KEY
```

```
Epoch 00030: saving model to /workspace/tlt-experiments/experiment_dir_retrain/weights/ssd_resnet18_epoch_030.tlt
```

TRANSFER LEARNING TOOLKIT的工作流程

7. 评估训练好的模型

```
!tlt-evaluate ssd -e $SPECS_DIR/ssd_retrain_resnet18_kitti.txt \
                  -m $USER_EXPERIMENT_DIR/experiment_dir_retrain/weights/ssd_resnet18_epoch_030.tlt \
                  -k $KEY
```

```
Number of images in the evaluation dataset: 1047
()
Producing predictions batch-wise:   0%|                  | 0/33 [00:00<?, ?it/s]2020-01-05 09:44:46.710822: I tensorflow/stream_exe
cutor/dso_loader.cc:152] successfully opened CUDA library libcublas.so.10.0 locally
Producing predictions batch-wise: 100%|##########| 33/33 [00:23<00:00,  1.42it/s]
Matching predictions to ground truth, class 1/3.: 100%|#| 141336/141336 [00:10<00:00, 13502.68it/s]
Matching predictions to ground truth, class 2/3.: 100%|#| 27458/27458 [00:00<00:00, 41144.65it/s]
Matching predictions to ground truth, class 3/3.: 100%|#| 42406/42406 [00:01<00:00, 25978.59it/s]
Computing precisions and recalls, class 1/3
Computing precisions and recalls, class 2/3
Computing precisions and recalls, class 3/3
Computing average precision, class 1/3
Computing average precision, class 2/3
Computing average precision, class 3/3
**********************************
car             AP    0.864
cyclist         AP    0.535
pedestrian      AP    0.538
                mAP   0.646
**********************************
```

TRANSFER LEARNING TOOLKIT的工作流程

8. 可视化推理结果

在本节中，我们将运行tlt-infer工具，以对经过训练的模型生成推理可视化结果。

```
# Running inference for detection on n images
!tlt-infer ssd -i $USER_EXPERIMENT_DIR/data/kitti_single/testing/image_2 \
               -o $USER_EXPERIMENT_DIR/ssd_infer_images \
               -e $SPECS_DIR/ssd_retrain_resnet18_kitti.txt \
               -m $USER_EXPERIMENT_DIR/experiment_dir_retrain/weights/ssd_resnet18_epoch_030.tlt \
               -l $USER_EXPERIMENT_DIR/ssd_infer_labels \
               -k $KEY
```

```
concatenate_3 (Concatenate)     (None, 59928, 1, 15) 0           mbox_conf_sigmoid[0][0]
                                                                 mbox_loc[0][0]
                                                                 mbox_priorbox[0][0]
__________________________________________________________________________________________________
ssd_predictions (Reshape)       (None, 59928, 15)    0           concatenate_3[0][0]
==================================================================================================
Total params: 6,111,212
Trainable params: 6,104,396
Non-trainable params: 6,816
__________________________________________________________________________________________________
WARNING:tensorflow:From ./ssd/box_coder/output_decoder_layer.py:83: to_float (from tensorflow.python.ops.math_ops) is deprecated an
d will be removed in a future version.
Instructions for updating:
Use tf.cast instead.
2020-01-05 09:51:33,074 [WARNING] tensorflow: From ./ssd/box_coder/output_decoder_layer.py:83: to_float (from tensorflow.python.op
s.math_ops) is deprecated and will be removed in a future version.
Instructions for updating:
Use tf.cast instead.

100%|##########| 7518/7518 [26:28<00:00,  4.73it/s]
```

TRANSFER LEARNING TOOLKIT的工作流程

`tlt-infer` 会有两个输出。

1. 图片存在 `$USER_EXPERIMENT_DIR/ssd_infer_images`
2. label存在 `$USER_EXPERIMENT_DIR/ssd_infer_labels`

```python
# Simple grid visualizer
import matplotlib.pyplot as plt
import os
from math import ceil
valid_image_ext = ['.jpg', '.png', '.jpeg', '.ppm']

def visualize_images(image_dir, num_cols=4, num_images=10):
    output_path = os.path.join(os.environ['USER_EXPERIMENT_DIR'], image_dir)
    num_rows = int(ceil(float(num_images) / float(num_cols)))
    f, axarr = plt.subplots(num_rows, num_cols, figsize=[80,30])
    f.tight_layout()
    a = [os.path.join(output_path, image) for image in os.listdir(output_path)
         if os.path.splitext(image)[1].lower() in valid_image_ext]
    for idx, img_path in enumerate(a[:num_images]):
        col_id = idx % num_cols
        row_id = idx / num_cols
        img = plt.imread(img_path)
        axarr[row_id, col_id].imshow(img)
```

```python
# Visualizing the sample images.
OUTPUT_PATH = 'ssd_infer_images' # relative path from $USER_EXPERIMENT_DIR.
COLS = 3 # number of columns in the visualizer grid.
IMAGES = 9 # number of images to visualize.

visualize_images(OUTPUT_PATH, num_cols=COLS, num_images=IMAGES)
```

TRANSFER LEARNING TOOLKIT的工作流程

9. 部署!

导出训练模型

```
!mkdir -p $USER_EXPERIMENT_DIR/export
# Export in FP32 mode.
!tlt-export $USER_EXPERIMENT_DIR/experiment_dir_retrain/weights/ssd_resnet18_epoch_030.tlt \
            -k $KEY \
            -o $USER_EXPERIMENT_DIR/export/ssd_resnet18_epoch_180.etlt \
            --outputs NMS \
            -e $SPECS_DIR/ssd_retrain_resnet18_kitti.txt \
            --export_module ssd
```

```
Using TensorFlow backend.
2020-01-05 10:21:32,558 [INFO] iva.ssd.scripts.export: Loading experiment spec at /workspace/examples/ssd/specs/ssd_retrain_resnet18
_kitti.txt.
2020-01-05 10:21:32,559 [INFO] /usr/local/lib/python2.7/dist-packages/iva/ssd/utils/spec_loader.pyc: Merging specification from /wor
kspace/examples/ssd/specs/ssd_retrain_resnet18_kitti.txt
2020-01-05 10:21:32.561583: I tensorflow/core/platform/cpu_feature_guard.cc:141] Your CPU supports instructions that this TensorFlow
binary was not compiled to use: AVX2 FMA
2020-01-05 10:21:32.690306: I tensorflow/stream_executor/cuda/cuda_gpu_executor.cc:998] successful NUMA node read from SysFS had neg
ative value (-1), but there must be at least one NUMA node, so returning NUMA node zero
2020-01-05 10:21:32.692069: I tensorflow/compiler/xla/service/service.cc:150] XLA service 0x7635c50 executing computations on platfo
rm CUDA. Devices:
2020-01-05 10:21:32.692113: I tensorflow/compiler/xla/service/service.cc:158]   StreamExecutor device (0): Tesla P100-PCIE-16GB, Com
pute Capability 6.0
2020-01-05 10:21:32.694748: I tensorflow/core/platform/profile_utils/cpu_utils.cc:94] CPU Frequency: 2499995000 Hz
2020-01-05 10:21:32.695908: I tensorflow/compiler/xla/service/service.cc:150] XLA service 0x769e990 executing computations on platfo
rm Host. Devices:
2020-01-05 10:21:32.695939: I tensorflow/compiler/xla/service/service.cc:158]   StreamExecutor device (0): <undefined>, <undefined>
2020-01-05 10:21:32.696140: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1433] Found device 0 with properties:
name: Tesla P100-PCIE-16GB major: 6 minor: 0 memoryClockRate(GHz): 1.3285
pciBusID: 0000:00:08.0
```

```
# Export in FP16 mode
!tlt-export $USER_EXPERIMENT_DIR/experiment_dir_retrain/weights/ssd_resnet18_epoch_030.tlt \
            -k $KEY \
            -o $USER_EXPERIMENT_DIR/export/ssd_resnet18_epoch_180_fp16.etlt \
            --outputs NMS \
            -e $SPECS_DIR/ssd_retrain_resnet18_kitti.txt \
            --data_type fp16 --export_module ssd
```

TRANSFER LEARNING TOOLKIT的工作流程

将模型转换成TensorRT Engine

```
# Convert to TensorRT engine
!tlt-converter -k $KEY \
               -d 3,384,1248 \
               -o NMS \
               -e $USER_EXPERIMENT_DIR/export/trt.engine \
               $USER_EXPERIMENT_DIR/export/ssd_resnet18_epoch_180.etlt
```

```
[INFO] After reformat layers: 88 layers
[INFO] Block size 1073741824
[INFO] Block size 122683392
[INFO] Block size 122683392
[INFO] Block size 46006272
[INFO] Block size 30670848
[INFO] Block size 11501568
[INFO] Block size 11501568
[INFO] Block size 2875392
[INFO] Block size 1966080
[INFO] Block size 1916928
[INFO] Block size 718848
[INFO] Block size 491520
[INFO] Block size 184320
[INFO] Block size 163840
[INFO] Block size 46080
[INFO] Block size 15360
[INFO] Total Activation Memory: 1427167232
[INFO] Detected 1 input and 2 output network tensors.
[INFO] Data initialization and engine generation completed in 0.0767436 seconds.
```

将模型转换成FP16格式的TensorRT Engine

```
# Convert to TensorRT FP16 engine
!tlt-converter -k $KEY \
               -d 3,384,1248 \
               -o NMS \
               -e $USER_EXPERIMENT_DIR/export/trt-fp16.engine \
               -t fp16 \
               $USER_EXPERIMENT_DIR/export/ssd_resnet18_epoch_180.etlt
```

```
[INFO] Block size 1073741824
[INFO] Block size 122683392
```

TRANSFER LEARNING TOOLKIT的工作流程

```
!tlt-converter -h
```

```
usage: tlt-converter [-h] [-v] [-e ENGINE_FILE_PATH]
        [-k ENCODE_KEY] [-c CACHE_FILE]
        [-o OUTPUTS] [-d INPUT_DIMENSIONS]
        [-b BATCH_SIZE] [-m MAX_BATCH_SIZE]
        [-w MAX_WORKSPACE_SIZE] [-t DATA_TYPE]
        [-i INPUT_ORDER]
        input_file

Generate TensorRT engine from exported model

positional arguments:
  input_file            Input file (.etlt exported model).

required flag arguments:
  -d            comma separated list of input dimensions
  -k            model encoding key

optional flag arguments:
  -b            calibration batch size (default 8)
  -c            calibration cache file (default cal.bin)
  -e            file the engine is saved to (default saved.engine)
  -i            input dimension ordering -- nchw, nhwc, nc (default nchw)
  -m            maximum TensorRT engine batch size (default 16)
  -o            comma separated list of output node names (default none)
  -t            TensorRT data type -- fp32, fp16, int8 (default fp32)
  -w            maximum workspace size of TensorRT engine (default 1<<30)
```

总结

- NVIDIA Transfer Learning Toolkit为深度学习训练部署流程提供了完整的工具链
- Transfer Learning Toolkit 的安装部署需要使用NGC
- 把训练和剪裁好的模型部署在边缘设备(Jetson 平台)上时，需要在边缘设备上转换成TRT的格式

https://developer.nvidia-china.com/forum.php?mod=viewthread&tid=11296&page=1&extra=#pid59795

更多资源：

https://developer.nvidia-china.com


THANK YOU
NVIDIA®