基于 PyTorch 的计算机视觉框架

东尼大佬 北京大学

导言

目前，深度学习技术已经广泛应用于计算机视觉的各个领域，Caffe、Tensorflow、PyTorch、MXNet 等深度学习框架已经逐渐成熟，大大简化了深度学习项目的研究和部署。本文主要介绍如何高效地设计和管理计算机视觉项目，抽象计算机视觉任务流程并进行高度模块化的开发和管理。将以开源项目 https://github.com/donnyyou/torchcv 为例进行详细阐述！

正文

1. TorchCV

1.1 TorchCV 整体介绍

TorchCV 是作者开发的基于 PyTorch 的计算机视觉框架，目前支持的任务包括图像分类、目标检测、图像分割、关键点定位以及生成对抗网络。主要复现了相关领域的经典论文方法，如目标检测相关算法中的 YOLOv3、SSD 和 Faster R-CNN、图像分割相关算法中的 PSPNet、DeepLabv3 和 DenseASPP、关键点定位领域的 OpenPose 以及生成对抗网络在图像生成方向的经典方法 Pix2Pix 和 CycleGAN 等等，其中大部分算法都能够达到论文效果，目前在进一步开发中。

1.2 TorchCV 流程抽象

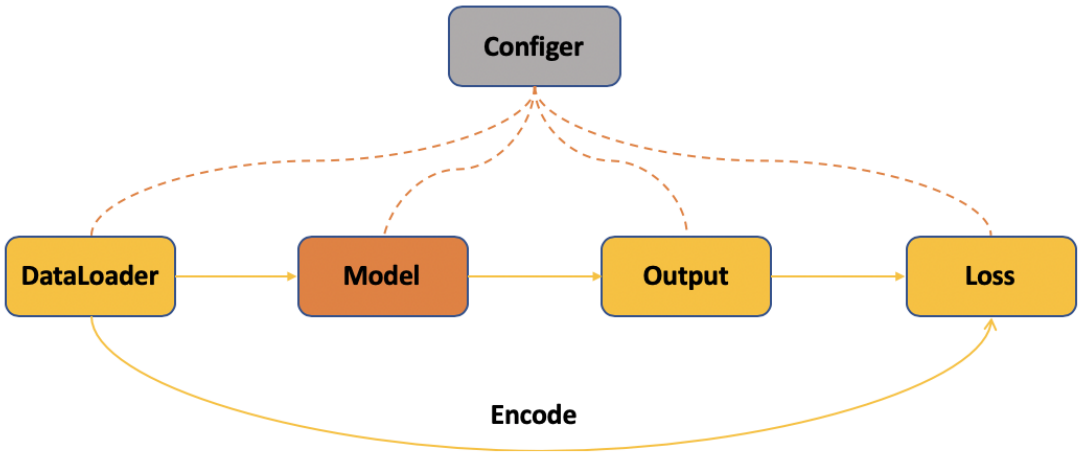

图 1: TorchCV training 过程抽象

训练过程主要包括数据读入、模型构建、模型输出以及计算损失值四个过程。其中 TorchCV 对每一种任务类型都定义了相应的数据格式，统一定义的数据增强方法以及可以自由选择的数据加载器。需要注意的是在计算损失的时候需要将 Ground Truth 编码成和模型输出对应的格式，然后计算预测和目标的损失值。

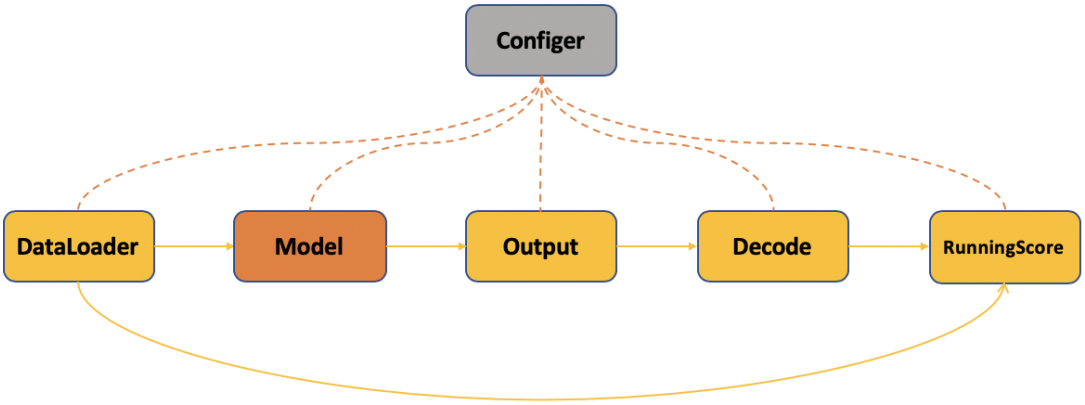

图 2: TorchCV validation 过程抽象

验证过程伴随着训练过程，其主要流程和训练过程相似，只是最后计算损失值变成了计

算模型效果，即对模型输出进行解码，生成和 Ground Truth 格式相同的结果计算训练过程中的模型的效果。

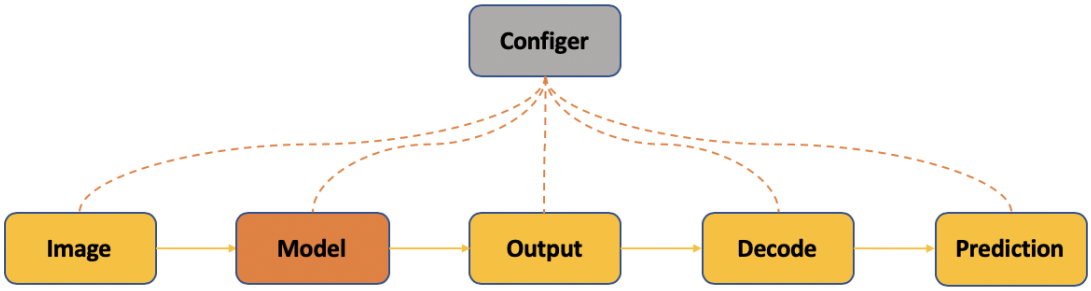

图 3: TorchCV testing 过程抽象

测试过程即图片作为输入，经过模型输出解码生成最后结果。其中解码过程即对模型的规则输出进行后处理生成我们需要的格式的结果。

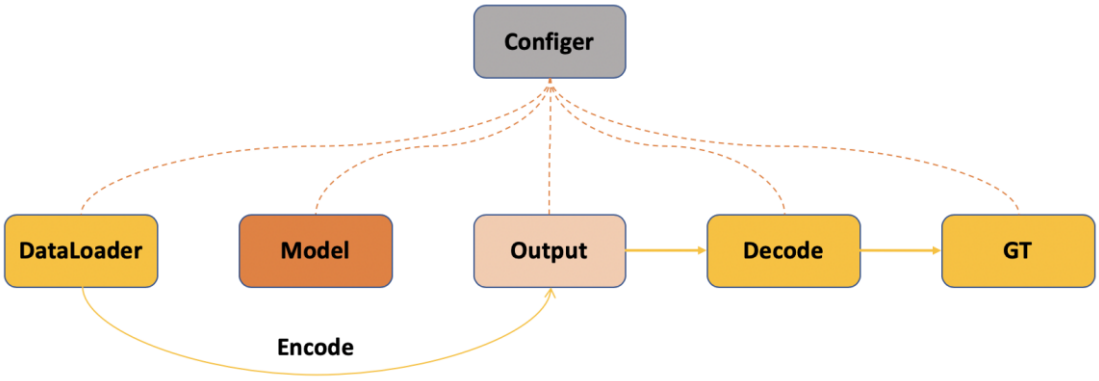

图 4: TorchCV debug 过程抽象

调试过程如图 4 所示，其中主要测试编码（Encode）和解码（Decode）过程的一致性，Ground Truth 通过编码使其与网络输出的格式一样，这样原来用来解码网络输出的部分代码就可以用来解码 Ground Truth 通过编码之后的结果，如果解码过程能够还原 Ground Truth，初步说明编码和解码过程在一定程度上是一致的和正确的。

2. 具体任务分析

2.1 图像分类

图像分类即对于一个输入图像输出对该图像内容分类的描述，图像的分类描述通过一个标签表示。目前主流的图像分类算法即通过卷积神经网络提取特征，然后对特征通过全连接进行分类。具体流程如图 5 所示，输入图像经过神经网络，对网络输出进行解码操作得到最后的类别，计算损失值的时候需要将 Ground Truth 的标签和网络输出对应上即可。

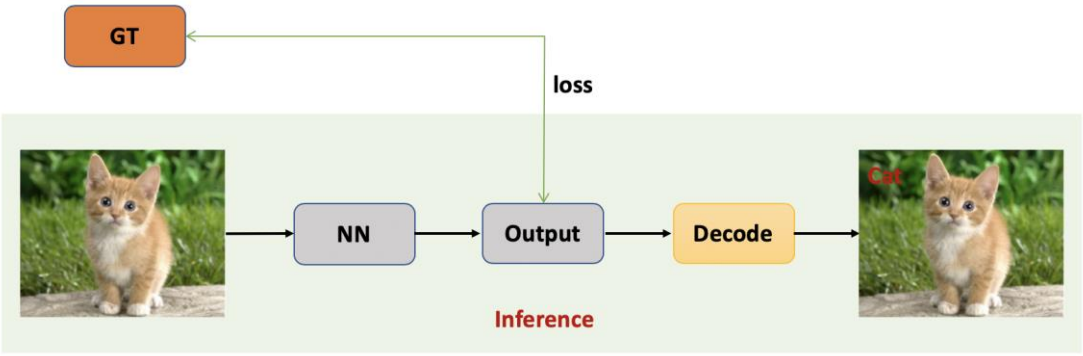

图 5: 图像分类算法流程图

目前对于图像分类的主要创新主要集中在如何构建更深更宽更密集连接的网络结构、如何尽可能在保持准确率的情况下减少参数和计算量以及如何通过设计更好的损失函数来学习到更有区分度的特征表示。

- Image Classifiers' Main Focus
  - Deeper & Wider & Denser. VGG vs. GoogLeNet vs. ResNet vs. DenseNet
  - Less Parameter & Less Computation. MobileNetv2 vs. ShuffleNetv2
  - Loss Function. Softmax vs. Center Loss vs. Triplet Loss vs. SphereFace vs. CosFace vs. ArcFace

2.2 目标检测

目标检测即找出图像中所有感兴趣的物体，包含物体定位和物体分类两个子任务，同时确定物体的类别和位置。目标分类任务负责判断输入图像或所选择图像区域（Proposals）

中是否有感兴趣类别的物体出现，输出一系列带分数的标签表明感兴趣类别的物体出现在输入图像或所选择图像区域中的可能性。目标定位任务负责确定输入图像或所选择图像区域中感兴趣类别的物体的位置和范围，输出物体的包围盒、或物体中心、或物体的闭合边界等，通常使用方形包围盒，即 Bounding Box 用来表示物体的位置信息。

目前主流的目标检测算法主要是基于深度学习模型，大概可以分成两大类别：（1）One-Stage 目标检测算法，这类检测算法不需要 Region Proposal 阶段，可以通过一个 Stage 直接产生物体的类别概率和位置坐标值，比较典型的算法有 YOLO、SSD 和 CornerNet；（2）Two-Stage 目标检测算法，这类检测算法将检测问题划分为两个阶段，第一个阶段首先产生候选区域（Region Proposals），包含目标大概的位置信息，然后第二个阶段对候选区域进行分类和位置精修，这类算法的典型代表有 R-CNN，Fast R-CNN，Faster R-CNN 等。

2.2.1 One-Stage 目标检测算法

One-Stage 目标检测算法可以在一个 stage 直接产生物体的类别概率和位置坐标值，相比于 Two-Stage 的目标检测算法不需要 Region Proposal 阶段，整体流程较为简单。如图 7 所示，在 Testing 的时候输入图片通过 CNN 网络产生输出，解码（后处理）生成对应检测框即可；在 Training 的时候则需要将 Ground Truth 编码成 CNN 输出对应的格式以便计算对应损失 loss。

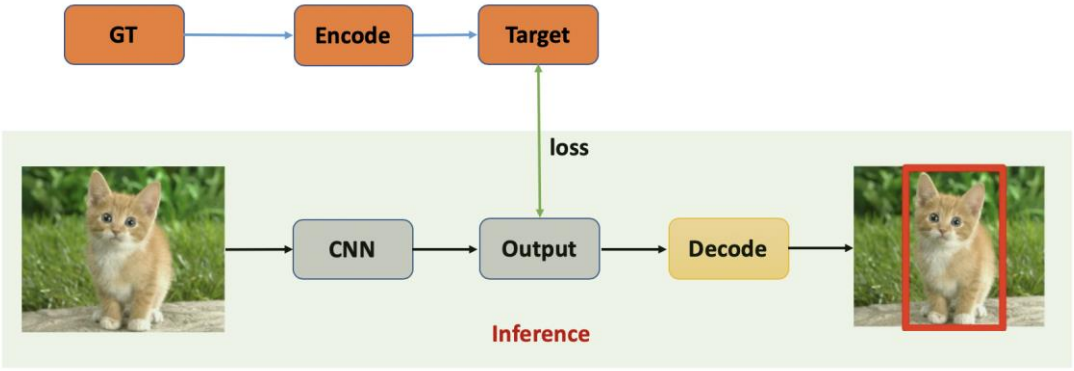

图 6: One-Stage 目标检测算法流程图

目前对于 One-Stage 算法的主要创新主要集中在如何设计更高效准确的 CNN 结构、如何更好地选择 Anchors，如何构建网络回归目标以及如何设计损失函数上。

- One-Stage Methods' Main Focus
  - More Efficient & Accurate Neural Network. SSD vs. DSSD vs. FSSD
  - How to Pre-select Anchors. YOLOv1 vs. SSD vs. YOLOv2-v3 vs. RefineDet
  - How to Construct Regression Targets. YOLO vs. SSD vs. CornerNet
  - How to Define Loss Functions. OHEM Loss vs. Focal Loss vs. ...

2.2.2 Two-Stage 目标检测算法

Two-Stage 目标检测算法本人认为可以看作是进行两次 One-Stage 检测，第一个 Stage 初步检测出物体位置，第二个 Stage 对第一个阶段的结果做进一步的精化，对每一个候选区域进行 One-Stage 检测。整体流程如图 9 所示，在 Testing 的时候输入图片经过卷积神经网络产生第一阶段输出，对输出进行解码处理生成候选区域，然后获取对应候选区域的特征表示（ROIs），然后对 ROIs 进一步经过卷积神经网络产生第二阶段的输出，解码（后处理）生成最终结果，解码生成对应检测框即可；在 Training 的时候需要将 Ground Truth 编码成 CNN 输出对应的格式以便计算对应损失 loss。

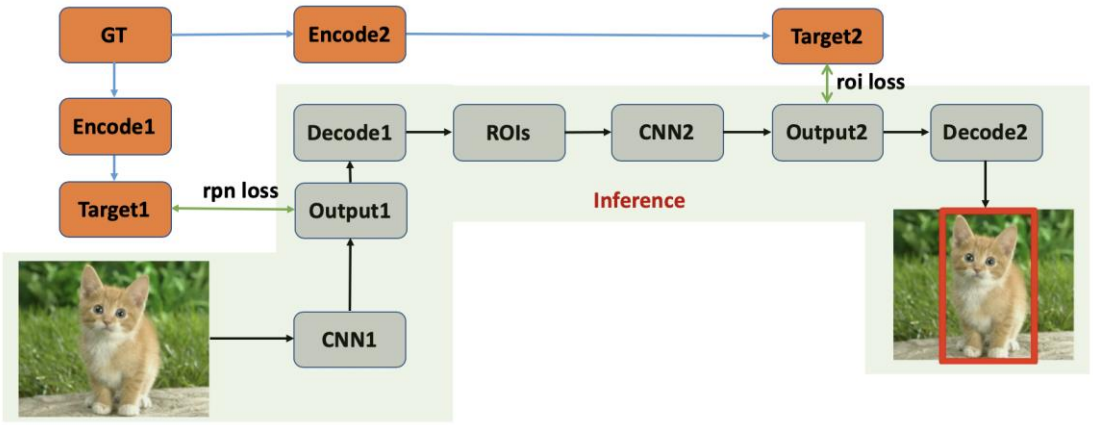

图 7: Two-Stage 目标检测算法流程图

如上图所示，Two-Stage 的两个阶段拆开来看均与 One-Stage 检测算法相似，所以我觉得 Two-Stage 可以看成是两个 One-Stage 检测算法的组合，第一个 Stage 做初步检测，剔除负样本，生成初步位置信息（Region of Interest），第二个 Stage 再做进一步精化并生成最终检测结果。目前对于 Two-Stage 算法的主要创新主要集中在如何高效准确地生成 Proposals、如何获取更好的 ROI features、如何 Align 获取到的 ROI features、如何加速 Two-Stage 检测算法以及如何改进后处理方法。

- Two-Stage Methods' Main Focus
  - How to Generate Proposals. R-CNN vs. Faster R-CNN vs. FPN vs. Cascade R-CNN
  - How to Obtain ROIs' Features. R-CNN vs. Fast/Faster R-CNN vs. FPN vs. R-FCN
  - How to Align ROIs' Features. ROI Pool vs. ROI Align vs. PSROI Pool vs. PrROI Pool
  - How to Accelerate Two-Stage Detector. R-CNN vs. Faster R-CNN vs. Light-Head R-CNN
  - Post-Processing methods. NMS vs. Soft NMS vs. IOU-Guided NMS

2.3 图像语义分割

图像语义分割即将图像分成若干具有相似性质的区域的过程，对每一个像素位置都会有一个类别标签。目前主流的图像语义分割算法都是基于 FCN 全卷积结构，输入一张图像，通过卷积神经网络对每一个像素位置提取上下文特征，然后对每一个像素位置进行分类得到最终结果。具体流程如图 12 所示，输入图像经过卷积神经网络，对网络输出进行解码操作得到最后的类别，计算损失值的时候需要将 Ground Truth 的标签和网络输出对应上即可。

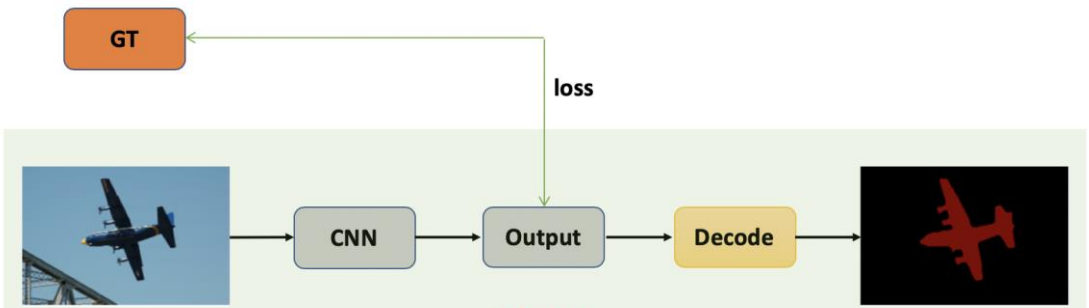

图 8：图像语义分割算法流程图

目前图像语义分割的主要创新主要集中在如下几点，即如何更高效地获取更大的感受野、如何更好地利用多尺度的信息、如何使用全局上下文信息、如何设计上采样方式以恢复分辨率、如何利用空间中像素之间的联系得到更平滑的像素特征、如何通过逐步求精的方式逐步获得更好的分割结果以及如何设计参数更少、速度更快，即高效的分割网络。

- FCN Segmentors' Main Focus
  - Larger Receptive Field. Standard Conv. vs. Dilated Conv.
  - Multi-Scale Information. ASPP vs. DenseASPP
  - Global Context. PSPNet vs. DeepLabv2 vs. DeepLabv3
  - Upsampling Method. Deconv. vs. Unpool vs. DUC vs. Interpolation
  - Spatial Correlation. Adaptive Affinity Fields vs. RelationNet
  - Coarse to Fine. G-FRNet vs. RefineNet (ICNet)
  - Less Parameter & Less Computation. ENet vs. ERFNet

2.4 人体关键点定位

人体骨骼关键点检测，即 Pose Estimation，主要检测人体的一些关键点，如关节，五官等，通过关键点描述人体骨骼信息；多人人体骨骼关键点检测主要有两个方向，一种是自上而下，一种是自下而上，其中自上而上的人体骨骼关键点定位算法主要包含两个部分，人体检测和单人人体关键点检测，即首先通过目标检测算法将每一个人检测出来，然后在检测框的基础上针对单个人做人体骨骼关键点检测，其中代表性算法有 G-RMI, CFN, RMPE, Mask R-CNN, and CPN；自下而上的方法也包含两个部分，关键点检测和关键点聚类，即

首先需要将图片中所有的关键点都检测出来，然后通过相关策略将所有的关键点聚类成不同的个体，其中对关键点之间关系进行建模的代表性算法有 Part Affinity Fields, Associative Embedding, Part Segmentation, Mid-Range offsets。

2.4.1 Bottom-up 人体关键点定位算法

自下而上（Bottom-Up）的人体骨骼关键点检测算法主要包含两个部分，关键点检测和关键点聚类，首先将图片中所有类别的所有关键点全部检测出来，然后对这些关键点进行聚类处理，将不同人的不同关键点连接在一块，从而聚类产生不同的个体。具体流程图如图 14 所示，在 Testing 的时候输入图片通过 CNN 网络产生关键点以及关键点之间的关系，然后对关键点通过关系进行聚类解码（后处理）生成对应个体即可；在 Training 的时候则需要将 Ground Truth 编码成 CNN 输出对应的格式以便计算对应损失 loss。

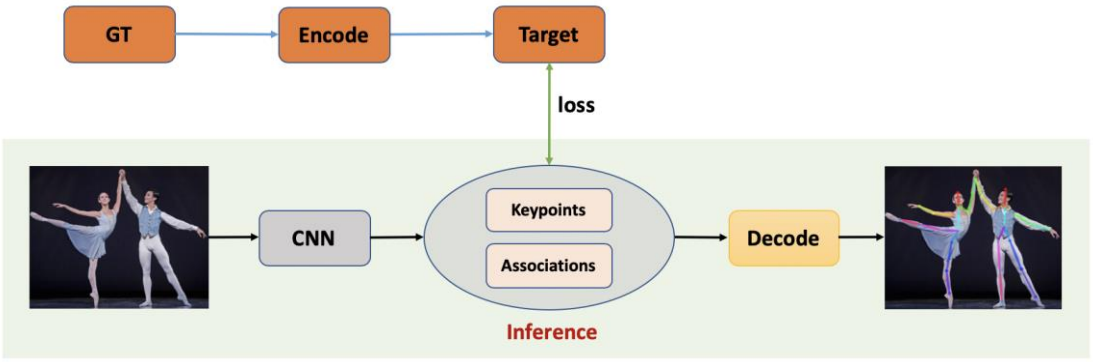

图 9：Bottom-up 关键点定位算法流程图

自下而上（Bottom-Up）的人体骨骼关键点检测算法的关键点检测和单人的关键点检测方法上是差不多的，区别在于这里的关键点检测需要将图片中所有类别的所有关键点全部检测出来，然后对这些关键点进行聚类处理，将不同人的不同关键点连接在一块，从而聚类产

生不同的个体。这方面的论文主要侧重于对关键点聚类方法的探索，即如何去构建不同关键点之间的关系。

- Bottom-Up Methods' Main Focus
  - How to Represent Keypoints. Coordinate vs. Heatmap + Offsets vs. Heatmap
  - How to Construct Part Associations. Part Segmentation vs. Part Affinity Fields vs. Associative Embedding vs. Mid-Range Offsets vs. Bounding Boxes

2.4.2 Top-down 人体关键点定位算法

自上而下（Top-Down）的人体骨骼关键点检测算法主要包含两个部分，目标检测和单人人体骨骼关键点检测，具体流程图如图 16 所示，首先通过目标检测算法获得对应的候选个体区域，然后对于候选区域通过卷积神经网络得到输出，最后对输出解码生成对应候选区域个体的关键点表示。在 training 的时候需要考虑目标检测的损失约束以及单人人体关键点的损失，对于目标检测的损失约束不再赘述，单人人体关键点的损失计算时需要将 Ground Truth 编码成对应格式。

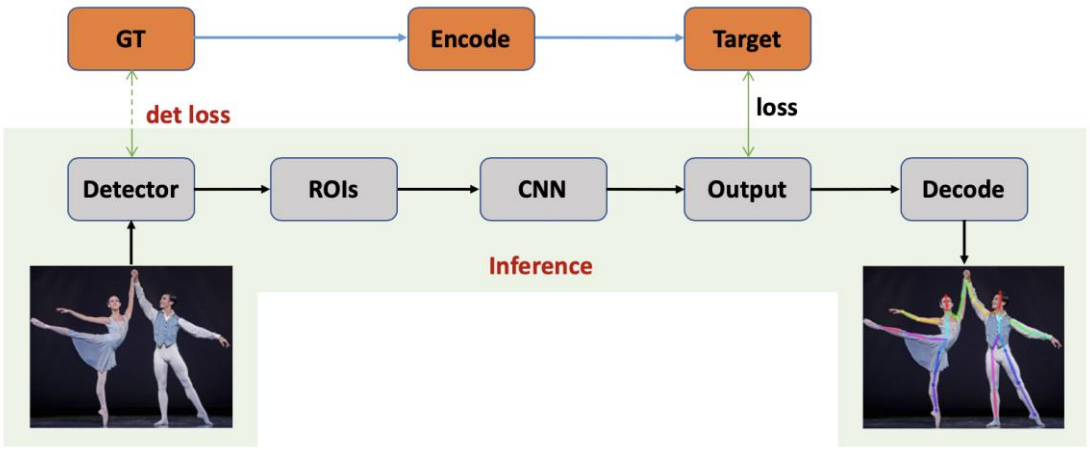

图 10：Top-down 关键点定位算法流程图

自上而下(Top-Down)的人体骨骼关键点检测算法比较依赖于目标检测算法的准确性，这里不再进行描述，然后再是单人的人体关键点定位问题，需要注意如下几个问题：首先关键点局部信息的区分性很弱，即背景中很容易会出现同样的局部区域造成混淆，所以需要考虑较大的感受野区域；其次人体不同关键点的检测的难易程度是不一样的，对于腰部、腿部这类关键点的检测要明显难于头部附近关键点的检测，所以不同的关键点可能需要区别对待；最后自上而下的人体关键点定位依赖于检测算法的提出的 Proposals，会出现检测不准和重复检测等现象，大部分相关论文都是基于以上这几个特征去进行相关改进。

- Top-Down Methods' Main Focus
  - More Robust Object Detector. SSD vs. Faster RCNN vs. YOLOv3
  - More Robust Single-Person Pose Estimation. CPM vs. RMPE vs. CPN vs. HRNet

小结

本文主要抽象介绍计算机视觉领域的相关任务，只有彻底地理解每个任务的相关细节，才能够更好地组织和开发，然而具体开发过程可能还需要有一定的调整。TorchCV 将所有方法集成在同一个框架中对于初学者并不是特别友好，但是对于部署多任务的系统比较方便快捷。欢迎持续关注 TorchCV，后期将持续开发整个框架，并将其切分成一些列的小框架，便于大家学习和理解！GitHub 链接：https://github.com/donnyyou/torchcv