车道线分割与强化学习在无人驾驶中的应用

陈其轩，高梓又，何金原，王启帆

摘　要

车道线分割与强化学习是自动驾驶系统开发中起到重要作用的技术。车道线分割技术使用语义分割模型读入车载摄像头画面，将其转换成一个具有明显特征的区域图像，从而检测出车道线在驾驶环境中所处的位置，辅助驾驶决策。强化学习算法则通过与环境的不断交互训练，得到在各类环境中进行自动驾驶的最优策略，代替人类进行驾驶决策。本文总结了车道线分割与强化学习在自动驾驶中应用的主流技术，并进行大量实验，通过数据总结各主流技术的优缺点。

关键词: 自动驾驶，语义分割，强化学习，车道线分割

I 语义分割及其在自动驾驶中的应用

1.1 语义分割的常见模型

语义分割是一种非常重要的计算机视觉技术，它可以用来实现图像识别、物体检测和语义分析等复杂任务。重新构建一个视觉掩膜，以便将原始的平面图像转换成一个具有明显特征的区域图像，这是一个关键的任务。在这个过程中，每个像素都会根据它们所属的视觉对象来进行分类，从而形成一个独特的区域图像。随着技术的进步，计算机视觉技术已经不再仅仅局限于检测线条、曲线等简单的图像边缘特征，而是可以更加深入地探索更多的信息，并且可以根据人类的视觉和感知，在像素级别上更好地表达出这些信息。语义图像分割算法是一种有效的技术，它能够将多个相似的图像元素进行分割，从而有效地解决图像分离的问题。它不仅能够提高图像分割的准确性，还能够扩展图像分割的应用范围，使得图像分割的技术变得更加广泛和实用。值得注意这一点关键的另一点则是，与其他一些或其他一些类型的基于语义图像信息的分析或任务系统分析相比，语义信息的图像分割算法则是在一个相对完全开放的多个不同技术领域之间且又更加灵活先进与有效的。

简而言之、语义分割遵循三个步骤：

1、分类：对图像中的某个对象进行分类。

2、定位：找到对象并在其周围绘制边界框。

3、分割：通过创建分割掩码对局部图像中的像素进行分组。

本质上，语义分割的任务可以被称为对某一类图像进行分类，并通过用分割掩码覆盖它来将其与其余图像类分开。它是一种像素级别的图像分类。

当我们要把任意一张像素图面上的某每一个像素点都重新进行颜色分类处理后，每一个新像素点都会同时有机会被再赋予上一个色彩类别。每个新的像素图都会被分配出一个特定的颜色类别，然后，我们可以通过重新分配该类别的像素来构建出一张完整的图片。在这张图片中，每一个元素都被重新分割，并且颜色也被重新串联起来，从而使得整张图片中的每一个物体都能够被清晰地呈现出来，并且保留了关于这类物体特征的几乎所有的图像语义信息。通过对图像的重新分割，我们不仅能够实现一次粗略的推断，还能够深入探究其中的规律，并将其转化为更加精确的结果。

1.1.1 全卷积网络（FCN）

全卷积网络（FCN）[1] 是一种仅执行卷积（以及子采样或上采样）操作的神经网络。等价的说，FCN 是没有完全连接层的 CNN。

2015 年,Jonathan Long, Evan Shelhamer, Trevor 和 Darrell 在 Fully Convolutional Networks For Semantic Segmentation[2] 一文中首次正式的提出来建立出了这样一种适用于语义分割和训练的全卷积网络。本文提出了一种全新的概念 FCN，它可以有效地进行分割语义，并且能够从端到端建立一个点对点的网络，从而实现 state-of-the-art 的模型。该研究首次将 FCN 应用于从端到端的预测，以达到像素级的效果；而且，这也标志着人类计算机第一次采用监督学习和预演训练的技术，从而更有效地实现 fcn 的学习与训练。" 完全卷积" 是一个系统设计，它旨在建立一个自动接收和处理任意大小输入的网络，通过密集的前向计算和反向

Figure 1

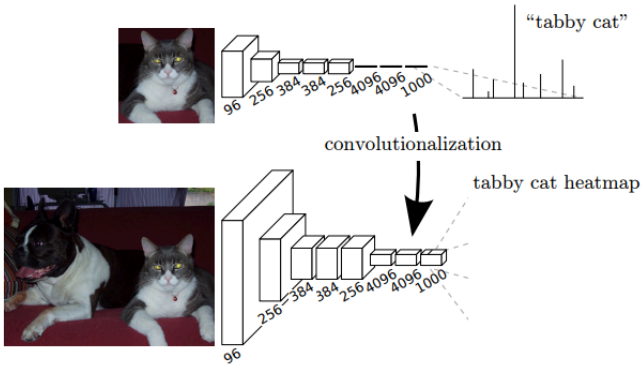


Figure 2

信息传播，实现更有效的推理判断和学习训练，从而产生相应的任意大小输出。在构建网络过程中，通过将这三个当代分类的网络 (AlexNet、VGG 网络和 GoogleLeNet) 调整为一个完全独立的卷积树网络，并将同时也通过微调将有关它们的学习和表示的信息传递给分割任务；

之后的定义又产生出了下一个分级结构，它还会重新将来自更高深层、粗层的语义信息与来自更深浅层、细层的外观信息重新有机的结合了一起来，以进一步确保产生出了更精确的语义和了更为详细的分段。

传统的 CNN 结构中主要还是由卷积层和与全的连接的层块组成，卷积连接层块的参数虽然和图像输入图像大小直接无关，但与全的连接的层块的参数往往就只与图像输入的图像大小直接有关了。如下所图中所示，卷积层利用输出向量的特征图 (feature map) 矩阵与全连接层中的一个权值参数矩阵进行先卷积之后再进行求和，每一次先卷积后得到的权值参数求和之后所得的值实际上就是全连接层中的其中一个元素，可以这样看出，如果网络输入的向量矩阵的维数都不太固定，那么整个全连接层得到的每个权值参数矩阵的参量值也是很不那么固定变化的，就会导致造成整个网络结构的动态变化，无法实现参数训练目的。FCN 通过删除另一个全连接层，并使用转置的卷积层对输入特征图像进行重新采样和处理，以恢复其原有的特征像素尺寸，从而实现了对输入特征图像的重建。因此，它可以被重新视为一种预测，

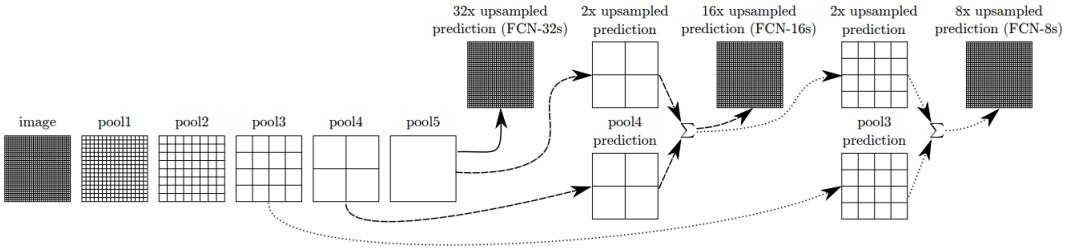


Figure 3

即每个特征像素尺寸都只产生出另一个输入特征，而且它们也能够保留住最初输入特征图像中所包含的空间信息。

尽管我们已经对 conv5 图中的反卷积核进行了操作和还原，但由于精度的限制，我们仍然无法完全准确地还原图像中的几乎所有特征。因此，我们需要进一步迭代，将 conv4 图中的反卷积核图与其他图进行比较，以便更好地操作和还原图像，并补充上一个细节。这个过程相当于插值，可以帮助我们更准确地操作和还原图像，从而提高图像的质量。通过在 Conv3 图中进行反卷积核，我们可以重新检查 upsampLing 后生成的图像，并且添加一些小细节，从而实现对整个图像的准确还原。具体点地来说，就是我们先去将不同的池化层的结果来分别来进行上一个采样，然后再在去结合上面的这些采样的结果再来进行做一些优化输出，分为 FCN-32s,FCN-16s,FCN-8s 这里面有三种，第一行对应于 FCN-32s，第二行是对应于 FCN-16s，第三行是则是对应于 FCN-8s。这便是实现精细分割的跳级结构，具体结构如图 Figure 3

FCN 的一个突出的优势也主要还在于：可以同时接受任意两种或任意多种空间大小类型的输入图像和输出图像，更加地紧凑与高效，避免漏掉了因为过度重复使用的邻域空间所带来的大量的重复计算、导致时间成本和空间费用的大幅浪费等严重的实际的问题。其最大缺点的不足在于：得到的图像的原始处理结果可能还会远而不够精细。进行高了 8 倍采样的原始上采样处理后虽然图像效果确实比原始的 32 倍采样出来的原始图像处理效果好得远了很多，但是从原始上采样处理出来得到的原始图像结果还是总体上明显的模糊和平细滑，对原始的图像结果中存在的某些细节也不感甚敏感。

1.1.2 U-net

U-net[3] 是一个用于医学图像分割的全卷积神经网络。目前很多神经网络的输出结果都是最终的分类类别标签，但对医学影像的处理，医务人员除了想要知道图像的类别以外，更想知道的是图像中各种组织的位置分布，而 U-net 就可以实现图片像素的定位，该网络对图像中的每一个像素点进行分类，最后输出的是根据像素点的类别而分割好的图像。因为 U-net 的操作过程如“U”形，故而称为 U-net，其具体结构如图 Figure 4

U-Net 网络非常简单，前半部分（左边）作用是特征提取，后半部分（右边）是上采样。在一些文献中也把这样的结构叫做 Encoder-Decoder 结构。因为此网络整体结构类似于大写的英文字母 U，故得名 U-Net。

每个蓝色框对应一个多通道特征图，其中通道数在框顶标。x-y 的大小位于框的左下角。灰色框表示复制和裁剪（Concat）的特征图。箭头表示不同的操作。

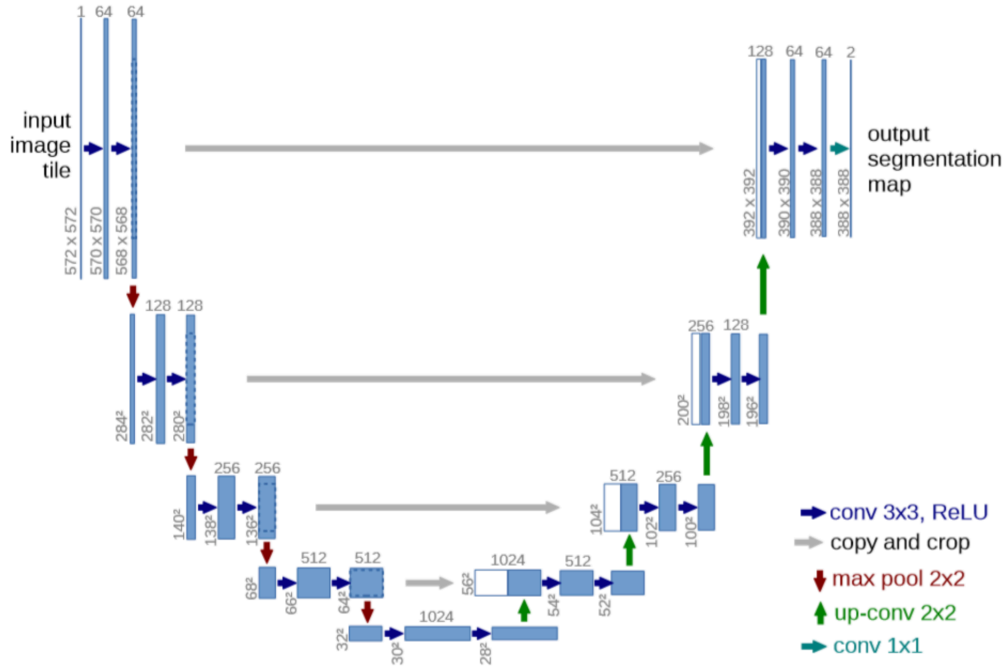


Figure 4

该网络由收缩路径（contracting path）和扩张路径（expanding path）组成。其中，收缩路径用于获取上下文信息（context），扩张路径用于精确的定位（localization），且两条路径相互对称。

语义分割网络在特征融合时有两种办法：

1、FCN 式的对应点相加。

2、U-Net 式的 channel 维度拼接融合。

除了上述新颖的特征融合方式，U-Net 还有以下几个优点：通过使用五个 pooling 和 layer，我们可以在网络中实时地识别图像中不同尺度的特征。通过上采样，我们可以将两个不同尺度的特征信息融合到一个整体中，从而更好地理解图像中的细节。最后一个上采样作为一个反例，其所携带的特征信息不仅可以从第一个卷积 block 的输出（同尺度特征）获取，还可以从第一个上采样的输出（大尺度特征）获取，而且这种联系必须完全融入到整个网络架构当中，才能发挥最大的效用。你可以通过直观地看到在上图中的网络结构中将会包含有至少一个四次以上的融合的过程，相对应的是 FCN 网络只能够在网络的最后的那一层之间才能进行融合。

U-Net 也有两点明显的不足：

1、该网络运行效率很慢。对于每个邻域，网络都要运行一次，且对于邻域重叠部分，网络会进行重复运算。

2、网络需要在精确的定位和获取上下文信息之间进行权衡。越大的 patch 需要越多的最大池化层，其会降低定位的精确度，而小的邻域使得网络获取较少的上下文信息。

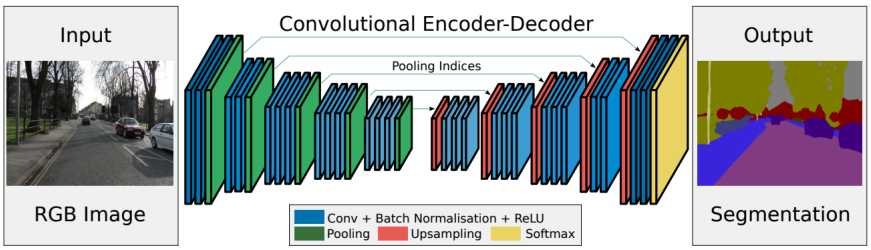


Figure 5

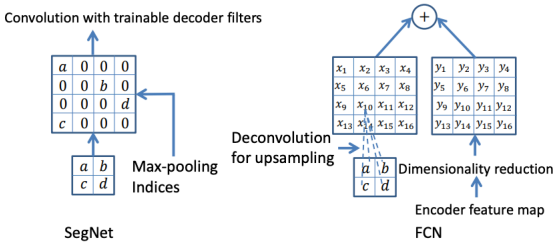


Figure 6

1.1.3 SegNet

SegNet[4] 是一种用于训练分割引擎的神经网络，它由一个编码器网络、一个与之相连的解码器网络以及一个像素级的分类架构层网络组成，这些网络的拓扑结构大致相似，可以用来模拟 VGG16 网络的第十三个卷积架构层。通过解码器网络，可以将低分辨率的特征图转换为高分辨率的特征图，这样就可以有效地提升分类效率，而无需进行复杂的编码操作。SegNet 的创新功能可以归结为它采用了一种独特的方法，即将原本需要较低输入的特征图网络转换为一个更高的输出分辨率，从而实现线性采样。换句话说，segnet 利用最大池化步骤表中可计算的最小索引，来取代传统的非线性采样，从而提高了解码器的效率和准确度。这在实际上就消除掉了直接学习的线性上的采样的方法上的需要。上的采样特征图首先应该至少是相对于稀疏一点的，然后才能通过与任何一个可用于训练的过滤器一起来进行卷积和计算以帮助自动的生成出更为密集的采样特征图。解码器网络的主要目的在于将低分辨率的特征图转换成高分辨率的特征图，从而使得它们能够更容易地被分类，而不需要复杂的编码过程。SegNet 的创新功能可以归结为它采用了一种独特的方法，即将原本需要较低输入的特征图网络转换为一个更高的输出分辨率，从而实现线性采样。换句话说，segnet 利用最大池化步骤表中可计算的最小索引，来取代传统的非线性采样，从而提高了解码器的效率和准确度。

FCN 网络利用卷积层以及其它一些跳跃连接构建出一个粗略的分割图，但为了获得更好的性能，它还将更多的跳跃连接纳入其中。相比之下，Segnet 网络将编码器特征转换成了最大的电池化指数，从而实现了更高的性能。SegNet 网络的三大优势使其能够超越 FCN 网络，从而实现对内存的极大可靠性和极高的性能。SegNet 的网络框架如图 Figure 5 所示

SegNet 和 FCN 的主要区别在于解码器（如图 Figure 6）

尽管 SegNet 在评价指标不如 FCN-8s 效果好，但其提出的编码-解码的思想影响着后面

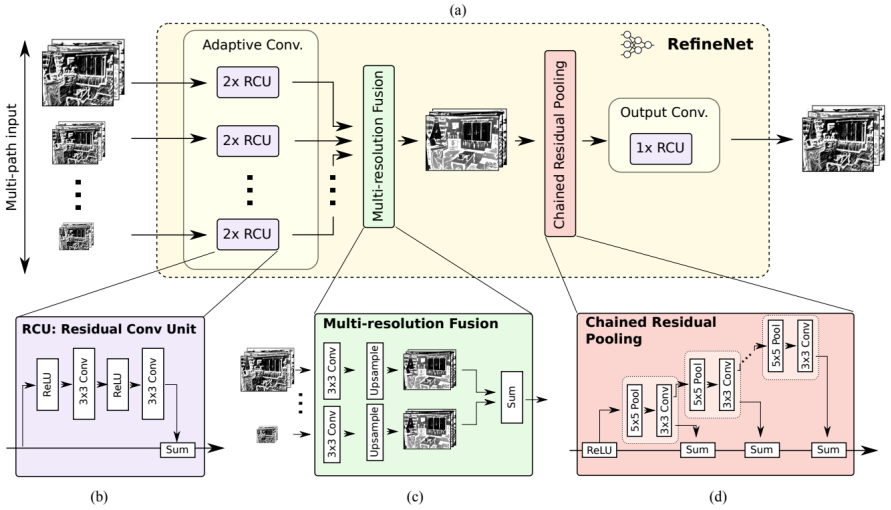


Figure 7

的很多模型。

1.1.4 RefineNet

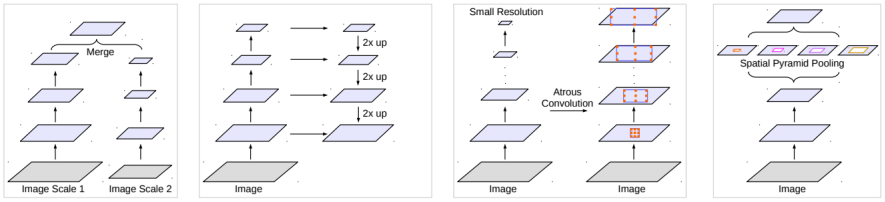
Guosheng Lin, Anton Milan, Chunhua Shen, Ian Reid 等人在 RefineNet: Multi-Path Refinement Networks for High-Resolution Semantic Segmentation 一文中提出了 RefineNet 这种结构

随着 CNN 的不断发展，涌现了很多深度较深的 CNN 如 ResNet 系列，它们非常适合用于完成稠密分割任务，如语义分割。但是由于 CNN 需要反复地进行下采样，这样导致了图像分辨率不断地降低，容易丢失了图像的一些空间信息，这样对于一些高分辨率的图像就非常不友好了。针对这个问题，作者提出一种 RefineNet[5]，引入了残差卷积模块（Residual Convolution Unit）[6]、多分辨率融合模块（Multi-Resolution Fusion）和串联残差池化模块（Chained Residual Pooling）等结构，非常有效地对空间分辨率进行恢复，在 7 个数据集中均达到 SOTA。

RefineNet 的结构如图 Figure 7 所示

作者提出的 RCU 模块参考了 ResNet 的残差块，在模块内分成两条线路，主干线为图像直接的输入，而支线的图像先经过 ReLU、3x3 卷积、ReLu、3x3 卷积，再与主干线路进行特征融合叠加，残差卷积部分可以理解为对特征图进行信息的补充，使得图像信息更加丰富。图像通过残差卷积模块以后便要进入 MRF 模块，MRF 模块主要是对不同尺度的图像进行特征提取和上采样到同样的分辨率，最后进行融合。

不同尺度的图像都进入对应的通道进行 3x3 卷积，再进行一个双线性插值法的上采样，不同通道的图像最终都上采样成同一分辨率的图像，最终进行融合叠加，将结果送往下一层。该网络用在目标解析方面表现良好。



(a) Image Pyramid (b) Encoder-Decoder (c) Deeper w. Atrous Convolution (d) Spatial Pyramid Pooling

Figure 2. Alternative architectures to capture multi-scale context.

Figure 8

1.1.5 DeepLab v3

随着空洞卷积的扩张率的增大，卷积核中有效的权重越来越少，因为随着扩张率的变大，会有越来越多的像素点的计算没法使用全部权重。当扩张率足够大时，只有中间的一个权重有作用，这时空洞卷积便退化成了卷积。这里丢失权重的缺点还是其次，重要的丢失了图像全局的信息。

为了解决这个问题，DeepLab v3 参考 ParseNet[8] 的思想，增加了一个由来提升图像的全局视野的分支。具体的说，它先使用 GAP 将 Feature Map 的分辨率压缩至，再使用卷积将通道数调整为，最后再经过 BN 以及双线性插值上采样将图像的分辨率调整到目标分辨率。因为插值之前的尺寸是，所以这里的双线性插值也就是简单的像素复制。

DeepLab v3 的另外一个分支则是由 1 个卷积核三个扩张率依次为的空洞卷积组成。最后两个分支通过拼接操作组合在一起，再通过一个卷积将通道数调整为 1。

与在 DeepLab v2 网络、空洞卷积中一样，这项研究也用空洞卷积/多空卷积来改善 ResNet 模型。

这篇论文还提出了三种改善 ASPP 的方法，涉及了像素级特征的连接、加入 1Œ1 的卷积层和三个不同比率下 3Œ3 的空洞卷积，还在每个并行卷积层之后加入了批量归一化操作。

级联模块实际上是一个残差网络模块，但其中的空洞卷积层是以不同比率构建的。这个模块与空洞卷积论文中提到的背景模块相似，但直接应用到中间特征图谱中，而不是置信图谱。置信图谱是指其通道数与类别数相同的 CNN 网络顶层特征图谱。

该论文独立评估了这两个所提出的模型，尝试结合将两者结合起来并没有提高实际性能。两者在验证集上的实际性能相近，带有 ASPP 结构的模型表现略好一些，且没有加入 CRF 结构。

这两种模型的性能优于 DeepLabv2 模型的最优值，文章中还提到性能的提高是由于加入了批量归一化层和使用了更优的方法来编码多尺度背景。

1.2 语义分割在自动驾驶中的应用

近年来，随着深度学习技术的快速发展，计算机视觉领域中的许多使用传统方法难以解决的任务都取得了巨大的突破。特别是在图像语义分割领域，深度学习技术的作用表现尤为突出。图像语义分割作为计算机视觉中一项基础且具有挑战性的任务，其目标是将对应的语义标签分配给图像中的每个像素，其结果是将给定图像划分为若干视觉上有意义或感兴趣的

区域，以利于后续的图像分析和视觉理解。

尽管存在着上述各种各样的困难，语义分割技术仍因其巨大的不可替代的价值，成为自动驾驶技术栈中不可或缺的一部分。自动驾驶技术中的许多地方都需要使用到语义分割技术。比如车道线识别中，毫末智行的感知算法工程师们就使用了语义分割技术来识别车道线的位置和轮廓。如下图所示，使用语义分割得到的车道线，相较于其他方法，有更加清晰的边缘，准确率和召回率也高很多。

1.2.1 自动驾驶的感知系统

无人驾驶汽车中的感知系统必须具有以下特性：

(1). 准确性：需要提供精确的驾驶环境信息；(2). 鲁棒性：在恶劣的天气下、在训练过程中没有覆盖的情况下 (开放条件)，以及在一些传感器退化甚至有缺陷的情况下，都应该正常工作；(3). 实时性：尤其是汽车高速行驶时。为了实现这些目标，自动汽车通常配备了多模态传感器（如摄像头、激光雷达、雷达），并将不同的传感方式进行融合，从而利用它们的互补特性（参见第二节-A）。

此外，深度学习在计算机视觉方面也取得了很大的成功，深度神经网络是一种强大的工具，可以在给定大量数据的情况下学习层次化的特征表示 [5]。在这方面，已经提出了许多采用深度学习融合多模态传感器的方法，以实现自动驾驶中的场景理解。图 2 显示了最近公布的一些方法和他们在 KITTI 数据集上的表现 [6]。所有性能最高的方法都是基于深度学习，许多融合摄像头和激光雷达信息的方法比单独使用激光雷达或摄像头的方法产生更好的性能。在本文中，我们重点研究两个基本的感知问题，即目标检测和语义分割。在本文的其余部分，除非另有提及，否则我们将把它们称为深度多模态感知。

1.2.2 基于语义分割的车道线识别

我们在 aistudio 上训练了一个基于飞桨 PaddleSeg 的自动驾驶道路分割模型，采用实时语义分割网络 BiSeNet，使用多个经预处理的训练集进行训练，并在训练结束后从多角度对模型进行了评估与可视化。

本项目共挂载 4 个数据集。由于 AI Studio 对挂载数据集数量的限制（最多可挂载 2 个数据集），我们将其中三个数据进行处理制作为一个大型数据集集合并进行挂载。所用到的数据集分别为：

车道线检测-初赛 https://aistudio.baidu.com/aistudio/datasetdetail/54076

车道线检测数据集 https://aistudio.baidu.com/aistudio/datasetdetail/68698

2020 中国华录杯 · 数据湖算法大赛—定向算法赛（车道线识别）

https://aistudio.baidu.com/aistudio/datasetdetail/54289

智能车 2022baseline 数据集 https://aistudio.baidu.com/aistudio/datasetdetail/125507

基于轻量化网络模型的设计作为一个热门的研究方法，许多研究者都在运算量、参数量和精度之间寻找平衡，希望使用尽量少的运算量和参数量的同时获得较高的模型精度。目前，轻量级模型主要有 SqueezeNet、MobileNet 系列和 ShuffleNet 系列等，这些模型在图像分类

领域取得了不错的效果，可以作为基本的主干网络应用于语义分割任务当中。

然而，在语义分割领域，由于需要对输入图片进行逐像素的分类，运算量很大。通常，为了减少语义分割所产生的计算量，通常而言有两种方式：减小图片大小和降低模型复杂度。减小图片大小可以最直接地减少运算量，但是图像会丢失掉大量的细节从而影响精度。降低模型复杂度则会导致模型的特征提取能力减弱，从而影响分割精度。所以，如何在语义分割任务中应用轻量级模型，兼顾实时性和精度性能具有相当大的挑战性。

BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation 中出了一种新的双向分割网络 BiSeNet。首先，设计了一个带有小步长的空间路径来保留空间位置信息生成高分辨率的特征图；同时设计了一个带有快速下采样率的语义路径来获取客观的感受野。在这两个模块之上引入一个新的特征融合模块将二者的特征图进行融合，实现速度和精度的平衡。

此外，我们还对集中常用的语义分割网络的性能作了对比，数据见后。

1.2.3 常见的语义分割网络的性能对比

用于评估语义分割算法性能的标准指标是平均 IoU（Intersection over Union，交并比），IoU 定义如下：

$$IoU = \frac{Area\quad of\quad Overlap}{Area\quad of\quad Union} = \frac{Area_{pred} \cap A_{true}}{Area_{pred} \cup A_{true}}$$

我们在 PASCAL VOC 2012 test 数据集上复现了若干著名论文中的结果, 对比如下

Model	MeanIoU
FCN	62.2%
Dilated convolutions	67.6%
Deeplabv3+	89.0%
DANet	82.6%
Multipath-RefineNet	84.2%

II 自动驾驶应用中的强化学习

强化学习 [7] 作为一类能够实现通用智能解决复杂问题的人工智能学习方法，在自动驾驶领域当中扮演着重要角色。当下主流自动驾驶技术可划分为三个部分 [8]：利用环境感知技术采集现实驾驶场景信息，使用计算机重建仿真驾驶场景 [9][10]；在仿真场景（如 Gym、Pybullet）中结合强化学习与深度学习方法进行大量的训练迭代 [10]，得到高可靠性的行为决策系统，用于对传入环境制定行驶策略；最后利用运动控制技术控制车辆的行为 [11]。

当然，也有学者借助不同思路来进行强化学习模型训练。随着当今对自动驾驶安全性要求不断提高，电脑仿真场景中进行的训练可靠性难以保证，与此同时真实车辆进行训练又面临成本过高的问题，一些学者选择直接将输入改为驾驶场景或高度仿真场景的图像，进而得到在现实场景中泛用性强的端到端强化学习模型 [12][13]。也有学者使用现实翻译网络等技术 [14][15]:，将环境中的虚拟场景翻译成真实场景，避免了虚拟到真实环境的模型迁移，有时能够获得泛化能力更好的强化学习模型。

依照主流思想，本文的算法原理介绍与性能测试均借助计算机仿真场景 (Gym、Pybullet) 进行。强化学习方法应用于自动驾驶领域具有相当的时间，类似的工作早在 2012 年就在进行 [16]（若将范围放宽至非实时的机器人路径规划，最早的尝试应出现于 2004 年 [17]）：学者通过使用模仿学习（imitation learning）对无人机进行自主控制、导航及避障。模仿学习是一种与强化学习类似的、由人类专家提供数据的学习方法，在早年的自动驾驶技术开发当中，模仿学习与强化学习相结合的方法 [18][19] 是十分常见的，时至今日，模仿学习仍被用于提高智能体在探索阶段的效率 [20][21]。在早年的技术开发中，也有学者使用计算机视觉与深度学习进行自动驾驶技术开发 [22]。2013 年，DQN[23] 开创性地将深度网络与强化学习结合在一起，借助深度学习的感知能力极大提升智能体与环境交互效率，形成了深度强化学习方法 (DRL)，至今仍然指导着自动驾驶主流技术开发。在本章节，我们将按照时间顺序，对各个时期应用于自动驾驶领域的强化学习主流技术算法进行介绍。

2.1 DP&MC&TD

动态规划算法（DP）、蒙特卡洛算法（MC）与时间差分算法（TD）是强化学习方法中最为传统的算法，在这其中，时间差分算法是前二者结合产生的。

传统强化学习类算法在自动驾驶技术开发领域的应用中面临着许多几乎无法克服的困难，其中最为显著的是维数灾难：传统强化学习方法无法高效求解大型连续状态问题，也无法高效求解连续控制问题。然而，自动驾驶问题是状态与控制连续型问题，故传统强化学习直接应用于自动驾驶技术开发十分困难。但如果采用全宽和采样备份（Full-Width Backups and Sample Backups）等较大的改进，算法可以有效避免维数灾难的增加，随着状态的增加复杂度仅常数增长。事实上，近年确实有一些学者在这一方面进行了研究 [24][25]。

尽管如此，我们仍可以将传统强化学习算法原理当作预备知识。事实上，这些预备知识中的内容即是现代主流强化学习方法中的基石，十分有了解的必要。

2.1.1 Reinforcement Learning-Elements

我们首先对强化学习模型中包含的元素及其运行过程进行介绍：

- 智能体（Agent）：智能体是价值函数、策略与模型的组合，能够对环境进行感知，并采取相应的策略，以使得自己的价值函数达到最高。在自动驾驶技术开发中，我们往往选择自动驾驶汽车作为智能体。
- 环境（Environment）：环境是我们需要进行解决的问题本身所在场景的环境仿真，它会随着智能体采取不同的行动发生改变，并将改变后的状态（State）返回给智能体。环境所包含的内容是由算法设计者决定的，一般的环境设计会包含设计者认为与决策相关的内容。在自动驾驶技术开发中，这往往会包括路线、地形、天气乃至于相关的交通规则与法律法规。
- 策略（Policy）：策略在早期强化学习算法中，是环境到行动的映射。也即是说，策略是智能体处于某一环境时，根据环境进行行动决定的方法。自动驾驶汽车在行驶过程中依照策略来进行转弯、加速、减速等动作。

- 奖励（Reward）：奖励是对智能体在特定环境下采取的行动的评价。
- 价值函数（value function）：价值函数是智能体从当前环境开始至未来所能够获得的奖励综合，价值函数也是强化学习模型的目标函数。
- 环境模型：如果算法是 model-based，强化学习模型还应包含环境的模型。

2.1.2 Finite Markov Decision Process

在强化学习过程中，智能体目前所处状态取决于上一状态与所采取的行动，而与其他因素无关，是标准的马尔可夫过程。而强化学习中的有限马尔科夫决策过程指的是由状态、动作与奖励构成的三元组：$< S, A, R >$ 中的每个元素个数都是有限的。

当智能体处于时间步 t 时，目前所处的状态记作 S_t，采取的动作记作 A_t，对应的奖励记作 R_t。在传统强化学习方法中，这些随机变量都是离散的。

形式化地，有限马尔可夫决策过程可以表示如下：

$$p(s', r|s, a) = P\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a)\}$$

我们可以通过上式得到对于传统强化学习模型的关键——状态转移方程：

$$p(s'|s, a) = \sum_{r \in R} P\{S_t = s'|S_{t-1} = s, A_{t-1} = a)\}$$

并由此得到状态-动作二元组与状态-动作-状态三元组的奖励期望计算公式：

$$r(s, a) = E[R_t|S_{t-1} = s, A_{t-1} = a] = \sum_{r \in R} r \sum_{s' \in S} p(s', r|s, a)$$

$$r(s, a, s') = E[R_t|S_{t-1} = s, A_{t-1} = a, S_t = s'] = \sum_{r \in R} r \frac{p(s', r|s, a)}{p(s'|s, a)}$$

借助强化学习有限马尔可夫决策过程的概念，我们可以形式化地描述智能体与环境交互的过程：

$$S_0 \to A_0 \to R_1 \to S_1 \to A_1 \to R_2 \to S_2 \to A_2 \to R_3 \to ...$$

也就是说，强化学习模型的训练过程就是不断地根据当前环境选取合适的策略，进而完成对环境的感知与策略制定的过程。

2.1.3 Action-Value Method&Monte Carlo

通过估计智能体在任一状态下所有动作的奖励预期，以进行动作选择决定的方法，称作动作价值方法。

智能体的任一动作 a 都有与之对应奖励期望值 $q_*(a)$，由于状态与动作空间离散且有限，这一期望是一定存在的。然而，在训练开始前，智能体并不知道 $q_*(a)$ 的准确值。智能体只能通过不断”试错搜索“来得到数据，估算得到 $Q_t(a)$，而我们希望 $Q_t(a)$ 越接近 $q_*(a)$ 越好。

为此，我们需要定义超参数 $\alpha_{step-size}$，使用以下公式计算 $Q_t(a)$：

$$\begin{aligned} Q_{n+1}(a) &= Q_n + \alpha_{step-size}(R_n - Q_n) \\ &= \alpha_{step-size}R_n + (1-\alpha_{step-size})Q_n \\ &= (1-\alpha_{step-size})Q_1 + \sum_{i=1}^{n} \alpha_{step-size}(1-\alpha_{step-size})^{n-i}R_i \end{aligned}$$

特别地，如果令 $\alpha_{step-size} = \frac{1}{n}$，那么由大数定律，$Q_t(a)$ 一定能够收敛至 $q_*(a)$。

事实上，我们将公式换成以下伪代码中的形式，就得到了 Monte Carlo 采样方法，也就是 MC 方法。值得注意的是，只有当一个 Episode 完成时，$Q(A)$ 才会被更新，所以 Monte Carlo 采样是一种典型的离线（offline）算法。

在进行采样的过程中，我们还会采用 $\varepsilon - greedy$ 方法，以鼓励智能体选择此前采取次数较少的行动进行探索。除 $\varepsilon - greedy$ 方法以外，还有 Upper-Confidence-Bound（UCB）与 Gradient method 等方法被应用于使智能体更好的进行探索采样 [26]，此处我们给出较简单的 UCB 方法的形式化表达：

$$A_t = \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$

Algorithm 1: Action-Value Method/Monte-Carlo Method

Input: ε, c

Initialize for $a = 1, 2, \ldots, k$:
    $Q(a) \leftarrow 0$;
    $N(a) \leftarrow 0$;// Episide number;
while $N(a) < c$;
do
    random $\leftarrow$ rand(0,1);
    if random $< \varepsilon$ **then**
        $A \leftarrow \text{argmax}(x_a Q(a))$;
    else
        $A \leftarrow$ a random action;
    end
    $R \leftarrow \text{bandit}(A)$;
    $N(A) \leftarrow N(A) + 1$;
    $Q(A) \leftarrow A(A) + \frac{1}{N(A)}[R - Q(A)]$;
end

2.1.4 Bellman equation&Temporal-Difference

我们将处于时间步 t 时，智能体将来能够得到的奖励总和记作 $G_t = \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$，其中可取 $T = \infty$ 或 $\gamma = 1$（不能同时取）。$\gamma = 1$ 适用于如走迷宫等有结局的情节性任务（episodic task），$0 < \gamma < 1$ 则适用于无限执行的连续性任务（continuing task）。

由于强化学习过程为马尔可夫决策过程，状态价值函数与动作价值函数可计算如下：

$$v_\pi(s) = E_\pi[G_t|S_t = s] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s\right]$$

$$q_\pi(s,a) = E_\pi[G_t|S_t = s, A_t = a] = E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1}|S_t = s, A_t = a\right]$$

v_π 与 q_π 的实际测定可以通过蒙特卡洛方法进行。

联立上式可以得到 v_π 的贝尔曼方程：

$$\begin{aligned} v_\pi(s) &= \mathbb{E}_\pi\left[G_t \mid S_t = s\right] \\ &= \sum_a \pi(a \mid s) \sum_{s'} \sum_r p\left(s', r \mid s, a\right)\left[r + \gamma \mathbb{E}_\pi\left[G_{t+1} \mid S_{t+1} = s'\right]\right] \\ &= \sum_a \pi(a \mid s) \sum_{s', r} p\left(s', r \mid s, a\right)\left[r + \gamma v_\pi\left(s'\right)\right], \quad \text{for all } s \in \mathcal{S}. \end{aligned}$$

利用贝尔曼方程的思想，时序差分方法可以通过自举法（bootstrapping）直接用自己的下一个状态估计来更新当前状态的估计，而不需要像蒙特卡洛采样方法等待整个 Episode 完成。

Algorithm 2: Temporal-Difference

Input: ε, c, α, π

$\forall s \in S$, Initialize V(s), V(terminal)=0;
for e in episodes **do**
  Initialize S;
  for t in episode **do**
    $A \leftarrow$ action given by π for S;
    $R \leftarrow$ Reward(A, S);
    $S' \leftarrow$ State(A, S);
    // Take action A and observe R,S';
    $V(S) \leftarrow V(S) + \alpha[R + \gamma V(S') - V(S)]$;
    $S \leftarrow S'$;
  end
end

2.1.5 Reinforcement Learning-Features

传统强化学习方法是机器学习的重要分支，在方法发展的初期，其内部各算法之间具有许多共性，我们将这些特征罗列如下：

- 试错搜索：强化学习是典型的无监督学习方法，智能体并未被告知在当前环境下需要采取什么动作，只能通过不断采取行动进行“试错”来获得在某一状态中采取某一动作的所能获得的奖励（reward）。
- 延迟奖励：根据算法设计的不同，智能体所采取的动作未必能够得到一个即时的反馈，但这一动作往往会对下一个状态与往后的奖励造成影响。
- 无监督性：在传统的强化学习的算法设计中，我们往往只对智能体的动作进行奖励反馈，而不是指导智能体的动作（模仿学习方法 [27] 则会指导智能体的动作），尽管这使得模型的训练时间变得漫长，但往往能够得到设计者本身都难以发现的优秀策略。

2.2 Q-learning & SARSA

Q-learning[28] 与 SARSA[29] 是基于 Temporal-Difference 发展的强化学习算法。二者的流程结构基本相同，仅在对于奖励预期的估算上有所不同。处于相同状态时，二者均需使用下一个状态的价值计算奖励预期，SARSA 算法使用了与当前阶段同样的策略来选择下一阶段的动作，而 Q-learning 则选择下一阶段价值最高的动作。从机器学习的角度来理解，二者都是“stochastic”方法：利用下一个状态的动作价值进行计算未必能够使得该阶段动作的价值估计更接近真实值，但一定能够使得估计达到局部最优，故经过一系列调优后，二者均能在实践中达到很好的效果。

强化学习方法基本算法在应用于自动驾驶技术开发时，仍面临着诸如维数爆炸的连续空间与连续状态问题，但这也并非全无解决办法。如果对精度的要求较低，也可以通过将连续的动作空间与状态空间进行离散化，以使用 Q-learning 与 SARSA 算法进行强化学习求解。如将连续的驾驶路径划分为多个离散节点、将连续的速度、转角控制转化为离散的“向左、向右、前进、后退”指令等。事实上，也有许多学者使用 Q-learning 与 SARSA 进行了自动驾驶导航系统 [30][31][32] 的开发。

2.3 Deep Q-Network & DDPG & TD3

自动驾驶是一个连续状态空间场景，例如车辆与车道线的夹角是一个实数。在 Q-learning 算法中，智能体通过离散化状态空间，使用 State-Action 映射表来寻找当前状态的最佳动作。随着自动驾驶的精度要求上升，映射表的复杂度面临维数灾难。一种自然的想法是建立一个映射 $F(State_t) = Action^*$，即建立连续的 State-Action 映射，使用连续函数来接收输入的状态空间。

2.3.1 Deep Q-Network

Deep Q-Network(DQN) 开创性地使用了深度网络来作为状态-动作连续函数。在 DQN 算法中，深度网络的更新目标为某一状态下各个动作的价值，实际上替代了 Q-learning 与

SARSA 中的 Q-table。更新过程与 Q-learning 十分相似，都通过下一状态的奖励与下一状态价值最大的动作价值来更新当前状态的价值，也即

$$Q(S,A) \leftarrow Q(S,A) + \alpha[R + \gamma \max_a Q(S',a) - Q(S,A)]$$

Deep Q-Network 使得高维数据、复杂策略、连续状态场景问题的强化学习方法应用不再成为难题，在 Deep Q-Network 算法提出后，许多学者使用 DRL 思想进行了自动驾驶技术开发 [33][34][13]。在路径规划问题上，也有许多 DQN 算法的应用实例 [35][36]。在 2017 年，更是有学者总结出了使用 DQN 进行自动驾驶技术开发的通用框架 [37]。

2.3.2 Deep Deterministic Policy Gradient

在 DQN 算法的基础上，更多 DRL 算法开始出现。我们不难从 Deep Q-Network 的深度网络更新过程公式中看出，DQN 的深度网络不能处理连续的控制问题（$\max_a Q(S',a)$ 只能处理离散数值）。在自动驾驶问题场景，例如车速、拐弯角度都是连续的数值，传统 DQN 算法并不能直接输出连续的动作值。深度确定性策略梯度算法 [38]（DDPG）通过引入 Actor-Critic 网络 [39] 解决了这一问题。其中，Actor 网络代替了 $\max_a Q(S',a)$ 的职能。Actor 网络接收环境参数，输出取值为实数的连续动作，使得 DDPG 算法能够处理连续控制型问题。与此同时，Critic 网络替代了 Q-learning 中的 Q-table，对各个状态下的动作价值进行估计。值得注意的是，由于 DDPG 也采用了“冻结更新”的方式，故 DDPG 算法中一共有两套 Actor-Critic 网络。

DDPG 能够很好地处理连续状态与连续控制问题，随着自动驾驶技术开发考虑的环境因素不断增多，控制系统精细程度不断上升，DDPG 成为了当下自动驾驶技术开发使用的主流算法。DDPG 可以直接用于自动驾驶技术开发 [40][41]，也可以与模仿学习相结合来缩短探索时间 [42][43]，或是与多智能体算法结合来改进探索采样的效果 [44]。

2.3.3 Twin Delayed Deep Deterministic policy gradient algorithm

双延迟深度确定性策略梯度 [45]（TD3）是在 DQN 与 DDPG 的基础上发展而来的一种改良算法，在 DDPG 的基础上做出了三大优化：double Network、Delayed 与 target policy smoothing regularization。

DQN 存在显著的 Q 值高估问题，这是由更新公式中的 $\max_a Q(S',a)$ 导致的，在每一探索阶段中都取价值最高的动作，在采样时就会使得 Q 值显著过高。一个自然的想法是取价值“较低”的动作进行采样，TD3 将原有的 Actor-Critic 网络中的 Critic 网络变为两个，每次采样时选择 Q 值较小的网络选取的动作，这也就是 double Network 优化。

除此之外，TD3 还提出 Delayed 优化，让 Critic 网络更新的频率变低，让 Critic 网络在较为确定的情况下，由 Actor 再进行梯度更新，避免陷入局部最优点；最后则是 target policy smoothing regularization 优化，在计算 target 时，在 Actor 网络上加上噪声，使得网络的泛化能力更强。

TD3 在交通领域的智能车辆能源管理问题中发挥了重要作用 [46][47]。

2.4 Actor-Critic & A3C & SAC

Actor-Critic 网络不仅在 DDPG 中起到"Magic Function" 的作用，也是现代诸多强化学习算法的通用框架。在 DDPG 中，Actor-Critic 网络分别起到了状态-动作连续映射与动作价值估计的作用，但在其他主流算法中，Actor-Critic 网络往往起到不同的作用。

专注于 Actor-Critic 网络与策略梯度问题的学者 John Schulman 在博士论文 [48] 中提出，策略梯度公式 $g = \mathbb{E}\left[\sum_{t=0}^{\infty} \Psi_t \nabla_\theta \log \pi_\theta\left(a_t \mid s_t\right)\right]$ 中，Ψ_t 取状态-行为值函数 $Q^\pi\left(s_t, a_t\right)$、优势函数 $A^\pi\left(s_t, a_t\right)$、TD 残差 $r_t + V^\pi\left(s_{t+1}\right) - V^\pi\left(s_t\right)$ 中的任一种时，该策略梯度称为广义 AC 框架。在 DDPG 中，AC 网络选择的就是其中的状态-行为值函数。

2.4.1 Asynchronous Advantage Actor–Critic Algorithm

Actor-Critic 衍生出了许多现代强化学习算法，在此基础上直接发展出的算法是 A3C[49]，至今仍是自动驾驶技术开发领域的主流算法之一。Actor-Critic 对行为价值进行预测逼近时，具有传统神经网络的一大通病：过分依赖数据样本独立同分布的假设。而强化学习场景的数据在时间上是具有较强的联系的，这样就造成了神经网络在训练时的不稳定。A3C 在 AC 网络的基本框架上加入了多线程的异步探索，即多个智能体共同探索，并行计算策略梯度。A3C 由多个智能体上传采样数据维护一个"Global Network"，自身各维护一个"Local Network"，定时将 Global Network 的参数覆盖 Local Network 的参数，以更好地进行探索采样，故 A3C 的名称为异步优化的 AC 算法（Asynchronous Advantage Actor–Critic Algorithm）。基于 A3C 进行的自动驾驶技术研究是近年的热门。事实上，该章节开头所提及的端到端的自动驾驶模型就全是基于 A3C 实现的，此处不重复引用。

当然，数据样本独立性差问题的解决方法并不止于此，也有学者在 A2C（A3C 的前身）[50] 的基础上引入了长短期记忆（LSTM），来组织不同时间步下联系较强的数据 [51]。

2.4.2 Soft Actor-Critic

Soft Actor-Critic（SAC）[52][53] 可以看作是熵最大优化的 DDPG（DDPG 也是使用 Actor-Critic 框架进行开发的）。SAC 在公开的 benchmark 中取得了非常好的效果，目前被广泛应用于机器人控制 [54][55] 与自动驾驶领域 [56][57][58] 当中。

与 DDPG 相比，SAC 在学习目标函数中额外加入了动作熵值，即

$$\pi^* = \arg\max_\pi \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[\sum_t \underbrace{R\left(s_t, a_t\right)}_{\text{reward}} + \alpha \underbrace{H\left(\pi\left(\cdot \mid s_t\right)\right)}_{\text{entropy}}]$$

这样就使得策略随机化，即采取每一个行动的概率尽可能相等，鼓励智能体尝试每一个可能有用的动作，避免学习过程过早停止探索，陷入局部最优解。

2.5 Policy Gradient & PPO

策略优化算法（Policy Optimization）有别于包括 DQN 及其衍生算法在内的的 Value-based 方法，是 Model-free 强化学习中的另一大类算法。

策略优化算法将奖励期望视作策略 θ 的函数，通过梯度上升来优化 θ，使得奖励期望达到最大。求取最终的回报函数关于 θ 的梯度，这个就是策略梯度（Policy Gradient），通过优化策略梯度来求解 RL 问题的算法就叫做策略梯度算法。可以看出，策略梯度算法仅考虑动作对奖励期望的影响，而不考虑状态，且仅对动作采取的概率进行调整，不给动作打分，显著区别于 Value-based 类算法。

2.5.1 Proximal Policy Optimization

近端策略优化算法（PPO）是当下最为流行的强化学习算法，具有很广的适用性，在自动驾驶技术开发领域发挥着重要作用 [59][60]。

PPO 算法是一种改良的策略梯度算法。传统的策略梯度算法在参数更新时，对步长十分敏感，但是又难以选择合适的步长，作为 On-Policy 算法在训练过程中新旧策略的的变化差异过大时不利于学习。与此同时，传统的策略梯度算法并没有考虑在连续动作状态空间中的应用问题。PPO 算法对此提出了两大优化，引入 AC 网络解决了连续动作状态空间中的应用问题，引入 Importce Sampling 的方法，将 Policy Gradient 中 On-policy 的训练过程转化为 Off-policy，即从在线学习转化为离线学习。

2.6 强化学习算法的测试数据

我们基于 gym 和 highway-env 中自带环境与自定义环境，使用 Stable-baselines3 测试了几种常见的强化学习算法。

下表反映了四种常见的强化学习算法在 highway-env 的'parking-v0' 环境中训练的成功率以及所需要的时间步。

	成功率	初次成功时间
A2C	98.9%	2500k(推定)
DDPG	99.9%	1300k
PPO	99.1%	2300k(推定)
TD3	99.4%	1700k

需要注意的是，

- 初次成功时间指代的是模型成功率初次达到 98% 时所使用的时间步。
- 由于部分模型的 log 并未给出成功率，因此部分数据由模型平均奖励、轮次平均耗时等数据推定而来。

另外，基于 Stable-baseline3 和 pybullet 的多种三维环境，araffin 先生测试了在不同情况下，不同算法在环境中得到的分数值 [61]。下表中给出了具体数值，其可以在 https://github.com/DLR-RM/stable-baselines3/issues/48 中查看。

需要指出的是，Gaussian 和 gSED 表示过程中使用了非结构化高斯噪声或广义状态依赖探索。

Environments	A2C Gaussian	A2C gSDE	PPO Gaussian	PPO gSDE
HalfCheetah	2003 +/- 54	2032 +/- 122	1976 +/- 479	2826 +/- 45
Ant	2286 +/- 72	2443 +/- 89	2364 +/- 120	2782 +/- 76
Hopper	1627 +/- 158	1561 +/- 220	1567 +/- 339	2512 +/- 21
Walker2D	577 +/- 65	839 +/- 56	1230 +/- 147	2019 +/- 64

Environments	SAC Gaussian	SAC gSDE	TD3 Gaussian	TD3 gSDE
HalfCheetah	2757 +/- 53	2984 +/- 202	2774 +/- 35	2592 +/- 84
Ant	3146 +/- 35	3102 +/- 37	3305 +/- 43	3345 +/- 39
Hopper	2422 +/- 168	2262 +/- 1	2429 +/- 126	2515 +/- 67
Walker2D	2184 +/- 54	2136 +/- 67	2063 +/- 185	1814 +/- 395

参考文献

[1] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *ArXiv*, abs/1706.05587, 2017.

[2] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2014.

[3] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *ArXiv*, abs/1505.04597, 2015.

[4] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:2481–2495, 2015.

[5] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *CoRR*, abs/1511.07122, 2015.

[6] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5168–5177, 2016.

[7] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[8] Nesma M Ashraf, Reham R Mostafa, Rasha H Sakr, and MZ Rashad. Optimizing hyperparameters of deep reinforcement learning for autonomous driving based on whale optimization algorithm. *Plos one*, 16(6):e0252754, 2021.

[9] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. Recent progress in road and lane detection: a survey. *Machine vision and applications*, 25(3):727–745, 2014.

[10] Hongbo Gao, Bo Cheng, Jianqiang Wang, Keqiang Li, Jianhui Zhao, and Deyi Li. Object classification using cnn-based fusion of vision and lidar in autonomous vehicle environment. *IEEE Transactions on Industrial Informatics*, 14(9):4224–4231, 2018.

[11] Il Bae, Jaeyoung Moon, Jaekwang Cha, and Shiho Kim. Integrated lateral and longitudinal control system for autonomous vehicles. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 406–411. IEEE, 2014.

[12] Maximilian Jaritz, Raoul De Charette, Marin Toromanoff, Etienne Perot, and Fawzi Nashashibi. End-to-end race driving with deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2070–2075. IEEE, 2018.

[13] Etienne Perot, Maximilian Jaritz, Marin Toromanoff, and Raoul De Charette. End-to-end driving in a realistic racing game with deep reinforcement learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 3–4, 2017.

[14] Xinlei Pan, Yurong You, Ziyan Wang, and Cewu Lu. Virtual to real reinforcement learning for autonomous driving. 2017.

[15] Victor Talpaert, Ibrahim Sobh, B Ravi Kiran, Patrick Mannion, Senthil Yogamani, Ahmad El-Sallab, and Patrick Perez. Exploring applications of deep reinforcement learning for real-world autonomous driving systems. *arXiv preprint arXiv:1901.01536*, 2019.

[16] Stephane Ross, Narek Melik-Barkhudarov, Kumar Shaurya Shankar, Andreas Wendel, Debadeepta Dey, J. Andrew Bagnell, and Martial Hebert. Learning monocular reactive uav control in cluttered natural environments, 2012.

[17] Dennis Barrios Aranibar and Pablo Javier Alsina. Reinforcement learning-based path planning for autonomous robots .

[18] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[19] Kshitij Judah, Alan Fern, Prasad Tadepalli, and Robby Goetschalckx. Imitation learning with demonstrations and shaping rewards. *Proceedings of the AAAI Conference on Artificial Intelligence*, 28(1), Jun. 2014.

[20] Jeffrey Hawke, Richard Shen, Corina Gurau, Siddharth Sharma, Daniele Reda, Nikolay Nikolov, Przemysław Mazur, Sean Micklethwaite, Nicolas Griffiths, Amar Shah, et al.

Urban driving with conditional imitation learning. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 251–257. IEEE, 2020.

[21] Seyed Mohammad Jafar Jalali, Parham M Kebria, Abbas Khosravi, Khaled Saleh, Darius Nahavandi, and Saeid Nahavandi. Optimal autonomous driving through deep imitation learning and neuroevolution. In *2019 IEEE international conference on systems, man and cybernetics (SMC)*, pages 1215–1220. IEEE, 2019.

[22] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *Proceedings of the IEEE international conference on computer vision*, pages 2722–2730, 2015.

[23] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[24] Li Cui, Chunyan Rong, Jingyi Huang, Andre Rosendo, and Laurent Kneip. Monte-carlo localization in underground parking lots using parking slot numbers. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, page 2267–2274. IEEE Press, 2021.

[25] Gero Friesecke and Daniela Vogler. Breaking the curse of dimension in multi-marginal kantorovich optimal transport on finite state spaces. *SIAM Journal on Mathematical Analysis*, 50(4):3996–4019, 2018.

[26] T.L Lai and Herbert Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[27] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

[28] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

[29] G. Rummery and Mahesan Niranjan. On-line q-learning using connectionist systems. *Technical Report CUED/F-INFENG/TR 166*, 11 1994.

[30] Shanqing Yu, Jing Zhou, Bing Li, Shingo Mabu, and Kotaro Hirasawa. Q value-based dynamic programming with sarsa learning for real time route guidance in large scale road networks. In *The 2012 international joint conference on neural networks (IJCNN)*, pages 1–7. IEEE, 2012.

[31] Feng Wen and Xingqiao Wang. Sarsa learning based route guidance system with global and local parameter strategy. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 98(12):2686–2693, 2015.

[32] Naoto Mukai, Toyohide Watanabe, and Jun Feng. Route optimization using q-learning for on-demand bus systems. In *Knowledge-Based Intelligent Information and Engineering Systems: 12th International Conference, KES 2008, Zagreb, Croatia, September 3-5, 2008, Proceedings, Part II 12*, pages 567–574. Springer, 2008.

[33] Xi Xiong, Jianqiang Wang, Fang Zhang, and Keqiang Li. Combining deep reinforcement learning and safety based control for autonomous driving. *arXiv preprint arXiv:1612.00147*, 2016.

[34] Matt Vitelli and Aran Nayebi. Carma: A deep reinforcement learning approach to autonomous driving. *Tech. rep. Stanford University, Tech. Rep.*, 2016.

[35] Daniel Paul Romero-Marti, Jose Ignacio Nunez-Varela, Carlos Soubervielle-Montalvo, and Alfredo Orozco-De-La-Paz. Navigation and path planning using reinforcement learning for a roomba robot. In *Robotica, XVIII Congreso Mexicano De*, 2016.

[36] Jing Xin, Huan Zhao, Ding Liu, and Minqi Li. Application of deep reinforcement learning in mobile robot path planning. In *2017 Chinese Automation Congress (CAC)*, 2018.

[37] Ahmad EL Sallab, Mohammed Abdou, Etienne Perot, and Senthil Yogamani. Deep reinforcement learning framework for autonomous driving. *arXiv preprint arXiv:1704.02532*, 2017.

[38] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[39] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12, 1999.

[40] Che-Cheng Chang, Jichiang Tsai, Jun-Han Lin, and Yee-Ming Ooi. Autonomous driving control using the ddpg and rdpg algorithms. *Applied Sciences*, 11(22), 2021.

[41] Sen Wang, Daoyuan Jia, and Xinshuo Weng. Deep reinforcement learning for autonomous driving, 2018.

[42] Dianzhao Li and Ostap Okhrin. Modified ddpg car-following model with a real-world human driving experience with carla simulator. *Transportation Research Part C: Emerging Technologies*, 147:103987, 2023.

[43] Qijie Zou, Kang Xiong, and Yingli Hou. An end-to-end learning of driving strategies based on ddpg and imitation learning. In *2020 Chinese Control And Decision Conference (CCDC)*, pages 3190–3195, 2020.

[44] Gaoyang Hua, Zhiqiu Huang, Jinyong Wang, Jian Xie, and Guohua Shen. Exploration strategy improved ddpg for lane keeping tasks in autonomous driving. *Journal of Physics: Conference Series*, 2347(1):012020, sep 2022.

[45] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[46] Jianhao Zhou, Siwu Xue, Yuan Xue, Yuhui Liao, Jun Liu, and Wanzhong Zhao. A novel energy management strategy of hybrid electric vehicle via an improved td3 deep reinforcement learning. *Energy*, 224:120118, 2021.

[47] Ruchen Huang, Hongwen He, Xuyang Zhao, Yunlong Wang, and Menglin Li. Battery health-aware and naturalistic data-driven energy management for hybrid electric bus based on td3 deep reinforcement learning algorithm. *Applied Energy*, 321:119353, 2022.

[48] Pieter Abbeel and John Schulman. Deep reinforcement learning through policy optimization. *Tutorial at Neural Information Processing Systems*, 2016.

[49] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

[50] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. *CoRR*, abs/1602.01783, 2016.

[51] Sampo Kuutti, Richard Bowden, Harita Joshi, Robert De Temple, and Saber Fallah. End-to-end reinforcement learning for autonomous longitudinal control using advantage actor critic with temporal context. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2456–2462. IEEE, 2019.

[52] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018.

[53] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[54] Junior Costa de Jesus, Victor Augusto Kich, Alisson Henrique Kolling, Ricardo Bedin Grando, Marco Antonio de Souza Leite Cuadros, and Daniel Fernando Tello Gamarra. Soft actor-critic for navigation of mobile robots. *Journal of Intelligent & Robotic Systems*, 102(2):31, 2021.

[55] Ching-Chang Wong, Shao-Yu Chien, Hsuan-Ming Feng, and Hisasuki Aoyama. Motion planning for dual-arm robot based on soft actor-critic. *IEEE Access*, 9:26871–26885, 2021.

[56] Xiaolin Tang, Bing Huang, Teng Liu, and Xianke Lin. Highway decision-making and motion planning for autonomous driving via soft actor-critic. *IEEE Transactions on Vehicular Technology*, 71(5):4706–4717, 2022.

[57] Jingliang Duan, Yangang Ren, Fawang Zhang, Yang Guan, Dongjie Yu, Shengbo Eben Li, Bo Cheng, and Lin Zhao. Encoding distributional soft actor-critic for autonomous driving in multi-lane scenarios. *arXiv preprint arXiv:2109.05540*, 2021.

[58] Maryam Savari and Yoonsuck Choe. Online virtual training in soft actor-critic for autonomous driving. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2021.

[59] Guanlin Wu, Wenqi Fang, Ji Wang, Pin Ge, Jiang Cao, Yang Ping, and Peng Gou. Dyna-ppo reinforcement learning with gaussian process for the continuous action decision-making in autonomous driving. *Applied Intelligence*, pages 1–15, 2022.

[60] John Holler, Risto Vuorio, Zhiwei Qin, Xiaocheng Tang, Yan Jiao, Tiancheng Jin, Satinder Singh, Chenxi Wang, and Jieping Ye. Deep reinforcement learning for multi-driver vehicle dispatching and repositioning problem. In *2019 IEEE International Conference on Data Mining (ICDM)*, pages 1090–1095, 2019.

[61] araffin. Performance check (continuous actions). https://github.com/DLR-RM/stable-baselines3/issues/48.