上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

学士学位论文

THESIS OF BACHELOR

论文题目：时空感知的深度学习动态点云语义分割研究

学生姓名：曹瀚文

学生学号：516021910367

专　　业：信息安全

指导教师：刘功申教授

学院(系)：电子信息与电气工程学院

上海交通大学

本科生毕业设计（论文）任务书

课题名称：时空感知的动态点云语义分割研究

执行时间：2019 年 12 月 至 2020 年 6 月

教师姓名：刘功申　　职称：教授

学生姓名：曹瀚文　　学号：516021910367

专业名称：信息安全

学院(系)：电子信息与电气工程学院

毕业设计（论文）基本内容和要求：

语义分割是计算机视觉几个最基本的任务之一，对于机器感知和理解周围环境有着重要的作用。点云作为一种新兴的数据格式，在自动驾驶及各类机器人相关的任务中有着重要应用。目前，神经网络对于静态点云处理取得了很大的进步，出现了类似 3D 卷积、PointNet、GCNN 等处理方法。但是，对于动态点云的处理，几乎没有相关的工作，而生物上的一些实验现象表明，人的感知系统主要依赖时间信息去理解动态环境。本课题计划对动态点云的语义分割进行开创性的研究，旨在设计一个通用的神经网络模块对时间信息进行有效的学习并适配各类静态点云神经网络。此研究对于神经网络时间与空间信息学习理解有着重大意义，是机器感知的基础之一。该课题是计算机视觉领域的前沿，在学术圈几乎没有可以直接参考的前人工作。但随着相关数据集的公开，课题具备了开展基本的条件。本课题将对已有的常规视频的语义分割网络模型进行全面的分析并进行借鉴，并设计新的网络结构对时间与空间信息进行表征和学习，然后在公开的 SemanticKITTI（目前唯一）数据集上进行训练和验证，与目前最好的静态方法进行比较。

课题主要工作包括：（1）对于常规视频语义分割模型的分析与借鉴以及对于已有的各类静态点云分割模型进行特点分析；（2）对网络结构进行设计并在数据集上进行训练验证以及不断改进；（3）完成一篇高质量的学位论文。

参考文献：

（1） 韩利丽, 孟朝晖. 基于深度学习的视频语义分割综述[J]. 计算机系统应用, 2019, 28(12): 1-8.

（2） Benuwa B B, Zhan Y, Liu J Q, et al. Group sparse based locality–sensitive dictionary learning for video semantic analysis[J]. Multimedia Tools and Applications, 2019, 78(6): 6721-6744.

（3） 熊汉江, 郑先伟, 丁友丽, 等. 基于 2D-3D 语义传递的室内三维点云模型语义分割[J]. 武汉大学学报· 信息科学版, 2018, 43(12): 2303-2309.

（4） Kataoka H, Suzuki T, Oikawa S, et al. Drive video analysis for the detection of traffic near-miss incidents[C]//2018 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2018: 1-8.

（5） Bi C, Yuan Y, Zhang J W, et al. Dynamic mode decomposition based video shot detection[J]. IEEE Access, 2018, 6: 21397-21407.

（6） Amosov O S, Ivanov Y S, Zhiganov S V. Semantic Video Segmentation with Using Ensemble of Particular Classifiers and a Deep Neural Network for Systems of Detecting Abnormal Situations[J]. IT in Industry, 2018, 6: 14-19.

毕业设计（论文）进度安排：

序号	毕业设计（论文）各阶段内容	时间安排	备 注
1	选题及任务书下达	2019.12-2019-12	
2	静态点云分割模型进行特点分析	2020.01-2020.02	
3	分割模型研究	2020.03-2020.03	
4	实验及分析	2020.04-2020.05	
5	论文撰写及答辩	2020.06-2020.06	

课题信息：

课题性质 ：设计☐　　论文√

课题来源*：国家级√　　省部级☐　　校级☐　　横向☐　　预研☐

项目编号　61772337

其他

指导教师签名：　刘功申

2019 年 12 月 19 日

学院（系）意见：

格式规范，内容合理，同意通过。

院长（系主任）签名：　刘功申

2019 年 12 月 19 日

学生签名：　曹瀚文

2019 年 12 月 19 日

上海交通大学
毕业设计（论文）学术诚信声明

本人郑重声明：所呈交的毕业设计（论文），是本人在导师的指导下，独立进行研究工作所取得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的作品成果。对本文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者签名：曹瀚文

日期：　2016 年 6 月 20 日

上海交通大学
毕业设计（论文）版权使用授权书

本毕业设计（论文）作者同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权上海交通大学可以将本毕业设计（论文）的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本毕业设计（论文）。

保密□，在___年解密后适用本授权书。

本论文属于

不保密√。

（请在以上方框内打“√”）

作者签名：曹瀚文　　指导教师签名：刘功申

日期：2020 年 6 月 20 日　　日期：2020 年 6 月 20 日

时空感知的深度学习动态点云语义分割研究

摘 要

点云的最新工作表明，利用跨帧信息，多帧时空联合处理的性能优于单帧版本。由于我们生活的世界是四维的，即三维空间 + 时间，动态点云才是许多实际应用的真实数据（如自动驾驶、机器人操作等）。因此，将静态点云的语义分割扩展到动态点云至关重要。相较于静态的点云，动态点云中不仅包含了空间信息，同时还包含了时间信息。因此，有效地处理动态点云需要同时对这两方面信息进行学习。为此，本文首先进行了神经网络时空感知方面的研究，提出了一种新的时空感知策略，然后在此基础上根据动态点云的特点将其扩展到 3D 点云中。

首先，本文从人脑处理时间空间信息的方式获得启发，提出了半耦合的网络结构设计和学习策略。对于网络结构设计，使用既相对独立又相互交织的双分支设计来实现半耦合；同时，我们还提出了新的梯度下降学习策略来解决计算量和两个分支相互干扰的问题，在训练层面实现了半耦合。然后，本文在时空感知研究的基础上，针对动态点云的特点，提出了一个名为 ASAP（Attention and Structure Aware Point cloud）的灵活模块进一步改进了动态点云的特征学习，该模块同时考虑了跨帧的时间和空间信息，这是成功进行动态点云分割的两个重要因素。ASAP 模块包含一个新的注意力机制的时间嵌入层，以递归的方式在帧与帧之间融合信息丰富的局部结构特征；本文还提出了一种有效的跨帧关联策略，该策略可以充分利用更多的局部结构特征进行时间嵌入，同时增强时间一致性，并降低计算复杂度。

最后，我们展示了本文所提出的 ASAP 模块在点云序列分割任务中对不同基干网络的泛化能力。我们的 ASAP-Net（基干网络 +ASAP 模块）在 Synthia 和 SemanticKITTI 数据集上均优于先前的方法（对不同基干网络有 + **3.4** 至 + **15.2** mIoU 提升）。此外，为了进一步验证本文提出的时空感知的网络设计策略和学习策略，我们在动作识别数据集 UCF-101，HMDB-51 和 Kinetics-400 以及驾驶行为数据集 comma.ai 和 LiVi-Set 上进行了验证，实验结果充分说明了本文提出的模型和策略的有效性。

关键词：时空感知, 动态点云, 语义分割

DEEP LEARNING BASED DYNAMIC POINT CLOUD SEMANTIC SEGMENTATION WITH SPATIAL-TEMPORAL UNDERSTANDING

ABSTRACT

Recent works of point cloud show that mulit-frame spatio-temporal modeling outperforms single-frame versions by utilizing cross-frame information. Since we live in a 4D (3D space + time) world, dynamic point cloud is the exact input for many real-world applications. Therefore, it's of critical importance to extend semantic segmentation from static point cloud to dynamic point cloud. Compared with static point cloud, dynamic point cloud contains not only spatial information but also temporal information. Therefore, we should learn the two aspects of information properly to process dynamic point cloud effectively. To this end, we first do some research on spatial-temporal understanding with neural network and propose a new spatial-temporal learning strategy. We then extend it to 3D point cloud considering about the characteristics of dynamic point cloud.

To start with, inspired by the mechanism of how human brain processes spatial and temporal information, we propose a semi-coupled network architecture design and training strategy. For network architecture design, we adopt a relatively independent but interactive two-stream design to achieve semi-coupling; we also propose a new gradient descent learning strategy to solve the problems of expensive computation and memory occupation, thus achieving semi-coupling in the training level. Then, based on the research about spatial-temporal understanding and considering about the characteristics of dynamic point cloud, we further improve spatial-temporal point cloud feature learning with a flexible module called ASAP considering both attention and structure information across frames, which we find as two important factors for successful segmentation in dynamic point clouds. Firstly, our ASAP module contains a novel attentive temporal embedding layer to fuse the relatively informative local features across frames in a recurrent fashion. Secondly, an efficient spatial-temporal correlation method is proposed to exploit more local structure for embedding, meanwhile enforcing temporal consistency and reducing computation complexity.

Finally, we show the generalization ability of the proposed ASAP module with different backbone networks for point cloud sequence segmentation. Our ASAP-Net (backbone plus ASAP module) outperforms baselines and previous methods on both Synthia and SemanticKITTI datasets (+**3.4** to +**15.2** mIoU points with different backbones). To further validate the effectiveness of our spatial-temporal understanding network architecture design and training strategy, we conduct experiments on action recognition datasets UCF-101[1], HMDB-51[2] 和 Kinetics-400[3] and driver behavior prediction datasets comma.ai[4] 和 LiVi-Set[5]. The experiments show the effectiveness of our model and strategy.

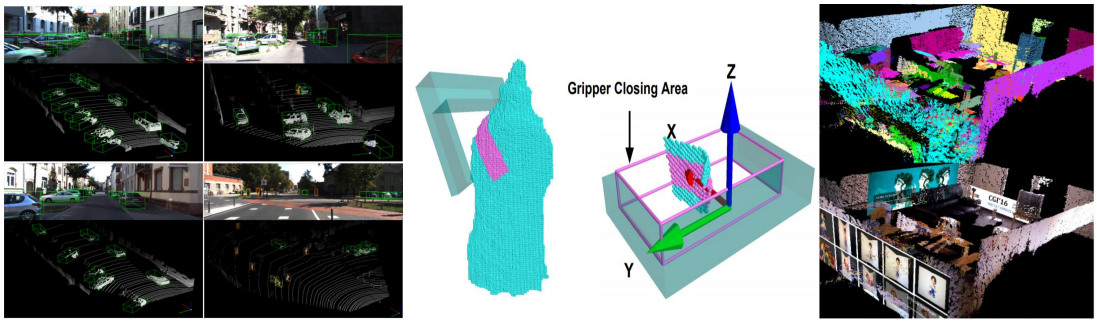
Key words: spatial-temporal understanding, dynamic point cloud, semantic segmentation

目 录

第一章　绪论

1.1　研究背景

随着深度学习的发展，计算机视觉近几年取得了很大的进步，深度神经网络大幅超越传统方法的效果。在 2D 视觉（常规的图像视频等）取得了重大成就的同时，人们越来越关注 3D 视觉。其中，点云数据是近几年新兴的一种 3D 视觉数据格式。点云即为三维空间中离散的点的集合，其具有的三维空间信息是传统 RGB 图像所不具备的，因此在多种实际任务中有着重要的应用，如自动驾驶[6]、机器人[7]、增强现实（AR）[7] 等（见图 1–1）。随着相关采集设备（如激光雷达（LiDAR）、深度相机等）的发展，出现了大量点云数据，为相关的研究提供了基本条件。与传统的 RGB 图像对应，点云的处理也包括多种任务，如物体分类、语义分割、目标检测等。由于语义分割在场景理解中的重要性以及实际生活中环境的动态性，本文主要研究动态点云的语义分割。



(a)　(b)　(c)

图 1–1 点云的不同应用示例。(a) 自动驾驶：道路场景的目标检测；(b) 机器人：物体的抓取；(c) 增强现实。

静态点云中的语义分割起步相对较早，已经有了很多优秀的工作。这些工作大致可以分为两类：直接处理点云的方法和基于表征空间转换的方法（见章节 2.1）。两类方法的代表工作分别有 PointNet++[8]、SqueezeSegV2[9] 等。但点云序列中的语义分割的研究相对欠缺，目前已知的有 MeteorNet[10] 和 4D MinkNet[11]（见章节 2.2）。

我们实际生活的环境是四维的（三维的空间 + 时间），为了有效实现场景理解，神经网络需要具备同时学习空间特征和时间特征的能力。生物上的一些实验表明，时间信息在人的视觉系统理解动态场景的过程中起着主要的作用[12]，但目前人们对时间信息能对动态场景理解起到多大作用所知甚少[13]，因此研究时间信息对动态点云语义分割的作用对实现真正的机器智能具有至关重要的作用。

在此之前，2D 的语义分割已经经历了静态图片到动态视频的过程。随着点云序列数据集 semanticKITTI[14] 的公开，3D 点云中的语义分割具备了基础条件。研究表明，在人脑中，时间信息和空间结构信息是分成两个支路分别输入到海马体中的[15]。之前的工作[10, 11] 没有有效地探究人脑的这种时空概念的处理机制，为此，本课题的研究目的在于探索如何实现有效的点云序列中时间和空间感知以提高动态场景下的语义分割（场景理解）效果，并希

望为之后相关的研究提供借鉴。

1.2 国内外研究现状

由于点云数据本身具有稀疏性、分布不规则性、无序性[16]，这给点云数据的处理带来了较大的困难，无法直接使用计算机视觉中广泛采用的卷积神经网络（CNN）进行处理。对于点云的处理方法，主要有两个方向：

- **基于表征空间转换的方法** 上文提到点云具有分布不规则性和无序性，自然想到将其先转换为规则的表示形式，从而可以利用卷积神经网络（CNN）等有效的特征提取工具进行处理。如 SEGCloud[17] 首先将点云转换为 3d 体素（规则的立方体方格），然后在规则的体素上使用 3D 卷积进行处理，同时还引入三线性插值来将 3D-FCNN[18] 生成的相对粗略的体素语义预测结果映射到每个点，然后使用全连接的条件随机场（FCCRF）来增强语义类别预测结果的空间一致性；针对激光雷达（LiDAR）的工作特性，SqueezeSeg[19] 首先根据方位角（$azimuth$）和天顶角（$zenith$），将点云投影到球面上，然后应用 SqueezeNet[20] 提出的卷积模块构建常见的编码器-解码器（encoder-decoder）结构的卷积网络进行语义分割，SqueezeSegV2[9] 进一步提出了环境聚合模块（Context Aggregation Module）来解决点云数据中部分点缺失的问题，并通过一种非监督的域自适应（domain adaptation）来解决域迁移的问题，从而可以使用大量 GTA-V 游戏中的数据进行训练。
- **直接点云处理的方法** 这类方法直接对点云数据进行处理，从而能够避免格式转换过程可能带来的不必要的信息损失。PointNet[16] 率先提出使用多层感知器（MLP）和池化层等对称函数来对无序的点云进行处理。但直接使用池化操作不能捕获点云中局部的结构特征，为此，PointNet++[8] 按照一定的半径范围对点进行分组，并使用最远点采样的方法逐层采取中心点以逐步扩大学习的局部特征范围，最后使用三线性插值的方法来恢复原始点云的点的数目。类似的，Wang 等人[21] 提出了动态图卷积神经网络对点云进行处理，通过使用 K-最近邻（KNN）来对点进行分组捕获局部结构特征。

目前已知的其他两种处理动态点云（点云序列）的方法，4D MinkNet[11] 和 MeteorNet[10] 分别属于上述两种处理思路。其中，4D MinkNet 首先将点云序列转换为四维的体素（三维空间 + 时间维度），并提出了广义的多维稀疏卷积（Sparse Convolution）库来解决点云的稀疏性所带来的计算复杂度和内存消耗问题，同时提出了一种混合了十字形卷积核和立方体卷积核的新的卷积核形状来提取时间和空间特征。MeteorNet[10] 则在 PointNet++[8] 的基础上，提出了聚合多帧点云特征的 Meteor 模块，并提出了根据时间增加搜索半径和点跟踪两种邻域点分组的方法来学习局部的时空特征。

但是，上述两种方法并没有对章节 1.1中提到的类似人脑的时空感知机制进行探索。在 2D 视频的处理中，有些方法对这种机制进行了探索。如在双支路卷积神经网络中，一个支路的输入是静态的图片，即空间信息，而另一个支路的输入是相应的光流，即时间信息；在 SlowFast 网络中，同样存在两路信息，空间支路的输入视频具有较大的特征长度和较低的帧率而时间支路正好相反，以此来分别提取两类信息。本文进一步对时空感知机制进行探索，并将其扩展到 3D 空间，具体而言为动态点云的语义分割。

1.3 研究内容与研究意义

为了有效进行动态点云的语义分割，神经网络需要充分利用点云序列中的时间和空间信息。因此，本文首先对神经网络的时空感知进行研究，我们从人脑海马体时间信息和空间信息分两路输入[15] 同时又是层层交织的[22] 得到启发，提出了半耦合的网络设计和学习策略。之后，我们针对动态点云的特性，将这种策略应用到 3D 动态点云的处理中，并引入注意力机制，提出一种名为 ASAP（Attention and Structure Aware Point cloud）的通用模块。该模块可以轻松地嵌入到各类点云处理的基干网络中，并显著提高其语义分割的结果。

对于时空感知的研究，自然的想法是设计两路神经网络分别处理时间和空间信息以实现一定的独立性，同时加深网络层数在多层网络中将这两部分信息进行融合。但这将带来两个问题：一是随着网络层数的增加，在处理序列数据时，反传路径数量大大增加，即增加了计算复杂度和内存的占用；二是随着网络层数的增加，这两部分信息在学习的过程中可能相互干扰。为了解决这两个问题，本文提出了一种名为 STSGD（Spatial-Temporal Switch Gradient Descent）的梯度下降学习策略，该策略会根据概率自动地关闭某些反传路径来降低计算复杂度以及减少两路神经网络之间地干扰，本文进一步提出了高级 STSGD（ASTSGD，Advanced Spatial-Temporal Switch Gradient Descent）来自动地调整概率，此外，本文还提出使用时间和空间学习的子任务来进一步使两路神经网络能够专注于自身的功能以实现更好的整体效果。

虽然本文提出时空感知的网络设计和学习策略具有普适性，但对于动态点云的语义分割，仍需要对点云序列的特点进行相应的调整和设计。为此，我们使用其他静态点云处理的方法作为我们网络的空间支路，并根据点云序列的特点提出了名为 ASAP 的模块来作为时间分路。设计该模块主要面临的挑战有：

（a）不同帧特征的融合：为了捕获时间信息，神经网络需要对来自不同帧的特征进行融合，但来自不同帧的特征可能对最终结果的重要程度不同，并且它们都可能包含不希望有的噪音或错误。理想情况下，神经网络应该有自动识别不同帧的重要性或置信度的能力，从而获得更好的融合效果。

（b）跨帧点的关联：为了融合来自不同帧的特征，我们需要关联不同帧的点。但是，动态点的分布会不时变化，并且它们是无序的，同时实际应用如自动驾驶等往往对处理速度要求较高，这些原因使得跨帧点的关联相对困难。

在我们的 ASAP 模块中，我们提出了一种新的注意力机制的时间嵌入层和一个时空关联策略来分别解决上述两个挑战。在融合不同帧的特征时，根据注意力分数进行加权求和来进行融合；在进行点的关联时，使用恒定的中心点来保持时间一致性并避免不必要的计算。

对于动态点云的语义分割，我们在目前开放的最大规模实际点云语义分割数据集 SemanticKITTI[14] 以及大规模模拟器生成的数据集 Synthia[23] 对我们的模型进行了充分的测试和拆解分析，并超过了目前已知的方法 MeteorNet[10] 和 4D MinkNet[11]。对于我们提出的用于时空感知的网络结构设计和学习策略，我们选取了两个需要对时空信息有很好理解的任务：动作识别和驾驶行为预测。对于动作识别，我们在三个常见数据集 UCF-101[1]、HMDB-51[2] 和 Kinetics-400[3] 上进行验证；对于驾驶行为预测，我们在 comma.ai[4] 和 LiVi-Set[5] 数据集上进行实验。实验结果充分证明了我们模型的有效性。

综上所述，本文的主要贡献可以归纳为：

- 我们对神经网络的时空感知进行了研究，提出了一种半耦合的网络设计策略，通过两个既相对独立又相互交织的支路分别处理时间和空间信息。

- 为了解决两个支路网络层数加深所带来的计算复杂度提升、内存占用以及训练过程中相互干扰的问题，我们提出了名为 STSGD 和高级 STSGD 的训练策略以及引入子任务以获得更好的时空学习效果。
- 为了能有效融合不同帧的点云特征以捕获时间信息，我们提出了一种新的注意力机制的时间嵌入层，并通过自动计算注意力来有效地融合跨帧的空间局部结构特征。
- 为了解决跨帧点云的关联性问题，我们提出一种时空关联策略，该策略能够充分利用结构信息，增强时间一致性并减少计算量。
- 我们提出一种称为 ASAP 模块的新的网络结构，并在点云序列中的语义分割进行了充分的实验，该框架可以灵活地嵌入到之前的静态点云处理网络中，并可以大幅度提高其语义分割的效果。同时对时空感知策略进行了充分的实验以验证其有效性。

1.4 论文章节安排

本文共分为五章，各个章节的内容安排如下：

- **第一章，绪论** 阐述了时空感知的动态点云语义分割的研究背景，对国内外的研究现状进行了总结和分析，并介绍了本文主要的研究内容和研究意义。
- **第二章，深度学习点云语义分割** 介绍了深度学习点云语义分割的发展以及各类别的点云语义分割网络，给读者对目前该领域的发展有初步的认识，并详细讲解了本文需要对照的两篇文章 4D MinkNet[11] 和 MeteorNet[10]，为介绍本文的方法提供了基本的知识背景。
- **第三章，时空感知的动态点云语义分割** 深入讲解了本文为神经网络时空感知任务所提出的半耦合的网络结构设计和学习策略，然后在此基础上，针对动态点云的特点，介绍了本文提出的 ASAP 模块以及整体网络的框架。
- **第四章，实验与分析** 在目前最大的实际点云数据集 SemanticKITTI[14] 和大规模模拟器生成的数据集 Synthia[23] 上对我们的动态点云语义分割模型进行了验证和分析，同时在动作识别数据集 UCF-101[1]、HMDB-51[2] 和 Kinetics-400[3] 以及驾驶行为数据集 comma.ai[4] 和 LiVi-Set[5] 上对我们提出的时空感知的网络结构设计和学习策略进行进一步的验证。
- **第五章，总结与展望** 对全文进行总结，同时思考不足之处，提出未来进一步的研究方向。

第二章　深度学习点云语义分割

给定一个点云，语义分割的目标是将每个点分到一个语义类别。随着深度学习的发展，传统的语义分割模型逐步被深度神经网络所取代。本章将介绍深度学习点云语义分割的方法。

2.1　静态点云的语义分割

静态点云的分割可以大致分为两类，一类方法首先对点云进行预处理，使用某种表征空间对点云进行表示，然后再进行后续处理；另一类方法则直接对点云进行处理。

2.1.1　基于表征空间转化的方法

该类方法主要包括以下几种：

2.1.1.1　多视图表示（Multi-view Representation）

Felix 等人[24] 首次从多个虚拟摄像机视角将 3D 点云投影到 2D 平面，然后使用多分支的全卷积网络（FCN）对各个视角的投影图像进行语义分割，最后通过融合从各个视角重建的点云语义分数来获得每个点的语义标签。同样，Boulch 等人[25] 首先使用不同的相机位置对点云生成了多个 RGB 和深度快照，然后他们在这些快照上使用 2D 的语义分割神经网络进行逐像素的语义标记，最后他们使用残差校正[26] 的方法进一步融合不同 RGB 和深度图像的语义分数。在点云是从局部欧几里得表面采样得到的假设下，Tatarchenko 等人[27] 引入了切点卷积来进行稠密点云的分割。此方法首先将每个点周围的局部表面几何结构投影到虚拟的正切的平面，然后直接对这些表面几何结构进行处理，这种方法显示出很大的可扩展性并能够处理数以百万计的大规模点云点。总体而言，基于多视角投影的方法对视角选择和遮挡很敏感。此外，投影步骤不可避免地会导致信息丢失，这些方法没有充分利用基本的几何和结构信息。

2.1.1.2　球面投影表示（Spherical Representation）

为了实现快速而准确的 3D 点云分割，Wu 等人[19] 在 SqueezeNet[20] 和条件随机场（CRF）的基础上提出了端对端的神经网络 SqueezeSeg。为了进一步提高分割的精度，他们又提出了 SqueezeSegV2[9]，通过一种非监督的域自适应流程来解决域迁移的问题。Milioto 等人[28] 提出了 RangeNet++ 来进行 LiDAR（激光雷达）采集的点云的实时语义分割。他们首先将 2D 距离图像的语义标签传递到 3D 点云，然后使用基于 KNN 的高效 GPU 支持的后处理步骤进一步缓解离散错误和输出模糊的问题。相比较于单视图投影，球形投影能够保留更多的信息，适用于 LiDAR 采集的点云。但是，这种中间表示不可避免地会带来一些问题，例如离散化错误和遮挡问题。

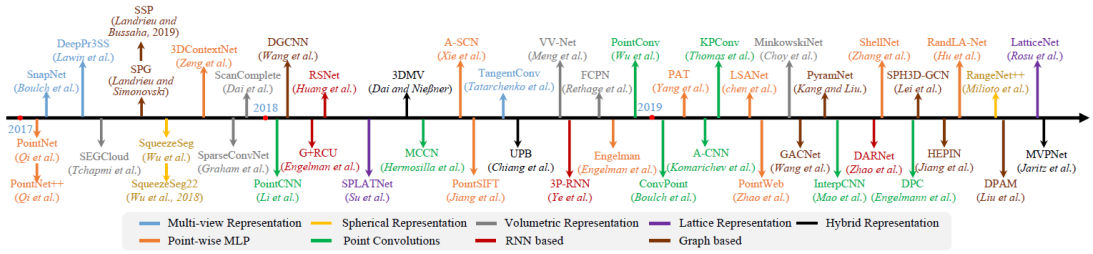


图 2–1 相关的基于深度学习的点云语义分割方法总览。

2.1.1.3 体素化表示（Volumetric Representation）

Huang 等人[29] 首先将点云划分为一组三维的体素。然后他们将这些中间数据输入到全 3D 卷积神经网络进行体素分割。最后，每个体素内的所有点被分配了与体素相同的语义类别。该方法的效果严重受体素划分的精细程度和人为边界的限制。此外，Tchapmi 等人[17] 提出了 SEGCloud 以实现细粒度和全局一致的语义分割。这个方法引入三线性插值来将 3D-FCNN[18] 生成的相对粗略体素语义预测映射到单个点，然后使用全连接的条件随机场（FCCRF）来增强这些推断的逐点语义类别的空间一致性。Meng 等人[30] 引入了基于内核的插值变分自动编码器（VAE）架构对每个体素内的局部几何结构进行编码。不同于二进制占用的表示形式，他们对每个体素使用径向基函数（RBF）进行处理以获得连续的表示并捕获每个体素中点的分布情况。同时使用自动编码器（VAE）进一步用将每个体素内的点分布映射到紧凑的隐空间。最后，他们使用这两组数据和等效的卷积神经网络（CNN）来实现鲁棒的特征学习。

基于体积网络可以方便地对不同的空间大小的点云进行神经网络的训练和测试，这是它的显著优势之一。全卷积点网络（FCPN）[31] 首先以分级的方式从 3D 点云中提取不同层级的几何关系，然后它使用 3D 卷积和加权平均池化来提取特征并处理远距离依赖。这种方法可以处理大规模的点云并在测试过程中对于点云规模具有良好的可伸缩性。Angela 等人[32] 提出了 ScanComplete 来实现 3D 扫描补全和对每个体素的语义标记。此方法利用了卷积神经网络的可伸缩性来适应训练和测试期间不同的输入数据大小。他们还使用了一种粗到精的策略以分层级的方式提高预测结果的分辨率。

点云数据天生具有稀疏性，因为点数非零体素的数量只占一小部分。因此，应用密集卷积神经网络来处理具有空间稀疏性的点云数据是极其低效的。为了解决这个问题，Graham 等人[33] 提出了一种子流形稀疏卷积网络。这种方法通过限制卷积神经网络的输出仅与点数不为零的体素有关大大节省了内存和计算成本。与此同时，它的稀疏卷积也可以控制提取的特征的稀疏性。该子流形稀疏卷积适用于高效处理高维和空间稀疏的数据。此外，Choy 等人[11] 提出了名为 MinkowskiNet4D 时空卷积神经网络，用于 3D 点云序列的感知。他们提出了一种广义的稀疏卷积来有效处理高维数据。最后，他们进一步应用了三边平稳条件随机场来增强语义分割的一致性。

总体而言，体积表示法可以保留 3D 点云原始的欧几里得空间结构。它允许直接应用标准的 3D 卷积。这些有利因素导致这类方法的性能不断地得到提高。但是体素化步骤不可避免地引入了人工离散化和信息损失。此外，高分辨率会导致高内存和计算成本成几何倍上升而低分辨率带来了细节信息地损失。因此，选择合适的网格分辨率在实践中并非易事。

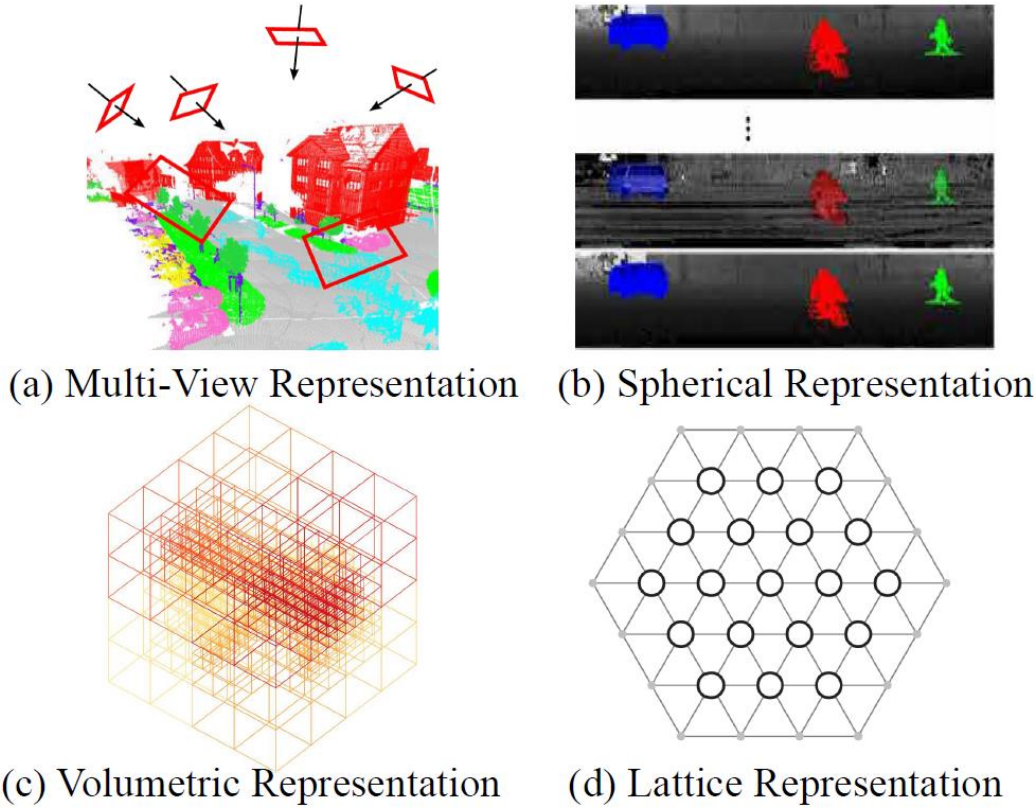


图 2–2 基于空间转换的方法的中间表示简单示意图。

2.1.1.4 多面体晶格排列表示（Permutohedral Lattice Representation）

Su 等人[34] 在双边卷积层（BCL）的基础上提出了稀疏晶格网络（SPLATNet）。这个方法首先将原始点云插值到规则排列的四面体稀疏晶格，然后在生成的稀疏晶格的被占部分使用 BCL 进行卷积。最后，卷积层输出通过插值返回到原始点云。这种方法还能灵活地联合处理多视图图像和点云数据。此外，Rosu 等人[35] 提出了 LatticeNet 来实现大规模点云的高效处理。他们提出了一个名为 DeformsSlice 的与数据相关的插值模块来将晶格中的特征映射到原来的点云。

2.1.1.5 混合表示（Hybrid Representation）

为了进一步利用所有可用的信息，学术界提出了几种方法从 3D 点云数据中学习多类型的特征。Angela 和 Matthias[36] 提出了一种联合的 3D 多视图网络来结合 RGB 特征和几何特征。他们使用一个 3D 卷积神经网络（CNN）分支和几个 2D 卷积神经网络（CNN）分支来进行特征提取，同时提出了一种可求导的反传层来融合学习的 2D 和 3D 几何特征。Hung 等人[37] 进一步提出了一个统一的网络框架来同时学习 2D 纹理外观、3D 结构和整体点云的特征。这种方法直接应用于稀疏采样的点集，没有任何体素化的过程。Jaritz 等人[38] 提出了 MultiviewPointNet（MVPNet）从 2D 多视图图像和点云空间中几何特征综合学习外观特征。

2.1.2 直接点云处理的方法

基于点的网络直接在不规则点云上工作。但是，点云是无序且无结构的，使其无法直接应用标准的卷积神经网络（CNN）进行处理。为此，Qi 等人提出了开拓性的工作 PointNet[16]，他们使用共享的多层感知器（MLP）和对称的池化层分别学习每个点的特征和全局特征。在 PointNet 的基础上，产生了一系列基于点的网络。总体而言，这些方法可以大致分为逐点多层感知器 MLP 处理方法，点卷积处理方法，基于循环神经网络（RNN）的方法和基于图的方法

2.1.2.1 逐点多层感知器处理（Pointwise MLP Methods）

这些方法通常使用共享的 MLP 作为其网络中基本的单元来实现高效处理。但是，通过共享的多层感知器（MLP）提取的逐点特征无法捕获点云中的局部几何结构以及点云和点之间的相互作用[16]。为了更好地学习局部结构特征以及扩大每一点的特征所覆盖的周围环境，学术界提出了几种复杂的网络，包括基于邻近特征池化的方法，基于注意力的特征聚合和局部-全局特征串联。

邻近特征池化：为了学习局部的几何特征，这些方法通过汇总来自局部邻近点的特征信息来学习每个点的特征。PointNet++[8] 以层级的方式按照一定半径范围对点进行分组，从而逐步学习更大的局部区域特征。他们提出了多尺度分组和多分辨率分组来克服点云的非均匀性和分部密度变化带来的问题。后来，Jiang 等人[39] 提出了一个名为 PointSIFT 的模块来实现方向和尺寸编码。这个模块通过三阶段有序卷积运算从八个空间方向对结构信息进行编码并提取和串联多尺度的特征以适应不同规模的结构。与 PointNet++ 中使用的分组技术（即球查询）不同，Francis 等人[40] 利用 K 均值聚类和 K-邻近算法（KNN）分别定义世界空间和学习的特征空间中的两个邻域。在同一个类别的点在特征空间中的距离会更近的假设下，他们引入了一个成对距离损失函数和中心损失函数来进一步正则化特征学习。为了学习不同点之间的相互作用，Zhao 等人[41] 提出了 PointWeb，通过密集构造一个局部全连通的网图来探索局部区域的每一对点之间的关系。他们还提出了一个自适应特征调整（AFA）模块以实现信息交换和功特征优化。这个聚合操作能帮助网络学习有辨别度的特征表示。Zhang 等人[42] 基于同心球面上的统计信息提出了一种名为 Shellconv 的排列不变的卷积操作。该方法首先在一些多尺度同心球上进行查询，然后对多个球面使用最大池化操作来总结统计数据，最后使用多层感知器（MLP）和一维卷积来获得最终卷积输出。Hu 等人[43] 提出了一种有效的名为 RandLA-Net 的轻量级的网络进行大规模的点云处理。该网络利用随机采样大大提高了在内存使用和计算量的效率。他们进一步使用局部特征聚合模块捕获和保存几何特征。

基于注意力的聚合：为了进一步提高分割的准确度，出现了一些使用注意力机制[44] 的点云分割网络。Yang 等人[45] 提出了一组乱序的注意力来表征点之间的关系，并提出了一种排列不变，任务无关并且可导的 Gumbel 子集采样（GSS）来替换广泛使用的最远点采样（FPS）方法。此模块对异常值不敏感并且可以选择具有代表性的点的子集。为了更好地捕捉空间点云的分布的情况，Chen 等人[46] 提出了能够感知本地空间（Local Spatial Aware）的网络层来根据点云的空间布局和局部结构来学习空间意识的权重。与条件随机场（CRF）类似，Zhao 等人[47] 提出了一种基于注意力的分数优化（ASR）模块对网络生成的分割结果进行后处理。他们通过使用学到的注意权重对邻近点进行池化来优化初始的分割结果。该模块很容易地结合到现有的深度网络以改善最终的分割效果。

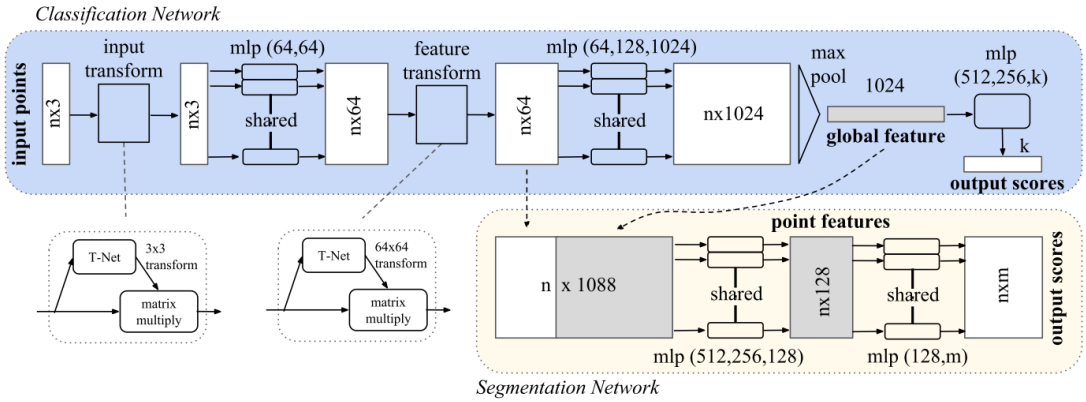


图 2–3 PointNet[16] 的网络结构示意图。

局部-全局串联：Zhao 等人[48] 提出了一个排列不变的 $PS^2 - Net$ 来合并点云的局部结构和全局信息。接着，他们使用多层 Edgeconv[21] 和 NetVLAD[49] 来捕获局部结构和场景级别的全局特征。

2.1.2.2 点卷积处理（Point Convolution Methods）

这些方法试图为点云设计有效的卷积运算。Hua 等人[50] 提出了一个逐点的卷积算子，相邻点被合并到卷积核单元中然后使用卷积核权重进行对点进行卷积运算。Wang 等人[51] 在参数化连续卷积层的基础上提出了一个名为 PCCN 的网络。这个网络层的核函数由多层感知器（MLP）参数化并扩展到连续向量空间。Hughes 等人[52] 在内核点卷积（KPConv）的基础上提出了一个内核全卷积网络（KP-FCNN）。具体来说，内核点卷积 KPConv 的权重由到内核点的欧几里得距离确定且内核点数是可变的。该方法通过求解球体空间最佳覆盖范围的优化问题来确定内核点的位置。他们使用了半径邻域来保持感受野的一致性并在每一层网络中二次采样以实现对点云分部的密度变化的鲁棒性。在[53] 中，Francis 等人提供了充足的拆解实验和可视化结果来展示感受野对基于聚合的方法效果的影响。他们还提出了扩张的点卷积（DPC）操作来汇集扩张的邻近特征，而不是采用 K 个最近的邻点。事实证明，此操作可以有效地增加感受野，并且可以很容易地与现有的基于聚合的网络结合。

2.1.2.3 基于 RNN 的方法（RNN-based Methods）

为了捕获点云里内在的语义环境，递归神经网络（RNN）也已用于点的语义分割云。在 PointNet[16] 的基础上，Francis 等人[54] 首次将一个点云块转换为多尺度的点云块合网格块以获得输入级别的语义环境。然后，他们使用 PointNet 对每一个点云块进行特征提取，然后按顺序输入到合并单位（CU）或递归合并单位（RCU）中来获得输出级别的语义环境。实验结果表明引入空间语义环境对于提升分割效果非常重要。Huang 等人[55] 提出了一种轻量级的局部依赖表征模块，并使用切片池化层将无序的点特征集转换为有顺序的特征向量序列。Ye 等人[56] 首先提出了一个逐点的金字塔池化（3P）模块来捕获从粗到细的局部结构，然后使用双向的分层级的递归神经网络（RNN）来进一步捕获远程的空间依赖性。递归神经网络（RNN）实现了端到端的学习。但是，这些方法在将局部邻近特征合全球结构特征聚合时丢失了点云里丰富的几何特征密度分布特性[57]。为了缓解因固定和静态池化引起的

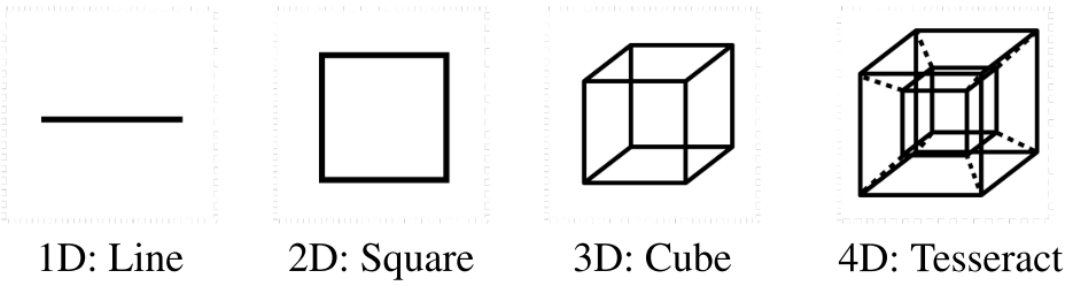


图 2–4 不同维度空间的示意图。

问题，Zhao 等人[57] 提出了动态聚合网络（DAR-Net）来同时考虑全局场景复杂性和局部几何特征，并通过使用自适应的感受野和节点权重来动态地聚合中间层地特征。Liu 等人[58] 提出了 3DCNN-DQN-RNN 来对大规模点云进行有效的点云语义分割。该网络首先使用 3D CNN 学习点云的空间分布和颜色特征，然后进一步使用 DQN 来定位目标类别的对象。最终串联的特征向量被送入残差递归神经网络（RNN）以获得最终的分割结果。

2.1.2.4 基于图的方法（Graph-based Methods）

为了捕获点云底层的 3D 形状和几何结构，一些方法诉诸于图网络。Loic 等人[59] 将点云表示为一组相互连接的简单形状和超点，并使用属性有向图（即超点图）捕获结构和语境信息。然后，大规模点云分割问题被分为三个子问题，即几何同质划分，超点特征提取和语境分割。为了进一步几何同质划分，Loic 和 Mohamed[60] 提出了一个自监督的框架将点云过细分为纯超点。这个问题被表述为构建在邻接图上的深度规则学习问题。此外，该文章还提出了图结构上的对比损失函数来更好地识别物体之间的边界。

为了更好地捕获高维空间中地局部几何关系，Kang 等人[61] 在图嵌入模块（GEM）和金字塔注意力网络（PAN）的基础上提出了 PyramNet。图嵌入模块（GEM）将点云表示为有向无环图并利用一个协方差矩阵来替换的欧几里德距离来构造相似邻接矩阵。金字塔注意力网络（PAN）中使用了具有四种不同大小的卷积核来提取具有不同密度的语义特征。Wang 等人[62] 提出了图注意力卷积（GAC）来有选择地在局部邻近区域学习相关特征。此操作根据点地空间位置和特征差异来动态地分配注意权重。图注意力卷积（GAC）可以学习捕获有区别度地特征来进行分割，并且具有与常用的条件随机长（CRF）有着相似地特性。

2.2 动态点云的语义分割

2.2.1 4D MinkNet

为了处理点云序列，Choy 等人[11] 针对点云的稀疏性[16] 提供了一个可自动求导的 4D（3D 坐标 + 时间维度）稀疏卷积库。

2.2.1.1 稀疏张量及卷积

稀疏张量的表示 为了将稀疏张量紧凑地存储起来，他们使用了 COO 的存储格式。具体来讲，他们为每个点设置一个 4D 坐标，所有点的坐标用集合 $C = \{(x_i, y_i, z_i, t_i)\}_i$ 或矩阵 C 来

表示，对应的点的特征使用集合 $\mathcal{F} = \{f_i\}_i$ 或矩阵 F 进行表示。由此，稀疏张量可以表示为：

$$C = \begin{bmatrix} x_1 & y_1 & z_1 & t_1 & b_1 \\ & & \vdots & & \\ x_N & y_N & z_N & t_N & b_N \end{bmatrix}, F = \begin{bmatrix} f_1^T \\ \vdots \\ f_N^T \end{bmatrix} \tag{2–1}$$

其中，b_i 和 f_i 分别是第 i 个坐标对应的批（batch）坐标和特征向量。

广义稀疏卷积表示 该文为输入输出坐标的一般表示和任意形状的卷积核提出了广义的稀疏卷积操作。他们定义 D 维空间的坐标为 $u \in \mathbb{R}^D$，其对应的 N^{in} 维输入特征向量为 $x_{\mathbf{u}}^{in} \in R^{N^{in}}$，卷积核的权重为 $\mathbf{W} \in \mathbb{R}^{K^D \times N^{out} \times N^{in}}$。他们将权重分为 K_D 个子权重张量 $\mathbf{W}_i \in \mathbb{R}^{N^{out} \times N^{in}}$。于是 D 维空间里常规的密集卷积操作可以表示为：

$$\mathbf{x}_{\mathbf{u}}^{out} = \sum_{\mathbf{i} \in \mathcal{V}^D(K)} W_i \mathbf{x}_{\mathbf{u}+\mathbf{i}}^{in} \quad for \quad \mathbf{u} \in \mathbb{Z}^D \tag{2–2}$$

其中，$\mathcal{V}^D(K)$ 是 D 维超空间中以原点为中心的偏移量列表（如 $\mathcal{V}^1(3) = \{-1, 0, 1\}$）。广义的稀疏卷积可以表示为：

$$\mathbf{x}_{\mathbf{u}}^{out} = \sum_{\mathbf{i} \in \mathcal{N}^D(\mathbf{u}, C^{in})} W_i \mathbf{x}_{\mathbf{u}+\mathbf{i}}^{in} \quad for \quad \mathbf{u} \in C^{out} \tag{2–3}$$

其中，$\mathcal{N}^D$ 是偏移量的集合，该集合定义了卷积核的形状。$\mathcal{N}^D(\mathbf{u}, C^{in}) = \{\mathbf{i} | \mathbf{u}+\mathbf{i} \in C^{in}, \mathbf{i} \in \mathcal{N}^D\}$ 是以 $\mathbf{u}$ 为中心的偏移量。C^{in} 和 $C^Z out$ 是输入和输出的张量坐标集合（为已知变量）。由于输入和输出变量是可以不相同的并且 $\mathcal{N}^D$ 可以定义任意形状的卷积核，这种广义卷积是可以表示多种卷积操作的，如常见的密集卷积、扩大卷积和典型的超立体卷积等。

2.2.1.2 闵可夫斯基引擎

该文章的作者将他们提出的广义卷积库以物理上的时空连续体——闵可夫斯基空间来命名。并提供了一些算法实现细节。

稀疏张量量子化 稀疏卷积神经网络的第一步是数据处理，即生成稀疏张量，将输入转换为唯一的坐标，关联的特征以及可选的标签（在训练语义分割任务时需要）。此过程的 GPU 运算如算法 2–1所示。给定密集标签，需要忽略具有多个标签的体素。这可以通过使用 IGNORE_LABEL 标记这些体素来完成。首先，将所有坐标转换为哈希键值，并找到所有唯一的哈希键值-标签对以消除哈希碰撞。这里，SortByKey，Unique-ByKey 和 ReduceByKey 都是标准的 Thrust 库函数 [19]。归约函数（reduction function）$f((l_x, i_x), (l_y, i_y)) =>$(IGNORE_LABEL,$i_x$) 以标签-键值对为输入并在同一密钥中至少有两个标签-键值对（冲突）时返回 IGNORE_LABEL。CPU 版本的工作方式类似，不同之处在于所有归约（reduction）和排序均时顺序执行的。

广义稀疏矩阵的实现 流程的下一步是根据输入坐标 C^{in} 生成输出坐标 C^{out}（等式 2–3）。在传统的神经网络中，此过程需要给定卷积核的滑动大小（stride size），输入的坐标，并输入稀疏张量的间隔大小（坐标之间的最小距离）。他们提出的广义稀疏卷积则动态地创建此输出坐标，这使得广义稀疏卷积能适应任意的输出坐标 C^{out}。

算法 2–1 GPU 稀疏张量量子化（Sparse Tensor Quantization）

输入: 坐标 $C_p \in \mathbb{R}^{N\times D}$，特征 $F_p \in \mathbb{R}^{N\times N_f}$，目标标签 $l \in \mathbb{Z}_+^N$，量化单位 v_l

$C_p^{'} \leftarrow floor(C_p/v_l)$

$\mathbf{k} \leftarrow hash(C_p^{'}), \mathbf{i} \leftarrow Sequence(N)$

$((\mathbf{i}^{'}, \mathbf{l}^{'}), \mathbf{k}^{'}) \leftarrow SortByKey((\mathbf{i}, \mathbf{l}), key = \mathbf{k})$

$(\mathbf{i}^{''}, (\mathbf{k}^{''}, \mathbf{l}^{''})) \leftarrow UniqueByKey(\mathbf{i}^{'}, key = (\mathbf{k}^{'}, \mathbf{l}^{'}))$

$(\mathbf{l}^{'''}, \mathbf{i}^{'''}) \leftarrow ReduceByKey((\mathbf{l}^{''}, \mathbf{i}^{''}), key = \mathbf{k}^{''}, fn = f)$

return $C_p^{'}[\mathbf{i}^{'''}, :], F_p[\mathbf{i}^{'''}, :], \mathbf{l}^{'''}$

接下来，为了使用卷积核对输入进行卷积，我们需要一个映射来确定哪些输入会影响哪些输出。在常规密集卷积中不需要此映射，因为可以轻松推断出影响关系。但是，稀疏卷积的坐标是任意分散的，因此需要指定映射。他们称此映射称为卷积核映射，并将其定义为输入索引-输出索引对的列表 $M = \{(I_i, O_i)\}_i$ for $i \in \mathcal{N}_D$。最后，在给定输入输出坐标、卷积核映射和卷积核权重 W_i 的情况下，可以用迭代的方式遍历每个偏移 $i \in \mathcal{N}^D$ 来计算广义的稀疏卷积（算法 2–2），其中 $I[n]$ 和 $O[n]$ 表示索引列表 I 和 O 的第 n 个元素，F_n^i 和 F_n^o 分别是第 n 个输入和输出的特征向量。转置的广义稀疏卷积（反卷积）的工作原理类似，只是输入和输出坐标互换角色。

算法 2–2 广义稀疏卷积（Generalized Sparse Convolution）

输入: 卷积核的权重 $\mathbf{W}$，输入的特征 F^i，输出特征的占位符 F^o，卷积位置映射 M

$F^o \leftarrow 0$ //设置为 0

for all $W_i, (I_i, O_i) \in (\mathbf{W}, \mathbf{M})$ **do**

$F_{tmp} \leftarrow W_i[F^i_{I_i[1]}, F^i_{I_i[2]}, ..., F^i_{I_i[n]}]$ // (cu)BLAS（基本线性算法子程序）

$F_{tmp} \leftarrow F_{tmp} + [F^o_{O_i[1]}, F^o_{O_i[2]}, ..., F^o_{O_i[n]}]$

$[F^o_{O_i[1]}, F^o_{O_i[2]}, ..., F^o_{O_i[n]}] \leftarrow F_{tmp}$

end for

最大池化的实现 与密集张量不同，在稀疏张量上，每个池化输出向量所涉及的输入特征向量的数量不同。因此，这为最大/平均池的实现带来了困难。对于 $i \in \mathcal{N}^D$，定义 $\mathbf{I}$ 和 $\mathbf{O}$ 分别是包含所有 $\{I_i\}_i$ 和 $\{O_i\}_i$ 的向量。他们首先找到与每个输出坐标对应的输入的数量及相应的索引。算法 2–3将这些映射到相同输出坐标的输入特征进行池化。$Sequence(n)$ 生成一个从 0 到 $n-1$ 的整数序列，归约函数（reduction function）$f((k_1, v_1), (k_2, v_2)) = min(v_1, v_2)$ 返回给定的两个键值对的最小值。MaxPoolKernel 是一个 CUDA 内核，它使用 S' 对指定通道上的所有特征进行池化，同时返回映射到相同输出的 I 中的第一个索引，以及对应的输出索引 O''。

全局/平均池化，求和池化的实现 平均池化和全局池化层为计算每个输出坐标计算输入特征的平均值来实现平均池化，或着计算唯一输出坐标对应的输入特征的平均值来实现全局池化。这可以以多种方式实现。他们使用稀疏矩阵乘法，可以在硬件上进行优化或使用更快的稀疏基本线性算法子程序（BLAS, Basic Linear Algorithm Sub-program）库。具体来讲，他们使用 cuSparse 库进行稀疏矩阵-矩阵乘法（cusparse_csrmm）和矩阵-矢量乘法（cusparse_csrmv）

算法 2–3 GPU 稀疏张量最大池化（Sparse Tensor MaxPooling）

输入: 特征 F，输出映射 O

$(\mathbf{I}', \mathbf{O}') \leftarrow SortByKey(\mathbf{I}, key = \mathbf{O})$

$\mathbf{S} \leftarrow Sequence(Length(\mathbf{O}'))$

$\mathbf{S}', \mathbf{O}'' \leftarrow ReduceByKey(\mathbf{S}, key = \mathbf{O}', fn = f)$

return $MaxPoolKernel(\mathbf{S}', \mathbf{I}', \mathbf{O}'', F)$

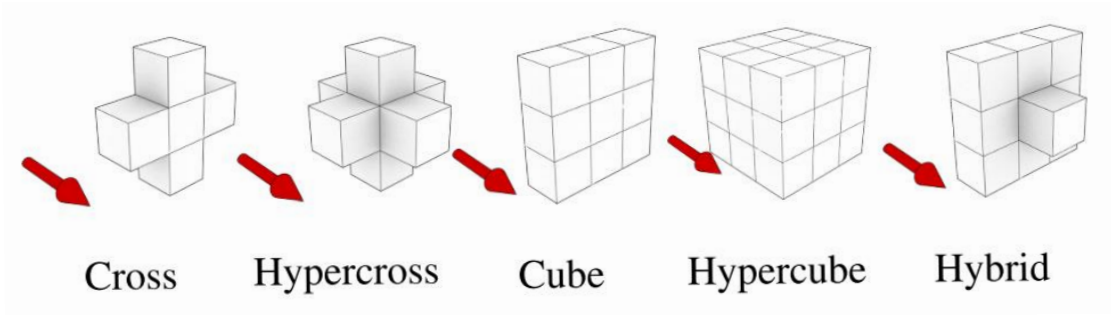


图 2–5 时空卷积的各种形状的卷积核。红色箭头表示时间维度，其他两个轴表示空间维度。这里隐藏了第三个空间维度，以实现更好的可视化。

来实现这些层。与最大池化算法类似，$\mathbf{M}$ 是输入到输出卷积核映射 $(\mathbf{I}, \mathbf{O})$。对于全局池化，他们创建了将所有输入映射到坐标原点的卷积核映射并同样使用算法 2–4进行运算。转置池化（逆池化）的工作原理类似。

在算法 2–4的最后一行，他们将汇总的特征除以映射到每个输出坐标的输入数量。但是，此过程可能会抹去密度信息。因此，我们提出了一种不除以输入数量的变体，并将其命名为求和池化。

算法 2–4 GPU 稀疏张量平均池化（Sparse Tensor AvgPooling）

输入: 映射 $\mathbf{M} = (\mathbf{I}, \mathbf{O})$，特征 F，向量 $\mathbf{1}$

$S_M = coo2csr(row = \mathbf{O}, col = \mathbf{I}, val = \mathbf{1})$

$F' = cusparse\_csrmm(S_M, F)$

$N = cusparse\_csrmv(S_M, \mathbf{1})$

return F'/N

非空间函数实现 对于不需要空间信息（坐标）的函数，例如 ReLU，他们可以直接将函数应用于特征 F 上。此外，对于批量归一化（Batch Normalization），由于 F 的每一行都代表一个特征，因此我们可以直接在 F 上使用一维的批量归一函数。

卷积核形状 为了降低计算复杂度和内存占用以及有效学习时间和空间特征，该文提出了一种结合了立方体卷积核（cubic kernel）和十字形卷积核（cross-shaped kernel）的混合型卷积核（hybrid kernel），如图 2–5所示。对于空间尺寸，我们使用立方核来精确捕获空间几何形状。对于时间维度，我们使用十字形内核将跨时间的空间中的同一点连接起来。

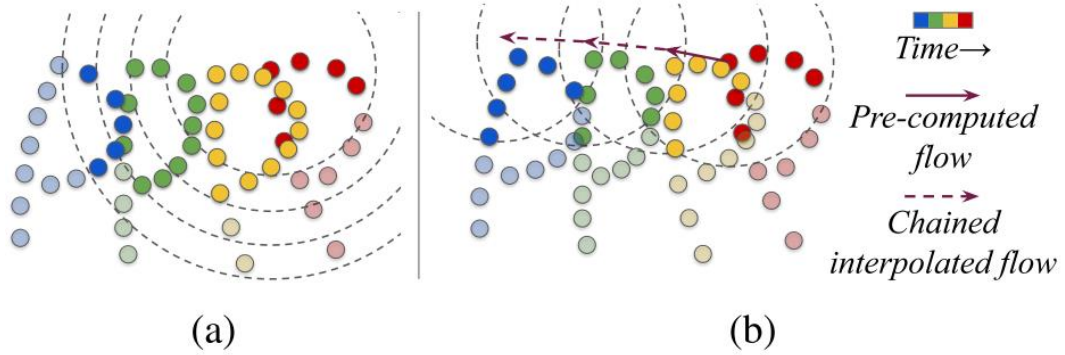


图 2–6 MeteorNet[10] 提出的两种分组（grouping）方法。(a) 为直接分组（direct grouping），(b) 为场景流分组（chained-flow grouping）。

2.2.2 MeteorNet

Liu 等人在 PointNet++[8] 的基础上提出了处理动态点云序列的方法 MeteorNet[10]。

该文将一个包含 T 帧的点云序列表示为 $S = (S_1, S_2, ..., S_T)$。每一帧是一个 3D 点集 $S_t = \{p_i^{(t)} | i = 1, 2, ..., n_t\}$，其中每个点 $p_i^{(t)}$ 包含欧几里得坐标 $x_i^{(t)} \in \mathbb{R}^3$ 和一个特征向量 $f_i^{(t)} \in \mathbb{R}^c$。特征向量可能来自传感器的输入（如颜色），或者来自神经网络的输出。文章中提出的 Meteor 模块以点云的序列为输入，为每个点输出一个更新的特征向量 $h(p_i^{(t)})$。

Meteor 模块首先在点云序列中找到点 $p_i^{(t)}$ 中的邻域 $\mathcal{N}(p_i^{(t)})$。之后，该模块将邻域 $\mathcal{N}(p_i^{(t)})$ 中的其他点 $p_j^{(t')}$ 的特征向量以及 $p_i^{(t)}$ 和 $p_j^{(t')}$ 的时空位置差输入到一个多层感知器（MLP）ζ 和最大池化层（Max Pooling）来计算新的特征向量：

$$h(p_i^{(t)}) = \underset{p_j^{(t')} \in \mathcal{N}(p_i^{(t)})}{MAX} \{\zeta(f_j^{(t')}, x_j^{(t')} - x_i^{(t)}, t' - t)\}. \tag{2–4}$$

为了找到点的邻域 $\mathcal{N}$，文章中提出了两种方法，分别是直接分组（direct grouping）和场景流分组（chained-flow grouping），下面将详细介绍。

2.2.2.1 分组方法

直接分组（direct grouping） 直觉上，一个物体运动的最大距离会随着时间的增加而增加。因此，可以随着 $|t - t'|$ 的增加而逐步增加邻域 $\mathcal{N}$ 的搜索半径来覆盖点的运动范围。邻域可以表示为：

$$\mathcal{N}_d(p_i^{(t)}; r) = \{p_j^{(t')} | |x_j^{(t')} - x_i^{(t)}| < r(|r - r'|)\} \tag{2–5}$$

其中，r 是一个单调增函数。值得一提的是 $r(0) > 0$ 因此同一帧中的点也会被搜索到。直接分组如图 2–6(a) 所示。

场景流分组（chained-flow grouping） 在现实中，某个物体上的点通常在其他帧的相邻区域内有对应的点，这些对应的点连起来就构成了该点的运动轨迹。这种运动可以由场景流（scene flow）[28] 来进行表示。

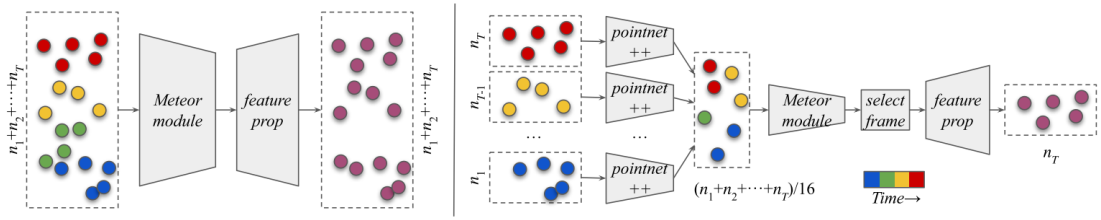


图 2–7 MeteorNet[10] 提出的两种结构，左边为初期融合（early fusion），右边为后期融合（late fusion）。

对于任一时间 t，Meteor 模块首先估计从时间 t 到 $t-1$ 的逆向场景流 $\delta_i^{(t,t-1)} \in \mathbb{R}^3$：

$$\{\delta_i^{(t,t-1)}\}_i = \mathcal{F}_0(\{p_i^{(t)}\}, \{p_j^{(t-1)}\}) \tag{2–6}$$

其中 $\mathcal{F}_0$ 是估计从时间 t 到 $t-1$ 的场景流的工具（如 FlowNet3D[63]），然后可以计算点 $p_i^{(t)}$ 在 $t-1$ 帧的虚拟点 $x'^{(t-1)}_i = x_i^{(t)} + \delta_i^{(t,t-1)}$。

为了估计点 $p_i^{(t)}$ 在第 $t-2$ 帧的位置，他们使用虚拟点 $p'^{(t-1)}_i$ 对从第 $t-1$ 帧到第 $t-2$ 帧的场景流估计结果 $\{\delta_j^{(t-1,t-2)}\}_j$ 进行插值。文章中使用最简单的距离倒数为权重的 k 最近点插值：

$$\delta'^{(t-1,t-2)}_i = \frac{\sum_{j=1}^{k} w(x_j^{(t-1)}, x'^{(t-1)}_i)\delta_j^{(t-1,t-2)}}{\sum_{j=1}^{k} w(x_j^{(t-1)}, x'^{(t-1)}_i)} \tag{2–7}$$

其中，$w(x_1, x_2) = \frac{1}{d(x_1,x_2)^2}$ 是插值的权重。该文章默认使用 $p=2, k=2$。然后使用如下公式计算第 $t-2$ 帧中的点 $p_i^{(t)}$ 的位置：$x'^{(t-2)}_i = x_i^{(t)} + \delta_i^{(t,t-1)} + \delta'^{(t-1,t-2)}_i$。比第 $t-2$ 帧更远的帧中的对应位置可以通过重复上述过程来计算。这个计算过程如图 2–6(b) 所示。场景流分组的邻域定义如下：

$$\mathcal{N}_c(p_i^{(t)}; r) = \{p_j^{(t')} | |x_j^{(t')} - x'^{(t')}_i| < r\} \tag{2–8}$$

这里 r 是一个常数，但是仍然可以使用 $|t-t'|$ 的单调增函数来补偿场景流估计的误差。

相比于直接分组（direct grouping），场景流分组（chained-flow grouping）可以追踪点的运动轨迹。

2.2.2.2 结构设计

对于使用 Meteor 模块构建神经网络，针对融合多帧的时间，可以有以下两种设计方案：

初期融合（early fusion） 在神经网络的第一层就应用 Meteor 模块。如图 2–7左侧所示，从一开始就混合了来自不同帧的点。

后期融合（late fusion） 我们应用了几层能够提取点云中的几何特征的网络（例如 PointNet++[8]）分别处理每个帧中的点云，然后再将它们在 Meteor 模块中混合，如图 2–7右侧所示。它使得模型能够捕获更高级别的语义特征。

该文对于语义分割任务使用了第一种方案，即前期融合。

2.2.2.3 理论分析

该文章对其所提出的 Meteor 模块对以点云序列为输入的连续函数的近似能力。

具体而言，假设 $\mathcal{X}_t = \{S_t | S_t \subseteq [0,1]^m, |S_t| = n, n \in \mathcal{Z}^+\}$ 是第 t 帧所有点云的集合。他们使用 $d_H(S_i, S_j)$ 来表示单帧点云之间的豪斯多夫距离，其中 $S_i \in \mathcal{X}_i, S_j \in \mathcal{X}_j$。$\mathcal{X} = \mathcal{X}_1 \times \mathcal{X}_2 \times ... \times \mathcal{X}_T$ 表示长度为 T 的点云序列的集合。定义点云序列之间的豪斯多夫距离 $d_{seq}(\cdot,\cdot)$ 为各对应帧之间单帧豪斯多夫距离的最大值，即 $d_{seq}(S, S') = max_t\{d_H(S_t, S'_t)\}$。该文定义 $f : \mathcal{X} \to \mathbb{R}$ 是在集合 $\mathcal{X}$ 上关于 $d_{seq}(\cdot,\cdot)$ 的连续函数，即 $\forall \epsilon > 0, \exists \delta > 0$ 对任何满足 $d_{seq}(S, S') < \delta$ 的序列 $S, S' \in \mathcal{X}$，有 $|f(S) - f(S')| < \epsilon$。下面的定理证明了具有足够大特征长度的神经网络和最大池化层能够无限接近该连续函数 f：

定理 2.1 假设 $f : \mathcal{X}_1 \times \mathcal{X}_2 \times ... \times \mathcal{X}_T \to \mathbb{R}$ 是关于豪斯多夫距离 $d_{seq}(\cdot,\cdot)$ 的连续的集函数。$\forall \epsilon > 0, \exists$ 一个连续函数 $\zeta(\cdot,\cdot)$ 和一个连续函数 γ，对任意 $S = (S_1, S_2, ..., S_T) \in \mathcal{X}_1 \times \mathcal{X}_2 \times ... \times \mathcal{X}_T$ 有：

$$\left| f(S) - \gamma \circ \left(\underset{x_i^{(t)} \in S_t, t \in \{1,2,...,T\}}{MAX} \{h(x_i^{(t)}, t)\} \right) \right| < \epsilon$$

其中，$x_1^{(t)}, x_2^{(t)}, ..., x_n^{(t)}$ 是集合 S 中的顺序无关的元素，MAX 是求多个向量每个位置元素最大值的函数。

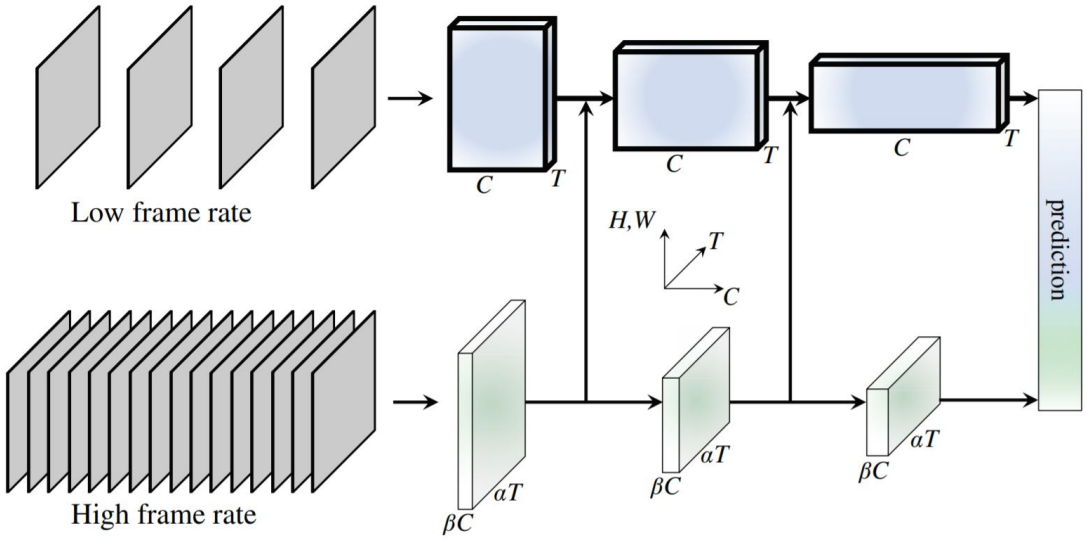


图 2–8 SlowFast[64] 的网络结构示意图，其中 $\alpha > 0$，即时间分路在时间维度上有着更高的分辨率而 $\beta < 0$，即时间分路更加轻量化。

2.3 神经网络的时空感知

对于神经网络的时空感知问题，之前有工作在 2D 视频的分析上进行过探索。如双支路卷积网络[65] 设计了一种双支路的网络结构，一个为空间支路（Spatial Stream）另一个为时间支路（Temporal Stream），空间支路以图片作为输入，而时间支路以多帧的光流信息作为

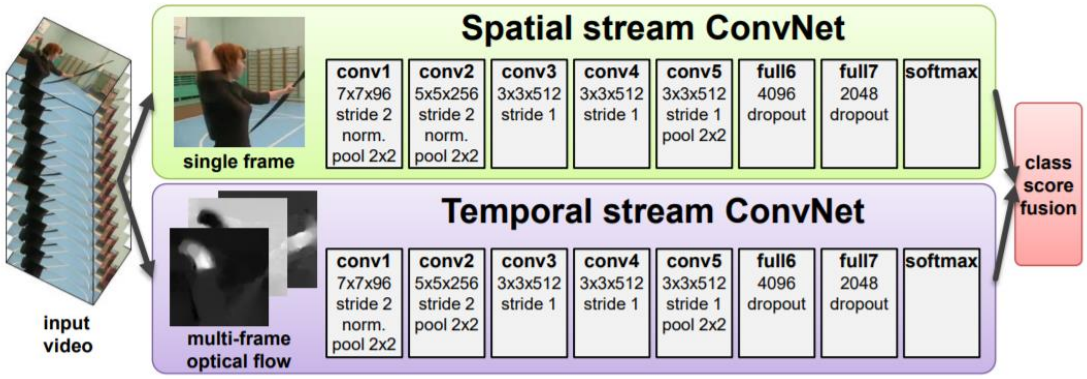


图 2–9 双支路卷积[65] 的网络结构示意图。

输入，最后两路的预测结果进行耦合，其整体结构如图 2–9所示。此外，考虑到物体的形状颜色等空间信息往往变化较慢而动作信息往往变化较快，SlowFast[64] 提出使用两个帧率不同的分路来分别处理时间和空间信息，即使用低帧率的分路处理空间信息而使用高帧率、轻量化的分路来处理时间信息，其网络结构示意图如图 2–8所示。

第三章　时空感知的动态点云语义分割

随着激光雷达（LiDAR）以及深度相机技术的发展，动态点云序列已经可以成为许多视觉任务的输入。对动态点云进行分割是感知系统的重要功能之一，对诸如自动驾驶[66]，机器人导航[67]和增强现实[68]等应用有着重大的意义。在本文中，我们致力于解决动态点云序列上的语义分割任务。尽管静态点云的语义分割[8, 9, 16, 17, 40]已取得了巨大的成功，但动态点云序列的语义分割问题尚未得到充分的探索。对于静态点云分割，开拓性的工作 PointNet[16]展示了其直接从原始点云中学习逐点特征的能力。但是，单个点包含的信息非常有限，只有与相邻近的点一起考虑时才具有语义。静态点云上的后续工作[8, 17, 40]对一组邻近的局部点云结构进行编码来捕获局部结构的信息以及点和点之间的关联。为了处理动态点云序列，一种直观而直接的方法是在时空邻域内对局部结构同时进行时间和空间上的编码，例如最新的工作[10, 11]便遵循这个方向。4D MinkNet[11]首先将点云转换为规则的体素，然后沿时间和在空间和时间维度应用广义的稀疏卷积。MeteorNet[10]则直接堆叠多帧点云，并根据点的欧几里得坐标对相邻点进行分组并计算局部特征，由于邻近点可以来自不同帧的点云，这种操作同时包含的一定的时间和空间信息。尽管这两种方法都可以学习时空特征，但他们的处理方式很难解耦时空信息。有证据表明，在现实世界中，人类视觉系统主要依赖于时间信息[12]来感知周围的动态环境，并将其视为主要的信息来源。如果不将空间和时间结构解耦，就无法很好地理解时间信息是如何辅助动态场景理解的。同时，时间和空间结构的解耦可以使模型更加可扩展和灵活，例如，我们可以非常方便地将静态点云方法用作基干网络，这将在章节 3.2.2和章节 4.3.1中看到。有效进行动态点云的语义分割需要对其中的时间和空间特征同时进行学习，因此，本文首先从人脑中时间信息和空间信息的处理获得启发，对神经网络的时空感知进行探索：

3.1　时空感知研究

3.1.1　概述

研究表明，在人脑中，时间信息和空间结构信息是分成两个支路分别输入到海马体中的[15]。这表明时间和空间概念在人脑中有不同的学习和感知机制。但同时，由于时间和空间信息是密不可分的，为了能有效地理解时序数据（如视频、点云序列等），这两方面的数据必须通过适当的方式进行融合。在人脑的这种工作方式的启发下，我们尝试对深度神经网络中时间和空间信息的解耦和融合进行研究。

给定输入 $\mathbf{x}$，深度神经网络对时间和空间的感知可以抽象为：

$$\mathcal{F}[h_s(\mathbf{x};\psi_s), h_t(\mathbf{x};\psi_t)] \tag{3–1}$$

其中，ψ_s 和 ψ_t 是需要学习的参数，h_s 用于提取空间信息，而 h_t 则用于处理时间信息。这两种信息被送到 $\mathcal{F}$ 中，$\mathcal{F}$ 就像海马体一样最终输出融合了两种信息的处理结果。

相关研究表明，在人脑中，这两种信息的处理是深度交织的[22]。这让我们自然想到使用多层神经网络来模拟这个过程。我们将第 i 层的处理单元表示为：

$$\mathcal{G}_i(\mathbf{x}_i) = \mathcal{F}[h_s(\mathbf{x}_i;\psi_s^i), h_t(\mathbf{x}_i;\psi_t^i)] \tag{3–2}$$

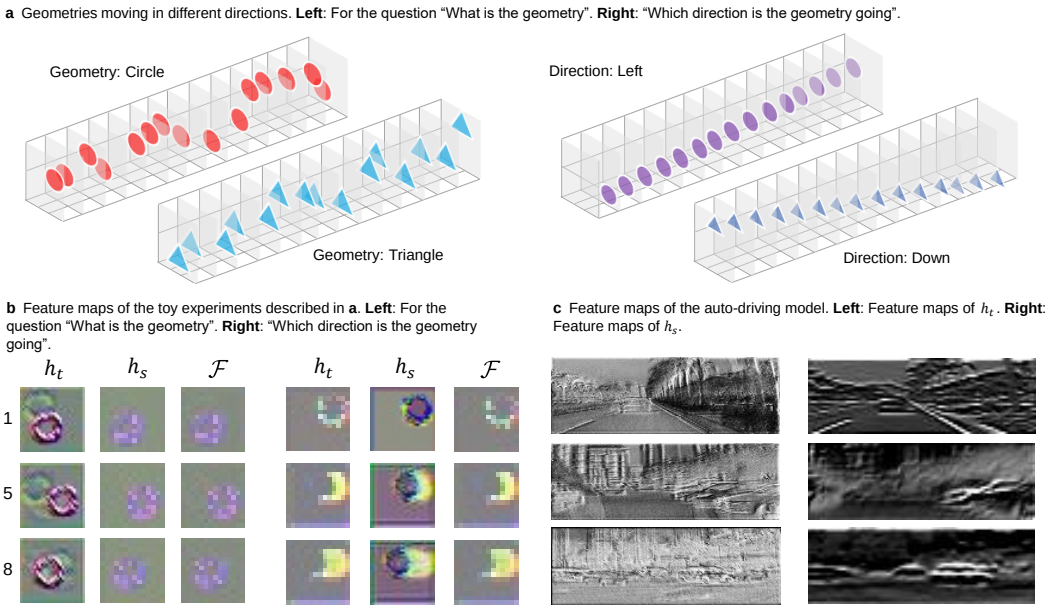


图 3–1 为说明我们提出的半耦合结构（SCS，Semi-Coupled Structure）的有效性进行的实验。a：用来说明 SCS 能够对时间和空间信息进行解耦的实验。输入的序列是移动的几何体，我们让 SCS 同时对几何体的形状（左边的两个序列）以及几何体的运动方向进行分类（右边的两个序列）。b：在 a 所描述的实验中，网络顶层的 h_s，h_t 和 $\mathcal{F}$ 的的特征图。可以看到，在进行几何体形状分类时，h_s 和 $\mathcal{F}$ 中仅包括形状特征，并且由于 $\mathcal{F}$ 的存在，h_t 中的特征并没有干扰到 h_s；对于几何体运动方向的分类，我们可以看到，$\mathcal{F}$ 中包含了 h_t 中的时间信息，而 h_s 中的时间信息也相对增多了。c：驾驶行为预测实验中 h_t 和 h_s 的特征图。可以看到，h_t 中多包含了场景的变化信息而 h_s 则更多是物体的形状和路面信息。

这时，多层（深度）神经网络便可以表示为：

$$\mathcal{T}[\mathbf{x};\Psi_s,\Psi_t] = \mathcal{G}_1 \circ \mathcal{G}_2 \circ ... \circ \mathcal{G}_n(\mathbf{x}) \tag{3–3}$$

其中 n 是神经网络的深度，而 $\Psi_s = \{\psi_s^1, ..., \psi_s^n\}$ 和 $\Psi_t = \{\psi_t^1, ..., \psi_t^n\}$ 是神经网络的可学习参数集。在这种结构中，$h_s(\cdot)$ 和 $h_t(\cdot)$ 分别负责空间和时间概念处理，两类信息具有一定的独立性（解耦），但两者又是深度交织在一起的（耦合），因此这是一个半耦合的结构。

我们将本文提出的半耦合结构称为 SCS（Semi-Coupled Structure）。图 3–1中可以看到 SCS 在时空解耦的有效性。接下来，我们将对神经网络的详细结构和训练过程进行详细介绍。

3.1.2 双支路网络结构设计

我们将神经网络表示为 $\mathcal{T}$，神经网络的层数为 n。在每个时间点（如视频中的一帧）t，该神经网络从数据集或环境中接收输入矩阵 $\mathbf{x}_t$ 并输出向量 $\mathbf{y}_t$ 来近似目标向量（真实情况）$\mathbf{z}_t$。

如上文说描述，每个半耦合神经网络层的结构为 $\mathbf{u}_t^l = \mathcal{F}(h_s(\mathbf{u}_t^{l-1}), h_t(\mathbf{u}_t^{l-1}))$，其中 $\mathbf{u}_t^l$ 是第 t 步第 l 层网络的输出，而 $\mathbf{u}_t^{l-1}$ 是其输入。我们定义 $\mathbf{u}_t^0 = \mathbf{x}_t$，于是我们得到：

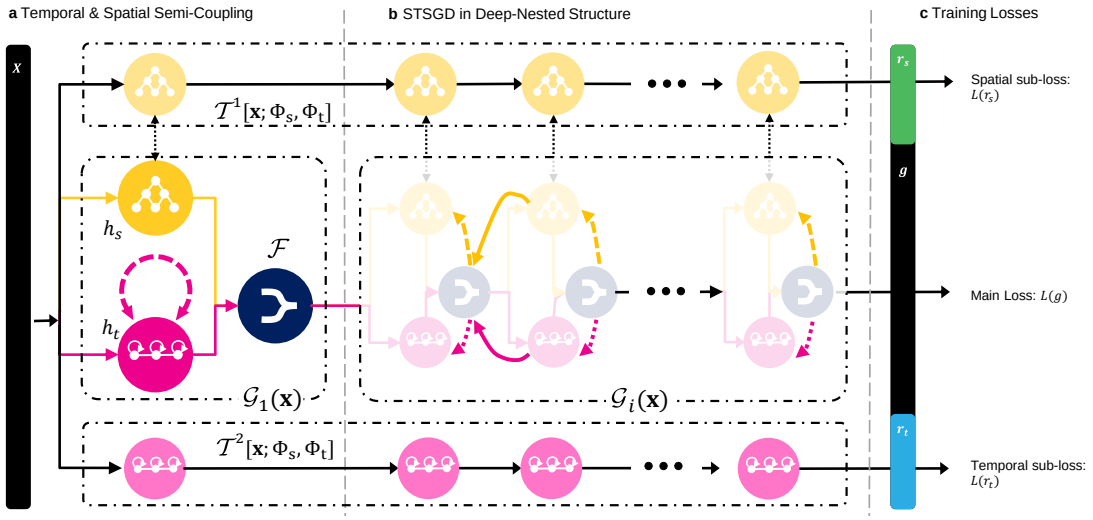


图 3–2 半耦合结构的整个流程。**a**，输入 $\mathbf{x}$，$\mathcal{G}$ 通过 h_t 和 h_s 解耦时空信息，h_t 专注于时间特征，而 h_s 主要提取空间特征，$\mathcal{F}$ 将它们融合在一起以形成完整的时空半耦合结构。**b**，为使半耦合的结构能适用于深度神经网络中，我们设计了时空转换梯度下降（STSGD）方法进行训练（请参见 3.1.3），该方法可阻止梯度沿某些路径向后传播（虚线），即在训练过程以一定的概率 p 解耦 h_s 和 h_t。$\mathcal{T}^1$ 和 $\mathcal{T}^2$ 用于使 $h_s(\cdot)$ 和 $h_t(\cdot)$ 进一步扮演好自己的角色并监控 h_s 的训练过程来调整 Advanced STSGD（ASTSGD）中的 q。**c**，除了基于主要目标 g 的半耦合结构的主要训练损失 $L(g)$ 之外，还有另外两个基于子目标 r_s 和 r_t 的损失 $L(r_s)$，$L(r_t)$，分别对应 $\mathcal{T}^1$ 和 $\mathcal{T}^2$，从而指导 h_s 和 h_t 分别关注空间和时间特征。

$$\mathbf{s}_t^l = h_s(\mathbf{u}_t^{l-1};\psi_s^l) = \mathrm{Conv}(\mathbf{u}_t^{l-1};\psi_s^l) \tag{3–4}$$

$$\mathbf{c}_t^l = h_t(\mathbf{u}_t^{l-1};\psi_t^l) = \mathrm{Conv}([\mathbf{u}_t^{l-1},\sigma(\mathbf{c}_{t-1}^l)];\psi_t^l) \tag{3–5}$$

其中 l 是神经网络层的编号，$\sigma(x) = 1/(1+exp(-x))$ 是 Sigmoid 函数，Conv 是卷积神经层，ψ_s^l 和 ψ_t^l 分别是第 t 帧，第 l 层的空间信息处理单元和时间信息处理单元的参数。对于所有 l 都有 $\mathbf{c}_0^l = \mathbf{0}$。公式 3–4描述了 h_s 的结构，我们使用 Conv 是因为它有出色的空间特征提取能力。当然，根据不同的任务，我们完全可以用其他运算代替 Conv（如全链接层）。公式 3–5描述了 h_t，这是一个简单的递归神经网络（RNN）的结构。同样，我们可以使用任何其他运算，如使用用 LSTM 架构[69] 替换 h_t 是可行的，但是计算复杂度太高，无法应用于实际任务，因此我们在本文中不进行实践。

合成单元 $\mathcal{F}$ 采用无参数的运算：

$$\mathbb{u}_t^l = \mathrm{Relu}(\mathbf{s}_t^l) * \mathrm{Sigmoid}(\mathbf{c}_t^l) \tag{3–6}$$

其中 $*$ 表示逐元素乘法，$\mathrm{Relu}(x) = max(0,x)$ 是校正的线性单位（Rectified Linear Unit），$\mathrm{Sigmoid}(x) = 1/(1+e^{-x})$ 是 sigmoid 函数。这样，h_t 的输出被归一化到 $(0,1)$ 的范围，因此从 $\mathcal{F}$ 的角度来看，h_t 可以被视为对 h_s 的结果过滤单元。

由于网络是递归的，其输出是序列 $(\mathbf{x}_1,...,\mathbf{x}_t)$ 的函数。我们可以进一步将网络的运算表

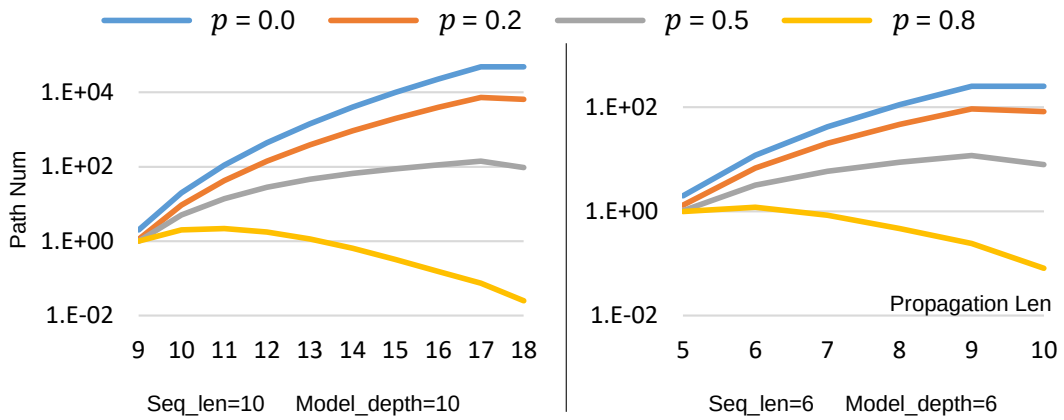


图 3–3 反向传播链的路径数目。水平轴是反向传播链的长度，垂直轴是路径的数目。可以看到，随着模型深度和序列长度的增加，反向传播路径的数量和长度也会显着增加。p 较大的 STSGD 可以有效减少长序列的数量。

示为

$$\mathbb{I}(\mathbf{u}_1^n, ..., \mathbf{u}_t^n) = \mathcal{T}([\mathbf{x}_1, ..., \mathbf{x}_t]; \Psi_s, \Psi_t) \tag{3–7}$$

其中 $\boldsymbol{\Psi}$ 是可训练网络权重的集合，而 $\mathbf{u}_t^n$ 是时间戳 t 上第 n 层的输出。最后，输出向量 $\mathbf{y}_t$ 由 $(\mathbf{u}_1^n, ..., \mathbf{u}_t^n)$ 的组合定义：

$$\mathbb{I}\mathbf{y}_t = [\mathbf{u}_1^n, ..., \mathbf{u}_t^n] \tag{3–8}$$

对于子网络 $\mathcal{T}^1$ 和 $\mathcal{T}^2$，我们采用与 $\mathcal{T}$ 相同的空间和时间单元——$h_s(\cdot)$ 和 $h_t(\cdot)$，而合成单元 $\mathcal{F}$ 不同。在 $\mathcal{T}^1$ 中，$\mathcal{F}$ 和子目标 r_s（或 $\mathbf{y}_t^{\mathcal{T}^1}$）定义为：

$$\hat{\mathbf{u}}_t^l = \text{Relu}(\mathbf{s}_t^l) \tag{3–9}$$

$$\mathbf{y}_t^{\mathcal{T}^1} = [\hat{\mathbf{u}}_1^n, ..., \hat{\mathbf{u}}_t^n] \tag{3–10}$$

在 $\mathcal{T}^2$ 中，$\mathcal{F}$ 和子目标 r_t（或 $\mathbf{y}_t^{\mathcal{T}^2}$）定义为：

$$\hat{\mathbf{u}}_t^l = \text{Relu}(\mathbf{c}_t^l) \tag{3–11}$$

$$\mathbf{y}_t^{\mathcal{T}^2} = [\hat{\mathbf{u}}_1^n, ..., \hat{\mathbf{u}}_t^n] \tag{3–12}$$

3.1.3 半耦合时空学习策略

如3.1.1部分所述，一方面，$\mathcal{G}$ 通过不同的模块分别计算空间和时间特征。另一方面，我们仿照人脑的时空耦合处理的方式，采用了多层 $\mathcal{G}$ 的深层嵌套结构。但是这种结构增加了训练过程的困难性，因为深层嵌套的结构实际上在较浅的层中就较早地合并了时间和空间信息，这加剧了深层中时空特征分解的压力，并减弱了相对独立的结构特征学习。同时，随着网络结构深度和序列长度的增加，反向传播链路径的数量和长度都将显著增加，这增加了计算复杂度，也使训练过程变得更有挑战性（参见图3–3）。

为了解决这一问题，本文提出了时空转换梯度下降（STSGD, Spatial-Temporal Switch Gradient Descent）的方法在训练这一层级进行半耦合，优化器在每个训练步骤以一定概率沿某些路径反传来更新空间或时间单元的参数。随着训练的进行，我们减少了这种分离更新参数的程度，最终网络可以学习所有信息。这种训练策略也是一种半耦合机制：首先解耦，然后进行耦合。

时空转换梯度下降（STSGD） STSGD 是一种基于梯度的优化方法，并且通过 BP（Back Propagation）算法来传播梯度[70]。它就像一个开关，以一定的概率关闭空间和时间模块上的梯度。该方案大大减少了由深层嵌套结构引起的 $h_s(\cdot)$ 和 $h_t(\cdot)$ 之间的互相干扰，并降低了计算复杂度。

神经网络的正向传播为：

$$\mathbf{y}_t = \mathcal{G}_n \circ \cdots \circ \mathcal{G}_1 \tag{3–13}$$

其中 $\mathcal{G}_i = \mathcal{F}[h_t(\cdot), h_s(\cdot); \psi_i]$ 是网络的第 i 层，而 $\boldsymbol{\psi}_i$ 是第 i 层中的可训练参数的集合。输出的结果 $\mathbf{y}_t$ 和事实（ground truth）$\mathbf{z}_t$ 之间的损失（loss）定义为：

$$E = \sum_{t=1}^{T} E_t = \sum_{t=1}^{T} L(\mathbf{y}_t, \mathbf{z}_t) \tag{3–14}$$

其中 L 是损失函数。

然后，在反向传播期间，根据 BPTT（随时间反向传播，Back Propagation Through Time）算法[71]，可以将 $\boldsymbol{\psi}_i$ 的梯度表示为：

$$\frac{\partial E}{\partial \boldsymbol{\psi}_i} = \sum_{t=1}^{T} \frac{\partial E}{\partial \mathcal{G}_i^t} \frac{\partial \mathcal{G}_i^t}{\partial \boldsymbol{\psi}_i} \tag{3–15}$$

在传统的随机梯度下降方法中，采用此精确梯度来更新参数并继续向后传播。在我们的 STSGD 中，我们需要根据这两个模块所携带的信息来解耦梯度。为此，我们将梯度重写为：

$$\frac{\partial E}{\partial \boldsymbol{\psi}_i} = \sum_{t=1}^{T} \frac{\partial E}{\partial \mathcal{G}_i^t} \left(\frac{\partial \mathcal{G}_i^t}{\partial \mathbf{c}_t^i} \frac{\partial \mathbf{c}_t^i}{\partial \boldsymbol{\psi}_i} + \frac{\partial \mathcal{G}_i^t}{\partial \mathbf{s}_t^i} \frac{\partial \mathbf{s}_t^i}{\partial \boldsymbol{\psi}_i} \right) \tag{3–16}$$

在等式中，括号中的第一项是来自 $h_t(\cdot)$ 的梯度，第二项是来自 $h_s(\cdot)$ 的梯度。由于 $h_t(\cdot)$ 和 $h_s(\cdot)$ 分别用于处理时间和空间信息，因此它们的梯度具有不同的含义。

为了解耦梯度，我们使用概率开关函数来关掉某个部分（空间或时间单元）在反向传播中的梯度，可以将其定义为：

$$\begin{aligned} \frac{\partial \hat{E}}{\partial \boldsymbol{\psi}_i} = \sum_{t=1}^{T} & \left(\gamma_t(p_t) \frac{\partial E}{\partial \mathcal{G}_i^t} \frac{\partial \mathcal{G}_i^t}{\partial \mathbf{c}_t^i} \frac{\partial \mathbf{c}_t^i}{\partial \boldsymbol{\psi}_i} \right. \\ & \left. + \gamma_t(p_s) \frac{\partial E}{\partial \mathcal{G}_i^t} \frac{\partial \mathcal{G}_i^t}{\partial \mathbf{s}_t^i} \frac{\partial \mathbf{s}_t^i}{\partial \boldsymbol{\psi}_i} \right) \end{aligned} \tag{3–17}$$

其中 γ 是概率开关函数：

$$\gamma(p) = \begin{cases} 0, \text{ with the probability of } p \\ 1, \text{ with the probability of } (1-p) \end{cases} \tag{3–18}$$

讨论 该方案通过将 p_s 和 p_t 初始化为一个较大的值（如 $p_s = p_t = 0.5$）来部分解除空间和时间学习过程的耦合，并且随着训练的进行，p 减小为 0 来同时进行时间和空间信息的学习。从宏观的角度来看，它以一定的概率切断了反向传播中的某些路径，从而大大减少了反向传播链的数量，从而使训练过程更容易处理（请参见图 3–3）。根据[72] 中的假设 4.3，如果我们设置 $p_s = p_t$，我们将得到 $E(\frac{\partial \hat{E}}{\partial \boldsymbol{\psi}_i}) = \frac{\partial E}{\partial \boldsymbol{\psi}_i}$，这使得 STSGD 有与常规随机梯度下降方法相似的收敛特性。

高级 STSGD（ASTSGD，Advanced STSGD） 注意，在 STSGD 中，$h_s(\cdot)$ 和 $h_t(\cdot)$ 中的 p 值是相等的，这是收敛的充分条件。但是，我们希望网络在开始时可以更多地学习空间信息，因为时间特征的捕获是建立在可靠的空间特征的基础之上的。在获得相对充足的空间特征之后，我们希望 STSGD 可以在空间和时间特征之间转移其学习重点。为此，我们将中的动态比率 $q \in [0, 1]$ 引入到公式 3–17中：

$$\begin{aligned}\frac{\partial \hat{E}}{\partial \boldsymbol{\psi}_i} = \sum_{t=1}^{T} (&\gamma_t(q) \frac{\partial E}{\partial \mathcal{G}_i^t} \frac{\partial \mathcal{G}_i^t}{\partial \mathbf{c}_t^i} \frac{\partial \mathbf{c}_t^i}{\partial \boldsymbol{\psi}_i} \\ &+ \gamma_t(1-q) \frac{\partial E}{\partial \mathcal{G}_i^t} \frac{\partial \mathcal{G}_i^t}{\partial \mathbf{s}_t^i} \frac{\partial \mathbf{s}_t^i}{\partial \boldsymbol{\psi}_i})\end{aligned} \tag{3–19}$$

尽管没有理论可以保证高级 STSGD（ASTSGD）的收敛性，但实验结果表明该方法是非常有效的。此外，为了自动控制 q 减小的过程，我们设计了以下公式：

$$q = q_0 + (1 - q_0) \frac{\max(0, L_s - \text{thresh})}{\text{InitL}_\text{g} - \text{thresh}} * (\alpha(\frac{L_g}{L_s} - 1) + 1) \tag{3–20}$$

其中 L_s 和 L_g 是 r_s 和 g 的损失值。q_0 通常设置为 0.5。在此等式中，我们根据 L_s 的下降来更新 q。thresh 是 L_s 的阈值超参数，考虑到 L_s 很难减小到 0，并且我们希望 q 在 L_s 减小到 L 时获得最小值。InitL_g 是 g 初始的训练损失值（例如，n 分类问题的初始损失，即初始交叉熵损失，是 $ln(n)$）。α 是一个超参数，$(\alpha(L_g/L_s - 1) + 1)$ 用于平衡整体和空间信息的比例。如果任务更依赖于空间信息，我们可以将 α 设置得更大，这样函数将具有相对较大的值，可以更好地学习空间特征。

3.2 动态点云语义分割

上文对神经网络的时空感知进行了研究，在此基础上，本文进一步拓展到 3D 点云中。3D 点云与传统的 RGB 图片或视频有着截然不同的数据格式，针对 3D 点云的特点，为了有效地编码时间信息，我们需要解决以下挑战：

（a）**不同帧特征的融合**：来自不同帧的特征可能对最终结果的重要程度不同，并且它们都可能包含不希望有的噪音或错误。理想情况下，神经网络应该有自动识别不同帧的重要性或置信度的能力，从而获得更好的融合效果。

（b）**跨帧点的关联**：为了融合来自不同帧的特征，我们需要使跨帧的点相关。但是，动态点的分布会不时变化，并且它们是无序的，同时实际应用如自动驾驶等往往对处理速度要求较高，这些原因使得跨帧点的关联相对困难。

在我们的 ASAP 模块中，我们提出了一种新的注意力机制的时间嵌入层和一个时空关联策略来分别解决上述两个挑战。我们还进行了详尽的实验，以证明我们的 ASAP 模块相

对于最新方法[10, 11] 的优势及其提高不同基干网性能的通用能力。Figure 4–3提供了我们方法的定性结果，显示了我们的 ASAP 模块的有效性。

在本章节，我们会介绍我们的 ASAP 模块和整体结构 ASAP-Net，即 ASAP 模块 + 基干网络。

3.2.1 PointNet++ 回顾

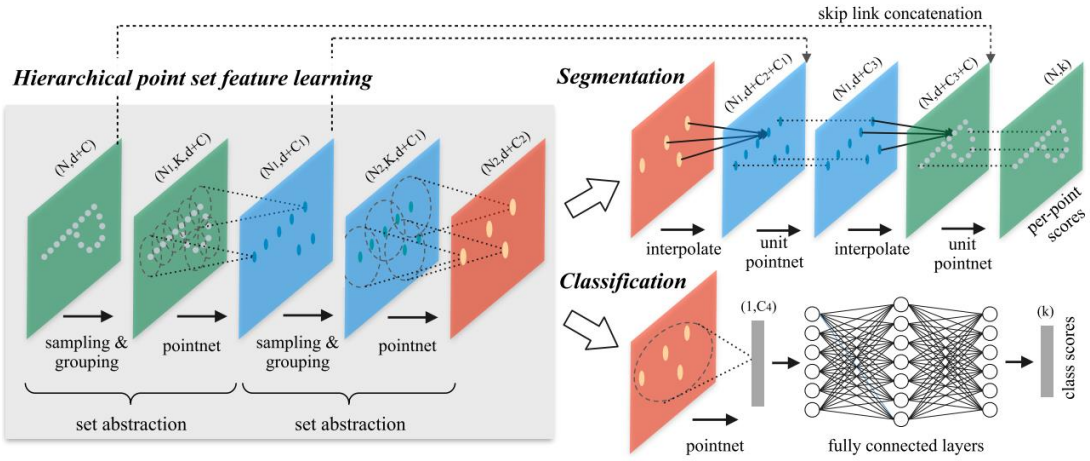


图 3–4 PointNet++[8] 的网络结构图，其中上半部分是为语义分割设计的结构，左侧为集合提取层（set abstract layer），右侧为特征传播层（feature propagation layer）。

PointNet[16] 是深度学习点云处理的开创性工作之一。为了解决点云中的点的无序性问题并捕获点与点之间的相互作用，PointNet 提出使用对称函数（顺序无关）多层感知器（MLP）和最大池化层对点云数据进行处理，并在理论上证明其所提出的网络结构能近似关于豪斯多夫距离的连续集函数 $f : \mathcal{X} \to \mathbb{R}, \mathcal{X} = \{S : S \subseteq [0,1]^m, |S| = n\}$，即：

定理 3.1 假设 $f : \mathcal{X} \to \mathbb{R}$ 是关于豪斯多夫距离 $d_H(\cdot,\cdot)$ 的连续的集函数，$\forall \epsilon > 0, \exists$ 一个连续函数 h 和一个对称函数 $g(x_1, x_2, ..., x_n) = \gamma \circ MAX$，对任意 $S \in \mathcal{X}$ 有：

$$\left| f(S) - \gamma \left(\underset{x_i \in S}{MAX} \{h(x_i)\} \right) \right| < \epsilon$$

其中，$x_1, x_2, ..., x_n$ 是集合 S 中的顺序无关的元素，γ 是一个连续函数，MAX 是求多个向量每个位置元素最大值的函数。

在实际中，γ 和 h 是通过多层感知器（MLP）实现的。

虽然 PointNet[16] 能够较好的完成语义分割任务，但是其最大池化层只能输出最终的全局特征向量，导致局部结构特征的缺失。PointNet++[8] 通过提出了新的集合提取层（set abstract layer）和特征传播层（feature propagation layer）对此进行了改进：

集合提取层（set abstract layer） 集合提取层（set abstract layer）分为三个部分：采样层、分组层和 PointNet 层。

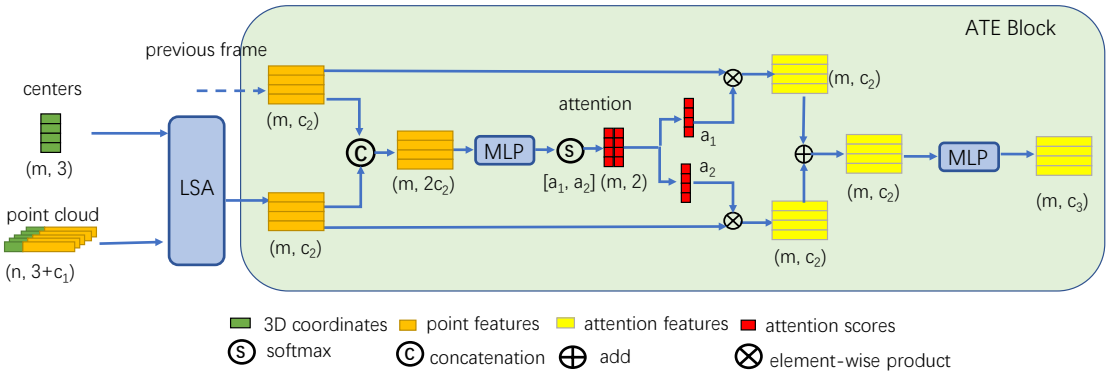


图 3–5 提出的 ASAP 模块。给定当前点云和相应的中心点，我们首先使用 LSA 来计算中心特征。然后，我们使用 ATE 将相邻帧中的中心点特征融合在一起。我们的时空关联策略可确保两组中心点特征之间的对应关系。

采样层　给定输入的点云 $\{x_1, x_2, ..., x_n\}$，他们使用迭代的最远点采样（FPS，Farthest Point Sampling）的方法来选择一个点的子集 $\{x_{i_1}, x_{i_2}, ..., x_{i_m}\}$，即 x_{i_j} 是其余点 $\{x_{i_1}, x_{i_2}, ..., x_{i_{j-1}}\}$ 的最远点。

分组层　设输入的点云的大小为 $N \times (d + C)$，采样得到的中心点坐标的大小为 $N' \times d$，该层按照一定的半径对每个中心点附近的点进行搜索，并输出点云分组 $N' \times K \times (d + C)$，其中 K 是每个中心点分组中邻点的个数。

PointNet 层　对每个分组的元素 $K \times (d + C)$，该层首先将局部坐标转换为相对于中心点的坐标 $x_i^{(j)} = x_i^{(j)} - \hat{x}^{(j)}$，其中 $\hat{x}^{(j)}$ 是中心点的坐标；然后使用 PointNet[16] 的方法对局部特征进行提取。

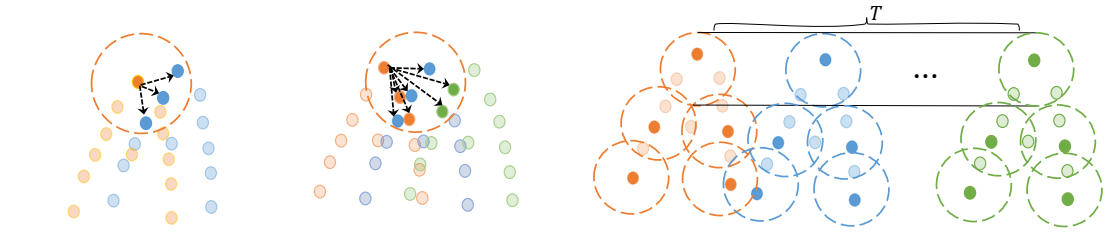
特征传播层（feature propagation layer）　由于语义分割任务需要对每个点进行分类，需要将点数恢复到原始点云中的点数，为此该文章提出了特征传播层（feature propagation layer）。给定大小为 $N_l \times (d + C)$ 的点云和个数为 $N_{(l-1)}$ 的目标点的坐标（其中 $N_l < N_{l-1}$，使用如下公式进行插值：

$$f^{(j)}(x) = \frac{\sum_{i=1}^{k} w_i(x) f_i^{(j)}}{\sum_{i=1}^{k} w_i(x)} \quad where \quad w_i(x) = \frac{1}{d(x, x_i)^p}, j = 1, 2, ..., C \tag{3–21}$$

该文章中默认使用 $p = 2, k = 3$。插值得到的特征向量会与之前集合提取层的特征连接然后再输入到 PointNet 层。

3.2.2　注意力机制的时空感知模块

图 3–5展示了我们的 ASAP（Attention and Structure Aware Point cloud）模块的结构。为了说明我们的模块，我们将长度为 T 的点云序列表示为 $\{S_1, S_2, ..., S_T\}$。点云的每个帧表示为 $S_t = \{p_i^{(t)}\}_{i=1}^{n}$，其中 n 是点数，每个点 $p_i^{(t)}$ 由其欧几里得坐标 $x_i^{(t)} \in \mathbb{R}^3$ 和特征向量 $f_i^{(t)} \in \mathbb{R}^c$（可以是传感器输入的反射强度、距离等或者网络学习的特征）组成。为了将中心点与普通点区分，我们将 S_t 中的一组中心点表示为 $\{c_j^{(t)}\}_{j=1}^{m}$，其中 m 是中心点数，$m < n$，中心点的获取将在下文解释，每个中心点同样包括欧几里得坐标 $x_j^{(t)} \in \mathbb{R}^3$ 和特征向量 $f_j^{(t)} \in \mathbb{R}^c$。



(a) Direct grouping (2-frame) (b) Direct grouping (multi-frame) (c) Spatio-Temporal correlation

图 3–6 所提出的时空关联策略的示意图。与（a）两帧直接分组[63] 和（b）多帧直接分组[10] 不同，在我们的时空关联中，每个中心（(c) 中的实心点）通过（i）每帧最近的中心搜索或（ii）帧间恒定中心点迭代的策略进行配对，然后输入到时间嵌入层，从而在 T 帧之间建立相关性。与（a）和（b）相比，我们的管状的形状能够更好地解耦时空结构。

3.2.2.1 局部结构聚合（LSA）

给定一个点云 $\{p_i^{(t)}\}_{i=1}^{n}$ 和相应的中心点 $\{c_j^{(t)}\}_{j=1}^{m}$，我们按照 PointNet++[8] 的方法提取局部结构特征。对于每个中心点，我们在一个半径内查找其相邻点，记为 $\mathcal{N}(c_j^{(t)})$，并使用下面的对称函数计算其特征：

$$f_j^{(t)} = \underset{p_i^{(t)} \in \mathcal{N}(c_j^{(t)})}{MAX} \{\eta(f_i^{(t)}, x_i^{(t)} - x_j^{(t)})\}. \tag{3–22}$$

其中 η 是一个 MLP（多层感知器），其输入是特征向量 $f_i^{(t)}$ 和空间位置差 $x_i^{(t)} - x_j^{(t)}$ 的串联向量。MAX 是逐元素的最大池化。

3.2.2.2 时空关联（STC）

上文提到，我们需要解决动态点云中的两个主要挑战：(a) 不同帧特征的融合 (b) 跨帧点的关联。在这里，我们首先介绍（b）如何关联不同帧之间的点，并在下一节中介绍（a）。具体来说，给定连续两帧的点云，我们需要获取它们之间的成对的对应关系以利用时间信息。MeteorNet[10] 使用对相邻点的直接分组或基于场景流的分组。前者无法准确了解跨帧不同点之间的对应关系，后者可以准确地知道每个点如何跨帧移动，但计算量很大，不能满足各类任务对处理速度的要求（如自动驾驶）。在下文中，我们提出了两种简单而有效的关联方法：

（i）**每帧最近的中心搜索**。使用最远点采样在每个帧中生成中心点，并分别计算局部特征。对于当前帧中的每个中心，我们基于欧几里得距离将其与上一帧中最近的中心相关联。

（ii）**帧间常量中心迭代**。仅使用最远点采样在第一帧中生成中心坐标，并在后续帧中使用相同的中心坐标，因此中心点自然而然是相对应的。

我们对设计（ii）的原因是，为了有效地捕获时间信息，网络应具有稳定的聚焦区域以实现时间一致性。虽然使用相同的中心坐标似乎并不常见，因为除了第一个帧外，它们不在后续各帧中存在。但是，它们的邻域确实存在。尽管点云是动态的，但它们的边界实际上是稳定的（由 LiDAR 扫描仪或深度相机的固有参数决定的）。因此，实际上不必担心某些中心点会由于点云的运动而偏离点云太远导致不再有相邻的点。策略（ii）使每个中心点在时间上形成一个时空管（请参见图 3–6），通过以迭代地计算中心特征，我们可以捕获时间信息并逐步适应邻域的结构特征变化。

我们采用（ii）作为最终策略。可以看出，（i）需要对每帧进行采样并计算距离，而（ii）则不需要，因此有着更高的计算效率。我们将在 Section 4.4.1中的实验中展示时空相关策略（ii）的优势。

采样计算复杂度讨论 为了说明策略（ii）不进行点云采样对计算效率的提升，我们对各类点云采样方法的计算复杂度进行讨论：

- 最远点采样（FPS） 给定输入的点云 $\{x_1, x_2, ..., x_n\}$，最远点采样（FPS）的方法通过迭代的方式选择点 x_{i_j} 使其为离剩下的点 $\{x_{i_1}, x_{i_2}, ..., x_{i_{j-1}}\}$ 最远的点。尽管该方法采样结果能较好的覆盖整个点云，但是其时间复杂度高达 $\mathcal{O}(N^2)$。
- 逆密度重要性采样（IDIS） 为了从 N 个点选出 K 个点，IDIS 使用与最近 K 个点的距离（K 最近邻算法已计算出）之和来近似逆密度，并对点云中的点进行排序，最终选择最前面的 K 个点。该算法的时间复杂度为 $\mathcal{O}(N)$。相比较于最远点采样（FPS），该算法计算复杂度有所降低，但对异常点较为敏感。
- 随机采样（RS） 对于输入的 N 个点，随机地选取 K 个点。该方法的时间复杂度为 $\mathcal{O}(1)$，即理论上对于任何数量的点云，其运行时间是恒定的。相比较上述最远点采样（FPS）和逆密度重要性采样（IDIS），该方法有着最高的计算效率，但是由于采样的随机性，采样得到的点往往不能很好地反映整个点云的空间分布。
- 生成器采样（GS） GS 通过学习的方式产生点集来近似原始点云，该方法通常需要使用最远点采样（FPS）的方法来与原始点云进行匹配，由此带来了额外的计算量。
- 连续松弛采样（CRS） 这类方法使用参数化的技巧将采样操作松弛到连续域上。每个采样点是由整个点云加权求和得到的，这导致采样过程中需要很大的权重矩阵，带来了极大的内存消耗。
- 策略梯度采样（PGS） PGS 将采样操作看作马尔可夫决策过程。通过序列式地产生一系列概率来对点云进行采样。但是，由于当点云规模较大时的巨大探索空间，产生的概率分部有着极大的方差，如为了从 10^6 个点中采样 10%，探索空间的大小为 $C_{10^6}^{10^5}$，这使得学习有效的采样策略变得极其困难。

Hu 等人[43] 对上述各类采样方法的运行时间和显存占用的测试结果如图 3–7所示。测试过程中，给定一个点云每次采样 25% 的点，重复 5 次，可以看到常用的最远点采样的方法，当点云的规模达到 10^5 的级别（SemanticKITTI[14] 单帧点云的点数约为 1.2e5）时，采样运行时间约为 $1s$ 并且需要占用约 1GB 的显存，这是非常低效的。由于我们的关联策略（ii）省略了第一帧之后的采样过程，可以为网络的运行速度带来较大的提升。

3.2.2.3 Temporal Embedding (TE)

时间嵌入层用于在帧与帧之间融合局部结构特征。该层将当前帧的中心点 $\{c_j^{(t)}\}_{j=1}^m$ 和经时空关联策略匹配的前一帧地中心点 $\{c_j^{(t-1)}\}_{j=1}^m$ 作为输入，将其融合并更新当前帧每个中心点的特征，我们将其表示为 $h(c_j^{(t)})$。我们也提出了两种时间嵌入方法：

（i）**直接时间嵌入（DTE）**：我们直接串联匹配的两帧中心点特征，并使用共享的多层感知器（MLP）ζ 来更新当前中心点的特征：

$$h(c_j^{(t)}) = \zeta(f_j^{(t-1)}, f_j^{(t)}). \tag{3–23}$$

（ii）**注意力机制的时间嵌入（ATE）**：我们首先将两组关联的中心点特征输入到共享的

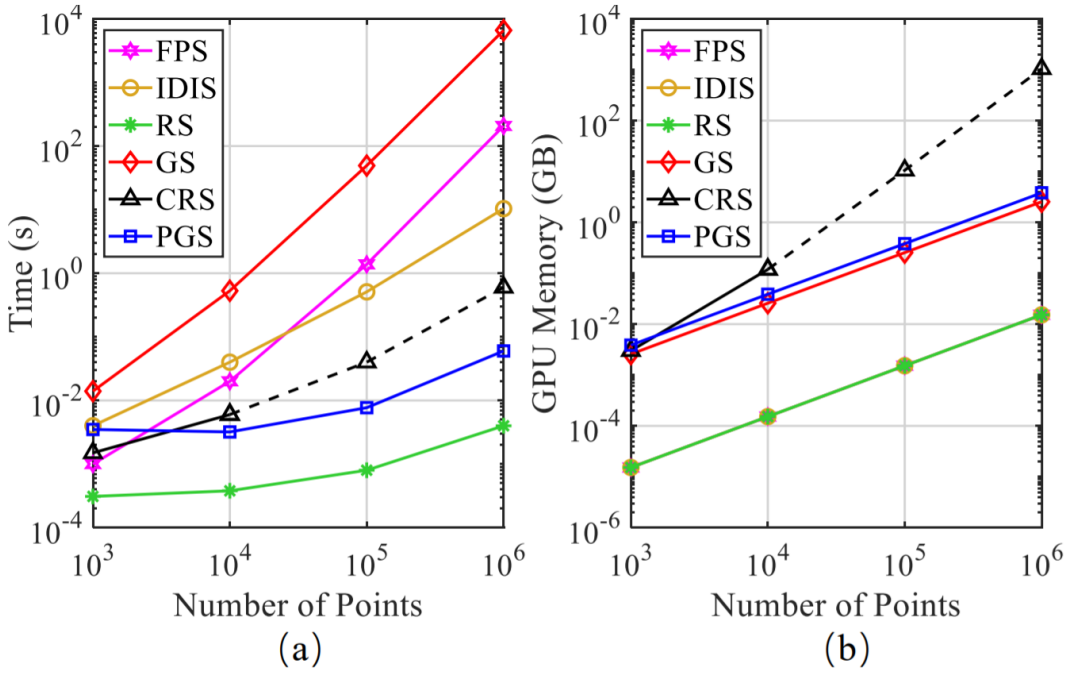


图 3–7 各采样算法运行时间和显存的占用，其中虚线部分由于显存大小的限制，为预测值。

多层感知器（MLP）γ 和 $Softmax$ 函数来计算两个标量注意力权重：

$$[a_1, a_2] = Softmax(\gamma(f_j^{(t-1)}, f_j^{(t)})). \tag{3–24}$$

然后，我们通过将两个注意力与相应特征相乘来计算加权的融合特征：

$$f'^{(t)}_j = a_1 f_j^{(t-1)} + a_2 f_j^{(t)}. \tag{3–25}$$

最后，我们使用另一个多层感知器（MLP）ζ 来更新当前中心点的特征：

$$h(c_j^{(t)}) = \zeta(f'^{(t)}_j). \tag{3–26}$$

我们设计 ATE 模块的原因是不同的帧可能对结果有不同的贡献。对于缓慢移动的物体，由于迭代计算的方式，前一帧中包含多帧信息，因此上一帧的特征可能更可靠，而对于快速移动的对象，最好更多地依赖当前帧的特征。此外，每帧中都可能出现不希望有的错误或噪音。如果网络可以自己估计来自不同帧的特征的重要性和可靠性，最后往往能做出更好，更可靠的预测。相比于 DTE，ATE 甚至可以使用更少的参数来获得更好的结果，这将在 4.3中显示。

LSA 和 TE 可以按层级的方式堆叠。具体来说，我们将使用 DTE 的模块称为 SAP-x 的模块（不含注意力机制），将使用 ATE 的模块称为 ASAP-x 的模块（包含注意机制），其中 x 是 LSA 和 TE 的层数。

讨论 在章节 3.1.1提到，半耦合的网络结构可以抽象为：

$$\mathcal{F}[h_s(\mathbf{x};\psi_s), h_t(\mathbf{x};\psi_t)] \tag{3–27}$$

其中，h_t 和 h_s 分别为时间单元和空间单元，$\mathcal{F}$ 是融合函数。

在我们的 ASAP 模块中，h_s 是由基干网络实现的，如 PointNet++[8] 和 SqueezeSegV2[9] 等。公式 3–24实现了时间模块 h_t 的功能，而公式 3–25和公式 3–26则实现了融合函数 $\mathcal{F}$ 的功能。

3.2.3 总体结构

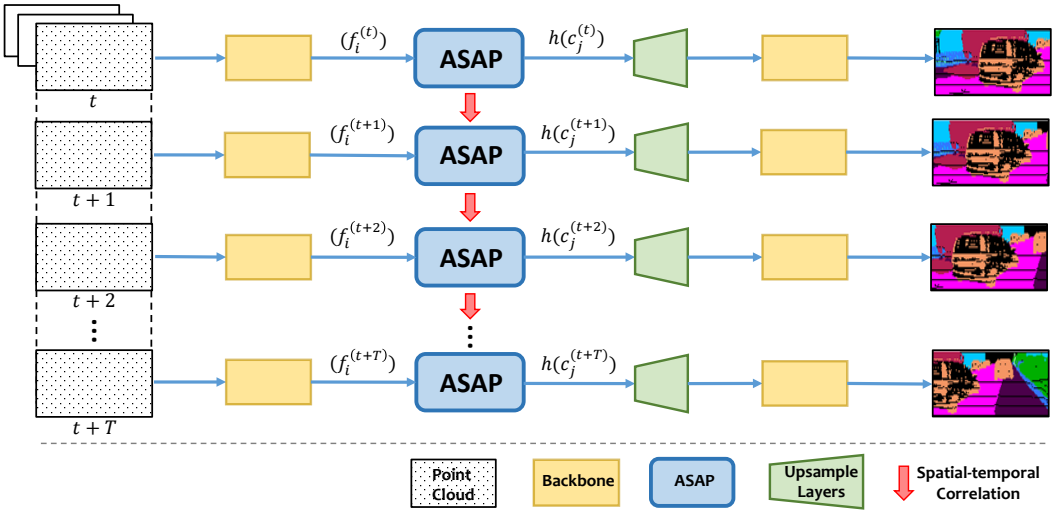


图 3–8 整体网络结构。我们的框架包括基干网络，提出的 ASAP 模块和上采样层。连续两帧之间的关联性是由我们提出的时空关联策略保证的。

总体结构显示在图 3–8中。我们首先使用[8, 9] 之类的基干网络来提取每个帧中的点特征。然后，我们的 ASAP 模块将计算得到中心特征以迭代的方式进行跨帧融合。接下来，我们使用[8] 提出的特征传播（*feature propagation*）层将点云上采样到原始大小，然后将其返回到骨干网络。最后，骨干网将继续运行并返回预测。

根据我们的时空相关性策略，对于每个序列，我们在第一帧中使用最远点采样生成中心点，并在后续帧中使用相同的坐标以实现时间一致性并减少计算量。

第四章　实验与分析

4.1　数据集及实验设定

4.1.1　时空感知数据集

我们在两个任务上对时空感知进行研究。首先，我们在常见的图像序列任务动作识别上对模型进行验证；其次，我们在当前非常热门的自动驾驶任务进行实验。完成这两个任务都需要模型能对时间特征和空间特征有很好的学习能力。

动作识别　动作识别实验是在 UCF-101[1]，HMDB-51[2] 和 Kinetics-400[3] 数据集上进行的，这些数据集分别包含 101、51、400 个动作类别。对于每个数据集，我们使用官方的训练集和测试集划分。对于每个视频，我们将每一帧的图像等比例缩小，处理后较短的边长为 368，然后从每一帧或其水平翻转中随机采样 224 x 224 的区域。在训练过程中，我们使用了颜色增强（color augmentation），其中所有随机增强参数在视频的每一帧中保持相同。

驾驶行为　自动驾驶是一项复杂的任务，彻底实现需要解决场景感知，路线规划，安全保证等任务。在这里，我们将其简化为序列图像处理的视觉任务：从驾驶员的角度给出简短的视频，并以方向盘角度的形式输出行驶方向。我们在 comma.ai[4] 和 LiVi-Set[5] 数据集上进行实验。这些数据集记录了驾驶员的真实驾驶行为，并且包含不同的道路条件，如城镇街道，高速公路和山区道路等。为了使模型更好地专注于道路，我们去掉了原始图像中天空等其他不相关的信息。最终输入图像的大小为 192×64。

4.1.2　动态点云分割数据集

我们在两个数据集上进行语义分割的实验。我们首先在大型模拟数据集 Synthia[23] 上测试我们的方法，然后与 MeteorNet[10] 和 4D MinkNet[11] 基线进行比较，并进行拆解实验研究。然后，我们在目前公开可使用的最大的实际 LiDAR 数据集 SemanticKITTI[14] 上进行实验，以展示我们模块的有效性和泛化能力。

Synthia　由九个不同天气条件下的六个驾驶场景序列组成。每个序列由从行驶中的汽车顶部的四个视角捕获的 RGBD 图像所组成。我们从 RGBD 图像重建 3D 点云，使用与 MeteorNet[10] 相同的方式创建点云序列。我们使用与[10, 11] 相同的训练/验证/测试集划分，将除了日落，春季和大雾之外的天气条件的序列 1-4 用作训练集；将大雾天气的序列 5 用作验证集；将包含日落和春季天气的序列 6 作为测试集。训练，验证和测试集分别包含 19,888、815 和 1,886 帧的点云数据。

SemanticKITTI　此数据集是在 KITTI 视觉数据集[73] 的里程表数据集的基础上创建的。驾驶场景包括德国中心市区、住宅区、高速公路和乡村道路，各种场景都提供了带标注的点云序列。该数据集提供了 23,201 个完整 3D 点云扫描用于训练，20351 个用于测试，这使它成为公开可用的最大数据集。我们使用与[14] 相同的数据集划分用于训练、验证和测试。为了与基干网络进行比较，我们遵循[14] 中提出的单扫描实验，并对 19 个类别进行了预测。由于

我们的模型将序列作为输入，因此我们将测试集分成固定长度的序列，并保存每帧扫描的语义分割结果。

4.2 时空感知实验结果

4.2.1 动作分类实验

我们采用两种半耦合结构：有基干网络做基础的结构和无基干的完全独立结构。对于基干网络做基础的结构，在 15 层 SCS 网络之前，我们使用在 ImageNet[74] 预训练的一个卷积网络（CNN）做基干（我们选择了 VGG 和 InceptionV1）。对于独立版，该模型仅包含一个 17 层 SCS 网络。SCS 网络中采用了 ResNet[75] 提出的层与层之间互相连接的结构，从而使深度网络更易于训练。

网络的主要目标 g 是使 softmax 输出各动作类别置信度与事实（groundtruth）的交叉熵最小化；最终输出是每个时间戳输出结果的平均值。空间单元的目标 r_s 与主要目标相同，时间单元的目标 r_t 是估计当前输入帧与上一帧之间的光流。每一步，网络都会处理一个新的视频帧，并根据当前处理的帧来预测动作类别上的概率分布。

我们使用高级 STSGD 的训练策略，表格 4–1列出了 SCS，LSTM 和纯 CNN 模型的动作识别实验的完整结果和超参数。我们可以看到，我们的 SCS 比 LSTM，ConvLSTM 和 CBM[76] 模型具有更好的性能。与 CBM 相比，新的 SCS 在学习过程中将空间和时间信息解耦，并在训练过程中有策略地调整其关注点（空间或时间信息），由此带来了效果的提升。

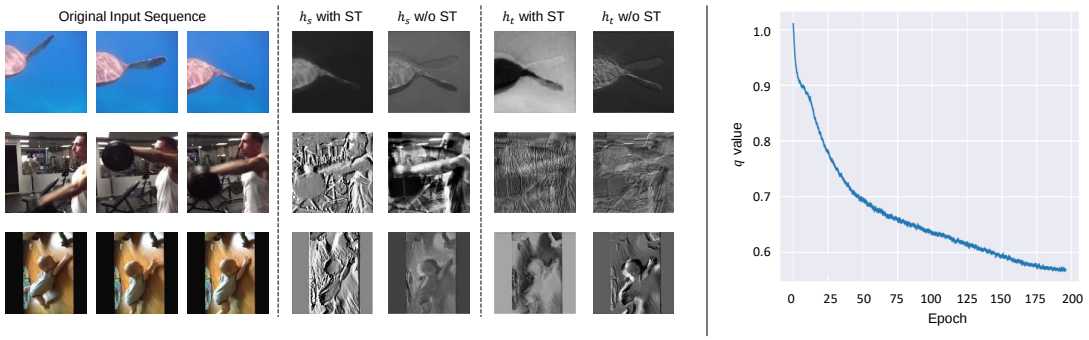
Architecture	Kinetics	UCF-101	HMDB-51
		Pre-trained on Kinetics	
LSTM with BB (VGG)[77]	53.9	86.8	49.7
3D-Fused[78]	62.3	91.5	66.5
Stand-alone CBM[76]	60.2	91.9	61.7
Stand-alone SCS	61.7	92.6	65.0
		Not pre-trained on Kinetics	
15-layer ConvLSTM	-	68.9	34.2
BB (VGG) supported CBM[76]	-	79.8	40.2
BB (VGG) supported SCS	-	82.1	42.5
BB (Inception) supported SCS	-	87.9	52.1

表 4–1 在 Kinetics[3]，UCF-101[1] 和 HMDB-51[2] 数据集上的动作识别结果. UCF-101[1] 和 HMDB-51[2] 是使用在 Kinetics[3] 进行预训练的模型进一步训练得到的。我们的 SCS 模型为 17 层. “BB” 代表基干网络。

4.2.2 驾驶行为预测实验

我们将 SCS 网络与常规 LSTM 模型和 CNN 模型进行了比较。SCS 结构与动作识别中使用的独立结构的模型相同，LSTM 模型采用 CNN 基干网络 VGG，如 LRCNs[77]。这两个模型均以短视频作为输入，并提取时空特征。对于 CNN 模型，我们采用 ResNet[75] 结构，它仅以当前行驶图像为输入，并利用空间特征进行预测。

这是一个回归问题，网络的主要目标 g 为最小化预测转向角和事实（ground truth）之间



a Input sequences and corresponding feature maps of h_s and h_t when adopting sub-tasks (ST) or not.

b Change trend of q value in STSGD during training SCS.

图 4–1 **a**：h_s 和 h_t 的特征图。可以看到，空间单元和时间单元各自有自己的关注点，空间单元更多地提取空间特征而时间单元则更多地提取时间特征。当引入子任务（ST，即 Sub-Task）后，这种分离变得更加明显。**b**：训练过程中公式 3–19中的 q 值。在训练开始时，q 约为 1，这导致 SCS 首先关注空间信息。随着训练的进行，q 的值逐渐接近 0.5 以合并时间和空间信息，并且模型将它们同等对待。

的均方误差损失（MSE loss，即 Mean Square Error loss）。与动作识别任务相同，空间目标 r_s 与主要目标相同，时间目标 r_t 用于估计光流。由于方向的角度相对集中分布在 0 附近，我们采用 Sigmoid 函数对角度进行归一化，该非线性函数在某种程度上可以使角度的分布更均匀。我们使用准确度用作为度量标准，其定义为：

$$Acc = \frac{\sum_i^I [\min(\frac{\lambda}{|\text{pred}_i - \text{label}_i| + \epsilon}, 1)]}{I} \tag{4–1}$$

其中 I 是样本数，λ 是阈值，而 ϵ 是用于防止分母为零的一个很小的数值。pred_i 和 label_i 是样本 i 的预测角度值和标签角度值。简而言之，如果预测角度和标签角度之间的差异小于阈值，我们将其视为准确的预测。

为了预测当前的行驶方向，模型的输入时驾驶视频的之前几帧（CNN 模型除外）。对于深层 SCS 网络，我们采用 STSGD 的策略进行训练。我们发现，STSGD 的训练策略使模型的平均性能提高了 9%。详细的比较结果见表格 4–2。

4.3 动态点云分割实验结果

4.3.1 基干网络的选取

我们提出的 ASAP 模块是一种通用体系结构，可以灵活地合并到不同的基干网络中。我们将整个网络称为 ASAP-Net（骨干 + ASAP 模块）。

我们首先选择 PointNet++[8] 作为我们的基干网络（因为我们的 ASAP 模块以相同的方式处理点云）。为了展示我们的 ASAP 模块的灵活性，我们进一步使用另一种基干网络 SqueezeSegV2[9] 进行实验。网络首先将 LiDAR 点云投影到一个球体上，以进行基于网格的

SCS model							
		length=7			length=3		
		λ=6	λ=3	MSE	λ=6	λ=3	MSE
LiVi	p=0.0	31.8	16.9	0.046	28.8	15.9	0.049
	p=0.3	34.1	17.4	0.045	30.5	16.1	0.048
	p=0.5	**35.1**	**19.4**	**0.044**	**33.4**	**17.6**	**0.046**
Comma	p=0.0	45.4	24.5	0.060	42.5	22.9	0.05
	p=0.3	48.8	**25.5**	0.043	46.9	23.9	0.044
	p=0.5	**49.2**	25.0	**0.037**	**47.4**	**24.1**	**0.041**
CNN+LSTM							
		length=7			length=3		
		λ=6	λ=3	MSE	λ=6	λ=3	MSE
LiVi		29.2	15.9	0.052	27.3	14.5	0.057
Comma		43.1	23.8	0.056	42.3	21.0	0.058
CNN							
		length=1					
		λ=6	λ=3	MSE			
LiVi		24.5	13.0	0.057			
Comma		45.2	25.3	0.056			

表 4–2 驾驶行为预测实验中 SCS 和对照方法 (CNN, CNN+LSTM) 的结果。包含在 comma.ai[4] 和 LiVi-Set[5] 数据集上的实验。其中 λ" 代表角度的阈值，p 代表 STSGD 中切断反传路径初始的的概率，length 是输入序列的长度.

密集表示，如下所示：

$$\begin{aligned} \theta &= \arcsin \frac{z}{\sqrt{x^2+y^2+z^2}}, \tilde{\theta} = \lfloor \theta/\Delta\theta \rfloor, \\ \phi &= \arcsin \frac{y}{\sqrt{x^2+y^2}}, \tilde{\phi} = \lfloor \phi/\Delta\theta \rfloor. \end{aligned} \tag{4–2}$$

这里 $\Delta\theta$ 和 $\Delta\phi$ 是离散化的分辨率，而 $(\tilde{\theta}, \tilde{\phi})$ 表示点在 2D 球形网格上的位置。在每个点云上应用这些方程后，我们可以获得大小为 $H \times W \times C$ 的 3D 张量。然后，他们使用 SqueezeNet[20] 中提出的名为 *fireModule* 和 *fireDeconvs* 的 CNN 模块构建了编码器-解码器（encoder-decoder）的结构。

要从 2D 球面网格重建点云，我们只需将中间层的特征图从 $H \times W \times C$ 更改为 $N \times C$，其中 $N = H \times W$ 是数字点数，C 是特征长度。我们在原始投影球面网格（通道包含坐标）上使用最大池化层，以大致获得相应的 3D 坐标。

请注意，尽管如 SemanticKITTI[14] 中所述，DarkNet53Seg 在 SemanticKITTI[14] 上达到了最佳的 mIoU 结果，但这不是已发表的著作。根据作者的说法，它与 SqueezeSegV2[9] 的是一类网络，并且性能高只是因为它具有更多的层。因此，对 SqueezeSegV2[9] 的改进应保证对 DarkNet53Seg 的提升。

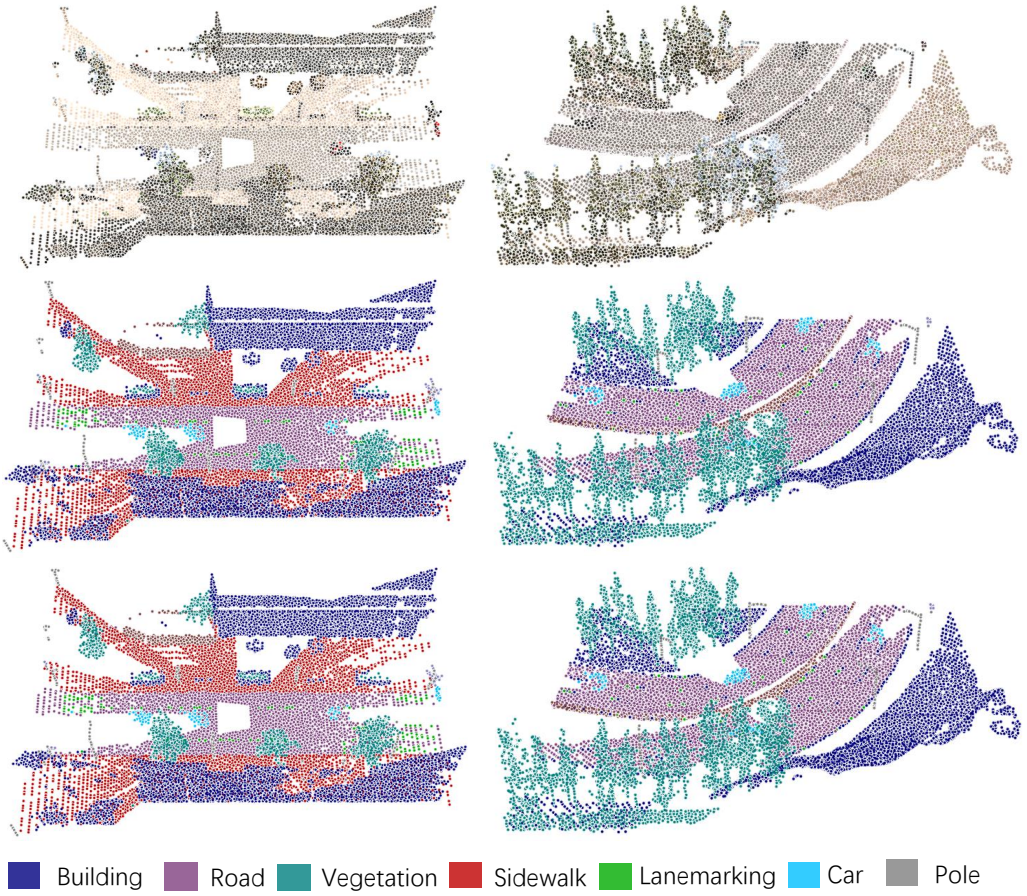


图 4–2 来自 Synthia 数据集的两个可视化结果。从上到下依次是：RGB 输入，实际的类别（groundtruth），预测的结果。

4.3.2 Synthia 的实验结果

结果列在表格 4–3中，我们的 ASAP-Net 超过了 4D MinkNet[11] 和 MeteorNet[10] 的结果，并建立了目前最好的 mIoU（mean Intersection over Union，交集比并集的平均值）结果。图 4–2显示了数据集中的两个样本和预测结果，可以看到我们的模型可以准确地分割大多数类别的物体。

我们可以看到，与 SAP 模块相比，我们的 ASAP 模块可以用更少的参数获得更好的结果。参数的减少有两个方面的原因：首先，因为输出注意力长度仅为 2，用于计算注意力的额外参数 γ 数量确实很少。其次，将两个相邻帧中的特征进行融合时是通过加法运算而不是串联运算，因此 ζ 的输入特征长度是 SAP 中的一半。这两方面原因使得 ASAP 模块的参数量得到了降低。

4.3.3 SemanticKITTI 的实验结果

表格 4–4中显示了基干网的 IoU 结果以及我们的 ASAP-Net 结果。当与我们的 ASAP 模块结合使用时，两个基干网络的语义分割结果都得到了显著的提升。同时，在图 4–3中可以找到一些定量的可视化结果。

Approach	param (M)	frame num	mIoU	mAcc	IoU Bldg	Road	Sdwlk	Fence	Vegitn	Pole	Car	T.Sign	Pdstr	Bicyc	Lane	T.light
3D MinkNet[11]	19.31	1	76.24	**89.31**	89.39	97.68	69.43	86.52	98.11	97.26	93.50	79.45	92.27	0.00	44.61	66.69
4D MinkNet[11]	23.72	3	77.46	88.01	90.13	**98.26**	73.47	87.19	**99.10**	97.50	94.01	79.04	**92.62**	0.00	50.01	68.14
PointNet++[8]	0.88	1	79.35	85.43	96.88	97.72	86.20	92.75	97.12	97.09	90.85	66.87	78.64	0.00	72.93	75.17
MeteorNet[10]	1.78	3	81.80	86.78	**98.10**	97.72	88.65	94.00	97.98	**97.65**	93.83	**84.07**	80.90	0.00	71.14	77.60
PNv2 with SAP-1	1.97	3	82.31	86.72	97.68	97.99	**90.16**	94.84	97.25	97.34	94.77	80.35	83.54	0.00	**75.13**	**78.62**
PNv2 with ASAP-1	1.84	3	**82.73**	87.02	97.67	98.15	89.85	**95.50**	97.12	97.59	**94.90**	80.97	86.08	0.00	74.66	77.51

表 4–3 在 Synthia 数据集[23] 上的语义分割结果。评估指标是平均 IoU（Intersection over Union）和平均准确性（%）。PNv2 代表 PointNet++[8]。SAP-x 和 ASAP-x 的含义在章节 3.2.2.3中定义。

Approach	mIoU	car	bicycle	motorcycle	truck	other-vehicle	person	bicyclist	motorcyclist	road	parking	sidewalk	other-ground	building	fence	vegetation	trunk	terrain	pole	traffic-sign
PointNet[16]	14.6	46.3	1.3	0.3	0.1	0.8	0.2	0.2	0.0	61.6	15.8	35.7	1.4	41.4	12.9	31.0	4.6	17.6	2.4	3.7
SPGraph[59]	17.4	49.3	0.2	0.2	0.1	0.8	0.3	2.7	0.1	45.0	0.6	39.1	0.6	64.3	20.8	48.9	27.2	24.6	15.9	0.8
SPLATNet[34]	18.4	58.2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	64.6	0.4	39.1	0.0	58.3	23.1	71.1	9.9	19.3	5.6	0.0
PointNet++[8]	20.1	53.7	1.9	0.2	0.9	0.2	0.9	1.0	0.0	72.0	18.7	41.8	5.6	62.3	16.9	46.5	13.8	20.0	6.0	8.9
PNv2 with SAP-1	31.4	79.7	9.5	5.5	0.1	8.1	7.9	22.3	1.1	81.2	34.2	56.3	7.9	75.7	38.5	58.8	26.8	50.1	16.1	16.9
PNv2 with ASAP-1	33.3	84.1	11.6	7.5	3.2	11.4	7.8	18.5	3.0	81.8	28.1	53.1	7.8	74.9	37.6	64.4	27.2	51.7	22.8	30.8
PNv2 with ASAP-2	35.3	86.4	9.3	6.4	8.1	13.0	12.8	25.2	3.8	80.3	29.7	57.6	13.2	77.7	40.1	66.7	31.9	52.9	26.7	28.5
SqueezeSeg[19]	29.5	68.8	16.0	4.1	3.3	3.6	12.9	13.1	0.9	85.4	26.9	54.3	4.5	57.4	29.0	60.0	24.3	53.7	17.5	24.5
SqueezeSegV2[9]	39.7	81.8	18.5	17.9	13.4	14.0	20.1	25.1	3.9	88.6	45.8	67.6	17.7	73.7	41.1	71.8	35.8	60.2	20.2	36.3
TangentConv[79]	40.9	**90.8**	2.7	16.5	15.2	12.1	23.0	28.4	**8.1**	83.9	33.4	63.9	15.4	**83.4**	**49.0**	**79.5**	**49.3**	58.1	**35.8**	28.5
SSv2 with SAP-1	40.7	83.0	21.1	**22.4**	4.1	15.5	25.7	**33.9**	1.3	89.5	52.9	68.9	19.2	73.7	40.1	69.2	36.1	61.4	14.0	**40.9**
SSv2 with ASAP-1	41.3	83.3	19.8	21.1	9.2	15.5	24.4	31.3	2.5	89.7	53.3	69.0	18.1	79.3	45.0	71.7	38.3	61.6	14.0	37.1
SSv2 ASAP-1-msg	42.5	84.0	19.8	22.3	13.1	17.2	24.0	31.9	5.5	90.4	**55.5**	**70.9**	**22.2**	80.5	46.2	71.5	37.0	63.1	15.3	36.5
SSv2 with ASAP-2	**43.1**	83.1	**22.5**	20.8	**16.0**	**18.1**	**26.6**	32.8	4.6	**90.5**	55.2	69.8	20.4	81.7	**49.0**	72.0	38.3	**63.2**	16.4	38.4

表 4–4 SemanticKITTI 数据集[14] 上的基干网和基干网络结合 ASAP 模块的语义分割结果。指标是 mIoU（mean Intersection over Union）。PNv2 代表 PointNet++[8]，SSv2 代表 SqueezeSegV2[9]。SAP-x 和 ASAP-x 的含义在章节 3.2.2.3中定义。所有的方法都在序列 00 至 10（除了将序列 08 用作验证集）进行训练。我们还列出了其他已发表工作的结果，以使读者对各类方法在该数据集上的表现有一个总体的了解。

4.4 模型分析

4.4.1 时空关联策略分析

在本节中，我们对两种时空关联策略和输入序列长度进行实验分析。不失一般性，我们基干网络使用 PointNet++[8]，ASAP 模块只使用一层的 ASAP-1。

结果显示在表格 4–5中。如我们所见，STC（i）效果较差。我们认为这是因为网络的聚焦区域（中心点半径内的区域）在不同的帧中不断变化，从而难以实现时间一致性。我们还可以看到，随着序列长度的增加，结果得到了改善，这表明了我们的 ASAP 模块利用时间信息的有效性，也印证了我们时空关联策略（ii）的合理性。

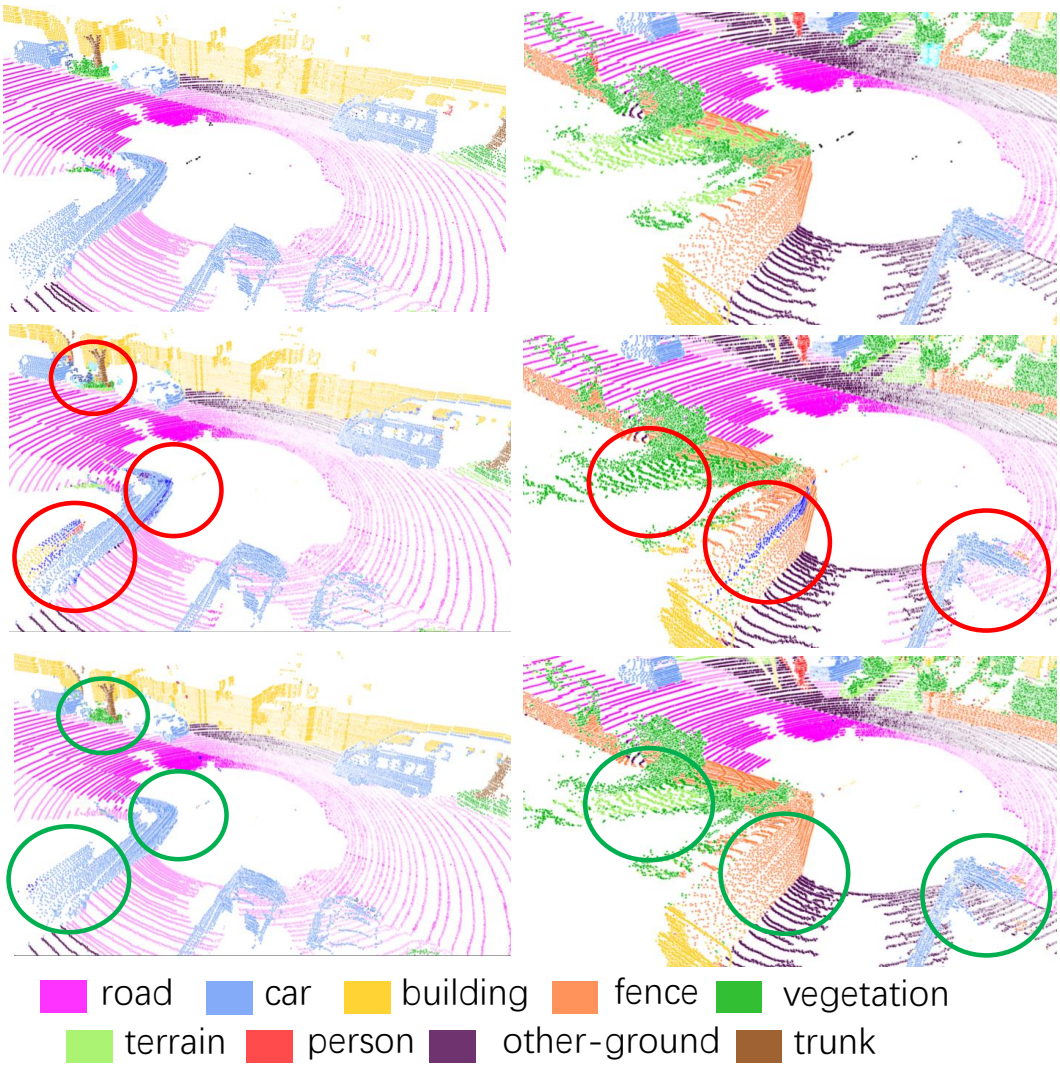


图 4–3 在 SemanticKITTI[14] 上进行语义分割实验的结果。从上到下依次是：实际的语义分割类别（groundtruth），在合并我们的 ASAP 模块之前和之后的 SqueezeSegV2[9] 的结果。红色的圆圈突出显示 SqueezeSegV2[9] 的错误的预测结果，而绿色的圆圈突出显示了结合了我们 ASAP 模块之后正确的预测结果。可以看到，结合了我们的 ASAP 模块后，基干网络的语义分割效果得到了提高。

4.4.2 对不同基干网络的提升效果

正如我们在 Table 4–4中所看到的，PointNet++[8] 的效果提升远远大于 SqueezeSegV2[9] 的效果提升。我们认为这是因为我们的 ASAP 模块架构与 PointNet++[8] 类似，但与 SqueezeSegV2[9] 完全不同。由于 SqueezeSegV2[9] 将点云转换为 2D 图像，要将其与我们的 ASAP 模块结合，我们必须首先从图像中重建点云，然后将点云投影回 2D。该过程会伴随着信息的丢失。例如，根据章节 4.3.1，我们使用最大池化层来重构点云，以输出点的坐标，但每个输出的坐标可以来自不同点。换句话说，这些输出的点可能根本不存在于原始点云中，从而在某种程度上降低了整个神经网络的语义分割性能。

Approach	param (M)	frame num	mIoU	mAcc	IoU											
					Bldg	Road	Sdwlk	Fence	Vegitn	Pole	Car	T.Sign	Pdstr	Bicyc	Lane	T.light
PointNet++[8]	0.88	1	79.35	85.43	96.88	97.72	86.20	92.75	97.12	97.09	90.85	66.87	78.64	0.00	72.93	75.17
PNv2 with STC (i)	1.84	3	81.78	86.47	97.18	97.73	89.91	94.46	96.61	97.01	94.25	78.28	83.74	0.00	74.69	**77.69**
PNv2 with STC (ii)	1.84	3	82.73	87.02	97.67	98.15	89.85	95.50	97.12	97.59	94.90	80.97	86.08	0.00	74.66	77.51
PNv2 with STC (ii)	1.84	4	82.82	86.71	**98.01**	98.33	92.14	**95.54**	99.12	**97.69**	95.65	81.62	84.84	0.00	74.91	76.04
PNv2 with STC (ii)	1.84	5	82.90	**87.14**	97.90	**98.36**	92.05	95.43	**99.16**	97.51	95.21	**82.27**	84.03	0.00	75.69	77.15
PNv2 with STC (ii)	1.84	6	**83.00**	87.00	97.63	98.34	**92.49**	94.85	97.38	97.53	**95.76**	80.65	**87.48**	0.00	**77.07**	76.81

表 4–5 在 Synthia 数据集[23] 上的语义分割结果。评估指标是 mIoU（mean Intersection over Union）和平均准确性（%）。PNv2 代表 PointNet++[8]。

	动作识别		驾驶行为预测	
	BB supported	Stand-alone	CNN+LSTM	SCS
批大小（Batch size）	16	40	128	128
学习率	1e-4	1e-4	2e-4	1e-4
基干网络	{VGG, Inception}	-	ResNet-18[75]	-
网络层数	BB layers + 15	17	18 + 1	15
训练方法	ASTSGD	ASTSGD	-	STSGD
λ	-	-	{3, 6}	{3,6}

表 4–6 时空感知实验的超参数。

4.4.3 多尺度（multi-scale）特征提取

正如我们在表格 4–4中看到的，在将基干 SqueezeSegV2[9] 与 ASAP-1 模块合并之后，卡车和杆两个类别的 IoU（Intersection over Union）结果有所下降。我们推测这是因为 ASAP-1 的查询半径相对较大（1 米）导致没有足够的精细结构信息。为了验证这一点，我们在 LSA 块中使用[8] 提出的多尺度（multi-scale）分组来提取中心点特征。如 Table 4–4所示，两个类的结果都得到了改善。

4.5 训练超参数

4.5.1 时空感知实验的超参数

时空感知实验包括两个子实验：动作识别和驾驶行为预测。其超参数如表 4–6所示。所有卷积层初始化都使用 Xavier[80] 方法，并且每个卷积层后面都有 Batch Normalization[81] 层。所有的网络都是使用 Adam[82] 优化器进行训练并且基干网络在 ImageNet 上进行了预训练。在沿时间向后反传（BPTT，Back Propagation Through Time）[71] 的过程中，循环神经网络（RNN）的参数的梯度被限制在 $[-5, 5]$ 之间。

4.5.2 动态点云分割实验的超参数

Synthia 实验 Synthia[23] 数据集实验的超参数设置如表 4–7所示。在实现我们的 ASAP 模块时，我们使用 1×1 的卷积来实现多层感知器（MLP），初始化使用 He[83] 方法，并且每层之后都有 Batch Normalization[81] 层。我们使用 Adam[82] 作为优化器对神经网络进行训练。

帧数	3	4	5	6
批大小（Batch size）	5	3	3	2
学习率	1.6e-3	1.6e-3	1.6e-3	1.6e-3
点数	16384	16384	16384	16384
训练方法	ASTSGD	ASTSGD	ASTSGD	ASTSGD

表 4–7 Synthia[23] 数据集实验的超参数。

	PointNet++[8]		SqueezeSegV2[9]	
	ASAP-1	ASAP-2	ASAP-1	ASAP-2
批大小（Batch size）	4	4	4	4
学习率	1.2e-2	1.2e-2	4e-3	4e-3
点数	45000	44000	原始点数约 1.2e5	原始点数约 1.2e5
帧数	4	4	4	4
基干 +ASAP 层数（仅 encoder）	BB 4 + 1	4 + 2	10 + 1	10+2
训练方法	ASTSGD	ASTSGD	ASTSGD	ASTSGD

表 4–8 SemanticKITTI[14] 数据集实验的超参数。

SemanticKITTI 实验超参数 SemanticKITTI[14] 数据集实验的超参数设置如表 4–8所示。在实现我们的 ASAP 模块时，我们使用 1×1 的卷积来实现多层感知器（MLP），初始化使用“He”[83] 方法，并且每层之后都有 Batch Normalization[81] 层。对于使用 PointNet++[8] 作为基干的网络，我们使用 Adam[82] 优化器对其进行训练，而对于使用 SqueezeSegV2[9] 作为基干的网络，我们使用 SGD[84] 优化器对其进行优化。

第五章　总结和展望

5.1　总结

随着激光雷达（LiDAR）和深度相机等技术的发展，点云序列数据变得日益丰富，这为动态点云的相关研究提供了基本条件。但是，尽管静态点云的处理取得了很大的成就[8, 9, 16]，动态点云的处理仍然停留在初级阶段，点云数据在自动驾驶[66]、机器人导航[67] 和增强现实（AR）[68] 等应用中的重要作用以及现实环境的动态性使得动态点云的研究变得尤其重要，由于语义分割对场景理解的重要性，本文对动态点云的语义分割进行了研究。

静态点云处理的方法基本上可以分为两类：基于投影的方法和基于点的方法。对于基于投影的方法，它们通常首先将不规则点转换为规则的表示形式。例如，SEGCloud[17] 首先将点云转换为规则的体素，然后应用 3D 卷积获得粗略的体素语义预测结果。然后，他们使用三线性插值法将体素预测投影到单点，并使用全连接的条件随机场（FCCRF）来增强预测结果的空间一致性。此外，根据 LiDAR 扫描仪的工作机制，SqueezeSeg[19] 根据方位角 *azimuth* 和天顶角 *zenith* 将点云投影到球形表面上。然后，它应用传统的编码器-解码器体系结构在 2D 投影表面上进行语义分割。对于基于点的方法，先驱工作 PointNet[16] 展示了其直接从原始点云中学习逐点特征的能力。但是，单个点只有在与相邻点一起考虑时才具有语义。静态点云上的后续工作[8, 40] 对一组相邻点的局部结构进行编码，并捕获局部结构之间的相互作用。但是所有这些方法仅限于单帧点云，无法同时学习时间和空间信息。

对于动态点云序列的特征学习，一种直观而直接的方法是在时空邻域内在空间和时间上对局部结构进行编码，例如，最新作品[10, 11] 都遵循这个方向。4D MinkNet[11] 是基于投影的，它首先将点云转换为规则体素，然后沿空间和时间维度应用广义的稀疏卷积。它提出了一个新的稀疏卷积库来解决点云稀疏性所带来的计算成本和内存占用问题。为了更好地学习时空特征，该文还提出了一种新的卷积核形状和时空条件随机场。相比之下，MeteorNet[10] 属于基于点的方法，它直接堆叠多帧点云，并通过根据特定半径对时空相邻点进行分组来计算局部特征。

尽管上述两种用于动态点云语义分割的方法都能够捕获时空特征，但是它们很难将序列中的时空信息解耦。有证据表明，在现实世界中，人类视觉系统主要依赖于时间信息[12]，并将其视为主要的信息来源。如果不将空间和时间结构解耦，就无法很好地理解时间信息可以在多大程度上有助于动态场景理解。关于人脑的一些研究还表明，时空信息是通过两条流传递到海马体的[15]，但又彼此深深地耦合在一起[22]。因此，如果能够在一定程度上将信息的两个方面解耦，将促进相关研究的发展。

从大脑的这种机制得到启发，我们提出了一种半耦合网络设计和训练策略，以类似的方式处理这两方面的信息。具体来说，我们首先将网络分为两个支路，来相对独立地处理信息的两个方面。然后，为了让它们彼此交互，我们将两个支路堆叠多层，并将这两路支路在某些层融合在一起。但是，随着网络的加深，计算复杂度变得越来越高，并且在向后传播梯度时它们之间的相互干扰问题变得非常严重。为了解决以上两个问题，我们提出了一种新的基于梯度的学习策略 STSGD，它根据一定的概率切断了一些反向传播路径，从而降低了计算复杂度和两个支路之间的相互干扰。我们进一步将其扩展到高级版本 ASTSGD，以自动调整训练过程中的概率。此外，空间和时间信息的解耦可以使模型更加可扩展和灵活，例

如，我们可以轻松地将静态点云方法用作骨干网络，这可以在章节3.2.2和章节 4.3.1中看到。

虽然我们提出的时空感知网络设计和学习策略是普适性，但对于动态点云的语义分割，仍需要对点云序列的特点进行相应的调整和设计。具体而言，我们使用别人的静态点云处理的网络作为我们的空间支路，并根据点云序列自身的特点提出了名为 ASAP 的模块来作为时间分路。设计该模块主要面临的挑战有：

（a）不同帧特征的融合：为了有效学习时间信息，神经网络需要对来自不同帧的特征进行融合，但来自不同帧的特征可能重要性不同（如快速运动的物体应该更多的依赖当前帧的特征而缓慢运动的物体应该同时考虑当前的几帧中的特征），并且都可能包含噪音或错误。理想的效果是，神经网络能够自动地识别不同帧的重要性或置信度，来获得更好的融合效果。

（b）跨帧点的关联：为了融合来自不同帧的特征，我们需要对点进行跨帧关联。但是，动态点的位置会随时间而变化，并且它们是无序的。实际应用如自动驾驶等往往对时延有所要求，因此，简单地通过计算特征相似度或者追踪点的运动轨迹所带来的计算量是不能忍受的，这些原因使得跨帧点的关联相对困难。

在我们的 ASAP 模块中，我们提出了一种新的注意力机制的时间嵌入层和时空关联策略来分别解决上述两个难题。对于时间嵌入，我们引入注意力机制以自动识别不同帧的重要性和信任度，然后将这些特征与相应的注意力权重相乘，并将它们相加在一起，以更新特征。对于点关联，我们提出了一种简单但有效的恒定中心策略，来增强时间一致性并避免额外的计算。具体来说，我们仅在第一帧采样中心点，然后以迭代方式计算后续特征。该模块是一般结构，可以合并到不同的基干网中。

对于动态点云中语义分割任务，我们在目前开放的最大规模实际点云语义分割数据集 SemanticKITTI[14] 以及大规模模数据集 Synthia[23] 对我们的模型进行了充分的测试，效果并超过了目前已知的方法 MeteorNet[10] 和 4D MinkNet[11]。对于我们提出的用于时空感知的网络结构设计和学习策略，我们选取了两个需要对时空信息有很好理解的任务：动作识别和驾驶行为预测。对于动作识别，我们在三个常见数据集 UCF-101[1]、HMDB-51[2] 和 Kinetics-400[3] 上进行验证；对于驾驶行为预测，我们在 comma.ai[4] 和 LiVi-Set[5] 数据集上进行实验。实验结果充分证明了我们模型的有效性。

综上所述，本文的主要贡献可以归纳为：

- 我们对神经网络的时空感知进行了研究，提出了一种半耦合的网络设计策略，通过两个既相对独立又相互交织的支路分别处理时间和空间信息。
- 为了解决两个支路网络层数加深所带来的计算复杂度提升、内存占用以及训练过程中相互干扰的问题，我们提出了名为 STSGD 和高级 STSGD 的训练策略以及引入子任务以获得更好的时空学习效果。
- 为了能有效融合不同帧的点云特征以捕获时间信息，我们提出了一种新的注意力机制的时间嵌入层，并通过自动计算注意力来有效地融合跨帧的空间局部结构特征。
- 为了解决跨帧点云的关联性问题，我们提出一种时空关联策略，该策略能够充分利用结构信息，增强时间一致性并减少计算量。
- 我们提出一种称为 ASAP 模块的新的网络结构，并在点云序列中的语义分割进行了充分的实验，该框架可以灵活地嵌入到之前的静态点云处理网络中，并可以大幅度提高其语义分割的效果。我们进一步在动作识别和驾驶行为预测数据集上进行实验来验证我们半耦合的时空感知策略的有效性，两个实验都取得了很好的效果。

我们希望我们的时空感知网络设计和学习策略以及针对点云提出地 ASAP 模块可以使

点云序列分割研究和相关研究受益。

5.2 展望

虽然本文在动态点云的语义分割任务上取得了阶段性的成果，但仍有很多地方需要继续完善：

1、如章节 2.1所提到的，目前处理点云有各种各样的方法，并且各有优缺点，还未出现像 2D 中卷积神经网络（CNN）这样有效的处理方法，亟需学术界提出更加有效和普适的方法；

2、当前点云的处理方法相较于 2D 的网络，计算复杂度和内存占用是非常高的，如体素化的方法往往由于点云的稀疏性，高达 90% 的体素为空，这就导致内存利用率极低，而直接处理点云的方法，由于需要捕获局部结构，往往需要对邻点进行查询（按一定的半径或者 k 最近邻），这个过程会消耗大量的资源，因此需要在这两方面继续改进；

3、虽然我们的跨帧关联策略（ii）通过去除除第一帧外其他帧中的最远点采样过程一定程度地降低了计算复杂度，但点云序列中是有大量冗余信息的，加上实际任务对速度的要求，进一步的提速非常有必要。

参考文献

[1] SOOMRO K, ZAMIR A R, SHAH M. UCF101: A dataset of 101 human actions classes from videos in the wild. ArXiv preprint arXiv:1212.0402, 2012.

[2] KUEHNE H, JHUANG H, GARROTE E, et al. HMDB: a large video database for human motion recognition//2011 International Conference on Computer Vision. 2011: 2556-2563.

[3] CARREIRA J, ZISSERMAN A. Quo vadis, action recognition? a new model and the kinetics dataset//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017: 6299-6308.

[4] SANTANA E, HOTZ G. Learning a driving simulator. ArXiv preprint arXiv:1608.01230, 2016.

[5] CHEN Y, WANG J, LI J, et al. Lidar-video driving dataset: Learning driving policies effectively//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 5870-5878.

[6] WANG Z, JIA K. Frustum convnet: Sliding frustums to aggregate local point-wise features for amodal 3d object detection. ArXiv preprint arXiv:1903.01864, 2019.

[7] LIANG H, MA X, LI S, et al. Pointnetgpd: Detecting grasp configurations from point sets//2019 International Conference on Robotics and Automation (ICRA). 2019: 3629-3635.

[8] QI C R, YI L, SU H, et al. Pointnet++: Deep hierarchical feature learning on point sets in a metric space//Advances in neural information processing systems. 2017: 5099-5108.

[9] WU B, ZHOU X, ZHAO S, et al. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud//2019 International Conference on Robotics and Automation (ICRA). 2019: 4376-4382.

[10] LIU X, YAN M, BOHG J. MeteorNet: Deep learning on dynamic 3D point cloud sequences//Proceedings of the IEEE International Conference on Computer Vision. 2019: 9246-9255.

[11] CHOY C, GWAK J, SAVARESE S. 4d spatio-temporal convnets: Minkowski convolutional neural networks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 3075-3084.

[12] PAN J S, BINGHAM G P. With an eye to low vision: Optic flow enables perception despite image blur. Optometry and Vision Science, 2013, 90(10): 1119-1127.

[13] DING L, TERWILLIGER J, SHERONY R, et al. Value of Temporal Dynamics Information in Driving Scene Segmentation. ArXiv:1904.00758, 2019.

[14] BEHLEY J, GARBADE M, MILIOTO A, et al. SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences//Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV). 2019.

[15] KITAMURA T, SUN C, MARTIN J, et al. Entorhinal cortical ocean cells encode specific contexts and drive context-specific fear memory. Neuron, 2015, 87(6): 1317-1331.

[16] QI C R, SU H, MO K, et al. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017.

[17] TCHAPMI L, CHOY C, ARMENI I, et al. Segcloud: Semantic segmentation of 3d point clouds//2017 international conference on 3D vision (3DV). 2017: 537-547.

[18] LONG J, SHELHAMER E, DARRELL T. Fully convolutional networks for semantic segmentation//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 3431-3440.

[19] WU B, WAN A, YUE X, et al. Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud//2018 IEEE International Conference on Robotics and Automation (ICRA). 2018: 1887-1893.

[20] IANDOLA F N, HAN S, MOSKEWICZ M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. ArXiv:1602.07360, 2016.

[21] WANG Y, SUN Y, LIU Z, et al. Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (TOG), 2019, 38(5): 1-12.

[22] OLIVERI M, KOCH G, CALTAGIRONE C. Spatial–temporal interactions in the human brain. Experimental Brain Research, 2009, 195(4): 489-497.

[23] ROS G, SELLART L, MATERZYNSKA J, et al. The SYNTHIA Dataset: A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016.

[24] LAWIN F J, DANELLJAN M, TOSTEBERG P, et al. Deep projective 3D semantic segmentation//International Conference on Computer Analysis of Images and Patterns. 2017: 95-107.

[25] BOULCH A, LE SAUX B, AUDEBERT N. Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks. 3DOR, 2017, 2: 7.

[26] AUDEBERT N, LE SAUX B, LEFÈVRE S. Semantic segmentation of earth observation data using multimodal and multi-scale deep networks//Asian conference on computer vision. 2016: 180-196.

[27] TATARCHENKO M, PARK J, KOLTUN V, et al. Tangent convolutions for dense prediction in 3d//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 3887-3896.

[28] MILIOTO A, VIZZO I, BEHLEY J, et al. Rangenet++: Fast and accurate lidar semantic segmentation//Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS). 2019.

[29] HUANG J, YOU S. Point cloud labeling using 3d convolutional neural network//2016 23rd International Conference on Pattern Recognition (ICPR). 2016: 2670-2675.

[30] MENG H Y, GAO L, LAI Y K, et al. VV-Net: Voxel vae net with group convolutions for point cloud segmentation//Proceedings of the IEEE International Conference on Computer Vision. 2019: 8500-8508.

[31] RETHAGE D, WALD J, STURM J, et al. Fully-convolutional point networks for large-scale point clouds//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 596-611.

[32] DAI A, RITCHIE D, BOKELOH M, et al. Scancomplete: Large-scale scene completion and semantic segmentation for 3d scans//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4578-4587.

[33] GRAHAM B, ENGELCKE M, van der MAATEN L. 3D Semantic Segmentation With Submanifold Sparse Convolutional Networks//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

[34] SU H, JAMPANI V, SUN D, et al. SPLATNet: Sparse Lattice Networks for Point Cloud Processing//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

[35] ROSU R A, SCHÜTT P, QUENZEL J, et al. Latticenet: Fast point cloud segmentation using permutohedral lattices. ArXiv preprint arXiv:1912.05905, 2019.

[36] DAI A, NIEßNER M. 3dmv: Joint 3d-multi-view prediction for 3d semantic scene segmentation//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 452-468.

[37] CHIANG H Y, LIN Y L, LIU Y C, et al. A unified point-based framework for 3D segmentation//2019 International Conference on 3D Vision (3DV). 2019: 155-163.

[38] JARITZ M, GU J, SU H. Multi-view pointnet for 3D scene understanding//Proceedings of the IEEE International Conference on Computer Vision Workshops. 2019: 0–0.

[39] JIANG M, WU Y, ZHAO T, et al. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. ArXiv:1807.00652, 2018.

[40] ENGELMANN F, KONTOGIANNI T, SCHULT J, et al. Know What Your Neighbors Do: 3D Semantic Segmentation of Point Clouds//The European Conference on Computer Vision (ECCV) Workshops. 2018.

[41] ZHAO H, JIANG L, FU C W, et al. PointWeb: Enhancing local neighborhood features for point cloud processing//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 5565-5573.

[42] ZHANG Z, HUA B S, YEUNG S K. Shellnet: Efficient point cloud convolutional neural networks using concentric shells statistics//Proceedings of the IEEE International Conference on Computer Vision. 2019: 1607-1616.

[43] HU Q, YANG B, XIE L, et al. RandLA-Net: Efficient semantic segmentation of large-scale point clouds. ArXiv preprint arXiv:1911.11236, 2019.

[44] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need//Advances in neural information processing systems. 2017: 5998-6008.

[45] YANG J, ZHANG Q, NI B, et al. Modeling point clouds with self-attention and gumbel subset sampling//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 3323-3332.

[46] CHEN L Z, LI X Y, FAN D P, et al. LSANet: Feature Learning on Point Sets by Local Spatial Aware Layer. ArXiv preprint arXiv:1905.05442, 2019.

[47] ZHAO C, ZHOU W, LU L, et al. Pooling scores of neighboring points for improved 3D point cloud segmentation//2019 IEEE International Conference on Image Processing (ICIP). 2019: 1475-1479.

[48] ZHAO Y, BIRDAL T, DENG H, et al. 3D point capsule networks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 1009-1018.

[49] ARANDJELOVIC R, GRONAT P, TORII A, et al. NetVLAD: CNN architecture for weakly supervised place recognition//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 5297-5307.

[50] HUA B S, TRAN M K, YEUNG S K. Pointwise convolutional neural networks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 984-993.

[51] WANG S, SUO S, MA W C, et al. Deep parametric continuous convolutional neural networks//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 2589-2597.

[52] THOMAS H, QI C R, DESCHAUD J E, et al. Kpconv: Flexible and deformable convolution for point clouds//Proceedings of the IEEE International Conference on Computer Vision. 2019: 6411-6420.

[53] ENGELMANN F, KONTOGIANNI T, LEIBE B. Dilated point convolutions: On the receptive field of point convolutions. ArXiv preprint arXiv:1907.12046, 2019.

[54] ENGELMANN F, KONTOGIANNI T, HERMANS A, et al. Exploring spatial context for 3d semantic segmentation of point clouds//Proceedings of the IEEE International Conference on Computer Vision Workshops. 2017: 716-724.

[55] HUANG Q, WANG W, NEUMANN U. Recurrent slice networks for 3d segmentation of point clouds//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 2626-2635.

[56] YE X, LI J, HUANG H, et al. 3d recurrent neural networks with context fusion for point cloud semantic segmentation//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 403-417.

[57] ZHAO Z, LIU M, RAMANI K. DAR-Net: Dynamic aggregation network for semantic scene segmentation. ArXiv preprint arXiv:1907.12022, 2019.

[58] LIU F, LI S, ZHANG L, et al. 3DCNN-DQN-RNN: A deep reinforcement learning framework for semantic parsing of large-scale 3D point clouds//Proceedings of the IEEE International Conference on Computer Vision. 2017: 5678-5687.

[59] LANDRIEU L, SIMONOVSKY M. Large-Scale Point Cloud Semantic Segmentation With Superpoint Graphs//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

[60] LANDRIEU L, BOUSSAHA M. Point cloud oversegmentation with graph-structured deep metric learning//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 7440-7449.

[61] ZHIHENG K, NING L. PyramNet: Point cloud pyramid attention network and graph embedding module for classification and segmentation. ArXiv preprint arXiv:1906.03299, 2019.

[62] WANG L, HUANG Y, HOU Y, et al. Graph attention convolution for point cloud semantic segmentation//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 10296-10305.

[63] LIU X, QI C R, GUIBAS L J. FlowNet3D: Learning Scene Flow in 3D Point Clouds//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2019.

[64] FEICHTENHOFER C, FAN H, MALIK J, et al. Slowfast networks for video recognition//Proceedings of the IEEE International Conference on Computer Vision. 2019: 6202-6211.

[65] SIMONYAN K, ZISSERMAN A. Two-stream convolutional networks for action recognition in videos//Advances in neural information processing systems. 2014: 568-576.

[66] AZAM S, MUNIR F, RAFIQUE A, et al. Object Modeling from 3D Point Cloud Data for Self-Driving Vehicles//2018 IEEE Intelligent Vehicles Symposium (IV). 2018: 409-414.

[67] SALAH I B, KRAMM S, DEMONCEAUX C, et al. Summarizing large scale 3d point cloud for navigation tasks//2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). 2017: 1-8.

[68] KUNG Y C, HUANG Y L, CHIEN S Y. Efficient Surface Detection for Augmented Reality on 3D Point Clouds//Proceedings of the 33rd Computer Graphics International. 2016: 89-92.

[69] XINGJIAN S, CHEN Z, WANG H, et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting//Advances in neural information processing systems. 2015: 802-810.

[70] RUMELHART D E, HINTON G E, WILLIAMS R J. Learning representations by back-propagating errors. Nature, 1986, 323(6088): 533-536.

[71] WERBOS P J. Backpropagation through time: what it does and how to do it. Proceedings of the IEEE, 1990, 78(10): 1550-1560.

[72] BOTTOU L, CURTIS F E, NOCEDAL J. Optimization methods for large-scale machine learning. Siam Review, 2018, 60(2): 223-311.

[73] GEIGER A, LENZ P, URTASUN R. Are we ready for autonomous driving? The KITTI vision benchmark suite//2012 IEEE Conference on Computer Vision and Pattern Recognition. 2012: 3354-3361. DOI: 10.1109/CVPR.2012.6248074.

[74] KRIZHEVSKY A, SUTSKEVER I, HINTON G E. Imagenet classification with deep convolutional neural networks//Advances in neural information processing systems. 2012: 1097-1105.

[75] HE K, ZHANG X, REN S, et al. Deep residual learning for image recognition//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.

[76] PANG B, ZHA K, CAO H, et al. Deep rnn framework for visual sequential applications//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019: 423-432.

[77] DONAHUE J, ANNE HENDRICKS L, GUADARRAMA S, et al. Long-term recurrent convolutional networks for visual recognition and description//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 2625-2634.

[78] FEICHTENHOFER C, PINZ A, ZISSERMAN A. Convolutional two-stream network fusion for video action recognition//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 1933-1941.

[79] TATARCHENKO M, PARK J, KOLTUN V, et al. Tangent Convolutions for Dense Prediction in 3D//The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2018.

[80] GLOROT X, BENGIO Y. Understanding the difficulty of training deep feedforward neural networks//Proceedings of the thirteenth international conference on artificial intelligence and statistics. 2010: 249-256.

[81] IOFFE S, SZEGEDY C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. ArXiv preprint arXiv:1502.03167, 2015.

[82] KINGMA D P, BA J. Adam: A method for stochastic optimization. ArXiv preprint arXiv:1412.6980, 2014.

[83] HE K, ZHANG X, REN S, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification//Proceedings of the IEEE international conference on computer vision. 2015: 1026-1034.

[84] KIEFER J, WOLFOWITZ J, et al. Stochastic estimation of the maximum of a regression function. The Annals of Mathematical Statistics, 1952, 23(3): 462-466.

致　谢

经过一个学期的努力，毕业设计终于圆满完成了，在这里感谢一下在这个过程中给我提供了无私帮助的人。

首先要感谢我的毕业设计导师刘功申教授。在整个毕业设计的过程中从选题、模型设计到实验论证以及论文撰写都提供了宝贵的意见。刘老师在毕设期间，定期开展组会，讨论研究内容以及时间安排。在组会上，刘老师对相关研究的深刻见解给了我极大的启发，在我有疑问的时候总是给出详细的解答。在解决研究难题时，刘老师不仅在专业知识上给了我很大帮助，也给予了我很大的鼓励，他严谨的治学态度、锲而不舍的精神以及乐观的心态让我获益匪浅，激励我在学术的道路上不断前进，勇攀高峰。因此在这里，我对刘老师表达最诚挚的感激和敬意，谢谢刘老师无私的指导和帮助。

另外，我也要感谢上海交通大学计算机系的卢策吾老师、美国约翰霍普金斯大学的 Alan Yuille 老师和卢永毅博士后。卢老师在大学期间也在科研上给了我非常多的指导，这篇文章的时空感知部分的很多核心内容是与卢老师讨论的结果。除此之外，他从大二开始就帮助我开始进行计算机视觉方面的研究，在专业知识以及研究方法论方面都给了我莫大的帮助；Yuille 教授和卢永毅博士后也在毕设论文的研究中给了我很大帮助，如在相关领域难点、实验设定、论文撰写等方面都提供了很大的帮助。本此论文的顺利完成离不开两位的远程指导。

还要感谢上海交通大学为我的个人发展提供了非常好的平台，从基础学科学习、工程实践到学术研究都提供了优越的环境，包括专业能力很高的老师的指导以及设备支持等。也感谢四年来所有同学的陪伴，感谢大家的激励以及在我学习科研遇到低谷时的鼓励。同时也很珍惜和怀念课余时间跟大家一起度过的快乐时光。最后感谢一直默默支持我的父母，你们是我最可靠的后盾，也是我不断努力的力量源泉。

四年时间很快就过去了，我也即将开始新的科研旅程，希望我在今后的学习中能不忘初心，努力为相关领域的发展贡献自己的力量。再次感谢四年来给我帮助的老师，同学和家长！

攻读学士学位期间已发表或录用的论文

[1] Hanwen Cao, Yongyi Lu, Bo Pang, Gongshen Liu, Cewu Lu, Alan Yuille. ASAP-Net: Attention and Structure AwarePoint Cloud Sequence Segmentation[c]. The British Machine Vision Conference (BMCV, CCF-C, under review).

[2] Bo Pang, Kaiwen Zha, Hanwen Cao, Jiajun Tang, Minghui Yu, Cewu Lu. Complex sequential understanding through the awareness of spatial and temporal concepts[j].Nature Machine Intelligence (ISSN 2522-5839), 2020: 1-9.

[3] Bo Pang, Kaiwen Zha, Hanwen Cao, Shi chen, Cewu Lu. Deep rnn framework for visual sequential applications[c]. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR, CCF-A). 2019:423-432.

[4] Yizhuo Li, Jiefeng Li, Hanwen Cao, Muchen Li, Cewu Lu. TDAF: Top-Down Attention Framework for Vision Tasks[c]. Proceedings of the European Conference on Computer Vision (ECCV, CCF-B, under review).

DEEP LEARNING BASED DYNAMIC POINT CLOUD SEMANTIC SEGMENTATION WITH SPATIAL-TEMPORAL UNDERSTANDING

Dynamic point cloud sequences are readily-available input sources for many vision tasks. The ability to segment dynamic point clouds is a fundamental part of the perception system and will have a significant impact on applications such as autonomous driving[66], robot navigation[67] and augmented reality[68]. While great success has been achieved in static point cloud[8, 9, 16, 17, 40], the literature on modeling point cloud sequence has not been fully-explored. Therefore, we conduct research on dynamic point cloud segmentation with spatial-temporal understanding.

Methods for static point cloud processing can be basically separated into two categories: projection-based methods and point-based methods. For projection-based methods, they usually first convert irregular points into regular representations. For example, SEGCloud[17] first converts the point cloud to voxels and apply 3D convolution to get coarse voxel-wise semantic prediction. They then use tri-linear interpolation to project voxel predictions to single point and enforce the consistency with fully connected conditional random field (FCCRF).According to the working mechanism of LiDAR scanner, SqueezeSeg[19] propose to project the point cloud onto spherical surface accoring to the *azimuth* and *zenith* angles. It then apply a traditional encoder-decoder architecture to perform segmentation on 2D projection surface. For point-based methods, the pioneer work PointNet[16] has shown its capability of directly learning point-wise features from the raw point cloud. However, a single point does not have semantic meaning until it is considered with its neighbouring points. Subsequent works on static point cloud[8, 40] encode local structures from set of neighboring points and capture interactions among local structures. All those methods are restricted to single frame point cloud and fail to learn spatial-temporal information jointly.

For modeling dynamic point cloud sequence, an intuitive and straightforward way is to encode the local structures both spatially and temporally, *i.e.*, within a spatial-temporal neighborhood. Latest works[10, 11] follow this direction. 4D MinkNet[11] is projection-based, it first transforms the point cloud into regular voxels and applies a generalized sparse convolution along spatial and temporal dimensions. It proposes a new sparse convolution library to address the computation cost caused by the sparsity of point cloud. To better learn both spatial and temporal features, it also proposes a new shape of convolution kernel and a spatial-temporal conditional random field. In the contrast, MeteorNet[10] follows the point-based direction, it directly stacks multi-frame point clouds and computes local features by grouping spatio-temporal neighboring points according a specific radius.

Although both methods for dynamic point cloud segmentation are capable of capturing spatial-temporal features, it is hard for them to decouple spatial and temporal information in sequences. There is evidence that in real world, the human visual system largely relies on temporal information[12] and treats it as a primary source of information. Without decoupling spatial and temporal structures, it is not well understood to which degree temporal information can contribute to dynamic scene understanding[13]. Some researches about human brain also show that spatial and temporal

information are transferred to the hippocampus through two streams but also deeply coupled with each other. Therefore, it will be beneficial to related research if we can decouple the two aspects of information to some extent.

Inspired by the mechanism of humam brain, we propose a semi-coupled network design and training strategy to deal with the two aspects of information in a similar way. Specifically, we first separate the network to two streams so that the two aspects of information can be processed in a relatively independent way. Then to let them interact with each other, we stack the two streams deep and fuse the two in several layers. However, as the network becomes deep, the computation becomes increasingly expensive and the problem of mutual interference between them when back propagating becomes severe. To address the above two problems, we propose a new gradient based learning strategy names STSGD which cuts off some back-propagation paths according to some probability to alleviate the expensive computation and mutual interference. We further provide its advanced version ASTSGD to automatcially adjust the probabality. Also, decoupling spatial and temporal structures can make the model more extensible and flexible, *i.e.*, we can easily apply the static point cloud methods as backbone networks, which can be seen in Section 3.2.2 and Section 4.3.1.

Based on the general insights about spatial-temporal understanding above, we extend it to 3D dynamic point cloud. compared with regular images or videos, point cloud has varieties of unique characteristics. To effectively encode temporal structure, we need to tackle the following challenges:

(a) **Feature fusion with different frames**: To capture temporal information, we need to fuse features from different frames. However, the features from different frames may contribute differently to the results (for example, fast-moving objects may rely more on current frame while slow-moving object may also rely much on previous frames)and they are all likely to contain undesired noise or mistakes. Ideally, the network should automatically identify the importance or confidence degree of different frames to achieve better fusing results.

(b) **Point correlation across frames**: To fuse features from different frames, we need to correlate points across frames. However, the distribution of dynamic points varies from time to time and they are unordered, making it challenging to correlate. Also, simply matching the points according to feature similarity or tracking through time can increase the computation which is unaffordable for many tasks.

In our ASAP module, we propose a novel attentive temporal embedding layer and spatial-temporal correlation strategy to tackle the above two challenges respectively. For temporal embedding, we introduce attention mechanism to automatically identify the importance and trust degree of different frames then multiply these features with corresponding attention scores and sum them together to get an updated feature. For point correlation, we propose a simply but effective constant-center strategy to enforce temporal consistency and avoid extra computation. Specifically, we only sample center points at the first frame and compute following features in an iterative way. The module is a general structure and can be incorporated into different backbone networks.

We conduct thorough experiments to show our ASAP module's advantage over state-of-the-art methods[10, 11] and its generalization ability of improving the performance of different backbones. Figure 4–3 provides qualitative results of our approach, showing the effectiveness of our ASAP module. We also further conduct experiments on action recognition datasets UCF-101[1], HMDB-51[2], Kinetics-400[3] and driver behavior datasets comma.ai[4], LiVi-Set[5]to show the effectiveness

of our semi-coupled spatial-temporal modeling strategy. Our key contributions are:

- We conduct research on spatial-temporal understanding which is really important to many real-world applications and propose a new semi-coupled network design strategy to process the two aspects of information with two relatively independent but interactive streams, just like the mechanism in human brain.
- To solve the problems of expensive computation, memory explosion and mutual interference when the network becomes deep, we propose a new training strategy named STSGD to cut off some paths according to a probability and its advanced version ASTSGD to automatically adjust the probability, thus achieving semi-coupling in the training level.
- We introduce a novel attentive temporal embedding layer to fuse the spatial local features across frames by automatically calculating attentions and summing those features with the attention weights, thus achieving better and more rubost results.
- We present a spatial-temporal correlation strategy which use constant centers and update their features in an iterative way to exploit structural information, enforce temporal consistency and reduce computation.
- We propose a novel architecture called ASAP module which can be flexibly plugged into previous static point cloud pipeline and conduct thorough experiments to show that our module can achieve improvement by a large margin. We also further evaluate our spatial-temporal learning strategy with action recognition and driver behavior prediction experiments. Both show the effectiveness of our strategy.