第三章　基于双曲流形映射的点云持续学习缓存重放方法研究

近年来，持续学习在深度学习领域受到了越来越多的关注。为了解决灾难性遗忘这一挑战，研究人员提出了多种应对策略，例如基于重放的策略 [85]、基于正则化的策略 [16] 以及基于参数隔离的策略 [86]。值得注意的是，这些策略在设计上具有正交性，因此在某些场景中可以进行组合，从而充分发挥各自的优势，取得更优异的效果。

在这些方法中，重放策略作为一种简单而有效的手段得到了广泛应用。重放策略通过存储部分原始样本或使用生成模型生成伪样本，在学习新任务的同时，通过重放先前任务的样本来缓解遗忘问题。通过周期性地回顾过往的经验和信息，该方法能够加强学习和记忆，将知识和技能更持久地存储于长期记忆中。然而，重放方法可能在样本选择过程中倾向于特定类型的样本，从而可能引入样本选择偏差，影响模型的泛化能力。

本章节聚焦于三维对象即点云的持续学习任务，针对现有方法的局限性提出了改进方法。三维点云数据具有显著的分层树状结构，每个点云由多个较小部分组合而成。利用这一特性，我们能够无缝地结合全局信息和局部信息，从而提升持续学习的能力。然而，传统的欧几里得空间 (Euclidean Space) 由于其体积随半径多项式增长的特性，无法很好地表示树状数据，从而导致表示能力的不足。幸运的是，由于负曲率的特性，双曲空间 (Hyperbolic Space) 能够以低失真嵌入分层结构 [87]，显著增强对三维数据特征的表示能力。

本章节的主要贡献如下：

(1) 优化并改进了基于贪心采样策略的重放缓存区，扩展其至三维点云数据，提升了其在持续学习中的适用性。

(2) 引入双曲空间映射，利用双曲空间更好表示点云特征信息，并在此基础上提出了一种结合双曲空间全局与局部信息的知识蒸馏策略，帮助模型在持续学习任务中有效克服灾难性遗忘问题。

3.1 持续学习任务设置

首先，我们将简要概述持续学习的任务设置，然后详细阐述我们提出的方法的具体内容。

我们先定义一组持续学习任务序列，记为 T，包含总共 N 个任务，即

$$T = (T_1, T_2, ..., T_N)$$

对于每个任务 t，其对应的数据集为

$$D_t = ((x_t^1, y_t^1), (x_t^2, y_t^2), ..., (x_t^n, y_t^n))$$

其中 x 表示样本，y 表示标签。我们规定在学习过程中，模型只能使用容量有限的缓存区 B 来存储少量来自先前任务的样本。

在评估标准方面，我们采用平均整体任务准确率 (ACC) 作为判别依据，其定义为

$$ACC = \frac{1}{T}\sum_{i}^{T} a_T \tag{3.1}$$

其中 a_i 表示任务 i 的准确率。

3.2 双曲空间投影

双曲空间 (Hyperbolic Space)[88] 是一种具有负曲率的非欧几里得空间。与欧几里得空间中的平面或球面不同，双曲空间呈现“马鞍形”负曲率。在双曲空间中，直线是最短路径，但直线之间的角度概念与欧几里得空间不同——直线之间的角度是双曲角。此外，在双曲空间中，两条平行线会越来越接近，但永不相交。因此，双曲空间在许多应用中表现出了优异性能，尤其是在表示和建模层次化结构数据方面，例如语言表示学习、图嵌入和神经网络中的层次建模等领域。

目前已有不少的工作 [89, 90, 91, 92] 指出，将点云数据映射到双曲空间，相比欧式空间能够更好地获取其特征信息，这促使我们考虑利用双曲投影来更好地处理点云数据。

在本章节，我们采用广泛使用的庞加莱圆盘模型 (Poincaré Disk Model) 来描述双曲空间。双曲空间的负曲率由以下用于描述空间度量的公式决定：

$$G_R = (\lambda_x^c)^2 g_E = \frac{2}{1+||x||} g_E \tag{3.2}$$

其中，λ 是保角因子 (Conformal Factor)，g_E 是欧几里得度量张量，c 是庞加莱圆盘的曲率。双曲空间不同于定义在欧几里得空间中的向量空间，但许多算子可以通过莫比乌斯向量空间 (Möbius Vector Spaces) 扩展到双曲空间。

庞加莱球是一种曲率为 $c=-1$ 的双曲空间，其距离定义为：

$$d_D(x,y) = cosh^{-1}(1 + \frac{2||x-y||^2}{(1-||x||^2)(1-||y||^2)}) \tag{3.3}$$

由于庞加莱球是一个黎曼流形，对于球面上的每一点，可以使用指数映射 (Exponential Map) 和对数映射 (Logarithmic Map) 将切向量映射到流形上，或将流形上的点映射回切空间：

$$exp_x^c(w) = x \oplus_c (tanh(\sqrt{c}\frac{\lambda_x^c||w||}{2})\frac{w}{\sqrt{c}||w||}) \tag{3.4}$$

$$log_x^c(y) = \frac{2}{\sqrt{c}\lambda_x^c} arctanh(\sqrt{c}||-x \oplus_c y||)\frac{-x \oplus_c y}{||-x \oplus_c y||} \tag{3.5}$$

其中，w 是流形在点 x 的切空间中的切向量。

点云作为一种具有树状分层结构的数据类型，非常适合映射到双曲空间中。每个点云由较小的部分向上生长而成。因此，我们旨在将点云数据表示在双曲空间中，利用双曲空间在表示树状分层结构方面的卓越能力，更好地理解点云的特征信息 [89]。具体而言，在庞加莱圆盘上，我们将完整点云定位在靠近圆盘边缘的位置，而将点云结构的部分定位在靠近圆盘中心的位置。此外，我们确保不同类别的点云彼此保持较远的距离，而同一类别的点云保持相对较近的距离。为实现这一目标，模型主要受以下两个正则化公式的约束：

$$L_{pos} = max(0, -||z_{whole}^+|| + ||z_{part}^+|| + \delta) \tag{3.6}$$

$$L_{neg} = max(0, d(z_{whole}^+, z_{part}^+) - d(z_{whole}^+, z_{part}^-) + \delta) \tag{3.7}$$

其中，δ 表示偏移量，z_{whole}^+ 表示完整的正样本点云，z_{part}^+ 表示正样本点云的部分，z_{part}^- 表示负样本点云的部分。通过这种方式，我们能够更好地理解点云的局部特征。

3.3 双曲流形缓存重放

最近的研究 [93] 表明，重放方法是一种应对灾难性遗忘的简单且有效的手段。然而，重放方法在平衡存储容量与性能以及样本选择方面面临挑战，此外，如何对这些缓存内容进行重放也是研究的重点问题之一，我们的方法将围绕解决这些问题展开。

重放方法设计

首先，我们在模型架构中引入一个双曲映射特征提取层，通过指数映射将点云特征从欧式空间映射到双曲空间，并对提取出的特征进行拼接等进一步的处理。

同时，如前所述，点云数据呈现出明显的层次化树状结构，每个复杂的点云由简单的部分组成。换句话说，点云数据具有显著的局部结构特征。然而，仅考虑局部特征并不够全面，因为全局信息同样至关重要。幸运的是，我们的网络设计能够同时满足对局部和全局信息的需求。在池化之前，我们可以在双曲空间中捕获点云数据的局部特征，而在池化之后，我们可以获得点云数据的全局特征。因此，我们通过公式3.3 从教师-学生模型中蒸馏出这两种特征，并最终将这些结果融合以应对灾难性遗忘的问题。这种综合利用局部和全局特征的方法有望使模型能够更好地理解点云数据，并促进不同任务之间的迁移学习和持续学习，从而更好地适应不同场景中的数据变化。因此，我们的损失函数定义如下：

$$\begin{aligned} Loss = & L_D(f_s(x;\theta), y) + L_M(f_s(x;\theta), y) \\ & + L_{distance}(\phi_s(x;\theta), \phi_t(x;\theta)) + L_{pos} + L_{neg} \end{aligned} \tag{3.8}$$

其中，L_D 是对应当前数据集的交叉熵，L_M 是对应重放缓存区中存储的数据集的交叉熵，$L_{distance}$ 表示 TS 蒸馏模型之间的距离 (包括整体结构和部分结构的距离)。

重放缓存设计

重放缓存区的大小是有限的，一旦达到其上限，随着新任务的加入，不可避免地会出现缓存替换问题。然而，简单的随机替换方法往往效果较差，因此选择高质量的旧数据变得至关重要。我们希望缓存的旧知识能够充分代表其对应的旧任务，

但在实践中，这一点很难实现。因此，我们需要精心挑选具有代表性的数据作为重放的候选内容。

我们的方法是通过将数据分布投影到一个流形球面上，让该球面的半径随着新任务的加入不断扩展。而由于数据特征分布在双曲空间上，基于双曲空间的质心和半径需要依赖迭代的梯度下降算法，需要耗费不少计算资源，因此我们对特征投影回欧式空间构造流形。

对于新样本，我们将其流形中心与当前流形直径进行比较。如果其距离超过当前半径，将其包含在缓存区中是有利的。至于被替换的旧样本，考虑到计算成本以及衡量旧样本对缓存区贡献的难度，我们选择采用随机替换策略。

此外，由于我们在重放方法中结合了局部和全局特征，因此分别为局部特征和全局特征建立独立的流形球面。我们基于归一化权重为每个特征独立选择样本，以充分利用数据的多样性和代表性。这种综合策略不仅有效解决了缓存替换问题，还增强了模型的泛化能力和任务的鲁棒性。我们的选择策略如算法1所示。

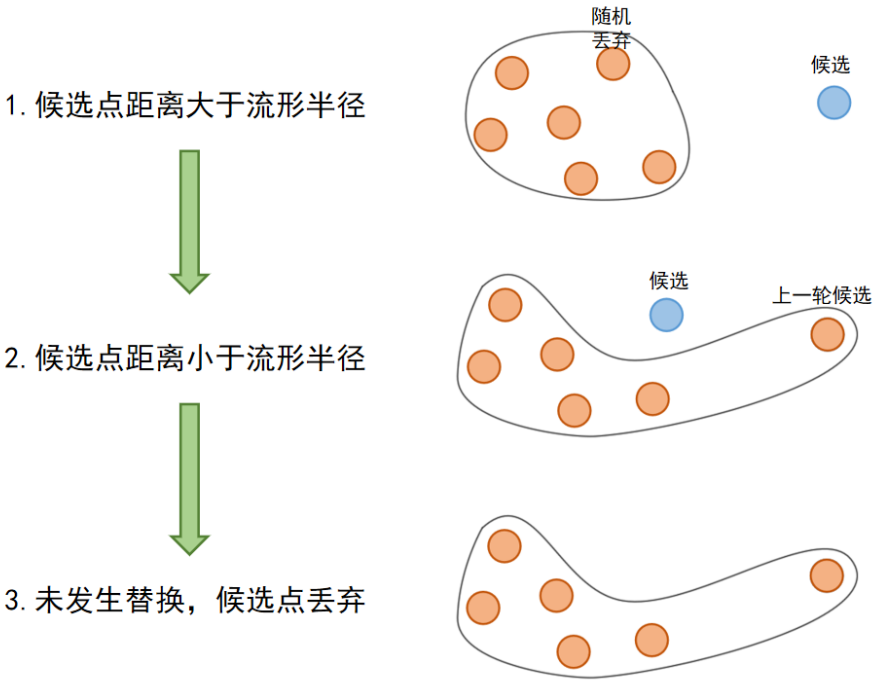

图 3.1　双曲流形扩张淘汰算法图解。

算法 1 双曲流形扩张淘汰算法

Input: $\Phi_s, D_t, \mathcal{M}$, 缓存区大小上限 buffer_size, n, α_1, α_2

Output: $\mathcal{M}$

$n \leftarrow 0$
for 所有 (x, y) 属于 D_t **do**
  if $|\mathcal{M}| <$ buffer_size **then**
    $\mathcal{M}$.append(x, y)
  else
    $f^{\mathcal{M}}_{local}, f^{\mathcal{M}}_{global} \leftarrow log^c_0(\Phi_s(\mathcal{M}))$
    $f^{x}_{local}, f^{x}_{global} \leftarrow log^c_0(\Phi_s(x))$
    $c_{local}, d_{local} \leftarrow$ 计算流形质心和半径$(f^{\mathcal{M}}_{local})$
    $c_{global}, d_{global} \leftarrow$ 计算流形质心和半径$(f^{\mathcal{M}}_{global})$
    计算是否比最远样本更偏离中心:
    $\delta \leftarrow \alpha_1(\text{dist}(c_{local}, f^{x}_{local}) - d_{local})$
      $+\alpha_2(\text{dist}(c_{global}, f^{x}_{global}) - d_{global})$
    if $\delta > 0$ **then**
      $i \leftarrow$ randint$(0,$ buffer_size$)$
      $\mathcal{M}[i] \leftarrow (x, y)$
    else
      $i \leftarrow$ randint$(0, n)$
      if $i <$ buffer_size **then**
        $\mathcal{M}[i] \leftarrow (x, y)$
      end if
    end if
  end if
  $n \leftarrow n + 1$
end for
return $\mathcal{M}$

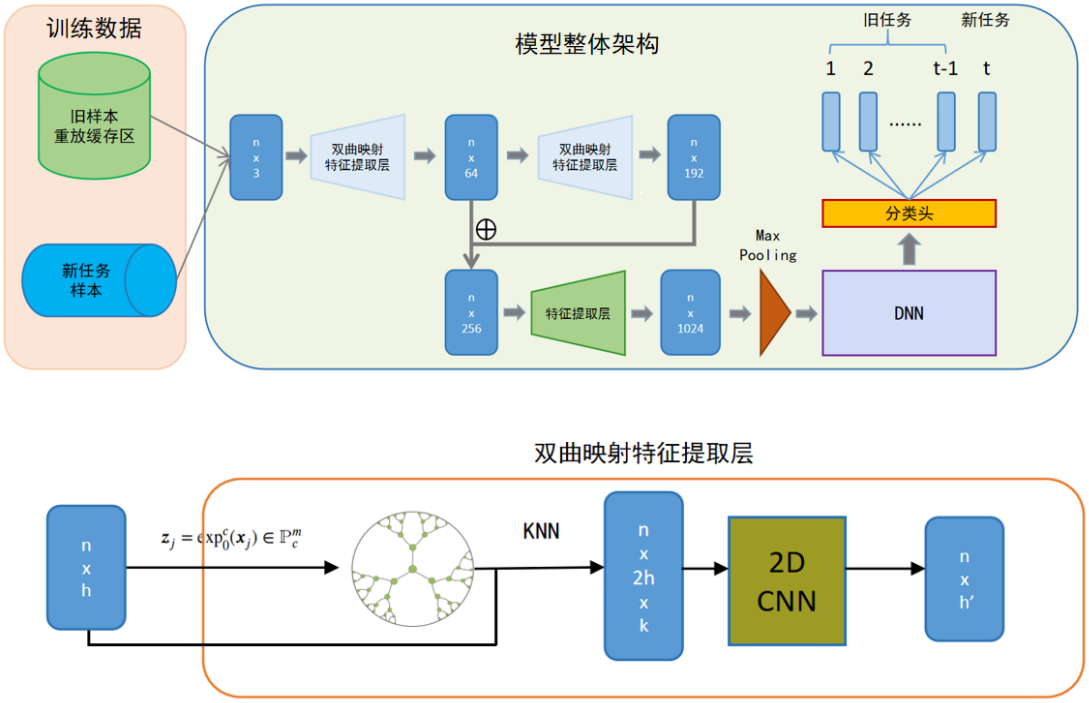

图 3.2　HyMR 模型整体架构。

3.4　实验及分析

我们将我们的方案取名为双曲流形重放 (HyMR)，整体架构如图3.2所示，并在不同数据集上使用 HyMR 进行多任务持续学习，然后将其与最新的以及常见的点云持续学习方法进行比较。

3.4.1　实验设置

我们在两个广泛使用的点云数据集 ModelNet[94] 和 ShapeNet[95] 上评估了算法的有效性。对于所有数据集，我们对每个点云数据统一采样 2048 个点，并进行共 20 轮的类增量任务。在初始任务中，我们选择样本数量最多的两个类别，而在随后的每一轮中，依次加入两个新类别。用于重放的缓存区大小设置为总样本数的 5%，约为 500 个样本。

我们使用 PyTorch 库实现了 HyMR 的所有模块，并在两块 NVIDIA TESLA V100 GPU 上进行训练。对于模型的骨干网络 (backbone)，我们选择了在点云分类中常用的 PointNet，并在此基础上引入了一个双曲映射模块。在双曲映射特征提取

层中，我们引入了一个双曲映射层，将特征投影到双曲空间中，然后使用卷积核大小为 1 的卷积网络层进行特征提取，最后将获得的特征进行拼接。

对于分类器，我们使用了三层全连接层 (512-256-类别数)。在分类头中，我们通过更改最终输出层来匹配类别数量的增长。具体来说，每当出现新任务时，模型会冻结之前的输出层，并为新类别创建一个专用的输出层。这种方法有助于将模型大小保持在合理范围内。学习率统一设置为 0.003，批量大小设置为 16，每个任务最多训练 50 个周期 (epochs)。

我们将提出的方法与几种较新的点云持续学习相关方法进行了比较，并以顺序微调 (sequential fine-tuning) 作为对照。需要指出的是，部分对比方法 (如 iCaRL[28]、LwF[15] 等) 最初设计用于图像分类任务，其核心思想分别为样本重放、知识蒸馏和正则化约束。在迁移到点云任务中时，我们对其进行了必要的调整，使其适用于点云数据的输入格式与特征表示。例如，对于 iCaRL，我们在提取点云特征后执行最近邻分类策略；对于 LwF，我们保留其损失函数的核心机制，并将主干网络替换为适用于点云处理的基础结构 (PointNet[21])。尽管这些方法在点云场景中并非原生设计，但通过适当的适配，仍具备一定的代表性，能够作为有效的 baseline 进行性能比较。我们在所有数据上进行了五次独立实验，并计算了平均结果。

3.4.2 数据集介绍

ModelNet40 是一个广泛用于三维点云分类任务的基准数据集，最早用于评估 3D 形状识别任务。该数据集是 ModelNet 数据集 [94] 的一个子集，专门包含了 40 个常见物体类别，因此得名 ModelNet40。该数据集包含 40 个不同类别的物体，每个类别涵盖各种日常物体，如桌子、椅子、飞机、汽车等，涵盖了现实世界中常见的三维结构。数据集中的每个对象均由 3D CAD 模型转换而来，并采用点云或网格的形式进行表示，为研究人员提供了一个标准化的测试环境，以验证模型在三维形状识别任务上的泛化能力。

ShapeNet 是一个大规模的三维模型数据库，专为计算机视觉、计算机图形学和深度学习研究而构建。该数据集涵盖了大量来自不同类别的三维物体，广泛涉及日常生活中的物品，如家具、交通工具和家用电器等。所有模型均采用标准化的

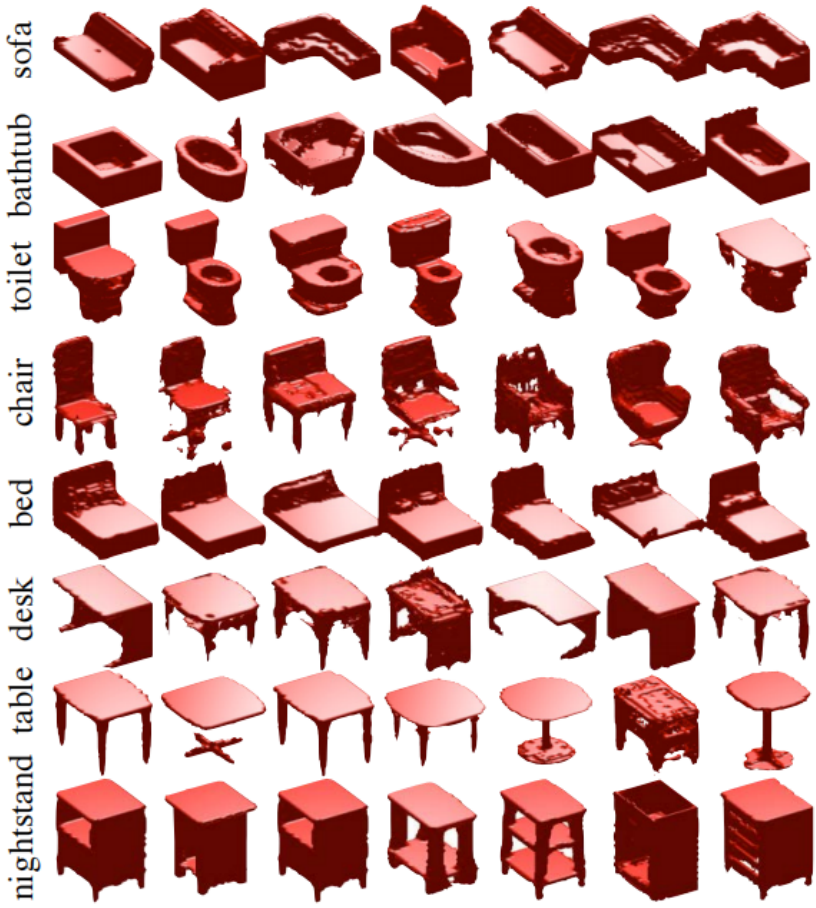

图 3.3　ModelNet 数据集预览。

格式，保证了数据的一致性，并在几何、拓扑结构和语义层面进行了细致的标注。这些标注信息不仅包括物体的类别，还涵盖了部分分割、对齐信息、物理尺寸和对称性等特征，使得 ShapeNet 在语义理解、形状生成和三维重建等任务中具有极高的应用价值。由于其规模庞大、类别丰富且标注详尽，ShapeNet 已成为三维视觉和计算机图形学领域最具代表性的数据集之一，为学术界和工业界的研究提供了强有力的支持。

3.4.3　结果分析

在表 3.3 中，我们展示了在 ShapeNet 和 ModelNet 数据集上的类别增量消融实验结果，比较了不同缓存区大小和是否使用双曲映射的情况。结果表明，HyMR 的性能在一定程度上依赖于缓存区的大小，并且采用双曲空间映射显著增强了模型

图 3.4　ShapeNet 数据集预览。

表 3.1　在类别增量学习中，HyMR(双曲映射) 与其他在欧式空间的不同方法在 ModelNet40、ShapeNet 数据集上经过若干轮训练后的平均准确率 (%)。

数据集	任务数	顺序微调	iCaRL[28]	LwF[15]	RPS-Net[96]	I3DOL[19]	HyMR(Ours)
ModelNet40	10	13.3	71.8	62.3	86.7	89.7	**90.3**
	20	2.1	57.2	58.3	31.5	61.5	**78.0**
ShapeNet	10	9.7	72.9	72.7	86.9	87.8	**89.8**
	20	1.3	36.4	48.3	69.5	74.1	**80.8**

对点云特征的理解能力。

在图 3.5、3.6和表3.1、3.2中，我们展示了在 ShapeNet 和 ModelNet 数据集上，随着任务轮数增加，HyMR 方法与其他方法在类别增量和任务增量设置下的实验结果。可以观察到，HyMR 相较于基线方法表现出更优越的性能，并且随着任务轮数的增加，其效果更加显著。

表 3.2　在任务增量学习中，HyMR(双曲映射) 与其他在欧式空间的不同方法在 ModelNet40、ShapeNet 数据集上经过若干轮训练后的平均准确率 (%)。

数据集	任务数	iCaRL[28]	LwF[15]	EWC[17]	HyMR(Ours)
ModelNet40	10	89.9	92.8	92.2	**98.6**
	20	88.2	91.7	87.7	**98.3**
ShapeNet	10	93.3	91.9	83.3	**98.6**
	20	82.0	75.3	70.3	**98.3**

表 3.3　类增量任务在 ModelNet40 和 ShapeNet 数据集上，对于不同消融方案的 HyMR 方法，经过 10 轮和 20 轮任务训练后的最终结果。

数据集	消融方案	准确率 (10 轮任务)	准确率 (20 轮任务)
ModelNet40	HyMR	90.3	**78.0**
	传统蒸馏	**90.9**	77.7
	缓存区随机淘汰	88.3	76.8
	无双曲映射	90.6	77.9
	缓存区 100 上限	85.7	70.3
	缓存区 250 上限	86.9	73.5
ShapeNet	HyMR	**89.8**	**80.8**
	传统蒸馏	89.6	80.1
	缓存区随机淘汰	87.6	77.7
	无双曲映射	88.5	80.3
	缓存区 100 上限	84.4	69.9
	缓存区 250 上限	87.4	74.1

3.5　章节小结

在本章节的工作中，我们提出了一种针对点云数据持续学习的新方法。该方法通过将特征映射到双曲空间，并结合全局与局部视角，利用重放策略来解决问题。我们认为，利用双曲空间出色的层次化表示能力，并全面考虑数据特征，可以有效缓解点云分类任务中的灾难性遗忘问题。然而，该方法在某些方面仍然存在一些问题，例如在选择缓存区时计算直径和距离的计算成本较高，并且过于依赖缓存区。在第四章中，我们将继续优化方法，并寻求更优的解决方案。

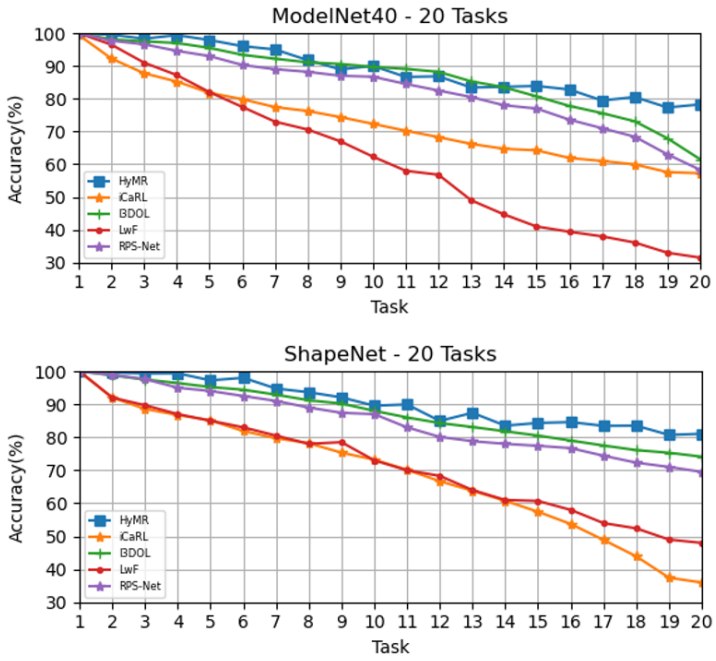

图 3.5　类增量学习设置下，在完成 t 个任务后的整体平均准确率，训练和验证期间，每个任务的类别信息是未知的。图中展示了 HyMR(双曲映射) 与其他基于欧式空间的方法在 ModelNet40 和 ShapeNet 数据集上训练 20 轮任务后的准确率变化。

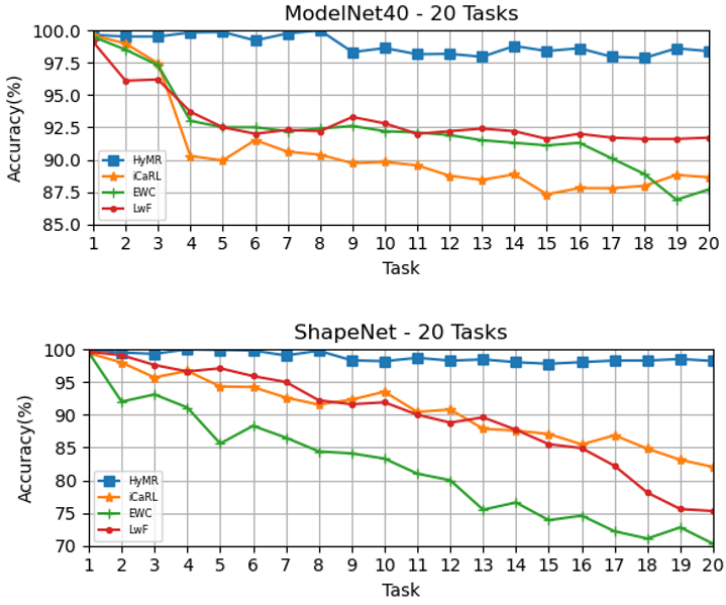

图 3.6　任务增量学习设置下，完成 t 个任务后的整体平均准确率，训练和验证期间，每个任务的类别信息是已知的。图中展示了 HyMR(双曲映射) 与其他基于欧式空间的方法在 ModelNet40 和 ShapeNet 数据集上训练 20 轮任务后的准确率变化。

第四章　基于生成式模型的双曲映射点云持续学习方法研究

在上一章节中，我们提出了一种基于双曲流形映射的点云持续学习缓存重放方法，通过优化贪心采样策略，提升了重放缓存区在点云数据上的适用性，并结合双曲空间的全局与局部信息，改善了模型的表示能力。然而，我们发现基于缓存的重放方法仍然存在一些局限性：缓存策略在采样过程中可能倾向于特定类型的样本，影响模型的泛化能力；缓存区的大小有限，无法完整保留所有历史任务的数据，导致旧任务的样本多样性受限；某些任务 (如医疗、金融) 可能无法直接存储原始数据，限制了缓存重放的应用范围；而且，目前点云生成持续学习的研究内容较少，传统的方法迁移到点云上后灾难性遗忘依旧明显。

同时，即使在之前工作中我们已经采用了优化的贪心采样策略和双曲映射提升了缓存重放效果，但从实验的结果中也能看出，旧任务表现下降仍然明显，尤其在任务数增加时效果迅速劣化，并且模型对早期任务的性能仍然随任务数增加持续下降，下降速度和缓存大小成反比例关系。

根据上述情况，我们意识到仅依赖缓存机制的持续学习方法在点云场景下存在天然的适用性边界。一方面，缓存区的有限容量难以适应点云数据的大规模与高维度特性；另一方面，在对隐私敏感或数据共享受限的场景中，缓存机制本身的适用性也受到制约。于是，我们尝试从“存储真实样本”的被动方案，转向“主动生成高质量样本”的生成式方法，期望以更高效、更灵活的方式缓解灾难性遗忘问题，使之成为一种理论上更普适的补充甚至替代方案，因为它不依赖保存真实样本，而是从模型中提取知识用于数据重构。

因此，本章提出了一种基于生成模型的双曲映射点云持续学习重放方法，用扩散模型 (Diffusion Model) 替代传统的重放缓存区，生成高质量、多样化的点云数据，并结合双曲空间的表示能力，提高点云数据的重放质量和持续学习能力。具体来说，本章节的主要贡献如下：

(1) 提出了一种基于扩散模型的点云生成式重放方法 (HyperDiffIL)，突破了传统缓存区方法的存储限制，提高了重放样本的多样性，并且支持分类和生成任务。

(2) 结合双曲空间进行特征映射和优化，提升了点云数据的表示能力，并在此基础上设计一种生成样本筛选机制，使生成样本更准确。

(3) 设计了一种适用于双曲空间下扩散模型特征的全过程知识蒸馏策略，确保生成样本的质量，同时减少灾难性遗忘，提高模型的长期学习能力。

4.1 基于双曲空间的扩散模型点云生成

4.1.1 传统扩散模型

去噪扩散概率模型 (Denoising Diffusion Probabilistic Model, DDPM)[40] 是一种基于马尔可夫链的生成方法，它通过正向扩散过程将数据逐步扰动为纯噪声，然后通过反向去噪恢复到原始数据分布。在生成任务上，这一反向过程的精确控制使得扩散模型能够在生成任务中表现出卓越的性能，尤其是在面对高维度、复杂数据时，能够有效地处理数据的细粒度结构，避免了传统生成模型中常见的模式崩塌问题。通过这种方式，扩散模型不仅能够生成高质量的样本，还能够在任务中进行有效的特征学习，展现出强大的生成能力和鲁棒性。

前向扩散过程

扩散模型的前向过程通过逐步向数据添加噪声，使其分布收敛到标准正态分布。设 $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 为从真实数据分布中采样的数据，正向扩散 (从 $t-1$ 步扩散到 t 步) 的过程定义如下：

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1}) \tag{4.1}$$

我们逐步向数据添加噪声，使其逐渐变得无序，最终趋近于一个标准高斯分布。这个过程可以用以下的马尔可夫链建模：

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = N(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \tag{4.2}$$

其中,β_t 是一个超参数，称为噪声调度 (variance schedule)，决定了每一步扩散的噪声大小,$N(\mu, \Sigma)$ 表示均值为 μ，协方差矩阵为 Σ 的正态分布。通过重参数化，我们可以很快地得到任意时刻 t 处的点云分布：

$$q(\mathbf{x}_t|\mathbf{x}_0) = N(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{4.3}$$

其中，

$$\alpha_t = 1 - \beta_t, \quad \bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s \tag{4.4}$$

于是，在具体实践中，我们可以直接通过一步采样得到 $\mathbf{x}_t$：

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim N(\mathbf{0}, \mathbf{I}) \tag{4.5}$$

反向去噪过程

为了从噪声恢复出原始数据，我们需要学习一个反向去噪模型 $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$，即：

$$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = N(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)) \tag{4.6}$$

其中，$\boldsymbol{\mu}_\theta(\mathbf{x}_t, t)$ 是可学习的均值函数 (由神经网络估计)，$\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)$ 是可学习的协方差矩阵。利用贝叶斯定理，我们可以计算真实的后验分布：

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = N(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \tag{4.7}$$

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\mathbf{x}_t \tag{4.8}$$

$$\tilde{\beta}_t = \frac{(1-\bar{\alpha}_{t-1})}{1-\bar{\alpha}_t}\beta_t \tag{4.9}$$

而在具体应用中，我们可以用神经网络 $\epsilon_\theta(\mathbf{x}_t, t)$ 来模拟 $\mathbf{x}_0$：

$$\hat{\mathbf{x}}_0 = \frac{\mathbf{x}_t - \sqrt{1-\bar{\alpha}_t}\epsilon_\theta(\mathbf{x}_t, t)}{\sqrt{\bar{\alpha}_t}} \tag{4.10}$$

最终，去噪步骤为：

$$\mathbf{x}_{t-1} = \boldsymbol{\mu}_\theta(\mathbf{x}_t, t) + \sqrt{\tilde{\beta}_t}\mathbf{z}, \quad \mathbf{z} \sim N(\mathbf{0}, \mathbf{I}) \tag{4.11}$$

训练目标

扩散模型的训练目标通常是最大化对数似然函数，这通常是一个难以直接计算的目标。因此，采用变分下界 (Variational Lower Bound, VLB) 来优化，目标是最小化变分下界：

$$L_{\text{VLB}} = \mathbb{E}_q \left[\sum_{t=1}^{T} D_{\text{KL}}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \parallel p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)) - \log p_\theta(\mathbf{x}_0) \right] \tag{4.12}$$

而为了简化训练过程，扩散模型通常采用一种更直接的训练目标，即通过最小化模型在各时间步 t 上输出的噪声估计与真实噪声之间的误差。简化后可以得到最终的训练目标为：

$$L_{\text{simple}} = \mathbb{E}_{t,\mathbf{x}_0,\boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \epsilon_\theta(\mathbf{x}_t, t)\|^2\right] \tag{4.13}$$

4.1.2 点云的扩散模型生成

设定点云数据集为：

$$\mathbf{X}_0 = \{\mathbf{x}_i\}_{i=1}^N, \quad \mathbf{x}_i \in \mathbb{R}^3 \tag{4.14}$$

传统扩散模型主要应用于图像生成，但如第三章所述，由于点云数据的非欧几里得结构，直接在欧几里得空间进行扩散可能无法很好地捕捉点云的层次化信息。因此，我们考虑将点云特征投影到双曲空间进行扩散和反向过程，并最终在三维欧几里得空间生成点云，以获取更好的效果。这里和第三章一样，我们选择庞加莱圆盘：

$$\mathbb{D}^n = \{\mathbf{z} \in \mathbb{R}^n \mid \|\mathbf{z}\| < 1\} \tag{4.15}$$

其中，庞加莱度量定义为：

$$ds^2 = \frac{4d\mathbf{z}^2}{(1 - \|\mathbf{z}\|^2)^2} \tag{4.16}$$

为了将点云投影到双曲空间，我们采用指数映射：

$$\mathbf{z}_i = \exp_{\mathbf{0}}(\mathbf{x}_i) = \tanh(\|\mathbf{x}_i\|)\frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \tag{4.17}$$

从而得到点云在双曲空间的表示：

$$\mathbf{Z}_0 = \{\mathbf{z}_i\}_{i=1}^N, \quad \mathbf{z}_i \in \mathbb{D}^2 \tag{4.18}$$

双曲空间中的正向扩散

在传统的扩散模型中，数据点在欧几里得空间内逐步添加噪声，以转换为标准高斯分布。然而，在双曲空间中，数据点的运动需要符合双曲几何。因此，我们定义双曲空间内的扩散过程：

$$q(\mathbf{z}_t|\mathbf{z}_{t-1}) = N_{\mathbb{D}}(\mathbf{z}_t; \sqrt{1-\beta_t}\mathbf{z}_{t-1}, \beta_t I) \tag{4.19}$$

其中 β_t 是噪声调度参数，决定每一步扩散的噪声大小。由于双曲空间具有负曲率，点之间的加法需要使用莫比乌斯加法 (Möbius Addition)：

$$\mathbf{a} \oplus_{\mathbb{D}} \mathbf{b} = \frac{(1 + 2\langle \mathbf{a}, \mathbf{b} \rangle + \|\mathbf{b}\|^2)\mathbf{a} + (1 - \|\mathbf{a}\|^2)\mathbf{b}}{1 + 2\langle \mathbf{a}, \mathbf{b} \rangle + \|\mathbf{a}\|^2 \|\mathbf{b}\|^2} \tag{4.20}$$

于是，正向扩散的计算变为：

$$\mathbf{z}_t = (1 - \beta_t)\mathbf{z}_{t-1} \oplus_{\mathbb{D}} \sqrt{\beta_t}\mathbf{n}, \quad \mathbf{n} \sim N(0, I) \tag{4.21}$$

这个扩散过程确保了点云在双曲空间中的分布遵循双曲几何结构。

双曲空间中的反向推导

和传统扩散模型一样，为了从噪声恢复点云，我们需要学习一个适用于双曲空间的反向去噪模型。而由于我们需要对生成的类别有所要求，即保证生成点云属于目标类别 y，我们在反向去噪过程中引入了分类器 (下文持续学习任务中介绍) 引导。设类别分类器为：

$$C_\phi(\mathbf{z}) \in \mathbb{R}^K \tag{4.22}$$

其中，$C_\phi(\mathbf{z})$ 是 PointNet 或类似神经网络的输出，表示点云属于 K 个类别的概率分布。分类器引导去噪的目标是修改去噪方向，使得生成点云朝向目标类别优化。因此，我们使用分类器梯度引导 (Classifier Guidance) 的方式：

$$\mathbf{g}_\phi = \nabla_{\mathbf{z}} \log p(y|\mathbf{z}) \tag{4.23}$$

其中，$\mathbf{g}_\phi$ 表示类别 y 对应的分类器梯度。最终的引导去噪方向为：

$$\mathbf{z}_{t-1} = \boldsymbol{\mu}_\theta(\mathbf{z}_t, t) + \lambda \mathbf{g}_\phi + \sqrt{\tilde{\beta}_t}\mathbf{z}, \quad \mathbf{z} \sim N(0, I) \tag{4.24}$$

其中，λ 为控制分类器引导强度的超参数。

而在双曲空间中完成去噪后，我们需要将点云投影回三维欧几里得空间，由庞加莱球模型的对数映射，我们可以得到所需的生成样本：

$$\mathbf{x}_i = \frac{\mathbf{z}_i}{\|\mathbf{z}_i\|} \tanh^{-1}(\|\mathbf{z}_i\|) \tag{4.25}$$

双曲扩散生成的训练目标

最终训练目标类似公式仍基于变分下界 (VLB)，但由于引入类别分类器梯度引导，因此根据公式4.12和公式4.13，我们可以得到最终的 VLB 和简化的损失函数：

$$L_{\text{VLB}} = \mathbb{E}_q \left[\sum_{t=1}^{T} D_{\text{KL}}(q(\mathbf{z}_{t-1}|\mathbf{z}_t, \mathbf{z}_0) \parallel p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)) - \log p_\theta(\mathbf{z}_0) \right] \tag{4.26}$$

$$L_{\text{simple}} = \mathbb{E}_{t,\mathbf{z}_0,\boldsymbol{\epsilon}} \left[\|\boldsymbol{\epsilon} - \epsilon_\theta(\mathbf{z}_t, t) + \lambda \nabla_{\mathbf{z}_t} \log p(y|\mathbf{z}_t)\|^2 \right] \tag{4.27}$$

4.2　基于双曲空间的生成式重放点云持续学习

我们设计的模型架构包含在第三章工作的基础上改进的做下游任务的分类器和作为生成器的扩散模型两大部分组成。

4.2.1　持续学习任务设置

同第三章的工作类似，在点云分类的持续学习任务中，模型需要在一系列任务 T 上进行学习，同时避免灾难性遗忘。设定一个任务序列：

$$T = \{T_1, T_2, \ldots, T_N\} \tag{4.28}$$

每个任务 T_t 由数据集 D_t 组成：

$$D_t = \{(\mathbf{x}_1^t, y_1^t), (\mathbf{x}_2^t, y_2^t), \ldots, (\mathbf{x}_n^t, y_n^t)\} \tag{4.29}$$

其中，$\mathbf{x}^t$ 代表点云样本，y^t 代表其类别标签。

在持续学习过程中，模型仅能访问当前任务的数据 D_t，而无法直接访问过去任务的数据 $D_{1:t-1}$，这会导致模型在学习新任务时遗忘旧任务的知识。为了解决这一问题，我们引入本章介绍的基于双曲空间的生成式重放方法。

4.2.2　评估指标

为了衡量模型的持续学习性能 (包括分类器的分类能力以及生成器的生产能力)，我们采用以下主要评估指标：

整体准确率

模型在 T 轮任务后，对所有已学习任务的准确率计算为：

$$ACC = \frac{1}{T}\sum_{i=1}^{T} a_i \tag{4.30}$$

其中 a_i 是模型在第 i 任务上的测试准确率。

JSD

Jensen-Shannon Divergence(JSD) 衡量真实点云分布 P_r 和生成点云分布 P_g 之间的相似度，基于 KL 散度计算：

$$\mathrm{JSD}(P_r, P_g) = \frac{1}{2}D_{KL}(P_r \| M) + \frac{1}{2}D_{KL}(P_g \| M) \tag{4.31}$$

$$D_{KL}(P \| Q) = \sum_{x} P(x) \log \frac{P(x)}{Q(x)} \tag{4.32}$$

其中，$M = \frac{1}{2}(P_r + P_g)$ 为两个分布的平均值，$D_{KL}(P\|Q)$ 是 KL 散度。JSD 越小，表示生成的点云分布与真实数据分布越接近，生成质量越高；反之，则代表生成点云与真实点云严重偏离。

MMD

Minimum Matching Distance(MMD) 衡量真实点云分布 P_r 和生成点云分布 P_g 之间的最小匹配距离，用于评估形状相似性：

$$\mathrm{MMD}(P_r, P_g) = \frac{1}{|P_r|}\sum_{x \in P_r} \min_{y \in P_g} d(x, y) \tag{4.33}$$

其中，$d(x, y)$ 表示两点云样本的距离，可使用推土机 (EMD) 距离计算：

$$d_{\mathrm{EMD}}(P_r, P_g) = \min_{\phi: P_r \to P_g} \sum_{x \in P_r} \|x - \phi(x)\| \tag{4.34}$$

MMD 越小，表示生成样本与真实样本更相似，生成模型质量更高。

COV

Coverage(COV) 衡量生成点云覆盖了多少真实点云的分布，用于评估生成多样：

$$\mathrm{COV}(P_r, P_g) = \frac{|\{x \in P_r \mid \exists y \in P_g, d(x,y) < \tau\}|}{|P_r|} \tag{4.35}$$

其中，τ 是一个匹配阈值，用于判断生成样本是否匹配真实样本，$d(x,y)$ 同样可使用 EMD 计算。COV 越高，表示生成样本分布覆盖了更多的真实数据，多样性更好。

1-NNA

1-Nearest Neighbor Accuracy(1-NNA) 衡量生成点云和真实点云在特征空间的区分度，它基于最近邻分类，计算一个样本的最近邻是否来自相同的分布，评估多样性和过拟合情况。首先计算所有样本的最近邻 (1-NN)：

$$\mathrm{NN}(F(X)) = \arg\min_{Y \in P_F \setminus \{X\}} d(F(X), F(Y)) \tag{4.36}$$

其中，$F(X), F(Y)$ 表示点云的特征，$P_F = P_r \cup P_g$ 是所有样本的特征集合。然后计算 1-NNA 评分：

$$\text{1-NNA}(S_g, S_r) = \frac{\sum_{X \in S_g} \mathbb{I}[\mathrm{NN}(X) \in S_g] + \sum_{Y \in S_r} \mathbb{I}[\mathrm{NN}(Y) \in S_r]}{|S_g| + |S_r|} \tag{4.37}$$

其中，$\mathbb{I}$ 是指示函数：

$$\mathbb{I}(x) = \begin{cases} 1, & \text{如果最近邻样本属于相同分布} \\ 0, & \text{如果最近邻样本属于不同分布} \end{cases} \tag{4.38}$$

表 4.1　1-NNA 结果分析。

1-NNA 值	**解释**
≈ 50%	生成点云与真实点云不可区分，表示分布高度相似，质量较好。
> 50%	生成点云过拟合，意味着生成样本与真实数据的特征差异较大。
< 50%	生成点云质量较低，可能缺乏多样性，或偏离真实点云分布。

4.2.3　分类器和评分机制

对于分类器的选取，我们沿用章节三中的适合双曲空间的基于 PointNet 的点云分类器，其核心结构包括局部特征提取层、全局特征聚合层以及分类头层。分类器的目标是学习从点云 $\mathbf{x}$ 到类别标签 y 的映射：

$$p_\phi(y|\mathbf{x}) = \text{softmax}(C_\phi(\mathbf{x})) \tag{4.39}$$

其中，C_ϕ 代表分类器的网络参数，softmax 用于计算类别概率。

但与工作一不同的是，在这里我们不使用传统的旧样本缓存区，而是使用上面介绍的基于双曲空间的点云扩散模型生成旧样本，并通过分类器来引导扩散模型生成高质量的旧样本。在每轮任务 t 训练时，分类器 C_ϕ 会对扩散模型生成的旧样本进行监督，目标是优化生成器，使其生成的点云更符合旧任务的类别分布。

对于扩散生成过程 $p_\theta(\mathbf{z}_{t-1}|\mathbf{z}_t)$，如公式4.27所示，我们在反向去噪过程的损失函数中引入分类器梯度。

生成样本质量评估与筛选

虽然扩散模型本身在反向推导过程中添加了噪声，但生成样本实际上还是一定程度上由分类器引导生成的，生成样本的质量仍然可能受到分类器自身参数的影响。为了降低这一方面的影响，同时平衡生成效率与重放效果，我们设计了一种基于评分机制的筛选机制。该算法首先对生成的旧样本进行质量评估，然后根据预设的评分标准筛选出具有较高代表性的样本，从而过滤掉那些质量较低、代表性差的样本。这样做不仅提高了生成样本的整体质量，还能有效避免在重放过程中因大量低质量旧样本数据引发分类器过拟合的问题，从而增强模型在新旧任务之间的迁移和适应能力。

对于扩散模型生成的旧类别样本 $\mathbf{z}_{\text{gen}}$，我们使用分类器的交叉熵损失来衡量其质量：

$$L_{\text{CE}}(C_\phi(\mathbf{z}), y) = -\sum_i y_i \log p(y_i|\mathbf{z}) \tag{4.40}$$

其中，$C_\phi(\mathbf{z})$ 是分类器的输出，表示类别预测概率，$p(y_i|\mathbf{z})$ 是生成样本属于类别 y 的概率。由于交叉熵损失越小，表明分类器越确信该样本属于正确类别，因此我们

对交叉熵取负作为评分值：

$$S(\mathbf{z}) = -L_{\mathrm{CE}}(C_\phi(\mathbf{z}), y) \tag{4.41}$$

如算法2所示，对于每个旧类别，我们首先使用分类器引导的扩散模型生成 n 个样本，并计算其分类评分。重放样本的数量上限为 M，当样本集未满时，直接加入生成样本；当样本集已满时，对比当前样本评分与样本集中的最高评分，若当前样本评分更高，则直接丢弃，否则以一定概率 ρ 随机替换样本集中的样本，反之，替换评分最高的样本。

基于这一算法，对于每个旧任务类别，我们可以获得一组代表性比较强的生成样本，将其与新任务的样本结合，得到当前任务下的训练数据集。利用这一集合，我们对当前的生成器和分类器进行学习和“复习”。

4.2.4　生成模型在双曲空间下的全过程知识蒸馏

为了防止模型在持续学习中遗忘旧任务的去噪能力，我们不只是通过生成样本来引导模型，同时考虑进行知识蒸馏，让学生模型 ϵ_{θ_i} 学习前一任务的教师模型 $\epsilon_{\hat{\theta}_{i-1}}$ 的知识。

传统知识蒸馏

在标准知识蒸馏中，学生模型通过减小与教师模型的分布距离来学习，通常采用 KL 散度：

$$L_{\mathrm{KD}} = D_{\mathrm{KL}}\big(p_\theta(\mathbf{z}) \| p_{\theta^*}(\mathbf{z})\big) \tag{4.42}$$

然而，该方法仅对最终输出进行匹配，无法保证扩散模型在整个去噪过程中的稳定性；而扩散模型如公式4.24所示，是由若干个时间步反向生成，仅仅对最终输出进行蒸馏并不全面，需要考虑全过程都有覆盖的蒸馏方法。

双曲空间下的全过程蒸馏

首先，在欧几里得空间中，我们使用 L_2 范数进行蒸馏：

$$L_{\text{distill, Euclidean}} = \mathbb{E}_{\mathbf{z}_t}\left[\|\epsilon_{\theta_{t-1}}(\mathbf{z}_t, t) - \epsilon_{\theta_t}(\mathbf{z}_t, t)\|^2\right] \tag{4.43}$$

算法 2 基于样本评分的筛选算法

初始化： 生成样本数量 n，每个旧任务类别生成样本上限 M，替换概率 ρ，评分函数 $S(\mathbf{z})$，当前类别重放样本集 $\mathbf{Z}_{\text{replay}} = \emptyset$
for 每个旧类别 y **do**
  生成 n 个样本 $\mathbf{Z}_{\text{gen}} = \{\mathbf{z}_{\text{gen},i}\}_{i=1}^{n}$
  计算每个样本的评分 $S(\mathbf{z})$
  for 每个样本 $\mathbf{z} \in \mathbf{Z}_{\text{gen}}$ **do**
    if $|\mathbf{Z}_{\text{replay}}| < M$ **then**
      直接加入样本 $\mathbf{Z}_{\text{replay}} \leftarrow \mathbf{Z}_{\text{replay}} \cup \{\mathbf{z}\}$
    else
      获取样本集中最高评分 $S_{\max} = \max_{\mathbf{z}' \in \mathbf{Z}_{\text{replay}}} S(\mathbf{z}')$
      if $S(\mathbf{z}) > S_{\max}$ **then**
        丢弃当前样本
      else
        以概率 ρ 替换样本集中的一个随机样本
        以 $1-\rho$ 的概率替换评分最高的样本：
        $\mathbf{Z}_{\text{replay}} \leftarrow \mathbf{Z}_{\text{replay}} \setminus \{\arg\max_{\mathbf{z}' \in \mathbf{Z}_{\text{replay}}} S(\mathbf{z}')\} \cup \{\mathbf{z}\}$
      end if
    end if
  end for
end for
返回： 过滤后的重放样本集 $\mathbf{Z}_{\text{replay}}$

但在双曲空间中，数据点的几何关系更加复杂，因此不能简单使用欧氏距离，而应该采用双曲空间度量。在庞加莱球模型中，两个点 $\mathbf{z}_1, \mathbf{z}_2$ 之间的双曲距离定义为：

$$d_{\mathbb{D}}(\mathbf{z}_1, \mathbf{z}_2) = \operatorname{arcosh}\left(1 + 2\frac{\|\mathbf{z}_1 - \mathbf{z}_2\|^2}{(1 - \|\mathbf{z}_1\|^2)(1 - \|\mathbf{z}_2\|^2)}\right) \tag{4.44}$$

在全过程蒸馏中，我们希望让新任务模型的去噪预测 ϵ_{θ_t} 与旧任务模型的去噪预测 $\epsilon_{\theta_{t-1}}$ 在双曲度量下尽量趋于一致，因此可以得到蒸馏损失：

$$L_{\text{distill, hyperbolic}} = \mathbb{E}_{\mathbf{z}_t}\left[d_{\mathbb{D}}\left(\epsilon_{\theta_{t-1}}(\mathbf{z}_t, t), \epsilon_{\theta_t}(\mathbf{z}_t, t)\right)\right] \tag{4.45}$$

而为了使得这一蒸馏操作对整个反向推导过程都有覆盖，同时兼顾效率，我们设计了相应的全过程蒸馏算法，如算法3所示。

最终，我们的生成模型的损失函数定义如下：

$$L = L_{\text{simple, D}} + \lambda L_{\text{simple, M}} + \mu L_{\text{distill, hyperbolic}} \tag{4.46}$$

其中，$L_{\text{simple, D}}$ 代表当前任务样本的损失，$L_{\text{simple, M}}$ 代表重放样本的损失，λ 和 μ 代表重放和蒸馏系数。

算法 3 双曲空间下的全过程蒸馏

1: 旧任务模型 $\epsilon_{\theta_{t-1}}$，新任务模型 ϵ_{θ_t}，时间步数 T
2: **for** 每个重放样本 $x_0 \sim \mathbf{Z}_{\text{replay}}$ **do**
3: 　从时间步长 $T,\ldots,1$ 中随机采样覆盖步 t
4: 　生成加噪样本：$x_t = \sqrt{\alpha_t}x_0 + \sqrt{1-\alpha_t}\epsilon$
5: 　计算去噪预测：$\epsilon_{\theta_t}(x_t,t) = f_{\theta_t}(x_t,t)\epsilon_{\theta_{t-1}}(x_t,t) = f_{\theta_{t-1}}(x_t,t)$
6: 　计算蒸馏损失 $L_{\text{distill, hyperbolic}}$
7: 　计算去噪损失 L_{simple}
8: 　计算总损失 L
9: 　更新模型参数
10: **end for**

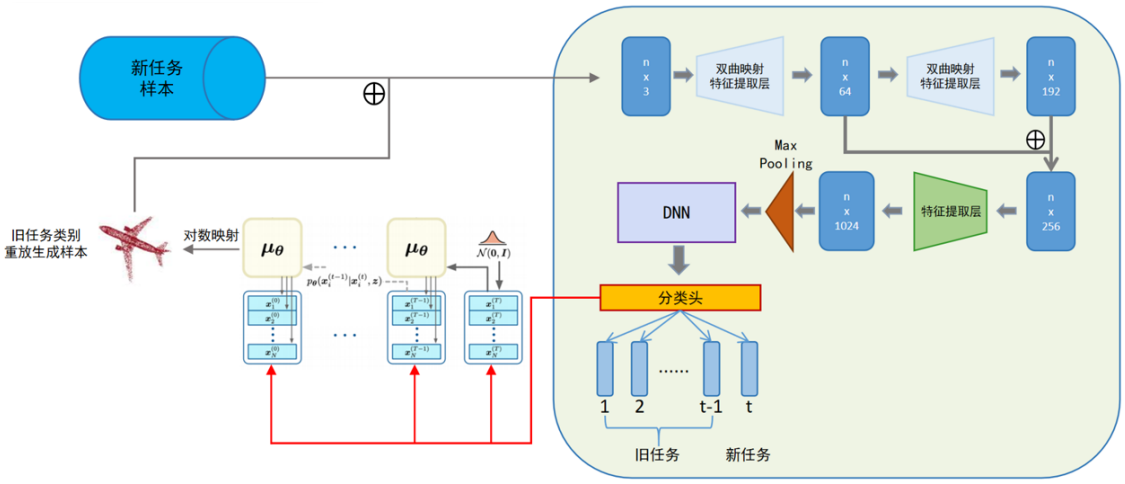

图 4.1　HyperDiffIL 模型架构。

4.3　实验及分析

为了和第三章工作进行对比实验以评价模型的分类精度，我们在实验部分的大部分超参数尽可能维持一致，并且所有实验同样在两块 NVIDIA Tesla V100 GPU 上进行，每块 GPU 具有 32GB 显存，代码主要使用 PyTorch 库实现。并且，由于本工作采用了生成式重放，因此我们还对生成模型自身的持续学习能力与目前主流的一些方法做了对比。

4.3.1　实验设置

如前文所述，本章主要聚焦于类增量任务，同样采用 ShapeNet 和 ModelNet 两个广泛用于点云分类任务的数据集每个任务涉及两个新类别，整个训练过程包括 20 轮任务。每个点云样本使用 FPS 采样 2048 个点作为输入，并在训练时进行数据增强，包括随机旋转、缩放和高斯噪声扰动，并最后做归一化处理。在此基础上，我们引入 ScanNet[97] 数据集，由于其提供的类别数量较少，因此在保证其他设置相同的情况下，总共进行 8 轮任务的训练。而由于 ScanNet 不是专门用于 3D 物体分类，所以我们通过提取一些具有较多点的类别的对象级点云，每个点云通过上、下采样的方式维持同样的 2048 个点，并转化为类似 ModelNet 的格式进行预处理。所有数据集的训练集、验证集和测试集比例设置为 70%、10% 和 20%，进行类增量分类任务的实验。

另一方面，由于 HyperDiffIL 主要采用了基于生成式重放的方式来对抗灾难性遗忘，性能也相当依赖于生成模型的生成能力，因此我们也对生成器的性能与常见的一些相关方法在 ShapeNet 和 ModelNet40 上进行了对比实验。

为了防止遗忘，每个旧类别的重放样本数量设为 128，择优选取率为 50%，即每轮任务前，扩散模型针对每个旧类别生成 256 个候选样本，最终根据评分算法选取 128 个样本加入重放集合。学习率设置为 0.003，使用 Adam 优化器，批量大小由于显存限制设置为 8，每个任务训练的迭代轮数上限设置为 50 (Epoch)。

框架方面，对于生成器，我们采用在双曲空间下基于类 U-Net 结构的扩散模型；对于分类器，我们在网络输入处添加了一个双曲投影层，并支持模型由第二层进行输入，同时舍弃了双曲特征投影层中保留的欧式空间信息，调整了神经元的

分配，以更好适配对生成模型的引导。

4.3.2　数据集介绍

图 4.2　ScanNet 数据集预览。

ModelNet 和 ShapeNet 数据集已于第三章中简单介绍，这里我们主要介绍 ScanNet 数据集。ScanNet[97] 是一个专门用于三维室内场景理解的大规模数据集，旨在推动计算机视觉和人工智能领域在三维感知方面的发展。该数据集通过 RGB-D 传感器采集真实世界的室内环境，涵盖了各种复杂的室内布局，如卧室、客厅、办公室和教室等，并提供了丰富的标注信息，目前被广泛用于三维目标检测、语义分割、实例分割、场景重建、对象识别等任务，并成为许多先进算法的实验基准。

4.3.3　结果分析

对比实验

如图4.3、4.4、4.5以及表4.2所示，我们选取了第三章的模型 HyMR，以及实验结果当中较为突出的两个模型 I3DOL 和 RPS-Net 与我们的 HyperDiffIL 方法进行对比实验，测试了不同方法在 ModelNet40、ShapeNet 和 ScanNet 数据集上的持续

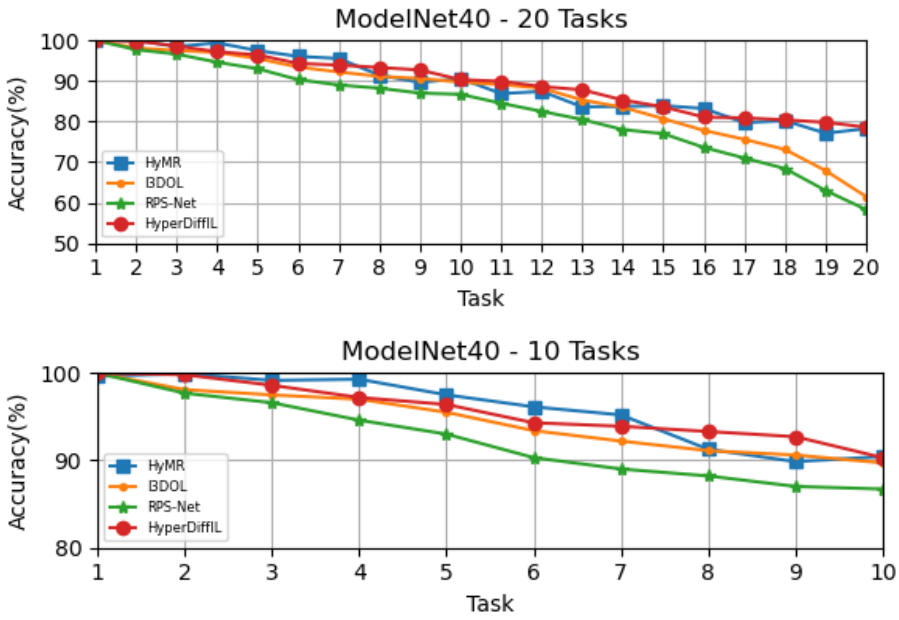

图 4.3　类增量学习设置下，在完成若干个任务后在 ModelNet40 数据集上的整体平均准确率。

表 4.2　在类别增量学习中，不同方法在 ModelNet40、ShapeNet 以及 ScanNet 数据集上经过若干轮训练后的平均准确率 (%)。

数据集	任务数	顺序微调	RPS-Net[96]	I3DOL[19]	HyMR(Ours-缓存式)	HyperDiffIL(Ours-生成式)
ModelNet40	10	13.3	86.7	89.7	**90.3**	90.2
	20	2.1	58.3	61.5	78.0	**78.6**
ShapeNet	10	9.7	86.9	87.8	89.8	**90.7**
	20	1.3	69.5	74.1	80.8	**80.9**
ScanNet	8	1.8	53.3	56.8	**65.9**	64.4

学习分类性能，可以看到所有方法的准确率随着任务数增加逐渐下降，表现出灾难性遗忘的现象。

在 ModelNet40 数据集上，HyperDiffIL 和 HyMR 方法在整个训练过程中准确率曲线较为接近，并且保持了较高的准确率，尤其是 HyperDiffIL，在经过 20 轮任务后，最终准确率达到了 78.6%，明显优于 RPS-Net(58.3%) 和 I3DOL(61.5%)，说明 HyperDiffIL 在长时间序列任务上的抗遗忘能力更强。虽然 HyMR 方法在经过完整任务训练后的最终准确率略低于 HyperDiffIL，但前 10 任务的总体表现区别不大，甚至略优。在 ShapeNet 数据集上，总体趋势和 ModelNet40 区别不大，并且在

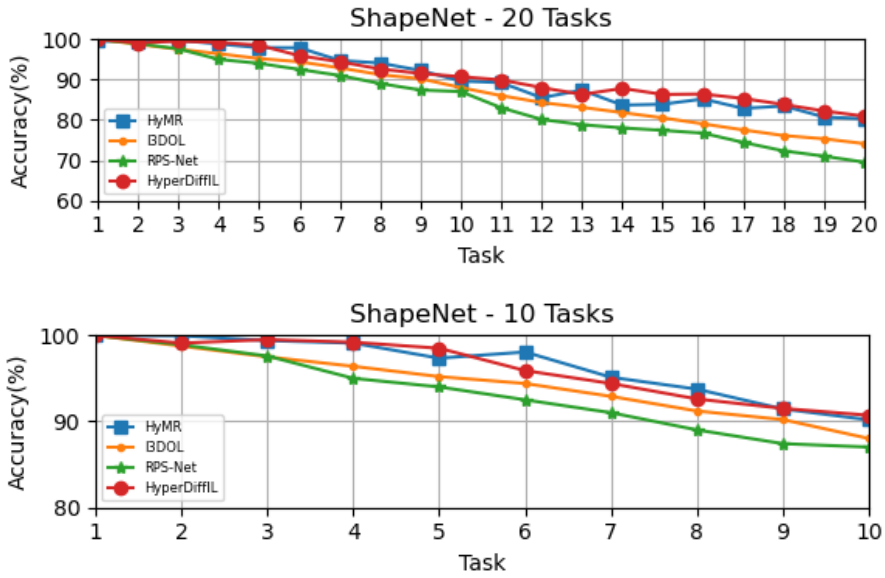

图 4.4　类增量学习设置下，在完成若干个任务后在 ShapeNet 数据集上的整体平均准确率。

经历 10 轮任务后，HyperDiffIL 方法的平均准确率取得了最好的成绩，但是从前 10 轮任务结果趋势来看，HyMR 依旧保持更平滑的水准；而在经过 20 轮任务训练后，HyperDiffIL 优势又重新回归。因此，HyperDiffIL 方法在 ModelNet 和 ShapeNet 上的长任务序列表现最佳，随着任务增多，虽然生成式样本的质量可能下降，但由于不像 HyMR 受限于重放缓存区，HyperDiffIL 凭借多样化的重放内容，仍然保持更好的抗遗忘能力。

而在 ScanNet 数据集上，由于其本身为 3D 真实场景数据，主要用于语义分割，应用到分类任务需要重构，重构后的质量可能不如前两个结构化的合成数据集，因此所有方法在 ScanNet 上经历相同数量任务后的准确率都比 ShapeNet 和 ModelNet40 低。而从整体趋势和最终准确率来看，HyMR 均表现最佳，说明针对这种短任务序列的场景，HyMR 这样精确的重放方法依旧保持优势。

在生成能力的评估上，我们在 ShapeNet 上按类别汽车、椅子、飞机、桌子的顺序，进行单类别任务的类增量训练。我们选取 HyperCloud[98] 作为基准模型顺序微调，并使用非持续学习的联合训练 (即将所有类别一次性训练) 作为参照。为了进一步验证我们提出方法的有效性，我们还选取了近年来常见的、可应用于点

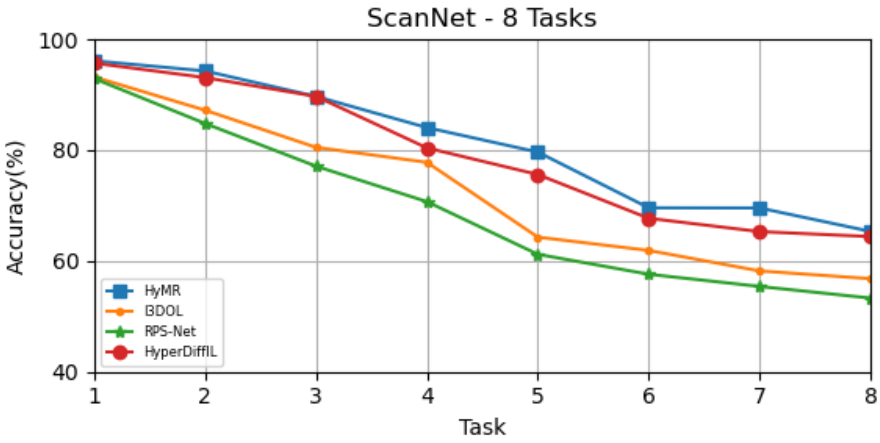

图 4.5　类增量学习设置下，在完成若干个任务后在 ScanNet 数据集上的整体平均准确率。

云生成任务的持续学习方法进行比较。此外，我们加入 L_2 正则化以及固定缓存区大小 (500) 的精准重放 (Exact replay) 作为对比。

在经历了四轮任务训练后，我们获得了如表4.3及图4.6所示的用于评估模型生成能力的各项指标数据及部分方法的生成点云可视化展示。其中，HyperDiffIL 的平均 JSD 值为 0.103，仅次于 PNN，并且与联合训练后的性能相差不大，这表明 HyperDiffIL 生成的点云分布与真实点云分布较为接近，优于大多数持续学习方法；在最小匹配距离上表现不佳，仍有一定优化空间，可能由于生成器基于双曲空间的扩散推导，而忽略了欧氏空间下的几何匹配能力；而在覆盖率和最近邻准确率分析中，HyperDiffIL 在所有持续学习任务方法中表现最佳，生成器更好地覆盖了真实的点云分布，并且与真实数据的可分性更低。

消融实验

为了分析 HyperDiffIL 关键模块对分类性能和生成质量的影响。我们设计了一系列消融实验，通过去除模型中的核心组件或者修改相关参数值，以验证每个模块在生成质量、多样性和抗遗忘能力方面的贡献。消融实验主要围绕双曲空间投影、分类器引导扩散、全过程知识蒸馏以及样本评分机制这四个模块进行，分类任务数据集采用针对类别相关任务质量较高的 ShapeNet 和 ModelNet40，生成任务数据集采用和之前一致的 ShapeNet。

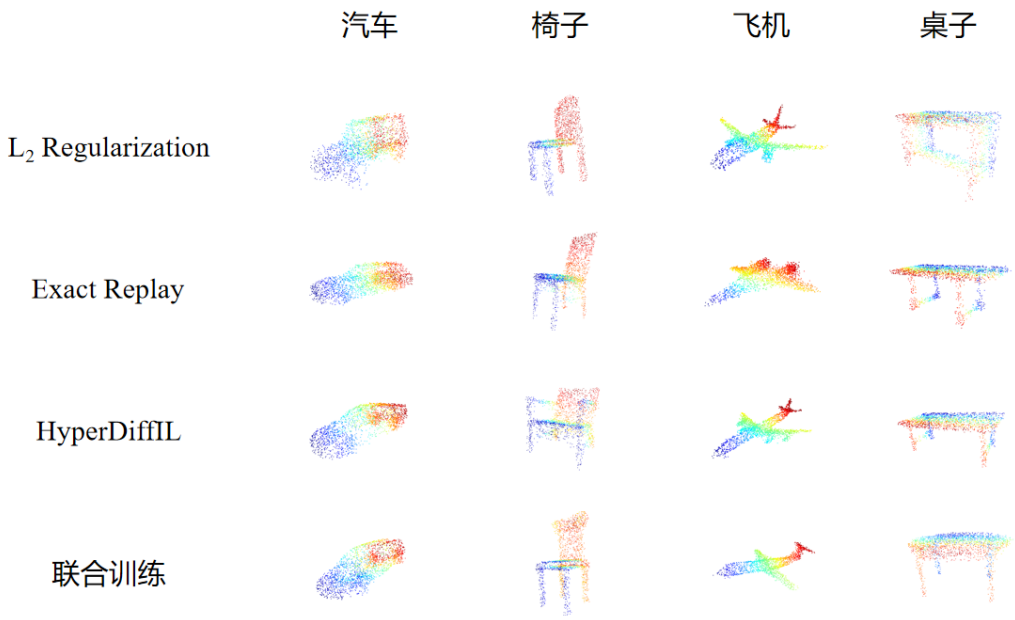

图 4.6　经过四轮单类别增量任务后，不同用于生成模型的持续学习方法的生成结果可视化对比。

对于双曲空间投影，我们取消了模型框架中的双曲指数映射模块以及分类器中的双曲特征提取层，并在欧氏空间中进行生成器中的扩散过程，遵循公式4.13作为训练目标，进行分类和生成任务。针对分类器引导扩散模块，我们在每次新任务重放过程中，取消了公式4.27引入的梯度损失，让生成器随机生成旧样本重放，生成的类别由分类器自主判断，保证生成样本总数与原方法一致。而对生成器的全过程知识蒸馏，我们采取不蒸馏和仅对最终步进行蒸馏作为对照。最后，由于样本评分对重放样本做了筛选，并且设置了选取上限，因此我们依次进行了不评分、不同样本淘汰率、不同样本类别数上限的对比。

首先如图4.7和4.8所示，我们将完整模型和去除双曲映射或分类器模块的模型进行了对比。在去除双曲空间投影后，处理前期任务时的整体平均准确率没有太大的影响，但是随着中期任务和后期任务的到来，欧式空间难以很好表征点云结构的劣势显现出来，双曲空间在长期任务中能够更好地表示点云特征，提升抗遗忘能力。而在去除了分类器的引导模块后，由于任务训练初始阶段的类别数目并不多，

表 4.3　经过四轮单类别增量任务后，不同用于生成模型的持续学习方法的生成能力指标对比。

方法	JSD(→0)	MMD(→0)	COV%(→100%)	1-NNA%(→50%)
HyperCloud[98]	0.353	16.7	22.7	80.5
EWC[17]	0.288	13.9	27.0	88.6
Generative replay[33]	0.192	15.1	48.4	72.3
L_2 regularization	0.286	14.4	26.3	93.9
Exact replay	0.181	**11.6**	39.8	77.1
PNN[82]	**0.095**	12.2	48.4	83.8
HyperDiffIL(Ours)	0.103	13.7	**49.1**	**69.8**
联合训练	0.080	10.6	53.5	65.0

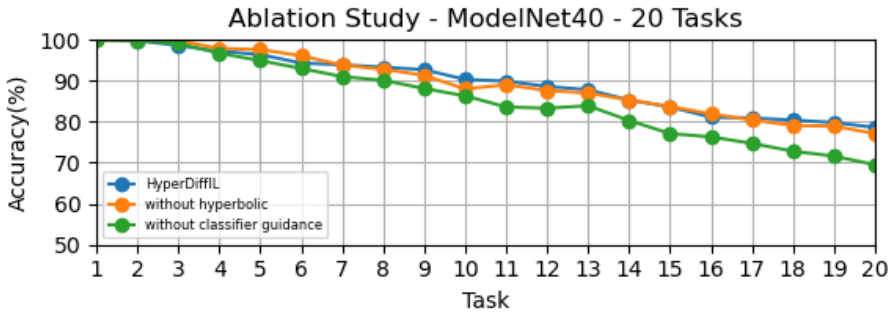

图 4.7　消融实验-双曲映射和分类器引导在 ModelNet40 数据集上完成若干个任务后类增量任务的整体平均准确率。

单靠生成模型随机生成和分类器识别类别的形式足以应对，但随着类别数目的增多，在没有分类器引导的情况下，生成器容易出现一定的生成类别倾向，导致对另外的类别识别性能下降，并且错误分类概率增加，从而出现较大的准确率下滑。

全过程蒸馏和评分机制模块对模型的影响则如表4.4所示。当把生成模型的蒸馏模块去除后，扩散模型在新任务中仅仅依靠自身生成样本重放来对抗遗忘，而生成的内容又很大程度上影响了生成器自身和分类器的效果，因此准确率出现了下降，尤其是在经历全部 20 轮任务后，掉点较为明显；只进行最终步蒸馏，依旧能保持部分性能，但仍不及全过程蒸馏，说明去噪过程中所有时间步的知识一致性对生成质量至关重要。不对生成样本进行质量筛选，所有样本均用于重放，最终的

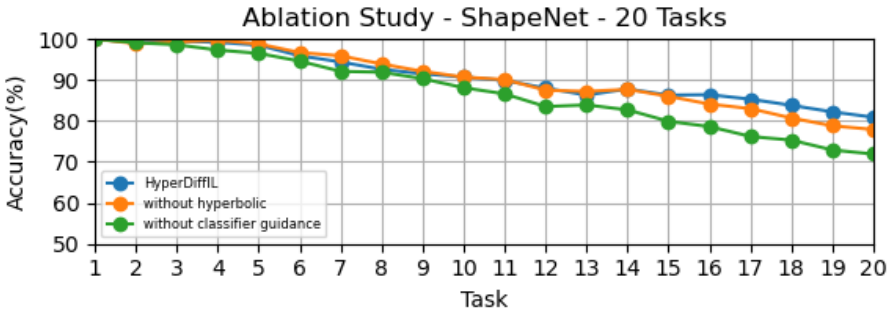

图 4.8　消融实验-双曲映射和分类器引导在 ShapeNet 数据集上完成若干个任务后类增量任务的整体平均准确率。

结果虽然变化不大，但仍有略微下跌，这说明样本评分机制虽然影响较小，但仍能提供一定的质量提升；而从调整了淘汰率和样本上限之后的结果可以看出，合理的样本筛选可以提升旧任务样本的质量，当提升淘汰率，在一定范围内可以提高整体的准确率，提升样本上限和传统缓存区方法一样，同样意味着更多的旧知识的回顾，但是过高的淘汰率和样本上限意味着更长的生成时间和更多的缓存，影响效率的同时还容易陷入低质量样本的陷阱。

由于生成质量是生成式重放模型性能的主要影响源，我们分析了 HyperDiffIL 的扩散模型生成器在不同消融方案下的生成能力，得到如表4.5所示的结果。从各项指标随着不同方案的变化可以看出，双曲空间投影对整体生成质量 (JSD 和 COV) 有积极作用，但对几何匹配 (MMD) 的影响相对较小，可能是由于实验中采用的是最终生成的在欧式空间下的点云进行评估所致，因此双曲映射的主要作用在于增强点云层级结构的表示能力，而不是直接影响点云几何形状匹配；去除蒸馏后，生成样本偏离真实数据分布，导致 1-NN 分类器更容易区分生成样本和真实样本，而针对扩散模型的全过程蒸馏有助于生成样本保持一致性，提高旧类别样本的再现质量，但若仅在最终步骤进行蒸馏虽有一定效果，但不足以完全避免生成样本的退化；评分机制可以提升多样性，但对生成分布影响较小，并且过高的淘汰率在陷入低质量样本的怪圈后，可能导致模式崩溃；而更大的单类别样本上限可以明显提升生成样本的多样性，并减少几何匹配误差，但同样也意味着更高的时间消耗导致的效率低下。

表 4.4　在 ModelNet40 和 ShapeNet 数据集上完成若干轮任务后类增量任务的整体平均准确率受全过程蒸馏和样本评分机制的影响。

数据集	消融方案	准确率 (10 轮任务)	准确率 (20 轮任务)
ModelNet40	HyperDiffIL	90.3	78.6
	不蒸馏	87.63	68.2
	仅最终步蒸馏	91.2	76.0
	不评分筛选	90.1	77.9
	25% 淘汰率	88.7	74.1
	75% 淘汰率	90.8	80.1
	64 样本上限	90.2	78.8
	256 样本上限	**91.4**	**80.3**
ShapeNet	HyperDiffIL	90.7	80.9
	不蒸馏	84.4	74.5
	仅最终步蒸馏	90.0	80.9
	不评分筛选	90.3	80.7
	25% 淘汰率	90.0	81.1
	75% 淘汰率	**90.8**	80.3
	64 样本上限	89.1	74.7
	256 样本上限	90.5	**82.2**

4.4　章节小结

本章提出了一种基于生成式重放和双曲映射的点云持续学习方法 HyperDiffIL，结合扩散模型的高质量生成能力以提升旧任务知识的保持效果，并且避免了传统缓存区方法的存储限制，并在数据多样性、隐私保护等方面具有适用性。我们的实验结果表明，HyperDiffIL 在点云分类任务中表现良好，特别是在长任务序列下的分类精度和生成质量方面具有明显优势，能够达到与 HyMR 相当甚至更优的分类准确率，但在较短任务序列中，其分类性能略低于 HyMR 这样基于缓存区的精准重放方法。生成质量评估方面，HyperDiffIL 在部分指标上表现优越，表明其生成

表 4.5　各种消融方案对 HyperDiffIL 生成器性能的影响。

消融方案	JSD(→0)	MMD(→0)	COV%(→100%)	1-NNA%(→50%)
HyperDiffIL	0.103	13.7	49.1	69.8
无双曲映射	0.111	**13.2**	48.7	69.9
不蒸馏	0.188	14.3	44.6	71.6
仅最终步蒸馏	0.129	13.9	50.3	72.8
不评分筛选	**0.101**	13.5	47.6	72.8
25% 淘汰率	0.109	13.9	48.8	70.3
75% 淘汰率	0.125	14.3	51.7	**68.4**
64 样本上限	0.131	15.6	47.8	73.9
256 样本上限	0.107	13.6	**52.3**	72.4
联合训练	0.080	10.6	53.5	65.0

样本与真实数据分布较为接近，且具有较高的多样性。此外，我们进行的一系列消融实验证明了双曲空间投影、分类器引导扩散、全过程知识蒸馏、样本评分机制等核心模块对模型的贡献。当然，HyperDiffIL 依旧存在一些缺陷，比如扩散模型计算成本较高，并且针对生成器的抗遗忘控制较少。在未来，我们的研究可以进一步优化生成模型、增强长序列任务的抗遗忘能力，并拓展至多模态任务，以提高其适用性和实际应用价值。