中国科学: 信息科学 2024 年 第 54 卷 第 10 期: 2428–2442
SCIENTIA SINICA Informationis
论文

基于原型学习的联邦持续学习方法

张浩东[1], 杨柳[1*], 于剑[2], 胡清华[1], 景丽萍[2]

1. 天津大学智能与计算学部, 天津 300354
2. 北京交通大学计算机与信息技术学院, 北京 100044
* 通信作者. E-mail: yangliuyl@tju.edu.cn

收稿日期: 2023–08–12; 修回日期: 2024–01–09; 接受日期: 2024–07–30; 网络出版日期: 2024–09–29

国家自然科学基金 (批准号: 62076179, 61925602, U23B2049, U23B2062) 资助项目

摘要 联邦学习能够在隐私保护的前提下联合多个参与者进行协同学习, 然而经典的联邦学习不具备持续学习的能力, 无法适应动态变化的应用场景. 联邦持续学习近期引起了广泛的关注, 其允许多个参与者在协同学习的同时进行持续学习. 联邦持续学习是一种更加复杂的学习场景, 其面临的挑战包括灾难性遗忘, 异构性以及通信资源受限. 为了应对这些挑战, 本文提出一种基于原型学习的联邦持续学习方法. 该方法利用原型进行知识的共享, 提升通信效率的同时增强了对模型异构性的适应能力. 此外, 该方法设计了基于知识蒸馏和回放的灾难性遗忘的预防机制. 本文提供了所提出方法的收敛性分析, 并且通过对比实验和消融实验验证了该方法的有效性.

关键词 联邦持续学习, 原型学习, 知识蒸馏, 灾难性遗忘, 数据异构

1 引言

联邦学习 [1~5] 是一种分布式的机器学习范式, 其能在隐私敏感的场景下协调多个客户端联合学习. 联邦学习中通常有多个客户端以及一个中心服务器, 各个客户端在本地存储任务相关的原始数据, 但是由于数据隐私的限制, 原始数据无法共享. 经典的 FedAvg [1] 通过聚合各个客户端本地模型的参数来共享知识, 从而不需要接触客户端的数据, 保护了隐私. 由于联邦学习对数据隐私保护的特性, 其应用场景非常广泛, 包括金融 [6]、医疗 [7] 和物联网 [8] 等.

经典的联邦学习系统是静态的, 即在系统初始化时就需要明确任务定义, 后续无法改变. 然而在大多数应用场景中, 任务是动态变化且无法预知的, 经典的联邦学习不具备持续学习的能力, 所以无法适应动态的任务场景, 这限制了模型的可拓展性. 联邦持续学习 [9] 的提出就是为了解决传统联邦学习的缺点, 使得多个客户端联合学习的同时能够进行持续学习, 不断适应新的任务. 在联邦持续学习中, 每个客户端拥有本地私有的任务序列, 客户端在本地进行持续学习, 不断更新模型; 同时客户端之间

引用格式: 张浩东, 杨柳, 于剑, 等. 基于原型学习的联邦持续学习方法. 中国科学: 信息科学, 2024, 54: 2428–2442, doi: 10.1360/SSI-2023-0239
Zhang H-D, Yang L, Yu J, et al. Federated continual learning based on prototype learning (in Chinese). Sci Sin Inform, 2024, 54: 2428–2442, doi: 10.1360/SSI-2023-0239

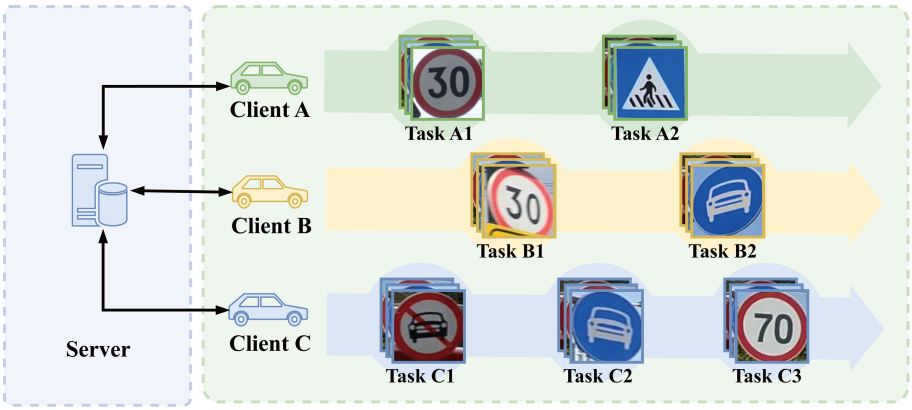

图 1 (网络版彩图) 联邦持续学习应用示例

Figure 1 (Color online) An application of federated continual learning

通过服务器共享知识, 从其他客户端的知识中进行学习. 图 1 中展示了联邦持续学习的一个应用场景, 其任务目标是在车联网上训练交通标识符识别模型, 每一个客户端均为一台车辆, 客户端本地不断采集新的交通标识符数据并进行持续学习, 同时客户端之间通过中心服务器交换各自学习到的新的数据类别, 从而增加模型可识别的标识符类别.

联邦持续学习面临的核心挑战包括如下内容. (1) 灾难性遗忘: 多个客户端在持续的学习新任务时会产生对旧任务的遗忘, 其导致模型在旧任务上的性能急剧下降. (2) 通信开销大: 联邦持续学习相较于联邦学习需要更多的通信次数, 因而存在更大的通信开销. (3) 异构性: 异构性包括数据异构和模型异构, 数据异构指各客户端数据分布的差异性, 会导致全局模型在本地性能表现差; 而模型异构指客户端硬件资源约束导致的可运行模型规模的不同, 其会阻碍客户端之间的知识交换.

现有研究聚焦于解决联邦持续学习中的灾难性遗忘, 通过参数隔离 [9]、正则化 [10] 和回放 [11] 等方法解决模型对旧任务的遗忘. 而这些方法的局限性表现在对通信开销和异构性问题的忽视. 在通信开销上, 现有方法 [9~13] 将模型参数作为知识的载体, 客户端和服务端之间的通信会带来巨大的通信开销. 在异构性问题上, 已有方法 [9~13] 采用基于参数聚合的知识融合方法, 且学习目标为单一的全局模型. 单一的全局模型难以解决数据异构性问题, 其难以在数据分布不同的客户端上均取得较好的性能表现, 而基于参数聚合的方法则要求各客户端模型同构, 无法应用于模型异构的场景.

针对现有研究的局限性, 本文提出了一种基于原型学习的联邦持续学习方法, 主要贡献如下. (1) 通过引入原型学习解决了已有方法在通信开销和异构性上的局限性. 原型学习中仅需要传递特征向量而不需要传递模型参数, 因此有效减少了通信数据量, 提升了通信效率. 原型学习天然的适应模型异构场景, 其仅要求各个客户端的特征空间维度相同, 而对模型结构没有要求. 原型学习中客户端本地学习个性化模型, 其相比统一的全局模型能更好的解决数据异构性问题, 可以学习到更适应客户端本地任务的参数. (2) 提出了适应动态场景的原型学习方法. 经典联邦学习是静态的, 原型学习中原型数量是固定不变的, 且已学习的原型不会再进行更新, 这使其无法适应联邦持续学习中的动态场景. 为了使原型学习能够适用于动态场景, 本文方法通过更新全局的原型库来不断学习新的原型; 此外, 客户端本地保留学习过类别的代表性样本, 利用其对已学习原型进行持续更新, 使其适应新的数据分布. (3) 设计了基于知识蒸馏和回放的灾难性遗忘的预防机制. 为解决联邦持续学习中的灾难性遗忘问题, 本文方法结合模型的输入和输出维度进行遗忘预防的设计以获得更好的模型性能. 在模型输入上, 利用基于数据的回放方法保留旧任务代表性样本, 在学习新任务时加入旧样本以缓解遗忘. 在

模型输出上, 利用基于知识的知识蒸馏将旧模型中的知识迁移到新模型中, 以维持模型中关于旧任务的知识. (4) 本文提供了方法的收敛性分析以及对蒸馏温度的分析, 并通过消融实验和对比实验验证了方法的有效性.

本文按以下结构组织: 第 2 节首先介绍了相关工作. 第 3 节介绍了基于原型学习的联邦持续学习方法. 第 4 节提供了收敛性分析以及对蒸馏温度的分析. 第 5 节提供了对本文方法的实验验证.

2 相关工作

2.1 联邦学习

联邦学习 (federated learning) 是一种由多个参与者在中心服务器的协调下合作训练同一模型的机器学习方法 [1]. McMahan 等 [1] 首次定义联邦学习这一问题, 并且提出基于参数平均的联邦学习方法. 该方法通过调整客户端本地迭代数量而减少全局通信次数, 减少了通信开销. Li 等 [2] 对异构场景的联邦学习进行优化, 通过在客户端本地的目标函数中引入正则项约束本地和全局模型的差异性来避免异构场景中的过拟合问题. 为了进一步解决联邦学习过程中的异构性问题, Collins 等 [3] 采用了个性化联邦学习的方法, 多个客户端共同训练一个共享的特征提取器, 同时各自在本地训练个性化的分类器. Ni 等 [14] 则将参数分解和正则化约束引入个性化联邦学习, 在时间和空间维度同时解决数据异构性问题. Tan 等 [4] 将原型学习 (Prototype Learning) 引入联邦学习, 利用原型传递知识, 提升通信效率的同时增强了方法对模型异构场景的适应性. Li 等 [5] 将知识蒸馏引入联邦学习, 能够在模型异构场景中实现客户端之间知识的共享.

2.2 持续学习

持续学习 (continual learning) 的目标是使人工智能系统可以持续不断的从新数据学习新知识, 同时保留已经学习到的知识 [15]. 持续学习的主要挑战是灾难性遗忘, 即模型在学习新任务之后, 几乎彻底遗忘掉之前学习过的知识. Li 等 [15] 将知识蒸馏 [16] 引入持续学习, 将旧模型的知识迁移至新模型中以减轻知识的遗忘. Dhar 等 [17] 则进一步提出基于注意力机制的知识蒸馏以更好的保护旧任务的知识. Rebuffi 等 [18] 在知识蒸馏的基础上增加了回放机制, 保留少量旧任务的代表性样本, 在新任务学习中加入. Castro 等 [19] 增加了额外的训练阶段, 构造类均衡的数据集以纠正模型在学习新任务中对旧任务产生的偏差. 持续学习方法也受到了很多生物学的启发, Kirkpatrick [20] 模拟生物神经网络中突触的作用机制, 利用 Fisher 信息矩阵评价参数的重要程度, 利用正则项减轻灾难性遗忘. Zenke [21] 则模拟突触的作用机制, 利用网络参数在时间维度的变化信息来评价参数的重要程度.

2.3 联邦持续学习

联邦持续学习 (federated continual learning) 中存在多个客户端, 每个客户端不断获取新的数据. 由于存储空间的限制, 客户端以持续学习的方式进行模型更新, 同时客户端之间通过共享知识提升本地模型性能. Yoon 等 [9] 最早给出了联邦持续学习的定义, 其提出的方法将模型参数拆分为通用参数和任务特定参数, 客户端之间共享通用参数, 且通过注意力机制选择性的学习其他客户端上对自身有用的任务特定参数, 后续有其他工作 [11, 22, 23] 基于该模型开展. 客户端可以从本地数据直接获取知识, 也可以间接的利用其他客户端学习到的知识提升本地模型性能. Dong 等 [10] 利用带权重的交叉熵损失平衡客户端本地新旧任务间的遗忘, 利用知识蒸馏将其他客户端的知识迁移至本地. Ma 等 [12] 则在客户端和服务端同时使用知识蒸馏以解决灾难性遗忘, 并引入公共数据集用于模型蒸馏, 同样采用知

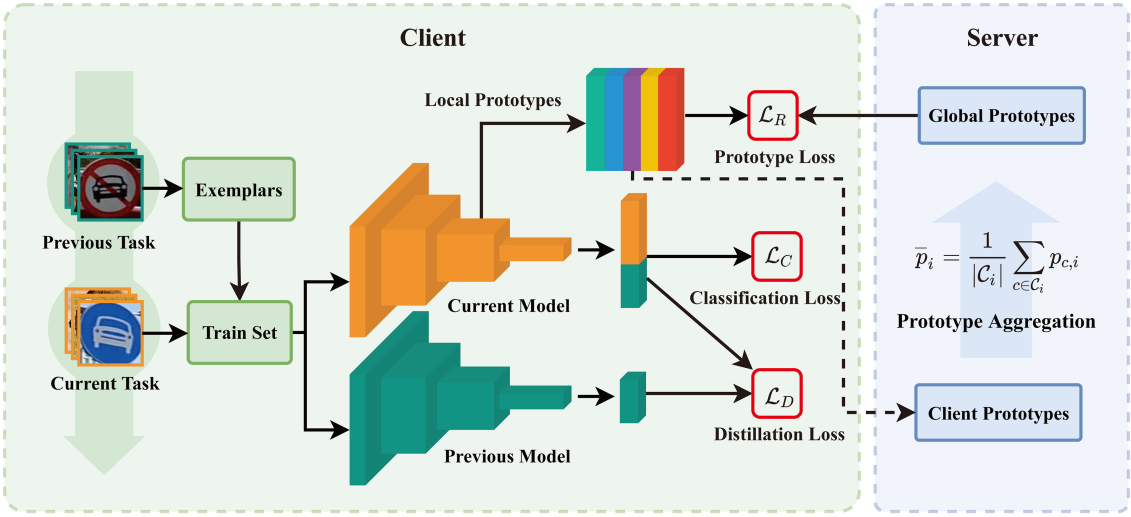

图 2 (网络版彩图) 基于原型学习的联邦持续学习方法

Figure 2 (Color online) Federated continual learning based on prototype learning

识蒸馏的方法包括 [24,25]. Hendryx [13] 则利用预训练的特征提取器提取各个客户端的知识, 并且设计了一种联邦持续学习的知识更新机制. 对联邦持续学习的研究已经在多个领域开展, 包括金融 [26], 移动边缘计算 [27], 车联网 [28] 和物联网 [29].

2.4 问题设置

联邦持续学习中包括 C 个客户端和一个负责协调训练的服务端. 每个客户端在其私有的任务序列 $\{\mathcal{D}_{c,t}\}_{t=1}^{T_c}$ 进行持续学习, T_c 表示客户端 c 的任务数量, 其在真实场景中无法提前确定, 因为客户端会不断地采集数据. 本文研究的问题为类增量学习, 客户端的任务之间没有类别的重叠, 推理过程中需要在未知样本所属任务的情况下预测其类别. 客户端 c 上的任意一个任务 t 可以表示为 $\mathcal{D}_{c,t} = \{(\boldsymbol{x}_{c,t}^i, y_{c,t}^i)\}_{i=1}^{N_{c,t}}$, 其是一个有监督数据集, 包括 $N_{c,t}$ 个样本. 每个样本都包含数据 $\boldsymbol{x}_{c,t}$ 和对应的标签 $y_{c,t}$. 为了保护客户端的数据隐私, 这些任务的原始数据不允许被转移出客户端本地. 此外, 在持续学习的过程中, 由于存储历史数据会带来巨大的存储开销, 因此客户端只可能存储一小部分历史任务的样本. 联邦持续学习的目的是通过交换客户端之间的知识来提升每客户端本地模型的性能, 同时也允许客户端不断的学习新任务, 系统整体的目标函数如下:

$$\min_{\{\boldsymbol{\theta}_c\}_{c=1}^{C}} \sum_{c=1}^{C}\sum_{t=1}^{T_c} \mathcal{L}_c(\mathcal{D}_{c,t};\boldsymbol{\theta}_c), \tag{1}$$

其中 $\mathcal{D}_{c,t}$ 代表客户端 c 上的任务 t, $\mathcal{L}_c$ 和 $\boldsymbol{\theta}_c$ 分别代表客户端 c 本地的目标函数和模型参数.

3 本文工作

为了应对联邦持续学习中的灾难性遗忘、通信资源受限以及异构性挑战, 本文提出一种基于原型学习的联邦持续学习方法, 方法框架如图 2 所示. 该方法包含 3 部分, 分别是基于原型学习的知识共享机制, 基于知识蒸馏的灾难性遗忘预防机制以及基于回放的灾难性遗忘预防机制.

3.1 基于原型学习的知识共享机制

现有的联邦持续学习方法大多采用基于参数聚合的知识共享方法, 即服务端将各个客户端的模型参数或梯度进行聚合. 然而这种方式不适用于动态性更强的任务场景, 联邦持续学习中各个客户端任务不同导致客户端本地模型异构, 使得无法通过参数聚合的方式进行知识共享. 在本文工作中, 我们引入基于原型学习的知识共享机制, 其为客户端本地模型建立统一的特征空间, 通过类原型共享知识, 能够适应模型异构的动态任务场景. 客户端 c 上的类别 i 的原型 $p_{c,i}$ 的计算方式为下式:

$$p_{c,i} = \frac{1}{|\mathcal{D}_{c,i}|} \sum_{(x,y)\in\mathcal{D}_{c,i}} \boldsymbol{\theta}_c(x), \tag{2}$$

其利用客户端 c 的特征提取器 $\boldsymbol{\theta}_c$ 获取客户端 c 上类别 i 的样本集 $\mathcal{D}_{c,i}$ 中样本的特征, 将特征均值记为原型. 对于已经学习到的类原型, 客户端保留对应类别的代表性样本, 采样策略见第 3.3 节, 利用代表性样本在持续学习中对原型进行持续更新以适应动态变化的数据分布. 在推理阶段, 原型学习根据式 (3) 进行分类, 最终预测的样本的类别 $\hat{y}$ 为样本 x 的特征表示距离最近的原型 p_i 的类别:

$$\hat{y} = \arg\min_i \|\boldsymbol{\theta}(x) - p_i\|_2. \tag{3}$$

然后各个客户端将本地的原型 $p_{c,i}$ 上传至服务端, 服务端对同类别的原型进行聚合, 如下式:

$$\overline{p}_i = \frac{1}{|\mathcal{C}_i|} \sum_{c\in\mathcal{C}_i} p_{c,i}, \tag{4}$$

其中 $\mathcal{C}_i$ 表示所有上传了类别 i 原型的客户端的集合, 服务端将所有客户端上传的本地原型聚合生成类别 i 的全局原型 $\overline{p}_i$, 随后将全局原型分发给各个客户端. 服务端的原型库保留所有已经学习到的原型, 并在获取到新原型时进行更新, 以此实现对动态场景中新类别的持续学习.

客户端在本地训练时通过原型损失 (5) 约束本地原型和全局原型的距离, 以此提升本地模型在全局数据分布上的泛化性能, 其中 $\mathbf{dis}(,)$ 表示距离度量, 本文中使用欧氏距离, 损失函数如下:

$$\mathcal{L}_R = \sum_i \mathbf{dis}(p_i, \overline{p}_i). \tag{5}$$

3.2 基于知识蒸馏的灾难性遗忘预防机制

知识蒸馏可以被引入持续学习中, 通过将旧模型的知识迁移到新模型中来预防灾难性遗忘. 首先利用旧模型计算当前任务的数据集的软标签, 该软标签包含了旧模型的知识. 然后在新任务过程中利用蒸馏损失项来约束新旧模型的偏差. 蒸馏损失如下式:

$$\mathcal{L}_D(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{i=1}^{N} y_i \log(\hat{y}_i), \tag{6}$$

其中 $\boldsymbol{y}$ 为软标签, 即旧模型对样本预测的概率分数, 而 $\hat{\boldsymbol{y}}$ 表示新模型对样本预测的概率分数, 蒸馏损失项 $\mathcal{L}_D$ 通过交叉熵损失计算两者的相似度, 从而利用旧模型来约束新模型的优化过程, 使新模型保持在旧任务上的性能. 而式 (6) 中的 $\boldsymbol{y}$ 和 $\hat{\boldsymbol{y}}$ 的计算方式采用了带温度的 softmax 函数, 如下式:

$$q_i = \frac{\exp(z_i/T)}{\sum_j \exp(z_j/T)}, \tag{7}$$

其中 q_i 为类别 i 的预测概率, 参数 T 代表温度. 当 T 为 1 的时候, 其等于一般的 softmax 函数, 而使用更大的 T 则能够使得更加各个类别之间的概率分布更加的平滑, 使得错误类别标签的预测概率得到保留, 从而保留更多的旧任务信息.

本文的方法基于原型学习, 模型输出的概率分数的计算方式与一般的神经网络不同. 计算方式如式 (8), 本文利用样本特征 $\boldsymbol{\theta}(\boldsymbol{x})$ 和类特征原型 p_k 的距离计算类别 k 概率分数, 类别 k 的概率分数为 $-\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}), p_k)$, 负号表示距离越小概率越大. 类别 k 的软标签的第 k 维 y_k 的计算方式如下:

$$y_k = \frac{\mathrm{e}^{-\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}),p_k)/T}}{\sum_{i=1}^{N} \mathrm{e}^{-\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}),p_i)/T}}. \tag{8}$$

3.3 基于回放的灾难性遗忘预防机制

回放是指保留一部分旧任务的代表性样本, 在学习新任务时加入训练过程, 从而缓解灾难性遗忘. 与传统的持续学习不同, 本文提出的联邦持续学习方法需要在保护客户端数据隐私的前提下处理分布式的灾难性遗忘问题, 每个客户端在本地均进行一个独立的持续学习过程, 数据不对外共享. 因此本文考虑在客户端本地保留一个私有的样本集. 在客户端本地保留样本集的主要限制是存储空间, 本文基于客户端的存储条件提出两种样本集保留的策略: 第一种策略是每个类别均保留相同数量的样本, 随着学习到的类别数目的增加, 样本集占的存储空间也增加; 第二种策略是客户端整体保留的样本数固定, 随着学习类别数目的增加, 每个类别保留的样本数减少. 上述两种策略中各个类别的样本数均相同, 目的是避免类别不均衡问题. 样本集保留中的核心问题是如何选择每个类别中最具代表性的样本. 本文的方法基于式 (9) 计算样本 $\boldsymbol{x}$ 和类别 i 原型的距离 $d_{\boldsymbol{x},i}$:

$$d_{\boldsymbol{x},i} = \mathbf{dis}\big(p_i, \boldsymbol{\theta}(\boldsymbol{x})\big), \tag{9}$$

其中 p_i 为类别 i 的原型, $\boldsymbol{\theta}(\boldsymbol{x})$ 为样本的特征表示, $\mathbf{dis}(,)$ 为距离度量.

3.4 整体算法流程

本节将介绍本文方法的整体流程, 包含服务端 (算法 1) 和客户端 (算法 2) 两部分.

服务端的主要功能是协调各个客户端的联合学习, 客户端之间通过原型共享知识, 以提升客户端本地模型的泛化性能. 服务端具体流程如下: 最外层循环为持续学习过程, 服务端制定持续学习的计划, 按任务依次进行持续学习; 每个持续学习任务均包含一个联邦学习过程, 包括客户端选择、客户端本地学习和服务端聚合三个步骤.

客户端本地学习的目标函数如式 (10), 共包括三个损失项: $\mathcal{L}_C$ 为分类损失项, 其主要针对学习的新类别; $\mathcal{L}_D$ 为蒸馏损失项, 其主要针对已经学习过的类别, 目的是减轻灾难性遗忘; $\mathcal{L}_R$ 为原型损失项, 通过最小化全局和本地模型的距离进行正则化约束, 以提升本地模型的泛化能力. α 和 β 为超参数, 用于调整蒸馏损失项和原型损失项的权重.

$$\mathcal{L} = \mathcal{L}_C + \alpha\mathcal{L}_D + \beta\mathcal{L}_R. \tag{10}$$

本文的分类损失项 $\mathcal{L}_C$ 与一般的交叉熵损失不同, 其使用原型学习的方式进行模型训练, 替换了作为分类器的全连接神经网络, 定义如下:

$$y_k = \frac{\mathrm{e}^{-\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}),p_k)/T}}{\sum_{i=1}^{N} \mathrm{e}^{-\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}),p_i)/T}}, \tag{11}$$

Algorithm 1 Server

Input: Clients $\{\mathcal{C}_i\}_{i=1}^{N}$, global prototypes $\mathcal{P} \leftarrow \{\}$;
Output: Global prototypes $\mathcal{P}$;

for task $t = 1, 2, ..., T$ **do**
  $\mathcal{C}_t \leftarrow$ Select clients from $\mathcal{C}$
  for round $r = 1, 2, .., R$ **do**
    Distribute $\mathcal{P}$ to each client in $\mathcal{C}_t$
    for each client $c \in \mathcal{C}_t$ in parallel **do**
      $\mathcal{P}_{c,t} \leftarrow$ Client c performs local update according to Algorithm 2
    end for
    $\mathcal{P} \leftarrow$ Aggregate $\{\mathcal{P}^{c,t} \mid c \in \mathcal{C}_t\}$ according to Eq. (4)
  end for
end for
return $\mathcal{P}$

Algorithm 2 Client local update

Input: Client c, task t, global communication round r, global prototypes $\mathcal{P}$;
Output: Client local prototypes $\mathcal{P}_t^c$;

$\hat{\mathcal{D}}_{c,t} \leftarrow \mathcal{D}_{c,t} \cup \mathcal{M}_c$ // Combine task dataset $\mathcal{D}_{c,t}$ and exemplar memory $\mathcal{M}_c$
for local epoch $e = 1, 2, ..., E$ **do**
  for each batch $b = \{x_i, y_i\}$ of $\hat{\mathcal{D}}_{c,t}$ **do**
    $\mathcal{L} = \mathcal{L}_C + \alpha\mathcal{L}_D + \beta\mathcal{L}_R$
    $\boldsymbol{\theta}_{c,t} \leftarrow \boldsymbol{\theta}_{c,t} - \eta\nabla\mathcal{L}$
  end for
end for
$\mathcal{P}_{c,t} \leftarrow$ Calculate local prototypes according to Eq. (2)
if $r == R$ **then**
  $\mathcal{M}_c \leftarrow$ Update local exemplar memory according to Eq. (9)
  $\boldsymbol{\theta}_{c,old} \leftarrow \boldsymbol{\theta}_{c,t}$
end if
return $\mathcal{P}_{c,t}$

$$\mathcal{L}_C = -\log P_{\boldsymbol{\theta}}(y = k \mid \boldsymbol{x}). \tag{12}$$

其中 $\mathbf{dis}(\boldsymbol{\theta}(\boldsymbol{x}), p_i)$ 表示样本 $\boldsymbol{x}$ 的特征 $\boldsymbol{\theta}(\boldsymbol{x})$ 与类别 i 的类原型 p_i 的距离, 负号表示将距离最小化优化.

客户端本地学习包括如下步骤: 首先利用当前任务数据和保留的样本集组成训练集; 然后根据目标函数式 (10) 进行训练; 训练结束后更新客户端的本地原型; 最后将客户端本地原型返回给服务端; 如果本次学习是当前任务的最后一次全局迭代, 则更新客户端的样本集并保留旧模型.

4 理论分析

4.1 收敛性分析

收敛性分析可以评估联邦持续学习的系统运行效率, 本节分析了本文方法的可收敛性.

假设 1 联邦持续学习中每个客户端本地的损失函数均是 L_1 利普希茨光滑 (L1-Lipschitz smooth) 的, 且损失函数为凸函数, 则每个客户端本地的损失函数的梯度是 L_1 利普希茨连续的 (L1-Lipschitz

continuous), 每个客户端的损失函数的梯度满足式 (13):

$$\|\nabla\mathcal{L}_{r_1}-\nabla\mathcal{L}_{r_2}\|_2\leqslant L_1\|\boldsymbol{\theta}_{c,r_1}-\boldsymbol{\theta}_{c,r_2}\|_2,\forall r_1,r_2>0,c\in\{1,2,...,C\},\tag{13}$$

其中 $\mathcal{L}$ 为损失函数, r_1,r_2 表示迭代轮次, L_1 为利普希茨常数, c 表示客户端, $\boldsymbol{\theta}_{c,r_1}$ 表示客户端 c 在轮次 r_1 时本地模型的权重.

根据式 (13), 可以引申出式 (14), 其是客户端本地损失函数变化的上界:

$$\mathcal{L}_{r_1}-\mathcal{L}_{r_2}\leqslant\langle\nabla\mathcal{L}_{r_2},(\boldsymbol{\theta}_{c,r_1}-\boldsymbol{\theta}_{c,r_2})\rangle+\frac{L_1}{2}\|\boldsymbol{\theta}_{c,r_1}-\boldsymbol{\theta}_{c,r_2}\|_2^2,\forall r_1,r_2>0,c\in\{1,2,...,C\}.\tag{14}$$

假设 2 设梯度 $g_{c,t}=\nabla\mathcal{L}(\boldsymbol{\theta}_{c,t})$ 是对每个客户端本地模型梯度的无偏估计, 则其期望:

$$\mathbb{E}_{\xi_c\sim D_c}[g_{c,t}]=\nabla\mathcal{L}(\boldsymbol{\theta}_{c,t})=\nabla\mathcal{L}_t,\forall c\in\{1,2,...,C\},\tag{15}$$

其方差的上界设为 σ^2:

$$\mathbb{E}[\|g_{c,t}-\nabla\mathcal{L}(\boldsymbol{\theta}_{c,t})\|_2^2]\leqslant\sigma^2,\forall c\in\{1,2,...,C\},\sigma^2\geqslant 0.\tag{16}$$

假设 3 假设梯度 $g_{c,t}$ 的二范数的期望的上界为 G:

$$\mathbb{E}[\|g_{c,t}\|_2]\leqslant G,\forall c\in\{1,2,...,C\}.\tag{17}$$

假设 4 所有客户端本地的特征提取器均为 L_2 利普希茨连续 (L2-Lipschitz continuous) 的, $\boldsymbol{\theta}_{c,r_1}$ 为客户端 c 在轮次 r_1 时的特征提取器参数, L_2 为利普希茨常数, 特征的变化率的上界为:

$$\|f_c(\boldsymbol{\theta}_{c,r_1})-f_c(\boldsymbol{\theta}_{c,r_2})\|\leqslant L_2\|\boldsymbol{\theta}_{c,r_1}-\boldsymbol{\theta}_{c,r_2}\|_2,\forall r_1,r_2>0,c\in\{1,2,...,C\}.\tag{18}$$

引理 1 根据假设 1 和假设 2, 可以推导出任意一个客户端在第 $r+1$ 次全局通信中结束本地训练后, 损失函数的期望的上界为式 (19), 其中 η 表示学习率, E 表示单轮全局迭代中本地迭代的轮次, $e=1/2$ 表示完成全局原型聚合的时刻.

$$\mathbb{E}[\mathcal{L}_{(r+1)E}]\leqslant\mathcal{L}_{rE+1/2}-\Big(\eta-\frac{L_1\eta^2}{2}\Big)\sum_{e=1/2}^{E-1}\|\nabla\mathcal{L}_{rE+e}\|_2^2+\frac{L_1E\eta^2}{2}\sigma^2.\tag{19}$$

引理 2 根据假设 3 和假设 4, 可以推导出在第 $r+1$ 次全局通信中结束并且服务器将客户端上传的原型聚合后, 各个客户端损失函数期望的上界为:

$$\mathbb{E}[\mathcal{L}_{(r+1)E+1/2}]\leqslant\mathcal{L}_{(r+1)E}+\lambda L_2\eta EG.\tag{20}$$

定理 1 结合引理 1 和引理 2, 可以推导出在第 $r+1$ 次全局通信中结束并且服务器将客户端上传的原型聚合后, 各个客户但损失函数期望的上界为:

$$\mathbb{E}[\mathcal{L}_{(r+1)E+1/2}]\leqslant\mathcal{L}_{rE+1/2}-\Big(\eta-\frac{L_1\eta^2}{2}\Big)\sum_{e=1/2}^{E-1}\|\nabla\mathcal{L}_{rE+e}\|_2^2+\frac{L_1E\eta^2}{2}\sigma^2+\lambda L_2\eta EG.\tag{21}$$

推论 1 由定理 1 可以推导出, 当满足式 (22) 和式 (23) 的条件时, 各客户端的损失其中函数对全局通信轮次满足单调递减, 即可以证明本文提出的联邦持续学习方法是可收敛的.

$$\eta_{e'}<\frac{2\big(\sum_{e=1/2}^{e'}\|\nabla\mathcal{L}_{rE+e}\|_2^2-\lambda L_2EG\big)}{L_1\big(\sum_{e=1/2}^{e'}\|\nabla\mathcal{L}_{rE+e}\|_2^2+E\sigma^2\big)},e'=1/2,1,...,E-1,\tag{22}$$

$$\lambda_t < \frac{\|\nabla \mathcal{L}_{rE+1/2}\|_2^2}{L_2 EG}, \tag{23}$$

其中, η 表示学习率, λ 为表示重要性的超参数, e 表示客户端本地迭代次数, r 表示全局通信的轮次.

4.2 对蒸馏温度的分析

本节对公式 (7) 中的蒸馏温度 T 进行了进一步分析, 分析了不同的蒸馏温度对模型性能的影响.

在本文提出的方法中, 知识蒸馏被用于缓解灾难性遗忘, 其中旧任务的模型作为教师模型, 而当前正在学习的模型作为学生模型, 通过知识蒸馏将旧模型的知识迁移至新模型中, 从而减少模型对旧知识的遗忘. 温度 T 直接影响了 softmax 函数的输出, 图 3 中对比了不同的温度下 softmax 函数的输出, 图中为了便于对比将模型输出的对各类别的预测概率按降序排序. 可以看到 T 越小, 不同类别预测概率之间的差别越大, 其曲线越陡峭, 当 $T = 1$ 时即为普通的 softmax 函数; 而当 T 越大, 不同类别的预测概率的差异越小, 对应的曲线越平滑.

教师模型的 softmax 输出作为学生模型的监督信息, 其对于知识蒸馏的效果有重要的影响. 新任务的标签为独热编码形式, 其中正确的类别对应的数值为 1, 其他类别为 0, 其中没有更加丰富的类别之间的对比关系. 而教师模型的目的是为了尽可能的保留更多的旧任务的知识, 其输出中不仅提供了正确类别的监督信息, 同时也包含了类别之间的对比关系. 当 $T > 1$ 时, 其能够使得预测向量更加平滑, 强化不同类别之间相互关系的信息. 但当 $T \to \infty$ 时, 如果预测类别数为 C, 则每个类别的预测概率均为 $1/C$, 这会导致类别的判别信息消失.

图 4 中对比了温度 $T \in \{1, 2, 3, 5, 10\}$ 下本文方法的性能, 可以看出当温度过小 ($T = 1$) 或者过大 ($T = 10$) 时, 模型的性能均会受到负面影响, 而其中性能最好的为 $T = 2$ 的温度设置. 这表明过小的 T 会使教师模型的标签缺少类别间的对比知识, 使得教师模型的知识无法被有效迁移至学生模型, 导致对旧知识的遗忘; 而过大的 T 则会弱化类别的判别知识, 造成学生模型对正确类别的混淆, 因此选取合适的 T 至关重要.

5 实验

5.1 实验数据

本文中实验使用的数据集为 CIFAR100 [30] 和 TinyImageNet [31]. CIFAR100 共包含 60000 张图像数据, 分为 100 个类别, 每个类别包含 500 个训练数据和 100 个测试数据, 其图像的尺寸为 $32 \times 32 \times 3$. TinyImageNet 共包含 120000 张图像数据, 分为 200 个类别, 每个类别包含 500 个训练数据, 50 个验证数据和 50 个测试数据, 其图像的尺寸为 $64 \times 64 \times 3$.

5.2 实验设置

在实验数据的设置上, 本文将每个数据集分为若干个持续学习任务, 每个持续学习任务包含若干个类别, 具体设置如表 1. 在进行联邦持续学习时, 各客户端会随机获得一个包含若干任务的序列, 各客户端任务数量可能不同, 客户端之间可能学习相似或无关的任务, 客户端学习的类别可能存在重叠, 且客户端间的样本没有重叠, 且由于每个客户端学习的类别数不同, 因此各个客户端模型的分类器结构也不同. 数据异构性表现在客户端学习的任务数量, 任务内容以及样本的差异性, 客户端间的数据异构性使得单个的全局模型难以在所有客户端上均取得较好的性能. 模型异构性表现在各个客户端学习任务不同导致的模型结构的不同, 其无法通过直接的参数平均进行聚合. 客户端数量为 10, 单个任务

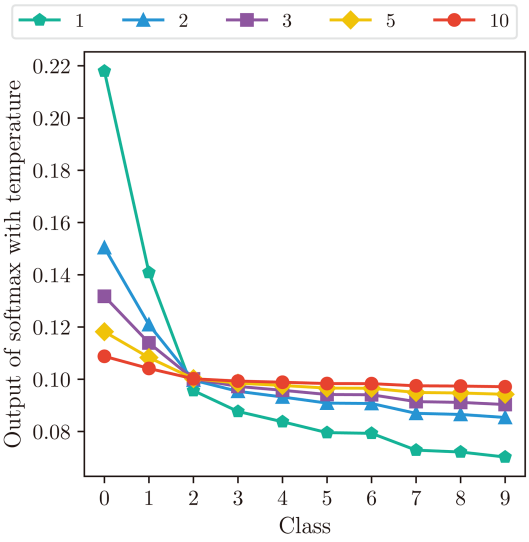

图 3 (网络版彩图) 不同蒸馏温度下 softmax 输出的对比

Figure 3 (Color online) Comparison of softmax function output at different distillation temperatures

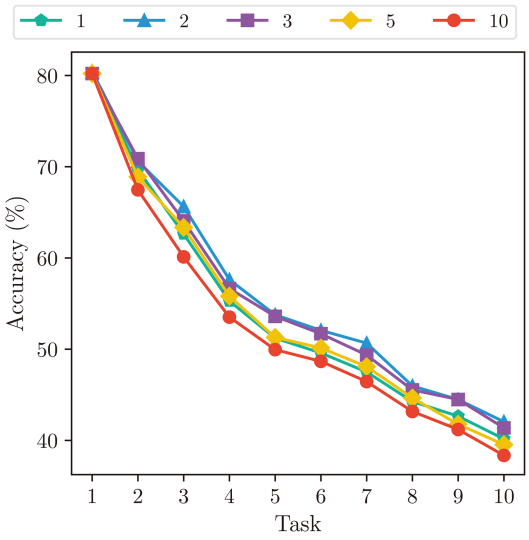

图 4 (网络版彩图) 不同蒸馏温度下模型在 CIFAR100 数据集上准确率的对比

Figure 4 (Color online) Comparison of model accuracy on CIFAR100 for different distillation temperatures

表 1 实验数据集的设置

Table 1 Experimental dataset setup

Dataset	Number of training samples	Number of classes	Number of tasks	Number of classes per task
CIFAR100	50k	100	10	10
TinyImageNet	100k	200	10	20

进行 10 轮全局通信, 在每轮全局通信中被选中的客户端在本地进行 3 轮迭代. 实验中模型训练使用 SGD 方法进行优化, 学习率初始设置为 0.01, 其根据损失的收敛情况进行动态调整, 调整范围为 0.01 到 0.0001.

实验的测试指标为模型的分类准确率 (Accuracy), 在实验结果中均以省略百分号的形式表示, 测试设置为类增量学习 (class-incremental learning), 即测试过程中样本所属的任务标识未知.

本文实验使用设备的操作系统为 Ubuntu 18.04.6 LTS, CPU 型号为 Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz, 内存大小为 128GB, GPU 型号为 GeForce RTX 3090. 实验使用语言为 Python3, 使用的深度学习框架为 Pytorch.

5.3 对比方法

本文的实验包括消融实验和对比实验两部分. 消融实验中对本文提出的联邦持续学习中的蒸馏损失、原型损失和回放这三项进行实验验证, 分析了每一项对于方法性能的影响. 在消融实验中对比了以下不同的设置, Ours w/o $\mathcal{M}$ 表示不使用回放的本文方法, Ours w/o $\mathcal{L}_R$ 表示不使用原型学习的本文方法, Ours w/o $\mathcal{L}_D$ 表示不使用知识蒸馏的本文方法.

对比实验中共包括 6 个联邦持续学习的对比方法: FedWeIT [9], GLFC [10], CFeD [12], FedRecon [13], FedSI [22] 和 FedPMR [11]. FedWeIT 将参数解耦为通用参数和任务特定参数以实现知识的选择性迁移.

表 2　在数据集 CIFAR100 上的准确率 (%) 对比

Table 2　Accuracy (%) comparisons on CIFAR100

Method	Task										Avg.	Δ
	1	2	3	4	5	6	7	8	9	10		
FedWeIT	73.0	59.3	53.6	45.6	43.7	42.9	40.5	38.2	36.4	33.9	46.7	⇓9.6
GLFC	78.1	63.0	56.4	48.0	40.0	41.1	42.2	40.0	35.6	34.0	47.8	⇓8.5
CFeD	75.8	63.0	57.0	51.2	47.0	46.7	43.7	39.5	39.6	38.5	50.2	⇓6.1
FedRecon	78.2	59.6	51.9	44.9	42.0	40.0	35.4	34.1	31.9	30.6	44.9	⇓11.4
FedSI	77.5	62.5	56.7	51.3	47.3	45.1	42.6	40.3	38.4	37.1	49.9	⇓6.4
FedPMR	76.1	61.3	55.7	46.9	45.6	43.2	41.3	38.4	37.1	35.2	48.1	⇓8.2
Ours w/o $\mathcal{M}$	80.8	45.4	29.4	20.8	18.0	16.5	12.5	11.2	10.4	8.8	25.4	⇓30.9
Ours w/o $\mathcal{L}_R$	79.5	61.0	54.9	45.7	48.0	41.2	41.0	40.1	35.7	35.5	48.3	⇓8.0
Ours w/o $\mathcal{L}_D$	80.8	58.7	48.9	41.9	36.3	35.4	32.4	29.9	28.6	25.4	41.8	⇓14.5
Ours	80.5	70.6	65.6	57.6	53.8	52.1	50.7	46.0	44.5	42.1	56.3	-

表 3　在数据集 TinyImageNet 上的准确率 (%) 对比

Table 3　Accuracy (%) comparisons on TinyImageNet

Method	Task										Avg.	Δ
	1	2	3	4	5	6	7	8	9	10		
FedWeIT	71.6	50.1	46.0	39.7	39.5	35.2	32.7	29.4	32.0	31.7	40.8	⇓8.4
GLFC	73.4	49.3	45.2	41.1	38.9	33.9	29.0	27.9	26.5	25.6	39.1	⇓10.1
CFeD	73.6	52.4	48.1	41.6	40.6	38.4	39.4	34.4	31.4	29.7	42.9	⇓6.3
FedRecon	67.1	52.0	46.8	43.7	42.7	39.7	37.6	35.4	34.0	33.2	43.2	⇓6.0
FedSI	73.7	52.5	47.1	45.3	41.4	39.5	36.1	33.5	32.6	30.3	43.2	⇓6.0
FedPMR	72.3	51.1	46.3	42.1	40.6	37.5	35.4	32.4	31.1	29.5	41.8	⇓7.4
Ours w/o $\mathcal{M}$	76.4	37.0	28.4	21.7	24.4	16.3	14.4	13.2	11.5	10.2	25.3	⇓23.9
Ours w/o $\mathcal{L}_R$	70.5	49.0	43.9	41.3	37.4	32.4	32.2	28.8	26.5	27.2	38.9	⇓10.3
Ours w/o $\mathcal{L}_D$	76.4	51.0	43.2	36.5	37.3	30.1	27.5	25.5	22.0	19.6	36.9	⇓12.3
Ours	76.1	60.4	54.9	50.2	51.5	44.4	42.3	40.9	36.7	34.5	49.2	-

GLFC 利用正则化约束以及全局模型选择解决灾难性遗忘问题. CFeD 通过客户端和服务端的双向知识蒸馏传递知识. FedRecon 利用预训练的特征提取器获取客户端数据的特征并构建分类器. FedSI 利用正则化约束缓解灾难性遗忘. FedPMR 则通过参数分解和回放提升模型的持续学习性能.

5.4　实验结果

对比实验的结果见表 2 和表 3, 表中 ⇓ 表示对比方法相较本文方法在平均准确率上降低的数值, 可见本文方法在平均准确率上更优, 在 CIFAR100 上的准确率较对比方法高出 6.1%~11.4%, 在 TinyImageNet 则高出 6.0%~10.1%. 此外, 从持续学习的角度看, 本文方法在学习新任务时的准确率的下降速度较对比方法更慢, 这表明本文方法能更有效的缓解灾难性遗忘. 本文方法和对比方法在任务 1 上的准确率不同, 原因是不同的方法采用了不同的知识交换策略, 导致联邦学习的结果不同.

消融实验的结果见表 2 和表 3, 可以看出本文方法相较于其他三个消融实验方法在平均准确率上

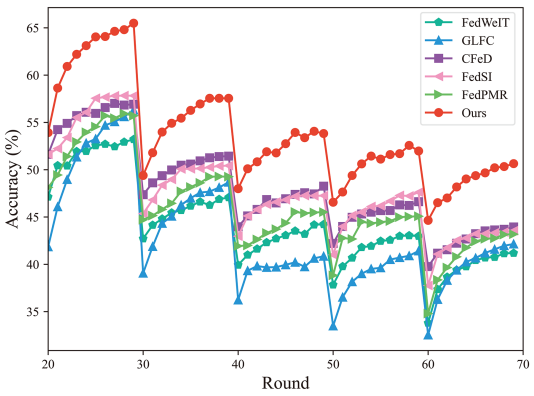

图 5 (网络版彩图) 在 CIFAR100 数据集上收敛速率对比

Figure 5 (Color online) Comparison of the convergence rates on CIFAR100

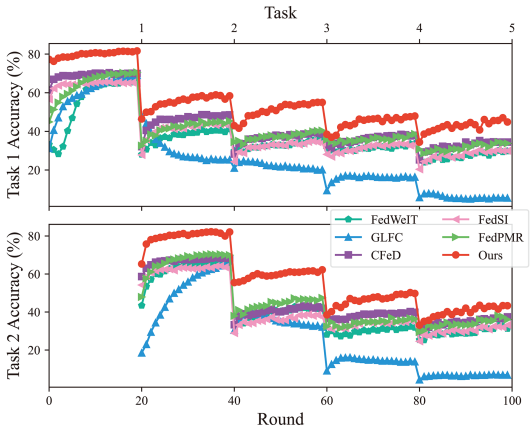

图 6 (网络版彩图) 在 CIFAR100 数据集上灾难性遗忘对比

Figure 6 (Color online) Comparison of catastrophic forgetting on CIFAR100

更优, 在 CIFAR100 上的准确率高出 8.0%~30.9%, 在 TinyImageNet 则高出 10.3%~23.9%, 这表明本文方法的三个改进点均是有效的. 从任务 1 的准确率看, Ours w/o $\mathcal{L}_R$ 较其他方法的准确率更低, 这是由于 Ours w/o $\mathcal{L}_R$ 没有进行客户端之间的知识交换, 使得客户端本地模型的泛化能力更差. 而 Ours w/o $\mathcal{M}$ 和 Ours w/o $\mathcal{L}_D$ 在越靠后的任务中性能越差, 这表明这回放和知识蒸馏能有效缓解遗忘.

此外, 本文还研究了本文方法的收敛速率和遗忘率. 收敛速率的对比见图 5, 其中每个任务进行 10 次全局通信. 图中存在周期性的波形, 这是因为开始学习新任务时模型还没有在新任务数据集上收敛, 因此准确率较低, 随着迭代的过程模型会逐渐收敛, 准确率也随之上升, 例如图中准确率在 20 到 30 轮次间持续上升. 根据图 5 可知本文方法有更快的收敛速率. 遗忘率的对比见图 6, 图中展示了两个任务在学习过程中的准确率变化, 每个任务进行 20 次全局通信, 遗忘指模型在学习新任务时在旧任务上的性能损失. 灾难性遗忘表现在图中准确率突然下降的节点, 例如在第 20 和第 40 轮次, 这里模型开始学习新任务后对旧任务性能会出现断崖式的下降. 从图中曲线可以看出, 本文方法的准确率下降更为缓慢, 且最终能够保持较高的准确率, 这表明本文方法能更有效的缓解灾难性遗忘.

本文对比了不同的增量学习任务数量下各对比方法的性能, 结果见图 7, 其中任务数量包括 5, 10, 15 和 20. 由图 7 可知, 本文方法在不同的增量学习任务下性能均优于其他对比方法. 图中任务数量越多开始学习时的准确率越高, 其原始是总的数据集规模固定, 单个任务包含的类别数量随任务数量增加而减小, 而类别数量越少的任务越容易被学习, 因此准确率更高.

本文还对比了不同方法的通信和计算开销, 结果见表 4, 表中对比了不同方法在单轮迭代的传输的参数数量和时间开销. 在通信开销上, 对比方法均需要传递模型参数, 而且 FedWeIT 和 FedPMR 采用了参数隔离的设计需要传递 2 倍的参数, 而本文方法仅需要传递类原型, 在通信开销上远小于对比方法. 在计算开销上, 本文方法的计算耗时仅次于 FedRecon, 而优于其他对比方法. 对比结果表明本文方法在通信开销上有绝对的优势, 在计算开销上同样也有较好的表现.

5.5 结果讨论

本文方法在通信效率优于其他对比方法. 现有联邦持续学习方法需要共享整个模型的参数, 在本

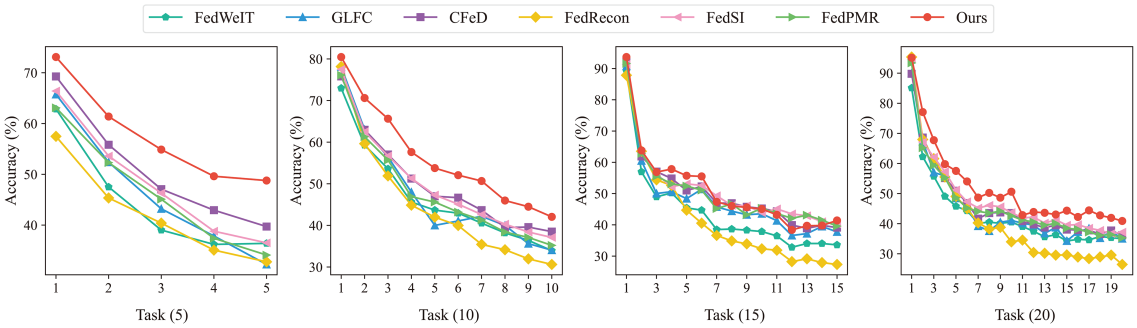

图 7　(网络版彩图) 在 CIFAR100 数据集上针对不同任务数量的分析实验, 任务数量 $T \in \{5, 10, 15, 20\}$

Figure 7　(Color online) Analytical experiments on CIFAR100 with different numbers of incremental tasks, where $T \in \{5, 10, 15, 20\}$

表 4　使用 Resnet34 模型时各方法的通信和计算开销对比

Table 4　Comparison of the communication and computation overheads of different methods when using the Resnet34

	FedWeIT	GLFC	CFeD	FedRecon	FedSI	FedPMR	Ours
Params	41M	21M	21M	21M	21M	41M	50K
Time (s)	17.2	14.5	23.1	11.1	15.6	17.6	13.6

文实验中使用的 Resnet34 的模型参数数量为 21M. 而本方法仅需要共享类原型, 其中每个类原型的参数数量为 512, 假设共享 50 个类别的参数, 总共仅需要 25600 个参数. 因此本文提出的方法在通信效率上相较于其他方法更优, 对于客户端的网络环境要求更低, 适用场景更加广泛.

本文方法更加适用于模型异构的学习场景. 在实际应用中, 各个客户端的模型由于硬件资源和学习任务的差异性可能在模型结构上存在异构性, 导致无法通过参数聚合实现知识共享. 例如客户端 A 在本地学习 a,b,c 三个类别, 而客户端 B 在本地学习 b,c,d,e 四个类别, 如果使用传统的联邦学习方法, 由于两个客户端分类模型所针对的任务不同, 因此无法直接进行参数的聚合和共享. 而本文方法仅需要共享客户端本地的类原型, 对模型的结构没有要求, 知识的共享仅要求各客户端的特征空间的维度相同即可, 因此本文方法能够适应模型异构的应用场景.

本文在隐私保护方面优于基于参数聚合的联邦持续学习方法. 首先, 客户端之间只共享原型而不共享模型参数, 攻击者在缺少模型的情况下无法根据特征逆向推算出客户端的原始数据. 其次, 客户端的本地原型的聚合过程是不可逆的, 原型聚合的过程保留了数据整体的统计特征而消除了单个样本的特性, 多个样本特征在聚合后难以反向分解出聚合前的特征, 因此攻击者无法获得客户端的原始数据. 而基于参数聚合的方法中客户端需要向服务端上传本地模型, 而攻击者则能够根据模型的参数更新信息更容易的推理出模型训练使用的数据, 这使得客户端的隐私安全受到威胁.

6　结论

联邦持续学习面临的挑战包括灾难性遗忘, 异构性和通信资源受限, 本文针对这些挑战提出了一种基于原型学习的联邦持续学习方法. 针对灾难性遗忘, 本文提出基于知识蒸馏和回放的预防机制, 利用旧任务模型以及旧任务数据的代表性样本缓解模型在学习新任务时对旧任务产生的遗忘. 本文将原型学习引入联邦持续学习, 解决了现有联邦持续方法通讯开销大的问题, 同时能够适应模型异构的

应用场景, 能够在异构场景中高效共享客户端的知识. 本文提供了方法的收敛性分析, 验证了方法的理论可行性. 此外, 本文在多个数据集上与其他方法进行对比, 证明了本文方法在性能上的优越性.

参考文献

1 McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, 2017. 1273–1282

2 Li T, Sahu A K, Zaheer M, et al. Federated optimization in heterogeneous networks. In: Proceedings of the 3rd Conference on Machine Learning and Systems, Austin, 2020. 429–450

3 Collins L, Hassani H, Mokhtari A, et al. Exploiting shared representations for personalized federated learning. In: Proceedings of the 38th International Conference on Machine Learning. Virtual, 2021. 2089–2099

4 Tan Y, Long G, Liu L, et al. Fedproto: Federated prototype learning across heterogeneous clients. In: Proceedings of the 36th AAAI Conference on Artificial Intelligence, Virtual, 2022. 8432–8440

5 Li D, Wang J. Fedmd: Heterogenous federated learning via model distillation. 2019. ArXiv:1910.03581

6 Gao S, Yuan L P, Zhu J M, et al. A blockchain-based privacy-preserving asynchronous federated learning. Sci Sin Inform, 2021, 51: 1755–1774 [高胜, 袁丽萍, 朱建明, 等. 一种基于区块链的隐私保护异步联邦学习. 中国科学: 信息科学, 2021, 51: 1755–1774]

7 Xiong Z, Cheng Z, Lin X, et al. Facing small and biased data dilemma in drug discovery with enhanced federated learning approaches. Sci China Life Sci, 2022, 65: 529–539

8 Peng B, Chi M M, Liu C, et al. Non-IID federated learning via random exchange of local feature maps for textile IIoT secure computing. Sci China Inf Sci, 2022, 65: 170302

9 Yoon J, Jeong W, Lee G, et al. Federated continual learning with weighted inter-client transfer. In: Proceedings of the International Conference on Machine Learning, Virtual, 2021. 12073–12086

10 Dong J, Wang L, Fang Z, et al. Federated class-incremental learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New Orleans, 2022. 10164–10173

11 Wang Z, Zhang Y, Xu X, et al. Federated Probability Memory Recall for Federated Continual Learning. Inf Sci, 2023

12 Ma Y, Xie Z, Wang J, et al. Continual Federated Learning Based on Knowledge Distillation. In: Proceedings of the Proceedings of the 31st International Joint Conference on Artificial Intelligence, Vienna, 2022

13 Hendryx S M, KC D R, Walls B, et al. Federated reconnaissance: Efficient, distributed, class-incremental learning. 2021. ArXiv:2109.00150

14 Ni X M, Shen X Y, Zhang H. Adaptive personalized federated learning for heterogeneous data: a method based on parameter decomposition and continual learning. Sci Sin Inform, 2022, 52: 2306–2320 [倪宣明, 沈鑫圆, 张海. 面向异构数据的自适应个性化联邦学习 —— 一种基于参数分解和持续学习的方法. 中国科学: 信息科学, 2022, 52: 2306–2320]

15 Li Z, Hoiem D. Learning without Forgetting. IEEE Trans Pattern Anal Mach Intell, 2017, 40: 2935–2947

16 Hinton G, Vinyals O, Dean J. Distilling the knowledge in a neural network. 2015. ArXiv:1503.02531

17 Dhar P, Singh R V, Peng K C, et al. Learning without memorizing. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, 2019. 5138–5146

18 Rebuffi S A, Kolesnikov A, Sperl G, et al. icarl: Incremental classifier and representation learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, 2017. 2001–2010

19 Castro F M, Marín-Jiménez M J, Guil N, et al. End-to-end incremental learning. In: Proceedings of the European Conference on Computer Vision, Munich, 2018. 233–248

20 Kirkpatrick J, Pascanu R, Rabinowitz N, et al. Overcoming catastrophic forgetting in neural networks. Proc Natl Acad Sci USA, 2017, 114: 3521–3526

21 Zenke F, Poole B, Ganguli S. Continual learning through synaptic intelligence. In: Proceedings of the 34th International Conference on Machine Learning, Sydney, 2017. 3987–3995

22 Zhang Z, Zhang Y, Guo D, et al. Communication-efficient federated continual learning for distributed learning system with Non-IID data. Sci China Inf Sci, 2023, 66: 122102

23 Chaudhary Y, Rai P, Schubert M, et al. Federated Continual Learning for Text Classification via Selective Inter-client

Transfer. 2022. ArXiv:2210.06101
24 Huang W, Ye M, Du B. Learn from others and be yourself in heterogeneous federated learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, New Orleans, 2022. 10143–10153
25 Usmanova A, Portet F, Lalanda P, et al. Federated Continual Learning through distillation in pervasive computing. In: Proceedings of the IEEE International Conference on Smart Computing, Espoo, 2022. 86–91
26 Schreyer M, Hemati H, Borth D, et al. Federated Continual Learning to Detect Accounting Anomalies in Financial Auditing. 2022. ArXiv:2210.15051
27 Zhang Z, Guo B, Sun W, et al. Cross-FCL: Toward a Cross-edge Federated Continual Learning Framework in Mobile Edge Computing Systems. IEEE Trans Mob Comput, 2022
28 Talpur A, Gurusamy M. GFCL: A GRU-based Federated Continual Learning Framework against Adversarial Attacks in IoV. 2022. ArXiv:2204.11010
29 Zhu M, Chen Z, Chen K, et al. Attention-based federated incremental learning for traffic classification in the Internet of Things. Comput Commun, 2022, 185: 168–175
30 Krizhevsky A, Hinton G, et al. Learning multiple layers of features from tiny images. 2009
31 Le Y, Yang X. Tiny imagenet visual recognition challenge. 2015

Federated continual learning based on prototype learning

Hao-Dong ZHANG[1], Liu YANG[1*], Jian YU [2], Qing-Hua HU[1] & Li-Ping JING[2]

1. *College of Intelligence and Computing, Tianjin University, Tianjin* 300354*, China*;
2. *School of Computer and Information Technology, Beijing Jiaotong University, Beijing* 100044*, China*
* Corresponding author. E-mail: yangliuyl@tju.edu.cn

Abstract Federated learning allows multiple participants to collaborate on training models while preserving privacy. However, traditional federated learning methods do not support continuous learning and are not well-suited for dynamic scenarios. Recently, federated continual learning has emerged as a promising approach that enables ongoing learning and collaboration among participants. This scenario introduces additional complexities, such as catastrophic forgetting, heterogeneity, and limited communication resources. Considerably, this paper proposes a prototype-based federated continual learning approach. The proposed method uses prototypes for knowledge transfer, which improves communication efficiency and adaptability to model heterogeneity. Additionally, we introduce a mechanism to mitigate catastrophic forgetting through knowledge distillation and replay. We also provide a convergence analysis of the proposed method and validate its effectiveness through comparative and ablation experiments.

Keywords federated continual learning, prototype learning, knowledge distillation, catastrophic forgetting, data heterogeneity