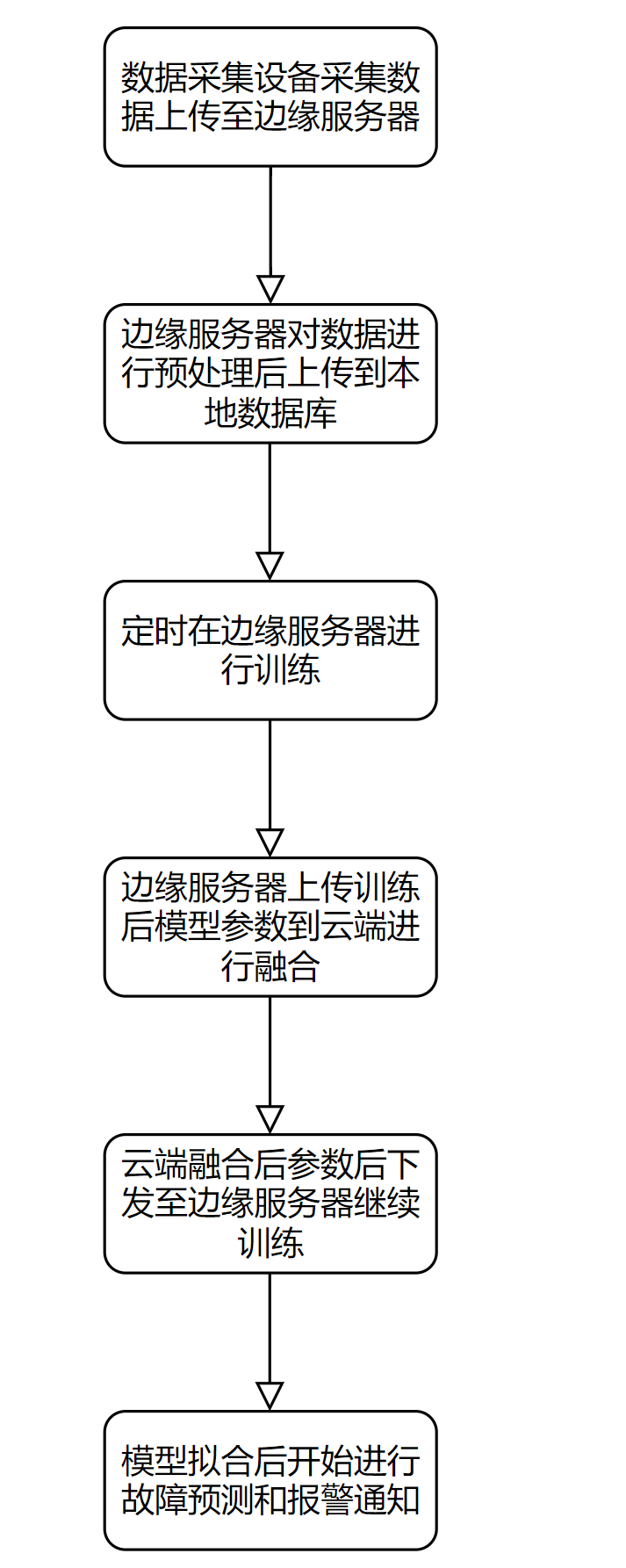
**说 明 书 摘 要**

本发明公开了一种基于云边端架构的列车故障诊断系统，包括：在高速铁路沿线部署边缘计算节点，用于实时采集列车运行数据；利用联邦学习算法，在各边缘服务器独立训练局部故障诊断模型，并将模型更新传输至中心云进行全局模型聚合；结合动态资源调度技术，优化边缘服务器的资源利用率；在诊断过程中，结合多模态数据，使用深度学习算法提升故障模式识别的准确性；通过智能决策支持系统，生成优化的维护计划。本发明有效提升了高速铁路故障诊断的实时性、准确性和安全性，避免了单点故障和数据传输延迟的问题。

**摘 要 附 图**



**权 利 要 求 书**

1. 一种基于云边端架构的列车故障诊断系统，其特征在于，包括以下步骤：

步骤1：基于列车运行过程中产生的数据，使用边缘服务器进行采集和预处理，包括视频、传感器及设备数据；

步骤2：设计联邦学习框架，边缘服务器分别训练局部故障诊断模型；

步骤3：通过网络将训练后的局部模型参数上传至中心云，生成全局诊断模型；

步骤4：利用云端Slurm调度器优化资源，执行全局模型的诊断任务；

步骤5：将诊断结果返回边缘服务器，提供实时监测与告警。

2. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，所述步骤1中的具体实施如下：

步骤1.1：通过不同协议如MQTT、HTTP、RTMP传输数据至边缘服务器；

步骤1.2：在边缘服务器上对传输数据进行事件匹配、清洗和格式整理；

步骤1.3：对数据进行分类处理，视频数据、传感器数据、设备数据分别进行独立处理。

3. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，所述步骤2中的具体实施步骤为：

步骤2.1：设计边缘服务器的局部模型训练框架，边缘服务器独立进行模型训练；

步骤2.2：通过边缘服务器采集到的数据进行故障诊断模型的局部训练；

步骤2.3：将训练完成的局部模型参数上传至中心云进行聚合，生成全局模型。

4. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，所述步骤3的具体实施步骤如下：

步骤3.1：各边缘服务器通过安全通道将本地训练得到的局部模型参数（如权重、偏置等）传输至中心云端。传输采用安全加密协议（如TLS）确保数据在传输过程中的安全性。

步骤3.2：云端服务器使用联邦学习算法对从各边缘服务器传来的局部模型参数进行聚合，生成全局故障诊断模型。聚合方法可以采用加权平均、FedAvg等常见方法，以确保不同边缘服务器模型的贡献合理融合。

步骤3.3：云端在聚合生成全局模型后，通过历史故障数据集对模型进行测试和校验。如果校验结果符合预期，则更新全局模型；否则，调整聚合策略或对边缘服务器下发新的训练指令。

5. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，所述步骤4的具体实施步骤如下：

步骤4.1：通过Slurm调度器调度云端计算资源，用于全局模型训练；

步骤4.2：根据诊断任务的优先级调整资源调度策略；

步骤4.3：对诊断任务的复杂度进行分析，动态优化资源配置。

6. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，所述步骤5的具体实施步骤如下：

步骤5.1：通过消息队列，将诊断结果反馈至边缘服务器，进行实时告警；

步骤5.2：在边缘服务器上实时更新局部模型，提高诊断的准确度和响应速度；

步骤5.3：通过API将诊断结果可视化，展示给用户。

7. 根据权利要求1所述的基于云边端架构的列车故障诊断系统，其特征在于，系统支持数据安全保障，所有数据传输均采用加密技术，确保数据的隐私与完整性。

**说 明 书**

**一种基于云边端架构的列车故障诊断系统**

**技术领域**

本发明涉及智能列车及工业系统中的故障诊断技术领域，具体涉及一种基于云边端架构的列车故障诊断系统。该系统旨在通过利用云计算和边缘计算相结合的方式，实现对列车运行设备的实时监测和故障诊断。

**背景技术**

随着现代工业和智能交通系统的迅速发展，列车等关键设备的运行可靠性变得至关重要。传统的故障诊断多依赖于中心化的云端数据处理，然而由于工业设备数量的增加及数据量的激增，单一依赖云端处理存在延迟问题，难以满足实时性要求。同时，部分远程场景存在网络传输限制，使得数据无法及时上传云端进行分析。

云边端架构的引入能够有效解决这一问题。通过在边缘服务器进行预处理和局部诊断，结合云端的全局模型和大规模数据处理能力，云边协同可以显著提高故障诊断的效率与精度。特别是在智能列车的运行环境中，列车设备的实时监测与故障预测对于保障交通安全、优化运行效率、减少故障停运至关重要。将云边端架构应用于故障诊断系统，能够实现设备状态的实时分析和预测，提前预防故障并提升维护效率，进而保障系统稳定运行。

**发明内容**

本发明的目的是提供一种基于云边端架构的列车故障诊断系统，解决传统中心化诊断模型存在的诊断延迟、数据传输瓶颈及网络环境不稳定等问题，提升系统的实时性、可靠性与扩展性。

本发明提供了一种基于云边端架构的列车故障诊断方法，具体包括以下步骤：

步骤1：将列车运行过程中产生的实时数据上传至边缘服务器进行预处理并根据数据类型存入边缘服务器的数据库，包括数值类型、时序序列、文件类型数据库，所述数据包括以下几类：

1. 视频数据和图片数据：由摄像头采集的列车车厢内外的视频流和图像。

2. 设备数据：来自直连设备（如嵌入式传感器、监控仪表）和非直连设备（如通过数据网关接入的外部设备）的运行状态信息。

3. CSV格式的数据：如历史运行记录、故障日志、传感器的批量测量结果等结构化数据。

步骤2：使用联邦学习框架，边缘服务器分别使用各自采集到的数据进行局部故障诊断模型的训练。

步骤3：通过网络将训练后的局部模型参数上传至中心云，生成全局诊断模型；

步骤4：利用部署在中心云服务器的Slurm调度器优化服务器资源（如CPU、GPU的分配）分配，确保全局模型的诊断任务能够高效执行。

步骤5：将诊断结果返回边缘服务器，提供实时监测与告警。

进一步的，所述步骤1的具体实现步骤为：

步骤1.1：通过不同协议如MQTT（用于低带宽、低延迟的实时传感器数据传输）、HTTP（适用于批量数据上传和设备状态数据的传输）、RTMP（用于视频数据的实时流式传输）传输数据至边缘服务器，以适应不同类型设备的数据交互需求；

步骤1.2：在边缘服务器上对传输数据进行事件匹配（根据预定义的匹配规则，将设备发送过来的数据根据匹配规则进行解析、提取）、清洗（缺失值处理）和格式整理（将来自不同设备和传感器的数据转换为统一的结构化格式）；

进一步的，所述步骤2的具体实现步骤为：

步骤2.1：各边缘服务器独立训练局部的故障诊断模型。边缘服务器根据本地收集的设备数据，独立构建包含故障特征的局部故障诊断模型，算法模型（Q1，Q2…Qn）、模型被训练的触发条件、以及故障决策的规则统一通过在中心云的配置界面配置，之后中心云服务器将模型下发到边缘云服务器同时中心云服务器根据各边缘服务器的负载情况通过Slurm调度器动态的分配边缘服务器资源供模型训练。

步骤2.2： 模型下发到边缘云服务器后，各边缘服务器利用本地的数据集和中心云服务器派发下来的模型进行迭代训练，在每一轮训练结束后，边缘服务器将局部模型的参数（如模型的权重、偏置）使用AES算法进行加密，确保数据的安全性和隐私保护。

步骤2.3：将加密后的局部模型参数（如模型的权重、偏置）上传至中心云服务器进行聚合。

进一步的，所述步骤3的具体实现步骤为：

步骤3.1：各边缘服务器通过安全通道将本地训练得到的局部模型参数（如权重、偏置等）传输至中心云端。传输采用安全加密协议（如TLS）确保数据在传输过程中的安全性。

步骤3.2：中心云服务器使用联邦学习算法对从各边缘服务器传来的局部模型参数通过序贯卡尔曼滤波聚合算法生成全局故障诊断模型。该全局模型结合了来自不同边缘节点的知识，能够更全面地反映设备的故障特征，进一步提高故障诊断的准确性和效率。

步骤3.3：中心云服务器在聚合生成全局模型后，使用历史故障数据集对该模型进行测试和验证。验证过程通过计算模型在历史故障数据集上的准确率、召回率和F1分数等指标来判断校验结果是否符合预期。如果这些性能指标达到预定阈值，则更新全局模型并启动定时任务以及开启数据变化监听；如果不符合预期，则调整聚合策略以改进模型的性能。为增加灵活性，系统会以20%的概率随机丢弃边缘服务器发送的局部模型参数，并继续等待下一次边缘服务器发送的局部故障模型参数。这种策略能够有效提高全局模型的准确性和可靠性。

进一步的，所述步骤4的具体实现步骤为：

步骤4.1：Slurm调度器根据全局模型的诊断任务需求动态分配计算资源，如CPU和GPU，以保证任务在计算资源最优配置下执行。这一过程涉及对系统负载的实时监控，以实现资源的最优使用。

步骤4.2：在执行多个诊断任务时，Slurm调度器根据任务的优先级进行调度，确保高优先级任务能在第一时间得到处理。通过设置任务的优先级，确保关键任务在资源竞争中获得优先执行权，提高系统的响应速度。

步骤4.3：在诊断任务进行期间，Slurm调度器实时监控任务执行状态，并根据实际运行情况调整资源分配策略。例如，如果某个任务的执行时间超出预期，系统将重新评估资源分配，并在必要时为该任务增加更多的计算资源以提高其效率。

进一步的，所述步骤5的具体实现步骤为：

步骤5.1：通过消息队列（如RabbitMQ），将中心云服务器生成的诊断结果反馈至边缘服务器。边缘服务器根据预设的故障决策规则，自动触发警报并通知相关人员。这一机制保证了告警的实时性和稳定性，避免因网络延迟导致的响应延误。

步骤5.2：在边缘服务器上，依据最新的诊断结果对局部模型进行在线更新，采用增量学习的方式不断优化模型，提升诊断的准确性。同时，实时更新模型能缩短模型的响应时间，确保系统在发生新类型故障时具备快速适应的能力。

步骤5.3：通过中心云的数据可视化、边缘云负载、中心云负载、调度可视化、故障通知API接口，将中心云服务器和边缘云服务器实时负载和根据中心云服务器上的全局故障诊断模型得到的诊断结果以图表、报表或仪表盘的形式进行可视化展示。用户可以通过Web端或移动端实时查看系统的运行状态、故障记录及告警信息。此外，系统还可生成故障分析报告，为后续决策提供支持。

本发明基于云边端架构的列车故障诊断系统，实现了边缘计算与云端协同的诊断框架，充分发挥了两者的优势：边缘计算的实时性与云计算的高性能资源。在边缘服务器完成数据采集和局部模型的训练，有效减少了数据传输过程中的延迟，提高了实时监测和响应能力。系统采用联邦学习技术，将各边缘服务器训练的局部模型参数上传至云端进行聚合，生成全局模型。这一方式不仅避免了传统集中式模型带来的隐私泄露风险和高数据传输成本，还支持灵活的数据处理和计算资源分配。此外，系统利用列车的实时位置感知能力和对边缘服务器资源的监控，通过Slurm调度器动态优化边缘服务器资源与服务的调度。系统可将全局模型从较远的边缘服务器卸载并迁移至靠近列车的服务器上，实现高效的故障诊断和快速响应。同时，云端生成的全局模型诊断结果会即时反馈至边缘服务器，支持实时监控与告警功能，进一步提升了故障诊断的准确性和响应速度。与传统故障诊断系统相比，本发明通过云边协同优化了资源分配和计算效率，能够适应复杂、多变的工业环境，并有效解决大规模、实时性要求高的场景中的诊断问题。在保障数据安全与隐私的同时，系统通过增量式的资源调整和灵活的任务迁移，确保诊断过程的稳定性与智能化水平。最终，这一系统不仅提高了列车故障诊断的效率，还为工业场景中的大规模应用提供了新的解决方案。

**附图说明**

图1是本发明一种基于云边端架构的列车故障诊断系统的流程图，展示系统各个组成部分及其相互作用的整体架构。

图2是**数据流图**：说明设备数据、视频数据和传感器数据如何从设备传输到边缘服务器，并经过处理后上传至云端的流程。，说明设备数据、视频数据和传感器数据如何从设备传输到边缘服务器，并经过处理后上传至云端的流程。

图3是模型训练流程图，说明了云边端模型的派发和参数融合的过程。

**具体实施方式**

下面结合附图及实施例对本发明做进一步详细说明。

参见附图1，本发明基于云边端架构下的列车故障诊断系统包括以下步骤：

步骤1：基于列车运行过程中产生的数据，使用边缘服务器进行采集和预处理，包括视频、传感器及设备数据，如附图1接入层部分所示；

步骤1.1：对列车产生的数据，通过不同协议如MQTT、HTTP、RTMP传输数据至边缘服务器；

步骤1.2：在边缘服务器上对传输数据进行事件匹配、清洗和格式整理；

步骤1.3：对数据进行分类处理，视频数据、传感器数据、设备数据分别进行独立处理。

步骤2：设计联邦学习框架，边缘服务器分别训练局部故障诊断模型

步骤2.1：设计边缘服务器的局部模型训练框架，边缘服务器独立进行模型训练；

步骤2.2：通过边缘服务器采集到的数据进行故障诊断模型的局部训练；

步骤2.3：将训练完成的局部模型参数上传至中心云进行聚合，生成全局模型。

步骤3：通过网络将训练后的局部模型参数上传至中心云，生成全局诊断模型如附图3；

步骤3.1：各边缘服务器通过安全通道将本地训练得到的局部模型参数（如权重、偏置等）传输至中心云端。传输采用安全加密协议（如TLS）确保数据在传输过程中的安全性。

步骤3.2：云端服务器使用联邦学习算法对从各边缘服务器传来的局部模型参数进行聚合，生成全局故障诊断模型。聚合方法可以采用加权平均、FedAvg等常见方法，以确保不同边缘服务器模型的贡献合理融合。

步骤3.3：云端在聚合生成全局模型后，通过历史故障数据集对模型进行测试和校验。如果校验结果符合预期，则更新全局模型；否则，调整聚合策略或对边缘服务器下发新的训练指令。

步骤4：利用云端Slurm调度器优化资源，执行全局模型的诊断任务；步骤4.1：通过Slurm调度器调度云端计算资源，用于全局模型训练；

步骤4.2：根据诊断任务的优先级调整资源调度策略；

步骤4.3：对诊断任务的复杂度进行分析，动态优化资源配置。

步骤5：将诊断结果返回边缘服务器，提供实时监测与告警。

步骤5.1：通过消息队列，将诊断结果反馈至边缘服务器，进行实时告警；

步骤5.2：在边缘服务器上实时更新局部模型，提高诊断的准确度和响应速度；

步骤5.3：通过API将诊断结果可视化，展示给用户。

**说 明 书 附 图**

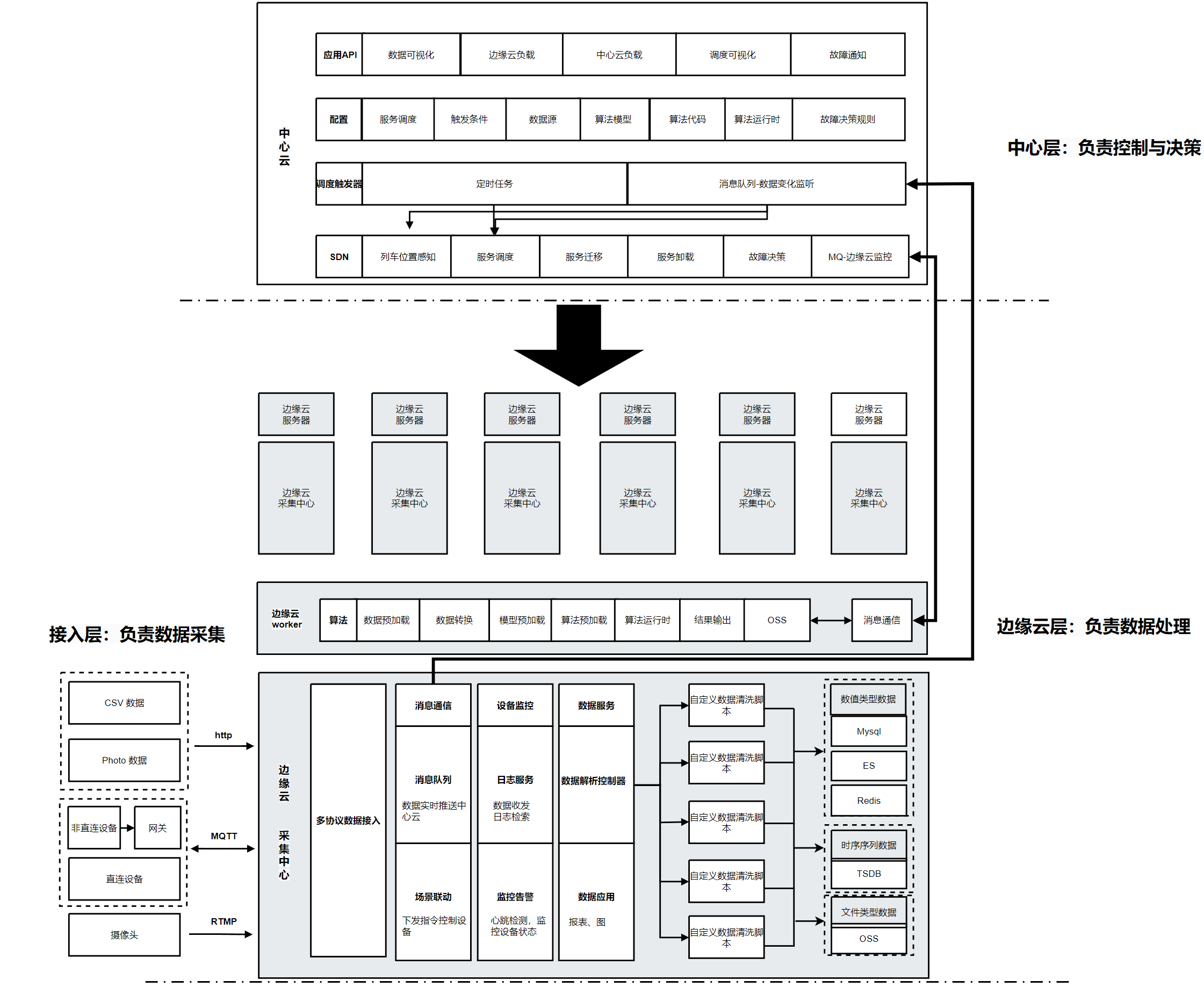


图1

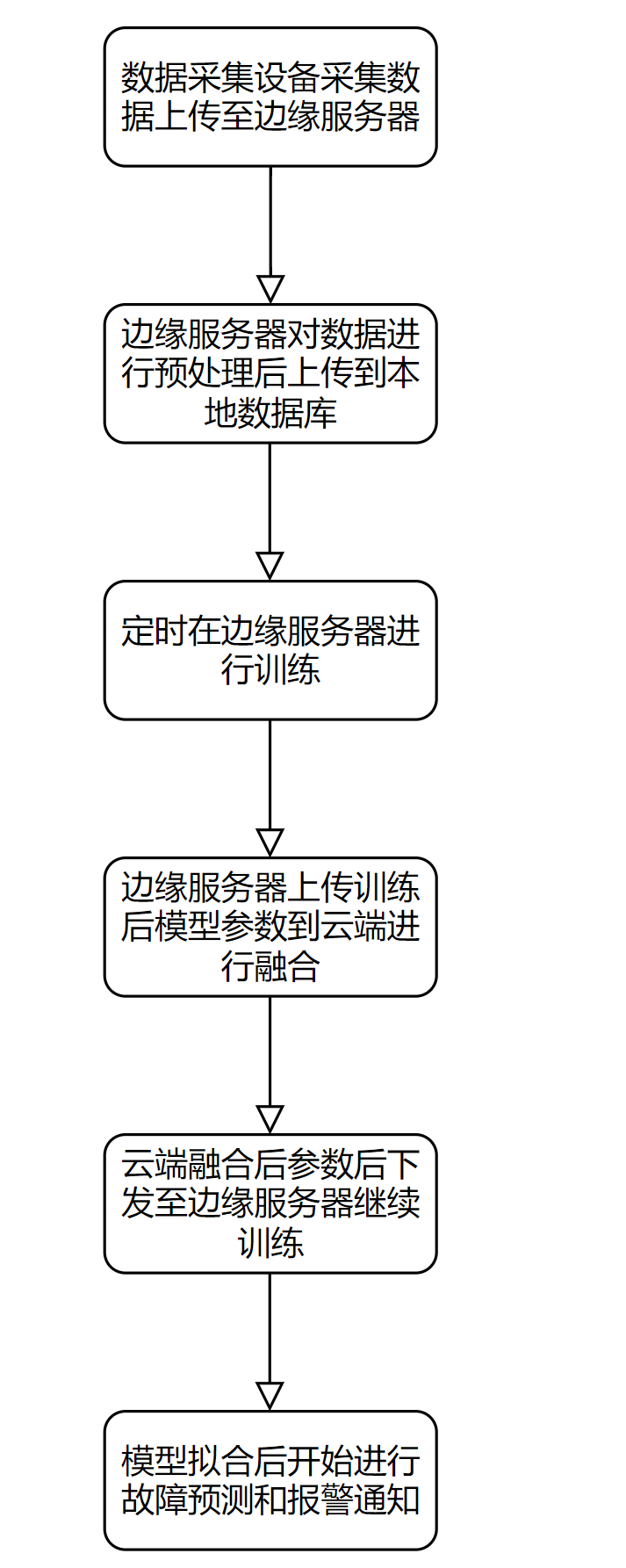


图2

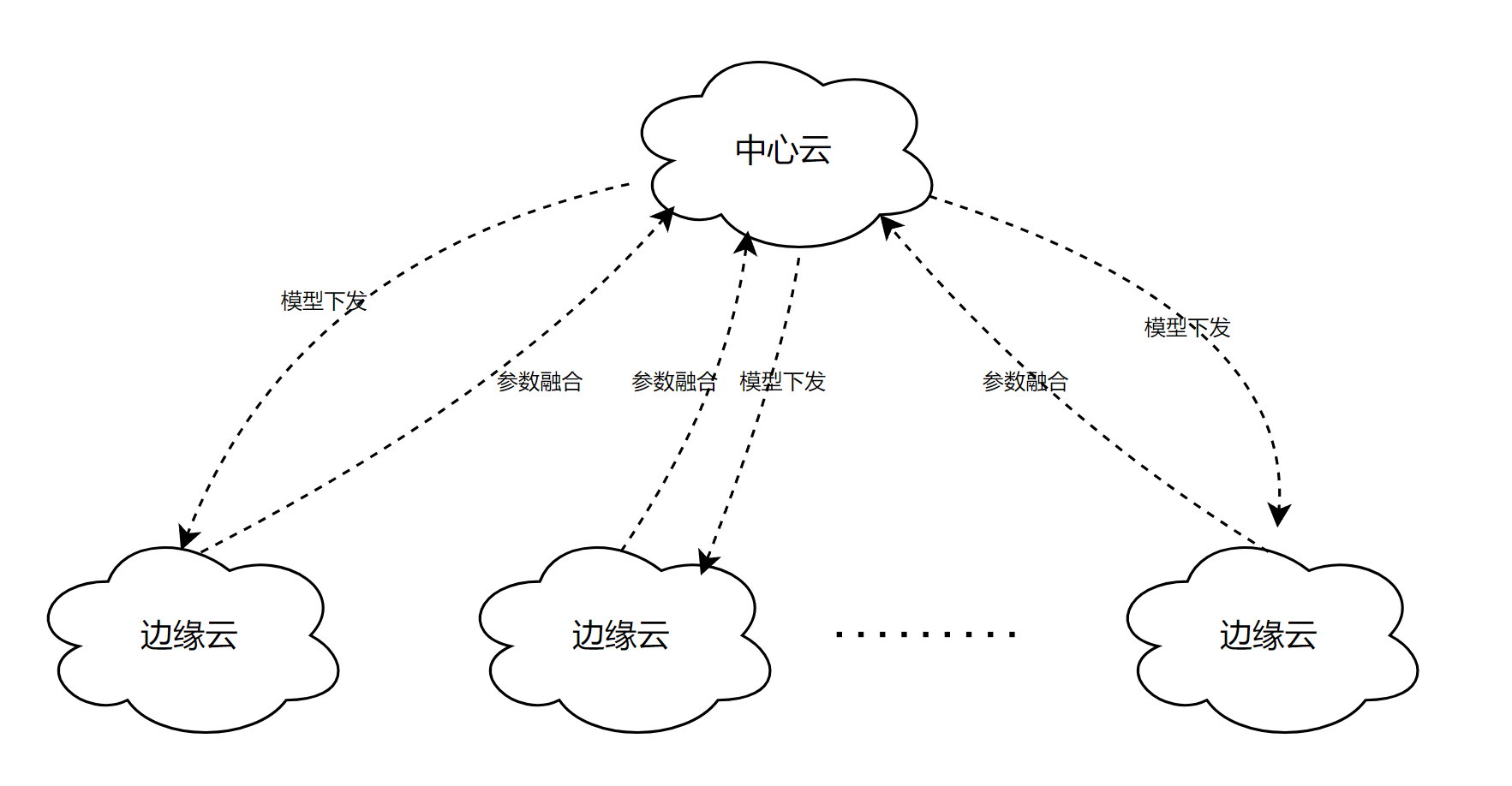


图3