이기종 엣지 디바이스 환경에서의 연합학습 기반 의료 영상 분류 시스템 구현

Implementation of Federated Learning-based Medical Image Classification System in Heterogeneous Edge Device Environment

이규민(Lee Kumin)

[1] 한국외국어대학교, 컴퓨터전자시스템공학부; steve918@naver.com

한글 요약: 본 연구는 이기종 Raspberry Pi 환경에서 연합학습(Federated Learning)을 활용하여 의료 영상 분류 모델을 구현하고 평가하였다. 실제 의료 기관의 다양한 하드웨어 환경을 모사하기 위해, 동일 아키텍처 내에서도 연산 성능과 메모리 용량의 차이가 뚜렷한 Raspberry Pi 3 종(Pi 5 8GB, Pi 5 4GB, Pi 4B 2GB)을 사용하여 시스템적 이질성(System Heterogeneity) 환경을 구축하였다. 데이터셋은 PathMNIST 대장 조직 병리 이미지를 Dirichlet 분포(α=0.5)로 Non-IID 분할하여 사용하였으며, 제한된 리소스 환경을 고려하여 경량 CNN 모델(약 100K 파라미터)을 설계하였다.

FedAvg 와 FedProx 알고리즘 비교 실험 결과, FedAvg 는 82.44%의 정확도와 0.7749 의 F1 Score(Macro)를 달성하여 우수한 성능을 입증하였다. 특히, 실제 엣지 디바이스 배포 단계에서의 추론 병목 현상을 해결하기 위해 ONNX(Open Neural Network Exchange) Runtime 을 적용하였다. 그 결과, 모델의 정확도 손실 없이 클라이언트 기기 성능에 따라 PyTorch 대비 1.29 배에서 최대 1.44 배의 추론 속도 향상을 달성하였다. 이는 고성능 서버 없이도 리소스가 제한된 이기종 엣지 디바이스 환경에서 실시간 의료 영상 분석 시스템이 효율적으로 운용될 수 있음을 시사한다.

핵심어: 연합학습, 엣지 디바이스, 의료 영상 분류, ONNX 추론 최적화

영문 요약: This study implements and evaluates a federated learning-based medical image classification system within a heterogeneous Raspberry Pi environment. To simulate the diverse hardware conditions of real-world medical institutions, we established an environment with system heterogeneity using three types of Raspberry Pi models (Pi 5 8GB, Pi 5 4GB, Pi 4B 2GB), which exhibit distinct differences in computational performance and memory capacity despite sharing the same architecture. We utilized the PathMNIST colorectal histology image dataset, partitioned under Non-IID conditions using a Dirichlet distribution (α=0.5), and employed a lightweight CNN model (approx. 100K parameters) to accommodate resource constraints.

In comparative experiments between FedAvg and FedProx algorithms, FedAvg demonstrated superior performance, achieving 82.44% accuracy and an F1 Score (Macro) of 0.7749. To address inference bottlenecks in real-world edge deployment, we applied ONNX (Open Neural Network Exchange) Runtime optimization. This approach achieved an inference speedup ranging from 1.29x to 1.44x across heterogeneous client devices without any degradation in model accuracy compared to PyTorch. These results suggest that real-time medical image analysis systems can be efficiently operated in resource-constrained heterogeneous edge environments without relying on high-performance central servers.

Keywords: Federated Learning, Edge Device, Medical Image Classification, ONNX Inference Optimization

본 논문은 2026 학년 2 월 졸업을 위해 제출된 한국외국어대학교 컴퓨터공학부 졸업논문이다.
2025.12.17

지도교수: 지수연

서명:

참여한 캡스톤 설계 (해당자만)

설계명: 의료 영상 향상 알고리즘 개발

팀원명: 이규민, 이서화, 이성민, 이수경, 박주희

1. 서론

인공지능 기술의 발전으로 의료 영상 분석에서 딥러닝 활용이 확대되고 있으나, 의료 데이터는 HIPAA 와 GDPR 등의 엄격한 개인정보 보호 규제 하에 있어 중앙집중형 학습 방식의 적용이 제한적이다[1,12].

연합학습(Federated Learning)은 각 클라이언트가 로컬 데이터로 모델을 학습하고 모델 파라미터만 공유하여, 데이터의 물리적 이동 없이 글로벌 모델을 학습하는 분산 학습 방식이다[2,6,7]. McMahan 등[2]이 제안한 FedAvg 알고리즘과 Li 등[3]의 FedProx 는 Non-IID 환경에서의 수렴 안정성을 개선하고자 하였으나, 최근 연구에서는 극단적 Non-IID 환경에서 FedProx 가 항상 우수하지 않다는 결과도 보고되었다[15].

한편, 이러한 연합학습 알고리즘들은 주로 시뮬레이션 환경에서 검증되어 왔으며, 실제 자원 제약이 있는 엣지 환경에서의 실증 연구는 부족한 실정이다. Lian 등[10]은 COVID-19 CT 영상을 활용한 DEEP-FEL 을 제안하였으나 고성능 GPU 시뮬레이션에서 검증되었다. Feng 등[11]은 Raspberry Pi 와 Jetson Nano 를 활용한 물리적 테스트베드를 구축하였으나 IID 환경만을 대상으로 하였다. Li 등[8]은 Non-IID 환경에서의 연합학습을 체계적으로 분석하며 Dirichlet 분포 기반 데이터 분할의 유효성을 입증하였다. 이처럼 기존 연구는 (1) 의료 영상 + 시뮬레이션, 또는 (2) 실제 엣지 + IID 환경으로 이분화되어 있으며, 이기종 엣지 디바이스에서 극단적 Non-IID 의료 데이터를 다룬 연구는 부족하다.

ONNX(Open Neural Network Exchange)는 프레임워크 독립적인 모델 표현 방식으로, ONNX Runtime 과 ONNX Simplifier 를 통해 추론 단계의 성능 최적화가 가능하다[13,14,15]. 본 연구는 이질적 Raspberry Pi 클러스터 환경에서 연합학습 알고리즘을 비교하고, ONNX 기반 최적화를 통해 실용적 배포 가능성을 검증한다.

본 연구의 목적은 다음과 같다:

1. 이질적 Raspberry Pi 클러스터로 실제 의료 기관의 다양한 하드웨어 환경 재현
2. 극단적 Non-IID 환경(Dirichlet α=0.5)에서 FedAvg 와 FedProx 성능 비교
3. ONNX 최적화를 통한 추론 속도, 메모리 효율성 개선 효과 검증
4. 개인정보 보호와 엣지 제약을 만족하는 연합학습 시스템의 실용성 평가

본 논문은 다음과 같이 구성된다. 2 장에서는 실험 환경과 방법론을 기술한다. 3 장에서는 실험 결과를 분석하고, 4 장에서 논의 및 결론을 제시한다.

2. 연구 방법 및 결과

2.1 실험 환경 및 데이터셋

2.1.1 하드웨어 및 소프트웨어 환경

본 연구에서는 실제 병원 환경의 이기종 엣지 디바이스를 모사하기 위해 서로 다른 사양의 Raspberry Pi 를 활용하였다. 표 1 은 사용된 하드웨어의 사양을 보여준다.

표 1: 시스템 사양

역할	기기	사양
Client0	Raspberry Pi 5	8GB RAM, Quad-core ARM CPU
Client1	Raspberry Pi 5	4GB RAM, Quad-core ARM CPU
Client2	Raspberry Pi 4B	2GB RAM, Quad-core ARM CPU
Server	MacBook Pro M1 Pro	16GB RAM, 10-core CPU

모든 기기는 정적 IP 를 통해 로컬 네트워크로 연결되었으며, 서버는 동일 라우터를 이용한 고정 주소로 클라이언트의 연결을 수신하였다.

실험에 사용된 주요 소프트웨어는 Python 3.9, PyTorch 2.7.(딥러닝 프레임워크), Flower 1.22(연합학습 프레임워크), MedMNIST 3.0.2(의료 영상 데이터셋 라이브러리), scikit-learn(평가 지표 계산), ONNX 1.19.1(모델 포맷 변환), ONNX Runtime 1.23.2(추론 최적화 엔진)이다.

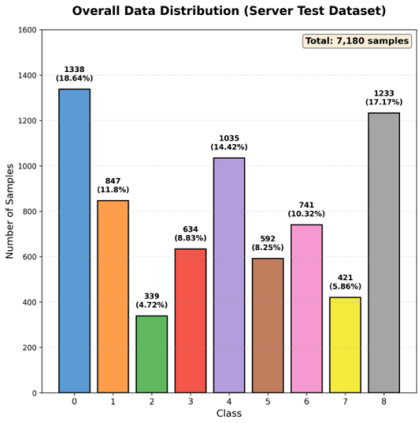
Raspberry Pi 의 제한된 리소스를 고려하여 모든 학습은 CPU 만을 사용하였으며, DataLoader 의 num_workers 는 0 으로 설정하여 멀티프로세싱으로 인한 메모리 문제를 방지하였다.

2.1.2 PathMNIST 데이터셋

본 연구는 MedMNIST v2 컬렉션[5]의 PathMNIST 데이터셋을 사용하였다. PathMNIST 는 NCT-CRC-HE-100K 데이터셋을 기반으로 하며, 대장 조직 병리 이미지로 구성된 9 개 클래스 분류 작업이다. 9 개 클래스는 Adipose(지방), Background(배경), Debris(세포 파편), Lymphocytes(림프구), Mucus(점액), Smooth Muscle(평활근), Normal Colon Mucosa(정상 점막), Cancer-associated Stroma(암 관련 기질), Colorectal Adenocarcinoma Epithelium(대장 선암 상피)으로 구성된다.

데이터셋은 훈련 데이터 89,996 개, 검증 데이터 10,004 개, 테스트 데이터 7,180 개로 구성되며, 이미지 크기는 28×28 픽셀(RGB)이다. 특히, 훈련 데이터는 독일 NCT(National Center for Tumor Diseases)에서 86 명의 환자로부터 수집되었고, 테스트 데이터(CRC-VAL-HE-7K)는 별도의 임상 기관에서 50 명의 환자로부터 수집되었다[5]. 이로 인해 훈련 데이터와 테스트 데이터 간 클래스 분포 차이가 존재하며, 이는 모델의 외부 일반화 능력을 평가하기 위한 의도적 설계이다.

그림 1: PathMNIST Test Data 분포



2.1.3 연합학습 프레임워크 (Flower)

본 연구에서는 이기종 엣지 디바이스 환경에서의 효율적인 연합학습 시스템 구축을 위해 오픈소스 프레임워크인 **Flower(Flwr)**를 채택하였다. Flower 는 모바일 및 엣지 디바이스부터 대규모 클라우드 서버까지 다양한 환경을 지원하는 플랫폼 독립적인(Device-agnostic) 특성을 가진다.

기존의 연합학습 시뮬레이션 도구들과 달리, Flower 는 실제 배포 환경과 유사한 gRPC 기반의 통신 프로토콜을 사용하여 클라이언트와 서버 간의 안정적인 파라미터 교환을 지원한다. 또한, PyTorch, TensorFlow 등 다양한 딥러닝 라이브러리와의 호환성이 뛰어나며, 사용자 정의 전략(Strategy)을 통해 본 연구에서 비교 분석한 FedAvg 및 FedProx 알고리즘을 유연하게 구현할 수 있다는 장점이 있다. 본 실험에서는 Raspberry Pi 와 같은 저사양 ARM 기반 아키텍처에서도 원활하게 동작하는 Flower 의 경량화된 클라이언트 런타임을 활용하였다.

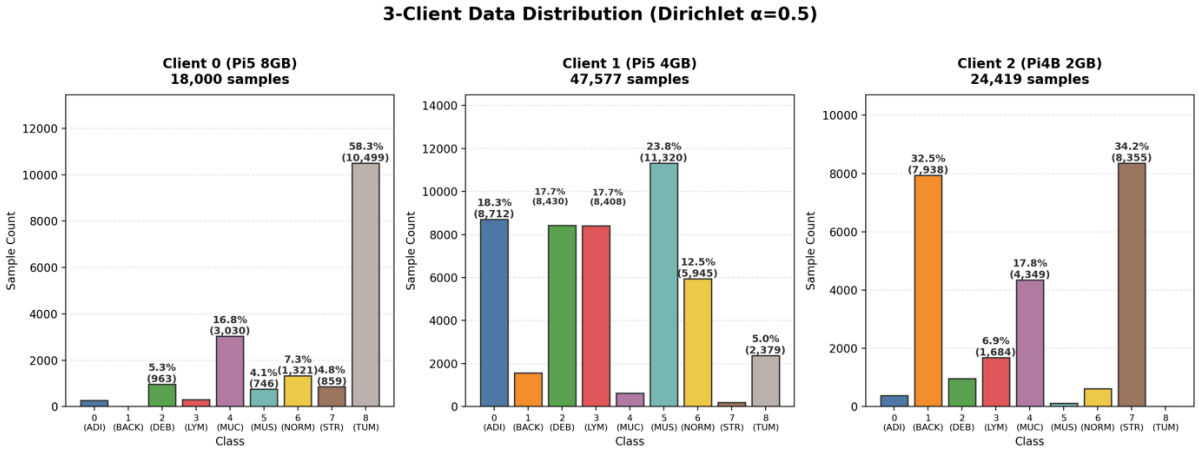
2.1.4 Non-IID 데이터 분할

실제 의료 환경에서는 각 병원이 보유한 데이터의 분포가 다를 수 있다. 이러한 이질적 데이터 환경을 모사하기 위해 Dirichlet 분포(α=0.5)를 사용하여 Non-IID 데이터 분할을 수행하였다[8,9]. 표 2 는 클라이언트별 데이터 분포를 보여준다.

표 2: 클라이언트별 데이터 분포

Client	총 데이터	비율	특징
Client0	18,000 개 (Train: 14,400, Val: 3,600)	20.0%	내부 데이터가 Class 8(58.3%)로 편향
Client1	47,577 개 (Train: 38,061, Val: 9,516)	52.9%	가장 많은 데이터 소유, 가장 큰 영향력
Client2	24,419 개 (Train: 19,535, Val: 4,884)	27.1%	전체 데이터 중 Class 7(88.9)% 보유

그림 2: 클라이언트별 데이터 분포



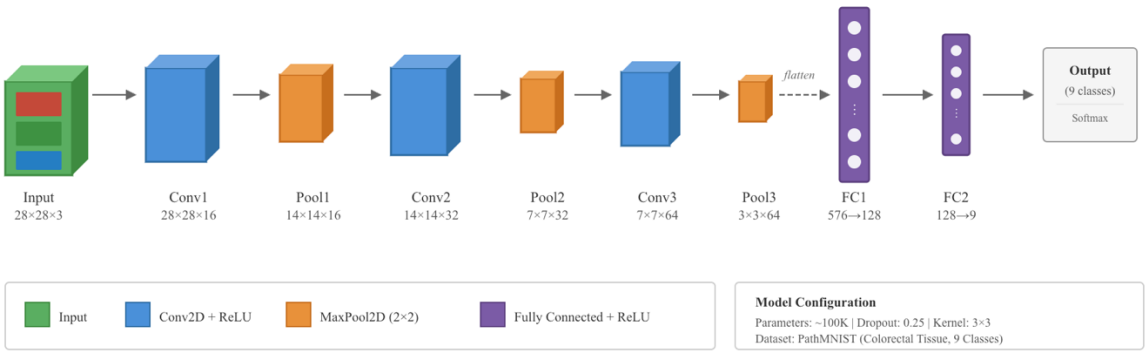
초기 실험에서는 테스트 데이터도 클라이언트별로 Non-IID 분할하였으나, 이는 평가 지표의 편향을 야기하였다. 따라서 본 연구는 다음과 같은 평가 전략을 채택하였다: (1) 훈련 데이터는 Dirichlet(α=0.5)로 Non-IID 분할, (2) 검증 데이터는 각 클라이언트의 훈련 데이터에서 20% 분리(로컬 평가용), (3) 테스트 데이터는 분할하지 않고 서버에서 글로벌 평가에 사용. 이를 통해 편향되지 않은 글로벌 성능 측정이 가능하며, 모델의 진정한 일반화 능력을 평가할 수 있다.

2.2 모델 및 연합학습 알고리즘

2.2.1 SimpleCNN 모델 아키텍처

Raspberry Pi 의 제한된 연산 능력을 고려하여 경량 CNN 모델을 설계하였다. SimpleCNN 은 3 개의 Convolutional layer 와 2 개의 Fully Connected layer 로 구성되며, 총 파라미터 수는 약 100K 개이다. 구체적인 구조는 다음과 같다:

그림 3: SimpleCNN 모델 구조



과적합 방지를 위해 **Dropout 0.25** 를 적용하였으며, 모든 Convolutional layer 와 첫 번째 FC layer 에는 ReLU 활성화 함수를 사용하였다.

2.2.2 하이퍼파라미터 설정

실험에 사용된 하이퍼파라미터는 다음과 같다: 전체 라운드 수 20, 로컬 에포크 수 5, 배치 크기 16, Optimizer 는 SGD(momentum=0.9)를 사용하였다. 학습률은 FedAvg 의 경우 0.001 을 기본값으로 하였으며, 비교 실험으로 0.002 도 테스트하였다. 학습률 스케줄러는 StepLR 을 사용하여 10 라운드마다 학습률을 0.5 배 감소시켰다[3]. 모든 클라이언트에서 동일한 배치 크기 16 을 유지하여 알고리즘 간 공정한 비교를 수행하였다.

클래스 불균형을 완화하기 위해 가중 손실 함수를 적용하였다. 각 클라이언트에서 클래스별 가중치는 $\omega_i = \frac{n_k}{C*c_i}$ 로 계산하였으며, 여기서 n_k는 클라이언트 k 의 총 샘플 수, C 는 전체 클래스 수(9), c_i는 클래스 i 의 샘플 수이다. 또한 극단적 불균형 환경에서 학습 안정성을 위해 gradient clipping(max_norm=1.0)을 적용하였다.

평가 지표로는 Accuracy, F1 Score(Macro), F1 Score(Weighted)를 사용하였다. 클래스 불균형이 존재하는 데이터셋에서는 Accuracy 만으로는 모델 성능을 정확히 평가하기 어려우므로, Macro F1 Score(각 클래스의 F1 Score 를 산술 평균)와 Weighted F1 Score(각 클래스의 F1 Score 를 샘플 수로 가중 평균)를 추가로 사용하였다.

2.2.3 연합학습 알고리즘

FedAvg 알고리즘

FedAvg 는 각 클라이언트가 로컬 데이터로 E epochs 학습한 후, 서버가 샘플 수에 비례한 가중 평균으로 글로벌 모델을 집계한다:

$$\omega^{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \omega_k^{t+1}$$

여기서 ω^{t+1}은 라운드 t+1 의 글로벌 모델 파라미터, ω_k^{t+1}은 클라이언트 k 의 로컬 모델 파라미터, n_k 는 클라이언트 k 의 샘플 수, n 은 전체 샘플 수이다.

FedProx 알고리즘

FedProx 는 로컬 학습 시 proximal term 을 추가하여 로컬 모델이 글로벌 모델에서 너무 멀리 벗어나지 않도록 제한한다:

$$minF_k(\omega) + \frac{\mu}{2} || \omega - \omega^t||^2$$

$F_k(\omega)$는 클라이언트 k 의 원래 손실 함수(Cross Entropy), ω^t는 현재 라운드 t 의 글로벌 모델 파라미터, μ 는 proximal term 의 강도를 조절하는 하이퍼파라미터이다. 본 연구에서는 μ=0.01, 0.1, 0.3 의 세 가지 값을 실험하였다.

2.2.4 ONNX 기반 모델 최적화

엣지 디바이스 환경에서의 실용적인 배포를 위해 ONNX(Open Neural Network Exchange) 런타임 기반 모델 최적화를 적용하였다. ONNX 는 서로 다른 딥러닝 프레임워크 간의 상호 운용성을 제공하는 개방형 포맷으로, 특히 추론(inference) 단계에서의 성능 최적화에 강점을 가진다.

2.3 ONNX 변환 및 최적화 과정

본 연구에서는 PyTorch 로 학습된 SimpleCNN 모델을 ONNX 포맷으로 변환하고, ONNX Simplifier 를 통한 추가 최적화를 수행하였다. 구체적인 변환 과정은 다음과 같다:

2.3.1 ONNX 변환

1 단계: PyTorch to ONNX 변환

각 연합학습 라운드가 종료되고 글로벌 모델 파라미터가 클라이언트에 전달될 때마다, 클라이언트는 업데이트된 PyTorch 모델을 ONNX 포맷으로 변환한다. 이 과정에서 상수 연산을 사전 계산하여 그래프를 단순화하고, dynamic_axes 를 설정하여 다양한 배치 크기에 대응할 수 있도록 하였다.

2 단계: ONNX Simplifier 를 통한 그래프 최적화

ONNX Simplifier 는 연산 그래프를 분석하여 불필요한 노드를 제거하고, 연속된 연산을 병합하며, 중복 계산을 제거하는 등의 최적화를 수행한다. 이를 통해 모델 크기 감소와 함께 추론 속도 향상을 기대할 수 있다.

3 단계: ONNX Runtime 을 통한 추론

최적화된 ONNX 모델은 ONNX Runtime 을 통해 실행되며, CPU 전용 실행 환경에서도 효율적인 추론이 가능하도록 CPUExecutionProvider 를 사용하였다.

2.3.2 ONNX 최적화

ONNX 최적화는 연합학습의 평가(evaluation) 단계에만 적용되었으며, 학습(training) 과정은 기존과 동일하게 PyTorch 로 수행되었다. 이는 ONNX 가 추론 최적화에 특화되어 있으며, 역전파(backpropagation)를 지원하지 않기 때문이다. 구체적인 워크플로우는 다음과 같다:

1. **학습 단계**: PyTorch 를 사용하여 로컬 데이터셋으로 모델 학습
2. **모델 집계**: 서버에서 FedAvg 알고리즘으로 글로벌 모델 생성
3. **모델 배포**: 업데이트된 글로벌 모델을 각 클라이언트에 전송
4. **ONNX 변환**: 각 클라이언트에서 PyTorch 모델을 ONNX 로 변환 및 최적화
5. **평가 단계**: ONNX Runtime 을 사용하여 검증 데이터셋으로 모델 평가

이러한 접근 방식은 학습 과정의 안정성을 유지하면서도, 자원이 제한된 엣지 디바이스에서 빠른 평가가 가능하도록 한다는 장점이 있다.

3. 결과 분석

3.1 실험 결과

3.1.1 FedAvg 결과

FedAvg (LR=0.001)

FedAvg 알고리즘을 학습률 0.001 로 실행한 결과, 최종 정확도 82.44%를 달성하였다. Round 1 에서 64.72%로 시작하여 Round 20 에서 82.44%로 향상되어 총 17.72%p 의 정확도 증가를 보였다. 평균 정확도는 79.57%였으며, 최고 정확도는 Round 19 에서 83.18%를 기록하였다. F1 Score 는 Macro 0.7749, Weighted 0.8282 를 달성하였다.

클래스별 최종 정확도는 다음과 같다: Class 0 (95.6%), Class 1 (100.0%), Class 2 (83.5%), Class 3 (92.4%), Class 4 (81.5%), Class 5 (61.5%), Class 6 (91.6%), Class 7 (38.5%), Class 8 (71.0%).

FedAvg (LR=0.002) - 비교 실험

학습률을 0.002 로 증가시킨 비교 실험에서는 최종 정확도 81.52%, F1 Score Macro 0.7612, Weighted 0.8184 를 기록하였다. 초기 정확도는 65.01%로 시작하여 평균 정확도는 79.98%를 보였다. 주요 클래스별 정확도는 Class 2 (76.4%), Class 5 (71.3%), Class 7 (31.1%),로 변화하였다.

3.1.2 FedProx 결과

FedProx (μ=0.01)

FedProx 를 μ=0.01 로 실행한 결과, 최종 정확도 81.53%를 달성하였다. Round 1 에서 61.07%로 시작하여 Round 20 에서 81.53%로 향상되어 총 20.46%p 의 정확도 증가를 보였다. 평균 정확도는 78.52%였으며, 최고 정확도는 Round 9 에서 83.75%를 기록하였다. F1 Score 는 Macro 0.7657, Weighted 0.8167 을 달성하였다.

클래스별 정확도는 다음과 같다: Class 0 (95.4%), Class 1 (100.0%), Class 2 (86.7%), Class 3 (92.7%), Class 4 (83.8%), Class 5 (62.7%), Class 6 (89.3%), Class 7 (35.2%), Class 8 (65.0%). FedAvg 와 유사하게 Class 7 에서 가장 낮은 성능(35.2%)을 보였다.

FedProx (μ=0.1)

μ 값을 0.1 로 증가시킨 실험에서는 최종 정확도 78.57%, F1 Score Macro 0.7240, Weighted 0.7927 을 기록하였다. 주요 클래스별 정확도는 Class 2 (44.0%), Class 7 (37.3%), Class 5 (56.8%)로, μ=0.01 대비 전반적인 성능 저하가 관찰되었다.

FedProx (μ=0.3)

μ 값을 0.3 으로 더욱 증가시킨 실험에서는 최종 정확도 69.97%, F1 Score Macro 0.6318, Weighted 0.6840 으로 가장 낮은 성능을 보였다. 주요 클래스별 정확도는 Class 2 (73.2%), Class 7 (9.0%), Class 5 (51.5%)로, 특히 Class 7 에서 9.0%의 매우 낮은 정확도를 기록하였다.

3.1.3 알고리즘 성능 종합 비교

표 3 은 모든 실험 결과를 종합적으로 비교한 것이다.

표 3: 실험 성능 정리 비교

알고리즘	Accuracy	F1 (Macro)	F1 (Weighted)	Class 7 Accuracy (고난도 케이스)
중앙 집중형 (simple CNN)	85.29%	0.8014	0.8588	39.27%
중앙 집중형 (ResNet 18)	87.45%	0.8213	0.8765	43.2%
IID-FedAvg (LR=0.001)	83.33%	0.7758	0.8343	43.47%
FedAvg (LR=0.001)	82.44%	0.7749	0.8282	38.5%
FedAvg (LR=0.002)	81.52%	0.7612	0.8184	31.1%
FedProx (μ=0.01)	81.53%	0.7657	0.8167	35.2%
FedProx (μ=0.1)	78.57%	0.7240	0.7927	37.3%
FedProx (μ=0.3)	69.97%	0.6318	0.6840	9.0%

전체적으로 Non-IID 데이터 연합학습에서는 FedAvg (LR=0.001)이 모든 평가 지표에서 가장 우수한 성능을 보였다. FedProx 는 μ 값이 증가할수록 전체 성능이 저하되는 경향을 보였으며, 특히 μ=0.3 에서는 심각한 성능 저하가 관찰되었다. 모든 실험에서 Class 7 은 지속적으로 낮은 정확도(9.0% ~ 43.2%)를 기록하였다.

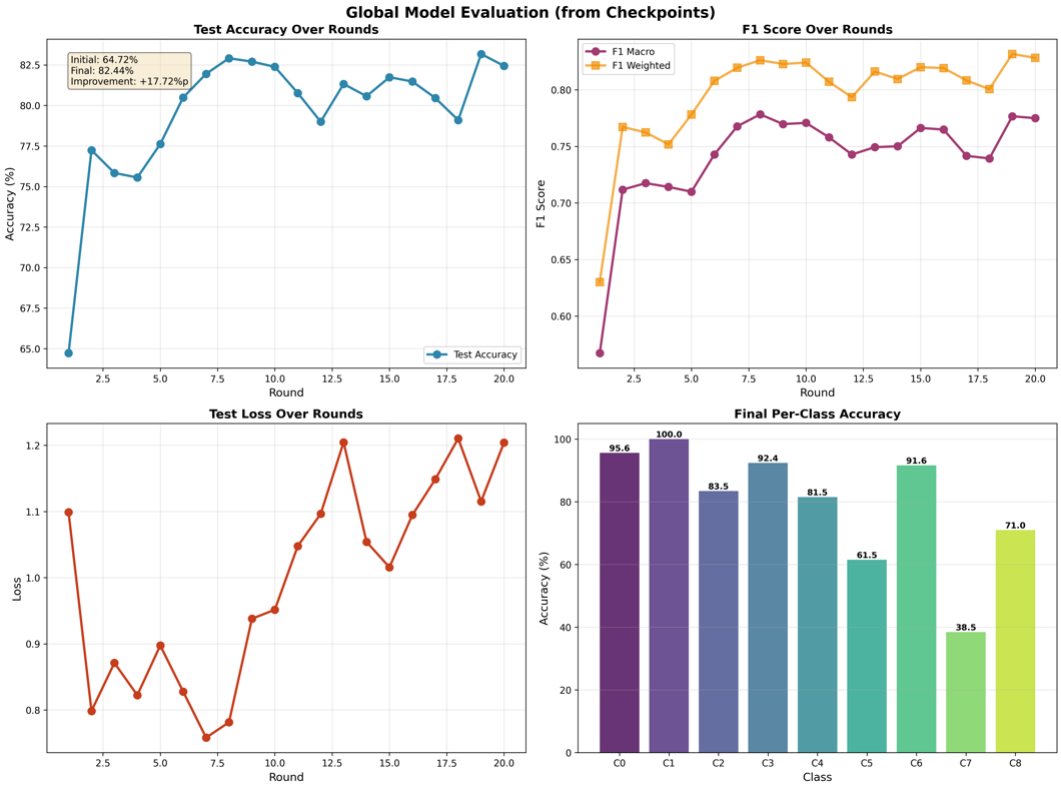
3.2 알고리즘 성능 비교

3.2.1 FedAvg 의 높은 성능

표 3 에서 확인할 수 있듯이, FedAvg (LR=0.001)이 전체 정확도 82.44%, F1 Score Macro 0.7749, Weighted 0.8282 로 모든 평가 지표에서 가장 우수한 성능을 보였다. 학습률 비교 실험에서는 LR=0.001 이 LR=0.002 보다 우수한 성능을 나타냈다(82.44% vs 81.52%). 이는 Non-IID 환경에서 과도하게 높은 학습률이 로컬 최적화를 불안정하게 만들 수 있음을 본 실험 결과에서 확인되었다.

FedAvg 는 20 라운드 동안 안정적인 수렴을 보였으며, Round 1 에서 64.72%로 시작하여 지속적으로 향상되어 Round 19 에서 최고 정확도 83.18%를 달성하였다. 평균 정확도 79.57%는 대부분의 라운드에서 일관된 성능을 유지했음을 보여준다.

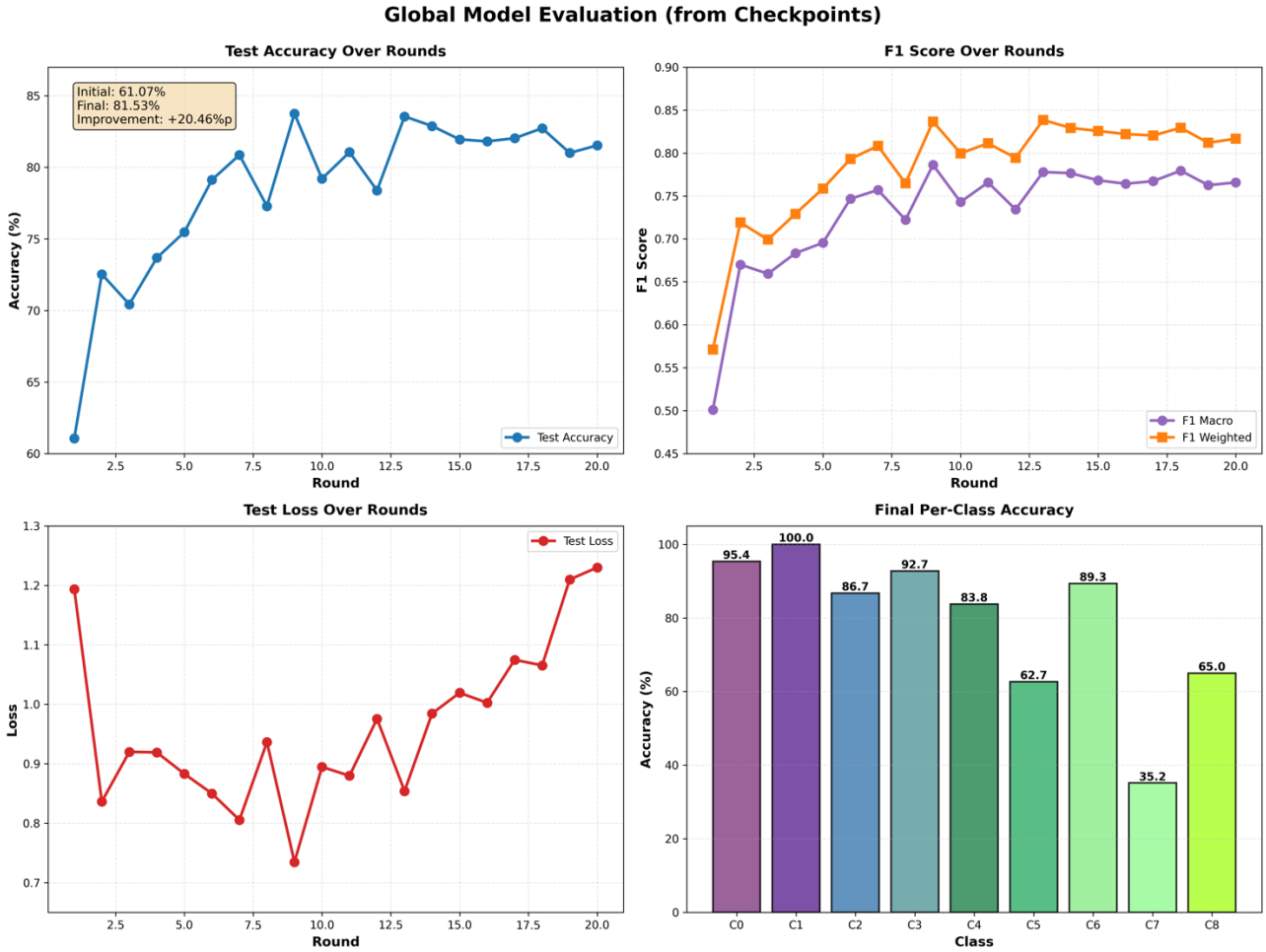
그림 4: FedAvg(LR=0.001) 결과 그래프



3.2.2 FedProx 의 한계

FedProx 는 μ 값이 증가할수록 전체 성능이 저하되는 경향을 보였다. μ=0.01 에서는 81.53%로 FedAvg 와 유사한 성능을 보였으나, μ=0.1 에서 78.57%, μ=0.3 에서 69.97%로 급격히 감소하였다. 이는 본 실험의 α=0.5 Non-IID 환경에서는 proximal term 이 오히려 역효과를 낸 것으로 해석된다.

그림 5: FedProx(mu=0.1) 결과 그래프



FedProx 의 proximal term $\frac{\mu}{2} || \omega - \omega^t ||^2$은 로컬 모델이 글로벌 모델에서 벗어나는 것을 제한하여 극단적 Non-IID 환경에서 안정성을 높이도록 설계되었다[3]. 그러나 μ 값이 클수록 로컬 최적화가 과도하게 제한되어, 각 클라이언트가 자신의 데이터 분포에 적합한 특징을 충분히 학습하지 못하게 된다. 특히 α=0.5 는 중간 정도의 Non-IID 환경으로, 클라이언트들이 일정 수준의 로컬 최적화를 수행할 필요가 있는데, 높은 μ 값은 이를 방해한 것으로 판단된다.

FedProx (μ=0.01)은 초기 라운드에서 61.07%로 FedAvg(64.72%)보다 낮게 시작했지만, 더 큰 향상폭(+20.46%p vs +17.72%p)을 보였다. 그러나 최종 성능에서는 FedAvg 에 미치지 못했으며, 최고 정확도는 Round 9 에서 조기에 달성된 후 소폭 하락하는 양상을 보였다.

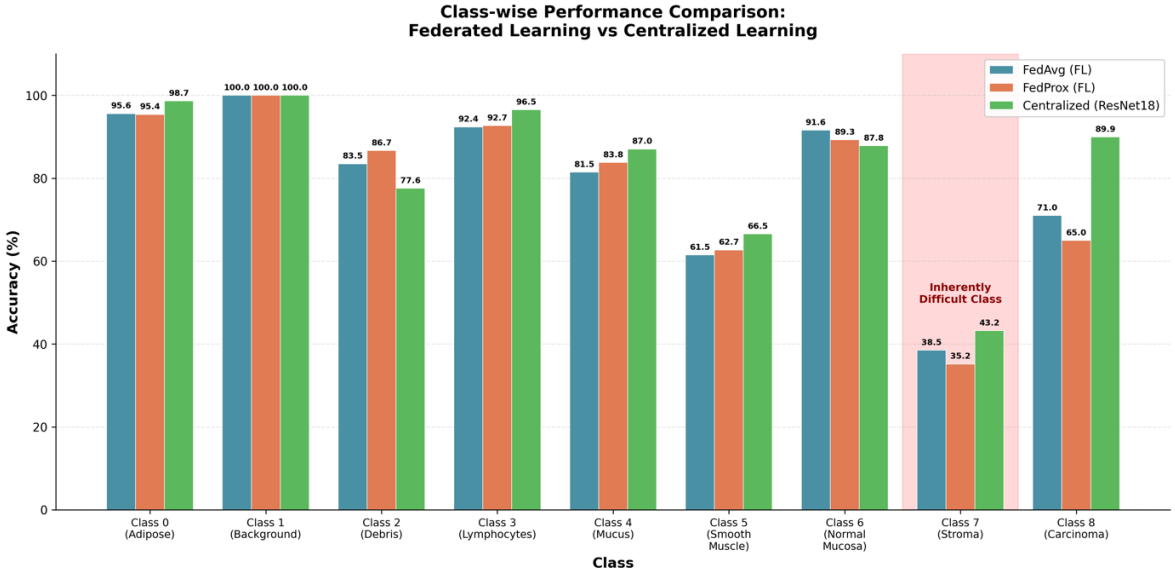
3.2.3 Class 7 성능 저하 원인 분석

모든 실험에서 Class 7(STR, Stroma)은 최종 결과에서 9.0%~38.5%의 낮은 정확도를 기록하였다. 이러한 성능 저하의 원인을 다각도로 분석한 결과, 세 가지 요인이 복합적으로 작용함을 확인하였다.

본질적 판별 난이도

원인 규명을 위해 동일한 데이터셋과 모델 구조를 사용하여 중앙집중형 학습(Centralized Learning)을 수행하였다. 그 결과, 중앙집중형 환경에서도 Class 7 은 SimpleCNN 기준 39.27%, ResNet18 기준 43.2%로 다른 클래스 대비 현저히 낮은 정확도를 보였다.(그림 6 참조)

그림 6: 중앙집중형과 연합학습 클래스별 정확도 비교



이는 시각적 특징이 유사하여 본질적으로 판별이 어려운 특성을 가진 데이터임을 시사한다.

Train-Test 분포 불일치와 연합학습 집계 메커니즘의 상호작용

그러나 본질적 난이도만으로는 설명되지 않는 현상이 관찰되었다. 표 4 와 같이 Client 2 의 STR 정확도는 Round 7 에서 68.70%의 정점을 기록한 후, Round 20 에서 53.80%로 하락하였다.

표 4: Client2 의 Class7(STR) 라운드별 성능 변화(FedAvg)

Round	Local Accuracy	F1 Score
1	0.78%	0.0156
7	68.70%	0.8042
10	61.94%	0.7599
15	44.05%	0.6098
20	53.80%	0.6953

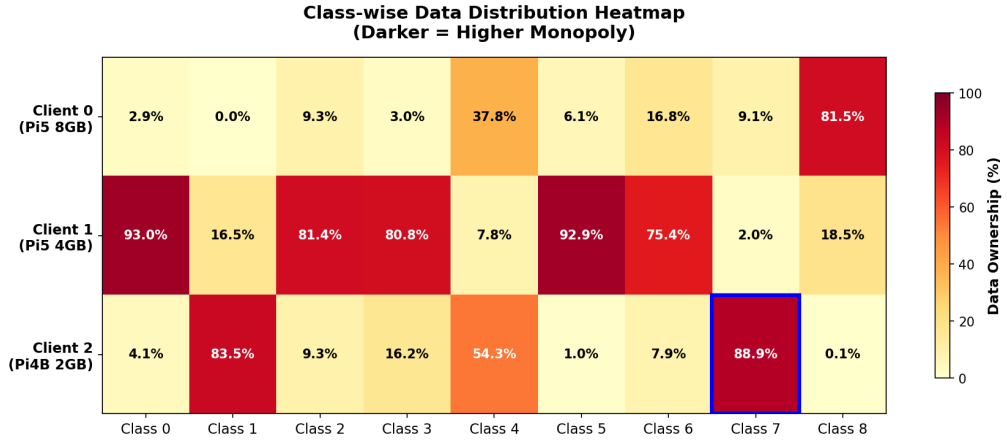
이러한"상승 후 하락" 패턴은 PathMNIST 의 Train-Test 분포 불일치에서 기인한다. PathMNIST 는 Train set 과 Test set 이 서로 다른 임상 기관에서 수집되어[5], STR 클래스의 경우 Train set 에서 10.4%를 차지하지만 Test set 에서는 5.9%에 불과하다. 연합학습에서 글로벌 모델은 서버의 테스트셋을 기준으로 평가되는데, Client 2 가 STR 데이터의 88.9%를 보유하고 있음에도 27.1%의 aggregation weight 만 부여받아, 로컬에서 학습한 STR 특화 지식이 집계 과정에서 희석된다.

표 5: PathMNIST Train-Test 차이가 가장 큰 Class

Class	Train 비율	Test 비율	차이
ADI (Class 0)	10.4%	18.6%	+8.2%p
DEB (Class 2)	11.5%	4.7%	-6.8%p
STR (Class 7)	10.4%	5.9%	-4.5%p

이로 인해 Client2 가 두 번째로 많은 데이터를 소유하고 있음에도 가장 적은 데이터를 소유한 Client0 보다 성능 평가시 Accuracy 가 낮은 것을 볼 수 있다(표 6 참고).

그림 7: Client 데이터 소유 히트맵



종합 분석

결과적으로 라운드가 진행될수록 글로벌 모델은 테스트셋 분포에 점진적으로 수렴하게 되며, 이 과정에서 특정 클라이언트가 보유한 소수 클래스에 대한 학습 성과가 약화되는 Local-Global Distribution Mismatch 현상이 발생했다.

따라서 연합학습 환경에서의 Class 7 성능 저하는 (1) 데이터 자체의 높은 판별 난이도, (2) Non-IID 분할로 인한 데이터 독점 문제, (3) Train-Test 분포 불일치로 인한 글로벌 모델 집계 과정에서의 로컬 학습 성과 희석이 복합적으로 작용한 결과로 분석된다. 이는 연합학습 시스템 설계 시 데이터 분포의 이질성뿐만 아니라 평가 기준이 되는 테스트셋의 분포 특성도 함께 고려해야 함을 시사한다.

3.3 ONNX Simplifier

3.3.1 최적화 효과 분석

ONNX 최적화의 효과를 검증하기 위해 FedAvg(LR: 0.001)과 동일한 실험 조건에서 순수 PyTorch 추론과 ONNX Runtime 추론의 성능을 비교하였다. 모든 측정은 동일한 검증 데이터셋을 사용하였으며, 정확도와 추론 시간을 기록하였다.

3.3.2 모델 크기 비교

- PyTorch 모델(.pt): 389.30 KB
- ONNX 원본(.onnx): 387.54 KB
- ONNX 최적화(.onnx simplified): 388.18 KB
- PyTorch 대비 ONNX 크기 감소율: 0.3%

모델 자체가 연산이 많지 않아 크기 감소는 거의 없는 것을 볼 수 있다.

3.3.3 정확도 비교 (최종 20 라운드 기준)

각 시스템 별 검증 정확도는 다음과 같다:

표 6: 시스템 별 내부 검증 정확도

구분	PyTorch 정확도	ONNX 정확도
Client 0	85.47%	85.47%
Client 1	91.81%	91.81%
Client 2	81.29%	81.29%
Server	80.43%	80.43%

ONNX 변환 후에도 정확도 손실이 없으며 모델의 수학적 연산이 동일하게 유지됨을 확인하였다.

3.3.4 추론 속도 비교

각 Raspberry Pi 모델과 서버의 추론 시간(검증 데이터셋 전체)의 20 라운드 평균은 다음과 같다:

표 7: 시스템 별 추론 시간

구분	PyTorch 평균 시간	ONNX 평균 시간	속도 향상	시간 단축률
Client 0	2.362s	1.637s	1.44 배	30.7%
Client 1	5.035s	3.700s	1.36 배	26.5%
Client 2	9.231s	7.159s	1.29 배	22.4%
Server	2.256s	1.345s	1.68 배	40.4%

ONNX Runtime 을 사용한 추론은 순수 PyTorch 대비 평균 1.36 배의 속도 향상을 보였다. 특히 성능이 낮은 Raspberry Pi 4B 에서도 1.29 배의 개선이 확인되어, 자원이 제한된 엣지 디바이스에서 ONNX 최적화의 효용성이 입증되었다. 또한 연산 성능이 높은 기기일수록 ONNX 최적화로 인한 시간 단축률이 더 크게 나타났으며, 이는 ONNX Runtime 이 고성능 하드웨어에서 더 효율적으로 동작함을 시사한다.

4. 결론 및 토론

4.1 결론

본 연구는 이기종 Raspberry Pi 환경에서 연합학습을 통해 의료 영상 분류 모델을 구현하고 평가하였다. 서로 다른 사양의 Raspberry Pi 3 종(Pi 5 8GB, Pi 5 4GB, Pi 4B 2GB)을 클라이언트로 사용하여 실제 의료 기관의 이기종 하드웨어 환경을 모사하였으며, PathMNIST 대장 조직 병리 이미지 데이터셋을 Dirichlet 분포로 Non-IID 분할하여 현실적인 데이터 이질성을 구현하였다. 본 연구의 주요 결과는 표 8 과 같다.

표 8. 연구 결과 요약

항목	결과
최고 정확도(FedAvg)	82.44%
F1 Score(Macro)	0.7749
중앙집중형 대비 연합학습 성능 비율	96.7%(85.29% 기준)
ONNX 추론 속도 향상	1.29x ~1.44x
ONNX 사용시 정확도 손실	0%

FedAvg 알고리즘은 동일한 모델 구조의 중앙집중형 학습(85.29%) 대비 96.7%의 성능을 달성하여, 데이터 프라이버시를 보장하면서도 실용적 수준의 정확도를 확보할 수 있음을 입증하였다. 약 100K 파라미터의 경량 SimpleCNN 모델을 사용하여 메모리가 2GB 에 불과한 Raspberry Pi 4B 에서도 안정적으로 학습이 가능함을 확인하였다.

특히, 본 연구에서는 실제 배포 환경에서의 효율성을 극대화하기 위해 PyTorch 모델을 ONNX(Open Neural Network Exchange) 포맷으로 변환하고 최적화하였다. 그 결과, 모델의 정확도 손실 없이 서버에서는 1.68 배, 이기종 클라이언트 환경에서는 1.29 배에서 최대 1.44 배의 추론 속도 향상을 달성하였다. 이는 고성능 GPU 서버 없이도 엣지 디바이스 기반의 연합학습 시스템이 의료 현장에서 실시간성을 확보하며 실용적으로 적용될 수 있음을 시사한다.

FedAvg 와 FedProx 의 비교 실험에서는 FedAvg 가 전반적으로 우수한 성능을 보였다. FedProx 는 μ 값이 증가할수록 성능이 저하되었으며(μ=0.01: 81.53%, μ=0.1: 78.57%, μ=0.3: 69.97%), 이는 본 실험의 α=0.5 수준의 Non-IID 환경에서 proximal term 이 로컬 최적화를 과도하게 제한하여 오히려 역효과를 낸 것으로 분석된다.

ONNX 최적화는 실용적 측면에서 여러 이점을 제공한다. 플랫폼 독립적인 ONNX 포맷은 다양한 하드웨어 환경에서의 배포를 간소화하며, 클라이언트에서 평균 1.36 배의 추론 속도 향상은 실시간 의료 영상 분석 시스템 구축에 있어 중요한 개선이다. 또한 최적화 과정에서 정확도 손실 없이 모델 성능을 유지하면서 효율성을 개선할 수 있었다. 본 연구는 연합학습으로 학습된 모델을 실제 엣지 디바이스 환경에 배포할 때, ONNX 기반 최적화가 효과적인 솔루션임을 실증적으로 보여준다.

본 연구에 사용된 전체 소스 코드와 실험 데이터는 아래의 GitHub 리포지토리에서 확인할 수 있다.[16]

4.2 토론

본 연구를 통해 이기종 엣지 디바이스 환경에서의 연합학습 가능성을 확인하였으나, 몇 가지 한계점과 향후 연구 방향을 제시한다.

클래스 불균형 및 통계적 이질성의 한계

본 실험에서는 FedProx 알고리즘과 가중 손실 함수를 도입하여 Non-IID 환경에서의 성능 저하를 완화하고자 하였다. 그러나 단순히 알고리즘을 교체하는 것만으로는 극단적인 데이터 편향 문제를 근본적으로 해결하기 어려웠다. 향후 연구에서는 Data Augmentation, Knowledge Distillation, 또는 클래스별 가중치 재조정과 같은 고도화된 접근 방식을 적용하여 Non-IID 환경에서도 강건한(Robust) 모델을 구축하는 연구가 필요하다.

Train-Test 분포 불일치로 인한 구조적 한계

본 연구에서 발견된 중요한 현상은 특정 클래스(Class 7, STR)의 성능이 라운드 중반에 정점을 찍은 후 하락하는 패턴이다. 이는 PathMNIST 데이터셋의 Train set 과 Test set 간 클래스 분포 차이(STR: Train 10.4% vs Test 5.9%)에서 기인한다. 연합학습에서 글로벌 모델은 서버의 테스트셋 분포를 기준으로 평가되므로, 특정 클라이언트가 소수 클래스 데이터를 다수 보유하더라도 해당 학습 성과가 집계 과정에서 희석되는 Local-Global Distribution Mismatch 문제가 발생한다. 이러한 구조적 한계를 극복하기 위해서는 클라이언트별 클래스 분포를 고려한 동적 가중치 조정 전략이나, 테스트셋 분포 편향을 보정하는 집계 알고리즘에 대한 후속 연구가 필요하다.

데이터셋의 한계

실험에 사용된 PathMNIST 데이터셋은 28×28 픽셀의 저해상도 이미지로, 실제 의료 현장에서 사용되는 고해상도 병리 이미지(Whole Slide Image, WSI)나 3D CT/MRI 영상의 복잡성을 완벽히 대변하기에는 한계가 있다. 향후 연구에서는 더 크고 복잡한 고해상도 의료 데이터셋을 활용하여, 모델의 복잡도를 높이면서도 엣지 디바이스의 경량화 및 가속화를 유지할 수 있는 방안을 모색하고자 한다.

Acknowledgments: 본 연구는 학부 과정에서 습득한 지식을 바탕으로 캡스톤 프로젝트와 별개로 수행된 개별 연구로서, 짧은 기간 동안 독립적인 연구 수행 능력을 기르는 귀중한 기회였습니다. 연구의 방향을 지도해 주시고 아낌없는 조언을 주신 지수연 교수님과, 컴퓨터공학부의 모든 교수님들께 깊은 감사를 드립니다.

참고문헌

[1] U.S. Department of Health & Human Services, "HIPAA Privacy Rule," 2003.

[2] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in Proc. AISTATS, 2017, pp. 1273-1282.

[3] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," Proc. MLSys, vol. 2, pp. 429-450, 2020.

[4] N. Rieke et al., "The future of digital health with federated learning," NPJ Digital Medicine, vol. 3, no. 1, pp. 1-7, 2020.

[5] J. Yang et al., "MedMNIST v2: A large-scale lightweight benchmark for 2D and 3D biomedical image classification," Scientific Data, vol. 10, no. 1, p. 41, 2023.

[6] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," ACM TIST, vol. 10, no. 2, pp. 1-19, 2019.

[7] P. Kairouz et al., "Advances and open problems in federated learning," Foundations and Trends in ML, vol. 14, no. 1-2, pp. 1-210, 2021.

[8] Q. Li, Y. Diao, Q. Chen, and B. He, "Federated learning on non-IID data silos: An experimental study," in Proc. IEEE ICDE, 2022, pp. 965-978.

[9] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-IID data," in Proc. ICLR, 2020.

[10] Z. Lian et al., "DEEP-FEL: Decentralized, efficient and privacy-enhanced federated edge learning for healthcare cyber physical systems," IEEE Trans. Netw. Sci. Eng., vol. 9, no. 5, pp. 3558-3569, 2022.

[11] C. Feng, N. Huber, A. Huertas Celdrán, G. Bovet, and B. Stiller, "Demo: A practical testbed for decentralized federated learning on physical edge devices," arXiv:2505.08033, 2025.

[12] G. Kaissis, M. Makowski, D. Rückert, and R. Braren, "Secure, privacy-preserving and federated machine learning in medical imaging," Nature Machine Intelligence, vol. 2, no. 6, pp. 305-311, 2020.

[13] ONNX Runtime developers, "ONNX Runtime," 2021. [Online]. Available: https://onnxruntime.ai/

[14] ONNX contributors, "Open Neural Network Exchange (ONNX)," 2019. [Online]. Available: https://onnx.ai/

[15] G. A. Baumgart, D. R. Bertolini, and M. Guzzo, "Not all federated learning algorithms are created equal: A performance evaluation study," arXiv:2403.xxxxx, 2024.

[16] K. Lee, "Federated Learning-based Medical Image Classification in Raspberry Pi," GitHub repository, 2025. [Online]. Available: https://github.com/29-min/Federated-Learning-based-Medical-Image-Classification-in-raspberry-pi