Journal of KIIT. Vol. 23, No. 2, pp. 221-232, Feb. 28, 2025. pISSN 1598-8619, eISSN 2093-7571
http://dx.doi.org/10.14801/jkiit.2025.23.2.221

모델 경량화를 통한 RESTful API 기반 동적 연합학습 플랫폼 개발

김동현*, 이석훈**

A Development of Dynamic Federated Learning Platform based on RESTful API through Model Lightweighting

Donghyun Kim*, Sukhoon Lee**

요 약

최근 개인정보 보호에 대한 요구가 커지면서, 데이터 공유와 활용이 제한되고 인공지능 학습 등에 문제가 발생하고 있으며, 이를 해결하기 위해 연합학습(Federated learning) 기법과 그 프레임워크 등이 개발되었다. 하지만 대부분의 연합학습 프레임워크들은 gRPC에 기반하여 초기 구성 변경과 학습 중 클라이언트 환경 변화에 즉각 대응할 수 없는 문제를 보인다. 따라서, 본 논문에서는 모델 경량화를 통한 RESTful API 기반 동적 연합학습 플랫폼을 개발한다. 제안 플랫폼은 gRPC가 지니는 문제를 해결하기 위하여 RESTful API를 활용하여 동적 연합학습을 구현하며, 모델 경량화 기법을 통하여 데이터 전송 효율을 높인다. 실험 및 평가 결과 제안 플랫폼은 연합학습을 잘 수행함을 확인하였으며, 모델 경량화를 통해 경량화 이전 대비 약 93%, 유사 플랫폼들이 채택한 전송방식과 비교하여 약 60%의 데이터 감소 효과가 보임을 확인하였다.

Abstract

With growing demands for privacy protection, data sharing and utilization are becoming restricted, posing challenges in AI training. Federated Learning(FL) and its frameworks have been developed to address this issue. However, most gRPC-based FL frameworks struggle to adapt to configuration changes and client environment shifts during training. This paper presents a RESTful API-based dynamic FL platform with model lightweighting. The platform overcomes gRPC limitations by using RESTful APIs for dynamic FL and improves data transmission efficiency with model lightweighting. Experiments show that the platform supports FL effectively, achieving around 93% data reduction compared to the original and about 60% compared to similar platforms.

Keywords
federated learning, RESTful API, model lightweighting, distributed systems, deep learning

* 국립군산대학교 소프트웨어학과 학부과정
- ORCID: https://orcid.org/0009-0006-7695-7804
** 국립군산대학교 소프트웨어학과 교수(교신저자)
- ORCID: https://orcid.org/0000-0002-3390-5602

· Received: Nov. 14, 2024, Revised: Dec. 18, 2024, Accepted: Dec. 21, 2024
· Corresponding author: Sukhoon Lee
Dept. of Software Science & Engineering, Kunsan National University, Korea
Tel.: +82-63-469-8914, Email: leha82@kunsan.ac.kr

Ⅰ. 서 론

최근 인공지능과 빅데이터 기술의 발전으로 다양한 산업 분야에서 기계학습 및 딥러닝 모델이 활발히 활용되고 있다[1]. 특히 의료, 금융, 교통 등의 분야에서 기계학습과 딥러닝 모델은 데이터 기반의 의사결정과 예측 분석에 중요한 역할을 하고 있다.

그러나 이러한 기술의 발전과 함께 개인정보 보호에 대한 사회적 관심도 높아지고 있으며 데이터 공유와 활용에 대한 법적, 윤리적 제약이 강화되고 있어 의료, 금융 분야와 같이 개인정보 보호에 민감한 분야에서는 대규모 데이터 확보 및 활용이 매우 어렵다[2].

이러한 문제를 해결하기 위해 연합학습(FL, Federated Learning)이 제안되었다[3]. 연합학습은 중앙 서버에 데이터를 집중시키지 않고 분산된 데이터를 활용하여 모델을 학습시키는 방식으로 진행된다.

한편, 연합학습 환경을 쉽게 구성하고 사용할 수 있게 해주는 다양한 프레임워크가 등장했다. 표 1은 연합학습 프레임워크인 Flower[4], PySyft[5], TFF (TensorFlow Federated)[6], FATE[7]의 특징을 보인다. 특히, 통신 방식(Communication method)에서 Flower를 비롯한 프레임워크들은 서버-클라이언트 구조의 데이터 전송방식으로 gRPC(google Remote Procedure Call)[8]을 사용함으로써 효율적이고 빠른 데이터 통신이 가능하다.

하지만, gPRC에서 서버와 클라이언트는 프로토콜 버퍼(Protocol buffer)라는 함수 및 데이터 포맷을 사용하여 서로 간 통신을 수행한다. 이때, 프로토콜 버퍼의 구성에 변화가 생기면 연합학습에 참여하고 있는 서버와 클라이언트들 모두 동일하게 업데이트 되어야 하므로 동적인 환경 변화에 유연하게 대응하기 어렵다는 단점이 있다[9].

한편, RESTful API 통신방식은 HTTP URI를 설계하여 필요한 리소스를 명시하고 HTTP Method를 통해 각 클라이언트에서 서버에 요청하여 원하는 동작을 수행하고 그에 대한 서버의 응답을 받는다[10]. 각 클라이언트의 요청은 서버에서 독립적으로 처리되기 때문에 서버와 클라이언트 간 느슨한 결합(Loosely coupled)을 구성할 수 있다. 이는 연합학습 라운드가 진행될 때 클라이언트의 환경 변화가 일어나거나 참여 상태가 달라져도 문제없이 라운드를 진행할 수 있음을 의미한다.

하지만 RESTful API 방식은 기본적으로 데이터 전송을 위해 문자열 기반의 JSON 형식을 사용하므로 실수 행렬 형태의 인공지능 모델 파라미터와 같은 데이터 구조를 전송하기에는 적절하지 않다. 따라서 인공지능 모델 파라미터의 효율적인 전송을 위해 추가적인 처리 과정이 필요하다.

본 논문에서는 모델 경량화를 통한 RESTful API 기반의 동적 연합학습 플랫폼을 개발한다. 본 논문은 제안 플랫폼의 시스템 구조를 기술하고 RESTful API를 위한 URI를 설계한다. 또한, 글로벌 모델과 로컬 모델의 효율적인 전송을 위하여 각 모델들의 파라미터를 추출하고 직렬화 및 압축을 통해 경량화 작업을 수행한다. 추가적으로, 본 논문은 사용자의 실험 편의성을 위해 연합학습의 현황을 모니터링하고 자동으로 연합학습 라운드를 수행할 수 있는 시스템인 대시보드를 구현한다.

표 1. 연합학습 프레임워크들의 특징
Table 1. Features of federated learning frameworks

Feature	Flower[4]	PySyft[5]	TFF[6]	FATE[7]
Main objectives	highly scalable support	for data science	large-scale federated learning	industrial system focus
Supported framework	Pytorch, TensorFlow	Pytorch	TensorFlow	Pytorch, TensorFlow
Communication method	gRPC	WebSocket	gRPC	gRPC, HTTP
Parameter transmission method	numpy serialization	pickle serialization	tensorflow serialization	numpy serialization

본 논문의 구성은 다음과 같다. 2장은 관련 연구를 소개하고 3장에서는 제안 플랫폼의 구조 및 RESTful API 기반의 연합학습에 관해 기술한다. 4장에서는 실험 및 평가를 진행한다. 마지막으로 5장에서는 결론 및 향후 연구에 대하여 기술한다.

II. 관련 연구

2.1 연합학습 플랫폼

BFLSID (Blockchain-based Federated Learning System for Infrutsion Detection)는 IoT 네트워크에서 침입탐지를 개선하기 위해 설계된 기계학습을 사용한 블록체인 기반의 연합학습 플랫폼이다[11]. IPFS (InterPlanetary File System)와 Hyperledger Fabric을 통해 글로벌 모델의 매개변수와 탐지 결과의 무결성과 보안을 강화한다. 또한 소수 클래스를 과도 샘플링하여 데이터 불균형 문제를 해결하였다. CNN (Convolutional Neural Network), LSTM (Long Short-Term Memory) 모델을 통해 평가하였고 중앙 집중식 모델과 유사한 성능을 보장한다.

Lee et al.은 연합학습에서 네트워크 및 통신 상황을 기반으로 학습에 참여할 기기를 선택하는 기법을 제안하였다[12]. 본 연구는 네트워크 상황이 다른 기기들 간의 성능 차이로 인해 연합학습이 지연되는 문제를 해결하기 위해 각 기기의 네트워크 및 통신 지표를 측정하고 이를 바탕으로 적합한 학습 참여 기기를 선정하는 방법을 고안하였다.

Shin et al.은 하이브리드 클라우드 환경에서의 연합학습 시스템 아키텍처를 제안하였다[13]. 제안된 아키텍처는 로컬 디바이스의 제한된 리소스에서도 안정적인 성능의 모델을 생성할 수 있도록 AWS (Amazon Web Services) 기반의 클라우드 환경에서 연합학습을 구현하였다. TCP/IP 통신을 적용하여 중앙 서버와 다수의 클라이언트 간의 통신을 통해 글로벌 및 로컬 모델의 성능을 평가하였다.

FedMon은 연합학습 시스템에서 모니터링을 효율적으로 수행하기 위해 제안된 도구이다[14]. FedMon은 연합학습 과정에서 발생하는 다양한 시스템 및 모델 지표를 자동으로 추출하고 분석하며, 학습 성능의 저하를 감지하고 이전 학습과 비교할 수 있는 기능을 제공한다.

2.2 연합학습 기법

FedProx는 시스템 및 통계적 이질성을 해결하기 위해 제안된 연합학습 최적화 프레임워크이다[15]. 이 프레임워크는 기기별로 수행할 수 있는 작업량을 조절할 수 있도록 하여 연합학습 내에서 더 안정적이고 효율적인 수렴 성능을 제공한다. FedProx는 기존 FedAvg 알고리즘을 일반화하고 재구성한 형태로, 시스템 제약이 있는 환경에서도 비 동일하게 분포된 데이터에 대해 안정적인 성능을 보장할 수 있도록 돕는다.

FedPer는 연합학습에서의 통계적 이질성 문제를 해결하기 위해 제안된 방법이다[16]. FedPer는 모델을 기본 계층과 개인화 계층으로 나누어 학습하는 접근법이다. 기본 계층은 모든 클라이언트가 공유하며 연합학습을 통해 학습되지만, 개인화 계층은 각 클라이언트에서 개별적으로 학습된다. 이를 통해 각 클라이언트의 고유한 데이터 특성을 반영하면서도 중앙 서버에서 전송되는 기본 계층을 활용하여 협력 학습이 가능하다.

Kim et al.은 연합학습에서 성능 저하 문제를 해결하기 위해 정확도 기반 참여 제한 연합학습을 제안하였다[17]. 이 방법은 로컬 모델의 평가 정확도를 기반으로 성능이 낮은 클라이언트의 파라미터를 집계에서 제외하여 글로벌 모델의 품질을 향상시킨다. 이 연구는 MNIST 데이터셋을 사용하여 IID (Independent and Identically Distributed)와 Non-IID (Non-Independent and Identically Distributed) 환경에서 실험을 수행하였으며, 결과적으로 기존 연합학습 방식에 비해 IID 환경에서는 평균 2 라운드, Non-IID 환경에서는 평균 7.77 라운드 더 빠르게 목표 성능에 도달함을 확인하였다.

Kim et al.은 연합학습의 성능을 개선하기 위해 Learning Agent를 활용하고 다중 경로 기반의 모델 분할 전송 기법을 제안하였다[18]. 이 연구는 기존의 연합학습이 개별 클라이언트의 성능과 통신 상황에 의해 학습 성능이 제한되는 문제를 해결하고

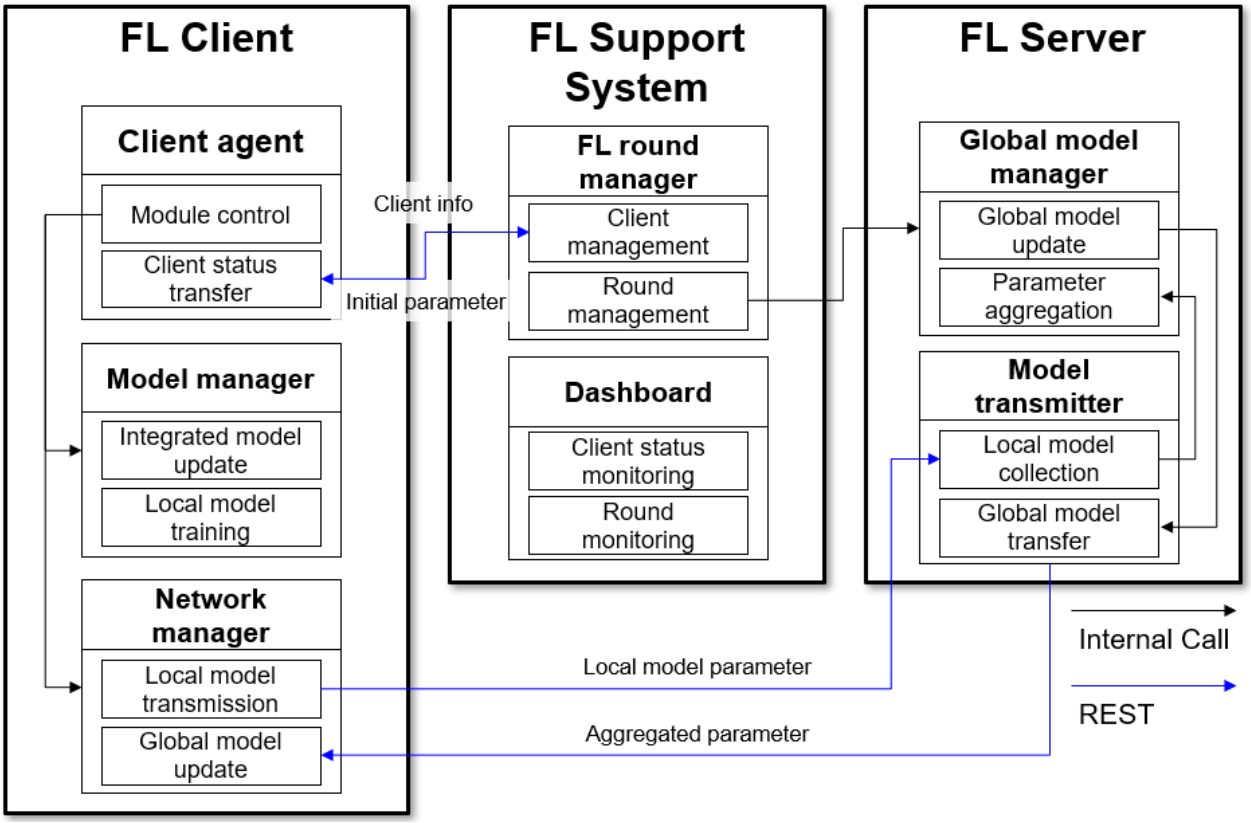

그림 1. 제안 플랫폼 구조
Fig. 1. Proposed platform architecture

자, 클라이언트 주변의 신뢰할 수 있는 유휴 장치를 통해 성능을 높이고, 일반적인 TCP 기반의 다중 경로 전송을 통해 통신 지연을 줄이는 방법을 제시하였다.

Lee et al.는 Non-IID 데이터 문제를 해결하기 위해 다중 서버 환경에서 계층적 클러스터링 기반 연합학습을 제안하였다[19]. 이 기법은 중앙 서버에 과도하게 의존하지 않도록 프록시 서버를 도입하여 중앙 서버의 부하를 줄이고, 클라이언트의 가중치를 클러스터링하여 병렬적으로 학습을 수행하는 방식을 사용한다.

III. RESTful API를 활용한 연합학습 플랫폼

이 장은 제안 플랫폼의 구조를 기술하고 제안 플랫폼을 위해 설계한 HTTP URI에 대해 기술한다. 이후 연합학습 프로세스에서 개선한 파라미터 전송 방식과 제안 플랫폼에서 제공하는 대시보드에 관해 기술한다.

3.1 제안 플랫폼 구조

그림 1은 제안 플랫폼 구조를 보인다. 크게 FL Client, FL Support System, FL Server로 나누어진다.

FL Client는 연합학습에 참여하는 클라이언트들의 모듈 구성이다. FL Client는 Client agent, Model manager, Network manager로 구성되어 있다.

Client agent는 클라이언트의 모듈들을 제어하고 클라이언트의 상태를 서버로 전송한다. 또한 클라이언트 등록을 요청하고 초기 파라미터를 수신한다. Model manager는 글로벌 모델 파라미터를 로컬 모델과 통합하고 로컬 모델을 학습시키는 역할을 수행한다. Network manager는 학습된 로컬 파라미터를 서버로 송신하고 업데이트된 글로벌 파라미터를 서버로부터 수신한다.

FL Support System은 연합학습 시스템의 라운드 진행 관리를 담당하는 연합학습 지원 시스템이다. FL Support System은 FL Round manager와 Dashboard로 구성되어 있다.

FL Round manager에서는 클라이언트 관리와 연합학습 라운드 관리를 수행한다. 클라이언트가 등록을 요청하면 클라이언트 등록과 동시에 초기 파라미터를 전송해준다. 또한 연합학습 진행 과정에서 클라이언트의 실시간 상태, 라운드 진행 여부, 자동학습의 여부 등의 데이터를 저장하여 Dashboard에 제공한다. Dashboard는 클라이언트의 상태를 실시간으로 모니터링하고 라운드별 학습 정확도와 진행 상황을 시각적으로 확인할 수 있게 한다. 또한 사용자가 Dashboard를 통해 임의로 연합학습의 진행과 자동학습을 제어할 수 있다.

FL Server는 글로벌 모델의 업데이트와 각 클라이언트로부터 수신한 로컬 파라미터들을 통합하는 역할을 수행한다. FL Server는 Global model manager와 Model transmitter로 구성되어 있다.

Global model manager는 글로벌 모델을 업데이트하고 클라이언트에서 수집된 로컬 모델 파라미터를 통합한다. 본 논문에서는 FedAvg[3]를 사용하여 클라이언트들의 파라미터를 통합한다. FedAvg는 각 클라이언트의 파라미터를 평균 내어 글로벌 파라미터를 생성하는 방식이다.

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^{(k)} \tag{1}$$

식 (1)은 FedAvg를 표현한 식이다. w_{t+1}은 글로벌 모델의 업데이트된 파라미터를 의미한다. 즉 FedAvg의 결과다. K는 참여 클라이언트의 수를 의미한다. n_k는 클라이언트 k의 데이터 샘플 수를 나타내고 n은 전체 클라이언트의 데이터 샘플 총합이다. $w_{t+1}^{(k)}$는 클라이언트 k에서 업데이트된 로컬 모델 파라미터를 의미한다. 파라미터 통합이 완료되면 글로벌 모델을 업데이트하고 해당 라운드의 정확도를 저장한다. 그 다음 글로벌 모델 파라미터를 추출하여 직렬화하고 압축하여 Model transmitter에게 전달한다.

표 2. RESTful API구성을 위한 URI 설계
Table 2. Designing URI structure for RESTful API configuration

Classification	URI	HTTP method	Description	Request data	Response data
Client	/client	POST	client registration	client id	de-identified client id
		GET	client initial parameter request	none	compressed initial parameters
	/client/{client_id}	DELETE	disconnect a specific client	client id	client disconnected confirmation
	/client/autorun	DELETE	stop automatic progression	none	autorun stopped confirmation
	/client/autorun/{rounds}	POST	automatically proceed for a specified number of rounds	number of rounds	autorun started confirmation
	/client/signal	POST	transmission of the current client status	client name, client signal	client status reception confirmation
Parameter	/transmitter	GET	transmission of updated global model parameters	none	compressed aggregated parameters
	/transmitter/{name}	POST	client local parameter transmission	client ID, client parameters	parameter transmission confirmation

Model transmitter는 클라이언트로부터 로컬 모델 파라미터를 수집하고 업데이트된 글로벌 모델 파라미터를 다시 클라이언트로 전송한다. 그 후 로컬 모델 파라미터를 클라이언트 파라미터 리스트에 저장한다.

3.2 RESTful API의 URI 설계

RESTful API를 활용한 연합학습 시스템을 구성하기 위해서는 연합학습을 진행할 때 필요한 리소스에 대해 URI를 통해 정의해야 한다. 표 2는 연합학습 시스템 구축을 위한 URI 설계를 보인다. 각 URI는 클라이언트와 서버 간의 데이터를 주고받기 위해 필요한 리소스를 정의하며, HTTP 메소드를 통해 특정 동작을 수행한다.

/client는 연합학습 시스템에 클라이언트를 등록하는 역할을 한다. 또한 클라이언트는 GET 메소드를 통해 서버로부터 초기 파라미터를 요청하며 서버는 압축된 초기 파라미터를 응답으로 제공한다. 또한 DELETE 메소드를 통해 연결 해제를 원하는 클라이언트 ID를 서버로 전송하고 서버는 해당 클라이언트의 연결을 해제한다.

/client/autorun는 클라이언트가 특정 라운드 수만큼 자동으로 학습을 진행할 수 있도록 설정하거나 학습 도중 자동 진행을 중단할 때 사용한다. POST 메소드를 통해 특정 라운드 수를 지정하여 자동 학습을 시작할 수 있으며 DELETE 메소드를 통해 자동 학습을 중단할 수 있다.

/client/signal는 클라이언트의 학습 상태나 연결 상태에 대해 서버에 보고하며 이를 통해 연합학습에 참여 중인 클라이언트들의 상태를 실시간으로 확인하고 필요시 사용자가 적절한 조치를 취할 수 있다.

/transmitter는 클라이언트로부터 로컬 파라미터를 전송받거나 업데이트된 글로벌 모델을 전송하는 연합학습의 핵심 기능을 담당한다. 클라이언트는 POST 메소드를 통해 클라이언트 ID와 학습된 로컬 파라미터를 서버로 전송하며 GET 메소드를 통해 업데이트된 글로벌 모델의 파라미터를 전송받아 로컬 모델에 적용시킨다.

3.3 연합학습 프로세스

그림 2는 제안 플랫폼을 통한 연합학습 프로세스를 보인다. 클라이언트는 처음 연합학습에 참여하면 클라이언트 정보를 등록하고 초기 파라미터를 학습한다. 그 후 학습한 로컬 모델 파라미터를 서버로 전송하고 서버에서는 각 클라이언트들이 보낸 파라미터들을 통합하여 업데이트한다.

그 후 현재 라운드가 마지막 라운드인지 확인한 후 마지막 라운드일 경우 연합학습을 종료하고 현재 라운드가 사용자가 설정한 마지막 라운드가 아니라면 각 클라이언트에게 업데이트된 글로벌 모델 파라미터를 전송한다.

한편, REST 기반의 데이터 통신에서는 일반적으로 JSON 형식으로 데이터를 전송한다. JSON 데이터 형식은 {키:값} 쌍으로 이루어져 모델 파라미터를 변환하기에 데이터 형식상의 문제는 없지만 모델 파라미터와 같은 Python 객체는 바로 JSON 형태로 바로 변환할 수 없고 Numpy 형태로 변환한 다음 JSON으로 변환할 수 있다.

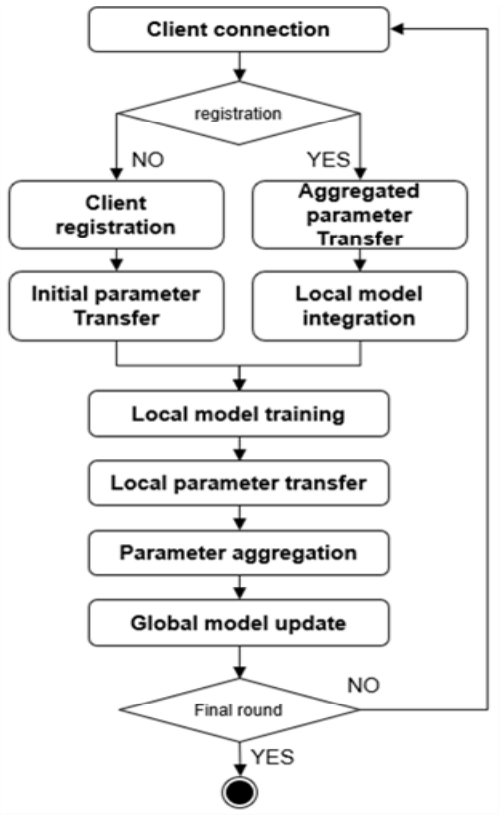

그림 2. 제안 플랫폼의 연합학습 프로세스
Fig. 2. Federated learning process of the proposed platform

이 과정에서 데이터 용량이 크게 증가해 전송 효율이 악화된다. 따라서 본 논문에서는 파라미터의 송수신 과정에서 추가적인 처리를 통해 데이터 용량을 감소시켜 전송하는 방식을 제안한다.

그림 3은 파라미터 용량 감소를 위한 추가적인 처리 프로세스를 보인다. 파라미터 처리는 파라미터 데이터를 추출한 후 송신하기 전과 수신한 후 모델에 적용하기 전에 진행한다. 이 과정은 서버와 클라이언트 각각 모두 해당된다. 파라미터를 송신하기 전 float32 형태의 모델 파라미터 자료형을 bfloat16[20] 형태로 변환한다.

bfloat16은 float32와 동일한 표현 범위를 가지지만 정밀도는 다르다. 따라서 bfloat16을 사용하면 메모리 사용량을 크게 절감할 수 있지만 정밀도의 차이로 인해 모델의 정확도가 감소할 수 있다. 연합학습 환경에서는 여러 클라이언트가 각자의 로컬 데이터로 모델을 학습하고 글로벌 서버에 모델 파라미터를 전송한다. 클라이언트가 다수인 연합학습 환경에서는 데이터 전송 비용이 학습 속도와 효율성에 큰 영향을 미치기 때문에 float32 대신 bfloat16을 사용하면 파라미터 통신에 필요한 데이터 전송 비용을 줄이는 데 매우 효과적이다.

그 다음 변환한 데이터를 직렬화하고 압축하여 송신한다. 압축된 데이터를 수신한 후 모델에 적용시키기 위해 데이터를 압축해제하고 역 직렬화한 다음 다시 float32 데이터로 변환해 원본 데이터를 모델에 로드한다.

3.4 연합학습 지원 시스템

제안 플랫폼을 활용한 연합학습 실험의 편의를 위해 제작한 대시보드를 통해 사용자는 클라이언트의 상태를 실시간으로 확인하고 연합학습의 진행을 제어할 수 있다. 그림 4는 제안 플랫폼을 통한 연합학습 진행 과정에서의 대시보드를 보인다. FL Support System에서 수집한 클라이언트 상태정보와 글로벌 모델의 메타데이터, 라운드 진행 상황에 대한 정보를 웹 시각화를 통해 사용자에게 제공한다. 클라이언트의 상태는 join, ready, training, update, finish로 구성된다.

또한 Global Model Status를 통해 글로벌 모델의 메타데이터 및 라운드 진행 상황에 대한 정보를 제공한다. Global Model Status는 round autorun, current round, number of clients, global model accuracy, last updated로 구성된다.

그림 4. 제안 플랫폼 대시보드
Fig. 4. Proposed platform dashboard

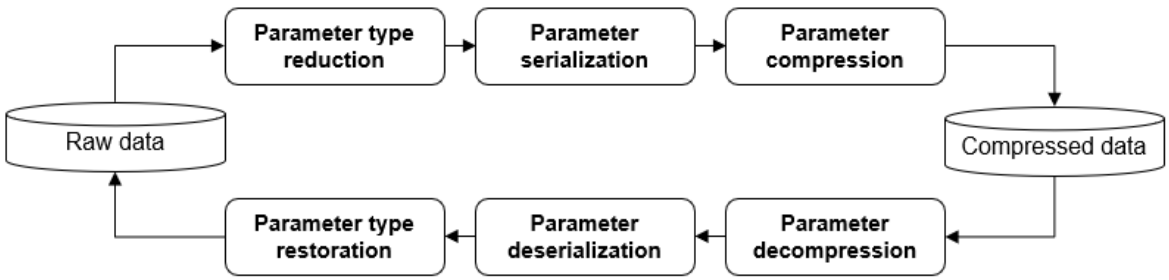

그림 3. 파라미터 데이터 처리 프로세스
Fig. 3. Parameter data processing process

또한 Round Autorun, Manual Control을 통해 사용자가 직접 연합학습 라운드를 자동/수동 진행할 수 있도록 제어하거나 클라이언트의 라운드 진행 도중 자유로운 이탈과 재참여를 위한 인터페이스를 제공한다. 사용자는 목표 라운드를 입력하여 자동학습을 진행시키거나 종료시킬 수 있다. 또한 로컬 모델 학습에 대한 명령과 글로벌 모델 업데이트에 대한 명령을 하여 수동으로 연합학습을 진행 시킬 수 있다.

그림 5는 연합학습 진행 중 클라이언트의 이탈 상황을 보인다. client_3이 라운드 중간에 이탈하게 되면서 나머지 클라이언트들만으로 연합학습이 수행된다. Global Model Status에서도 참여 중인 클라이언트 수가 실시간으로 업데이트되어 5에서 4로 변경된다.

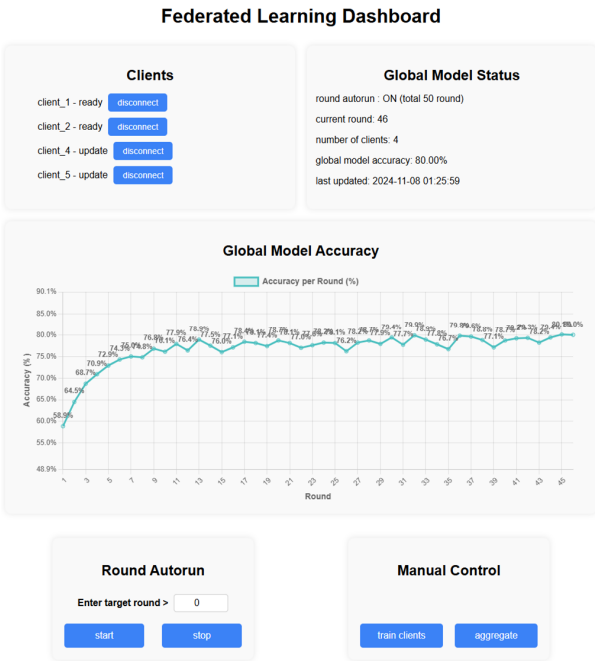

그림 5. 연합학습 중 클라이언트의 이탈
Fig. 5. Client dropout during federated learning

그림 6은 연합학습 중 클라이언트가 중도 참여한 상황을 보인다. 중도 참여한 클라이언트는 현재 라운드가 진행 중인 상황에서는 대기하다가 다음 라운드가 진행될 때 기존의 클라이언트들과 마찬가지로 서버로부터 업데이트된 글로벌 모델 파라미터를 수신하여 연합학습에 참여하게 된다. Global Model Status에서도 이탈 상황과 마찬가지로 실시간 업데이트되어 4에서 5로 변경된다.

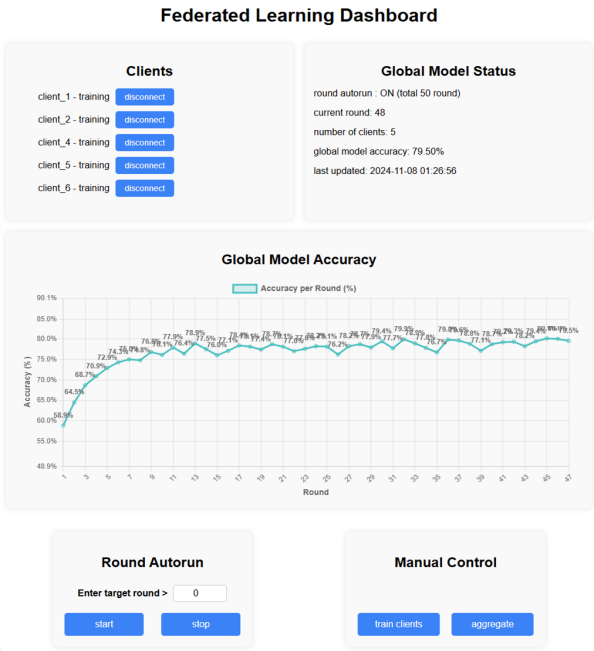

그림 6. 연합학습 중 클라이언트의 중도 참여
Fig. 6. Client late participation during federated learning

IV. 실험 및 결과

4.1 실험 방법

제안 플랫폼으로 연합학습 환경을 구성하여 진행해보고 제안 방안이 실제로 동작함을 보인다. 실험을 위한 서버, 클라이언트 환경은 각각 표 3, 표 4와 같다.

표 3. 연합학습 서버 환경
Table 3. Federated learning server environment

Features	Specification
OS	Windows 11 pro
CPU	Intel(R) Core(TM) i9-13900K 3.00 GHz
GPU	NVIDIA RTX A6000
RAM	32GB
Software	Python, Flask

표 4. 연합학습 클라이언트 환경
Table 4. Federated learning client environment

Features	Specification
OS	Windows 10 pro
CPU	Intel(R) Core(TM) i7-13700 2.10 GHz
GPU	NVIDIA RTX 3080 Ti
RAM	16GB
Software	Python, Flask

CIFAR-10 데이터는 10개의 클래스를 가진 이미지 데이터이며 학습데이터 총 50,000개에 클래스당 5,000개로 구성되어 있고 테스트 데이터는 총 10,000개에 클래스당 1,000개로 구성되어 있다. 연합학습을 위해 해당 데이터셋을 클라이언트의 수에 맞추어 분할하여 저장한 후 실험하였다.

표 5, 6은 각각 IID, Non-IID 환경에서 클라이언트 수 별 데이터 분포를 보인다. 연합학습 참여 클라이언트 수가 각각 2, 3, 5개일 때와 중앙집중형 학습방식을 비교하였고 또한 데이터가 Non-IID 한 상황에서의 연합학습 환경도 구성하여 중앙집중식 학습방식과 비교하였다. Non-IID는 각 클라이언트끼리 클래스별 데이터의 분포나 데이터 수가 균일하지 않은 환경을 의미한다. 모델 학습을 위해 분할한 데이터는 클라이언트 내에서 train : validation : test의 비율을 8:1:1로 구성하여 학습을 진행하였다.

표 5. IID 환경에서의 클라이언트 수 별 데이터 분포
Table 5. Data distribution by number of clients in an IID environment

Number of clients	Client	Train	Val	Test
2	client_1	25,000	2,500	2,500
	client_2	25,000	2,500	2,500
3	client_1	16,666	1,666	1,666
	client_2	16,666	1,666	1,666
	client_3	16,666	1,666	1,666
5	client_1	10,000	1,000	1,000
	client_2	10,000	1,000	1,000
	client_3	10,000	1,000	1,000
	client_4	10,000	1,000	1,000
	client_5	10,000	1,000	1,000

표 6. Non-IID 환경에서의 클라이언트 수 별 데이터 분포
Table 6. Data distribution by number of clients in an Non-IID environment

Number of clients	Client	Train	Val	Test
2	client_1	42,754	4,275	4,275
	client_2	7,245	724	724
3	client_1	19,067	1,906	1,906
	client_2	23,281	2,328	2,328
	client_3	7,651	765	765
5	client_1	9,695	969	969
	client_2	18,545	1,854	1,854
	client_3	3,712	371	371
	client_4	12,339	1,233	1,233
	client_5	5,706	570	570

중앙집중식 학습에서는 전체 데이터셋을 활용하여 학습 데이터와 검증 데이터, 테스트 데이터의 비율 분배만 진행한 후 학습하여 비교하였다. 실험은 중앙집중식 학습방식으로 100번 학습하여 얻은 최고 정확도를 기준으로 각각의 연합학습을 통해 구성한 학습 환경에서 충분히 도달할 수 있는지 확인한다.

4.2 실험 결과

그림 7, 8은 각각 IID 환경과 Non-IID 환경에서의 클라이언트 수에 따른 라운드별 글로벌 모델 정확도를 보인다. 제안 플랫폼을 통해 연합학습을 수행하여 중앙집중식 학습방식에서 100회 학습하여 얻은 최고 정확도에 근접하는 정확도를 얻었다.

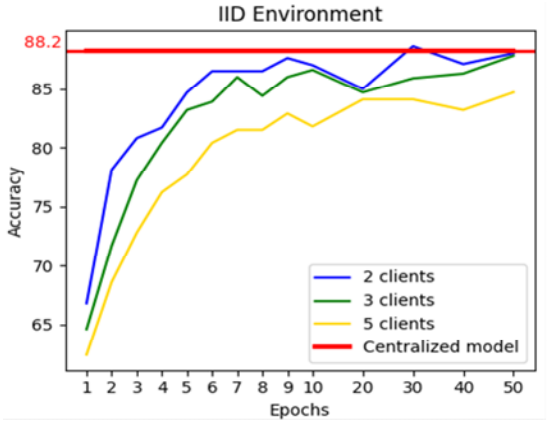

그림 7. IID 환경에서의 클라이언트 수에 따른 라운드별 모델 정확도
Fig. 7. Model accuracy per round according to the number of clients in an IID environment

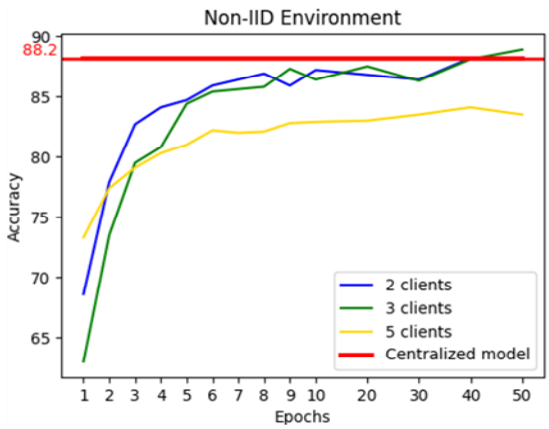

그림 8. Non-IID 환경에서의 클라이언트 수에 따른 라운드별 모델 정확도
Fig. 8. Model accuracy per round according to the number of clients in an Non-IID environment

중앙집중식으로 100회 학습하여 0.882의 최고 정확도를 얻었고 IID 환경에서 각각 2, 3, 5개의 클라이언트가 참여한 연합학습을 수행하여 0.89, 0.882, 0.855의 정확도를 얻었다. Non-IID 환경에서 각각 2, 3, 5개의 클라이언트가 참여한 연합학습을 수행하여 0.889, 0.891, 0.842의 정확도를 얻었다. IID 환경과 Non-IID 환경 모두 10라운드가 진행되기 전에 중앙집중식 학습 결과 정확도에 근접하게 수렴하는 모습을 보인다.

한편, 클라이언트의 수가 5개일 경우 글로벌 모델의 정확도가 비교적 낮다. 데이터의 수가 제한적인 상황에서 클라이언트의 수가 많아질수록 각 클라이언트의 로컬 모델이 로컬 데이터에 과적합되어 글로벌 모델의 정확도가 높지 않은 것으로 보인다.

본 논문에서는 모델 파라미터의 자료형 변환 후 직렬화 및 압축을 통한 파라미터 전송방식을 제안했다. Python 객체인 파라미터 Raw data를 직렬화하기 위해서는 Python 객체의 직렬화를 지원하는 Pickle 모듈을 사용해야 한다. 또한 직렬화된 파라미터를 압축하기 위해 다양한 압축 알고리즘을 적용하여 비교했다.

표 7은 각각 ResNet-18과 ResNet-50 모델파라미터에 압축 알고리즘들을 적용하여 압축 소요시간, 압축률, 압축 결과 데이터 크기를 보여준다.

표 7. 압축 알고리즘 크기 비교
Table 7. Compression algorithm size comparison

Model	Algorithm	Time (sec)	Rate (%)	Size (MB)
ResNet-18	gzip	1.24	7.34	39.60
	zlib	1.23	7.33	39.60
	bz2	2.65	5.46	40.40
	lzma	11.78	8.23	39.22
	lz4	0.04	0.03	42.72
	brotli	60.85	8.67	39.03
	zstd	0.02	7.55	39.50
	snappy	0.02	0.0	42.73
ResNet-50	gzip	2.58	7.29	83.50
	zlib	2.56	7.29	83.50
	bz2	5.76	5.22	85.37
	lzma	25.57	8.09	82.78
	lz4	0.07	0.04	90.03
	brotli	124.77	8.54	82.38
	zstd	0.06	7.45	83.36
	snappy	0.04	0.0	90.06

model은 사용한 모델, algorithm은 사용한 압축 알고리즘, time은 압축 소요 시간, comp rate는 압축률, size는 압축결과 데이터 크기이다.

각각 파라미터 Raw data 42.83MB, 90.46MB를 기준으로 압축한 결과 brotli 알고리즘이 압축률은 가장 높았으나 압축 소요시간이 가장 길다. 압축 소요 시간과 압축률을 고려했을 때 zstd 알고리즘이 가장 효율적인 압축 알고리즘임을 알 수 있다. 따라서 본 논문에서는 zstd 압축 알고리즘을 채택하여 직렬화된 파라미터에 zstd 알고리즘을 이용한 압축을 수행했다.

표 8은 ResNet-18, ResNet-50 모델을 사용하여 제안한 파라미터 전송 효율을 위한 처리 방안과 JSON, Binary, Raw data 형태에서의 파라미터 데이터 크기를 비교한다. JSON 데이터는 Raw data를 추출하여 Numpy 변환 후 JSON으로 변환하였고 Binary 데이터는 추출한 모델 파라미터를 pickle 모듈을 통해 직렬화한 데이터이다.

Raw data는 모델에서 state_dict() 메소드를 통해 추출한 파라미터 데이터를 의미한다. 제안 방안은 일반적인 REST 통신방식의 데이터 전송방식인 JSON 방식보다 약 93%의 데이터 용량이 감소하였고 유사 연합학습 플랫폼에서 일반적으로 채택한 직렬화 방식과 Raw data보다 60%의 데이터 용량이 감소하였다.

표 8. 파라미터 전송 방식 데이터 크기 비교
Table 8. Parameter transmission method data size comparison

Model	Data format	Size (MB)
ResNet-18	json	247.72
	binary	42.73
	raw data	42.83
	proposed	16.72
ResNet-50	json	556.51
	binary	90.46
	raw data	90.46
	proposed	35.29

V. 결 론

본 논문에서는 연합학습 진행 중에도 클라이언트의 환경변화에 동적으로 대응할 수 있고 모델 경량화를 통해 파라미터 데이터 전송 방식을 개선한 연합학습 플랫폼을 구현하였다.

제안한 RESTful API 기반의 연합학습 플랫폼은 기존의 gRPC 기반 시스템이 가진 서버와 클라이언트의 강한 결합의 문제를 해결하는 동시에, 데이터 전송 효율성 측면에서도 개선된 성능을 보여주었다. 실험 결과, 제안된 플랫폼은 다양한 클라이언트의 동적 참여와 탈퇴에도 안정적으로 학습을 진행할 수 있었으며, 데이터 전송량을 줄여 제안 플랫폼을 활용한 연합학습 시스템의 효율성을 향상했다.

향후에는 모바일, 엣지 디바이스 등 다양한 클라이언트 환경에서도 연합학습에 참여할 수 있도록 환경을 구성하고, 다양한 모델 아키텍처와 학습 데이터에 대한 실험을 통해 제안 플랫폼의 범용성을 더욱 확장할 계획이다.

References

[1] Y. J. Kim and K. G. Kim, "Development of an Optimized Deep Learning Model for Medical Imaging", Journal of the Korean Society of Radiology, Vol. 81, No. 6, pp. 1274-1289, Nov. 2020. https://doi.org/10.3348/jksr.2020.0171.

[2] H. S. Choi and Y. H. Cho, "Analysis of Security Problems of Deep Learning Technology", Journal of the Korea Convergence Society, Vol. 10, No. 5, pp. 9-16, May 2019. https://doi.org/10.15207/JKCS.2019.10.5.009.

[3] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-Efficient Learning of Deep Networks from Decentralized Data", International Conference on Artificial Intelligence and Statistics, Cadiz, Spain, Vol. 54, May 2016.

[4] Flower, https://flower.ai/docs/framework/main/ko/index.html [accessed: Oct. 05, 2024]

[5] PySyft, https://openmined.github.io/PySyft/ [accessed: Oct. 08, 2024]

[6] TensorFlow Federated, https://www.tensorflow.org/federated?hl=ko [accessed: Oct. 05, 2024]

[7] FATE, https://fate.fedai.org/fateboard/ [accessed: Oct. 10, 2024]

[8] gRPC, https://grpc.io/ [accessed: Oct. 11, 2024]

[9] H. T. Seo, S. K. Ko, H. Jung, and C. Cho, "Implementation of Deep-learning based Mobile Application using gRPC Protocol", 2018 IEIE Summer Conference, Jeju, Korea, pp. 1069-1072, Jun. 2018.

[10] REST, https://www.ibm.com/kr-ko/topics/rest-apis [accessed: Oct. 11, 2024]

[11] S. J. Lim, E. H. Choi, P. J. Choi, S. H. Lee, and K. R. Kwon, "Blockchain-Based Federated Learning Using Machine Learning Classifier for IoT Intrusion Detection System", Journal of Korea Multimedia Society, Vol. 27, No. 9, pp. 1076-1086, Sep. 2024. https://doi.org/10.9717/kmms.2024.27.9.1076.

[12] M. Lee, M. Sung, and W. Lee, "A Study on Participant Selection Scheme Based on Network Situation Prediction of Server-to-Device Communication Paths", Journal of Next-generation Convergence Technology Association, Vol. 7, No. 3, pp. 340-349, Mar. 2023. https://doi.org/10.33097/JNCTA.2023.07.03.340.

[13] H. Shin, S. Oh, S. Oh, K. Kim, and J. Kim, "A Study on the Federated Learning System Architecture in Hybrid Cloud Environment", Journal of knowledge information technology society, Vol. 18, No. 2, pp. 353-365, Apr. 2023. https://doi.org/10.34163/jkits.2023.18.2.009.

[14] M. Symeonides, D. Trihinas, and F. Nikolaidis, "FedMon: A Federated Learning Monitoring Toolkit", IoT, Vol. 5, No. 2, pp. 227-249, Apr. 2024. https://doi.org/10.3390/iot5020012.

[15] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated Optimization in Heterogeneous Networks", Proc. 3rd MLSys Conference, Austin, TX, USA, Apr. 2020.

[16] M. G. Arivazhagan, V. Aggarwal, A. K. Singh, and S. Choudhary, "Federated Learning with Personalization Layers", arXiv preprint, arXiv:1912.00818, Dec. 2019. https://doi.org/10.48550/arXiv.1912.00818.

[17] J. Lee, S. Jeon, J. Bang, and H. Kim, "Accuracy-based Limited Participation Federated Learning Study", The Journal of Korean Institute of Information Technology, Vol. 21, No. 12, pp. 47-57, Dec. 2023. http://dx.doi.org/10.14801/jkiit.2023.21.12.47.

[18] M. Kim, D. Lee, Y. Kwon, and W. Lee, "Improvement of Federated Learning using Learning Agent and Multipath-based Model Split Transmission", Journal of Next-generation Convergence Technology Association, Vol. 8, No. 3, pp. 607-620, Mar. 2024. https://doi.org/10.33097/JNCTA.2024.08.3.607.

[19] M. S. Lee, I. R. Jeong, and J. Y. Chun, "Efficient and Secure Multi-Server Based Hierarchical Clustering Federated Learning on Non-IID Data", Asia-Pacific Journal of Convergent Research Interchange, Vol. 9, No. 12, pp. 11-19, Dec. 2023. http://dx.doi.org/10.47116/apjcri.2023.12.02.

[20] M. Abadi, A. Agarwal, P. Barham, et al., "A Study of BFLOAT16 for Deep Learning Training", Proc. Conference on Neural Information Processing Systems (NeurIPS), pp. 123-135, Dec. 2023. https://doi.org/10.48550/arXiv.1905.12322.

저자소개

김 동 현 (Donghyun Kim)

2019년 3월 ~ 현재 : 국립군산대학교 소프트웨어학과 학부과정

관심분야 : 인공지능, 연합학습, 데이터 분석

이 석 훈 (Sukhoon Lee)

2009년 2월 : 고려대학교 전자및정보공학부(학사)

2011년 2월 : 고려대학교 컴퓨터·전파통신공학과(공학석사)

2016년 2월 : 고려대학교 컴퓨터·전파통신공학과(공학박사)

2016년 3월 ~ 2017년 3월 : 아주대학교 의료정보학과 연구강사

2017년 4월 ~ 현재 : 국립군산대학교 소프트웨어학과 부교수

관심분야 : 사물인터넷, 메타데이터 레지스트리, 데이터 품질, 연합 학습