

# Deep Learning Personalization Methods in Peer-to-peer Heterogeneous Systems

---

Šajina, Robert

Doctoral thesis / Disertacija

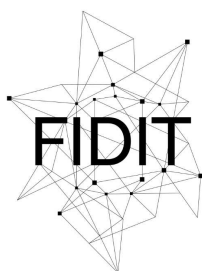
2025

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Rijeka / Sveučilište u Rijeci**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:195:812527>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2025-02-06**



Sveučilište u Rijeci  
**Fakultet informatike  
i digitalnih tehnologija**

Repository / Repozitorij:

[Repository of the University of Rijeka, Faculty of Informatics and Digital Technologies - INFORI Repository](#)



UNIVERSITY OF RIJEKA  
FACULTY OF INFORMATICS AND DIGITAL  
TECHNOLOGIES

Robert Šajina

**DEEP LEARNING PERSONALIZATION  
METHODS IN PEER-TO-PEER  
HETEROGENEOUS SYSTEMS**

DOCTORAL THESIS

Rijeka, 2025



UNIVERSITY OF RIJEKA  
FACULTY OF INFORMATICS AND DIGITAL  
TECHNOLOGIES

Robert Šajina

**DEEP LEARNING PERSONALIZATION  
METHODS IN PEER-TO-PEER  
HETEROGENEOUS SYSTEMS**

DOCTORAL THESIS

Supervisor: prof. dr. sc. Ivo Ipšić  
Faculty of Informatics and Digital Technologies, University of  
Rijeka

Co-supervisor: doc.dr. sc. Nikola Tanković  
Faculty of Informatics in Pula, Juraj Dobrila University of  
Pula

Rijeka, 2025



SVEUČILIŠTE U RIJECI  
FAKULTET INFORMATIKE I DIGITALNIH TEHNOLOGIJA

Robert Šajina

**METODE PERSONALIZACIJE DUBOKOG  
UČENJA U ISTORAZINSKIM  
HETEROGENIM SUSTAVIMA**

DOKTORSKI RAD

Mentor: prof. dr. sc. Ivo Ipšić  
Fakultet informatike i digitalnih tehnologija, Sveučilište u  
Rijeci

Komentor: doc.dr. sc. Nikola Tanković  
Fakultet informatike u Puli, Sveučilište Jurja Dobrile u Puli

Rijeka, 2025

Mentor: prof. dr. sc. Ivo Ipšić

Fakultet informatike i digitalnih tehnologija, Sveučilište u Rijeci

Komentor: doc.dr. sc. Nikola Tanković

Fakultet informatike u Puli, Sveučilište Jurja Dobrile u Puli

Doktorski rad obranjen je dana 13. siječnja, 2025. na Fakultetu informatike i digitalnih tehnologija, Sveučilište u Rijeci, pred povjerenstvom u sastavu:

1. Prof. dr. sc. Marina Ivašić-Kos, predsjednica,  
Fakultet informatike i digitalnih tehnologija, Sveučilište u Rijeci
2. Prof. dr.sc. Ivan Štajduhar, član,  
Tehnički fakultet, Sveučilište u Rijeci
3. Izv. prof. dr. sc. Darko Etinger, vanjski član,  
Fakultet informatike u Puli, Sveučilište Jurja Dobrile u Puli

## Zahvale

Zahvaljujem svima koji su na bilo koji način doprinijeli nastanku ove disertacije i pružili podršku tijekom mog doktorskog istraživanja.

Prije svega, izražavam duboku zahvalnost svom mentoru, prof. dr. sc. Ivi Ipšiću, na iznimnom vodstvu, strpljenju i stručnim savjetima koji su značajno usmjeravali i oblikovali moj istraživački rad.

Posebno zahvaljujem svom komentoru, doc. dr. sc. Nikoli Tankoviću, na kontinuiranoj podršci, inspiraciji i motivaciji, ne samo tijekom ovog istraživanja nego i u prethodnim fazama mog akademskog razvoja. Njegova predanost, ustrajnost i radišnost služili su mi kao primjer u trenucima suočavanja s izazovima.

Iskrena zahvalnost i mojoj obitelji na njihovoj i podršci razumijevanju tijekom cijelog mog doktorskog studija. Njihova vjera u mene i njihova prisutnost pružali su mi snagu i ohrabrenje u najzahtjevnijim trenucima.

Također, zahvaljujem svojim kolegama s Fakulteta informatike u Puli i Fakulteta informatike i digitalnih tehnologija na njihovoj suradnji, raspravama i podršci. Njihovi uvidi i zajednička razmjena ideja značajno su obogatili moje razumijevanje ove kompleksne teme.

# Abstract

Recent research has shown considerable interest in collaborative training of deep neural networks utilizing edge devices. Two predominant architectural paradigms for this training process include centrally orchestrated Federated Learning and fully decentralized peer-to-peer learning. Edge devices, termed agents, harbor local deep neural network models and distinct local datasets, composed of data collected specifically by each agent. While peer-to-peer learning techniques have been extensively investigated assuming independent and identically distributed (IID) data across agents, the learning efficacy significantly diminishes under non-IID assumptions, resulting in reduced model accuracy and slower convergence rates. The thesis aims to identify viable strategies for alleviating the impact of non-IID data on the overall learning process and to devise novel methodologies applicable in peer-to-peer deep learning contexts. These methodologies are subsequently evaluated using realistic non-IID datasets to assess their efficacy and applicability. The thesis will analyze autonomous personalized peer connection creation and present two methods of improving the peer-to-peer learning process in non-IID environments. The methods relate to improving peer-to-peer learning by enabling multi-task collaboration between agents learning two distinct tasks, and improving agent's local model performance by a personalization technique. The results indicate a statistically significant increase of 11.6% in the mean relative accuracy for the proposed multi-task technique, and 16.9%-29.8% relative accuracy increase (depending on the topology) for the personalization technique. Compared to existing approaches, presented methods can be used to enhance the performance and scalability of peer-to-peer learning systems, and improve personalization resulting in greater model accuracy in diverse real-world scenarios.

**Keywords:** peer-to-peer, non-IID, machine learning, natural language processing, personalization, multi-task

## Prošireni sažetak

Suradničko obučavanje dubokih neuronskih mreža na rubnim (mobilnim i ugradbenim) uređajima izazvalo je znatan interes u nedavnoj literaturi, pri čemu su istaknute dvije vodeće paradigme: centralno savezno (eng. *Federated Learning*) i potpuno decentralizirano učenje na istorazinskim rubnim uređajima. Rubni uređaji, nazvani agentima, posjeduju lokalne modele dubokih neuronskih mreža i različite lokalne skupove podataka. Iako su tehnike učenja istorazinskih agenata detaljno istražene pod pretpostavkom identično distribuiranih i neovisnih podataka, njihova učinkovitost opada pod pretpostavkom da su podaci neovisno i nejednako distribuirani (NND), što rezultira smanjenom točnošću modela i sporijim konvergenijskim stopama. Stoga je cilj ovog istraživanja razviti postupke i metode za ublažavanje utjecaja NND podataka na ukupni proces učenja te osmisliti nove metodologije primjenjive u kontekstu istorazinskog učenja među agentima, s fokusom na uporabu realističnih skupova podataka.

Decentralizirano učenje među istorazinskim agentima je paradigma učenja koja se oslanja na razmjenu lokalnih modela među agentima prateći neku mrežnu topologiju. Mrežna topologija obično je unaprijed definirana, a putem nje su utvrđene veze između agenata, koje se koriste za međusobnu komunikaciju. Tijekom komunikacije, agenti razmjenjuju samo svoje lokalne modele, dok se lokalni podaci agenata nikada ne razmjenjuju. Priljubljeni modeli susjednih agenata agregiraju se s lokalnim modelom na način da se izračuna prosjek svih priljubljenih modela (u slučaju više priljubljenih) zajedno s lokalnim modelom, stvarajući tako novi model. Alternativno, svaki priljubljeni model može se direktno agregirati s lokalnim modelom te na taj način dobiti novi lokalni model. Ciklički proces u kojem svaki agent najprije lokalno trenira svoj model na vlastitim lokalnim podacima, a zatim razmjenjuje taj model sa susjednim agentima, ponavlja se sve dok se ne postigne određeni kriterij zaustavljanja, obično vezan uz konvergenciju modela.

Heterogenost lokalnih skupova podataka među agentima ima značajan utjecaj na konačne vrijednosti parametara modela nakon lokalnog treniranja. Različite distribucije, karakteristike i veličine podataka mogu rezultirati različitim lokalnim modelima, čak i ako su agenti početno inicijalizirani s istim parametrima modela. Kada dođe do razmjene lokalnih modela među različitim agentima, proces agregacije modela koji imaju značajne razlike u parametrima može rezultirati stvaranjem novog modela koji ima znatno slabije performanse. Ovaj fenomen

direktno utječe na stabilnost procesa učenja, brzinu kojom modeli konvergiraju te na općenite sposobnosti generalizacije konačnih modela.

Ova disertacija pisana je prema takozvanom Skandinavskom modelu u sklopu kojeg su objavljena tri znanstvena rada koja prikazuju originalni doprinos u području decentraliziranog istorazinskog učenja između agenata.

Prvi rad prikazuje decentraliziranu varijantu istorazinskog učenja agenata s prilagodbom postojeće tehnike *gossip averaging* u kombinaciji s normalizacijskim slojevima koji čine sastavni dio arhitekture modela. Ispituje se učinkovitost normalizacijskih slojeva u ublažavanju negativnog utjecaja NND podataka među decentraliziranim agentima. Uz to, uvodi se i varijanta tehnike uranjenog zaustavljanja, koja u kombinaciji s normalizacijskim slojevima djeluje kao personalizacijska tehnika za fino podešavanje lokalnog modela agenata. Predložena metoda evaluirana je kroz brojne simulacije koristeći zadatak predikcije sljedeće riječi na korisničkim komentarima iz skupova podataka Reddit i StackOverflow. Rezultati simulacija pokazuju da predložena metoda u prosjeku postiže relativno povećanje točnosti između 16.9%-29.8% u usporedbi s najboljim baznim decentraliziranim pristupom učenja, u različitim mrežnim topologijama.

Drugi rad istražuje primjenu tehnike višezadačnog učenja za rješavanje dva zasebna zadatka u obradi prirodnog jezika. Predstavlja se nova metoda koji koristi transformer arhitekturu koja se sastoji samo od enkodera, kako bi se omogućila suradnja između agenata koji uče različite zadatke. Metoda je evaluirana simuliranjem različitih skupina agenata koji uče različite zadatke kako bi se ispitalo na koji način se može ostvariti međusobna korisnost djeljenja modela između svih agenata i na taj način ostvariti bolje rezultate lokalnih modela. Simulacije provedene u radu pokazuju da suradnja među agentima, čak i kada agenti uče različite zadatke, može poboljšati lokalnu točnost modela svih agenata, posebno kada su veze između agenata pažljivo razmotrene i ograničene. Višezadačna suradnja dovela je do statistički značajnog povećanja od 11.6% u prosječnoj relativnoj točnosti u usporedbi s rezultatima baznih eksperimenata za pojedinačne zadatke.

U trećem radu se istražuje autonomno uspostavljanje veza između agenata tijekom decentraliziranog učenja u kontekstu NND skupova podataka među agentima, u sintetičkim i stvarnim okruženjima. Fokus je na analizi učinkovitosti različitih metodologija u scenarijima koji obuhvaćaju zadatke klasifikacije slika i obrade prirodnog jezika. Kroz eksperimente provedene u sintetičkim i realnim NND okruženjima, metode PANMGrad i PANMLoss pokazale su se kao

optimalna rješenja, pokazujući kako učinkovitost komunikacije tako i otpornost na tendencije centralizacije.

Publikacije zajednički čine koherentno tijelo rada i doprinose shvaćanju procesa decentraliziranog učenja među istorazinskim agentima u kontekstu heterogenih skupova podataka. Jedan od radova analizira strategije za autonomno uspostavljanje komunikacijskih veza između agenata, dok se u druga dva rada predlažu nove metode temeljene na poboljšanju performansi učenja kroz višezadaćno učenje ili personalizaciju lokalnih modela. Iz navedenog proizlaze sljedeći znanstveni doprinosi:

1. Metoda uspostave veza i razmjene modela između heterogenih agenata i različitih ciljeva učenja,
2. Metoda treniranja zajedničkih slojeva modela za brzu konvergenciju u okruženju s više ciljeva učenja,
3. Metoda personalizacije modela istorazinskih agenata temeljena na normalizacijskim slojevima,
4. Evaluacija, usporedba i identifikacija optimalnih metoda za autonomno uspostavljanje veza između agenata nad sintetičkim i realističnim podacima.

**Ključne riječi:** istorazinsko učenje, neovisno i nejednako distribuirani skupovi podataka, strojno učenje, obrada prirodnog jezika, personalizacija, višezadaćno učenje

# Contents

|   |           |
|---|-----------|
| <b>1. INTRODUCTION</b>  | <b>1</b>  |
| 1.1. Peer-to-peer learning . . . . .  | 2         |
| 1.1.1. Network topology . . . . .   | 3         |
| 1.1.2. Synchronization . . . . .  | 6         |
| 1.1.3. Learning process and objective . . . . .   | 7         |
| 1.2. Non-IID environment . . . . .  | 9         |
| 1.3. Research motivation . . . . .  | 12        |
| 1.4. Research overview . . . . .  | 13        |
| 1.5. Thesis objectives, hypotheses and scientific contributions . . . . .                                 | 14        |
| <b>2. ELABORATION</b>   | <b>17</b> |
| 2.1. An overview of conducted studies . . . . .   | 17        |
| 2.1.1. Peer-to-peer learning method and personalization technique for non-IID data environments . . . . . | 20        |
| 2.1.1.1. Introduction . . . . .   | 20        |
| 2.1.1.2. Peer-to-peer learning method . . . . .   | 23        |
| 2.1.1.3. Personalization method . . . . .   | 25        |
| 2.1.1.4. Impact of Batch Normalization on peer-to-peer learning in non-IID environments . . . . .         | 26        |
| 2.1.1.5. Discussion . . . . .   | 27        |
| 2.1.2. Multi-task peer-to-peer learning method . . . . .  | 30        |
| 2.1.2.1. Introduction . . . . .   | 30        |
| 2.1.2.2. Multi-task learning method . . . . .   | 33        |
| 2.1.2.3. Topology organization method . . . . .   | 34        |
| 2.1.2.4. Shared model layers training method . . . . .  | 36        |
| 2.1.2.5. Discussion . . . . .   | 39        |
| 2.1.3. Autonomous connection establishment methods . . . . .  | 41        |
| 2.1.3.1. Introduction . . . . .   | 41        |
| 2.1.3.2. Personalized autonomous peer connection establishment methods . . . . .                          | 46        |
| 2.1.3.3. Evaluation methodology . . . . .   | 47        |



|   |            |
|---|------------|
| 2.1.3.4. Experimental results . . . . .   | 48         |
| 2.1.3.5. Discussion . . . . .   | 52         |
| <b>3. CONCLUSION</b>  | <b>54</b>  |
| 3.1. Scientific contribution . . . . .  | 55         |
| 3.2. Future research directions . . . . .   | 56         |
| <b>4. ABSTRACTS OF ARTICLES</b>   | <b>58</b>  |
| 4.1. Peer-to-peer deep learning with non-IID data . . . . .   | 58         |
| 4.2. Multi-task peer-to-peer learning using an encoder-only transformer model . . .                                       | 60         |
| 4.3. An Overview of Autonomous Connection Establishment Methods in Peer-to-<br>Peer Deep Learning . . . . .               | 61         |
| <b>REFERENCES</b>   | <b>62</b>  |
| <b>LIST OF FIGURES</b>  | <b>75</b>  |
| <b>LIST OF TABLES</b>   | <b>76</b>  |
| <b>APPENDIX</b>   | <b>77</b>  |
| <b>Appendix A. Peer-to-peer deep learning with non-IID data</b>   | <b>78</b>  |
| <b>Appendix B. Multi-task peer-to-peer learning using an encoder-only trans-<br/>        former model</b>                 | <b>109</b> |
| <b>Appendix C. An Overview of Autonomous Connection Establishment Meth-<br/>        ods in Peer-to-Peer Deep Learning</b> | <b>135</b> |
| <b>Appendix D. Supplemental Materials</b>   | <b>174</b> |

# 1. INTRODUCTION

The proliferation of interconnected edge devices, encompassing mobile phones, tablets, and laptops, entails a wealth of valuable yet often personal data. Although there are various definitions of personal, private and sensitive data, personal data generally includes both sensitive and private information [1]. Throughout this dissertation, a user’s local data is assumed to include personal, sensitive information. Advanced machine learning models can leverage this data from edge devices by analyzing user behavior patterns, preferences, and contextual information to enhance user experiences. For instance, intelligent recommender systems utilize historical user interactions to suggest personalized content, while voice recognition systems adapt to individual accents and speech patterns for improved accuracy. This trend has increased research interest in using local data and computational capabilities of edge devices for collaborative model training with strong privacy safeguards [2].

Two primary methodologies for collaborative learning have been established: Federated Learning (FL) [3] and peer-to-peer (P2P) learning [4]. Federated Learning coordinates the training of a shared model through a centralized process utilizing data from edge devices. In contrast, P2P learning techniques rely on a predefined protocol followed by each edge device, termed agent. Dispensing with the need for a central parameter server, the P2P approach fosters a system more resilient to the challenges inherent in centralized methodologies; nonetheless, it confronts additional issues such as synchronization between agents.

An edge device includes any hardware unit equipped with memory, computational capabilities, internet connectivity, and potentially valuable datasets suitable for model training [3]. These components facilitate local model training on the device using its local dataset. Collaborative learning aims to ensure data privacy by only sharing the model with the outside world while keeping the local data private [3]. From hereafter, we refer to an edge device as an agent. Each agent houses private local data, a machine learning framework like TensorFlow [5] or PyTorch [6], and a pre-trained or randomly initialized machine learning model (see Figure 1). The architectural design and model parameters are typically established prior to the start of the learning process.

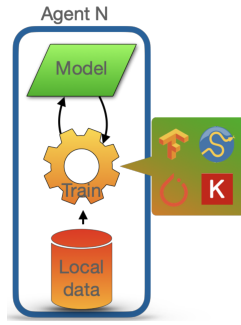


Figure 1: Agent contains local data, a machine learning framework, and a model trained by the ML framework.

## 1.1. Peer-to-peer learning

In its simplest form, peer-to-peer learning follows a cyclical process in which all agents initially conduct a predefined number of local model training iterations. This is succeeded by a communication phase, where agents exchange their local models with peers. The received models are aggregated to form a new local model, which is then utilized in the subsequent training phase. Figure 2 shows the described cyclical process.

The efficacy of the peer learning process is significantly dependent on the nature of local data accessible to agents. When local data follow an identically distributed and independent (IID) distribution, the learning process becomes more streamlined. In this scenario, all agents train their local models on comparable data, resulting in models with similar parameters. Subsequently, during the communication phase and the exchange of models between agents, the ensuing local models represent an enhancement over the previous iteration of the model parameters. This improvement is attributable to the similarity in the model parameters between agents induced by the similarity in their respective datasets. Non-IID data, as observed in real-world scenarios, present a more realistic and prevalent challenge to the peer-to-peer learning process [7, 8]. This divergence from identically distributed and independent data substantially disrupts the agent learning processes, resulting in compromised performance outcomes.

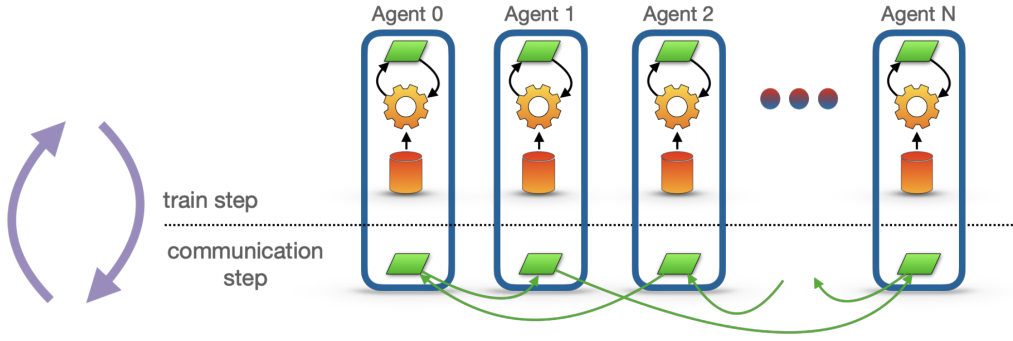


Figure 2: Synchronous peer-to-peer learning process [9].

### 1.1.1. Network topology

In a decentralized peer-to-peer system, agents engage in information exchange, represented by the transmission of model parameters, through connections established between themselves. These connections are encapsulated within a communication graph:

$$G = (\llbracket N \rrbracket, E, W) \quad (1)$$

where  $\llbracket N \rrbracket = \{1, \dots, N\}$  denotes the set of all nodes in the network,  $E \in \llbracket N \rrbracket \times \llbracket N \rrbracket$  represents the set of edges, and  $W \in \mathbb{R}^{N \times N}$  constitutes a nonnegative weighted matrix. The weight of an edge  $(i, j) \in E$  is denoted by  $W_{ij}$ , following the convention  $W_{ij} = 0$  if  $(i, j) \notin E$  or  $i = j$ . An agent  $i$  transmits messages solely to agent  $j$  if  $W_{ij} > 0$ , indicating that agent  $i$  communicates exclusively with neighbors (peers) within the set  $N_i = \{j : W_{ij} > 0\}$ , without awareness of other non-connected neighbors ( $W_{ij} = 0$ ) in the network. When visualized, the communication graph  $G$  illustrates the connections among all agents in the network.

Consider the following communication matrix,  $W_1 \in \mathbb{R}^{6 \times 6}$ , representing communication among six agents, each communicating with three neighbors:

$$W_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}. \quad (2)$$

Column indices of matrix  $W_1$  correspond to sending agents, while the row indices correspond to receiving agents. For the agent with index 0, denoted as  $\text{Agent}_0$ , the sending indices (column at index 0) are  $\{0, 1, 0, 0, 1, 1\}$ , indicating that  $\text{Agent}_0$  sends its message to agents  $\text{Agent}_1$ ,  $\text{Agent}_4$  and  $\text{Agent}_5$ . The row at index 0 shows that  $\text{Agent}_0$  receives messages from agents  $\text{Agent}_1$ ,  $\text{Agent}_4$  and  $\text{Agent}_5$ . This matrix example exemplifies an undirected/symmetric communication graph, often utilized in various peer-to-peer learning methods [10–22]. Figure 3 visualizes the connections from the  $W_1$  communication graph.

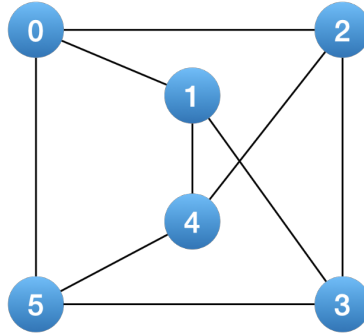


Figure 3: Visual communication representation of the  $W_1$  undirected communication graph.

Conversely, a directed/asymmetric communication graph implies that an agent can send messages to a set of neighbors but receive messages from a completely different set of neighbors. In a directed graph, an agent has an in-neighbor if  $(i, j) \in E$  and an out-neighbor if  $(j, i) \in E$ . The out-neighbor set represents the agents to which the agent sends messages, while the in-neighbor set represents the agents from which messages are received. Generally, the number of in-neighbors and out-neighbors is identical. The following communication matrix  $W_2 \in \mathbb{R}^{6 \times 6}$  illustrates connections between agents in a directed communication scheme, where each agent maintains three neighbor connections.

$$W_2 = \begin{pmatrix} 0 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} \quad (3)$$

Column indices of matrix  $W_2$  still correspond to sending agents, with Agent<sub>0</sub> (column at index 0) sending messages to agents Agent<sub>1</sub>, Agent<sub>2</sub> and Agent<sub>3</sub>, and receiving messages from agents Agent<sub>1</sub>, Agent<sub>3</sub> and Agent<sub>4</sub>. Agent Agent<sub>0</sub> communicates in an undirected manner with agents Agent<sub>1</sub> and Agent<sub>3</sub>. However, this occurrence is primarily attributable to the small number of overall agents relative to the number of neighbor connections per agent. The  $W_2$  communication matrix is visually depicted in Figure 4.

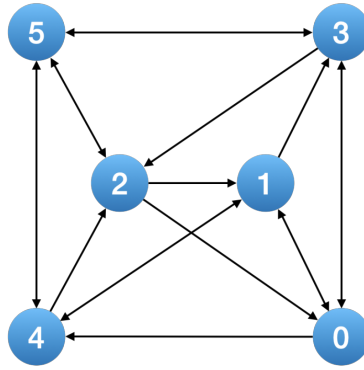


Figure 4: Visual communication representation of the  $W_2$  directed communication graph.

Network topology refers to the method of generating communication matrices according to specific conditions and rules. In scientific literature, network topology is typically predefined [11, 19–21, 23], whereby all connections between agents are established in advance. However, recent research has begun exploring the possibility of autonomous creation of neighbor connections [24–28]. The potential for dynamic and autonomous creation of neighbor connections emerges as a more practical solution for applications in a realistic and dynamic environment of decentralized agents. Several common network topologies are depicted in Figure 5. The number of in-out neighbors can vary depending on the chosen topology. Ring, fully connected, or torus topologies inherently restrict customization of the number of neighbors. In a ring topology, each agent has two neighbors in undirected communication and one neighbor in directed communi-

cation. Conversely, in a fully connected graph, there is no distinction between undirected and directed communication, but the number of neighbors is fixed at  $N - 1$  in a network comprising  $N$  fully connected agents. Sparse networks afford each agent the flexibility to determine the number of neighbors [10, 12–18, 22, 29–31].

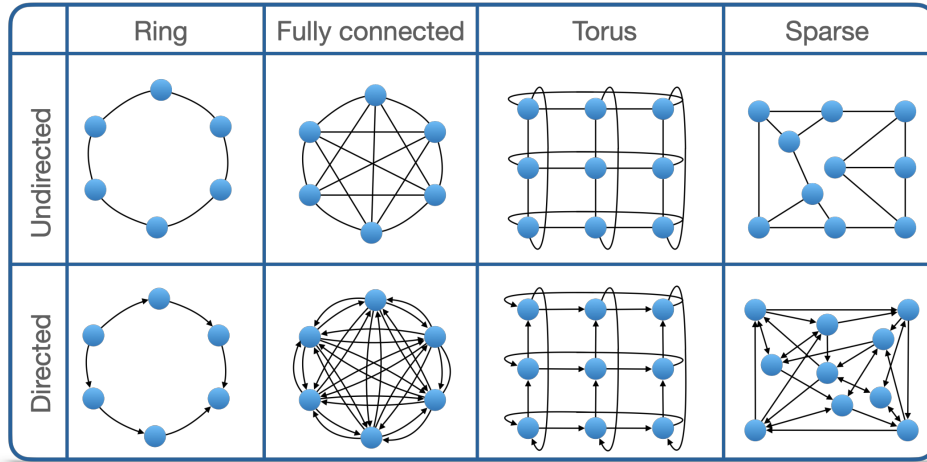


Figure 5: Examples of most common network typologies in an undirected and directed communication [32].

### 1.1.2. Synchronization

Synchronization plays a pivotal role in decentralized systems, dictating the strategy that each agent adheres to when communicating with its neighbors. In synchronous systems, a synchronization barrier often halts the learning process of an agent while all messages are exchanged. This approach may involve sending messages to all out-neighbors and waiting for responses from all in-neighbors. As all agents must execute the same steps, the learning process of each agent can only resume once all messages have been exchanged between neighbors. Given that decentralized approaches do not entail central authority overseeing the learning process, agents may opt to use a central clock for synchronization purposes. Alternatively, the asynchronous time model [33] enables each agent to independently track time with its own clock, ticking at a Poisson distribution [10, 15, 31]. Considering all local clocks as independent and identically distributed (IID), they collectively function as a single global clock ticking at a rate of  $N$  Poisson processes governed by a Poisson distribution, awakening one or more network agents uniformly at random. This uniform clock proves useful when two agents must awaken simultaneously to

communicate.

Different approaches involve varying message content exchanged between agents; typically, an agent is required to send its message to all its out-neighbors and receive messages from all its in-neighbors. A loss of a message during this phase can potentially halt the overall learning process, as an agent may indefinitely await a message that never arrives. Such a stalled agent could trigger a cascade effect, stalling all other agents in turn. This phenomenon, which we termed synchronous undirected communication, is characterized by a synchronization barrier where all messages are exchanged simultaneously [10–12, 17–19, 21, 22, 29]. However, an approach proposed by Assran *et al.* [29] aims to mitigate this issue by potentially delaying synchronization for a specific number of iterations, allowing messages to be processed as they arrive. Similarly, an asynchronous approach proposed by Guo *et al.* [18] enables agents to query which neighbors will participate in the following communication step, thereby reducing the synchronization burden. If an approach features a synchronization barrier but supports directed communication, we term it as synchronous directed communication. Here, communication is directed, but synchronization is still required, necessitating the reception of all in-neighbor messages before continuing the learning process [23, 29, 30]. Another scenario arises when two agents communicate asynchronously without disrupting the learning process but still require message exchange between them to occur in the same time interval. We term this case as asynchronous undirected communication [10, 13–16, 20, 34, 35]. Several approaches modify the communication step to allow an agent to continue the training process and process messages upon arrival. We term this as asynchronous directed communication. Among the four, the most permissive approach is asynchronous directed communication [31, 36, 37], where communication is asynchronous, does not disrupt the learning process, and does not require a reply message from the receiving agent. Each agent conducts local training steps uninterrupted and aggregates contributions received from in-neighbors as they arrive.

### 1.1.3. Learning process and objective

Each agent aims to train its local model by utilizing its local dataset  $D_i$  and information received from its neighbors to minimize its average loss function  $F_i$ . To achieve this, an agent employs its local data  $D_i$  to train a local model by computing the mini-batch gradient



$\nabla F_i(x_i; \xi_i)$  where  $\xi_i \sim D_i$ , and updating its local model using the gradient descent update rule  $x_i = x_i - \eta F_i(x_i; \xi_i)$ ,  $\xi_i \sim D_i$ , where  $\eta$  denotes the learning rate,  $\xi_i$  denotes a local data batch sampled from  $D_i$ , and  $x_i$  denotes agent’s model parameters. This process is repeated for  $E$  batch iterations, where  $E$  can be set to one for communication after each training batch or increased to any arbitrary number to increase local computation and reduce communication frequency. After the local training step, agents engage in a communication step where they exchange information. Algorithm 1 [32] illustrates an abstract agent training scheme. At the beginning of the training process, agents initialize their model parameters with identical values, which facilitates a faster and smoother learning curve [37]. A communication matrix  $W$  specifies the connections between agents, and the parameter  $E$  specifies the number of local batch training iterations before agent communication. Each agent first conducts local model training using its local data, followed by a communication step in which each agent sends and may receive model updates from its neighbors. This cyclic process continues for a predetermined and arbitrary number of rounds.

---

**Algorithm 1** Agents abstract training process according to [32]

---

**Require:**

- Initialize  $\eta > 0$ , agents  $A$ , communication matrix  $W$ , number of local batch iterations  $E$
- $x_i = 0$  for all agents  $i \in A$  ▷ Initialize all agents models to identical value
- 1: **repeat** for agent  $i \in A$  ▷ In parallel
- 2:     **for**  $e = 0, 1, 2, \dots, E$ , at agent,  $i$  **do**
- 3:          $\xi_i \sim D_i$  ▷ Sample new mini-batch from local distribution
- 4:          $x_i = x_i - \eta F_i(x_i; \xi_i)$  ▷ Train model on batch
- 5:         SEND( $x_i W_{ji}$ ) ▷ Send model to peers
- 6:         RECEIVE( $x_j W_{ij}$ ) ▷ Receive model from peers
- 7:          $x_i = \text{AGGREGATE}(x_j W_{ij})$
- 8: **until** Maximum iteration reached

**Output:**  $\frac{1}{N} \sum_{i=1}^N x_i$ ,  $N = |A|$  or  $x_i \forall i \in A$  ▷ Output is either one global model produced by averaging all models or personal model for each agent

---

The primary objective of collaborative learning is typically to generate a single global model by averaging the model parameters of all agents at the conclusion of the training process:

$$x_{global} = \frac{1}{N} \sum_{i=1}^N x_i. \quad (4)$$

Producing a single global model is typically favored in large-scale machine learning facilities to expedite the learning process by leveraging multiple training agents [10, 11, 19–22, 29, 30,

37]. However, alternatively, each agent may pursue its own learning objective, resulting in one model per agent [13–18, 23–25, 27, 28, 31, 35, 38, 39]. In this scenario, each model is tailored to the specific characteristics of its respective agent, exhibiting optimal performance when utilized by the agent who trained it. However, if deployed on an agent with a different data distribution, the model may yield suboptimal performance.

## 1.2. Non-IID environment

In collaborative learning systems (FL and P2P), the assumption of identically and independently distributed (IID) data [40] is often challenged by the reality of real-world datasets [41]. Non-IID data, characterized by varying distributions and dependencies across different agents, presents a significant challenge in collaborative learning scenarios. Non-IID data pertains to datasets wherein the distribution of data across agents or devices is non-uniform, resulting in distinct data distributions among different agents and potentially leading to heterogeneous local datasets. According to Ma *et al.* [7], types of data heterogeneity in decentralized systems are: feature distribution skew, label distribution skew, same label different features, same feature different labels, and quantity skew.

To examine various forms of data heterogeneity, we will investigate an image classification task involving three classes, each representing different geometrical shape (square, circle, triangle) in an environment with two agents.

**IID environment:** In an IID environment, both agents exhibit an identical distribution of class examples with consistent features. Figure 6 illustrates the data distribution in an IID environment, wherein both agents possess an equal number of examples per class, and each class is represented by identical features.










| Agent 0           |   |   |   | Agent 1           |   |   |   |
|-------------------|---|---|---|-------------------|---|---|---|
| Feature           |  |  |  | Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  | Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  | Number of samples | 10  | 5   | 10  |

Figure 6: Data distribution in IID environment.

**Feature distribution skew:** In scenarios where an agent’s local data displays feature distribution, the representation of certain features may vary from one agent to another, while the distribution of examples per class remains consistent across all agents. In Figure 7, both agents possess an equal number of data examples per class; however, the features pertaining to the *Square* and *Triangle* classes differ between *Agent 1* and *Agent 0*. Despite the variation in features, such as *Agent 0* possessing dark blue squares and *Agent 1* possessing light blue squares, all data examples represent the same class, namely *Square*.

|                   |   |   |   |
|-------------------|---|---|---|
| Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  |







|                   |   |   |   |
|-------------------|---|---|---|
| Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  |

Figure 7: Feature distribution skew in non-IID environment.

**Label distribution skew:** Label distribution skew denotes varying amounts of data examples per class within the local datasets of agents. In Figure 8, *Agent 0* possesses 100 examples of the *Square* class and 5 examples of each of the other classes. Conversely, *Agent 1* contains 100 examples of the *Circle* class, along with 3 examples of the *Square* class and 5 examples of the *Triangle* class. Label distribution skew is apparent when the distribution of all classes are compared between the two agents. However, if  $y = \textit{Triangle}$ , both agents share an equal probability for a feature  $x$  to belong to the *Triangle* class.

|                   |   |   |   |
|-------------------|---|---|---|
| Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 100   | 5   | 5   |







|                   |   |   |   |
|-------------------|---|---|---|
| Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 3   | 100   | 5   |

Figure 8: Label distribution skew in non-IID environment.

**Same label, different features:** Different features associated with the same label indicate that the distribution of features varies across different agents, while the distribution of examples per class remains consistent. In Figure 9, *Agent 1* exhibits different features for the *Square* and *Triangle* classes compared to *Agent 0*. Nevertheless, the number of data examples per class

remains unchanged between the two agents. Unlike feature distribution skew, where differences are subtle (e.g., color variation), this scenario involves more pronounced distinctions, such as alternate borders or entirely different shapes associated with each label.

| Feature           |  |  |  |
|-------------------|---|---|---|
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  |







| Feature           |  |  |  |
|-------------------|---|---|---|
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  |

Figure 9: Same label, different features in non-IID environment.

**Same features, different label:** Different labels associated with the same feature indicate that the distribution of labels associated with a certain feature varies across different agents, while the distribution of examples per class remains consistent. In Figure 10, *Agent 0* assigns different labels to identical features, compared to *Agent 1*. For instance, features representing a square geometric shape are labeled as *Square* by *Agent 0*, whereas *Agent 1* labels them as *Circle*. This discrepancy applies to all other data in this example; for each feature, each agent assigns a different label. However, the data representing each feature is identical between the two agents.

| Feature           |  |  |  |
|-------------------|---|---|---|
| Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  |




| Feature           |  |  |  |
|-------------------|---|---|---|
| Label             | Circle  | Triangle  | Square  |
| Number of samples | 10  | 5   | 10  |

Figure 10: Same features, different label in non-IID environment.

**Quantity skew:** Large disparities in the quantity of available data examples per class are referred to as quantity skew. Figure 11 illustrates a scenario where *Agent 1* possesses a significantly different number of data examples compared to *Agent 0*.

It's crucial to recognize that non-IID environments can manifest varying degrees of each type of data heterogeneity, rather than being limited to a single type. This diversity in heterogeneity amplifies the differences in the local datasets of agents even further.







| Agent 0           |   |   |   | Agent 1           |   |   |   |
|-------------------|---|---|---|-------------------|---|---|---|
| Feature           |  |  |  | Feature           |  |  |  |
| Label             | Square  | Circle  | Triangle  | Label             | Square  | Circle  | Triangle  |
| Number of samples | 10  | 5   | 10  | Number of samples | 10000000  | 500000  | 1000  |

Figure 11: Quantity skew in non-IID environment.

### 1.3. Research motivation

Non-IID data present a big challenge to the applicability of peer-to-peer learning systems in realistic world applications [7]. Non-equality in data distribution between decentralized agents significantly impacts the overall learning process which negatively impacts the usability of such learning methods in practice. Related studies already proposed some methods and techniques that try to minimize the negative impact of the non-IID on the learning process through personalization [42, 43], multi-task learning [2, 44–49] and meta-learning [50–52]. All methods can be perceived as forms of personalization techniques aimed at optimizing each agent’s model performance within non-IID environments.

However, related research frequently concentrates solely on simulated synthetic non-IID environments, achieved by artificially inducing specific data transformations [24–28]. According to Ma et al. [7], among the most frequently employed datasets in FL, which is also susceptible to the challenges posed by non-IID environments, are image classification datasets like MNIST [53], CIFAR-10 [54], and Fashion-MNIST [55]. Synthetic non-IID environments are crafted by introducing data manipulations, such as image rotation, to achieve skewed feature distributions, label permutations across features to induce heterogeneity in same-feature, different-label scenarios, or by creating quantity skew through data examples partitioning, where certain agents possess examples of specific classes in substantial quantities while having minimal or no examples from the remaining classes. Such simulated environments may not align closely with realistic non-IID datasets. Hence, there exists a gap in research that necessitates attention, specifically in the realm of realistic non-IID environments within peer-to-peer learning systems.

## 1.4. Research overview

This doctoral dissertation will explore three key areas of personalized peer-to-peer learning aimed at mitigating the adverse effects of non-IID data within peer-to-peer learning environments. Specifically, the research will focus on on-device personalization, multi-task learning, and personalized peer connection establishment within peer-to-peer environments.

The most advances regarding on-device personalization were made within FL. Studies have shown that model personalization, achieved by fine-tuning agents' local models post-FL training, enhances local model accuracy [42, 43]. Batch Normalization (BN) [56] layers have also been used for local personalization in FL with certain modifications, such as not averaging BN parameters. In approaches like FedBN [57] and MTFB [58], BN layers are stored locally by each agent and excluded from FL averaging, while SiloBN [59] employs standard FL for averaging all model parameters, including BN layers. However, in the context of peer-to-peer learning environments, there are no existing studies that explore the use of BN layers for model personalization.

In multi-task learning scenarios, the objective is to group agents based on their heterogeneity properties to minimize the negative effects of non-IID data. Various strategies for organizing multi-task learning have been proposed. In FL, the most common approach involves storing one part of the model locally on the agents while sharing the other part with a central server, as demonstrated in several studies [50, 57, 58, 60, 61]. By splitting the model, agents retain both general and task-specific knowledge. An extreme approach, studied by [35], allows any part of the neural network model to be shared among peers in peer-to-peer environments. This method achieved a relative accuracy increase of approximately 9% with 80% of model parameters used in averaging, compared to 100% averaging on the MNIST [53] and FEMNIST [62] datasets. An alternative FL method explored grouping agents based on the similarity of their models, where only the models within the same group are aggregated to form a new group-global model [48]. In light of current research, the primary objective has consistently been to train a single task across all agents. However, it remains unexplored whether multi-task learning can be effectively applied in environments with such high heterogeneity between agents that they are essentially learning different tasks. This can be examined by facilitating peer-to-peer learning among agents working on distinct tasks within the same data modality, allowing for a more

nuanced understanding of how collaboration can be leveraged despite task variation.

Multi-task learning can also be facilitated by strategically manipulating agent connections and clustering agents with similar data [39] or model properties [24–28, 38]. By establishing personalized connections only between agents with comparable data or model characteristics, the negative impact of heterogeneity is reduced, thereby improving the overall learning outcomes. However, several challenges may arise when enabling agents to form personalized connections, such as accurately and efficiently identifying similar peers, mitigating communication imbalances, and preventing tendencies toward centralization. Current research does not provide sufficient data on these issues for existing methods of personalized peer connection establishment.

## **1.5. Thesis objectives, hypotheses and scientific contributions**

The research aims to explore and enhance existing techniques for mitigating the adverse effects of non-IID data on the learning process of agents within peer-to-peer learning systems. The objective is to investigate the feasibility of adapting personalization techniques from FL and to explore potential benefits of collaboration among clusters of agents addressing different tasks. The primary goal is to enable more effective peer-to-peer learning in non-IID environments, thereby increasing the practical applicability of such systems in real-world scenarios, particularly for independent researchers and organizations lacking the necessary equipment, software, and resources to establish and maintain servers managing the FL process.

A sequence of studies were conducted within this doctoral thesis to either devise personalization techniques or assess methods for autonomous connection establishment in peer-to-peer environments within realistic non-IID scenarios. The studies primarily focus on mitigating the adverse effects of non-IID data in peer-to-peer systems, primarily through personalized learning processes for individual agents. These approaches include both model personalization and the establishment of personalized peer connections.

The first study [9] investigates the efficacy of integrating BN [56] layers into the model architecture as a means to alleviate the negative impact of non-IID data and enhance the learning process of agents. BN layers are widely employed in training deep neural network (DNN) models [63–66] and have been demonstrated to be beneficial in FL as a personalization tech-

nique [57, 58, 67, 68]. However, additional methods are required when utilizing BN layers in standard FL under non-IID constraints to achieve model convergence. Realistic user-generated datasets Reddit [62] and StackOverflow [69] are utilized to evaluate the advantages of incorporating BN layers in peer-to-peer learning. Previous research has illustrated the considerable representational capacity of BN layers, as evidenced by achieving high test accuracy solely by training these layers on a randomly initialized model [70, 71]. Therefore, the first study aims to explore whether a similar methodology can be employed within a peer-to-peer environment as a personalization technique. The hypothesis is as follows:

**Hypothesis 1: Personalizing normalization layers in peer-to-peer learning systems positively affects local accuracy on realistic data.**

The second study [72] delves into an extreme scenario of data heterogeneity, where distinct groups of agents are engaged in learning entirely different tasks within a multi-task environment. The objective of the study is to ascertain whether all agents can engage in collaboration and exchange information (model parameters) such that the collaborative effort yields benefits for all agents without incurring any communication or computation overhead. Building upon previous demonstrations of clustering agents based on model and data similarity conditions [15, 35, 45, 46, 73], the study seeks to ascertain whether an effective method exists for establishing connections between agents that enhances the average accuracy and convergence of individual local models in a multi-task environment with entirely different tasks. The following two hypotheses were tested in the study:

**Hypothesis 2: Collaboration between agents learning different tasks within the same data modality positively affects the average accuracy and convergence of individual models in the context of multi-task learning.**

**Hypothesis 3: Implementing a method for determining connections between agents positively affects the average accuracy and convergence of individual models in the context of multi-task learning.**



The third study [32] conducts an empirical evaluation and comparison of methods for autonomously establishing connections between agents in a peer-to-peer system. Recent research has witnessed a surge in studies focusing on devising methods for a personalized approach to creating peer connections in non-IID environments, fostering communication among agents sharing similar properties such as data distribution or model parameters. Empirical investigations carried out in synthetic non-IID environments have demonstrated promising benefits of the proposed methods across various types of data heterogeneity, outperforming the baseline approach of random connection creation. This study aims to compare the latest methods in both simulated synthetic and realistic non-IID environments. The average model accuracy metric is utilized to measure the average local model accuracy, while the efficiency of methods' communication is measured by the number of exchanged messages. Furthermore, the Gini coefficient is employed to quantify the average communication balance of an agent [74–76]. The optimal method, based on achieved accuracy and the number of messages required to attain top accuracy, is determined using the Pareto method. Based on previous research results and the stated objective of the study, the following hypothesis is tested by the study:

**Hypothesis 4: Methods of autonomously establishing connections between agents based on the similarity of local models and data positively impact the accuracy of local models on realistic data compared to randomly assigned connections.**

Drawing from the preceding exposition of the thesis hypotheses and the published studies, the principal scientific contributions of this thesis include:

- C1** A method for personalizing peer-to-peer agents' models based on normalization layers.
- C2** A method for establishing peer connections and exchanging model weights among agents within a multi-task learning environment.
- C3** A method for training shared model layers to achieve faster convergence and better accuracy in a multi-task learning environment.
- C4** A framework for evaluation, comparison, and selection of optimal methods for autonomous and personalized peer connection creation, in synthetic and realistic non-IID environments.

## 2. ELABORATION

The doctoral thesis is structured in the format of a compilation thesis, comprising three scientific articles published in indexed journals. This section provides a thorough overview of the research questions, methodology, results, and conclusions presented in these articles to establish the coherence of the thesis. Each paper addresses specific research questions within the broader context of mitigating the adverse effects of non-IID data in peer-to-peer systems, primarily focusing on personalized learning processes for individual agents.

### 2.1. An overview of conducted studies

The first study, *Peer-to-peer deep learning with non-IID data* [9], aimed to develop a more suitable peer-to-peer learning method tailored for non-IID environments, particularly addressing challenges related to asynchrony and inactive agents. Traditional peer-to-peer methods typically involve synchronous model exchanges among agents, which may not be ideal in scenarios where some agents are inactive or do not participate in model exchange. To address this limitation, a novel approach was introduced, wherein agents are allowed to receive a variable number of messages upon completing the training step, just before transitioning to the subsequent training iteration. Through computer simulations, it was demonstrated that this new method outperformed both synchronous and asynchronous baseline approaches, particularly in sparse network topologies. Moreover, the new method required fewer exchanged messages per agent while achieving higher average local model accuracy. Additionally, the first study introduced a personalization technique leveraging Batch Normalization layers within the model architecture. By fine-tuning Batch Normalization layers, agents could personalize their models to better fit their local data, consequently improving the local model accuracy. Experimental results illustrated a significant relative increase in local accuracy, especially in experiments involving datasets exhibiting higher degrees of non-IID properties.

The second study, *Multi-task peer-to-peer learning using an encoder-only transformer model* [72], aimed to explore the feasibility of promoting collaboration among agents operating in an extreme non-IID environment, where each group of agents is assigned a distinct learning task

within the same data modality. Specifically, the study focused on two Natural Language Processing (NLP) tasks: masked-token prediction (MPT) and named-entity recognition (NER). Leveraging the encoder-only transformer architecture employed in the BERT model [77], the study utilized the encoder as a shared model base for both tasks. The output produced by the attention layers within the encoder was directed to task-specific classifiers. Notably, while the task-specific components of the models were shared only among agents learning the same task, the encoder segment could be exchanged among all agents. This approach harnessed the encoder’s capacity to retain shared knowledge across all agents, irrespective of their assigned tasks. The experiments conducted in the second study demonstrated a degradation in the accuracy of local models when employing a randomly generated sparse topology for agent connections, as opposed to the results obtained when agents exclusively exchanged models with those learning the same task. The reason for this degradation was identified as some agents were connected only to those learning different task, preventing certain agents from ever exchanging task-specific layers relevant to their own task. Hence, a method for establishing connections between agents in a multi-task setting was introduced to mitigate agent isolation. The experimental findings demonstrated that collaboration among agents learning different tasks can enhance local accuracy when the connections between agents are thoughtfully established.

The third study, *An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning* [32], aims to investigate whether the autonomous establishment of peer connections between agents positively influences the performance of local models on realistic data compared to randomly assigned connections. The study examines recently proposed methods for autonomous connection establishment in non-IID environments. Through a series of computer simulations, it aims to identify the optimal method and approach in various synthetic and realistic non-IID environments. The study revealed that since all methods lack restrictions on the number of connections per agent, certain approaches exhibit high centralization tendencies. This means that a limited set of agents end up sending messages to most of the agents in the network, leading to a high imbalance in communication load. The Pareto method was employed to identify the optimal methods on the Pareto front, considering the achieved top average accuracy and the average number of messages per agent required to reach the top accuracy. Among the methods evaluated, PANMLoss [27] and PANMGrad [27] emerged as optimal solutions in both synthetic and realistic non-IID environments. Additionally, DAC [24] and DiPLe [25] were identified as optimal solutions specifically in the realistic non-IID envi-

ronment. These findings confirm the beneficial impact on agents' model accuracy in non-IID environments when connections between agents are autonomously established compared to randomly assigned connections.

Together, these studies collectively underscore the considerable potential of personalization techniques in non-IID environments to enhance the performance of agents' local model.

### 2.1.1. Peer-to-peer learning method and personalization technique for non-IID data environments

The first study devises a novel peer-to-peer learning approach wherein an agent could receive a variable number of messages between training steps. In the same study, the impact of incorporating Batch Normalization [56] layers into the model architecture on the learning process within a non-IID environment was examined, comparing it with both the new proposed method and existing baseline methods. Furthermore, a personalization technique that leverages Batch Normalization layers was introduced. See Appendix A for the entire article and Appendix D for supplemental materials containing the results and reproducible Python code.

#### 2.1.1.1. Introduction

When examining the communication dynamics among agents within a communication model, both communication symmetry and synchronicity are key considerations. Synchronous approaches [23, 29, 30] entail a synchronization barrier that must be overcome once all messages between agents are exchanged, rendering the direction of communication less significant. Undirected asynchronous communication [13–16, 20, 34, 35], on the other hand, indicates that there is no synchronization barrier restricting the agent training process; however, communication between two agents remains synchronous. In a case when two agents exchange model parameters between themselves, both agents must send and receive model parameters before continuing to the training step, or communicating with other peers. Among these, asynchronous directed communication [34, 36, 37] is the most permissive, as the sending agent does not await a response. The receiving agent integrates the received model parameters with its local model and can continue learning according to its designated learning behavior.

In realistic environments, opting for a sparse communication topology is typically the simplest strategy. For instance, arranging agents into a ring topology [19–21, 78] presents more challenges, as it necessitates adherence to specific connection rules, which can be difficult when agents lack a comprehensive overview of the entire network. While asynchronous directed communication methods like *gossip averaging* [34, 36, 37] are designed with sparse topologies in mind, communication is structured such that an agent sends its message to a random peer in the

network. This assumes that each agent possesses knowledge of the entire network, which may be impractical in real-world scenarios. More practically feasible solutions involve asynchronous communication with a limited set of peers, as seen in approaches like [13–16, 31], which mimic Blockchain applications [79–81] already deployed in production. Sparse peer-to-peer networks formed in these approaches were primarily proposed for training linear models. To address the mentioned gaps in research, the study aims to propose a new asynchronous method with directed communication that follows a given network topology by building upon the GoSGD [36] method.

Batch Normalization (BN) [56] is a well-established technique known to enhance the convergence and precision of machine learning models. Widely used in training deep learning neural network (DNN) models [63–66], BN has been shown to improve model performance in the Federated Learning (FL) setting. However, utilizing BN in a vanilla FL setup does not necessarily lead to model convergence, prompting the development of alternative approaches to address this issue [57, 58, 67, 68]. Despite its effectiveness in FL, the application of BN in a peer-to-peer (P2P) learning setting remains relatively understudied. This study explores the effectiveness and utility of incorporating BN layers as part of the model architecture in a peer-to-peer learning setting.

Model personalization has been shown to significantly enhance the accuracy of agents’ models in FL by conducting additional training, or fine-tuning, of the agent’s model after the completion of FL training [42, 43]. This approach is particularly advantageous for agents whose data distributions diverge significantly from the global distribution in FL scenarios [42]. The study introduces a model personalization technique that leverages BN layers to conduct model personalization during the peer-to-peer learning process, rather than at its conclusion. This method notably enhances the accuracy of agents’ local models.

Experiments were conducted to evaluate all addressed problems and proposed methods using two datasets for the next-word prediction task [82]. The datasets used were the Reddit [62] dataset and the StackOverflow [69] dataset, which consist of user comments. These datasets represent two characteristic domains: Reddit, a more general discussion forum, and StackOverflow, which has a narrow focus on software engineering. In the experiments, each dataset was partitioned such that each agent’s dataset consisted of all comments from a unique user. It’s important to note that each comment may comprise multiple sentences. In line with previous experiments [42, 82–85], the vocabulary size was fixed at the 10,000 most common words,

resulting in 10,000 classes for the model output. Any words beyond this set are treated as out-of-vocabulary tokens and are disregarded when computing prediction accuracy. Sentences are segmented into sequences of 10 words, and the objective of the model is to predict the subsequent word, thereby framing the task as a classification problem.

This study also aimed to quantify the degree of non-IID properties within the Reddit and StackOverflow datasets. Utilizing the Jensen-Shannon divergence (JSD) metric, which measures the similarity between two probability discrete distributions, the objective was to assess the *label distribution skew* (see Section 1.2) among all agents’ datasets. A JSD value of 0 implies the absence of label distribution skew, while a value closer to 1 indicates higher label distribution skew. The obtained JSD values of 0.6633 for the Reddit dataset and 0.5687 for the StackOverflow dataset indicate relatively high label distribution skew among agents. The *quantity skew* data heterogeneity is evident in both the Reddit and StackOverflow datasets. On average, each Reddit agent trains on 654 examples, with approximately 80% of agents having fewer than 700 examples. Similarly, each StackOverflow agent has an average of 1,134 training examples, with around 74% of agents having fewer than the average number of training examples. Considering that the next-word prediction task involves 10,000 unique classes, each agent typically has representation from only a small fraction of these classes in their training split, with a maximum of approximately 10% of the classes represented in the best-case scenario.

The Average User Model Accuracy (UA) metric [58] was utilized to evaluate the overall performance of the learning process. UA represents the average accuracy across all agents on their respective local test datasets and can be computed as follows:

$$UA = \frac{1}{n} \sum_{i=1}^n acc_i \quad (5)$$

where  $acc_i$  represents the prediction accuracy of model  $i$  on its local test dataset  $i$  (both owned by agent  $i$ ). Prediction accuracy ( $acc_i$ ) is determined by the fraction of correct predictions (True Positives (TP) and True Negatives (TN)) over the total predictions (TP, TN, False Positives (FP), and False Negatives (FN)):

$$acc_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}. \quad (6)$$

### 2.1.1.2. Peer-to-peer learning method

The proposed P2P-BN method is inspired by the GoSGD approach [36] and builds upon [34, 86], where the received model parameter updates are averaged upon receipt, rather than waiting to average all received models once all are received [18–23, 29, 30, 78]. In P2P-BN, an agent communicates based on any predetermined topology, with communication occurring in either an directed or undirected manner. During the communication step, instead of sending a model update to a random agent in the network as in GoSGD, in P2P-BN, an agent maintains a set of peers to which it is connected and with which it communicates, as inspired by other peer-to-peer methods [13–16, 31].

The P2P-BN method allows an agent to train the local model for a predetermined but arbitrary number of local iterations through its local dataset before proceeding to the communication step. During the communication step, an agent sends its model to all of its peers and then waits for a period of time to receive updates from its peers. In a non-IID environment, continuing local training without receiving any model updates may lead to overfitting and sub-optimal model performance due to low representation in data examples and quantity deficiency in the local dataset. Therefore, an agent may receive one or more updates from its neighboring peers before continuing to the training step. This training-communication cycle is repeated for a predetermined number of rounds, as with all learning methods. Figure 12 showcases the steps performed by three agent in a unbalanced undirected communication topology in which agent  $A_2$  is neighboring with agents  $A_1$  and  $A_3$ , while the agents  $A_1$  and  $A_3$  are only neighboring with agent  $A_2$ . Despite this unbalanced communication, the learning process in the P2P-BN method is not stalled because each agent only requires the reception of at least a single model update from its peers.

The P2P-BN method was compared to several baseline methods, including  $D^2$  [21], SGP [87], and GoSGD [36]. Since GoSGD typically sends messages to random agents in the network, its communication behavior was adjusted in the experiments so that agents select their neighbors based on the specific experiment’s network topology. All experiments utilized a single model architecture, which consisted of an Embedding layer with a dense embeddings size of 100, followed by a GRU layer [88] with 100 units, a Fully Connected layer with 100 units, and finally an output layer with 10,002 units. This version of the model was termed as non-BN



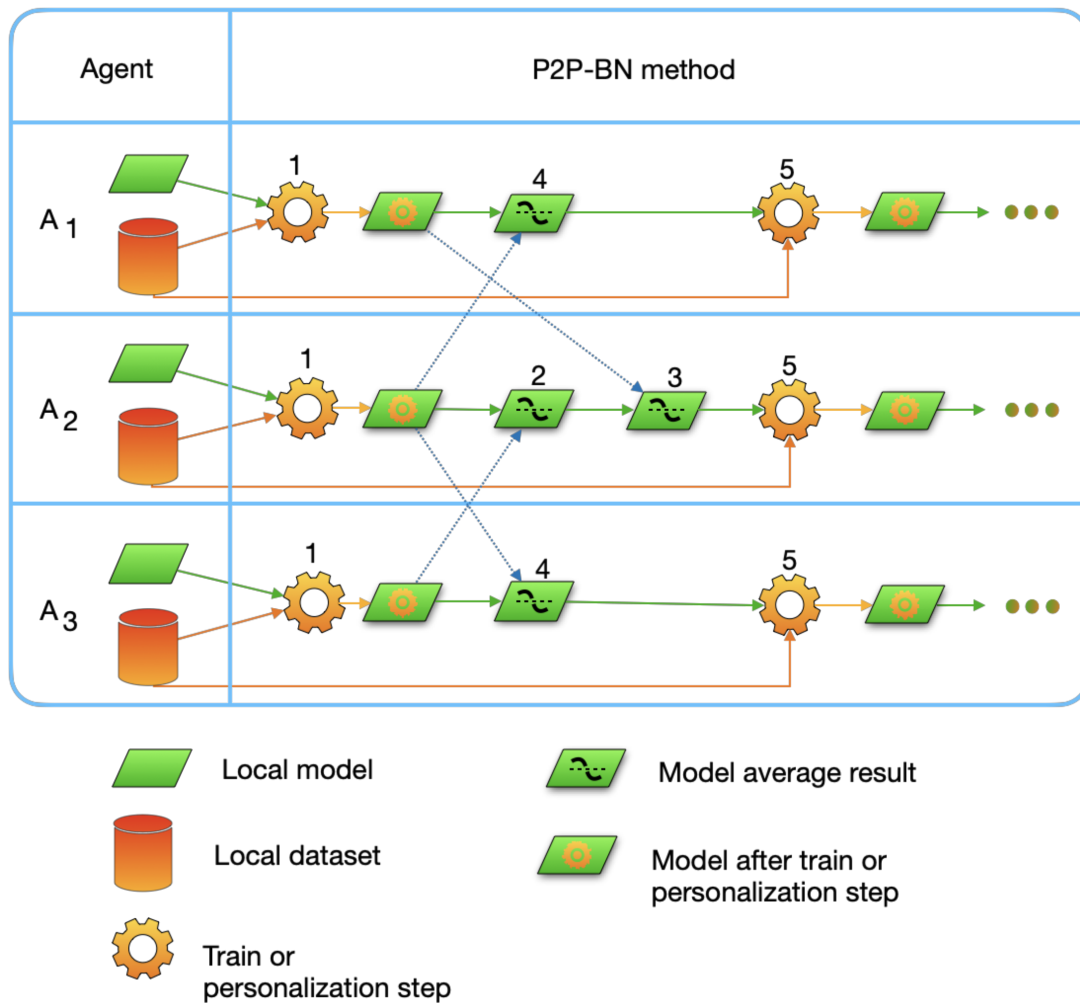


Figure 12: The learning and communication process of three P2P-BN agents in an unbalanced topology [9].

model, since it did not contain any BN layers.

The initial experiment was conducted using a ring topology with both undirected and directed communication, involving 100 agents. Table 1 presents the average top UA achieved by agents in each approach and the average number of exchanged messages per agent required to reach that accuracy. In terms of exchanged messages, the P2P-BN consistently required the fewest, although it only outperformed other methods in undirected communication for the Reddit dataset. GoSGD performed best in undirected communication for the StackOverflow dataset, while the SGP method was optimal in directed communication for both datasets.

Table 1: The average top UA and the average number of exchanged messages per agent for both the Reddit and StackOverflow datasets in ring topology using the non-BN model. Best results are emphasized.

| Dataset       | Approach       | Undirected    |              | Directed      |              |
|---------------|----------------|---------------|--------------|---------------|--------------|
|               |                | # of messages | Test UA      | # of messages | Test UA      |
| Reddit        |                |               |              |               |              |
|               | D <sup>2</sup> | 275730        | 4.86%        | 142410        | 6.19%        |
|               | GoSGD          | 18662         | 4.71%        | 13288         | 4.84%        |
|               | SGP            | 257720        | 4.64%        | 144020        | <b>6.29%</b> |
|               | P2P-BN (ours)  | <b>16318</b>  | <b>5.55%</b> | <b>7029</b>   | 5.28%        |
| StackOverflow |                |               |              |               |              |
|               | D <sup>2</sup> | 321300        | 7.44%        | 227409        | 9.59%        |
|               | GoSGD          | 36648         | <b>7.49%</b> | 20916         | 7.39%        |
|               | SGP            | 310464        | 7.21%        | 222500        | <b>9.61%</b> |
|               | P2P-BN (ours)  | <b>17336</b>  | 7.19%        | <b>7029</b>   | 6.66%        |

### 2.1.1.3. Personalization method

Since similar studies [42, 43] in Federated Learning have shown that additional training iterations on the local model can improve local accuracy in non-IID environments, and Batch Normalization layers have been shown to provide positive benefits in FL as a form of personalization [57, 58, 67, 68], BN layers were adopted as a basis for personalization in P2P-BN. Given the assumption of non-IID data, agents may differ in the quantity of local data they possess. After a certain number of learning rounds (train-communication steps), agents with limited data are likely to degrade their model after the local training step due to the low representation in training examples. If an agent cannot improve the local model, it freezes all non-BN layers. The proposed personalization technique, referred to as the BN fine-tuning method, involves freezing the non-BN layers and training only the BN layers. This improvement is based on the

previously demonstrated remarkable representational capacity of BN layers [71]. Similar to the early stopping technique [89], a validation dataset is used to determine when the layers need to be frozen or unfrozen.

The ablation study aimed to assess the impact of BN fine-tuning on local accuracy. In the experiments, a model with BN layers added after the GRU and Fully Connected layers, termed the BN model, was utilized. The experiments involved 100 and 300 agents connected by a directed or undirected sparse graph with three neighbors. Training with the BN fine-tuning (BN-TF) technique resulted in a relative top UA increase of approximately 20% for the Reddit dataset and around 15% for the StackOverflow dataset. Statistical significance was confirmed by the Student t-test, with a p-value of  $p < 2 \times 10^{-2}$  for both the 100 and 300 agent groups.

#### **2.1.1.4. Impact of Batch Normalization on peer-to-peer learning in non-IID environments**

The impact of Batch Normalization layers was examined by comparing a model without BN layers to a model with BN layers. The experiments were conducted in the ring topology, using both undirected and directed communication. Table 2 provides a summary of the average top UA and the average number of exchanged messages per agent required to achieve that accuracy. In the BN model experiments, compared to the closest second-ranked GoSGD, P2P-BN achieved a mean relative top UA increase of 19.9% for the Reddit dataset and 13.9% for the StackOverflow dataset. For all approaches, the use of BN layers enabled faster convergence and higher accuracy. The exceptions were the  $D^2$  and SGP methods, which obtained slightly lower results in directed communication. However, they still benefited from faster convergence, thereby requiring a lower number of exchanged messages.

The performance of P2P-BN was further evaluated and compared to baseline approaches on both undirected and directed, fixed and time-varying sparse network topologies. When creating a sparse communication matrix, each agent was assigned three neighbors. In the time-varying network scenario, the communication matrix was altered every five rounds, causing each agent to change its peers every five rounds. The time-varying scenario aimed to investigate the impact of highly dynamic environments on the learning process. Table 3 summarizes the top UA achieved for each variation of the experiment.  $D^2$  and SGP achieved higher top UA in undirected

Table 2: The average top UA and the average number of exchanged messages per agent for both the Reddit and StackOverflow datasets in ring topology using the BN model. Best results are emphasized.

| Dataset       | Approach             | Undirected    |              | Directed      |               |
|---------------|----------------------|---------------|--------------|---------------|---------------|
|               |                      | # of messages | Test UA      | # of messages | Test UA       |
| Reddit        |                      |               |              |               |               |
|               | D <sup>2</sup>       | 84840         | 6.08%        | 57570         | 6.85%         |
|               | GoSGD                | 23779         | 6.43%        | 10419         | 6.71%         |
|               | SGP                  | 139472        | 6.08%        | 62178         | 6.74%         |
|               | <b>P2P-BN (ours)</b> | <b>11144</b>  | <b>7.18%</b> | <b>6633</b>   | <b>8.54%</b>  |
| StackOverflow |                      |               |              |               |               |
|               | D <sup>2</sup>       | 254847        | 7.92%        | 189924        | 8.76%         |
|               | GoSGD                | 46906         | 8.5%         | 20667         | 8.96%         |
|               | SGP                  | 159968        | 7.87%        | 147500        | 8.71%         |
|               | <b>P2P-BN (ours)</b> | <b>6895</b>   | <b>8.69%</b> | <b>7920</b>   | <b>10.69%</b> |

communication, while GoSGD and P2P-BN performed better in directed communication. Similarly, in the varying undirected topology, D<sup>2</sup> and SGP achieved better results compared to the fixed topology experiments, whereas in the varying directed topology, GoSGD and P2P-BN achieved higher top UA compared to the fixed topology experiments. In the sparse topology experiments, P2P-BN exhibited a mean relative top UA increase of 32.9% for the Reddit dataset and 26.6% for the StackOverflow dataset compared to the closest second-ranked GoSGD.

Table 3: Average top UA results of 100 agents in fixed undirected and directed, and varying undirected and directed sparse topology experiments conducted on the Reddit and StackOverflow datasets. Best results are emphasized.

| Dataset       | Approach             | Fixed         |               | Varying       |               |
|---------------|----------------------|---------------|---------------|---------------|---------------|
|               |                      | Undirected    | Directed      | Undirected    | Directed      |
| Reddit        |                      |               |               |               |               |
|               | D <sup>2</sup>       | 6.91%         | 6.84%         | 7.26%         | 6.84%         |
|               | GoSGD                | 6.95%         | 7.63%         | 7.64%         | 7.66%         |
|               | SGP                  | 6.94%         | 6.73%         | 7.16%         | 6.73%         |
|               | <b>P2P-BN (ours)</b> | <b>9.40%</b>  | <b>10.17%</b> | <b>9.89%</b>  | <b>10.26%</b> |
| StackOverflow |                      |               |               |               |               |
|               | D <sup>2</sup>       | 8.99%         | 8.85%         | 9.37%         | 8.77%         |
|               | GoSGD                | 8.91%         | 9.65%         | 9.63%         | 9.65%         |
|               | SGP                  | 8.92%         | 8.62%         | 9.35%         | 8.62%         |
|               | <b>P2P-BN (ours)</b> | <b>11.49%</b> | <b>12.61%</b> | <b>12.13%</b> | <b>12.27%</b> |

### 2.1.1.5. Discussion

The P2P-BN method developed in this study falls within the gossip averaging family of peer-to-peer learning methods [11, 34, 36, 86]. P2P-BN is asynchronous, meaning it does not

require agents to synchronize once the synchronization barrier is met. After an agent completes a training step, it forwards the new model parameters to its neighbors. Subsequently, an agent may receive one or more model parameter updates from its peers before continuing to the next training step. This flexibility in the number of messages an agent can receive allows it to maintain participation in the learning process even in cases of lost messages or disconnection from some of its peers. Methods like  $D^2$  and SGP necessitate that each agent receives messages from all its neighbors before proceeding to the next training step. If a message is lost, an agent would stall at the communication step and would be unable to proceed to the training step. This lack of resilience to lost messages can impede the learning process, especially in scenarios where network communication is prone to errors or disruptions.

The P2P-BN method incorporates a personalization technique tailored for non-IID environments, allowing agents to enhance the accuracy of their local models. This technique involves training only the Batch Normalization layers of the model when it's determined that the agent cannot improve the overall model accuracy. A validation dataset is employed to decide whether to perform full model training or solely train the BN layers. Results from an ablation study showcased an average relative increase of approximately 17% in agents' local accuracy when employing BN fine-tuning personalization technique. With the BN fine-tuning personalization technique integrated, P2P-BN demonstrated significant improvements in the next-word prediction task. Simulations revealed that, on average, P2P-BN achieved a mean relative top accuracy increase of 16.9% in ring topologies (19.9% for Reddit and 13.9% for StackOverflow datasets) and 29.8% in sparse communication topologies (32.9% for Reddit and 26.6% for StackOverflow datasets) compared to the best-performing baseline approach.

There are several limitations to consider in this study. Firstly, the evaluation was confined to a next-word prediction task, indicating the need for further investigations across diverse non-IID datasets employing various data modalities to ascertain the generalizability of P2P-BN. Additionally, the reliance on Batch Normalization layers imposes constraints on the model architecture utilized by the agents, as the model must contain BN layers in order for the BN fine-tuning technique to work. The performance of P2P-BN may vary in scenarios where agents exhibit disparities in hardware capabilities, necessitating further examination to address potential imbalances. The variability in training speed among agents can introduce challenges in maintaining synchronization and ensuring effective communication in the peer-to-peer learning process. Agents with faster training speeds may complete their training steps quickly and

communicate with their peers sooner than slower agents. This could lead to imbalances in the communication frequency and timing, potentially affecting the overall convergence and performance of the learning process. While the asynchronous nature of P2P-BN may mitigate some of these issues, comprehensive studies are essential to fully understand its effectiveness various scenarios.

This study addresses the contribution **C1**. Contribution **C1** focuses on introducing a novel personalization technique that leverages Batch Normalization layers for agent model personalization. The experiments conducted in the study validate the positive impact of this technique on the accuracy of agents' local models. By fine-tuning Batch Normalization layers, agents can adapt their models to better suit their individual datasets, leading to improved model accuracy and performance.

### **2.1.2. Multi-task peer-to-peer learning method**

The second study introduced a novel P2P learning method designed to foster collaboration among agents tasked with different NLP objectives. The study focused on agents engaged in two specific NLP tasks, MPT and NER. In addition, the study also introduces a technique for constructing a network topology that connects agents based on their task objective. Furthermore, the study introduced a method for strategically training shared model layers, optimizing the learning process and further boosting local model accuracy. See Appendix B for the entire article and Appendix D for supplemental materials containing the results and reproducible Python code.

#### **2.1.2.1. Introduction**

Under high non-IID constraints, agents may essentially be learning different tasks due to variations in features and learning objectives. One way to address this scenario is through multi-task learning, which involves dividing the model into two parts: a shared part and a task-specific part. The shared part, shared among all agents, captures general knowledge and understanding of the underlying data. On the other hand, the task-specific part is responsible for predicting target values based on the outputs of the shared model part. This approach allows agents to collaborate effectively while accommodating their diverse learning objectives and data distributions.

In FL, multi-task learning is commonly implemented by keeping the task-specific or personalized parts locally on the agents, while the shared part is exchanged with the server to form a new global model, comprised only of the shared part. This approach has been explored in various studies [50, 57, 58, 60, 61]. Experimental results have demonstrated that the base layers contain valuable knowledge, as evidenced by the drop in accuracy when the model was tested solely with the personalized part [50]. Another approach involves dividing agents into groups based on the similarity of their models [48]. In this setup, only the models within a certain group of agents are aggregated to form a new group-global model. This approach essentially creates several disjoint groups of agents learning different tasks.

Taylor *et al.* [73] applied multi-task learning as a form of personalization technique for pre-

dicting future mood, stress, and health outcomes. Their approach involved dividing the model into two components: a shared part and multiple task-specific parts. User data was clustered based on personality and gender. The shared part of the model extracted features relevant to all clusters, while the task-specific parts generated predictions tailored to each cluster. Multi-task learning outperformed the single-task baseline, improving accuracy by approximately 12%.

Pilet *et al.* [35] extended the concept further by suggesting that any part of the neural network model could be shared with any peer or groups of peers. Through their experiments, they determined that allocating 80% of the model parameters as shared was optimal for multi-task learning. This allocation led to a relative increase in accuracy by 9% compared to scenarios where 100% of the model parameters were shared (essentially mono-task learning). However, the authors underscored that the number of performed mini-batch updates had a more significant impact than the size of the shared part. These experiments were conducted on the FEMNIST [62] dataset of handwritten characters, which was grouped by the author of the handwritten characters.

Mohammadi *et al.* [46] adopted a similar approach to P2P learning as in FL studies, where each agent maintains a residual model specific to its data while sharing general knowledge model with neighboring agents. Multi-task learning can also be implemented in P2P environments by strategically managing agent connections to cluster agents with similar tasks, akin to the FedAMP method [48]. Zantedeschi *et al.* [15] introduced a technique for dynamically forming P2P connections by leveraging the similarity among agents' local linear models based on empirical loss. Subsequent studies have extended this approach to neural networks, enhancing model performance in scenarios with diverse synthetic non-IID environments [24–28]. These studies primarily focus on the image classification task, by applying image transformations such as varying rotations, label semantics, and data distributions to simulate a synthetic non-IID environment. By leveraging techniques like dynamic formation of P2P connections based on local model similarities and empirical loss on local datasets, these methods have shown promise in improving model accuracy and convergence in non-IID settings.

This study aims to facilitate multi-task collaboration between two separate groups of agents, with each group focusing on a different NLP task. The objective is for individual agents to improve their learning capability by engaging in collaboration with agents tackling the same task as well as those addressing a different task. To evaluate this method, two NLP tasks will be employed: MTP and NER. The study adopts the encoder-only transformer architecture from



the BERT [77] model as the base model part for both the MTP and NER tasks. The output produced by the attention layers in the encoder is directed to a task-specific classifier, either for MTP or NER. In essence, each agent possesses a single encoder model and a final task-specific output layer, enabling focused learning on the designated task. Leveraging the encoder’s capability to retain shared knowledge across all agents, irrespective of their assigned clusters, the research ensures that the task-specific components of the models are confined within their respective clusters. The study adopts the BERT-Tiny [90] architecture, featuring two attention heads, two transformer blocks in the hidden layers, and a hidden size of 128 units. This configuration results in a compact model comprising eight million parameters. The choice of this architecture is justified by the constrained resources of agents, such as mobile or IoT devices, which have limited memory and computational capabilities. Moreover, sharing larger models over the network would incur additional communication costs, and the scarcity of data available to agents reduces the necessity for larger models. The primary focus of the study lies in facilitating multi-task learning across diverse tasks within decentralized networks rather than achieving state-of-the-art accuracy on any specific NLP task. Figure 13 shows the architecture of the models used for the MTP and NER tasks.

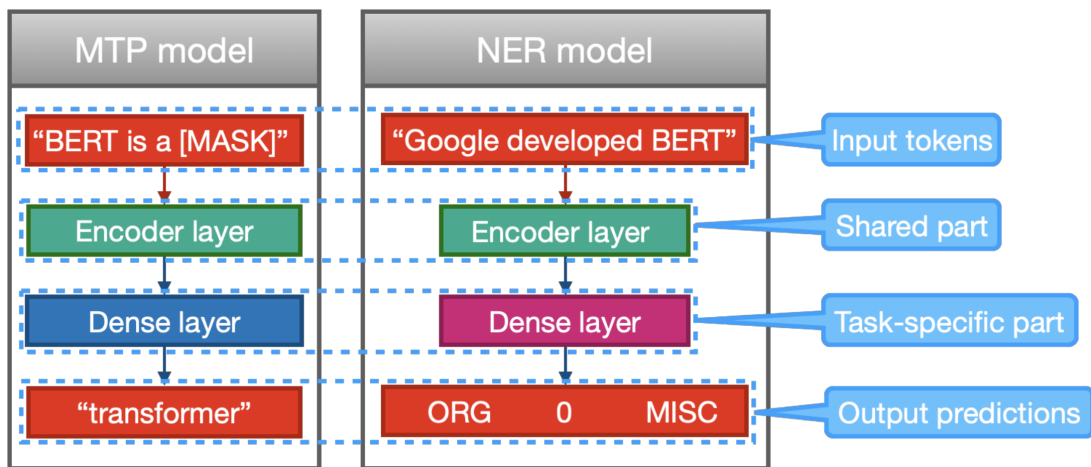


Figure 13: Architecture of MTP and NER models [72].

Using the Reddit [62] and StackOverflow [69] datasets for the MTP task, and the CoNLL-2003 [91] and Few-NERD [92] datasets for the NER task, the experiments incorporate diverse data sources for both tasks. In the experiments, each agent is assigned a dataset from a unique user in the Reddit and StackOverflow datasets, as these datasets are collected at the user level. For the NER datasets, data is partitioned uniformly among agents. Given that the Few-NERD dataset is considerably larger than the CoNLL-2003 dataset, only 20% of the Few-

NERD dataset is utilized in the experiments. On average, each agent has approximately a thousand training examples, regardless of the dataset or task. For the MTP task, a *[MASK]* token serves as a placeholder for the word being predicted, transforming the objective into a next-word prediction (NWP) task [84, 85]. Solely predicting the final word of a sentence might not fully leverage the contextual information encoded in the BERT-based model, as it restricts the model’s ability to consider the entire sentence context. This limitation could diminish the advantage of BERT’s contextual understanding. However, this characteristic also introduces additional task heterogeneity, which can further introduce complexity when establishing the similarities between the MTP and NER tasks.

The study employs metrics measured at the agent level using each agent’s local data. The Average User Model Accuracy (UA) [58] metric, is utilized to evaluate the performance of the overall MTP learning process. Additionally, the F1 score metric is employed to assess the performance of the NER task. F1 metric is defined as the unweighted average (macro f1-score) of precision and recall, given as follows for agent  $i$ :

$$precision_i = \frac{TP_i}{TP_i + FP_i}, \quad (7)$$

$$recall_i = \frac{TP_i}{TP_i + FN_i}, \quad (8)$$

$$F1-score_i = \frac{2 \times recall_i \times precision_i}{recall_i + precision_i}. \quad (9)$$

### 2.1.2.2. Multi-task learning method

The proposed technique builds upon the P2P-BN [9] approach as the P2P learning method. The personalization technique, BN fine-tuning, relying on Batch Normalization layers is omitted from the proposed method since the model architecture does not contain Batch Normalization layers. Each agent is tasked with solving either the MTP or NER task, denoted as  $T_i$ , where  $type(T_i) \in \{MTP, NER\}$ . Every model  $m_i$  consists of an encoder-only transformer, referred to as the encoder layer, along with additional task-specific layers. The encoder layer is shared among all agents, while the task-specific layers are shared only among agents within the same

task  $T$ . It’s important to note that each agent is trained to solve only one task  $T_i$  but leverages multi-task collaboration to enhance its local model accuracy. During the model update exchange process between agents  $i$  and  $j$ , if  $type(T_i) = type(T_j)$ , a standard model averaging operation is performed:  $m_i = \frac{m_i + m_j}{2}$ . However, if  $type(T_i) \neq type(T_j)$ , only the encoder layer of the model is averaged:  $m_i^{(encoder)} = \frac{m_i^{(encoder)} + m_j^{(encoder)}}{2}$ , where  $m_i^{(encoder)}$  and  $m_j^{(encoder)}$  represent only the encoder layer of models  $m_i$  and  $m_j$ , respectively.

In the initial experiments, a directed sparse topology is employed, with each agent having three in-neighbors and three out-neighbors. As a baseline, individual clustered task-specific (CTS) learning is utilized, where agent groups are trained separately for each task and dataset. For instance, agents learning the MTP task exclusively communicate with other agents assigned to the same task. In all experiments, a group consisting of twenty agents is trained for each task (dataset). The topology is predefined and established at the beginning of the training process based on the specific learning tasks assigned to the agents. This setup allows for a direct comparison between the proposed multi-task learning approach and the baseline CTS approach. The experiments were designed such that two groups of agents, with each group learning a different task on a specific dataset, collaborated using the proposed multi-task method. The results presented in Table 4 indicate that the collaboration between agents learning different tasks may not be advantageous. Specifically, only the agents learning the NER task using the CoNLL-2003 dataset experienced a slight increase in accuracy when exchanging information with agents learning the MTP task, for both the Reddit and StackOverflow datasets.

Table 4: Average top Test UA achieved by agents in a sparse topology for each of the multi-task combinations [72].

| Collaboration | Dataset (Task) | Reddit | StackOverflow | CoNLL-2003    | Few-NERD |
|---------------|----------------|--------|---------------|---------------|----------|
| CTS baseline  |                | 9.10%  | 9.02%         | 53.13%        | 31.69%   |
| MT            |                |        |               |               |          |
|               | Reddit         | -      | 8.85%         | <b>53.63%</b> | 31.14%   |
|               | StackOverflow  | 8.43%  | -             | 53.03%        | 30.51%   |
|               | CoNLL-2003     | 8.43%  | 8.35%         | -             | 31.58%   |
|               | Few-NERD       | 8.68%  | 8.81%         | <b>56.17%</b> | -        |

### 2.1.2.3. Topology organization method

The initial experiment revealed that collaboration between agents learning different tasks had a negative impact on the local agents’ accuracy. Upon investigation, it was discovered that

due to the random assignment of peer connections in the sparse topology, some agents only collaborated with agents learning a different task. As a result, these agents did not receive the task-specific parts of the model from any peer, leading to less robust task-specific components and decreased local accuracy compared to the CTS baseline.

To address this issue, a solution was devised where an agent communicates with a predefined portion of agents sharing the same task and a predefined portion of agents learning a different task. This approach limits the number of connected peers from another task ( $|\{W_{ij} > 0 \text{ and } T_i \neq T_j\}|$ ) to a specific fraction of the total number of connected peers ( $|\{W_{ij} > 0\}|$ ) to achieve better results. This restriction is denoted by the parameter  $PT$  as a fraction of connected peers from a different task to the total number of connected peers:

$$PT = \frac{|\{W_{ij} > 0 \wedge T_i \neq T_j\}|}{|\{W_{ij} > 0\}|} \quad (10)$$

In the experiments with limited non-task-specific peer connections, the parameter  $PT$  is set to 0.33, defining a *clustered sparse* topology. Consequently, in the *clustered sparse* topology, each agent establishes two peer connections with agents within the same task and one with an agent from a different task. Figure 14 shows an example of task-specific clusters, sparse, and *clustered sparse*.

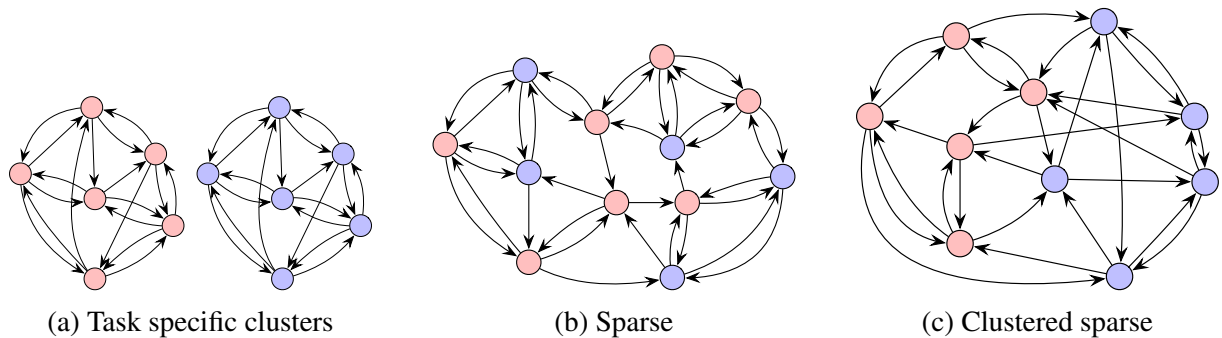


Figure 14: An example of task-specific clusters (a), sparse (b), and *clustered sparse* (c) topology [72]. Agent color represents the learning task.

The results depicted in Table 5 demonstrate a notable improvement in agents' local accuracy across most experiments conducted in the *clustered sparse* topology. However, there were instances of degraded performance, particularly in the experiments involving the NER task with the Few-NERD dataset. Furthermore, agents using the Reddit dataset exhibited lower accuracy when collaborating with agents using the CoNLL-2003 dataset.

Table 5: Average top Test UA achieved by agents in a *clustered sparse* topology for each of the multi-task combinations [72].

| Collaboration | Dataset (Task) | Reddit       | StackOverflow | CoNLL-2003    | Few-NERD |
|---------------|----------------|--------------|---------------|---------------|----------|
| CTS baseline  |                | 9.10%        | 9.02%         | 53.13%        | 31.69%   |
| MT            |                |              |               |               |          |
|               | Reddit         | -            | <b>9.90%</b>  | <b>53.40%</b> | 30.24%   |
|               | StackOverflow  | <b>9.46%</b> | -             | <b>53.43%</b> | 30.98%   |
|               | CoNLL-2003     | 8.86%        | <b>9.10%</b>  | -             | 31.61%   |
|               | Few-NERD       | <b>9.20%</b> | <b>9.22%</b>  | <b>56.54%</b> | -        |

#### 2.1.2.4. Shared model layers training method

Due to the suboptimal results achieved in the case of the Few-NERD dataset, an additional technique is introduced to improve the convergence and consensus of the shared model part. The additional technique introduced aims to enhance convergence and consensus of the shared model part, particularly in scenarios with mixed-task neighborhoods. When an agent communicates with peers from a different task, only the encoder layer is averaged, and subsequently, the receiving agent freezes its encoder layer for the next training round. This freezing mechanism ensures that the shared encoder layer, representing common knowledge, converges slowly, allowing agents to update task-specific layers without altering the shared part significantly. Formally, denoting the frozen model as  $\|m_i\|$ , agent  $i$  performs the next local training round on  $m_i$  as follows:  $m_i = m_i - \eta F_i(\|m_i\|; \xi_i), \xi_i \sim D_i$  for  $E$  iterations. This approach, akin to personalization, enables agents to adapt their models while maintaining a consensus on shared knowledge. It’s crucial to highlight that freezing the encoder layer occurs selectively, triggered only when an agent interacts with peers from different tasks, ensuring stability and convergence in heterogeneous environments. The new method is termed as *MT-EF*.

The final multi-task learning method is succinctly summarized by the pseudo-code presented in Algorithm 2, delineating the global agent behavior. Analogous to the P2P-BN paradigm, all agents commence the learning process by initializing their models with uniform parameters, ensuring homogeneity in the encoder layer weights across all agents, irrespective of the task assignment. At each iteration, an agent selected from those that have received updates since the last active state undergoes local training. Following the training step, the chosen agent unfreezes the encoder layer, if previously frozen, before disseminating the model parameters to its peer agents. In instances where a peer agent is engaged in a distinct task  $T_i \neq T_j$ , the peer agent freezes the encoder layer subsequent to the aggregation of the received update.

---

**Algorithm 2** Agents training simulation pseudo-code [72]

---

```
1: function TRAIN( $A, W, E$ )
2:   A: agents
3:   W: communication matrix
4:   E: number of epochs to perform locally
5:   INITIALIZE( $A$ )    ▷ Initialize all agents models and encoder layers to identical model
                        parameters
6:   repeat
7:      $a_i = \text{RANDOMACTIVEAGENT}(A)$     ▷ Random agent which received at least one
                        update from last active state
8:     LOCALTRAIN( $a_i, E$ )
9:     if ISENCODERFROZEN( $a_i$ ) then
10:      UNFREEZEENCODERLAYER( $a_i$ )
11:    for  $a_j$  in NEIGHBORS( $W, a_i$ ) do
12:      if  $T_j \neq T_i$  then
13:        ENCODER( $a_j$ ) = AVERAGEENCODER( $a_j, a_i$ )
14:        FREEZEENCODER( $a_j$ )
15:      else
16:        MODEL( $a_j$ ) = AVERAGEMODEL( $a_j, a_i$ )
17:    until Maximum iteration reached
```

---

Table 6 delineates the outcomes garnered from experiments conducted employing the *clustered sparse* topology alongside the *MT-EF* technique for each of the two multi-task combinations. The results evince a substantial augmentation in the local agents’ accuracy, particularly for MTP agents, with all experiments manifesting a statistically significant average increase of 11.6% in the mean relative accuracy in comparison to the CTS baseline. Moreover, the strategic freezing of the shared model part facilitated swifter consensus attainment among agents, consequently resulting in enhanced accuracy outcomes.

Table 6: Average top Test UA achieved by agents in a *clustered sparse* topology using the *MT-EF* method for each of the multi-task combinations [72].

| Collaboration | Dataset (Task) | Reddit        | StackOverflow | CoNLL-2003    | Few-NERD      |
|---------------|----------------|---------------|---------------|---------------|---------------|
| CTS baseline  |                | 9.10%         | 9.02%         | 53.13%        | 31.69%        |
| MT-EF         |                |               |               |               |               |
|               | Reddit         | -             | <b>11.10%</b> | <b>54.20%</b> | <b>32.23%</b> |
|               | StackOverflow  | <b>10.73%</b> | -             | <b>53.89%</b> | <b>32.92%</b> |
|               | CoNLL          | <b>10.35%</b> | <b>10.43%</b> | -             | <b>32.19%</b> |
|               | Few-NERD       | <b>10.35%</b> | <b>10.60%</b> | <b>56.22%</b> | -             |

As prior experiments have indicated the beneficial nature of *MT-EF* collaboration between two distinct tasks for both clusters of agents, the subsequent experiment assesses the efficacy of training all four tasks within a *clustered sparse* topology using the *MT-EF* approach. Predi-

cated on the parameter  $PT$  (set to 0.33), each agent establishes a solitary peer connection with another agent tasked with a different objective, and two peer connection with agents learning the identical task. Consequently, once all connections are established, all clusters become interconnected through at least one agent, albeit not all agents engage in communication with agents from all tasks (refer to Figure 15 for an illustration). All four distinct groups of agents achieved higher top accuracies when trained within a multi-task framework, indicating that limited collaboration between agents tasked with different objectives invariably yields benefits for all agents. Table 7 encapsulates the attained top UA accuracies, with the average results from previous experiments involving two-task combinations provided for reference (2 tasks (avg)). It is discernible that all agents attained a higher top accuracy when concurrently learning all four tasks compared to the average top accuracy achieved in preceding experiments focusing on two-task combinations, with the exception of the Few-NERD agents, whose results, while slightly lower, remained comparable.

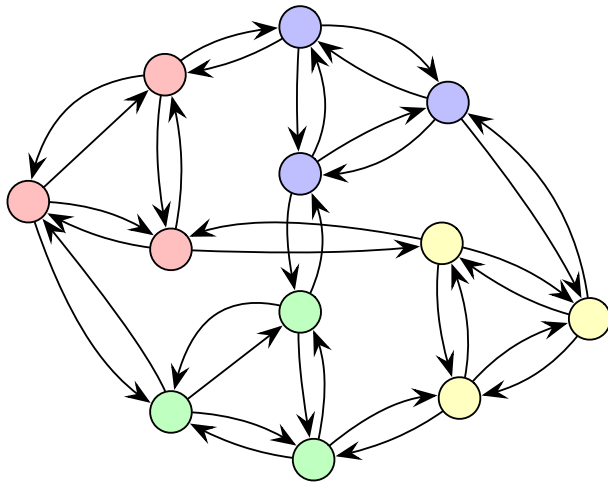


Figure 15: An example of four different agent clusters with two task-specific and one non-task-specific connection randomly formed with another cluster of agents, regardless of the task [72].

Table 7: Average top Test UA achieved by agents in a *clustered sparse* topology using the *MT-EF* method when all four multi-task groups of agents were collaborating simultaneously [72].

| <b>Collaboration</b> | <b>Dataset (Task)</b> | Reddit        | StackOverflow | CoNLL-2003    | Few-NERD      |
|----------------------|-----------------------|---------------|---------------|---------------|---------------|
| CTS baseline         |                       | 9.10%         | 9.02%         | 53.13%        | 31.69%        |
| <i>MT-EF</i>         | 2 tasks (avg)         | 10.48%        | 10.71%        | 54.77%        | <b>32.45%</b> |
| <i>MT-EF</i>         | 4 tasks               | <b>10.55%</b> | <b>10.88%</b> | <b>57.32%</b> | 32.33%        |

### 2.1.2.5. Discussion

The multi-task P2P learning method developed in this study proved to be effective in enhancing the local model accuracy of agents when collaborating with peers engaged in distinctly different tasks within the same data modality. Outperforming the CTS baseline approach, wherein agent solely exchange information within their respective task clusters, proposed method showcased its potential when peer connections were thoughtfully organized. It became evident that in randomly generated sparse topologies, agents might exclusively interact with peers learning different tasks, leading to suboptimal local model performance due to the absence of peers sharing the task-specific model part. Through the introduction of a carefully designed *clustered sparse* topology, where a portion of agent connections was dedicated to cross-task communication, this limitation was addressed. Furthermore, the integration of the *MT-EF* method further augmented the benefits of multi-task collaboration by strategically freezing the encoder part upon receiving updates from agents learning different tasks. This approach facilitated quicker model convergence and enhanced local model performance by fostering consensus on the shared encoder part. Overall, experiments demonstrated an average relative increase in accuracy of approximately 11.6% compared to the CTS baseline, thus validating the efficacy and practicality of the proposed multi-task learning method.

Several limitations are identified in this study. Firstly, the proposed approach requires all tasks to use identical sequence lengths during multi-task learning. While in the domain of causal language modeling, the next-word prediction (NWP) task often employs a 20-word sequence as model input [84, 85], our experiments did not strictly adhere to this practice in the context of the MTP task. Preliminary experiments revealed that using different sequence lengths favored the task with the shorter sequence while significantly disadvantaging the task with a longer sequence. This discrepancy arises because the model, when trained on short sequences, tends to disregard information beyond the short sequence length, often treating it as irrelevant padding. Moreover, the scope of the studied method is currently limited to evaluation solely on NLP tasks. Future research should investigate whether similar setups can be applied to other domains, such as vision processing. Additionally, in NLP tasks, each word must be converted to a numeric representation using a tokenizer. While existing literature typically assumes a globally shared tokenizer, we acknowledge this as a limitation and an avenue for future inves-



tigation. Specifically, research could explore methods for achieving consensus on the tokenizer to be used for the purpose of the agent local data preprocessing. Furthermore, the experiments conducted in this study were confined to two distinct NLP problems, utilizing four different datasets with a set of twenty agents from each dataset. To gain deeper insights into the presented approach, future research should focus on scaling up by increasing the number of agents and tasks. Additionally, the impact of the parameter  $PT$  on the multi-task learning process warrants further investigation through additional experiments.

This study contributes to **C2** and **C3**. Contribution **C2** involves the development of a method for facilitating peer connections and exchanging models among agents within a multi-task learning environment. This method utilizes an encoder-only transformer as a shared component exchanged among all agents, alongside task-specific parts exclusively exchanged among those sharing the same task. Furthermore, the approach integrates a *clustered sparse* topology, ensuring that each agent communicates with only a small percentage of peers learning a different task. Contribution **C3** centers on enhancing model convergence speed and local model accuracy by introducing a form of personalization technique. This technique involves refraining from training the shared encoder-only transformer part under specific conditions, such as when an agent receives a message from a peer learning a different task. The experiments conducted in the study confirm the positive impact of this technique on the accuracy and convergence of agents' local models.

### **2.1.3. Autonomous connection establishment methods**

The third study conducted a comprehensive review, analysis, and evaluation of recently proposed methods for autonomous and personalized peer connection establishment within peer-to-peer learning environments. The evaluation spanned both synthetic and realistic non-IID datasets, aiming to assess and compare the average local model accuracy of agents and the communication load per agent. Additionally, the study delved into the centralization tendencies exhibited by the analyzed methods. Employing the Pareto method, the study identified solutions lying on the Pareto front for individual non-IID scenarios, as well as for the collective results across all diverse non-IID scenarios. See Appendix C for the entire article and Appendix D for supplemental materials containing the results and reproducible Python code.

#### **2.1.3.1. Introduction**

Clusters of agents can exhibit comparable non-IID attributes, emphasizing similarities within each cluster while simultaneously revealing significant differences across distinct groups or individual agents. This scenario was evident in our previous study where two clusters of agents, each learning a distinct task, collaborated between themselves. Various methods have been developed in the context of Federated Learning to identify coherent groups of agents using clustering techniques, aiming to enhance personalization [93–96]. In Federated Learning, the identification and estimation of agent clusters are more feasible since all agents communicate solely with a central server, which has the capability to create different models for different clusters of agents, thereby essentially enabling multi-task learning. In contrast, in decentralized peer-to-peer systems, there is no single entity orchestrating the entire process, leaving each agent responsible for its own learning trajectory. Identifying clusters within peer-to-peer systems presents a challenge, as agents themselves are tasked with identifying other similar agents within the network. However, if executed effectively, facilitating peer connections between agents sharing similar non-IID attributes can effectively transform the learning problem towards a more IID environment. This may facilitate the production of agents’ local models with similar parameters, ultimately aiding model convergence and the attainment of higher local accuracy.

As part of our first study [9], we sought to quantify the disparities among agents’ local

datasets to explore the variations and commonalities across different agents for Reddit [62] and StackOverflow [69] datasets. To achieve this, we employed the Jensen-Shannon divergence (JSD) metric, a technique for assessing the dissimilarity between two discrete probability distributions. In our context, JSD served as a means to assess the disparity in target label distributions between pairs of agents, essentially measuring the label distribution skew between agents. Let's denote the label probability distributions of two training datasets as  $P$  and  $Q$ , and let  $M$  represent the midpoint of the probability vectors  $P$  and  $Q$ . Vectors  $P$  and  $Q$  represent the probability of specific label being available in an agent's local dataset. Consider a scenario involving two agents with differing distributions of examples across distinct labels. In this case, the label probability for one agent can be represented by the vector  $P$ , while the label probability for the other agent can be expressed by the vector  $Q$ . The following vector examples illustrate the differing label probabilities for two distinct agents in a scenario where the dataset consists of ten unique class labels, resulting in vectors of length ten:

$$P_1 = \left[ 0.39 \quad 0.45 \quad 0.01 \quad 0.02 \quad 0.01 \quad 0.01 \quad 0.01 \quad 0.02 \quad 0.01 \quad 0.01 \right], \quad (11)$$

$$Q_2 = \left[ 0.01 \quad 0.02 \quad 0.01 \quad 0.02 \quad 0.01 \quad 0.01 \quad 0.40 \quad 0.43 \quad 0.01 \quad 0.01 \right]. \quad (12)$$

In this example,  $P_1$  represents the label probabilities for the first agent, while  $Q_2$  denotes the probabilities for the second agent. It is evident that the first agent has a majority of its local training examples concentrated in classes indexed as 0 and 1, whereas the second agent has the highest proportion of examples for class labels indexed as 6 and 7. This demonstrates a clear difference in the distribution of label examples between the two agents, reflecting heterogeneity in their respective local datasets.

The Jensen-Shannon divergence (JSD) between  $P$  and  $Q$  using the KL divergence  $D_{KL}$  can be expressed as follows:

$$JSD(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M), \quad (13)$$

$$M = \frac{1}{2}(P + Q), \quad (14)$$

$$D_{KL}(P\|Q) = \sum_{x \in X} P(x) \log \left( \frac{P(x)}{Q(x)} \right). \quad (15)$$

The distance measured by the metric falls within the  $[0, 1]$  range. A small divergence value (close to 0) indicates a high similarity between the class distributions in both datasets. Conversely, a high divergence value (close to 1) suggests a significant difference in the class distributions. Applying these equations to example vectors  $P_1$  and  $Q_2$  will result in:

$$M = \left[ 0.21 \quad 0.25 \quad 0.01 \quad 0.02 \quad 0.01 \quad 0.01 \quad 0.22 \quad 0.24 \quad 0.01 \quad 0.01 \right], \quad (16)$$

$$D_{KL}(P_1\|M) = 0.7262, D_{KL}(Q_2\|M) = 0.7254 \quad (17)$$

$$JSD(P_1\|Q_2) = \frac{1}{2} \cdot 0.7262 + \frac{1}{2} \cdot 0.7254; JSD(P_1\|Q_2) = 0.7258, \quad (18)$$

obtaining a JSD value of 0.72 signifying large divergence between label distributions for the two example agents.

The JSD was also computed for several baseline synthetic IID and non-IID environments to provide further insights into the non-IID properties of the Reddit and StackOverflow datasets. The MNIST [53] dataset of handwritten human digits, consisting of ten unique classes, was utilized to generate these synthetic environments. The following synthetic and realistic environments were assessed:

1. IID data setting [3]: This setting distributes all classes uniformly across different agents.
2. *Pathological* non-IID data setting [3]: Here, the dataset is partitioned such that each agent receives only two classes out of the ten.
3. *Practical* non-IID data setting [48]: In this scenario, the data is partitioned between agents so that each agent has data from all classes, but with varying distributions; specific classes may have a higher occurrence probability than others.
4. Reddit: Includes data grouped by individual user.
5. StackOverflow: Includes data grouped by individual user.

In all evaluated environments, 100-agent datasets were utilized. For the synthetic environments, this involved partitioning the MNIST data among the 100 agents according to specific data settings. In the case of the Reddit and StackOverflow datasets, each agent was allocated data from a distinct user. In the *practical* non-IID data setting, agents received 80% of examples from two dominant classes and 20% of examples from the remaining eight non-dominant classes. To analyze the differences between agents' datasets, the JSD metric was computed for each pair of agents, resulting in a matrix of distances between all agents. These distance matrices were sorted before plotting to accurately represent clusters of agents with similar class distributions.

Figure 16 visualizes the JSD distances among agents' datasets across various scenarios. In the IID data setting, the distances are predominantly close to zero, indicating high similarity among agents' datasets and forming a single cluster. Conversely, in the *pathological* non-IID data setting, substantial differences emerge between agents' datasets, leading to the formation of multiple distinct clusters around the diagonal axis. The similarities between agents within the same cluster are notably high, indicating consistency in their respective datasets. Conversely, the similarity between agents belonging to different clusters is significantly lower, signifying distinct differences in their datasets. In the *practical* non-IID data setting, the matrix reveals differences between agents organized into five clusters, with high inner-cluster similarities and consistent lower similarities between agents from different clusters (highlighted in green). For the Reddit dataset, four prominent clusters are observed, with lower similarities between agents from different clusters compared to inner-cluster similarities. Similarly, the StackOverflow dataset produces multiple clusters, each encapsulating agents with progressively decreasing similarity within an encompassing larger cluster.

The findings from the label distribution skew analysis indicate that within realistic non-IID datasets, there exist datasets with similar characteristics, albeit identifying them presents a challenge. Recent research endeavors have introduced methodologies utilizing data [39] or model [24–28, 38] similarity techniques to facilitate autonomous peer connection establishment among agents, deviating from randomly assigned connections. However, empirical investigations within these studies have predominantly concentrated on synthetic non-IID environments. In such environments, the developed methodologies have demonstrated favorable outcomes across a spectrum of synthetic non-IID scenarios when compared with a baseline approach representing random agent connections.

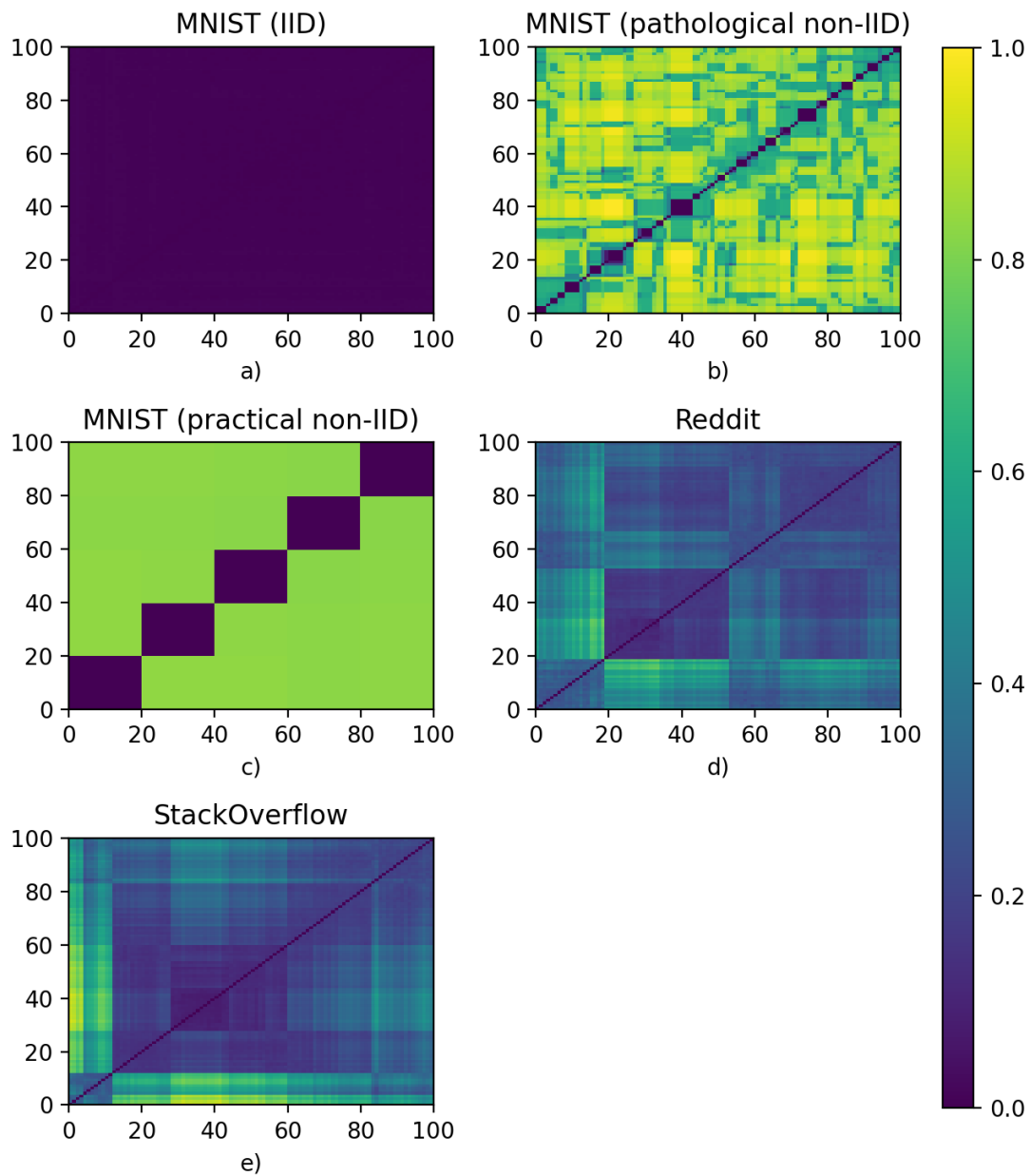


Figure 16: Distance matrices of Jensen-Shannon divergence between agents' local dataset label distribution for MNIST in: a) an IID data setting, b) a *pathological* non-IID data setting, and c) a *practical* non-IID data setting; and for Reddit (d) and StackOverflow (e) datasets [72]. Dark blue indicates substantial similarity between agents' datasets, while bright yellow suggests significant differences.

### 2.1.3.2. Personalized autonomous peer connection establishment methods

Personalized peer connections among agents have been explored within the realm of multi-task learning [15, 45, 73]. This approach involves organizing agents into distinct clusters, each tasked with learning different objectives toward a common goal. By segregating agents based on data heterogeneity, favorable learning outcomes can be achieved. These agent clusters work on identical problems but exclusively communicate with peers possessing similar datasets, mitigating the adverse effects of data heterogeneity. Various similarity metrics are employed to group peers together, drawing from metadata [73], gradients [95–97], model weights [40, 94, 95, 98], or local data model loss [16, 24, 27, 28, 99]. Metadata, for instance, may encompass geolocation, language, or user attributes like personality and gender [73]. Furthermore, data distribution skew can serve as metadata when designing topologies [39]. Similarity between gradients or model weights is often computed using metrics like cosine or Euclidean distance, with small Euclidean distances and large cosine similarities indicating agents with similar data distributions [27]. Finally, model similarity can be assessed by evaluating the loss value derived from the loss function on local data, with lower loss values suggesting greater similarity.

Several methods, including DAC [24], DiPLe [25], L2C [26], PANM [27], and PENS [28], have leveraged empirical model loss as a fundamental metric for quantifying model similarity. Despite variations in their implementations, each of these approaches shares a common characteristic: the absence of a predetermined communication mixing matrix. Instead, agents engage in communication with randomly selected peers, suggesting that every agent possesses a comprehensive overview of the entire network. PANM also introduced PANMGrad, a more efficient method for calculating model similarity, utilizing cosine similarity to express similarity between gradients. However, PANMGrad requires both current gradients  $g$  and accumulated weight deltas  $h$  to calculate similarity.

Moreover, the AUCCCR algorithm [38] possesses the capability to infer the number of clusters from the data without requiring a pre-determined number of clusters. In the context of peer-to-peer learning, this data can be represented as each agent’s local model output probabilities for a given test dataset. High similarity between two sets of probabilities indicates similar agents, thereby facilitating the grouping of agents into clusters. The current implementation of the AUCCCR method exhibits a centralized aspect, as it necessitates gathering all agents’

model output probabilities on a single test dataset to establish groups and form connections between agents within each group. However, if demonstrated as effective method for peer-to-peer learning, further research endeavors could focus on enhancing and decentralizing this approach. Furthermore, the D-Cliques method [39] designs a communication matrix based on label distribution between different local datasets, grouping agents into cliques with a maximum size of  $M$ .

### 2.1.3.3. Evaluation methodology

The learning process of agents was simulated in memory in a synchronized manner, as all approaches presumed a synchronized network. These methods were evaluated on both a computer vision (image classification) task and a NLP (next-word prediction) task. For the computer vision task, one hundred agents were simulated using the CIFAR-10 dataset [54] under various synthetic non-IID scenarios, while for the NLP task, the simulation involved one hundred to two hundred agents utilizing the Reddit dataset [62] involving a realistic non-IID scenario. The primary objective of the experiments is to quantify the average local model accuracy of agents and the average number of exchanged messages for each method. The number of neighbor peers was set to three, and specific parameters of individual methods were adjusted accordingly. A fixed random *sparse* topology served as a baseline environment, wherein connections between agents were formed randomly, regardless of their cluster affiliation. Additionally, results of *oracle* training, wherein agents only collaborated with peers sharing the same data properties (e.g., image rotation), were provided. Experiments were conducted in scenarios with two and four clusters of agents, with agents within the cluster sharing similar local data properties. Furthermore, all experiments were conducted for scenarios involving a small and large amount of local training data available to each agent.

Given the distinctive peer selection protocols utilized in the examined methods, a communication analysis was conducted to assess the equitable distribution of communication load. To quantify this communication equilibrium, the Gini coefficient was employed. The Gini coefficient ranges from zero, denoting perfect equality in communication distribution, to one, indicating near-maximal centralization where all communication is directed to a single peer. Furthermore, Pareto analysis was utilized to discern optimal solutions within the conducted ex-



periments, facilitating the identification of strategies that achieve a favorable balance between communication efficiency and model performance.

### 2.1.3.4. Experimental results

**Synthetic non-IID environment.** To create a synthetic non-IID environment, agents clustered together were provided with tailored CIFAR-10 datasets, incorporating variations such as image rotation, label-swapping, or adjustments to data partitioning. In experiments involving two clusters, the agents were divided such that half of them received unmodified CIFAR-10 data, while the remaining half were provided with modified CIFAR-10 data. Similarly, each cluster in the four-cluster experiments received a differently altered data.

To simulate a scenario where agents possess distinct features (such as rotation) associated to identical labels [7], agents from separate clusters were allocated data with varying degrees of rotation. Results shown in Table 8 highlight only PANMGrad, PANMLoss and PENS methods as consistently outperforming the *sparse* baseline. In the two-cluster scenario, optimal solutions identified by the Pareto method encompassed *oracle*, AUCCCR, PANMGrad, and PANMLoss. Conversely, in the four-cluster scenario, only *oracle* and AUCCCR were deemed optimal.

Table 8: Accuracy results for the synthetic non-IID environment simulated by varying image rotation between different clusters of agents [32].

| Method    | {0°, 180°}          |                     | {0°, 90°, 180°, 270°} |                     |
|-----------|---------------------|---------------------|-----------------------|---------------------|
|           | T <sub>s</sub> =100 | T <sub>s</sub> =400 | T <sub>s</sub> =100   | T <sub>s</sub> =400 |
| Oracle    | 43.80               | 49.44               | 41.52                 | 46.79               |
| Sparse    | 40.94               | 46.16               | 36.41                 | 41.96               |
| AUCCCR    | <b>41.33</b>        | <b>49.22</b>        | 33.85                 | <b>42.61</b>        |
| DAC       | 40.61               | <b>47.98</b>        | 36.32                 | <b>42.61</b>        |
| D-Cliques | 33.69               | 27.16               | 28.97                 | 16.77               |
| DiPLe     | 39.51               | <b>49.35</b>        | 35.50                 | <b>44.80</b>        |
| L2C       | 33.05               | 42.83               | 29.11                 | <b>42.12</b>        |
| PANMGrad  | <b>41.12</b>        | <b>49.62</b>        | <b>36.56</b>          | <b>45.48</b>        |
| PANMLoss  | <b>41.11</b>        | <b>49.67</b>        | <b>36.51</b>          | <b>46.17</b>        |
| PENS      | <b>41.29</b>        | <b>48.06</b>        | <b>36.46</b>          | <b>42.74</b>        |

To evaluate the effectiveness of the analyzed methods in a scenario where agents share identical features (images) associated with different labels [7], an experiment involving label swapping was conducted. Agents within specific clusters were allocated data with modified la-

bels, resulting in certain labels being swapped between features. In the two-cluster experiments, agents in the first cluster received unaltered data, while agents in the second cluster received data where the class index 0 was swapped with the class index 2, and vice versa. Table 9 presents the results obtained. In this non-IID scenario, no discernible best solution was observed, although the PENS method consistently outperformed the baseline in all scenarios except the four-cluster scenario with a large training dataset. The solutions residing on the Pareto front in both the two-cluster and four-cluster scenarios, involving label swap experiments, include *oracle*, PANMGrad, PANMLoss, and DiPLe, which emerged as optimal solutions. Additionally, PENS was identified as residing on the Pareto front in the two-cluster experiments.

Table 9: Accuracy results for the synthetic non-IID environment simulated by varying label swaps between different clusters of agents [32].

| Method    | {None, [0, 2]} |              | {None, [0, 1], [2, 3], [4, 5]} |              |
|-----------|----------------|--------------|--------------------------------|--------------|
|           | $T_s=100$      | $T_s=400$    | $T_s=100$                      | $T_s=400$    |
| Oracle    | 43.98          | 49.90        | 40.72                          | 46.83        |
| Sparse    | 45.51          | 50.77        | 42.10                          | 48.53        |
| AUCCCR    | 42.21          | 49.83        | 37.58                          | 45.21        |
| DAC       | 44.82          | 49.41        | 42.02                          | 46.28        |
| D-Cliques | 36.59          | 30.46        | 32.36                          | 26.49        |
| DiPLe     | 44.73          | <b>54.92</b> | 40.57                          | <b>51.86</b> |
| L2C       | 36.29          | 43.68        | 33.20                          | 43.13        |
| PANMGrad  | 45.34          | <b>52.10</b> | <b>42.40</b>                   | <b>49.77</b> |
| PANMLoss  | 45.47          | 49.92        | <b>42.17</b>                   | 46.32        |
| PENS      | <b>45.52</b>   | <b>52.17</b> | <b>42.35</b>                   | 48.52        |

To evaluate the performance of the studied methods under label distribution skew [7], the methods were assessed across three distinct scenarios:

1. A two-cluster setting with one cluster exclusively containing animal examples and the other exclusively containing vehicle classes [24].
2. The *pathological non-IID* data setting [3], where the dataset is partitioned such that each agent is allocated examples from only two out of ten possible classes.
3. The *practical non-IID* data setting [48], where the data is distributed among agents ensuring that each agent possesses data from all classes, albeit with varying class distributions, resulting in some classes having higher probabilities than others.

In both the *pathological* and *practical non-IID* scenarios, five distinct clusters are formed. Table 10 summarizes the results of the experiment. PANMGrad, PANMLoss and PENS methods

consistently outperformed the *sparse* baseline in all scenarios. PANMGrad emerged as the optimal solution across all scenarios. Furthermore, additional solutions residing on the Pareto front include DAC in the two-cluster scenario, PANMLoss in the *pathological non-IID* scenario, and DiPLe in the *practical non-IID* scenario.

Table 10: Accuracy results for the synthetic non-IID environment simulated by varying data partitioning methods applied between different clusters of agents [32].

| Method    | {Vehicles, Animals} |              | Pathological non-IID |              | Practical non-IID |              |
|-----------|---------------------|--------------|----------------------|--------------|-------------------|--------------|
|           | $T_s=100$           | $T_s=400$    | $T_s=100$            | $T_s=400$    | $T_s=100$         | $T_s=400$    |
| Oracle    | 55.18               | 61.22        | 83.73                | 84.83        | 40.83             | 67.70        |
| Sparse    | 49.70               | 55.87        | 76.68                | 57.68        | 48.21             | 67.50        |
| AUCCCR    | <b>55.07</b>        | <b>61.45</b> | <b>83.50</b>         | <b>83.82</b> | 42.38             | 67.48        |
| DAC       | <b>52.11</b>        | <b>56.84</b> | <b>79.10</b>         | <b>75.54</b> | <b>47.74</b>      | 64.80        |
| D-Cliques | <b>49.79</b>        | 53.76        | 34.29                | 29.48        | 42.31             | 35.10        |
| DiPLe     | <b>52.74</b>        | <b>61.18</b> | <b>77.14</b>         | 54.13        | <b>48.22</b>      | <b>71.09</b> |
| L2C       | 46.59               | <b>55.36</b> | <b>77.70</b>         | <b>76.28</b> | 38.70             | <b>70.65</b> |
| PANMGrad  | <b>55.85</b>        | <b>62.43</b> | <b>79.70</b>         | <b>78.49</b> | <b>48.98</b>      | <b>69.37</b> |
| PANMLoss  | <b>55.27</b>        | <b>61.89</b> | <b>83.68</b>         | <b>86.84</b> | <b>48.93</b>      | <b>68.95</b> |
| PENS      | <b>52.25</b>        | <b>58.84</b> | <b>77.81</b>         | <b>72.33</b> | <b>48.72</b>      | <b>68.94</b> |

**Realistic non-IID environment.** The user data from the Reddit dataset was segmented into groups based on the subreddit where the user posted. Only users who exclusively posted in one of the four most popular subreddits (*politics*, *leagueoflegends*, *nba*, or *Bitcoin*) were considered. Each resulting cluster comprises users with shared interests within the cluster, while demonstrating dissimilarity in interests compared to all other clusters. For example, users who exclusively engage in the *nba* subreddit may not share any common interests with users who solely participate in the *Bitcoin* subreddit. Experiments were conducted for each pair of clusters, along with an experiment involving all four clusters combined, yielding a total of seven distinct scenarios. Experiments with two clusters were merged due to their similar accuracies. The AUCCCR method was omitted from the subsequent experiments due to its requirement for vector sizes proportional to the square of the number of classes. This resulted in excessively large vectors for the NWP task involving 10,000 classes.

Table 11 presents a summary of the results. DAC, DiPLe, PANMGrad, and PENS consistently surpassed the *sparse* baseline across all scenarios. DAC and DiPLE emerged as solutions residing on the Pareto front for the realistic non-IID environment. In the two-cluster scenario with limited data, additional solutions included L2C, PANMGrad, and PANMLoss. In the two-cluster scenario with a large dataset, additional solutions were *oracle* and *sparse*. For

the four-cluster scenario with limited data, additional solutions comprised *oracle*, D-Cliques, PANMGrad, and PANMLoss. In the four-cluster scenario with a large dataset, additional solutions were *oracle*, D-Cliques, and PANMGrad.

Table 11: Accuracy results for the realistic non-IID environment simulated by dividing agents into clusters based on the subreddit to which the users posted [32].

| Method          | 2-cluster average |                 | 4-cluster         |                 |
|-----------------|-------------------|-----------------|-------------------|-----------------|
|                 | $300 < T_s < 700$ | $1k < T_s < 5k$ | $300 < T_s < 700$ | $1k < T_s < 5k$ |
| Oracle          | 6.48              | 8.11            | 6.40              | 8.10            |
| Sparse          | 6.72              | 8.35            | 6.76              | 8.65            |
| <b>DAC</b>      | <b>7.20</b>       | <b>8.94</b>     | <b>7.33</b>       | <b>9.20</b>     |
| D-Cliques       | 5.60              | <b>8.92</b>     | <b>8.58</b>       | <b>10.70</b>    |
| <b>DiPLe</b>    | <b>7.37</b>       | <b>9.98</b>     | <b>7.65</b>       | <b>10.50</b>    |
| L2C             | 6.12              | 7.64            | <b>6.85</b>       | <b>10.50</b>    |
| <b>PANMGrad</b> | <b>7.10</b>       | <b>8.56</b>     | <b>7.22</b>       | <b>8.82</b>     |
| PANMLoss        | 5.86              | 7.13            | 6.02              | 7.39            |
| <b>PENS</b>     | <b>7.14</b>       | <b>8.52</b>     | <b>7.31</b>       | <b>8.82</b>     |

**Communication analysis.** To assess the centralization tendencies of the studied methods under various synthetic and realistic non-IID environments, the Gini coefficient was employed to gauge the distribution of communication between agents. A low Gini coefficient suggests that all agents have a similar communication load, whereas a higher Gini coefficient indicates that agents primarily source messages from a restricted peer set. The Gini coefficient values for the topologies utilized in the *oracle*, *sparse*, and AUCCCR experiments unsurprisingly yielded a coefficient of 0, indicating perfect communication balance. Methods D-Cliques and DiPLe demonstrated Gini coefficients below 0.1 in both synthetic and realistic non-IID environments, indicating equitable distribution of communication load among agents. Furthermore, PANMGrad, PANMLoss, and PENS exhibited Gini coefficients below 0.1 in the synthetic non-IID environment, but showed increased coefficients ( $gini < 0.3$ ) in the realistic non-IID environment, suggesting higher centralization tendencies. While DAC demonstrated a Gini coefficient below 0.1 in the realistic non-IID environment, it exhibited a higher coefficient for synthetic non-IID environments. Conversely, the L2C method showed very high centralization tendencies in both synthetic and realistic non-IID environments.

In terms of the number of exchanged messages, all methods were adjusted to maintain communication with approximately the same number of peers. Compared to the baseline *sparse* topology, AUCCCR, DAC, PANMGrad, and PANMLoss methods exhibit similar message com-

munication frequencies, while the PENS method shows a 15% increase. The L2C method demonstrates a 48% increase in message communication in synthetic non-IID environments and a 12-fold increase in realistic non-IID environments. D-Cliques and DiPLe methods display significant inefficiencies in terms of communication load, with D-Cliques experiencing a minimum 50-fold increase and DiPLe demonstrating a 6-fold increase in message communication.

#### 2.1.3.5. Discussion

This study investigated communication-efficient peer-to-peer learning methodologies within the context of non-IID data distributions, focusing on the autonomous and personalized creation of connections between agents. The studied methods were analyzed and compared through a series of experiments varying in cluster counts, dataset sizes, and the degree of non-IID properties in agents' local data. Overall, in experiments regarding the synthetic non-IID environment, PANMGrad and PANMLoss methods were identified as the optimal solutions, suggesting their effectiveness in the image classification task. Coupled with low communication requirements per agent and low centralization properties, the PANM method was deemed a superior approach in synthetic non-IID environments compared to the *sparse* baseline, even outperforming the *oracle* reference.

In the experiments involving the next-word prediction task on a realistic non-IID dataset, the *oracle*, DAC, DiPLe, PANMGrad, and PANMLoss methods were identified as residing on the Pareto front. DiPLe was found to be very inefficient regarding the communication load for agents, requiring a 6-fold increase in messages compared to the *sparse* baseline, which significantly diminishes its applicability in realistic scenarios. DAC, PANMGrad, and PANMLoss demonstrated comparable communication loads per agent as the *sparse* baseline, with DAC displaying low centralization tendencies, while PANMGrad and PANMLoss exhibited higher centralization properties.

Given the communication load on agents and the observed centralization tendencies, the PANM method, in both its gradient-based (PANMGrad) and loss-based (PANMLoss) variants, emerges as the optimal choice in conducted experiments. The centralization tendencies observed in the PANM method may be subject to investigation and mitigation in future studies.

Furthermore, researchers should prioritize the development of modifications that maintain or enhance communication efficiency while mitigating centralization tendencies. Currently, all examined methods require access to the complete network, involving all agents, especially in the initial phases. However, this assumption is impractical for large and volatile decentralized systems with a high churn ratio. Hence, future investigations should focus on developing communication-efficient methods that enable agents to identify similar peers without necessitating communication with the entire network.

We consider methodologies reliant on measuring agent similarity based on model loss as more efficient than the PANMGrad method, which requires double the payload size exchanged between agents. However, methods based on model loss similarity may not be optimal when considering adversarial agents within the peer-to-peer network. Given that the objective of adversarial attackers is to inject triggers that cause targeted misclassifications without compromising model accuracy or disrupting convergence, it is quite challenging to detect adversarial models based solely on loss similarity.

This study contributes to Contribution **C4**, which involves designing a framework for the evaluation, comparison, and identification of optimal methods for autonomous and personalized peer connection creation in synthetic and realistic non-IID environments. The experiments conducted by simulating agents' learning process within these environments identified PANMGrad and PANMLoss as optimal for synthetic environments, and DAC, DiPLe, PANMGrad, and PANMLoss for realistic non-IID environments. While PANMGrad and PANMLoss exhibited low centralization tendencies in synthetic non-IID environments, these tendencies were heightened in realistic non-IID environments. It was observed that the DiPLe method is highly inefficient in terms of communication, while the DAC method was only optimal in realistic non-IID environments with low centralization tendencies. In contrast, the DAC method underperformed in synthetic non-IID environments with higher centralization tendencies. Although the PANM method was identified as optimal in both synthetic and realistic non-IID environments, further studies are needed to mitigate its centralization tendencies.

### 3. CONCLUSION

This doctoral thesis is centered around mitigating the adverse effects of non-IID data in peer-to-peer learning environments. Given the significant diversity in agents' local data, methods aimed at addressing the negative impact of non-IID can be categorized as personalization techniques. These personalization techniques encompass on-device personalization, multi-task learning, and personalized peer connection establishment between peers. For the purpose of investigating the impact of realistic non-IID data, realistic dataset collected at user level were utilized in the thesis, namely Reddit and StackOverflow datasets on a next-word prediction task.

A novel peer-to-peer learning method combined with a personalization technique was devised and contrasted with baseline peer-to-peer learning methods using the Reddit and StackOverflow datasets, consisting of users' posts and comments. Simulations indicate that the developed approach, on average, attains a mean relative top accuracy increase of 16.9% in ring (19.9% for Reddit, 13.9% for StackOverflow dataset) and 29.8% in sparse (32.9% for Reddit, 26.6% for StackOverflow dataset) communication topologies compared to the best baseline approach.

When considering heterogeneous environments where groups of agents may share similar data characteristics within their group but not with agents outside the group, an extreme case was explored in which groups of agents are learning distinctly different NLP tasks. A new multi-task method was developed, employing an encoder-only architecture based on the BERT model as the shared component across all agents, alongside a task-specific component exchanged solely between agents learning the same task. With additional consideration of communication topology and convergence enhancement strategies, the experiments demonstrated a statistically significant increase of 11.6% in mean relative accuracy compared to baseline results where agents only collaborated with peers learning the same task. These results can be interpreted in two ways. Firstly, they affirm that the proposed multi-task method is effective in scenarios where groups of agents share similar data properties. Secondly, this method can also be applied in scenarios where agents learning a specific task lack sufficient local data due to scarcity and non-IID properties, allowing them to collaborate with agents learning a different task within the same data modality to enhance their local model performance.

Identifying similar peers poses a significant challenge, particularly in decentralized systems.

This thesis evaluated several methods proposed in related studies for this purpose, assessing their performance in synthetic and realistic non-IID environments. The evaluation criteria included the top average accuracy of agents’ local models, the number of exchanged messages, and the centralization tendencies of each method, measured using the Gini coefficient. The experiments revealed that the PANMGrad and PANMLoss methods overall outperformed others. Notably, the PANM method also surpassed the *sparse* baseline, where connections between agents were randomly assigned, affirming that personalized peer connections indeed lead to enhanced local model performance.

### 3.1. Scientific contribution

The doctoral thesis is founded on anticipated scientific contributions outlined in Section 1.5. Herein, we seek to underscore these contributions once more, offering further elucidation for each distinct contribution alongside evidence demonstrating the attained results. Accordingly, the expected scientific contributions were:

**C1 A method for personalizing peer-to-peer agents’ model based on normalization layers.**

A method for personalizing agents’ local models is introduced, which leverages Batch Normalization layers as the central component of the personalization technique. An early stopping-inspired approach is employed to ascertain the optimal timing for conducting the personalization process.

**C2 A method for establishing peer connections and exchanging model weights among agents within a multi-task learning environment.**

A technique is proposed that restricts the number of peer connections between agents learning different tasks. Experimental findings demonstrate that this approach leads to improved average local model accuracy among agents, compared to the baseline scenario where agents exclusively collaborate with peers learning the same task.



### **C3 A method for training shared model layers to achieve faster convergence and better accuracy in a multi-task learning environment.**

A technique is introduced designed to strategically freeze the shared model layers to accelerate convergence in multi-task learning environments. Specifically, upon receiving model parameters from a peer learning a different task, the shared model layers are frozen and maintained as such for the subsequent training step.

### **C4 A framework for evaluation, comparison, and selection of optimal methods for autonomous and personalized peer connection creation, in synthetic and realistic non-IID environments.**

Assessment of methods for autonomous and personalized peer connection establishment across were conducted across diverse simulated synthetic non-IID and realistic non-IID environments. Findings indicate that the PANM method, in both its variants, emerges as the optimal approach overall. Moreover, the PANM method (including some other methods) outperformed the *sparse* baseline, underscoring the positive influence of personalized peer connection establishment on agents' local model accuracy.

## **3.2. Future research directions**

The scientific contributions outlined in this doctoral thesis focus on mitigating the impact of non-IID data on the learning process through diverse personalization techniques. To further facilitate personalization for each agent, the model architecture should also be adaptable to customization. Each agent ought to possess the autonomy to define its own model based on the availability of local data and computational capabilities of the local device. Given the potential existence of heterogeneous agents, where some agents learn at a faster pace than others, future research should address such scenarios and explore their implications in non-IID environments.

This doctoral thesis employed the next-word task across experiments conducted in the three studies. Notably, the data preprocessing involved the use of a predefined tokenizer tasked with converting sentences into sequences of numbers. Although it is customary in the literature to

assume a globally shared tokenizer, this is identified as a limitation and a potential area for future investigation. Specifically, this could entail research aimed at establishing a consensus on the tokenizer to be utilized in the training process among different agents within the network.

In examining autonomous methods for personalized peer connection establishment, a common absence of restrictions on the number of peer connections per agent was observed. This lack of constraints could potentially lead to heightened centralization tendencies. Additionally, these methods often necessitate knowledge of the entire network of peers, particularly in the initial stages, which may not be feasible in realistic peer-to-peer learning environments. This aspect represents an open area for further research and exploration.

The continuation of research in peer-to-peer machine learning, particularly when applied to realistic datasets, holds significant promise for addressing several pressing concerns in the evolving landscape of decentralized learning systems. By fostering privacy-preserving methods and improving security protocols, such research aligns with increasing global demands for robust data protection and regulatory compliance. Furthermore, peer-to-peer machine learning offers a compelling opportunity to reduce reliance on centralized cloud infrastructure, translating into lower operational costs for cloud services. This is particularly advantageous for smaller organizations or independent researchers who may not have the resources to maintain extensive cloud-based infrastructure. An equally important future benefit is the mitigation of centralization biases, as data generated from diverse edge devices could provide a more accurate and representative model of the real world, overcoming biases often inherent in centrally aggregated datasets. Finally, peer-to-peer systems are inherently more resilient, offering no single point of failure (SPOF), thus enhancing system robustness and making them less vulnerable to large-scale disruptions. Given these long-term benefits, it is crucial that research continues to push forward in this area, ensuring that future machine learning models are more equitable, secure, and efficient.

## 4. ABSTRACTS OF ARTICLES

### 4.1. Peer-to-peer deep learning with non-IID data

Collaborative training of deep neural networks using edge devices has attracted substantial research interest recently. The two main architecture approaches for the training process are centrally orchestrated Federated Learning and fully decentralized peer-to-peer learning. In decentralized systems, edge devices, known as agents, collaborate in a peer-to-peer architecture, avoiding the need for a central system to orchestrate the process. Decentralized peer-to-peer (P2P) learning techniques are well researched under the assumption of independent and identically distributed (IID) data across the agents. IID data is seldom observed in real-world distributed systems, and the training performance varies significantly with non-IID data. This paper proposes a decentralized learning variant of the P2P gossip averaging method with Batch Normalization (BN) adaptation for P2P architectures. It is well-known that BN layers accelerate the convergence of the non-distributed deep learning models. Recent research confirms that Federated Learning methods benefit from using the BN method with some aggregation alterations. Our work demonstrated BN effectiveness in P2P architectures by mitigating the non-IID data characteristics across decentralized agents. We also introduce a variant of the early stopping technique that, combined with BN layers, acts as a fine-tuning technique for agent models. We validated our approach by conducting numerous simulations of different model-topology-communication combinations and comparing them to other decentralized baseline approaches. The evaluations were conducted on the next word prediction task using user comments from the Reddit and StackOverflow datasets representing comments from two different domains. Simulations showed that our approach, on average, achieves a mean relative top accuracy increase of 16.9% in ring (19.9% for Reddit, 13.9% for StackOverflow) and 29.8% in sparse (32.9% for Reddit, 26.6% for StackOverflow) communication topologies compared to the best baseline approach. Our code is available at [https://github.com/fipu-lab/p2p\\_bn](https://github.com/fipu-lab/p2p_bn).

Available at:

<https://www.sciencedirect.com/science/article/abs/pii/S0957417422021777>

*Robert Šajina, Nikola Tanković, Ivo Ipšić, 2023. Peer-to-peer deep learning with non-IID data.  
Expert Systems with Applications, Volume 214, ISSN 0957-4174,  
DOI: 10.1016/j.eswa.2022.119159.*

## 4.2. Multi-task peer-to-peer learning using an encoder-only transformer model

Peer-to-peer (P2P) learning is a decentralized approach to organizing the collaboration between end devices known as agents. Agents contain heterogeneous data, and that heterogeneity is disrupting the convergence and accuracy of the collectively learned models. A common technique to mitigate the negative impact of heterogeneous data is to arrange the learning process in a multi-task setting where each task, although it has the same learning objective, is learned separately. However, the multi-task technique can also be applied to solve distinct learning tasks. This paper presents and evaluates a novel approach that utilizes an encoder-only transformer model to enable collaboration between agents learning two distinct Natural Language Processing (NLP) tasks. The evaluation of the approach studied revealed that collaboration among agents, even when working towards separate objectives, can result in mutual benefits, mainly when the connections between agents are carefully considered. The multi-task collaboration led to a statistically significant increase of 11.6% in the mean relative accuracy compared to the baseline results for individual tasks. To our knowledge, this is the first study demonstrating a successful and beneficial collaboration between two distinct NLP tasks in a peer-to-peer setting.

Available at:

<https://www.sciencedirect.com/science/article/abs/pii/S0167739X23004053>

*Robert Šajina, Nikola Tanković, Ivo Ipšić, 2024. Multi-task peer-to-peer learning using an encoder-only transformer model. Future Generation Computer Systems, Volume 152, ISSN 0167-739X, DOI: 10.1016/j.future.2023.11.006.*

### **4.3. An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning**

The exchange of model parameters between peers is critical in peer-to-peer deep learning. Historically, connections between agents were assigned randomly based on network topology. However, recent methodologies enable agents to autonomously establish their connections, which is especially beneficial for non-IID data settings. Recent studies suggest favorable learning outcomes for autonomous connections compared to fixed topology when evaluated in synthetic non-IID environments. To that end, this study will explore various methodologies to enhance learning outcomes. Through several large-scale experiments with synthetic and realistic non-IID data sets, it evaluates communication efficiency, message exchange frequency, and centralization tendencies. The findings underscore the potential of these methods to enhance local model accuracy, uphold communication efficiency, and resist centralization, rendering them highly suitable for decentralized learning systems. The evaluation results establish PANMGrad and PANMLoss as the most effective solutions in such environments.

Available at:

<https://ieeexplore.ieee.org/document/10633710>

*Robert Šajina, Nikola Tanković, Ivo Ipšić, 2024. An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning. IEEE Access, Volume 12, ISSN 2169-3536,*

*DOI: 10.1109/ACCESS.2024.3442014.*

## REFERENCES

- [1] Yuanxin Li and Darina Saxunová. “A perspective on categorizing Personal and Sensitive Data and the analysis of practical protection regulations”. In: *Procedia Computer Science* 170 (2020). The 11th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 3rd International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops, pp. 1110–1115. ISSN: 1877-0509. DOI: <https://doi.org/10.1016/j.procs.2020.03.060>. URL: <https://www.sciencedirect.com/science/article/pii/S187705092030497X>.
- [2] Peter Kairouz et al. “Advances and Open Problems in Federated Learning”. In: *Foundations and Trends® in Machine Learning* 14.1–2 (2021), pp. 1–210. ISSN: 1935-8237. DOI: 10.1561/22000000083. URL: <http://dx.doi.org/10.1561/22000000083>.
- [3] Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 1273–1282. URL: <https://proceedings.mlr.press/v54/mcmahan17a.html>.
- [4] Michael Blot et al. “Gossip training for deep learning”. In: (Nov. 2016).
- [5] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from [tensorflow.org](https://www.tensorflow.org), 2015. URL: <https://www.tensorflow.org/>.
- [6] Adam Paszke et al. “PyTorch: An Imperative Style, High-Performance Deep Learning Library”. In: *Advances in Neural Information Processing Systems* 32. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: <http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf>.
- [7] Xiaodong Ma et al. “A state-of-the-art survey on solving non-IID data in Federated Learning”. In: *Future Generation Computer Systems* 135 (2022), pp. 244–258. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2022.05.003>.

URL: <https://www.sciencedirect.com/science/article/pii/S0167739X22001686>.

- [8] Rong Yu and Peichun Li. “Toward Resource-Efficient Federated Learning in Mobile Edge Computing”. In: *IEEE Network* 35.1 (2021), pp. 148–155. DOI: 10.1109/MNET.011.2000295.
- [9] Robert Šajina, Nikola Tanković, and Ivo Ipšić. “Peer-to-peer deep learning with non-IID data”. In: *Expert Systems with Applications* 214 (2023), p. 119159. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2022.119159>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417422021777>.
- [10] Igor Colin et al. “Gossip Dual Averaging for Decentralized Optimization of Pairwise Functions”. In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, June 2016, pp. 1388–1396. URL: <https://proceedings.mlr.press/v48/colin16.html>.
- [11] Jeff A. Daily et al. “GossipGraD: Scalable Deep Learning using Gossip Communication based Asynchronous Gradient Descent”. In: *ArXiv abs/1803.05880* (2018).
- [12] Anastasia Koloskova, Sebastian Stich, and Martin Jaggi. “Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 3478–3487. URL: <https://proceedings.mlr.press/v97/koloskova19a.html>.
- [13] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. “Decentralized Collaborative Learning of Personalized Models over Networks”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 509–517. URL: <https://proceedings.mlr.press/v54/vanhaesebrouck17a.html>.



- [14] Inês Almeida and João Xavier. “DJAM: Distributed Jacobi Asynchronous Method for Learning Personal Models”. In: *IEEE Signal Processing Letters* 25.9 (2018), pp. 1389–1392. DOI: 10.1109/LSP.2018.2859596.
- [15] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. “Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 864–874. URL: <https://proceedings.mlr.press/v108/zantedeschi20a.html>.
- [16] Karim Boubouh et al. “Robust P2P Personalized Learning”. In: *2020 International Symposium on Reliable Distributed Systems (SRDS)*. 2020, pp. 299–308. DOI: 10.1109/SRDS51746.2020.00037.
- [17] Jiyi Li et al. “Distributed Multi-task Learning for Sensor Network”. In: *Machine Learning and Knowledge Discovery in Databases*. Ed. by Michelangelo Ceci et al. Cham: Springer International Publishing, 2017, pp. 657–672. ISBN: 978-3-319-71246-8.
- [18] Shangwei Guo et al. “Differentially Private Decentralized Learning”. In: *ArXivCoRR* (June 2020). URL: <http://arxiv.org/abs/2006.07817>.
- [19] Xiangru Lian et al. “Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/f75526659f31040afeb61cb7133e4e6d-Paper.pdf>.
- [20] Xiangru Lian et al. “Asynchronous Decentralized Parallel Stochastic Gradient Descent”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 3043–3052. URL: <https://proceedings.mlr.press/v80/lian18a.html>.
- [21] Hanlin Tang et al. “D<sup>2</sup>: Decentralized Training over Decentralized Data”. In: *Proceedings of the 35th International Conference on Machine Learning* 80.1 (2018), pp. 4848–4856. URL: <http://proceedings.mlr.press/v80/tang18a.html>.

- [22] Jianyu Wang and Gauri Joshi. “Cooperative SGD: A Unified Framework for the Design and Analysis of Local-Update SGD Algorithms”. In: *Journal of Machine Learning Research* 22.213 (2021), pp. 1–50. URL: <http://jmlr.org/papers/v22/20-147.html>.
- [23] Anusha Lalitha et al. “Peer-to-peer Federated Learning on Graphs”. In: *ArXiv abs/1901.11173* (2019). URL: <http://arxiv.org/abs/1901.11173>.
- [24] Edvin Listo Zec et al. “Decentralized Adaptive Clustering of Deep Nets is Beneficial for Client Collaboration”. In: *Trustworthy Federated Learning*. Ed. by Randy Goebel et al. Springer International Publishing, 2023, pp. 59–71. ISBN: 978-3-031-28996-5.
- [25] Xue Zheng, Parinaz Naghizadeh, and Aylin Yener. “DiPLe: Learning Directed Collaboration Graphs for Peer-to-Peer Personalized Learning”. In: *2022 IEEE Information Theory Workshop (ITW)*. Ed. by Institute of Electrical and Electronics Engineers. 2022, pp. 446–451. DOI: 10.1109/ITW54588.2022.9965838.
- [26] Shuangtong Li et al. “Learning to Collaborate in Decentralized Learning of Personalized Models”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 9756–9765. DOI: 10.1109/CVPR52688.2022.00954.
- [27] Zexi Li et al. “Towards Effective Clustered Federated Learning: A Peer-to-peer Framework with Adaptive Neighbor Matching”. In: *IEEE Transactions on Big Data* (2022), pp. 1–16. DOI: 10.1109/TBDATA.2022.3222971.
- [28] Noa Onoszko et al. “Decentralized federated learning of deep neural networks on non-iid data”. In: *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML*. Vol. abs/2107.08517. 2021.
- [29] Mahmoud Assran et al. “Stochastic Gradient Push for Distributed Deep Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 344–353. URL: <https://proceedings.mlr.press/v97/assran19a.html>.
- [30] Chaoyang He et al. “Central Server Free Federated Learning over Single-sided Trust Social Networks”. In: *ArXiv abs/1910.04956* (2019). URL: <http://arxiv.org/abs/1910.04956>.

- [31] Aurélien Bellet et al. “Personalized and Private Peer-to-Peer Machine Learning”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, Apr. 2018, pp. 473–481. URL: <https://proceedings.mlr.press/v84/bellet18a.html>.
- [32] Robert Šajina, Nikola Tanković, and Ivo Ipšić. “An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning”. In: *IEEE Access* 12 (2024), pp. 111752–111768. DOI: 10.1109/ACCESS.2024.3442014.
- [33] S. Boyd et al. “Randomized gossip algorithms”. In: *IEEE Transactions on Information Theory* 52.6 (2006), pp. 2508–2530. DOI: 10.1109/TIT.2006.874516.
- [34] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “Robust Privacy-Preserving Gossip Averaging”. In: *SSS 2019 - 21st International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Pisa, Italy: Springer, Oct. 2019, pp. 38–52. DOI: 10.1007/978-3-030-34992-9\_4. URL: <https://hal.archives-ouvertes.fr/hal-02373353>.
- [35] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “Simple, Efficient and Convenient Decentralized Multi-Task Learning for Neural Networks”. In: *IDA 2021 - 19th Symposium on Intelligent Data Analysis*. Vol. 12695. Lecture Notes in Computer Science. Porto, Portugal: Springer, Apr. 2021. DOI: 10.1007/978-3-030-74251-5\_4. URL: <https://hal.archives-ouvertes.fr/hal-02373338>.
- [36] Michael Blot et al. “Distributed optimization for deep learning with gossip exchange”. In: *Neurocomputing* 330 (2019), pp. 287–296. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2018.11.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231218313195>.
- [37] Robert Šajina, Nikola Tanković, and Darko Etinger. “Decentralized trustless gossip training of deep neural networks”. In: *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. 2020, pp. 1080–1084. DOI: 10.23919/MIPRO48935.2020.9245248.
- [38] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “AUCCCR: Agent Utility Centered Clustering for Cooperation Recommendation”. In: *Networked Systems*. Ed. by

- Karima Echihabi and Roland Meyer. Cham: Springer International Publishing, 2021, pp. 111–125.
- [39] Aurélien Bellet, Anne-Marie Kermarrec, and Erick Lavoie. “D-Cliques: Compensating for Data Heterogeneity with Topology in Decentralized Federated Learning”. In: *2022 41st International Symposium on Reliable Distributed Systems (SRDS)* (2021), pp. 1–11.
- [40] Christopher Briggs, Zhong Fan, and Péter András. “Federated learning with hierarchical clustering of local updates to improve training on non-IID data”. In: *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), pp. 1–9.
- [41] Kevin Hsieh et al. “The non-IID data quagmire of decentralized machine learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. ICML’20. JMLR.org, 2020.
- [42] Kangkang Wang et al. “Federated Evaluation of On-device Personalization”. In: *ArXiv abs/1910.10252* (2019). URL: <http://arxiv.org/abs/1910.10252>.
- [43] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. *Adaptive Personalized Federated Learning*. 2021.
- [44] Virginia Smith et al. “Federated Multi-Task Learning”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017, pp. 4425–4435. URL: <https://proceedings.neurips.cc/paper/2017/file/6211080fa89981f66b1a0c9d55c61d0f-Paper.pdf>.
- [45] Mohammad Rostami et al. “Multi-Agent Distributed Lifelong Learning for Collective Knowledge Acquisition”. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. Ed. by Andre Elisabeth et al. AAMAS ’18. Stockholm, Sweden: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 712–720.
- [46] Javad Mohammadi and Soheil Kolouri. “Collaborative Learning Through Shared Collective Knowledge and Local Expertise”. In: *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. 2019, pp. 1–6. DOI: 10.1109/MLSP.2019.8918888.

- [47] Roula Nassif et al. “Multitask Learning Over Graphs: An Approach for Distributed, Streaming Machine Learning”. In: *IEEE Signal Processing Magazine* 37.3 (2020), pp. 14–25. DOI: 10.1109/MSP.2020.2966273.
- [48] Yutao Huang et al. “Personalized Federated Learning: An Attentive Collaboration Approach”. In: *CoRR* abs/2007.03797 (2020). URL: <https://arxiv.org/abs/2007.03797>.
- [49] Rui Hu et al. “Privacy-Preserving Personalized Federated Learning”. In: *ICC 2020 - 2020 IEEE International Conference on Communications ICC*. 2020, pp. 1–6. DOI: 10.1109/ICC40277.2020.9149207.
- [50] Manoj Ghuhan Arivazhagan et al. “Federated Learning with Personalization Layers”. In: *ArXiv* abs/1912.00818 (2019).
- [51] Yihan Jiang et al. “Improving Federated Learning Personalization via Model Agnostic Meta Learning”. In: *CoRR* (2020). URL: <http://arxiv.org/abs/1909.12488>.
- [52] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. “Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 3557–3568. URL: <https://proceedings.neurips.cc/paper/2020/file/24389bfe4fe2eba8bf9aa9203a44cdad-Paper.pdf>.
- [53] Li Deng. “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.
- [54] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 0. Toronto, Ontario: University of Toronto, 2009. URL: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
- [55] Han Xiao, Kashif Rasul, and Roland Vollgraf. *Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms*. Aug. 28, 2017. arXiv: cs.LG/1708.07747 [cs.LG].

- [56] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning*. Vol. 37. ICML’15. Lille, France: JMLR.org, 2015, pp. 448–456.
- [57] Xiaoxiao Li et al. “FedBN: Federated Learning on Non-IID Features via Local Batch Normalization”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [58] J. Mills, J. Hu, and G. Min. “Multi-Task Federated Learning for Personalised Deep Neural Networks in Edge Computing”. In: *IEEE Transactions on Parallel and Distributed Systems* 33.03 (Mar. 2022), pp. 630–641. ISSN: 1558-2183. DOI: 10.1109/TPDS.2021.3098467.
- [59] Mathieu Andreux et al. “Siloed Federated Learning for Multi-Centric Histopathology Datasets”. In: Lima, Peru: Springer-Verlag, 2020, pp. 129–139. ISBN: 978-3-030-60547-6. DOI: 10.1007/978-3-030-60548-3\_13. URL: [https://doi.org/10.1007/978-3-030-60548-3\\_13](https://doi.org/10.1007/978-3-030-60548-3_13).
- [60] Qiang Shen et al. “Federated Multi-Task Attention for Cross-Individual Human Activity Recognition”. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. Ed. by Lud De Raedt. International Joint Conferences on Artificial Intelligence Organization, July 2022, pp. 3423–3429. DOI: 10.24963/ijcai.2022/475. URL: <https://doi.org/10.24963/ijcai.2022/475>.
- [61] Othmane Marfoq et al. “Federated Multi-Task Learning under a Mixture of Distributions”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 15434–15447. URL: <https://proceedings.neurips.cc/paper/2021/file/82599a4ec94aca066873c99b4c741ed8-Paper.pdf>.
- [62] Sebastian Caldas et al. “LEAF: A Benchmark for Federated Settings”. In: *ArXiv abs/1812.01097* (2018).
- [63] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.

- [64] Jie Hu, Li Shen, and Gang Sun. “Squeeze-and-Excitation Networks”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7132–7141. DOI: 10.1109/CVPR.2018.00745.
- [65] Mingxing Tan and Quoc Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 6105–6114. URL: <https://proceedings.mlr.press/v97/tan19a.html>.
- [66] Asher Trockman and J Zico Kolter. *Patches Are All You Need?* 2022. URL: <https://openreview.net/forum?id=TVHS5Y4dNvM>.
- [67] Pramod Kaushik Mudrakarta et al. “K For The Price Of 1: Parameter Efficient Multi-task And Transfer Learning”. In: *International Conference on Learning Representations*. Vol. abs/1810.10703. 2019, pp. 1–15.
- [68] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. “Asynchronous Federated Optimization”. In: *ArXiv abs/1903.03934* (2019). URL: <http://arxiv.org/abs/1903.03934>.
- [69] StackOverflow. *Stack Overflow Data*. Kaggle. 2018. URL: <https://www.kaggle.com/datasets/stackoverflow/stackoverflow>.
- [70] Amir Rosenfeld and John K. Tsotsos. “Intriguing Properties of Randomly Weighted Networks: Generalizing While Learning Next to Nothing”. In: *2019 16th Conference on Computer and Robot Vision (CRV)* (2019), pp. 9–16. DOI: 10.1109/CRV.2019.00010.
- [71] Jonathan Frankle, David J. Schwab, and Ari S. Morcos. “Training BatchNorm and Only BatchNorm: On the Expressive Power of Random Features in CNNs”. In: *International Conference on Learning Representations*. 2021.
- [72] Robert Šajina, Nikola Tanković, and Ivo Ipšić. “Multi-task peer-to-peer learning using an encoder-only transformer model”. In: *Future Generation Computer Systems* 152 (2024), pp. 170–178. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2023.11.006>. URL: <https://www.sciencedirect.com/science/article/pii/S0167739X23004053>.

- [73] Sara Taylor et al. “Personalized Multitask Learning for Predicting Tomorrow’s Mood, Stress, and Health”. In: *IEEE Transactions on Affective Computing* 11.2 (2020), pp. 200–213. DOI: 10.1109/TAFFC.2017.2784832.
- [74] Theoni Pitoura and Peter Triantafillou. “Load Distribution Fairness in P2P Data Management Systems”. In: *2007 IEEE 23rd International Conference on Data Engineering*. 2007, pp. 396–405. DOI: 10.1109/ICDE.2007.367885.
- [75] Fabio Caccioli, Giacomo Livan, and Tomaso Aste. “Scalability and Egalitarianism in Peer-to-Peer Networks”. In: *Banking Beyond Banks and Money: A Guide to Banking Services in the Twenty-First Century*. Cham: Springer International Publishing, 2016, pp. 197–212. ISBN: 978-3-319-42448-4. DOI: 10.1007/978-3-319-42448-4\_11. URL: [https://doi.org/10.1007/978-3-319-42448-4\\_11](https://doi.org/10.1007/978-3-319-42448-4_11).
- [76] Bartosz Kusmierz and Roman Overko. “How centralized is decentralized? Comparison of wealth distribution in coins and tokens”. In: *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. Los Alamitos, CA, USA: IEEE Computer Society, Aug. 2022, pp. 1–6. DOI: 10.1109/COINS54846.2022.9854972. URL: <https://doi.ieeecomputersociety.org/10.1109/COINS54846.2022.9854972>.
- [77] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: <https://aclanthology.org/N19-1423>.
- [78] Qinyi Luo et al. “Prague: High-Performance Heterogeneity-Aware Asynchronous Decentralized Training”. In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 401–416. ISBN: 9781450371025. URL: <https://doi.org/10.1145/3373376.3378499>.
- [79] Yue Gao et al. “Topology Measurement and Analysis on Ethereum P2P Network”. In: *2019 IEEE Symposium on Computers and Communications (ISCC)*. 2019, pp. 1–7. DOI: 10.1109/ISCC47284.2019.8969695.



- [80] Taotao Wang et al. “Ethna: Analyzing the Underlying Peer-to-Peer Network of Ethereum Blockchain”. In: *IEEE Transactions on Network Science and Engineering* 8.3 (2021), pp. 2131–2146. DOI: 10.1109/TNSE.2021.3078181.
- [81] Andrew Howell, Takfarinas Saber, and Malika Bendeche. “Measuring node decentralisation in blockchain peer to peer networks”. In: *Blockchain: Research and Applications* 4.1 (2023), p. 100109. ISSN: 2096-7209. DOI: <https://doi.org/10.1016/j.bcra.2022.100109>. URL: <https://www.sciencedirect.com/science/article/pii/S2096720922000501>.
- [82] Andrew Hard et al. “Federated Learning for Mobile Keyboard Prediction”. In: *CoRR* abs/1811.03604 (2018). URL: <https://arxiv.org/abs/1811.03604>.
- [83] Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic Controlled Averaging for Federated Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 5132–5143. URL: <https://proceedings.mlr.press/v119/karimireddy20a.html>.
- [84] Sashank J. Reddi et al. “Adaptive Federated Optimization”. In: *International Conference on Learning Representations*. 2021.
- [85] Joel Stremmel and Arjun Singh. “Pretraining Federated Text Models for Next Word Prediction”. In: *Advances in Information and Communication*. Ed. by Kohei Arai. Springer International Publishing, 2021, pp. 477–488. ISBN: 978-3-030-73103-8.
- [86] Márk Jelasity, Alberto Montresor, and Özalp Babaoglu. “Gossip-based aggregation in large dynamic networks”. In: *ACM Trans. Comput. Syst.* 23 (2005), pp. 219–252.
- [87] Angelia Nedić and Alex Olshevsky. “Stochastic Gradient-Push for Strongly Convex Functions on Time-Varying Directed Graphs”. In: *IEEE Transactions on Automatic Control* 61.12 (2016), pp. 3936–3947. DOI: 10.1109/TAC.2016.2529285.
- [88] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing EMNLP*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/d14-1179. URL: <https://aclanthology.org/D14-1179>.

- [89] Nelson Morgan and Hervé Bourlard. “Generalization and Parameter Estimation in Feed-forward Nets: Some Experiments”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 2. Morgan-Kaufmann, 1989. URL: <https://proceedings.neurips.cc/paper/1989/file/63923f49e5241343aa7acb6a06a751e7-Paper.pdf>.
- [90] Iulia Turc et al. “Well-Read Students Learn Better: On the Importance of Pre-training Compact Models”. In: *arXiv preprint arXiv:1908.08962v2* (2019).
- [91] Erik F. Tjong Kim Sang and Fien De Meulder. “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. CONLL ’03. Edmonton, Canada: Association for Computational Linguistics, 2003, pp. 142–147. DOI: 10.3115/1119176.1119195. URL: <https://doi.org/10.3115/1119176.1119195>.
- [92] Ning Ding et al. “Few-NERD: A Few-shot Named Entity Recognition Dataset”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 3198–3213. DOI: 10.18653/v1/2021.acl-long.248. URL: <https://aclanthology.org/2021.acl-long.248>.
- [93] Avishek Ghosh et al. “An Efficient Framework for Clustered Federated Learning”. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Ed. by Larochelle H. et al. NIPS’20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.
- [94] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26 (June 2022), pp. 1–20. DOI: 10.1007/s11280-022-01046-x.
- [95] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints”. In: *IEEE Transactions on Neural Networks and Learning Systems* PP (Aug. 2020), pp. 1–13. DOI: 10.1109/TNNLS.2020.3015958.

- [96] Moming Duan et al. “Flexible Clustered Federated Learning for Client-Level Data Distribution Shift”. In: *IEEE Transactions on Parallel and Distributed Systems* PP (Dec. 2021), pp. 1–1. DOI: 10.1109/TPDS.2021.3134263.
- [97] Moming Duan et al. “FedGroup: Ternary Cosine Similarity-based Clustered Federated Learning Framework toward High Accuracy in Heterogeneous Data”. In: *CoRR* abs/2010.06870 (2020). arXiv: 2010.06870. URL: <https://arxiv.org/abs/2010.06870>.
- [98] Minh N. H. Nguyen et al. “Self-organizing Democratized Learning: Towards Large-scale Distributed Learning Systems”. In: *IEEE transactions on neural networks and learning systems* PP (2020).
- [99] Yi Sui et al. “Find Your Friends: Personalized Federated Learning with the Right Collaborators”. In: *Workshop on Federated Learning: Recent Advances and New Challenges, in Conjunction with NeurIPS 2022 (FL-NeurIPS’22)*. 2022. DOI: 10.48550/ARXIV.2210.06597. URL: <https://arxiv.org/abs/2210.06597>.

# LIST OF FIGURES

|    |   |    |
|----|---|----|
| 1  | Agent contains local data, a machine learning framework, and a model trained by the ML framework. . . . .   | 2  |
| 2  | Synchronous peer-to-peer learning process [9]. . . . .  | 3  |
| 3  | Visual communication representation of the $W_1$ undirected communication graph. . . . .  | 4  |
| 4  | Visual communication representation of the $W_2$ directed communication graph. . . . .  | 5  |
| 5  | Examples of most common network typologies in an undirected and directed communication [32]. . . . .  | 6  |
| 6  | Data distribution in IID environment. . . . .   | 9  |
| 7  | Feature distribution skew in non-IID environment. . . . .   | 10 |
| 8  | Label distribution skew in non-IID environment. . . . .   | 10 |
| 9  | Same label, different features in non-IID environment. . . . .  | 11 |
| 10 | Same features, different label in non-IID environment. . . . .  | 11 |
| 11 | Quantity skew in non-IID environment. . . . .   | 12 |
| 12 | The learning and communication process of three P2P-BN agents in an unbalanced topology [9]. . . . .  | 24 |
| 13 | Architecture of MTP and NER models [72]. . . . .  | 32 |
| 14 | An example of task-specific clusters (a), sparse (b), and <i>clustered sparse</i> (c) topology [72]. Agent color represents the learning task. . . . .  | 35 |
| 15 | An example of four different agent clusters with two task-specific and one non-task-specific connection randomly formed with another cluster of agents, regardless of the task [72]. . . . .  | 38 |
| 16 | Distance matrices of Jensen-Shannon divergence between agents' local dataset label distribution for MNIST in: a) an IID data setting, b) a <i>pathological</i> non-IID data setting, and c) a <i>practical</i> non-IID data setting; and for Reddit (d) and StackOverflow (e) datasets [72]. Dark blue indicates substantial similarity between agents' datasets, while bright yellow suggests significant differences. . . . . | 45 |

## LIST OF TABLES

|    |   |    |
|----|---|----|
| 1  | The average top UA and the average number of exchanged messages per agent for both the Reddit and StackOverflow datasets in ring topology using the non-BN model. Best results are emphasized. . . . .                          | 25 |
| 2  | The average top UA and the average number of exchanged messages per agent for both the Reddit and StackOverflow datasets in ring topology using the BN model. Best results are emphasized. . . . .                              | 27 |
| 3  | Average top UA results of 100 agents in fixed undirected and directed, and varying undirected and directed sparse topology experiments conducted on the Reddit and StackOverflow datasets. Best results are emphasized. . . . . | 27 |
| 4  | Average top Test UA achieved by agents in a sparse topology for each of the multi-task combinations [72]. . . . .   | 34 |
| 5  | Average top Test UA achieved by agents in a <i>clustered sparse</i> topology for each of the multi-task combinations [72]. . . . .  | 36 |
| 6  | Average top Test UA achieved by agents in a <i>clustered sparse</i> topology using the <i>MT-EF</i> method for each of the multi-task combinations [72]. . . . .  | 37 |
| 7  | Average top Test UA achieved by agents in a <i>clustered sparse</i> topology using the <i>MT-EF</i> method when all four multi-task groups of agents were collaborating simultaneously [72]. . . . .                            | 38 |
| 8  | Accuracy results for the synthetic non-IID environment simulated by varying image rotation between different clusters of agents [32]. . . . .   | 48 |
| 9  | Accuracy results for the synthetic non-IID environment simulated by varying label swaps between different clusters of agents [32]. . . . .  | 49 |
| 10 | Accuracy results for the synthetic non-IID environment simulated by varying data partitioning methods applied between different clusters of agents [32]. . . . .  | 50 |
| 11 | Accuracy results for the realistic non-IID environment simulated by dividing agents into clusters based on the subreddit to which the users posted [32]. . . . .  | 51 |

# APPENDIX

Appendix A, B, and C comprise the three published papers, respectively. Appendix D contains supplementary materials, including a link to a public GitHub repository containing instructions and reproducible code for all three published papers.

## **A. Peer-to-peer deep learning with non-IID data**

This work was published as: Robert Šajina, Nikola Tanković, Ivo Ipšić, 2023. Peer-to-peer deep learning with non-IID data. *Expert Systems with Applications*, Volume 214, ISSN 0957-4174, DOI: 10.1016/j.eswa.2022.119159

For clarity, the article has been reformatted, otherwise the content is the same as the published version of work. © 2023 by the author. Reproduced with permission from Elsevier.

<https://www.sciencedirect.com/science/article/abs/pii/S0957417422021777>

# 1. Introduction

An increasing amount of connected edge devices, such as mobile phones, tablets, and laptops, contain a vast amount of valuable but often private data. Sophisticated machine-learning models can utilize edge device data to alleviate user experience with intelligent recommender systems, voice recognition, typing predictions, or upgrade service quality in financial, health-care, and insurance domains, to name a few. There is a growing research interest in utilizing edge devices' local data and their computational capabilities to train the models collaboratively with strong privacy guarantees.

The two main approaches to collaborative training are: (1) Federated Learning (FL) [1], and (2) peer-to-peer (P2P) learning [2]. Federated Learning is a centralized approach to orchestrating the process of training a single shared model using the data from edge devices. Figure 1 shows the FL process overview. Prominent vendors often employ FL architectures (e.g., Google Keyboard, Netflix, Siri). P2P learning techniques rely on an established choreography that every edge device follows. In the P2P setting, the edge devices are referred to as *agents*. A central parameter server is not required, and the whole system is more resilient to different issues specific to centralized approaches.

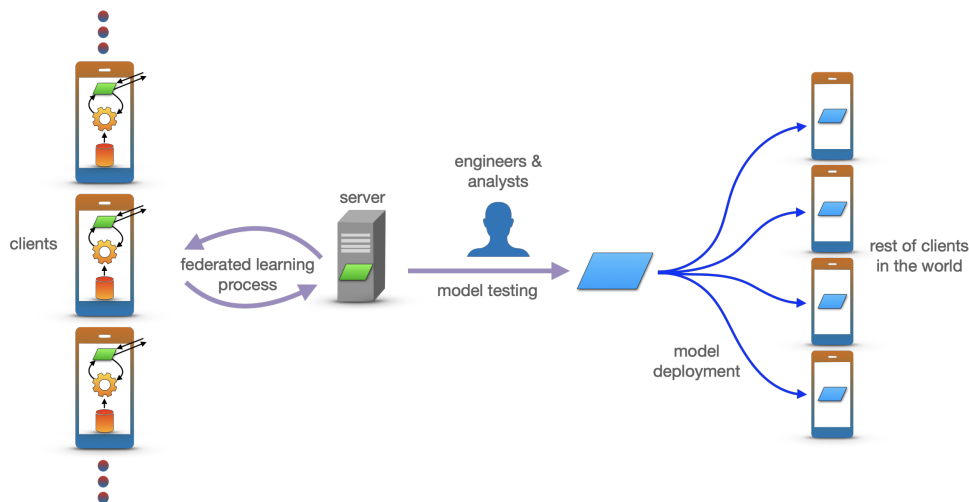


Figure 1: Federated Learning process and the lifecycle of the FL-trained global model. In each training round, the server chooses a subset of the online devices and sends them a copy of the current global model. Clients perform local training steps on the received model before returning the model to the server. This cycle is repeated for a finite number of rounds and, as a result, produces a robust global model. The global model is then deployed as a part of an application and used by the users during regular application usage.



The data on the edge devices is either identically distributed and independent (IID) or non-IID. Heterogeneous or non-IID data is a common scenario in real-world use cases [3, 4]. The data heterogeneity between devices is usually caused by feature and label distribution skew, feature-label mapping differences, and quantity skew [4]. Training models based on the IID assumption is a much more straightforward task compared to the non-IID case. A well-known technique proven to enhance the convergence and precision of machine-learning models is Batch Normalization (BN) [5]. BN is widely used when training deep learning neural network (DNN) models [6–9]. Using BN with linear aggregation techniques, such as those used in FL, will not lead to model convergence. Several studies propose alternative approaches to mitigate this issue [10–13]. However, using BN in a P2P learning setting is not sufficiently studied.

This paper studies the effect of BN on existing decentralized approaches and presents a variant of *gossip averaging* [14, 15] that uses BN layers as a crucial part of the model architecture. Using the BN layers leads to faster agent model convergence, and the proposed early-stopping BN fine-tuning technique further improves the agent’s accuracy. The proposed method was evaluated using two datasets on the next word prediction task [16], the Reddit [17] and the StackOverflow [18] datasets of user comments. We used these datasets to evaluate two characteristic domains: one being a more general discussion forum and the other with a narrow focus on software engineering, demonstrating that the proposed approach applies to both scenarios. Our contributions can be summarized as follows:

- BN fine-tuning - an adaptation of the early-stopping technique for model personalization specific to BN layers;
- P2P-BN - a gossip averaging-based approach compatible with Batch Normalization layers that uses BN fine-tuning to produce stable and consistent models for the next word prediction task, resulting in better accuracy with a significant decrease in the number of exchanged messages;
- evaluation of using Batch Normalization on existing decentralized approaches for improving the model convergence and accuracy on non-IID data.

The rest of this paper is organized as follows. Section 2 provides more details on Batch Normalization and decentralized learning. Section 3 reviews and describes related work. The P2P-BN is presented in Section 4. Section 5 presents the methodology used and the results. The limitations are enumerated in Section 6, and Section 7 concludes the paper.

## 2. Background

This section will introduce the concepts of decentralized learning and the Batch Normalization method used extensively in DNNs.

### 2.1. Decentralized learning

A decentralized approach does not require a central server to orchestrate the learning process. Agents are organized in a peer-to-peer network topology and exchange messages with their neighbors. Agents form connections based on the provided communication graph, such as ring, fully connected, or sparse (see figure 2). Agents can have fixed connections (fixed) or change neighbors through time (time-varying).

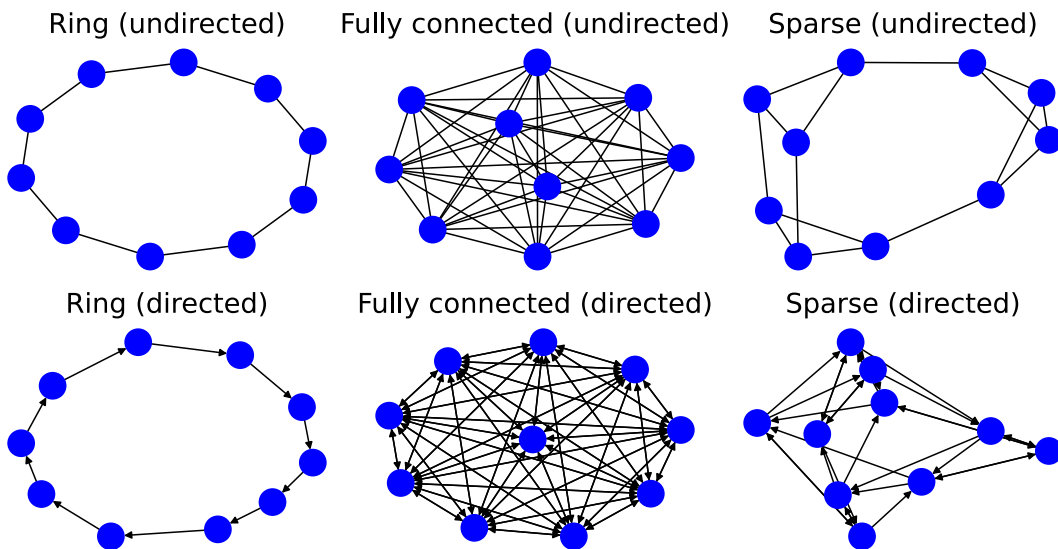


Figure 2: Examples of 10 agents organized in directed/undirected ring, fully connected, and sparse graph topologies. In this example, the sparse graph was formed by connecting each agent with three other agents (each agent has three neighbors).

Communication between agents can be synchronous (sending agent waits for a response from the receiving agent) or asynchronous (sending agent does not wait for a response). The communication is either directed (asymmetric) or undirected (symmetric). In an undirected setting, an agent  $a$  sends to and receives updates from agent  $b$ . In a directed setting, there is no symmetry in model update exchanges. Agent  $a$  sends an update to agent  $b$  but does not neces-

sarily receive updates from  $b$ . Information exchanged between agents is most commonly in the form of model parameters. Depending on the approach, receiving agent averages each received model update with local model parameters or simultaneously averages multiple updates.

## 2.2. Batch Normalization

It has been demonstrated that Batch Normalization (BN) dramatically accelerates the training of DNN models [5]. BN also makes the training process more resilient to large learning rates and prevents the model gradients from exploding or getting stuck in the local minima.

The tensor  $t$  at the input of the BN layer is renormalized as:

$$BN(t) = \gamma \frac{t - \mu}{\sqrt{\sigma^2 + \epsilon}} + \beta \quad (1)$$

where the moving average ( $\mu$ ) and moving variance ( $\sigma^2$ ) are the BN statistics of each channel computed across both spatial and batch dimensions while scaling ( $\gamma$ ) and shifting ( $\beta$ ) are *trainable* parameters where all computations are performed along the channel axis. An arbitrarily small constant  $\epsilon$  is added to the denominator for numerical stability, while an additional *momentum* parameter is used for updating the BN statistics. Let  $B$  denote a mini-batch of size  $m$ ,  $B = \{t_{1\dots m}\}$ , then the moving mean and variance are updated through the training process for each batch as follows:

$$\mu = \mu * momentum + \frac{1}{m} \sum_{i=1}^m t_i * (1 - momentum) \quad (2)$$

$$\sigma^2 = \sigma^2 * momentum + \frac{1}{m} \sum_{i=1}^m (t_i - \mu_B)^2 * (1 - momentum) \quad (3)$$

The contributing factor of BN to overall DNN accuracy was evaluated by training only the BN layers on a randomly initialized model [19, 20]. The end model achieved a surprisingly high test accuracy, suggesting that BN parameters have a remarkable representational capacity. However, Batch Normalization in multi-agent learning is very much an open area of research. It is unclear if the same optimization benefits will occur.

### 3. Related work

This paper focuses on agent collaboration and model training in a completely decentralized or peer-to-peer manner. However, in doing so, some of the approaches and solutions proposed for FL are investigated and adopted as well.

#### 3.1. Decentralized learning approaches

Decentralized approaches differ in communication, model aggregation method, synchronicity, and P2P topology in which they can be applied.

**Communication model.** Under the communication model, both communication symmetry and synchronicity are considered so far. The most lenient is the asynchronous directed communication [14, 15, 21, 22] since the sending agent does not wait for a response. The receiving agent aggregates the received parameters with the local model. Synchronous approaches [23–25] have a synchronization barrier that is passed once all messages between agents are exchanged; thus, a communication direction does not have a significant impact. Undirected asynchronous communication [15, 26–31] implies that a synchronization barrier does not bound the agent training process; however, the communication between two agents is synchronous. When two agents communicate, both must send and receive model updates before continuing the training process.

**Model aggregation.** Model updates are aggregated by averaging received model parameters with a uniform contribution. In *gossip averaging* approaches [14, 15, 21, 22] model updates are averaged with local model parameters upon receiving the update. An agent of a *stochastic gradient push* (SGP) approach [23–25] and *decentralized parallel* approach [26, 32–36] calculates weighted averages of all received model updates including local model parameters. All agents are presumed to have the same impact, so the weights used in averaging are uniform.

**Topologies.** Ring topology is often presumed or only experimented with [26, 33–35]. More robust approaches allow the usage of any topology specified by the communication matrix [15, 23, 24, 32, 36]. Studied *gossip averaging* approaches (apart from [15]) require the knowledge of the whole network, and each agent only exchanges knowledge with one random agent. Sparse

peer-to-peer networks were formed in approaches proposed for training linear models [27–30, 37].

Our approach achieves convergence even in a sparse peer-to-peer network compared to previous approaches. Before computing a new local model update, P2P-BN agent can receive one or more updates. In case of lost messages or unbalanced communication, the learning process of an individual agent is not disturbed. Similar to GoSGD [14], our approach averages the received model upon receiving the update message, unlike approaches such as [32, 33], which require receiving all neighbor messages before continuing the learning process.

### 3.2. Federated learning approach in non-IID setting

FL approaches require relatively homogeneous (IID) data since the training on non-IID data achieves slower convergence with lower accuracy and robustness [38]. Working with non-IID data in FL was studied by applying personalization, meta-learning [39–41], and multi-task [13, 42] learning techniques. *Model personalization* was demonstrated to boost the agents’ model accuracy [43, 44] by additionally training (fine-tuning) the agent’s model after the FL training is completed. BN layers were also used in FL as a form of local personalization, albeit with some adjustments regarding averaging the BN parameters. In FedBN [12] and MTFB [13], each agent’s BN layers are stored locally and are never shared. The BN layers are only used locally by an agent for training and inference but are excluded from regular FL averaging. SiloBN [45] averages all model parameters using the vanilla FL. However, the BN layer’s local statistics moving average ( $\mu$ ) and moving variance ( $\sigma^2$ ) are stored only on agents, and only learned parameters, gamma ( $\gamma$ ) and beta ( $\beta$ ) are averaged.

The solution proposed in this paper adopts the model personalization technique by training only the BN layers. If an agent can not improve the model accuracy, only the BN layers are trained (fine-tuned).

## 4. The P2P-BN method

**Network topology.** We consider a set of  $N$  independent agents in a peer-to-peer network topology training individual models in a shared learning task. Each agent communicates with its peers based on the predefined communication graph  $G = (\llbracket N \rrbracket, E, W)$ , where  $\llbracket N \rrbracket = \{1, \dots, N\}$  is a set of all agents,  $E \in \llbracket N \rrbracket \times \llbracket N \rrbracket$  is the set of edges, and  $W \in \mathbb{R}^{N \times N}$  is a nonnegative weighted matrix. Weight of edge  $(i, j) \in E$  is given by  $W_{ij}$  with the convention  $W_{ij} = 0$  if  $(i, j) \notin E$  or  $i = j$ . Since P2P-BN approach does not consider weighted averaging or trust between agents, we consider that agent  $i$  only sends messages to agent  $j$  if  $W_{ij} > 0$ , which means that agent  $i$  communicates with peers  $N_i = \{j : W_{ij} > 0\}$  without the knowledge of other peers in the network, and operates without synchronisation with non-connected peers ( $W_{ij} = 0$ ). For a directed graph we use the terms in-neighbour if  $(i, j) \in E$  and out-neighbor if  $(j, i) \in E$ . In practice, this presumed communication graph can be replaced by a random peer sampling service (RPS) [46].

**Local model training.** Each agent aims to train its local model with the local dataset  $D_i$  and information received from its peers to minimize its loss function  $F_i$ . Each agent trains a local model by calculating mini-batch gradient  $\nabla F_i(x_i; \xi_i), \xi_i \sim D_i$  and updating its local model  $x_i = x_i - \eta F_i(x_i; \xi_i), \xi_i \sim D_i$  for  $E$  epochs over its local training dataset, where  $\eta$  denotes agent’s personal learning rate. Let  $\| x_i \|$  denote model  $x_i$  with frozen non-BN layers, then the agent performing local personalization on model  $x_i$  is formed as  $x_i = x_i - \eta F_i(\| x_i \|; \xi_i), \xi_i \sim D_i$  for  $E$  epochs. When the agent receives a model  $x_j$  from a peer, a new model  $x_i$  is formed as  $x_i = \frac{x_i + x_j}{2}$ .

**Communication model.** Our proposed approach builds upon a gossip averaging method [15, 47] combined with an additional on-device personalization technique (fine-tuning). Each agent has its local copy of the DNN model  $x_i$  and a local dataset  $D_i$ . Messages exchanged between agents only contain the sending agent’s model parameters. The whole system follows a shared choreography where each agent will: (I) upon joining the network, initialize a model and prepare the local dataset for training, (II) train its model for  $E$  epochs on the local dataset, (III) asynchronously send the local model  $x_i$  to a set of connected peers, (IV) wait for at least one update from other peers and go back to step II. An example of model exchanges between three

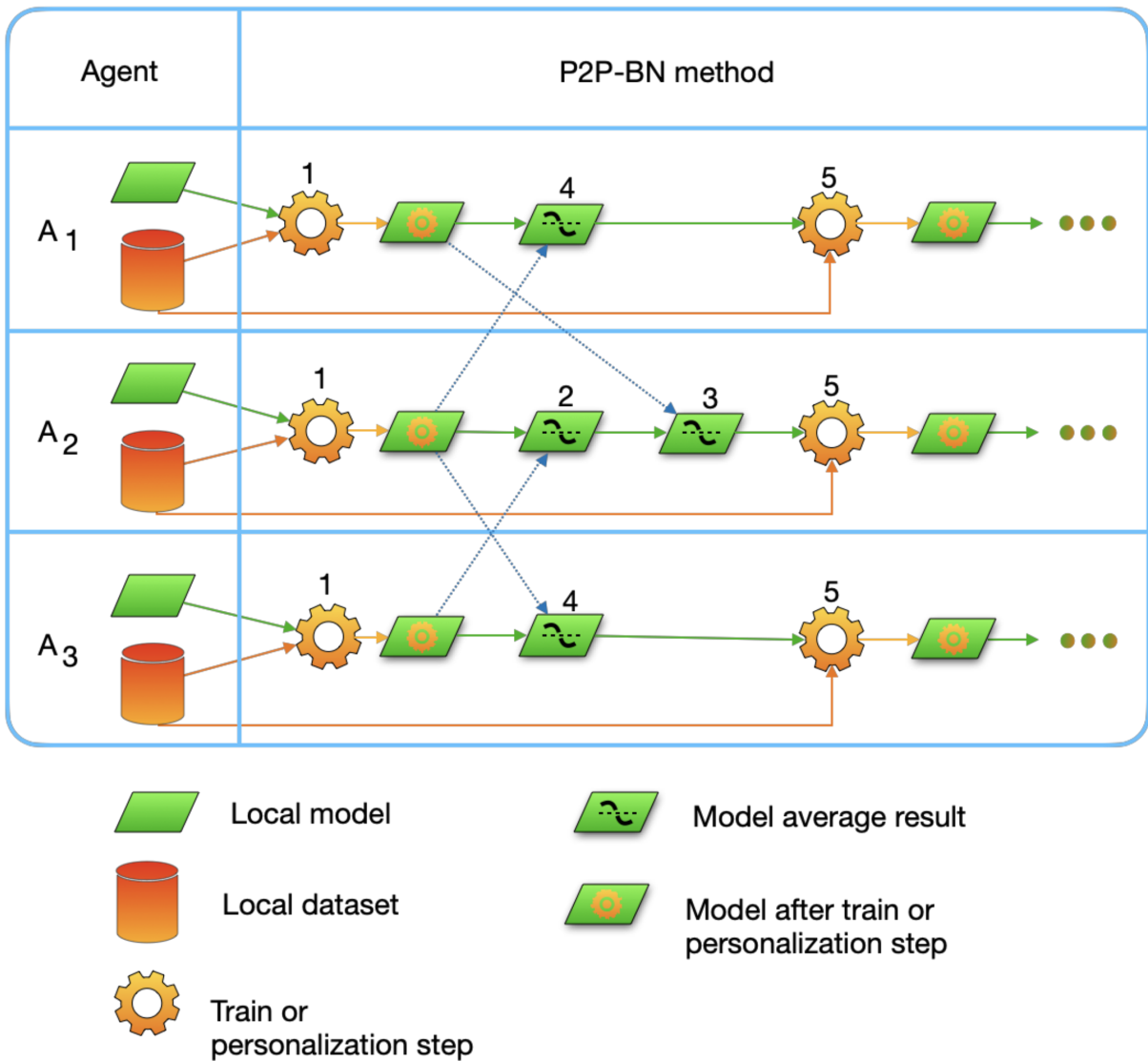


Figure 3: The figure shows the learning process of three agents using the P2P-BN method in an unbalanced topology in which the agent  $A_2$  is neighboring with agents  $A_1$  and  $A_3$ . Imbalanced topology is chosen to demonstrate receiving multiple model updates before continuing the learning process (of the agent  $A_2$ ). At the start of the learning process, all agents first train their model (step 1), followed by the communication step. In the example presented above, the agent  $A_2$  receives models from peers  $A_3$  and  $A_1$  (steps 2 and 3). The agent  $A_2$  then sends the model to its peers, agents  $A_1$  and  $A_3$ , who then average their model with the received model (step 4). Once all messages are exchanged, agents train their model (step 5), and the described process is repeated until the stopping condition is met.

P2P-BN agents is shown in figure 3.

Upon receiving a model from a peer, the agent performs model averaging without sending any information back, which differs from the pairwise aggregation methods [15, 47]. Model averaging will be repeated for every model received. Given the non-IID assumption, agents can differ in the quantity of local data. Agents with scarce data will most likely degrade their models after the local training step due to low representation in training examples. If the agent cannot improve the local model, the agent will freeze all non-BN layers. Freezing the non-BN layers and training only the BN layers is our proposed personalization technique that we refer to as the BN fine-tuning step. The reasoning behind this improvement is the previously-demonstrated remarkable representational capacity of BN layers [20]. Similarly to the early stopping technique [48], a validation dataset is used to determine when the layers need to be frozen/unfrozen.

Pseudo-code shown in algorithm 1 details the global agent behavior as it was performed in the simulation environment:

- at the start of the learning process, all agents initialize their models with identical model parameters,
- in each iteration, one of the agents that received at least one update from the last active state is locally trained,
- when an agent is trained, the agent performs the training step, followed by the model evaluation on validation data,
- if the validation accuracy is not improved, the model freezes non-BN layers, and vice-versa; if the validation accuracy is improved, the model unfreezes all the non-BN layers.

**Model initialization.** Since model averaging is performed, all agents should initialize their models with identical initial values [22].



---

**Algorithm 1** Agents training simulation pseudo-code

---

```
1: function TRAIN( $A, W, E$ )
2:    $A$ : agents
3:    $W$ : communication matrix
4:    $E$ : number of epochs to perform locally
5:   MODEL( $A$ ) = 0           ▷ Initialize all agents models to identical model parameters
6:   repeat
7:      $a_i = \text{RANDOMACTIVEAGENT}(A)$    ▷ Random agent which received at least one
      update from last active state
8:     LOCALTRAIN( $a_i, E$ )
9:     if ISMODELIMPROVED( $a_i$ ) then
10:      UNFREEZEALLLAYERS( $a_i$ )
11:    else
12:      FREEZENONBNLAYERS( $a_i$ )
13:    for  $a_j$  in NEIGHBORS( $W, a_i$ ) do
14:      MODEL( $a_j$ ) = AVERAGEMODEL( $a_j, a_i$ )
15:  until Maximum iteration reached
```

---

## 5. Evaluation

The evaluation of the proposed P2P-BN approach was carried out according to the following research questions:

- **RQ1:** What is the BN personalization effect on P2P-BN performance?
- **RQ2:** How does P2P-BN compare to the baselines in different topologies, with and without BN layers?
- **RQ3:** How does P2P-BN scale according to the number of agents compared to the baseline approaches?

The following subsection will describe the methodology applied to answer these research questions, together with the results and discussion.

### 5.1. Methodology

**Dataset.** The evaluation was performed on Reddit [17] and StackOverflow [18] datasets consisting of unique users and their comments. Each comment can have multiple sentences. In the

experiments, a single agent dataset consisted of all comments from a unique user. Furthermore, agent datasets are divided into training, validation, and test subsets in a 60%-20%-20% split. Following the experiments from previous work [16, 43, 49–51], the vocabulary size was set to 10,000 most common words, which resulted in 10,000 classes on model output. Other words are characterized as an out-of-vocabulary token and are not considered when calculating prediction accuracy. Sentences are broken down into sequences of 10 words, and the model goal is to predict the next word, which gives us a classification problem. Therefore, the context in which the word appears and possible word variations that could also be acceptable solutions are not considered. For example, the sentence "He is walking down the street" can be broken into five sequences, and all sequences need to be of a fixed length (10 words in experiments). Padding is added on the left-hand side of sequences shorter than ten words. Figure 4 shows one of the sequences that could be formed from the sentence to predict the next word. Even though multiple solutions could be considered correct in the given context, only the word "down" is correct.

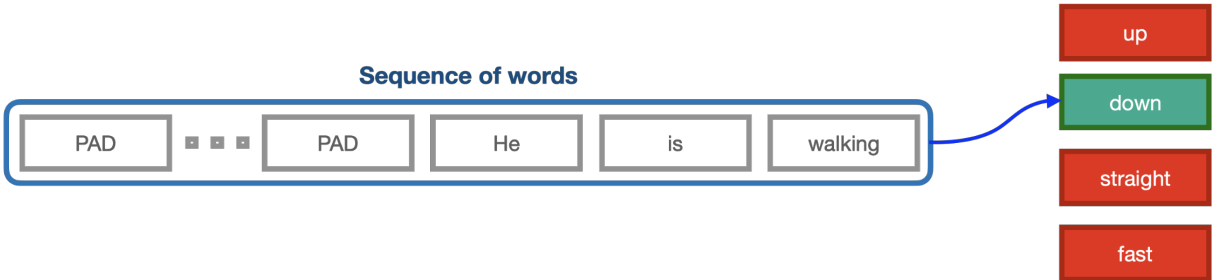


Figure 4: One of the sequences that could be formed from the sentence "He is walking down the street". Only the word "down" is correct, even though multiple words could be considered correct in this context.

Each Reddit agent, on average, trains on 654 examples, with around 80% of agents having less than 700 examples. Following our train-test split, each Reddit agent, on average, has 222 test examples, and around 77% of agents have less than 230 test examples. StackOverflow agents, in general, have a higher number of examples. Each StackOverflow agent has 1134 train, 378 validation, and 378 test examples, with around 74% of agents having less than the average number of train, validation, and test examples.

**Non-IID property.** To evaluate the degree of non-IID property between the agents, we used the Jensen-Shannon divergence (JSD) metric. JSD is a method of measuring the similarity between two probability discrete distributions:

$$JSD(P\|Q) = \frac{1}{2}D_{KL}(P\|M) + \frac{1}{2}D_{KL}(Q\|M), \quad (4)$$

$$M = \frac{1}{2}(P + Q). \quad (5)$$

JSD metric is chosen because it satisfies the three axioms: identity of indiscernibles ( $d(x, y) = 0 \Leftrightarrow x = y$ ), symmetry ( $d(x, y) = d(y, x)$ ), and triangle inequality ( $d(x, y) \leq d(x, z) + d(z, y)$ ) for all  $x, y, z \in X$ . Metrics such as Kullback-Leibler (KL) divergence or Cross-Entropy do not satisfy the symmetry axiom and, therefore, could not be applied for our use case. Suppose we denote the label probability distributions of two training datasets with  $P$  and  $Q$  and express  $M$  as the mid-point of the probability vectors  $P$  and  $Q$ . We can express JSD using the KL divergence  $D_{KL}$  with formula 4. The distance yielded by the metric is located within the  $[0, 1]$  range. Slight divergence (close to 0) means that the distribution of classes in both datasets is very similar. In contrast, significant divergence (close to 1) implies that the distribution of classes is very different.

To express the non-IID degree of the Reddit and StackOverflow datasets, the JSD metric was applied to agents' datasets and compared the results with the well-known MNIST dataset [52] used in numerous previous P2P ML studies [21, 22, 27, 32, 53]. We measured JSD in the following settings: 1) IID data setting [1] that distributes all classes uniformly across different agents; 2) *pathological* non-IID data setting [1] that partitions the dataset so that each agent gets only two classes (out of 10); 3) *practical* non-IID data setting [42] that partitions the data between agents so that every agent has data from all the classes, but with different distributions; some classes have a higher probability than other; 4) the Reddit dataset; 5) the StackOverflow dataset. Distance matrices were sorted before plotting to represent clusters of agents with similar class distributions more accurately. Figure 5 visualizes the JSD distances between agents' datasets for the Reddit, StackOverflow and MNIST datasets. Average JSD results are also given in table 1 and show that the Reddit and StackOverflow dataset heterogeneity levels are between the *practical* and *pathological* non-IID datasets. For the MNIST (IID), the JSD distances between agents are close to zero, resulting in a single cluster of agents. The *pathological* non-IID MNIST matrix displays substantial differences between agents' datasets. However, clusters forming around the diagonal axis can still be observed. In the *practical* non-IID MNIST matrix, it is noticeable that there are indeed differences between agents, but only in the form of five

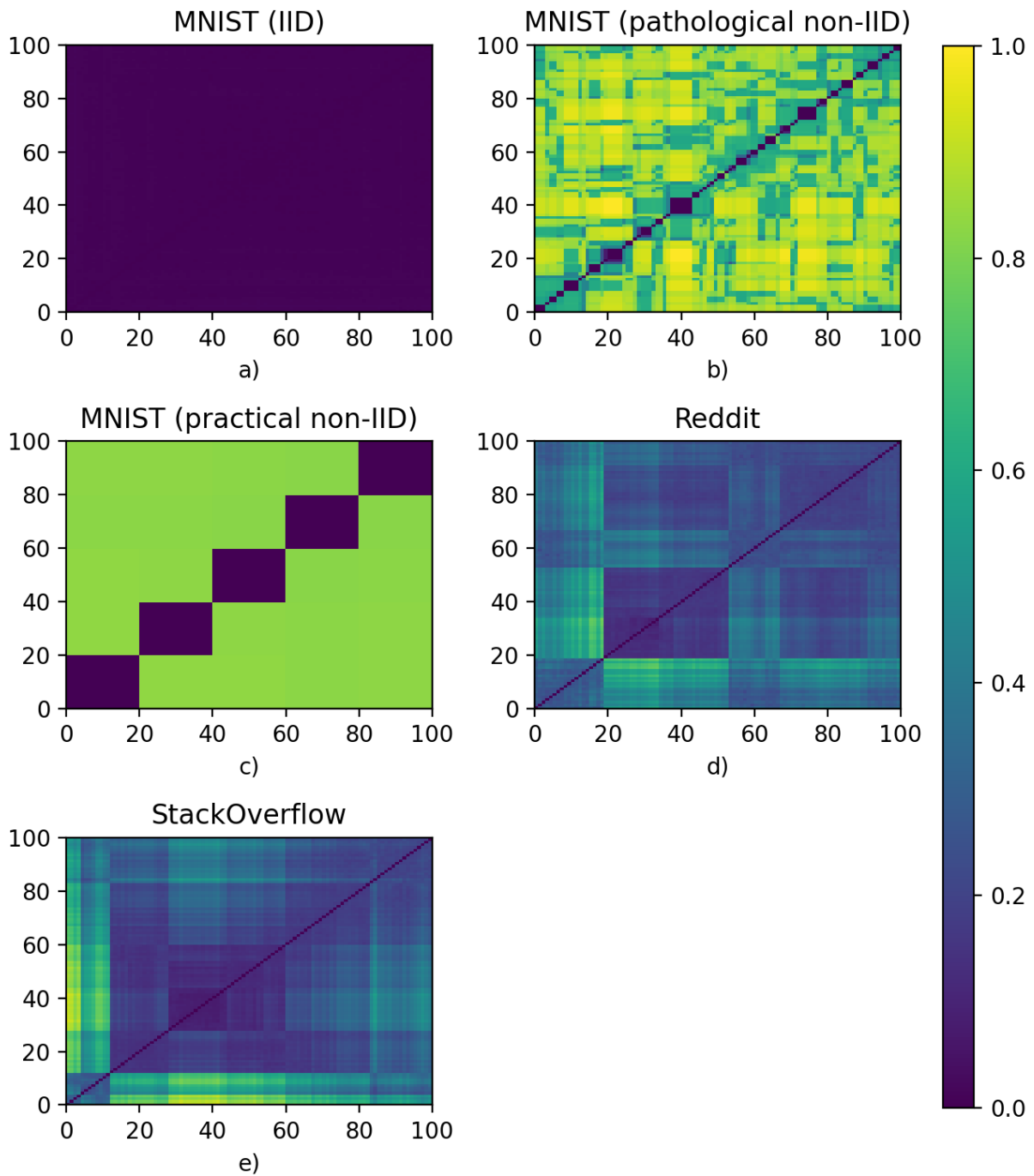


Figure 5: Distance matrices of Jensen-Shannon divergence between agents' local dataset label distribution for MNIST in a: a) IID data setting, b) *pathological* non-IID data setting, and c) *practical* non-IID data setting. All MNIST settings were partitioned into shards of 100 agents, and five groups were created for the *practical* non-IID setting. For the Reddit (d) and StackOverflow (e) datasets, one hundred agents were randomly sampled from the agents used in the experiments. Dark blue represents a substantial similarity between agents' datasets, while bright yellow suggests substantial differences between agents' datasets.

clusters, inside which the inner-similarities between agents are very high (IID). Reddit agents formed numerous visible clusters, while StackOverflow agents formed clusters with lower heterogeneity levels than Reddit agents. The inner-cluster differences of Reddit agents are more considerable, but the differences between clusters are lower compared to *practical* non-IID dataset.

Table 1: Average Jensen-Shannon divergence metric across all agents’ datasets. The MNIST dataset is partitioned into shards of 100 agents for all three settings, and five groups were created for the practical non-IID setting. One hundred Reddit and StackOverflow agents’ datasets were randomly sampled from all agents used in the experiments.

| Dataset                      | Jensen-Shannon divergence |
|------------------------------|---------------------------|
| MNIST (IID)                  | 0.0057                    |
| MNIST (pathological non-IID) | 0.8116                    |
| MNIST (practical non-IID)    | 0.5229                    |
| Reddit                       | 0.6633                    |
| StackOverflow                | 0.5687                    |

**Metrics.** Average User model Accuracy (UA) [13] metric was used to measure the overall learning process performance. UA measures the average accuracy across all agents on their local test data and can be expressed as:

$$UA = \frac{1}{n} \sum_{i=1}^n acc_i \quad (6)$$

where  $acc_i$  is the prediction accuracy of model  $i$  on local test dataset  $i$  (both owned by agent  $i$ ). Prediction accuracy is calculated as the fraction of correct predictions (TP and TN) over total predictions (TP, TN, FP, and FN):

$$acc_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}. \quad (7)$$

**Agent model description.** Model architecture applied on agents is adopted from [50, 51], however, the Long Short-Term Memory (LSTM) [54] layer was replaced with a recurrent neural network layer called Gated Recurrent Unit (GRU) [55]. The adopted model is evaluated with and without BN layers (see table 2), termed as BN and non-BN models.

**Baseline training methods.** We compared our proposed P2P method against several prior methods: (I)  $D^2$  [35] as a representative of the parallel stochastic gradient descent family (e.g., [26, 32, 33]) in which the optimal topology is an undirected ring graph (doubly stochastic symmetric); (II) SGP [56] from the stochastic gradient push family (eg. [23, 24]); (III) GoSGD

Table 2: Model architectures used in the experiments.

| BN model                | Non-BN model    |
|-------------------------|-----------------|
| Embedding (100)         | Embedding (100) |
| GRU (100)               | GRU (100)       |
| BatchNormalization(200) | -               |
| Dense (100)             | Dense (100)     |
| BatchNormalization(200) | -               |
| Dense (10,002)          | Dense (10,002)  |

[14] from decentralized gossip exchange approaches (eg. [21]). Approaches  $D^2$  and SGP are simulated by synchronously training all agents using a *uniformly mixing communication matrix*. A uniformly mixed communication matrix implies that all received models from peers are equally important. GoSGD is also simulated by synchronously training all agents. In all compared methods, agents first perform a local training step, followed by a communication step, repeating these two steps in a loop (see figure 6). We modified the communication behavior of the GoSGD approach so that the agent chooses its neighbor(s) based on the tested topology. The probability of sending a model to each peer is set to 0.1 in all experiments. We simulate the P2P-BN approach by randomly and uniformly choosing one of the eligible agents. Eligibility is met by the criteria given in algorithm 1: the agent has received at least one model update since the last training step. The chosen agent then performs the local training step and communication step.

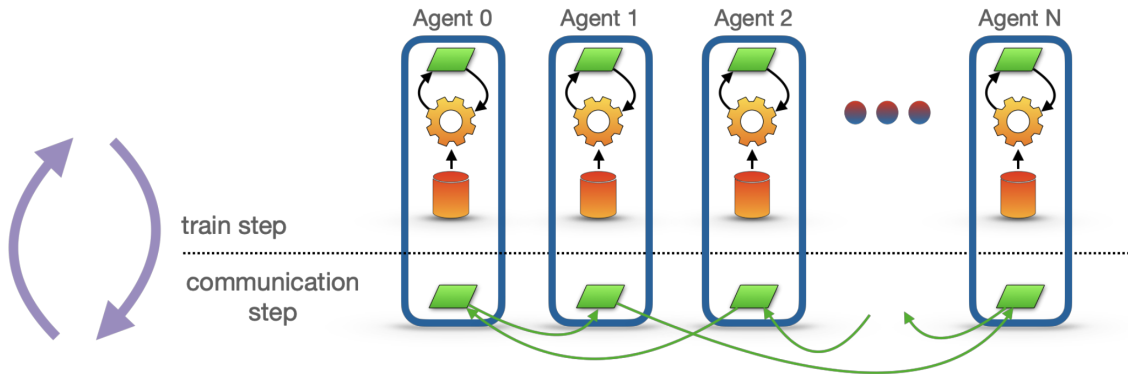


Figure 6: Synchronous agent training, in which all agents first perform a local training step, followed by a communication step, repeating these two steps in a loop.

## 5.2. Experiments

For all simulated approaches, the batch size was set to 50, and the learning rate to  $5 \times 10^{-3}$ . Depending on the experiment, the BN and the non-BN model was used. Unless differently stated, the BN momentum was set to 0.9, and parameters for scaling ( $\gamma$ ) and shifting ( $\beta$ ) were ignored (not trained). In the experiments, five separate runs were conducted for each approach-topology-communication combination, with a new set of agents for each run.

An ablation study was performed first to understand the proposed BN fine-tuning technique. The BN fine-tuning technique was then used in all following experiments with P2P-BN. The following two experiments analyze performances in the ring and sparse topologies and investigate the scaling properties of both P2P-BN and baseline methods.

**BN fine-tuning ablation study.** To answer **RQ1** and demonstrate the impact of the proposed BN fine-tuning on the overall learning process, we conducted experiments on P2P-BN with and without the BN fine-tuning technique, noted as *test* and *control*, respectively. The BN model is used in all simulations, and the number of agents involved was 100 and 300, connected by a directed/undirected sparse graph with three neighbors. Figure 7 shows that using BN fine-tuning improves agents’ learning ability in 100 and 300 agent settings, regardless of whether the communication is directed or undirected.

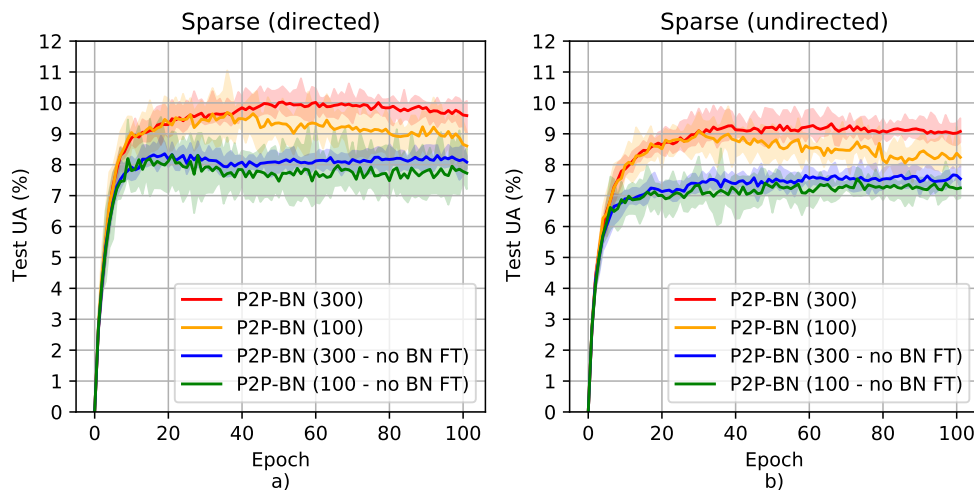


Figure 7: Test UA of 100 and 300 agents for P2P-BN in directed (a) and undirected (b) fixed sparse graph with or without the BN fine-tuning technique for the Reddit dataset.

The maximum achieved UA of each run was selected for both test and control experiments and analyzed using the Student t-test. The Student t-test was performed to statistically con-

firm the difference between the UA achieved in the test and control. For both 100 and 300 agent groups, the p-value was  $p < 2 \times 10^{-2}$ , suggesting statistical significance. Results are summarized in Table 3.

Table 3: Maximum UA obtained with and without the BN fine-tuning technique for the 100 and 300 agent groups in a directed and undirected sparse topology. Training with the BN fine-tuning (BN-TF) technique achieved a relative top UA increase of around 20% for the Reddit dataset and around 15% for the StackOverflow dataset.

| <b>Dataset</b>       |    |       |   |   |
|----------------------|----|-------|---|---|
| #                    | of | BN-FT | Directed                                | Undirected                              |
| agents               |    |       |   |   |
| <b>Reddit</b>        |    |       |   |   |
| 100                  |    | No    | 8.32                                    | 7.55                                    |
| 100                  |    | Yes   | 9.69 (+16.4%, $p < 3 \times 10^{-3}$ )  | 9.10 (+20.5%, $p < 5 \times 10^{-4}$ )  |
| 300                  |    | No    | 8.35                                    | 7.67                                    |
| 300                  |    | Yes   | 10.03 (+20.1%, $p < 7 \times 10^{-6}$ ) | 9.33 (+21.6%, $p < 1 \times 10^{-5}$ )  |
| <b>StackOverflow</b> |    |       |   |   |
| 100                  |    | No    | 10.89                                   | 9.90                                    |
| 100                  |    | Yes   | 12.49 (+14.7%, $p < 2 \times 10^{-2}$ ) | 11.82 (+19.4%, $p < 5 \times 10^{-5}$ ) |
| 300                  |    | No    | 11.92                                   | 10.29                                   |
| 300                  |    | Yes   | 13.16 (+10.4%, $p < 2 \times 10^{-2}$ ) | 11.92 (+15.8%, $p < 5 \times 10^{-5}$ ) |

**Batch Normalization effect on baseline approaches.** To study the effect of using the BN model in all approaches and answer **RQ2**, we simulated the P2P training process of 100 agents in the ring and sparse topologies. The BN and the non-BN model were used on D<sup>2</sup>, GoSGD, (PushSum) SGP, and P2P-BN approaches in a directed and undirected ring topology (see Figure 2) for 100 agent epochs. Note that when the non-BN model was used in the P2P-BN method, BN fine-tuning was omitted. All experiments start with identical agent model parameters and identical agent peer connections. Additionally, P2P-BN was evaluated and compared to the baseline approaches on both undirected and directed, fixed and time-varying sparse network topologies. The number of neighbors was set to three to prevent agent separation into groups. The communication matrix was altered every five epochs for the time-varying network scenario. Figure 8 shows graph topology used in fixed communication setting. In the time-varying setting, similar graphs were produced but with different peer connections.

**Ring topology.** Ring topology setting demonstrated that all approaches benefit by using the BN model regarding convergence speed and maximum UA (see figure 9). Directed graphs also demonstrated slightly better accuracies than undirected versions. P2P-BN achieved superior convergence speed and maximum UA compared to any baseline approach and benefited the most from the directed graph in the BN setting. In the BN model experiments, compared to the



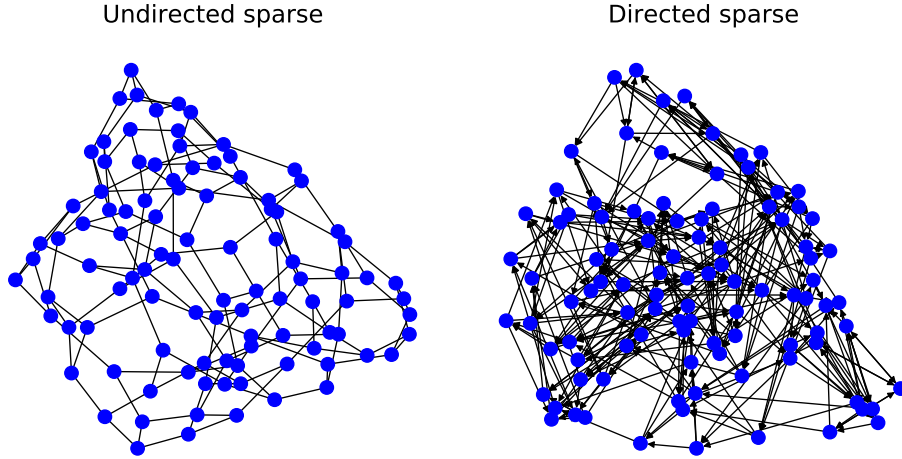


Figure 8: Fixed sparse undirected and directed graph used for the evaluation, comprising of 100 agents with three in- and out-neighbors.

closest second-ranked GoSGD, P2P-BN achieved a mean relative top UA increase of 19.9% for the Reddit dataset and 13.9% for the StackOverflow dataset. Table 4 summarizes the results. For all approaches, BN layers enabled up to four times faster convergence.

Table 4: The average number of exchanged messages per epoch and the average top UA for the Reddit and StackOverflow datasets. Averages are calculated by averaging data from all runs conducted in the ring topology experiments using the BN model.

| Dataset       | Approach       | # of messages | Test UA      |
|---------------|----------------|---------------|--------------|
| Reddit        |                |               |              |
|               | D <sup>2</sup> | 2272          | 6.47%        |
|               | GoSGD          | 226           | 6.57%        |
|               | SGP            | 2327          | 6.38%        |
|               | <b>P2P-BN</b>  | <b>149</b>    | <b>7.88%</b> |
| StackOverflow |                |               |              |
|               | D <sup>2</sup> | 3640          | 8.24%        |
|               | GoSGD          | 362           | 8.67%        |
|               | SGP            | 3642          | 8.20%        |
|               | <b>P2P-BN</b>  | <b>148</b>    | <b>9.87%</b> |

**Sparse topology.** Figure 10 shows the simulation results in the sparse topologies. D<sup>2</sup> and SGP perform similarly, producing similar results in both directed and undirected communication settings. GoSGD demonstrates a slight increase in UA, and P2P-BN significantly outperforms all other approaches, producing the best results in a directed communication setting. Table 5 summarizes the average number of exchanged messages per epoch and averaged maximum accuracies across all runs in this experiment. In sparse topology experiments, P2P-BN, compared to the closest second-ranked GoSGD, achieved a mean relative top UA increase of

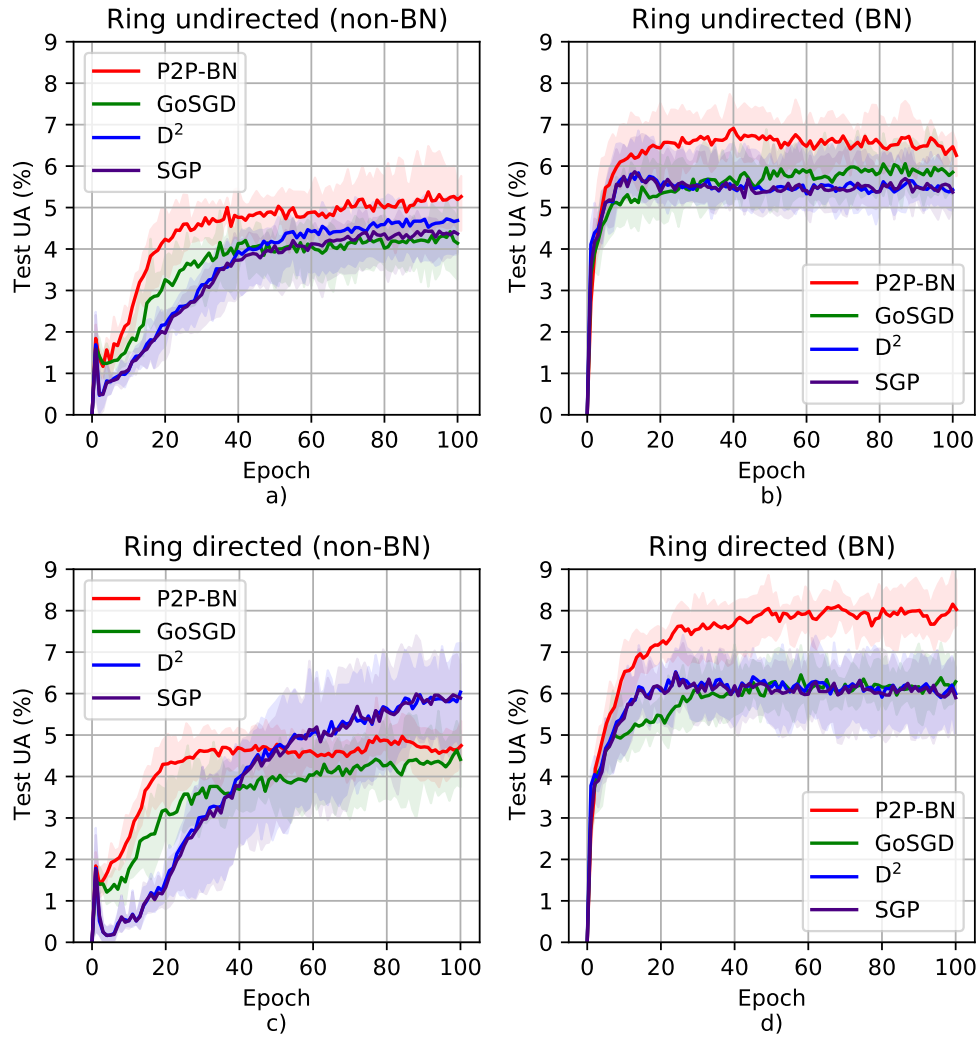


Figure 9: Test UA of 100 agents for  $D^2$ , GoSGD, SGD, and P2P-BN in: undirected ring topology using the non-BN (a) and the BN (b) model; directed ring topology using the non-BN (c) and the BN (d) model for the Reddit dataset.

32.9% for the Reddit dataset and 26.6% for the StackOverflow dataset.

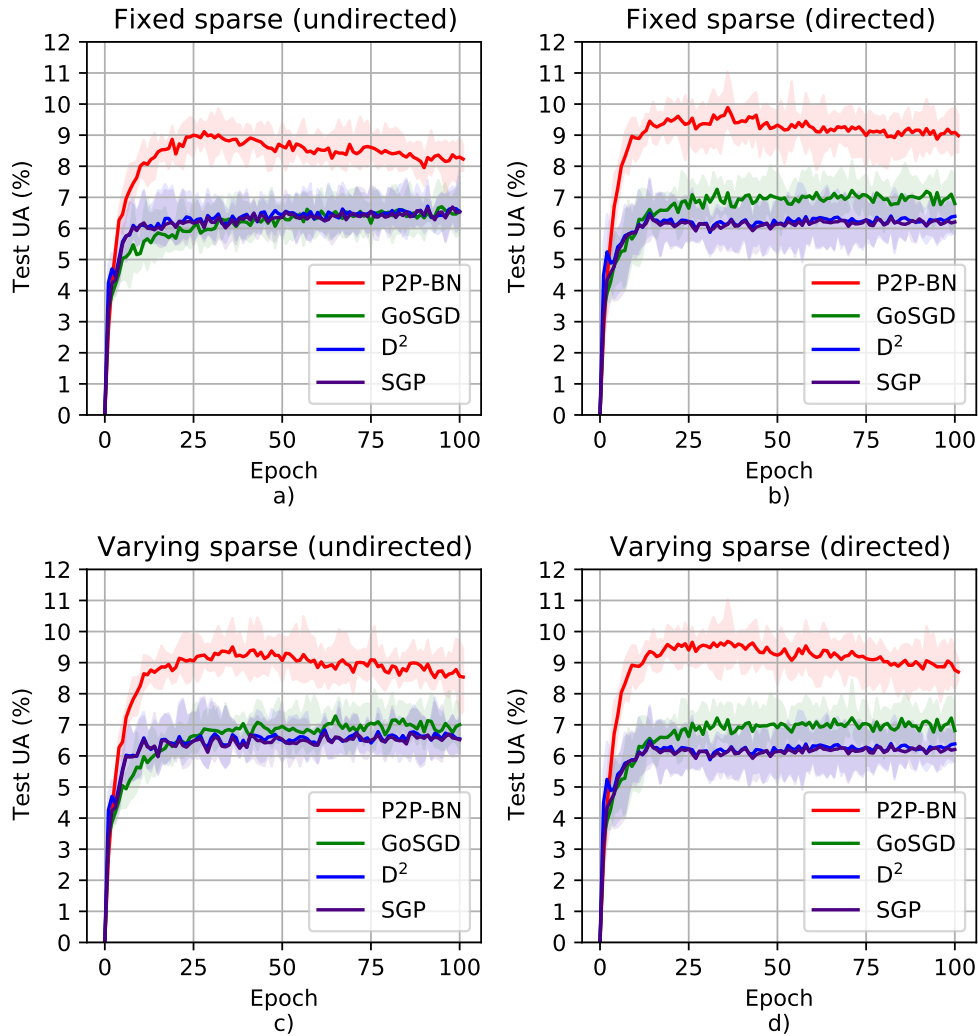


Figure 10: Test UA of 100 agents for D<sup>2</sup>, GoSGD, SGP, and P2P-BN in: fixed undirected (a) and directed (b) sparse topology; varying undirected (c) and directed (d) sparse topology for the Reddit dataset.

Finally, we also performed a training variant where agents did not exchange models. On average, agents reached a maximum UA of 5% using the Reddit dataset and 6% using the StackOverflow dataset. When training one single model on all training data pooled, the model, on average, achieved a maximum accuracy of 12.8% using the Reddit dataset and 15.5% using the StackOverflow dataset.

**Evaluating scaling properties.** To investigate **RQ3**, the scaling properties of P2P-BN, simulations with 100 and 500 agents were conducted. An additional simulation was carried out for P2P-BN in a 1000 agent setting. The goal of scaling evaluation was to study whether increasing the number of agents will hamper individual agents’ learning process, i.e., will the network take

Table 5: The average number of exchanged messages per epoch and average top UA for the Reddit and StackOverflow datasets. Averages are calculated by averaging data from all runs conducted in the sparse topology experiments.

| Dataset       | Approach       | # of messages | Test UA       |
|---------------|----------------|---------------|---------------|
| Reddit        |                |               |               |
|               | D <sup>2</sup> | 4544          | 6.96%         |
|               | GoSGD          | 452           | 7.47%         |
|               | SGP            | 4547          | 6.89%         |
|               | <b>P2P-BN</b>  | <b>298</b>    | <b>9.93%</b>  |
| StackOverflow |                |               |               |
|               | D <sup>2</sup> | 7281          | 9.10%         |
|               | GoSGD          | 725           | 9.58%         |
|               | SGP            | 7284          | 9.07%         |
|               | <b>P2P-BN</b>  | <b>299</b>    | <b>12.13%</b> |

more time to synchronize. A fixed directed sparse topology was used for all experiments with three in- and out-neighbors. In this simulations, the momentum of BN layers was increased exponentially, using the formula:  $\min(0.9 * 1.01^{(epoch/5)}, 0.99)$ , where *epoch* is the number of local epochs performed by an agent. Increasing momentum allows each agent a more expansive optimal function search space in the early stages of the learning. However, it narrows this search space every time an agent trains its local model for an epoch. The optimal strategy for increasing momentum should be studied in future research and applied individually rather than uniformly across all agents. As figure 11 shows, D<sup>2</sup> and SGP do not show any significant increase in UA, while GoSGD achieves a slightly better UA with 500 agents. P2P-BN agents benefit the most from a bigger network size while keeping the number of messages constant due to a fixed number of neighbors in all simulations. Table 6 summarises the average top UA achieved in the 100 and 500 agent simulations. The Student t-test shows no statistically significant differences in average top UA between the 100 and 500 agent simulations for D<sup>2</sup>, GoSGD, and SGP ( $p > 0.29$ ).

## 6. Limitations

Further studies are needed to evaluate P2P-BN on more IID and non-IID datasets and different model architectures. Although P2P-BN performs well on non-IID data, some limitations

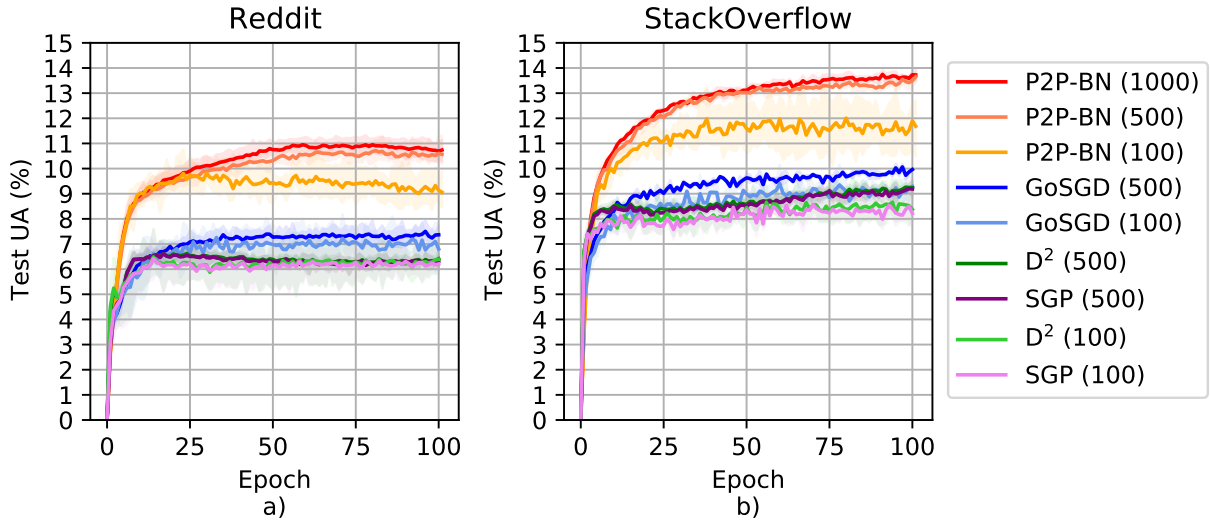


Figure 11: Test UA with varying agent count for  $D^2$ , GoSGD, SGP, and P2P-BN on a fixed directed sparse topology with three in- and out-neighbors using the BN model for the Reddit and StackOverflow datasets.

Table 6: Average top accuracies for  $D^2$ , GoSGD, SGP and P2P-BN in the 100 and 500 agent setting on a directed sparse topology for the Reddit (a) and StackOverflow (b) datasets. P values obtained through the Student t-test suggest that only P2P-BN displays a statistically significant difference between the 100 and 500 agent simulations.

| Dataset       | Approach | Test UA of 100 agents | Test UA of 500 agents         |
|---------------|----------|-----------------------|-------------------------------|
| Reddit        |          |                       |                               |
|               | $D^2$    | 6.49                  | 6.62 (+2.0%, $p > 0.72$ )     |
|               | GoSGD    | 7.26                  | 7.50 (+3.31%, $p > 0.91$ )    |
|               | SGP      | 6.42                  | 6.57 (+2.34%, $p > 0.91$ )    |
|               | P2P-BN   | 9.82                  | 10.72 (+9.16%, $p < 0.035$ )  |
| StackOverflow |          |                       |                               |
|               | $D^2$    | 8.67                  | 9.26 (+6.81%, $p > 0.55$ )    |
|               | GoSGD    | 9.40                  | 10.07 (+7.13%, $p > 0.44$ )   |
|               | SGP      | 8.56                  | 9.20 (+7.48%, $p > 0.29$ )    |
|               | P2P-BN   | 12.01                 | 13.65 (+13.66%, $p < 0.084$ ) |

should be addressed. Due to the proposed fine-tuning method, our approach requires BN layers in agent models. Furthermore, P2P-BN currently only supports training on homogeneous models and does not address the possibility of malicious agents looking to degrade peers' models. A way towards tackling malicious agents could be based on Byzantine resilience [15]. Supporting heterogeneous models could be solved by using knowledge transfer techniques from FL approaches [57, 58]. Standard differential privacy techniques could be a good tool for ensuring an agent's privacy, with an additional study on trade-offs between privacy and model accuracy/knowledge. Besides investigating the need for heterogeneous models, there is a need to address the possible performance inequalities between the agents.

## 7. Conclusion

In this paper, we considered the problem of decentralized learning on a next-word prediction task given non-IID data over a network of connected agents in different topologies. We introduced and validated a novel asynchronous variant of the gossip averaging approach, P2P-BN, and demonstrated the effectiveness of BN fine-tuning in normalizing non-IID data trained models across agents. The BN fine-tuning technique demonstrated substantial performance gain when applied in P2P-BN compared to the baseline approaches. The evaluation was performed on a rich set of simulations in different static and dynamic topologies. For the next word prediction task, simulations showed that P2P-BN, on average, achieves a mean relative top accuracy increase of 16.9% in ring (19.9% for Reddit, 13.9% for StackOverflow) and 29.8% in sparse (32.9% for Reddit, 26.6% for StackOverflow) communication topologies compared to the best baseline approach. Future research will focus on establishing connections between agents, specifically, studying whether grouping agents based on their common interests and data similarity is beneficial or detrimental to their learning ability. Furthermore, it would be interesting to investigate the trade-offs between local computation and communication to find the optimal agent configuration parameters for P2P-BN. The additional research direction also includes evaluating more realistic scenarios: dynamic agent arrivals and churn levels, model heterogeneity, and how to efficiently use new data points acquired by the agents.

## References

- [1] Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 1273–1282. URL: <https://proceedings.mlr.press/v54/mcmahan17a.html>.
- [2] Michael Blot et al. “Gossip training for deep learning”. In: (Nov. 2016).
- [3] Rong Yu and Peichun Li. “Toward Resource-Efficient Federated Learning in Mobile Edge Computing”. In: *IEEE Network* 35.1 (2021), pp. 148–155. DOI: 10.1109/MNET.011.2000295.
- [4] Xiaodong Ma et al. “A state-of-the-art survey on solving non-IID data in Federated Learning”. In: *Future Generation Computer Systems* 135 (2022), pp. 244–258. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2022.05.003>. URL: <https://www.sciencedirect.com/science/article/pii/S0167739X22001686>.
- [5] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning*. Vol. 37. ICML’15. Lille, France: JMLR.org, 2015, pp. 448–456.
- [6] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: 10.1109/CVPR.2016.90.
- [7] Jie Hu, Li Shen, and Gang Sun. “Squeeze-and-Excitation Networks”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2018, pp. 7132–7141. DOI: 10.1109/CVPR.2018.00745.
- [8] Mingxing Tan and Quoc Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings

- of Machine Learning Research. PMLR, June 2019, pp. 6105–6114. URL: <https://proceedings.mlr.press/v97/tan19a.html>.
- [9] Asher Trockman and J Zico Kolter. *Patches Are All You Need?* 2022. URL: <https://openreview.net/forum?id=TVHS5Y4dNvM>.
- [10] Pramod Kaushik Mudrakarta et al. “K For The Price Of 1: Parameter Efficient Multi-task And Transfer Learning”. In: *International Conference on Learning Representations*. Vol. abs/1810.10703. 2019, pp. 1–15.
- [11] Cong Xie, Oluwasanmi Koyejo, and Indranil Gupta. “Asynchronous Federated Optimization”. In: *ArXiv abs/1903.03934* (2019). URL: <http://arxiv.org/abs/1903.03934>.
- [12] Xiaoxiao Li et al. “FedBN: Federated Learning on Non-IID Features via Local Batch Normalization”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.
- [13] J. Mills, J. Hu, and G. Min. “Multi-Task Federated Learning for Personalised Deep Neural Networks in Edge Computing”. In: *IEEE Transactions on Parallel and Distributed Systems* 33.03 (Mar. 2022), pp. 630–641. ISSN: 1558-2183. DOI: 10.1109/TPDS.2021.3098467.
- [14] Michael Blot et al. “Distributed optimization for deep learning with gossip exchange”. In: *Neurocomputing* 330 (2019), pp. 287–296. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2018.11.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231218313195>.
- [15] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “Robust Privacy-Preserving Gossip Averaging”. In: *SSS 2019 - 21st International Symposium on Stabilization, Safety, and Security of Distributed Systems*. Pisa, Italy: Springer, Oct. 2019, pp. 38–52. DOI: 10.1007/978-3-030-34992-9\_4. URL: <https://hal.archives-ouvertes.fr/hal-02373353>.
- [16] Andrew Hard et al. “Federated Learning for Mobile Keyboard Prediction”. In: *CoRR abs/1811.03604* (2018). URL: <https://arxiv.org/abs/1811.03604>.
- [17] Sebastian Caldas et al. “LEAF: A Benchmark for Federated Settings”. In: *ArXiv abs/1812.01097* (2018).



- [18] StackOverflow. *Stack Overflow Data*. Kaggle. 2018. URL: <https://www.kaggle.com/datasets/stackoverflow/stackoverflow>.
- [19] Amir Rosenfeld and John K. Tsotsos. “Intriguing Properties of Randomly Weighted Networks: Generalizing While Learning Next to Nothing”. In: *2019 16th Conference on Computer and Robot Vision (CRV) (2019)*, pp. 9–16. DOI: 10.1109/CRV.2019.00010.
- [20] Jonathan Frankle, David J. Schwab, and Ari S. Morcos. “Training BatchNorm and Only BatchNorm: On the Expressive Power of Random Features in CNNs”. In: *International Conference on Learning Representations*. 2021.
- [21] Jeff A. Daily et al. “GossipGraD: Scalable Deep Learning using Gossip Communication based Asynchronous Gradient Descent”. In: *ArXiv abs/1803.05880 (2018)*.
- [22] Robert Šajina, Nikola Tanković, and Darko Etinger. “Decentralized trustless gossip training of deep neural networks”. In: *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. 2020, pp. 1080–1084. DOI: 10.23919/MIPRO48935.2020.9245248.
- [23] Mahmoud Assran et al. “Stochastic Gradient Push for Distributed Deep Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 344–353. URL: <https://proceedings.mlr.press/v97/assran19a.html>.
- [24] Chaoyang He et al. “Central Server Free Federated Learning over Single-sided Trust Social Networks”. In: *ArXiv abs/1910.04956 (2019)*. URL: <http://arxiv.org/abs/1910.04956>.
- [25] Anusha Lalitha et al. “Peer-to-peer Federated Learning on Graphs”. In: *ArXiv abs/1901.11173 (2019)*. URL: <http://arxiv.org/abs/1901.11173>.
- [26] Xiangru Lian et al. “Asynchronous Decentralized Parallel Stochastic Gradient Descent”. In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, July 2018, pp. 3043–3052. URL: <https://proceedings.mlr.press/v80/lian18a.html>.

- [27] Karim Boubouh et al. “Robust P2P Personalized Learning”. In: *2020 International Symposium on Reliable Distributed Systems (SRDS)*. 2020, pp. 299–308. DOI: 10.1109/SRDS51746.2020.00037.
- [28] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. “Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 864–874. URL: <https://proceedings.mlr.press/v108/zantedeschi20a.html>.
- [29] Inês Almeida and João Xavier. “DJAM: Distributed Jacobi Asynchronous Method for Learning Personal Models”. In: *IEEE Signal Processing Letters* 25.9 (2018), pp. 1389–1392. DOI: 10.1109/LSP.2018.2859596.
- [30] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. “Decentralized Collaborative Learning of Personalized Models over Networks”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 509–517. URL: <https://proceedings.mlr.press/v54/vanhaesebrouck17a.html>.
- [31] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “Simple, Efficient and Convenient Decentralized Multi-Task Learning for Neural Networks”. In: *IDA 2021 - 19th Symposium on Intelligent Data Analysis*. Vol. 12695. Lecture Notes in Computer Science. Porto, Portugal: Springer, Apr. 2021. DOI: 10.1007/978-3-030-74251-5\_4. URL: <https://hal.archives-ouvertes.fr/hal-02373338>.
- [32] Shangwei Guo et al. “Differentially Private Decentralized Learning”. In: *ArXivCoRR* (June 2020). URL: <http://arxiv.org/abs/2006.07817>.
- [33] Xiangru Lian et al. “Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/f75526659f31040afeb61cb7133e4e6d-Paper.pdf>.

- [34] Qinyi Luo et al. “Prague: High-Performance Heterogeneity-Aware Asynchronous Decentralized Training”. In: *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*. New York, NY, USA: Association for Computing Machinery, 2020, pp. 401–416. ISBN: 9781450371025. URL: <https://doi.org/10.1145/3373376.3378499>.
- [35] Hanlin Tang et al. “D<sup>2</sup>: Decentralized Training over Decentralized Data”. In: *Proceedings of the 35th International Conference on Machine Learning* 80.1 (2018), pp. 4848–4856. URL: <http://proceedings.mlr.press/v80/tang18a.html>.
- [36] Jianyu Wang and Gauri Joshi. “Cooperative SGD: A Unified Framework for the Design and Analysis of Local-Update SGD Algorithms”. In: *Journal of Machine Learning Research* 22.213 (2021), pp. 1–50. URL: <http://jmlr.org/papers/v22/20-147.html>.
- [37] Aurélien Bellet et al. “Personalized and Private Peer-to-Peer Machine Learning”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, Apr. 2018, pp. 473–481. URL: <https://proceedings.mlr.press/v84/bellet18a.html>.
- [38] Xiang Li et al. “On the Convergence of FedAvg on Non-IID Data”. In: *International Conference on Learning Representations*. 2020.
- [39] Manoj Ghuhana Arivazhagan et al. “Federated Learning with Personalization Layers”. In: *ArXiv* abs/1912.00818 (2019).
- [40] Yihan Jiang et al. “Improving Federated Learning Personalization via Model Agnostic Meta Learning”. In: *CoRR* (2020). URL: <http://arxiv.org/abs/1909.12488>.
- [41] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. “Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 3557–3568. URL: <https://proceedings.neurips.cc/paper/2020/file/24389bfe4fe2eba8bf9aa9203a44cdad-Paper.pdf>.

- [42] Yutao Huang et al. “Personalized Federated Learning: An Attentive Collaboration Approach”. In: *CoRR* abs/2007.03797 (2020). URL: <https://arxiv.org/abs/2007.03797>.
- [43] Kangkang Wang et al. “Federated Evaluation of On-device Personalization”. In: *ArXiv* abs/1910.10252 (2019). URL: <http://arxiv.org/abs/1910.10252>.
- [44] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. *Adaptive Personalized Federated Learning*. 2021.
- [45] Mathieu Andreux et al. “Siloed Federated Learning for Multi-Centric Histopathology Datasets”. In: Lima, Peru: Springer-Verlag, 2020, pp. 129–139. ISBN: 978-3-030-60547-6. DOI: 10.1007/978-3-030-60548-3\_13. URL: [https://doi.org/10.1007/978-3-030-60548-3\\_13](https://doi.org/10.1007/978-3-030-60548-3_13).
- [46] Márk Jelasity et al. “Gossip-Based Peer Sampling”. In: *ACM Trans. Comput. Syst.* 25.3 (Aug. 2007), 8–es. ISSN: 0734-2071. DOI: 10.1145/1275517.1275520. URL: <https://doi.org/10.1145/1275517.1275520>.
- [47] Márk Jelasity, Alberto Montresor, and Özalp Babaoglu. “Gossip-based aggregation in large dynamic networks”. In: *ACM Trans. Comput. Syst.* 23 (2005), pp. 219–252.
- [48] Nelson Morgan and Hervé Bouchard. “Generalization and Parameter Estimation in Feed-forward Nets: Some Experiments”. In: *Advances in Neural Information Processing Systems*. Ed. by D. Touretzky. Vol. 2. Morgan-Kaufmann, 1989. URL: <https://proceedings.neurips.cc/paper/1989/file/63923f49e5241343aa7acb6a06a751e7-Paper.pdf>.
- [49] Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic Controlled Averaging for Federated Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 5132–5143. URL: <https://proceedings.mlr.press/v119/karimireddy20a.html>.
- [50] Sashank J. Reddi et al. “Adaptive Federated Optimization”. In: *International Conference on Learning Representations*. 2021.

- [51] Joel Stremmel and Arjun Singh. “Pretraining Federated Text Models for Next Word Prediction”. In: *Advances in Information and Communication*. Ed. by Kohei Arai. Springer International Publishing, 2021, pp. 477–488. ISBN: 978-3-030-73103-8.
- [52] Li Deng. “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.
- [53] Hsin-Pai Cheng et al. “LEASGD: an Efficient and Privacy-Preserving Decentralized Algorithm for Distributed Learning”. In: *Proceedings of NIPS Workshop on Privacy Preserving Machine Learning*. 2018.
- [54] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. URL: <https://doi.org/10.1162/neco.1997.9.8.1735>.
- [55] Kyunghyun Cho et al. “Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation”. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing EMNLP*. Doha, Qatar: Association for Computational Linguistics, Oct. 2014, pp. 1724–1734. DOI: 10.3115/v1/d14-1179. URL: <https://aclanthology.org/D14-1179>.
- [56] Angelia Nedić and Alex Olshevsky. “Stochastic Gradient-Push for Strongly Convex Functions on Time-Varying Directed Graphs”. In: *IEEE Transactions on Automatic Control* 61.12 (2016), pp. 3936–3947. DOI: 10.1109/TAC.2016.2529285.
- [57] Hongyan Chang et al. “Cronus: Robust and Heterogeneous Collaborative Learning with Black-Box Knowledge Transfer”. In: *New Frontiers in Federated Learning: Privacy, Fairness, Robustness, Personalization and Data Ownership* (2021). URL: <http://arxiv.org/abs/1912.11279>.
- [58] Daliang Li and Junpu Wang. “FedMD: Heterogenous Federated Learning via Model Distillation”. In: *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with NeurIPS* (2019).

## **B. Multi-task peer-to-peer learning using an encoder-only transformer model**

This work was published as: Robert Šajina, Nikola Tanković, Ivo Ipšić, 2024. Multi-task peer-to-peer learning using an encoder-only transformer model. *Future Generation Computer Systems*, Volume 152, ISSN 0167-739X, DOI: 10.1016/j.future.2023.11.006.

For clarity, the article has been reformatted, otherwise the content is the same as the published version of work. © 2024 by the author. Reproduced with permission from Elsevier.

<https://www.sciencedirect.com/science/article/abs/pii/S0167739X23004053>

# 1. Introduction

A substantial amount of decentralized data is generated daily on end devices such as mobile phones or laptops. Keeping data decentralized has several potential benefits, including increased privacy and security for users and reduced reliance on a central authority for data storage and processing. An end device’s private data may contain valuable information that can be used to train deep neural network models, either through Federated Learning [1] or peer-to-peer learning [2]. Compared to the Federated Learning approach, which orchestrates the learning process using a central server, peer-to-peer allows multiple devices to train machine learning models collaboratively without a central authority. While both approaches share the models, the private data never leaves an end device known as an agent.

Agents’ data heterogeneity affects the collective learning process, slowing it down and resulting in sub-optimal models [3, 4]. To mitigate these challenges, significant advances were made to enable multi-task learning [5–9]. In multi-task learning, agents are divided into clusters based on model and data similarity conditions. Clustering reduces data heterogeneity, leading to better model convergence and learning outcomes. In an extreme case where clusters are reduced to a single agent, this technique can be considered a personalization technique [10]. Consider the speech recognition task, where voice samples are stored at the agent level and vary depending on the language, accent, pitch, etc. Solving the speech recognition problem through multi-task learning would imply that the clusters would be organized according to similar accents or pitch. However, all agents are learning to solve a common problem, e.g. speech recognition.

In a previous study [11], we addressed the problem of data heterogeneity by developing a personalization technique that, in combination with Batch Normalization (BN) layers [12], enabled agents to achieve faster convergence and higher top accuracy on a single task. This paper studies an approach of collaboratively training agents in a peer-to-peer network that can learn two distinct Natural Language Processing (NLP) tasks: masked-token prediction (MTP) as the *masked language modeling* task, and the named-entity recognition (NER) as a *token classification* task. The hypothesis is that agents can boost the performance of their local models by collaborating with agents learning a completely different task within the same data modality. Encoder-only transformer architecture used in the BERT [13] model is used as a model base for

both tasks. It is well established that pre-training BERT models and fine-tuning for a specific task improves the model’s performance [13], which this research strives to apply in a peer-to-peer setting. This research uses the encoder’s ability to store the common knowledge shared among all agents, regardless of the assigned agent cluster. The task-specific elements of the models are shared only within the cluster. The output generated by the attention layers in the encoder is forwarded to a task-specific classifier. For this purpose, this research suggests a peer-to-peer multi-task learning approach for collaboration between agents learning distinctly different NLP tasks. The contributions can be summarized as follows:

- a multi-task method for agent collaboration on distinct NLP objectives: masked-token prediction and named-entity recognition,
- a technique for constructing a network topology that establishes connections among agents in different objective clusters,
- a method for the strategical training of shared model layers for faster convergence.

To ensure the reproducibility of our results, our code is publicly available at [https://github.com/fipu-lab/p2p\\_bn](https://github.com/fipu-lab/p2p_bn).

The rest of this paper is organized as follows. Section 2 provides more details on the BERT transformer model and decentralized learning. Section 3 reviews and describes related work. The multi-task method is presented in Section 4. Section 5 presents the evaluation methodology and experimental results. The limitations are discussed in Section 6, and Section 7 concludes the paper.

## **2. Background**

This section introduces the concepts of decentralized learning and the BERT transformer model.



## 2.1. BERT transformer model

The BERT model, or Bidirectional Encoder Representations from Transformers, is a powerful language model developed by Google researchers in 2018/2019 [13]. BERT is designed to handle natural language processing (NLP) tasks such as language translation, question answering, and language generation. It has achieved state-of-the-art results on a wide range of benchmarks. One of the key features of BERT is its use of transformer architecture, which allows the model to process long-range dependencies in language effectively. BERT also uses self-attention mechanisms [14], which allow the model to attend to different input parts simultaneously and weigh their relative importance. Self-attention helps the model capture contextual relationships between words in a sentence and use this information to make more accurate predictions. In addition to its impressive performance on natural language processing tasks, BERT has also been widely adopted for various other tasks, including sentiment analysis [15] and named entity recognition [16]. Different sizes of BERT models can be constructed by varying the number of attention heads, the number of hidden layers of transformer blocks, and hidden size [17]. The hidden size refers to the number of units in the transformer’s self-attention layers. Figure 1 shows the architecture of the BERT model. This study utilizes the encoder-only transformer architecture as used in the BERT model. The encoder’s sequence output is processed and forwarded to the final output layer for a specific task (MTP or NER), i.e., each agent owns only one encoder model and one final task-specific output layer. The encoder architecture employed in this study is derived from the BERT-Tiny [17].

## 2.2. Decentralized learning

Agents adhere to a network topology in a decentralized network and only exchange messages with their neighbors or peers. In most studies, the network topology is predetermined, such as ring, sparse, or fully connected topology. A sparse topology is the most lenient, as any rule does not bind it; agents are randomly connected and only limited by the number of neighbors they can have. In directed (asymmetric) communication, an agent receives messages from one set of peers and sends messages to another set of peers [11, 18]. Undirected communication

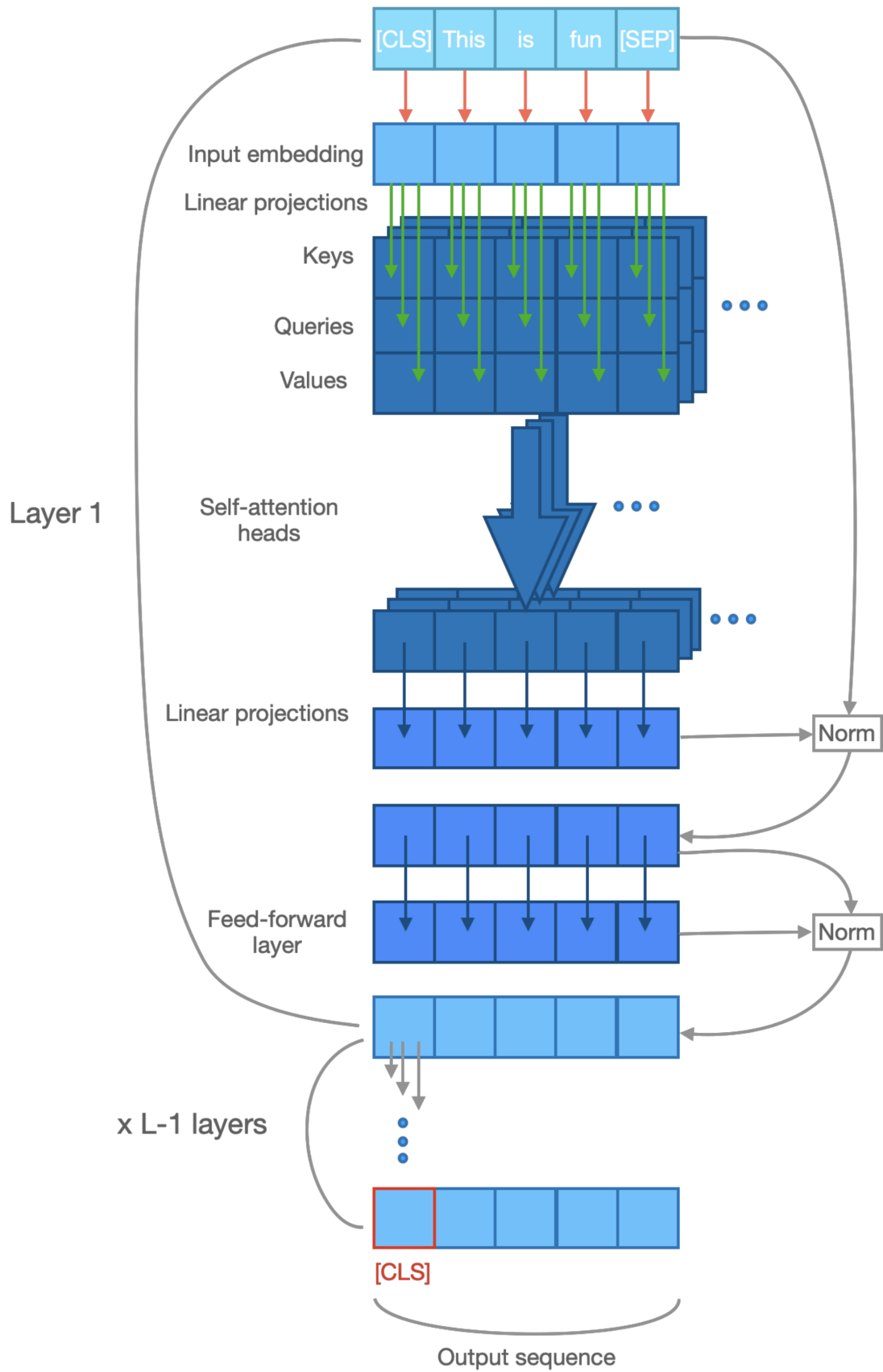


Figure 1: BERT model architecture.

is used when an agent sends and receives messages from the same set of peers [19, 20]. In most cases, information relayed between agents are the learned model parameters. Incoming model parameters are commonly aggregated by averaging with local model parameters or aggregating upon receiving all messages [21, 22].

### 3. Related work

Multi-task learning has been considered for peer-to-peer and Federated Learning (FL). In FL, the most commonly studied approach is keeping one model part or module stored locally on the agents while sharing the other part with the server [10, 23–26]. By splitting the model, agents own a part of general and task-specific knowledge. Other studies divided agents into groups based on the similarity of agents’ models [27], and only the models within a certain group are aggregated to form a new group-global model.

A very lenient approach was studied in [9] for FL and peer-to-peer, where any part of the neural model can be shared with peers. The authors evaluated their approach by classifying handwritten characters using the FEMNIST [28] dataset. The FEMNIST character images are grouped by the author of the handwritten characters, and data bound by one author was assigned to one agent as the training data. The experiments in the study demonstrated that the optimal percentage of the model shared is around 80% for maximum accuracy, but with the emphasis that the number of performed mini-batch updates is more impactful than the averaging level. A relative accuracy increase of around 9% was achieved with the 80% averaging level compared to the 100% averaging level. Taylor *et al.* [8] utilized a multi-task approach for predicting tomorrow’s mood, stress, and health outcomes. The proposed method divides the model into two distinct components: a shared part and multiple task-specific parts. User data was organized into clusters based on their personality and gender. The single shared part of the model extracted features pertinent to all clusters, while the task-specific parts generated tailored predictions for each cluster. Compared to the single-task baseline, multi-task learning improved the accuracy by around 12%. In a peer-to-peer environment, Mohammadi *et al.* [7] implemented a system where individual agents retained specific expertise through local skills while simultaneously

sharing general knowledge with nearby agents. This collaborative approach allowed agents to benefit from knowledge exchange while maintaining their proficiency in solving localized data, improving overall performance.

Additionally, multi-task learning can be enabled by strategically manipulating agent connections and effectively clustering agents with comparable tasks. Zantedeschi *et al.* [6] introduced a method to enable the dynamic formation of peer-to-peer connections by exploiting the resemblance among agents’ local linear models using empirical loss on the agent’s local dataset. Building on this idea, several other studies have further applied this approach to neural networks, yielding better model performance in scenarios with heterogeneous data distribution [29–33]. These scenarios encompass diverse image rotations, varying label semantics, and disparate data distributions, with most experiments conducted using the CIFAR-10 dataset [34].

All of the mentioned studies divide agents based on the heterogeneity of their data, considering one or a group of agents as a separate task. However, the objective is always to learn one task for all agents (e.g., handwritten character classification). Our presented approach allows for distinct groups of agents to collaboratively learn different NLP tasks, such as masked-token prediction (MTP) and named-entity recognition (NER), and for an agent group learning one task to enhance its learning ability through collaboration with a group of agents learning another task.

## 4. Multi-task learning

The proposed technique uses the P2P-BN [11] approach as the peer-to-peer gossiping method. An early stopping technique that uses Batch Normalization layers was proposed for P2P-BN as a personalization technique, which was omitted for our approach since our models do not contain any BN layers. P2P-BN considers a set of  $N$  independent agents communicating with each other. Each agent has its local model  $m_i$ , local dataset  $D_i$ , and is solving the task  $T_i$  where  $type(T_i) \in \{MTP, NER\}$ . Each model  $m_i$  is comprised of the encoder-only transformer (referred to as the encoder layer) and additional task-specific layers (see Figure 2). The encoder layer is shared between all agents, while the task-specific layers are shared only between agents within the same task  $T$ . Note that an agent only trains to solve one task  $T_i$  but utilizes the

multi-task collaboration to achieve higher local model accuracy.

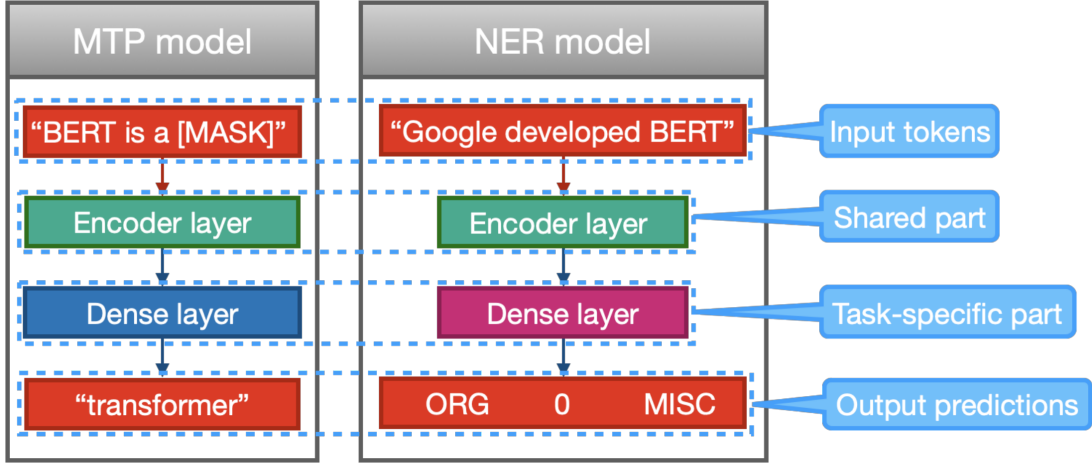


Figure 2: Architecture of MTP and NER models.

**Network topology.** Messages exchanged between agents contain the sending agent’s model parameters  $m_i$ , and the receiving agent  $j$  decides how to aggregate the received parameters with its model  $m_j$  based on the sending agent’s task  $T_i$  and own task  $T_j$ . In a set of  $N$  agents, each agent only communicates with its peers based on the predetermined communication graph  $G = (\llbracket N \rrbracket, E, W)$ , where  $\llbracket N \rrbracket = \{1, \dots, N\}$  is a set of all agents,  $E \in \llbracket N \rrbracket \times \llbracket N \rrbracket$  is the set of edges, and  $W \in \mathbb{R}^{N \times N}$  is a non-negative weighted matrix. Weight of edge  $(i, j) \in E$  is given by  $W_{ij}$  with the convention  $W_{ij} = 0$  if  $(i, j) \notin E$  or  $i = j$ . As in P2P-BN, agent  $i$  only sends messages to agent  $j$  if  $W_{ij} > 0$ , which means that agent  $i$  communicates with peers  $N_i = \{j : W_{ij} > 0\}$  without knowledge of other peers in the network and operates without synchronization with non-connected peers ( $W_{ij} = 0$ ). An agent may have different in-neighbour  $(i, j) \in E$  and out-neighbor  $(j, i) \in E$  connections in a directed graph. The hypothesis is that the number of connected peers from another task  $|\{W_{ij} > 0 \wedge T_i \neq T_j\}|$  should be limited to a specific portion of the total number of connected peers  $|\{W_{ij} > 0\}|$  to achieve better results. We denote the parameter  $PT$  as a fraction of connected peers from a different task and the total number of connected peers:

$$PT = \frac{|\{W_{ij} > 0 \wedge T_i \neq T_j\}|}{|\{W_{ij} > 0\}|} \quad (1)$$

**Local model training.** An agent aims to train its local model by leveraging its local dataset  $D_i$  and updates received from its peers to try to minimize its loss function  $F_i$  for a specific task  $T_i$ . An agent trains its local model by performing mini-batch gradient updates to its local model

$m_i = m_i - \eta F_i(m_i; \xi_i)$ ,  $\xi_i \sim D_i$  for  $E$  epochs over its local training dataset, with  $\eta$  denoting agent's learning rate. When an agent  $i$  receives an update from an agent  $j$ , a standard model averaging  $m_i = \frac{m_i + m_j}{2}$  is performed if  $type(T_i) = type(T_j)$ . Otherwise, only the encoder layer of the model is averaged  $m_i^{(\text{encoder})} = \frac{m_i^{(\text{encoder})} + m_j^{(\text{encoder})}}{2}$ , where  $m_i^{(\text{encoder})}$  and  $m_j^{(\text{encoder})}$  denotes only the encoder layer of the  $m_i$  and  $m_j$  models, respectively. An additional introduced technique is slowing down the shared layer learning in a setting where an agent has a neighborhood with a different type of tasks: if  $type(T_i) \neq type(T_j)$  and only the encoder layer is averaged, the receiving agent freezes the encoder layer for the next training round. Let  $\|m_i\|$  denote model  $m_i$  with frozen encoder layer, then an agent  $i$  performs the next local training round on model  $m_i$  as follows:  $m_i = m_i - \eta F_i(\|m_i\|; \xi_i)$ ,  $\xi_i \sim D_i$  for  $E$  epochs. The reasoning behind freezing the encoder layer upon receiving updates from a different task is that the encoder layer represents a shared common knowledge of the network. Agents slowly reach a consensus over the shared encoder layer by only updating the task-specific layers and not training the encoder layer. It is important to note that freezing the encoder layer does not occur in every training cycle. An agent may receive updates from its peers who are learning the same task, in which case there is no freezing of the encoder layer. This technique also acts as personalization by only training a part of a model (task-specific layers). All model weights should be initialized for optimal convergence with identical model parameters and identical shared model parameters [35].

Pseudo-code shown in algorithm 1 is a modified version of the P2P-BN method and details the global agent behavior:

- Initialization: all agents initialize their models with identical model parameters, which implies that the weights of the encoder layer would be identical across agents, regardless of the task
- Train agent: at each iteration, one of the agents that received at least one update from the last active state is locally trained
- Local training: the chosen agent performs the training step and unfreezes the encoder layer (if previously frozen). The agent then sends the model parameters to its peers. If any of the peers have a different task  $T_i \neq T_j$ , the peer freezes the encoder layer after aggregating the received update

---

**Algorithm 1** Agents training simulation pseudo-code

---

```
1: function TRAIN( $A, W, E$ )
2:    $A$ : agents
3:    $W$ : communication matrix
4:    $E$ : number of epochs to perform locally
5:   INITIALIZE( $A$ )    ▷ Initialize all agents models and encoder layers to identical model
                        parameters
6:   repeat
7:      $a_i = \text{RANDOMACTIVEAGENT}(A)$     ▷ Random agent which received at least one
                        update from last active state
8:     LOCALTRAIN( $a_i, E$ )
9:     if ISENCODERFROZEN( $a_i$ ) then
10:      UNFREEZEENCODERLAYER( $a_i$ )
11:    for  $a_j$  in NEIGHBORS( $W, a_i$ ) do
12:      if  $T_j \neq T_i$  then
13:        ENCODER( $a_j$ ) = AVERAGEENCODER( $a_j, a_i$ )
14:        FREEZEENCODER( $a_j$ )
15:      else
16:        MODEL( $a_j$ ) = AVERAGEMODEL( $a_j, a_i$ )
17:    until Maximum iteration reached
```

---

## 5. Evaluation

This section evaluates the proposed contributions through several experiments comparing multi-task learning to the baseline single-task learning in different directed sparse topologies. The methodology and metrics employed in the experiments are first described, followed by a comprehensive analysis and discussion of the results obtained.

### 5.1. Methodology

**Model and datasets.** For the MTP task, Reddit [28] and StackOverflow [36] datasets are used. Both datasets are comprised of unique internet users and their comments. CoNLL-2003 [37] and Few-NERD [38] datasets are used for the NER task. The CoNLL-2003 dataset contains four different entities: person, location, organization, and miscellaneous, while the Few-NERD dataset consists of eight main types of entities: art, building, event, location, organization, person, product, and miscellaneous. In the following experiments, an agent has either received

comments from a unique user for the MTP task or a uniform sample of data for the NER task.

The approach taken in the initial BERT study is followed, and the WordPiece embeddings [39] with a 30,000 token vocabulary are used for both tasks. For the MTP task, a *[MASK]* token is used as a placeholder for the word being predicted. Encoder layer outputs related to the *[MASK]* token are then forwarded to the output layer. The MTP task is formed as a next-word prediction (NWP) [40, 41] task. Consequently, the *[MASK]* token consistently appears at the end of the token sequence. Solely predicting the final word of the sentence diminishes a crucial advantage of the BERT-based model, namely its capacity to incorporate pertinent contextual information for computing word probability distribution. However, this characteristic can be perceived as an additional task heterogeneity, introducing complexity in establishing parallels between the two tasks. For the NER task, all encoder layer outputs are forwarded to the output layer as a prediction is needed for each input word besides the padding tokens. For both MTP and NER, the first token of the sequence is always *[CLS]* and the last *[SEP]*. Sentences are broken down into sequences of 128 words, and padding is added to the end of sentences shorter than 128 words. For the NWP task, most studies chose a sequence length of ten to twenty words. However, our preliminary experiments showed that one task using a substantially smaller sequence length could cause accuracy degradation for the other task. Figures 3 and 4 showcase an example of an MTP and NER sequence, respectively.

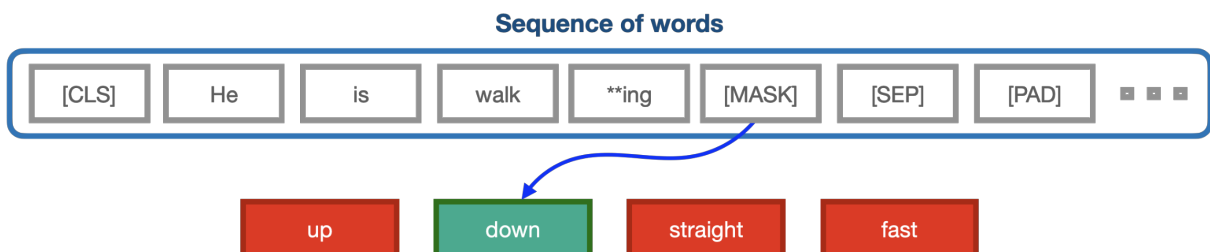


Figure 3: MTP sequence formed from the sentence "He is walking down the street". Token *[MASK]* is a placeholder for the predicting word. Only "down" is correct, even though multiple words could be considered correct in this context.

The encoder architecture utilized in the experiments is derived from the BERT-Tiny architecture comprised of two attention heads, two transformer blocks in the hidden layers, and a hidden size of 128 units. The resulting architecture consists of eight million parameters. The selection of the smallest variant stems from several justifications. Notably, the choice is informed by the constrained resources of agents, such as mobile or IoT devices, which are limited in memory and computing capacities. Furthermore, relaying larger models over the network



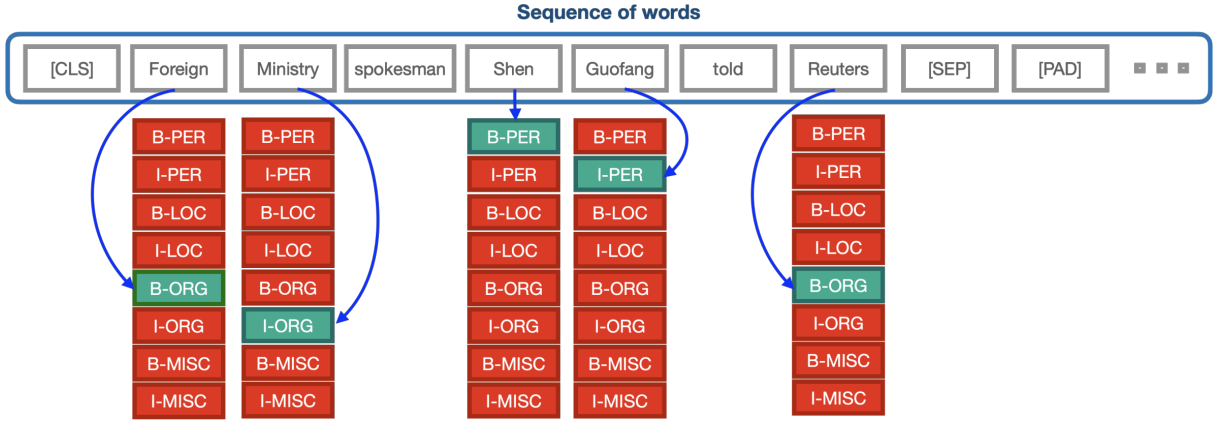


Figure 4: NER sequence formed from the sentence "Foreign Ministry spokesman Shen Guofang told Reuters". Specific words must be classified as one of the possible entities.

entails additional communication costs, and the limited data accessible to agents alleviates the need for larger models. However, the main focus of this study is to enable multi-task learning among diverse tasks in decentralized networks rather than aiming for state-of-the-art accuracy on any particular NLP task.

**Metrics.** All metrics are measured at the agent level using the agent's local data. Average User model Accuracy (UA) [26] metric is used to measure the performance of the overall learning process. UA metric is calculated as the average accuracy across all agents and can be expressed as:

$$UA = \frac{1}{n} \sum_{i=1}^n acc_i \quad (2)$$

where  $acc_i$  is the accuracy of metric  $acc$  on model  $i$  using local test dataset  $i$ . MTP task can be considered as a classification task, and the prediction accuracy is calculated as the fraction of correct predictions (TP and TN) over total predictions (TP, TN, FP, and FN):

$$accuracy_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \quad (3)$$

The F1 score metric measures the model's accuracy for the NER task and is defined as the unweighted average (macro f1-score) of precision and recall, given as follows:

$$precision_i = \frac{TP_i}{TP_i + FP_i} \quad (4)$$

$$recall_i = \frac{TP_i}{TP_i + FN_i}, \quad (5)$$

$$F1-score_i = \frac{2 \times recall_i \times precision_i}{recall_i + precision_i}. \quad (6)$$

## 5.2. Experiments

Several experiments are performed and analyzed to evaluate and compare our approach to individual clustered task-specific (CTS) learning, where agent groups are trained separately for each task and dataset. Experiments are carried out with and without encoder layer freezing, termed as *MT-EF* and *MT*, respectively. The first experiment demonstrates a naive approach of randomly creating peer connections (regardless of the task) in both *MT-EF* and *MT*. The second experiment demonstrates clear benefits in both *MT-EF* and *MT* when limiting the number of non-task-specific peers. The third experiment presents *MT-EF* benefits when collaboratively learning four different tasks simultaneously.

For all experiments, the batch size is set to 50, and all agents use the Adam [42] optimizer with the learning rate set to  $5 \times 10^{-4}$ . When training BERT transformers, it is common to have a warmup period with a high learning rate for a certain number of training iterations [13]. The learning rate is then gradually decreased using a linear decay. In these experiments, the learning rate is permanently fixed to  $5 \times 10^{-4}$  to reduce the learning rate’s impact on the overall learning process. Data is processed in sequences of 128 tokens for both MTP and NER tasks, and the learning process is stopped once the agents, on average, perform 300 epochs over the local dataset. A directed sparse topology is used in experiments, with each agent having three in-neighbors and three out-neighbors. The experiments in [11] showed that compared to undirected, directed sparse communication produced better results, which we confirmed with our preliminary evaluations. The term *random sparse* topology is used when connections between agents are formed randomly, regardless of the task. The parameter *PT* is set to 0.33 in the experiments with limited non-task-specific peer connections, which we termed as *clustered sparse* topology. As a result, in *clustered sparse* topology, each agent established two peer connections with agents within the same task and one with an agent from a different task. In the

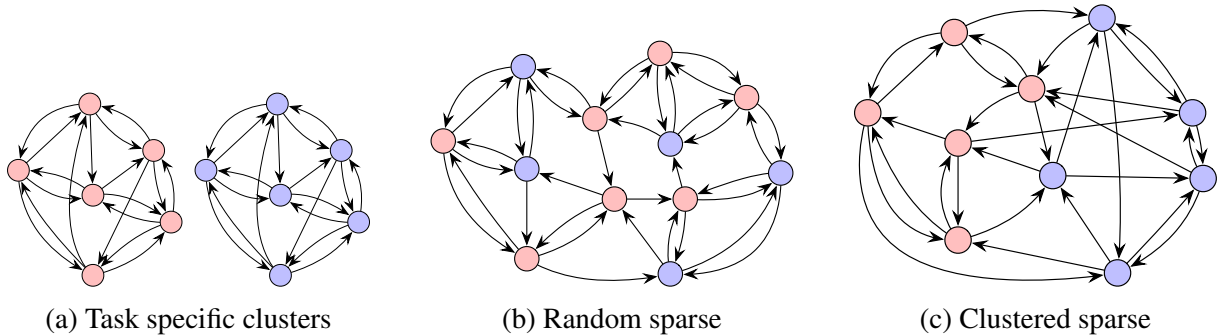


Figure 5: An example of task-specific clusters (a), *random sparse* (b), and *clustered sparse* (c) topology. Agent color represents the learning task.

conducted experiments, the topology is predefined and established at the onset of the training process based on the specific learning task assigned to the agents. An example of task-specific clusters, *random sparse*, and *clustered sparse* topology is shown in Figure 5. A group of twenty agents is trained for each task (dataset) for all experiments. As the Reddit and StackOverflow datasets are partitioned at the user level, data regarding one user is assigned to one agent. For the NER datasets, data is partitioned uniformly to twenty agents. Since the Few-NERD dataset is substantially larger than the CoNLL-2003 dataset, only 20% of the Few-NERD dataset is used in the experiments. In summary, on average, an agent has around a thousand training examples, regardless of the dataset or task. For each experiment, three runs are conducted, each time with a different set of users for the MTP task and different data distributions for the NER task. The Student t-test evaluates the statistical difference between CTS baseline and multi-task results. Since the word vocabulary is identical for both MTP datasets, the results of training the MTP problem as a mono task are also given in the experiment results.

**Multi-task in *random sparse* topology.** The first experiment demonstrates the multi-task characteristics in a *random sparse* network topology. Evaluations are performed for all two-task combinations for *MT-EF* and *MT*. Figure 6 shows that both MTP tasks converge faster and achieve higher top accuracy when trained individually compared to both types of multi-task learning. Similarly, CTS learning of the NER task produces better results than any multi-task combination. The only exception is the NER task using the CoNLL-2003 dataset, which achieved higher accuracy for all *MT* combinations. Agents learning the NER task using the CoNLL-2003 dataset achieve higher accuracy in *MT* setting while simultaneously degrading the accuracy of the MTP and the Few-NERD agents, suggesting that, in *random sparse* topology, multi-task optimization is not beneficial for all agents. Table 1 summarizes the top UA

accuracies and the relative increase/decrease in accuracy compared to the baseline CTS results. Additionally, the MTP mono task displayed some interesting results. Even though StackOverflow is generally focused on programming questions and answers and Reddit is not on any specific topic, both tasks achieve higher top accuracy with faster convergence when trained as a mono task. This result questions previous findings that dividing agents into separated multi-task problems benefits all agents, at least in the case of MTP tasks.

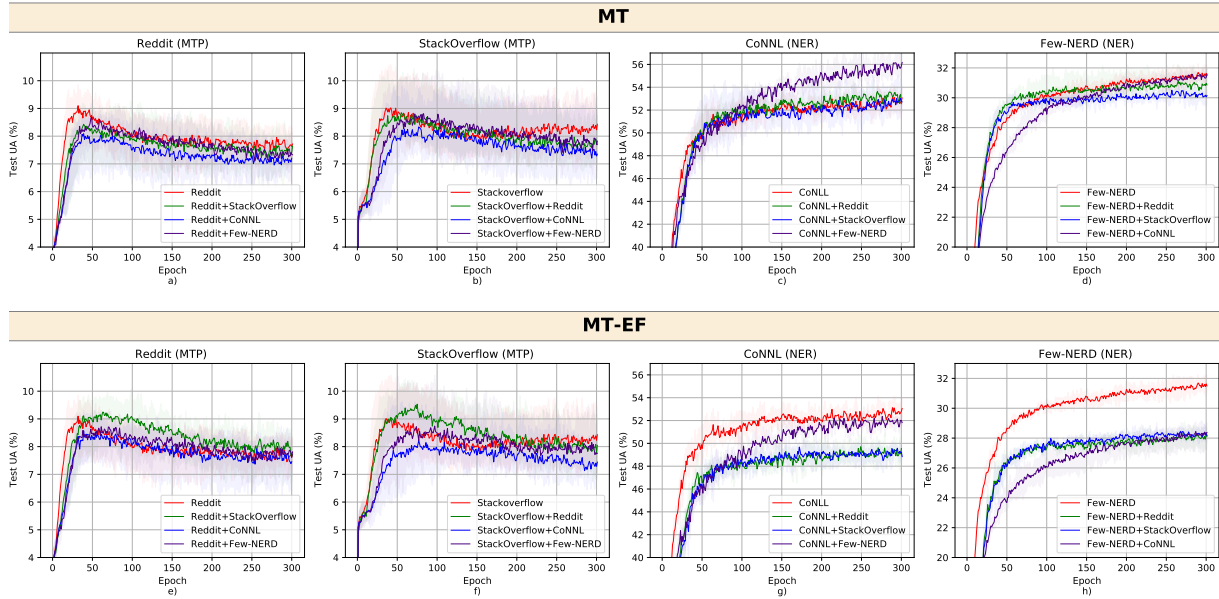


Figure 6: Test UA in a sparse topology. The plus sign (+) denotes multi-task collaboration between the two tasks, while only the dataset name (color red) represents the CTS baseline. Results are shown for both *MT* and *MT-EF*.

Table 1: Top Test UA in a sparse topology. Column values summarize the results of each dataset task for each of the multi-task combinations. Statistically significant results ( $p < 5 \times 10^{-3}$ ) are marked with **\*\***.

| Collaboration | Dataset (Task) | Reddit                   | StackOverflow              | CoNLL                     | Few-NERD            |
|---------------|----------------|--------------------------|----------------------------|---------------------------|---------------------|
| CTS baseline  |                | 9.10%                    | 9.02%                      | 53.13%                    | 31.69%              |
| Mono          |                | <b>9.56% (+5.05%) **</b> | <b>10.39% (+15.19%) **</b> | -                         | -                   |
| MT            |                |                          |                            |                           |                     |
|               | Reddit         | -                        | 8.85% (-1.88%) **          | <b>53.63% (+0.94%) **</b> | 31.14% (-1.74%)     |
|               | StackOverflow  | 8.43% (-7.36%) **        | -                          | 53.03% (-0.19%) **        | 30.51% (-3.72%) **  |
|               | CoNLL          | 8.43% (-7.36%) **        | 8.35% (-7.43%) **          | -                         | 31.58% (-0.35%) **  |
|               | Few-NERD       | 8.68% (-4.62%) **        | 8.81% (-2.33%) **          | <b>56.17% (+5.72%) **</b> | -                   |
| MT-EF         |                |                          |                            |                           |                     |
|               | Reddit         | -                        | <b>9.53% (+5.65%) **</b>   | 49.71% (-6.44%) **        | 28.30% (-10.70%) ** |
|               | StackOverflow  | <b>9.25% (+1.65%) **</b> | -                          | 49.68% (-6.49%) **        | 28.45% (-10.22%) ** |
|               | CoNLL          | 8.57% (-5.82%) **        | 8.22% (-8.87%) **          | -                         | 28.44% (-10.26%) ** |
|               | Few-NERD       | 8.72% (-4.18%) **        | 8.74% (-3.10%) **          | 52.36% (-1.45%) **        | -                   |

**Multi-task clustered sparse topology.** In *clustered sparse* topology, an agent communicates with two peers that share the same task and one peer from another task. *Clustered sparse* topol-

ogy can be considered a mixture between task-specific clusters and a *random sparse* topology. Since each agent communicates with three peers, the following results are achieved using an equal number of exchanged messages (models). Figure 7 shows the results obtained for *MT* and *MT-EF*. While the *MT* setting obtained slightly higher accuracies for MTP and NER tasks using the CoNLL-2003 dataset, the Few-NERD agents achieved lower accuracy for all multi-task combinations. In contrast, in the *MT-EF* setting, all tasks obtained a substantially higher top accuracy in any multi-task combination. Table 2 summarizes the obtained top UA accuracies and the relative increase/decrease in accuracy compared to baseline CTS results. Training the MTP datasets as a mono task in *clustered sparse* topology achieved slightly higher test UA than in *random sparse* topology, suggesting that connection manipulation may improve accuracy even in mono tasks.

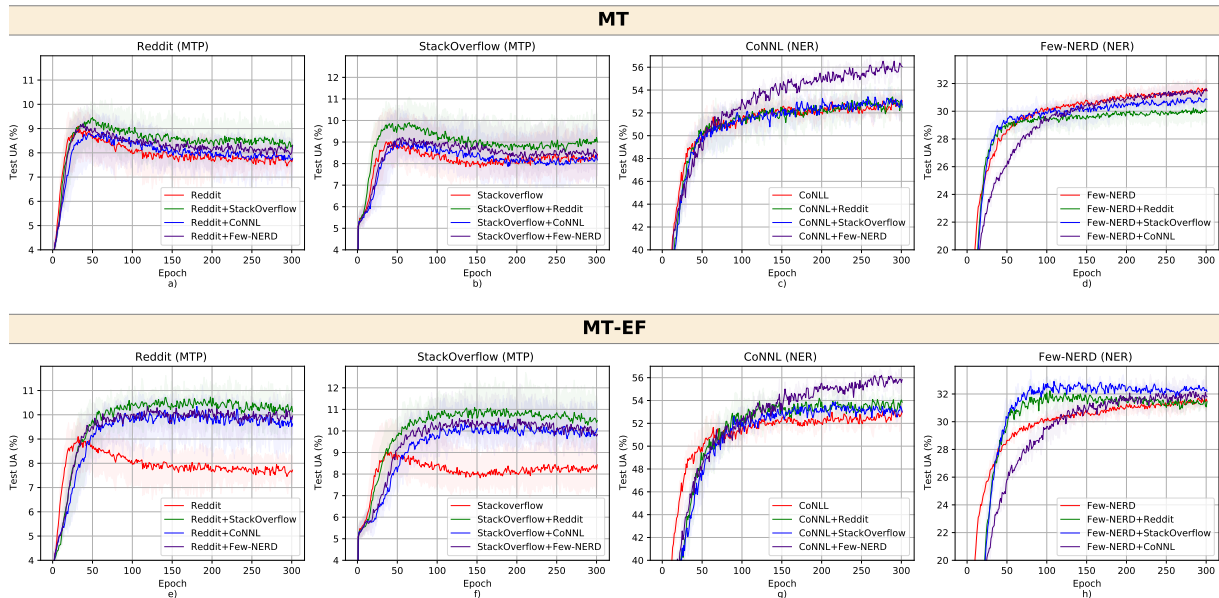


Figure 7: Test UA in a clustered sparse topology. The plus sign (+) denotes multi-task collaboration between the two tasks, while only the dataset name (color red) represents the CTS baseline. Results are shown for both *MT* and *MT-EF*.

**Four task *MT-EF* in *clustered sparse* topology.** Previous experiments demonstrated that the *MT-EF* collaboration between two different tasks benefits both clusters of agents. This experiment evaluates the collaboration of training all four tasks in a *clustered sparse* topology using the *MT-EF* approach. Based on the parameter *PT* (set to 0.33), each agent only forms one peer connection with another agent with a different task. Once all connections are formed, all clusters are connected through at least one agent; however, not all agents communicate with agents from all tasks (see Figure 8 as an example). As Figure 9 shows, all four different groups

Table 2: Top Test UA in a clustered sparse topology. Column values summarize the results of each dataset task for each of the multi-task combinations. Statistically significant results ( $p < 5 \times 10^{-3}$ ) are marked with \*\*.

| Collaboration | Dataset (Task) | Reddit                     | StackOverflow              | CoNLL                     | Few-NERD                  |
|---------------|----------------|----------------------------|----------------------------|---------------------------|---------------------------|
| CTS baseline  |                | 9.10%                      | 9.02%                      | 53.13%                    | 31.69%                    |
| Mono          |                | <b>9.71% (+6.70%) **</b>   | <b>10.41% (+15.41%) **</b> | -                         | -                         |
| MT            | Reddit         | -                          | <b>9.90% (+9.76%) **</b>   | <b>53.40% (+0.51%)</b>    | 30.24% (-4.58%) **        |
|               | StackOverflow  | <b>9.46% (+3.96%) **</b>   | -                          | <b>53.43% (+0.56%)</b>    | 30.98% (-2.24%) **        |
|               | CoNLL          | 8.86% (-2.64%) **          | <b>9.10% (+0.89%) **</b>   | -                         | 31.61% (-0.25%) **        |
|               | Few-NERD       | <b>9.20% (+1.10%) **</b>   | <b>9.22% (+2.22%) **</b>   | <b>56.54% (+6.42%) **</b> | -                         |
| MT-EF         | Reddit         | -                          | <b>11.10% (+23.06%) **</b> | <b>54.20% (+2.01%) **</b> | <b>32.23% (+1.70%) **</b> |
|               | StackOverflow  | <b>10.73% (+17.91%) **</b> | -                          | <b>53.89% (+1.43%) **</b> | <b>32.92% (+3.88%) **</b> |
|               | CoNLL          | <b>10.35% (+13.74%) **</b> | <b>10.43% (+15.63%) **</b> | -                         | <b>32.19% (+1.58%)</b>    |
|               | Few-NERD       | <b>10.35% (+13.74%) **</b> | <b>10.60% (+17.52%) **</b> | <b>56.22% (+5.82%) **</b> | -                         |

of agents were able to achieve higher top accuracies when trained in a multi-task setting, suggesting that limited collaboration between different tasks of agents is always beneficial for all agents. Table 3 summarizes the achieved top UA accuracies with the average results from prior experiments included for reference (2 tasks (avg)). All agents achieved a higher top accuracy when learning all four tasks simultaneously compared to the average top accuracy achieved in prior experiments involving two-task combinations, except for the Few-NERD agents, whose results were lower but comparable.

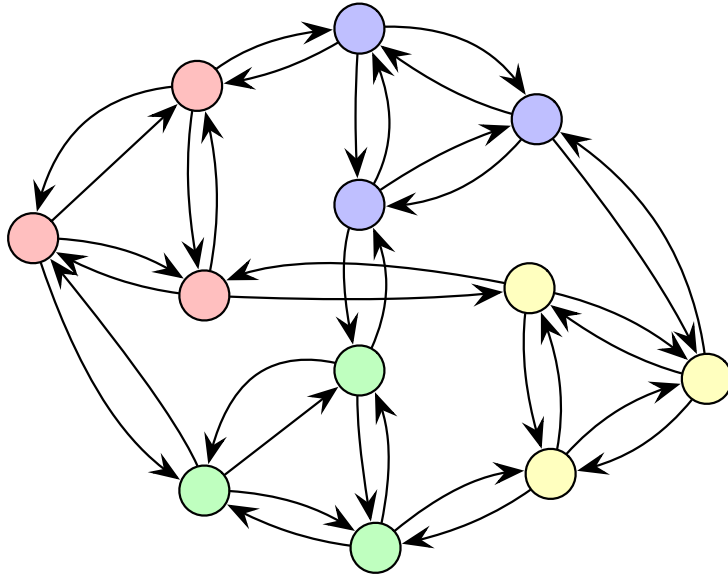


Figure 8: An example of four different agent clusters with two task-specific and one non-task-specific connection randomly formed with another cluster of agents, regardless of the task.

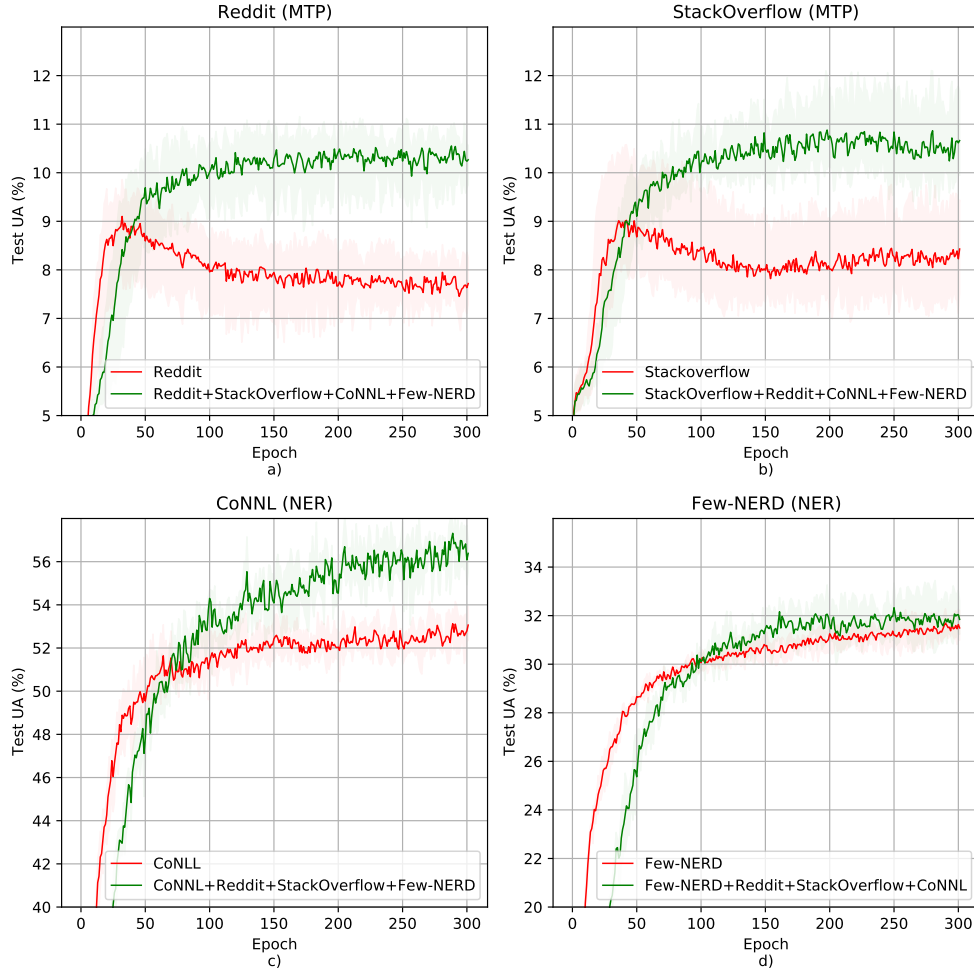


Figure 9: Test UA for *MT-EF* in a clustered sparse topology. The plus sign (+) denotes multi-task collaboration between the (four) tasks, while only the dataset name (color red) represents the CTS baseline.

Table 3: Test UA for *MT-EF* in a clustered sparse topology. Column values summarize the average results from the previous experiment using the *MT-EF* (2 tasks) and results of four way *MT-EF* collaboration between all four tasks (4 tasks). Statistically significant results ( $p < 5 \times 10^{-3}$ ) are marked with \*\*.

| Collaboration | Dataset (Task) | Reddit                     | StackOverflow              | CoNLL                     | Few-NERD        |
|---------------|----------------|----------------------------|----------------------------|---------------------------|-----------------|
| CTS baseline  |                | 9.10%                      | 9.02%                      | 53.13%                    | 31.69%          |
| MT-EF         | 2 tasks (avg)  | 10.48% (+15.16%)           | 10.71% (+18.74%)           | 54.77% (+3.09%)           | 32.45% (+2.40%) |
| MT-EF         | 4 tasks        | <b>10.55% (+15.93%) **</b> | <b>10.88% (+20.62%) **</b> | <b>57.32% (+7.89%) **</b> | 32.33% (+2.02%) |

## 6. Limitations

**Sequence length.** One of the limitations of the proposed approach is that all tasks must use identical sequence lengths when training collaboratively. In the domain of causal language modeling, the next-word prediction (NWP) task often employs a 20-word sequence as model input [40, 41]. However, it is worth noting that this particular approach was not adhered to in the context of the MTP task. We observed that using different sequence lengths only benefits the task with the shorter sequence length while greatly disadvantaging the task with a longer sequence length.

**Natural language processing.** Our approach is currently limited to NLP tasks. While alternative versions of transformer models are also used in other domains, such as vision processing, it needs to be determined to which degree there is an overlap with the NLP models and study if any collaboration between tasks may be established.

This study aimed to demonstrate beneficial cooperation among clusters of agents assigned with distinct tasks. However, to attain results comparable to state-of-the-art research, pre-trained encoder-only transformer weights may be used as the agent’s initial model weights, which would greatly improve the results given the limited amount of data accessible to the agents.

**Shared tokenizer.** It is common in the literature to assume a globally shared tokenizer. However, we identify this as a limitation and an opportunity for future research. Specifically, this could include research on reaching a consensus on the tokenizer to be used in the training process among different agents in the network.

**Scaling.** Performed experiments were limited to two distinct NLP problems, using four different datasets with a set of twenty agents from each dataset. To further understand the presented approach, future research should focus on scaling by increasing the number of agents and tasks. Additionally, the impact of the parameter  $PT$  on the multi-task learning process should be investigated through additional experiments.



## 7. Conclusion

The main contribution of this research is a technique that utilizes multi-task learning in a peer-to-peer architecture to improve the accuracy of deep neural networks trained on heterogeneous data and distinctly different tasks. The approach presented utilized the encoder-only transformer model in a multi-task peer-to-peer network for the masked-token prediction (MTP) and named-entity prediction (NER) tasks, successfully demonstrating the benefits of a careful collaboration between the agents with distinct tasks. Our experiments demonstrated the negative impact of randomly connecting agents in a sparse topology. Random peer connections left some agents with no task-specific neighbors, which resulted in lower accuracy. Limiting the connections between agents with different tasks improved the inner task-specific collaboration while retaining non-task-specific collaboration. Furthermore, agents reached consensus faster by strategically freezing the shared part of the model, resulting in improved accuracy results. The *MT-EF* collaboration led to a statistically significant increase of 11.6% in the mean relative accuracy compared to the baseline results for individual tasks. Future research should explore the generalization of these findings to other types of tasks and model architectures, as well as the scalability of this approach to larger networks with more agents and different tasks. Additionally, as mentioned in the limitations, pre-trained encoder-only transformer weights (such as BERT weights) may be utilized as local model’s initial weights which would further improve the learning outcomes.

## References

- [1] Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 1273–1282. URL: <https://proceedings.mlr.press/v54/mcmahan17a.html>.

- [2] Michael Blot et al. “Gossip training for deep learning”. In: (Nov. 2016).
- [3] Rong Yu and Peichun Li. “Toward Resource-Efficient Federated Learning in Mobile Edge Computing”. In: *IEEE Network* 35.1 (2021), pp. 148–155. DOI: 10.1109/MNET.011.2000295.
- [4] Xiaodong Ma et al. “A state-of-the-art survey on solving non-IID data in Federated Learning”. In: *Future Generation Computer Systems* 135 (2022), pp. 244–258. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2022.05.003>. URL: <https://www.sciencedirect.com/science/article/pii/S0167739X22001686>.
- [5] Mohammad Rostami et al. “Multi-Agent Distributed Lifelong Learning for Collective Knowledge Acquisition”. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. Ed. by Andre Elisabeth et al. AAMAS ’18. Stockholm, Sweden: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 712–720.
- [6] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. “Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 864–874. URL: <https://proceedings.mlr.press/v108/zantedeschi20a.html>.
- [7] Javad Mohammadi and Soheil Kolouri. “Collaborative Learning Through Shared Collective Knowledge and Local Expertise”. In: *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*. 2019, pp. 1–6. DOI: 10.1109/MLSP.2019.8918888.
- [8] Sara Taylor et al. “Personalized Multitask Learning for Predicting Tomorrow’s Mood, Stress, and Health”. In: *IEEE Transactions on Affective Computing* 11.2 (2020), pp. 200–213. DOI: 10.1109/TAFFC.2017.2784832.
- [9] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “Simple, Efficient and Convenient Decentralized Multi-Task Learning for Neural Networks”. In: *IDA 2021 - 19th Symposium on Intelligent Data Analysis*. Vol. 12695. Lecture Notes in Computer Science.

- Porto, Portugal: Springer, Apr. 2021. DOI: 10.1007/978-3-030-74251-5\\_4. URL: <https://hal.archives-ouvertes.fr/hal-02373338>.
- [10] Qiang Shen et al. “Federated Multi-Task Attention for Cross-Individual Human Activity Recognition”. In: *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*. Ed. by Lud De Raedt. International Joint Conferences on Artificial Intelligence Organization, July 2022, pp. 3423–3429. DOI: 10.24963/ijcai.2022/475. URL: <https://doi.org/10.24963/ijcai.2022/475>.
- [11] Robert Šajina, Nikola Tanković, and Ivo Ipšić. “Peer-to-peer deep learning with non-IID data”. In: *Expert Systems with Applications* 214 (2023), p. 119159. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2022.119159>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417422021777>.
- [12] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning*. Vol. 37. ICML’15. Lille, France: JMLR.org, 2015, pp. 448–456.
- [13] Jacob Devlin et al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Minneapolis, Minnesota: Association for Computational Linguistics, June 2019, pp. 4171–4186. DOI: 10.18653/v1/N19-1423. URL: <https://aclanthology.org/N19-1423>.
- [14] Ashish Vaswani et al. “Attention is All You Need”. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*. NIPS’17. Long Beach, California, USA: Curran Associates Inc., 2017, pp. 6000–6010. ISBN: 9781510860964.
- [15] Mohammed Baz et al. “Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning”. In: *Sensors* 22 (May 2022). DOI: 10.3390/s22114157.
- [16] Suyu Ge et al. “FedNER: Privacy-preserving Medical Named Entity Recognition with Federated Learning”. In: *ArXiv* (2020). DOI: 10.48550/ARXIV.2003.09288. URL: <https://arxiv.org/abs/2003.09288>.

- [17] Iulia Turc et al. “Well-Read Students Learn Better: On the Importance of Pre-training Compact Models”. In: *arXiv preprint arXiv:1908.08962v2* (2019).
- [18] Aurélien Bellet et al. “Personalized and Private Peer-to-Peer Machine Learning”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, Apr. 2018, pp. 473–481. URL: <https://proceedings.mlr.press/v84/bellet18a.html>.
- [19] Xiangru Lian et al. “Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/f75526659f31040afeb61cb7133e4e6d-Paper.pdf>.
- [20] Hanlin Tang et al. “D<sup>2</sup>: Decentralized Training over Decentralized Data”. In: *Proceedings of the 35th International Conference on Machine Learning* 80.1 (2018), pp. 4848–4856. URL: <http://proceedings.mlr.press/v80/tang18a.html>.
- [21] Michael Blot et al. “Distributed optimization for deep learning with gossip exchange”. In: *Neurocomputing* 330 (2019), pp. 287–296. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2018.11.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231218313195>.
- [22] Mahmoud Assran et al. “Stochastic Gradient Push for Distributed Deep Learning”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 344–353. URL: <https://proceedings.mlr.press/v97/assran19a.html>.
- [23] Manoj Ghuhana Arivazhagan et al. “Federated Learning with Personalization Layers”. In: *ArXiv abs/1912.00818* (2019).
- [24] Xiaoxiao Li et al. “FedBN: Federated Learning on Non-IID Features via Local Batch Normalization”. In: *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

- [25] Othmane Marfoq et al. “Federated Multi-Task Learning under a Mixture of Distributions”. In: *Advances in Neural Information Processing Systems*. Ed. by M. Ranzato et al. Vol. 34. Curran Associates, Inc., 2021, pp. 15434–15447. URL: <https://proceedings.neurips.cc/paper/2021/file/82599a4ec94aca066873c99b4c741ed8-Paper.pdf>.
- [26] J. Mills, J. Hu, and G. Min. “Multi-Task Federated Learning for Personalised Deep Neural Networks in Edge Computing”. In: *IEEE Transactions on Parallel and Distributed Systems* 33.03 (Mar. 2022), pp. 630–641. ISSN: 1558-2183. DOI: 10.1109/TPDS.2021.3098467.
- [27] Yutao Huang et al. “Personalized Federated Learning: An Attentive Collaboration Approach”. In: *CoRR* abs/2007.03797 (2020). URL: <https://arxiv.org/abs/2007.03797>.
- [28] Sebastian Caldas et al. “LEAF: A Benchmark for Federated Settings”. In: *ArXiv* abs/1812.01097 (2018).
- [29] Noa Onoszko et al. “Decentralized federated learning of deep neural networks on non-iid data”. In: *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML*. Vol. abs/2107.08517. 2021.
- [30] Xue Zheng, Parinaz Naghizadeh, and Aylin Yener. “DiPLe: Learning Directed Collaboration Graphs for Peer-to-Peer Personalized Learning”. In: *2022 IEEE Information Theory Workshop (ITW)*. Ed. by Institute of Electrical and Electronics Engineers. 2022, pp. 446–451. DOI: 10.1109/ITW54588.2022.9965838.
- [31] Shuangtong Li et al. “Learning to Collaborate in Decentralized Learning of Personalized Models”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 9756–9765. DOI: 10.1109/CVPR52688.2022.00954.
- [32] Zexi Li et al. “Towards Effective Clustered Federated Learning: A Peer-to-peer Framework with Adaptive Neighbor Matching”. In: *IEEE Transactions on Big Data* (2022), pp. 1–16. DOI: 10.1109/TBDATA.2022.3222971.
- [33] Edvin Listo Zec et al. “Decentralized Adaptive Clustering of Deep Nets is Beneficial for Client Collaboration”. In: *Trustworthy Federated Learning*. Ed. by Randy Goebel et al. Springer International Publishing, 2023, pp. 59–71. ISBN: 978-3-031-28996-5.

- [34] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 0. Toronto, Ontario: University of Toronto, 2009. URL: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
- [35] Robert Šajina, Nikola Tanković, and Darko Etinger. “Decentralized trustless gossip training of deep neural networks”. In: *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. 2020, pp. 1080–1084. DOI: 10.23919/MIPRO48935.2020.9245248.
- [36] StackOverflow. *Stack Overflow Data*. Kaggle. 2018. URL: <https://www.kaggle.com/datasets/stackoverflow/stackoverflow>.
- [37] Erik F. Tjong Kim Sang and Fien De Meulder. “Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition”. In: *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003 - Volume 4*. CONLL ’03. Edmonton, Canada: Association for Computational Linguistics, 2003, pp. 142–147. DOI: 10.3115/1119176.1119195. URL: <https://doi.org/10.3115/1119176.1119195>.
- [38] Ning Ding et al. “Few-NERD: A Few-shot Named Entity Recognition Dataset”. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Online: Association for Computational Linguistics, Aug. 2021, pp. 3198–3213. DOI: 10.18653/v1/2021.acl-long.248. URL: <https://aclanthology.org/2021.acl-long.248>.
- [39] Yonghui Wu et al. “Google’s neural machine translation system: Bridging the gap between human and machine translation”. In: *arXiv preprint arXiv:1609.08144* (2016).
- [40] Sashank J. Reddi et al. “Adaptive Federated Optimization”. In: *International Conference on Learning Representations*. 2021.
- [41] Joel Stremmel and Arjun Singh. “Pretraining Federated Text Models for Next Word Prediction”. In: *Advances in Information and Communication*. Ed. by Kohei Arai. Springer International Publishing, 2021, pp. 477–488. ISBN: 978-3-030-73103-8.

- [42] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: <http://arxiv.org/abs/1412.6980>.

## **C. An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning**

This work was published as: Robert Šajina, Nikola Tanković, Ivo Ipšić, 2024. An Overview of Autonomous Connection Establishment Methods in Peer-to-Peer Deep Learning. IEEE Access, Volume 12, ISSN 2169-3536, DOI: 10.1109/ACCESS.2024.3442014

For clarity, the article has been reformatted, otherwise the content is the same as the published version of work. © 2024 by the author.

<https://ieeexplore.ieee.org/document/10633710>



# 1. Introduction

Decentralized learning has emerged as a promising paradigm in machine learning, particularly in scenarios where data is distributed among numerous agents in a peer-to-peer network. However, real-world data often has non-identical and non-independent (non-IID) distributions across these agents, which makes collaborative learning challenging [1, 2]. The opposite of a non-IID scenario is an IID scenario, wherein the data distribution among agents is identical and independent. In such cases, agents share similar distributions of local data samples. The IID scenario is favorable as it facilitates convergence towards a similar solution for all models, thereby enabling faster convergence and enhanced model performance. Conversely, learning in non-IID data environments impedes the learning processes between agents, resulting in reduced performance of machine learning models [3, 4].

Clusters of agents frequently display comparable non-IID attributes, which promote similarities within each cluster while simultaneously revealing significant disparities across different groups or individual agents. This similarity is also seen in parameter-server Federated Learning methods, where clients are put into coherent groups using clustering techniques to improve personalization [5–9]. Notably, in Federated Learning, characterized by a centralized server orchestrating the learning procedure, the identification and estimation of agent clusters are more feasible, as emphasized in the aforementioned studies. In decentralized peer-to-peer systems, on the other hand, there is not a single entity driving the whole process, so each agent is responsible for its own learning path. This makes it more difficult to elicit and manage agent clusters during the learning process.

To mitigate the challenge of identifying similar agent clusters, recent research studies have proposed methodologies enabling agents to autonomously ascertain their potential neighboring peers, thereby departing from random assignment strategies. The studies' empirical investigations, which focused on synthetic non-IID environments, show that these methods can be used with a variety of non-IID properties. Notably, these methods exhibit superior performance compared to a baseline approach that represents random organization of connections among agents.

This research evaluates established methodologies for the adaptive establishment of peer connections in two distinct learning tasks. The first task involves image classification within a

synthetically generated environment, while the subsequent task addresses next-word prediction using a dataset gathered at the user level, which represents a realistic non-IID environment.

The objective of this study is to evaluate the efficacy of adaptive peer connection establishment in various synthetic and realistic non-IID peer-to-peer environments. This evaluation is compared against the performance of randomly assigned peer connections between agents, with the aim of identifying the optimal method for autonomous peer connection establishment across all non-IID scenarios.

The primary research questions guiding this study are as follows:

- RQ1** How can autonomous personalized peer connections be created to optimize learning outcomes in peer-to-peer learning environments with non-IID data distributions?
- RQ2** What are the impacts of different peer connection methodologies on the efficiency of communication within the network?
- RQ3** What are the impacts of different peer connection methodologies on the centralization tendencies within the network?
- RQ4** Which methods demonstrate the best balance between communication efficiency, and overall improved learning outcomes for agents?

By answering these research question, our contributions are as follows:

- overview of the methods and algorithms used for adaptive and personalized establishment of peer connections in decentralized deep learning,
- evaluation and comparison of studied methods in synthetic and realistic non-IID environments,
- pareto analysis of the best methods based on the achieved accuracy and the number of messages exchanged per agent.

Access to the code implementation and detailed results of the conducted experiments are available at the publicly accessible repository located at [https://github.com/fipu-lab/p2p\\_bn](https://github.com/fipu-lab/p2p_bn).

The rest of this paper is organized as follows: Section 2 provides more details on non-IID data, decentralized learning and network topologies used in decentralized learning. Section 3

presents related studies, while Section 4 describes methods that enable autonomous agent peer connection creation. Section 5 outlines the selection criteria for including relevant autonomous adaptive peer connection creation methods in this study and details the methodology used to evaluate and compare the studied methods. The results are presented and analyzed in Section 6. Section 7 concludes the study and offers suggestions for future research directions.

## 2. Background

As previously stated, non-IID data setting is a key differentiator for peer-to-peer deep learning, so this section will briefly describe the key characteristics of non-IID data. It will also describe the key difference between decentralized and federated learning and lay the basics for an adaptive and personalized peer selection process.

### 2.1. Non-IID data

The agent's local data encompasses all data gathered by the agent. This data encompasses various types of information: images, texts, sensor values, location data, etc. Since agents are all individual devices in different contexts, they generate and own different data [10], which happens to be non-independently and non-identically distributed (non-IID). On the other hand, having IID data at the agent level means that each batch of data used for an agent's local model update is statistically identical between agents, as if it were uniformly drawn from the entire training dataset, which is a union of all agents' local datasets.

Consider a dataset of numeric values. A sample of size  $n$  consists of  $n$  random values:  $\{X_1, X_2, \dots, X_n\}$ . This sample is IID if the random values have the following properties:

**Independent:** The random values  $X_1, X_2, \dots, X_n$  are independent, meaning that the occurrence of any  $X_i$  does not depend on any other  $X_j$ .

**Identically Distributed:** The random values  $X_1, X_2, \dots, X_n$  are from the same population and thus have the same distribution or cumulative distribution function (CDF)  $F$ :

$$F_{x_1} = F_{x_2} = \dots = F_{x_n} = F_x. \quad (1)$$

Training on IID data is easier and faster because all model updates converge toward one global solution. Conversely, when agents have non-IID data, one global model solution is not optimal for all agents, making this case a challenging learning problem. Unfortunately, non-IID data is common occurrence in decentralized environments [1, 11]. The data between different agents may vary in size, label mapping, features, and other data properties, which makes it difficult for multiple agents to work together to train neural network (NN) models. Where possible, a generative adversarial network (GAN) may be employed to counterfeit data directly on agents, thereby enriching their local dataset both in size and data diversity [12].

## 2.2. Decentralized learning

In a decentralized network, agents communicate only with their neighbors or peers, adhering to a network topology such as ring, sparse, or fully connected topology (see Figure 1). The number of the agent’s peers is predetermined in ring and fully connected topologies. The least restrictive topology is one with sparse connections, where agents can have as many peers as they like. Communication in network topology can be directed (asymmetric), where an agent receives messages from one set of peers and sends messages to another set [4, 13], or undirected (symmetric), where agents send and receive messages from the same set of peers [14, 15]. A notable exception are gossip methods [16, 17], where an agent sends its messages to a random peer in the network, disregarding a specific network topology. The information exchanged between agents generally comprises the agent’s model parameters, which are often aggregated by immediately averaging each received model with the local model parameters or by aggregating all models once they have been received. When combining received model parameters, different aggregation strategies can be used, such as those based on trust [18], model accuracy [19, 20], or other variables. However, this study focuses on assessing the advantages of peer connection design; thus, a simple averaging method is employed for aggregating the model parameters. An abstract decentralized learning process is outlined in Algorithm 1.

Consider the example of learning a next-word prediction (NWP) task in a decentralized peer-

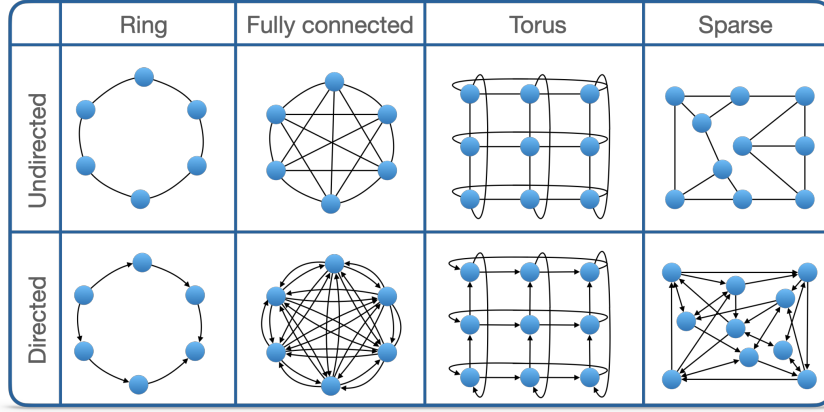


Figure 1: Examples of network typologies in undirected and directed communication.

to-peer environment. The objective of NWP task is to predict the next word given a sequence of previous words, necessitating a dataset of sentences or paragraphs before model training begins. Devices such as smartphones and laptops, serving as agents, possess user-typed text that can be collected for local model training. Each agent maintains its own distinct dataset and model instance. Once these prerequisites are met, the agent can participate in the peer-to-peer learning process. As outlined in Algorithm 1, the initial step at the agent level involves local training of the model on the agent’s own text data. This training imparts some knowledge to the model, though its performance may remain suboptimal due to the limited amount of locally available training data [4]. To enhance model performance, the agent communicates with other agents engaged in the same task, exchanging only the model parameters in a peer-to-peer manner, often following a predetermined network topology. Peer-to-peer information exchange between

---

**Algorithm 1** Agents abstract training process

---

**Require:**

- Initialize  $\eta > 0$ , agents  $A$ , communication matrix  $W$ , number of local batch iterations  $E$
- $x_i = 0$  for all agents  $i \in A$  ▷ Initialize all agents models to identical value
- 1: **repeat** for agent  $i \in A$  ▷ In parallel
- 2:     **for**  $e = 0, 1, 2, \dots, E$ , at agent,  $i$  **do**
- 3:          $\xi_i \sim D_i$  ▷ Sample new mini-batch from local distribution
- 4:          $x_i = x_i - \eta F_i(x_i; \xi_i)$  ▷ Train model on batch
- 5:         SEND( $x_i W_{ji}$ ) ▷ Send model to peers
- 6:         RECEIVE( $x_j W_{ij}$ ) ▷ Receive model from peers
- 7:          $x_i = \text{AGGREGATE}(x_j W_{ij})$
- 8:     **until** Maximum iteration reached

**Output:**  $\frac{1}{N} \sum_{i=1}^N x_i, N = |A|$  or  $x_i \forall i \in A$  ▷ Output is either one global model produced by averaging all models or personal model for each agent

---

agents is extensively utilized in blockchain applications [21–23]. After receiving updates from peers, each agent integrates these updates into its local model, benefiting from the collective knowledge within the network. This collaborative aggregation improves the model’s ability to predict the next word across diverse contexts. As each agent’s model becomes more informed through peer interactions, the accuracy of local model’s next-word predictions improves. Once the model converges, it may be used by the agent’s owner, such as person owning a smartphone or a laptop.

### 2.3. Network topology

The connection between agents is captured through a communication graph  $G = (\llbracket N \rrbracket, E, W)$  where  $\llbracket N \rrbracket = \{1, \dots, N\}$  is a set of all nodes in the network,  $E \in \llbracket N \rrbracket \times \llbracket N \rrbracket$  is the set of edges, and  $W \in \mathbb{R}^{N \times N}$  is a nonnegative weighted matrix. The weight of edge  $(i, j) \in E$  is given by  $W_{ij}$  with the convention  $W_{ij} = 0$  if  $(i, j) \notin E$  or  $i = j$ . An agent  $i$  only sends messages to agent  $j$  if  $W_{ij} > 0$ , which means that agent  $i$  communicates with peers  $N_i = \{j : W_{ij} > 0\}$  without knowledge of other peers in the network and operates without synchronization with non-connected peers ( $W_{ij} = 0$ ). Commonly, the communication matrix is comprised of fractions rather than integers. These fractions may signal trust between agents or are simply a decentralized method to control each agent’s contributions. In the majority of cases, the parameter  $W$  falls within the range of  $W_{ij} \in [0, 1]$ . The resulting communication matrix shows the connection between agents. Consider the next communication matrix,  $W \in \mathbb{R}^{4 \times 4}$ :

$$W = \begin{pmatrix} 0 & 1/2 & 0 & 1/2 \\ 1/2 & 0 & 1/2 & 0 \\ 0 & 1/2 & 0 & 1/2 \\ 1/2 & 0 & 1/2 & 0 \end{pmatrix} \quad (2)$$

Column indices of matrix  $W$  correspond to sending nodes, while row indices correspond to receiving nodes. For the node with index 0,  $\text{Node}_0$ , sending indices (column) are  $\{0, 1/2, 0, 1/2\}$ , meaning that  $\text{Node}_0$  sends its message to nodes  $\text{Node}_1$  and  $\text{Node}_3$ . Row at index 0 shows that  $\text{Node}_0$  receives messages from both  $\text{Node}_1$  and  $\text{Node}_3$ . This matrix example demonstrates an undirected/symmetric communication graph (see figure 2). The opposite is directed/asymmetric

communication, in which an agent can send messages to a set of peers but receive messages from a completely different set of peers. In a directed graph, an agent has in-peer if  $(i, j) \in E$  and out-peer if  $(j, i) \in E$ . The out-peer set denotes the peers to whom the agent transmits messages, whereas the in-peer set comprises the peers from whom messages are received. Typically, the number of in and out peers is equivalent, which is commonly referred to as the node's in-degree and out-degree.

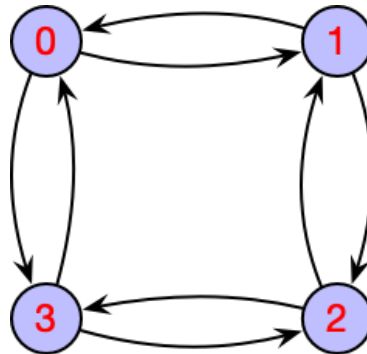


Figure 2: Symmetric/undirected communication graph with four nodes.

## 2.4. Adaptive and personalized autonomous peer connection establishment

Instead of having a fixed and pre-specified topology, connections between agents can be dynamically established during the learning process to identify the most similar peers, thereby enhancing learning outcomes such as faster model convergence and higher local model accuracy [24]. In this scenario, each agent typically retrieves models from more peers than it would in a fixed topology, but only aggregates models from the top N most similar peers. The pull-gossip method [25] is generally employed to request models from peers. Peers with the most similar properties, such as the most similar model parameters, are marked as the most similar and used for subsequent communication rounds, along with newly sampled peers. This approach, although potentially increasing the communication load and complexity compared to fixed topology communication, offers the advantage of identifying similar peers, thereby potentially enhancing learning outcomes. Additionally, as methods generally utilize the pull-gossip method, there can be cases where a large number of agents pull the model from the same peer or a small subset of peers, resulting in a highly imbalanced communication load at the agent

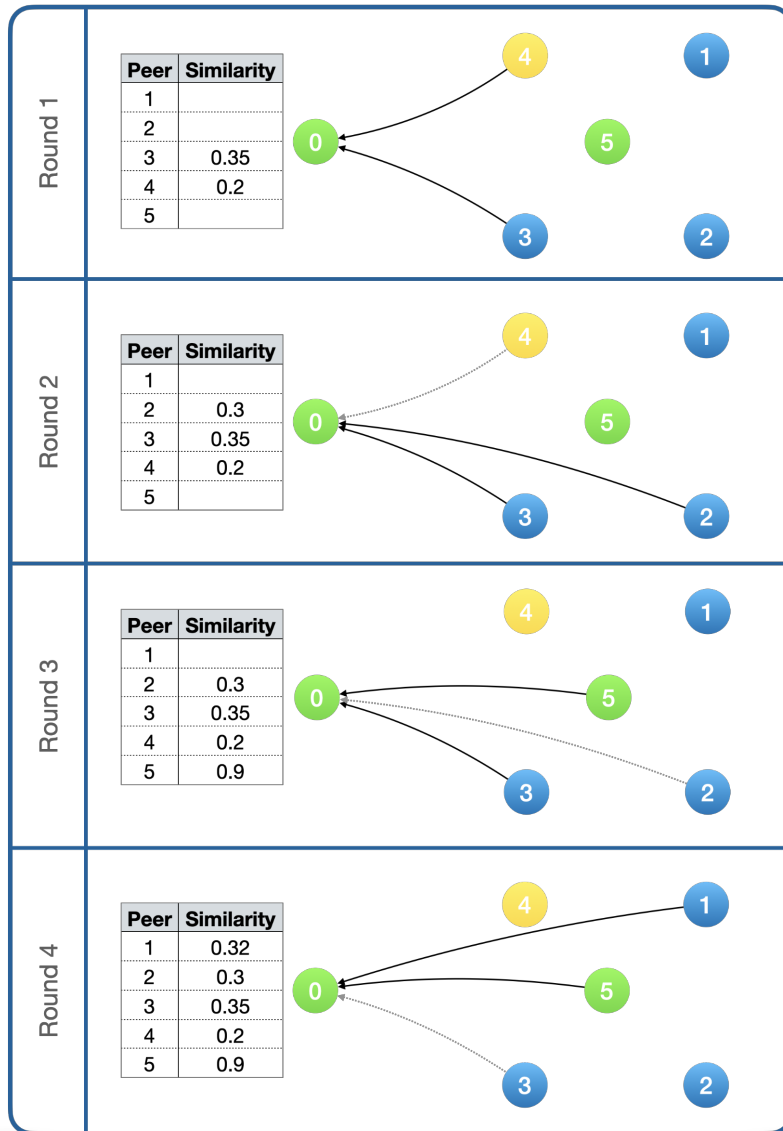


Figure 3: Example of adaptive peer selection in a network of six agents. The color of each agent indicates its similarity. Only behaviour of agent 0 is demonstrated, recognizing that all agents in the network follow similar steps. Agent 0, in each communication round, randomly selects peers and aggregates the model updates from the peer with the highest similarity. In each subsequent round, the agent receives models from a peer with highest known similarity and one additional unknown random peer. This process illustrates how an agent autonomously discovers and maintains communication with the most similar peers, enhancing learning outcomes.

level and high centralization properties of the network.

Consider a network of six agents as depicted in Figure 3, where the colors represent agent similarity. For simplicity, we focus on the behavior of agent 0, recognizing that all agents in the network follow similar steps. In this scenario, the agent aggregates updates from only one peer. In a fixed topology, agent 0 would consistently receive model updates from a single



predetermined peer, which are then aggregated with its local model. However, in an autonomous peer selection scenario, agent 0 must communicate with multiple peers to determine the most suitable ones for aggregation, ultimately selecting the top N similar peers, in this case, the top one. During the first communication round, agent 0 randomly selects two peers (peers 3 and 4) and retrieves their models. Although peer 3 has a low similarity, it is higher than that of peer 4, so agent 0 aggregates the model from peer 3 with its local model. In the second round, agent 0 communicates with peer 3 again and adds a new random peer, peer 2. Since peer 2 has a lower similarity than peer 3, the model from peer 3 is aggregated once more. In the third round, agent 0 selects a new random peer, peer 5, in addition to the most similar known peer, peer 3. Peer 5 exhibits very high similarity, surpassing peer 3, thus the model from peer 5 is aggregated. In the fourth round, agent 0 continues with peer 5 and adds another random peer, peer 1, who has a lower similarity than peer 5. Consequently, agent 0 will continue to communicate with peer 5 until new agents enter the network. This example illustrates the general approach and methodology for autonomously selecting the most similar peers. The number of models an agent aggregates is a tunable parameter that can be adjusted, as well as the number of new random peers sampled in each round. Additionally, there may be constraints on the duration of this discovery process, after which the agent communicates only with the most similar peers. It is important to note that the similarity between an agent and its peers can be calculated using various methods, as described in Section 4.

### 3. Related work

Several recent studies have highlighted significant advancements and challenges in the exploration of decentralized peer-to-peer learning and Federated Learning (FL).

A comprehensive overview of the recent developments in Federated Learning and decentralized peer-to-peer methods is provided by Kairouz *et al.* [26]. Their work outlines an extensive collection of open problems and challenges, shedding light on the future directions and hurdles that need to be addressed to advance these fields. One of the major challenges is non-IID environments.

Aledhari *et al.* [27] offers a detailed examination of Federated Learning, focusing on enabling software and hardware platforms, protocols, and practical applications. Their study covers real-life use cases, demonstrating FL's practical implementation and potential benefits in various domains.

The unique challenges of running machine learning and deep learning models on edge devices in a distributed manner have been investigated by Filho *et al.* [28]. This study highlights how various techniques are adapted or specifically designed to function on these resource-constrained devices. Fundamental processes such as caching, training, inference, and offloading on edge devices are thoroughly analyzed, providing insights into the benefits and drawbacks of these strategies.

Witt *et al.* [29] conducted a systematic literature review to evaluate FL frameworks that integrate blockchain technology to decentralize the learning process while employing reward mechanisms to incentivize participation. Their analysis reveals the potential of combining these technologies to enhance the robustness and security of FL systems.

The challenges of resource sharing in unstructured peer-to-peer networks, which pose unique difficulties in resource location due to their decentralized nature, have been addressed by Shoab *et al.* [30]. They propose an intelligent neighbor selection (INS) algorithm using Q-learning, which enhances retrieval effectiveness and reduces search costs by making informed decisions on neighbor selection. Their results show significant improvements over traditional methods, demonstrating the effectiveness of the INS algorithm in optimizing resource sharing in decentralized networks. This study is closely related to ours as it focuses on peer selection optimization to enhance retrieval effectiveness, while our study focuses on selecting optimal peers to optimize the training of deep neural networks in non-IID scenarios.

Our research diverges from existing studies by focusing specifically on the autonomous personalized creation of peer connections in peer-to-peer learning within non-IID environments, an area not thoroughly explored in previous works.

## 4. Establishing personalized connections in peer-to-peer networks with non-IID data

To answer **RQ1**, we analyzed existing methods for autonomous and personalized peer connection creation. The methods were selected according to the criteria provided in Section 5.1.

One approach to mitigating the effects of non-IID data involves assuming the existence of groups of agents with similar data distribution properties and organizing the learning process as a multi-task learning objective. Multiple agent groups learn different tasks in multi-task learning towards an identical objective. Due to data heterogeneity, it has been demonstrated that separating agents into clusters can yield favorable learning outcomes [31, 32]. In this scenario, all agent clusters learn the same problem but communicate only with similar peers, thereby reducing the negative impact of heterogeneous data. A similarity metric is employed to group peers together. This metric can be based on metadata [31], gradients [7, 8, 33], model weights [6, 7, 34, 35], or local data model loss [20, 36–39]. Metadata may include information such as the geolocation or language of an agent or information from the training data, such as the personality and gender of the user [31]. Additionally, data distribution can serve as metadata when designing topologies with a data distribution skew [40]. Similarity between two gradients or model weights can be calculated using a distance metric, such as cosine or Euclidean distance. It has been shown that agents with similar data distributions exhibit a small Euclidean distance and a considerable cosine similarity in their gradients or model weights [38]. Finally, model similarity can be assessed using the loss value obtained from the loss function on local data, where a model with a lower loss value is considered to exhibit a higher degree of similarity. Personalized peer connections between agents have already been considered in the context of multi-task learning [24, 31, 41]. This approach ensures that agents with similar learning tasks and data distributions are more effectively grouped, enhancing overall learning outcomes.

Building on the conclusions drawn from multi-task learning in non-IID scenarios, where learning outcomes are enhanced by grouping agents into clusters that collaborate exclusively within the cluster, a new research direction has emerged. This direction involves enabling agents to collaborate with peers who share similar properties in an autonomous and person-

alized manner. Learning outcomes are expected to improve by collaborating with peers who have similar interests and data distributions, as evidenced by the success of multi-task learning and preliminary results from research studies addressing this scenario (presented below). The following subsections analyze personalized and autonomous peer creation methods, categorized by model-based and data-based similarity approaches.

#### 4.1. Model similarity

**Similarity based on empirical model loss.** In peer-to-peer deep learning, empirical model loss on local data has been used most often to measure how similar models are to each other. Within this context, several notable approaches, namely DAC [37], DiPLe [42], L2C [43], PANM [38], and PENS [36] have effectively employed empirical model loss as the fundamental metric for quantifying model similarity. Each of the aforementioned approaches exhibits distinct variations in the manner in which the similarity measure is employed. However, a common characteristic among all these approaches is the absence of a predetermined communication mixing matrix. Instead, each agent engages in communication with randomly selected peers, thereby implying that every agent possesses a comprehensive overview of the entire network.

In each round of the PENS approach, a specific number of peers, denoted as  $n_{sampled}$ , are selected and their models are retrieved. These models are then used to assess empirical loss on the local training data. Among the sampled peers, only the top  $m$  peers with the lowest empirical loss values, indicative of similar data distributions, are chosen as potential peers and their models are aggregated with the agent’s local model. After a predetermined number of *rounds*, each agent continues to randomly sample  $m$  peers from a pool consisting of those peers who have been selected as potential peers more frequently than expected. The expected frequency is calculated as  $m/N \times rounds$ , where  $N$  represents the total number of agents in the network.

The PENS method laid the groundwork for DAC, which maintains a comprehensive list of similarity values derived from the reciprocal values of empirical loss measurements. These similarity values serve as the basis for determining the probability of peer sampling in subsequent rounds. DAC uses an adaptive strategy to deal with the problems that come up with large peer-to-peer networks. During communication, it asks sampled peers for their similarity values.

This means that DAC covers not only the peers with whom direct communication has occurred but also extends its reach to peers that were previously unexplored or had not been directly communicated with.

Similarly to the PENS approach, the PANM method requires a predetermined number of *rounds* to determine similar peers by considering peers from the previous round along with new peers, but only aggregating models from the top  $m$  performing models. In the second stage, PANM samples  $m$  peers from a peer bag, which is updated periodically every  $\tau$  rounds by sampling  $n_{sampled}$  peers from the peer bag and  $n_{sampled}$  randomly sampled peers. The expectation maximization of the Gaussian mixture model is used to determine new peer bag based on the model similarities expressed as empirical losses on training data. The PANM version that utilizes empirical loss as a measure of similarity is referred to as PANMLoss.

L2C approach utilizes a learnable vector of mixing weights that is maintained by each individual agent. After requesting models from peers, an agent updates the mixing weights vector based on the empirical loss of each received model on the validation data. In the first  $T_0$  rounds, each agent pulls models from all peers, after which each agent stops communicating with  $K_0$  peers corresponding with the smallest mixing weight  $w_{i,j}$ .

In DiPLe, each agent maintains a mixing weight vector  $w$ . At the beginning of training, an agent communicates with all peers in the network. After pooling models from peers where  $w > 0$ , each agent utilizes a *bisection method* with threshold  $\epsilon$  to minimize the empirical risk, where if  $w_{i,j} < \frac{\epsilon}{2}$ , then  $w_{i,j} = 0$ .

**Model parameter similarity.** PANM also introduced PANMGrad, a more efficient method of calculating model similarity, whereby a cosine similarity is utilized to express similarity between gradients. However, the PANMGrad method requires both current gradients  $g$  and accumulated weight deltas  $h$  to calculate the similarity as follows:

$$S_{i,j} = \alpha \cos \theta^g + (1 - \alpha) \cos \theta^h, \cos \theta^g = \frac{\langle g_i, g_j \rangle}{\|g_i\| \cdot \|g_j\|}, \quad (3)$$

$$\cos \theta^h = \frac{\langle h_i, h_j \rangle}{\|h_i\| \cdot \|h_j\|}. \quad (4)$$

**Model prediction similarity.** AUCCCR algorithm [44] can infer the number of clusters from the data without requiring a pre-determined number of clusters. Even though AUCCCR is a

clustering method similar to K-Means, we can also consider it a clustering method based on parameter similarity. In their experiments on the MNIST dataset [45], the authors obtained the average output of agents’ models for each of the ten classes in the MNIST dataset, using the first 1000 images from the test split. The average outputs of the ten classes were concatenated in one  $\mathbb{R}^{100}$  vector. These vectors were used as the input to the AUCCCR clustering algorithm, with the Euclidean distance used as the measure of distance between two vectors. The output of the algorithm is clusters of agents with similar model predictions.

## 4.2. Dataset similarity

D-Cliques [40] method designs a communication matrix based on label distribution between different local datasets which groups agents into cliques with a maximum size of  $M$ . Cliques are formed by iterative Greedy swap approach that randomly swaps a pair of agents between cliques if the distribution skew is smaller after the swap. Let  $p_C(y) = \frac{1}{|C|} \sum_{i \in C} p_i(y)$  denote the distribution of label  $y$  in clique  $C$  and  $p(y) = \frac{1}{N} \sum_{i \in N} p_i(y)$  the distribution of label  $y$  in global distribution. The *skew* of clique  $C$  is measured as absolute difference of  $p_C(y)$  and  $p(y)$ :

$$skew(C) = \sum_{l=1}^L |p_C(y=l) - p(y=l)| \quad (5)$$

To ensure a global consensus and convergence, the D-Cliques approach introduces inter-clique connections between a small number of agent pairs that belong to different cliques.

In summary, the answer to **RQ1** is that the autonomous personalized peer connections are based on the pull-gossip communication method [25], wherein agents pull model parameters from peers. The similarity between agents can be based on empirical model loss obtained on a subset of data, model prediction similarity, and dataset similarity.

## 5. Methodology

### 5.1. Methods selection criteria

In this study, we employed a systematic approach to selecting methods and algorithms for evaluating autonomous personalized peer connection creation. The selection process involved the following criteria to ensure fairness and relevance. We conducted a comprehensive literature search using Scopus and WoS (article titles and abstracts) with the query: *TITLE-ABS((p2p OR peer OR agent OR decentralised OR decentralized) AND (learning OR machine learning OR artificial intelligence OR training) AND (personalized OR autonomous OR adaptive OR collaboration OR cooperation OR cluster))*. This query was designed to capture a broad range of relevant methodologies and algorithms related to peer-to-peer and decentralized learning systems within the context of autonomous and adaptive creation of peer connections in non-IID environments. By limiting the results to conference papers and journal articles, 4,781 relevant documents were identified as results for Scopus, and 18,945 for WoS. We also search through Google Scholar using similar terms and conditions. Based on article titles and abstracts, we excluded studies that are primarily concerned with Federated Learning or those that assume a fixed network topology. These studies were not considered relevant to our research focus, as they do not align with the dynamic and adaptive nature of peer connection creation that this study aims to explore. Furthermore, we also excluded results that do not align with the core objectives of this study, which emphasizes the adaptation and personalization of peer connections in a decentralized learning environment, focusing on non-IID data and with the support for deep neural network models. To ensure thorough coverage, we also employed forward and backward snowballing techniques. This involved examining the references of selected papers (backward snowballing) and checking citations of these papers (forward snowballing) to identify additional relevant studies. By adhering to these criteria, we ensured that the methods evaluated in this study are both relevant and capable of addressing the key aspects of autonomous and personalized peer connection creation in non-IID environments.

## 5.2. Evaluation methodology

This study seeks to investigate and evaluate whether a substantial advantage exists in a connection arrangement through any of the specified methodologies. Previous research generally created a multi-task heterogeneous environment by image rotation or by creating agent groups with disjoint label sets. For our analysis, all the methods mentioned are compared on a computer vision task and a natural language processing task.

Agents’ learning process was simulated in memory in a synchronised manner since all approaches presume a synchronised network. As a result, a cyclical process develops in which a communication step follows each local training step, with these two phases iteratively repeating. All approaches presume a pull-gossip [25] communication, which was also utilized in the AUCCCR and baseline approaches. Once an agent pull models from its peers, a new model is formed by aggregating all received models  $x_i = \sum_{\{w_{ij}>0\}} w_{ij}x_j$ . Agent’s local test accuracy is measured after performing local SGD updates (i.e. an epoch) for all evaluated methods.

**Datasets.** The evaluation of the assessed methodologies encompassed two distinct tasks: image classification and next-word prediction (NWP). CIFAR-10 [46] dataset was used for the image classification task with various image transformations that simulated different non-IID environments. The goal was to create non-IID clusters of agents that were defined by using different transformation rules, such as label permutation, data segmentation, and image rotation. CIFAR-10 consists of 60,000 32x32 color images in 10 different classes, with 6,000 images per class. The dataset is divided into 50,000 training images and 10,000 test images. In the simulations, the dataset was divided uniformly between agents for both training and test splits. Depending on the scenario, certain group of agents applied specific image transformation rules, thereby creating synthetic non-IID environments in which the data distribution between groups of agents is different.

The NWP experiments utilized the publicly accessible Reddit dataset [47], comprising posts and comments from diverse individual internet users. The dataset is inherently non-IID, as users in different subreddits often discuss different topics and share different types of content. In the experiments, each agent’s dataset comprised all comments from a unique user. The datasets were then partitioned into training, validation, and test subsets in a 60%-20%-20% split [47]. Following established experimental protocols [4, 48–52], the vocabulary size was



limited to the 10,000 most common words, resulting in 10,000 output classes for the model. Words outside this vocabulary were classified as out-of-vocabulary tokens and excluded from prediction accuracy calculations. Sentences were segmented into sequences of 10 words [4], with the model’s objective being to predict the subsequent word. For sequences shorter than ten words, padding was applied on the left-hand side.

The accompanying metadata, such as subreddit names, was considered when creating groups of users sharing similar interests. Firstly, users that exclusively participated in a single subreddit were identified. Filtering the users based on a minimum of 90% metadata-related texts revealed that the four most popular subreddits were *politics*, *leagueoflegends*, *nba* and *Bitcoin*. Each derived cluster consists of users with common interests within the cluster, while exhibiting dissimilarity in interests with respect to all other clusters. For instance, users who solely post in the *politics* subreddit forum may not share any common interests with users who exclusively participate in the *leagueoflegends* forum.

**Models.** A simple CNN model was used to perform experiments on the CIFAR-10 dataset, as shown in Table 1. Although the size and architecture of this network may not be considered state-of-the-art for visual classification tasks, it possesses adequate capacity to facilitate the comparison in the conducted experimental analysis. The RNN model architecture presented in [4] was used for the NWP experiments. In order to perform a next-word prediction task, sentences were segmented into sequences of ten tokens, with shorter sequences padded as necessary. A batch size of 32 was employed for the image classification task, and a batch size of 50 for the NWP task. The Adam optimizer [53] was utilized by all agents, with a learning rate of  $1 \times 10^{-3}$  for the image classification task, and  $5 \times 10^{-3}$  for the NWP task.

Table 1: Model architecture used for the CIFAR-10 experiments.

| Layer        | Parameters  |
|--------------|---|
| Conv2D       | $filters = 6, kernel\_size = 5,$<br>$activation = relu$       |
| MaxPooling2D | $pool\_size = 3$  |
| Conv2D       | $filters = 16, kernel\_size =$<br>$5,$<br>$activation = relu$ |
| MaxPooling2D | $pool\_size = 3$  |
| Flatten      |   |
| Dense        | $units = 128, activation =$<br>$relu$                         |
| Dense        | $units = 10, activation =$<br>$softmax$                       |

**Baselines and communication parameters.** A fixed random *sparse* topology was used as a baseline environment in which all agents communicated mutually, regardless of the agent’s cluster affiliation. Additionally, results of *oracle* training wherein agents only collaborate with fixed peers that share the same data properties (e.g. image rotation) are provided. The number of neighbor peers was set to three and specific parameters of individual methods were adjusted accordingly. For PENS and PANM, the number of first stage *rounds* was set to 100 as in [36] and [38], and the  $n_{sampled}$  number of sampled peers was set to 6 with the parameter *top\_m* set to three. In accordance with the configuration employed by the authors, communication was terminated with 96 peers (denoted as  $K_0$ ) after 10 rounds (designated as  $T_0$ ) within the L2C approach. A *fully connected* inter-clique topology was used with D-Cliques as it achieved the best results in the experiments [40]. In a *fully connected* inter-clique topology, each clique has exactly one edge connection with each of the other cliques, spreading these additional edge connections equally among the agents of a clique. Following the experiments conducted in the D-Cliques study, maximum clique size  $M$  was set to 10. In the AUCCCR method, inter-agent connections were randomly established exclusively within individual clusters, with no connectivity established between agents situated in distinct clusters.

**Metrics.** Average User model Accuracy (UA) [54] metric was used to measure the overall learning process performance. UA measures the average accuracy across all agents on their local test data and can be expressed as:

$$UA = \frac{1}{n} \sum_{i=1}^n acc_i \quad (6)$$

where  $acc_i$  is the prediction accuracy of model  $i$  on local test dataset  $i$  (both owned by agent  $i$ ). Prediction accuracy is calculated as the fraction of correct predictions (TP and TN) over total predictions (TP, TN, FP, and FN):

$$acc_i = \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i}. \quad (7)$$

Considering the distinctive peer selection protocols employed in the examined methods, a communication analysis was conducted to evaluate the equitable distribution of communication load and answer **RQ2**. To quantify this communication equilibrium, the Gini coefficient, a commonly employed inequality measure in economics and social statistics, was used to an-

swer the **RQ3**. The Gini coefficient, already previously utilized to measure centralization in decentralized systems [55–57], spans from zero, denoting perfect equality, to one, indicating near-maximal centralization where all communication is directed to a single peer. Finally, the Pareto analysis was used to discern optimal solutions within the conducted experiments and answer **RQ4**.

## 6. Experiments

The experiments involved varying numbers of agents, ranging from 100 to 200, based on specific experimental conditions. The learning process concluded with the attainment of accuracy convergence. Each experiment was run three times, and the resulting average was considered the result. Experiments employing two and four clusters were conducted for the image classification and next-word prediction task. Within the synthetic non-IID environment experiments involving the image classification task, agents shared identical data distribution and transformation characteristics within each cluster. Conversely, within the realistic non-IID environment experiments involving the next-word prediction task, the categorization of agent clusters was predicated upon the agent’s pertinent metadata, the subsidiary thread for the Reddit dataset.

### 6.1. Results in a synthetic non-IID environment

Agents within a cluster were provided with a customized CIFAR-10 dataset to simulate a synthetic non-IID environment. The customized dataset encompassed transformations such as image rotation [5, 36, 38], and label-swapping [7, 38] or modifications to data partitioning [32, 37, 58]. In experiments involving two clusters, half of the agents were provided with unmodified CIFAR-10 data, whereas the other half received modified CIFAR-10 data. Similarly, in experiments involving four clusters, a quarter of the agents were assigned regular CIFAR-10 data, while the remaining three quarters were distributed into distinct groups, each of which received differently altered CIFAR-10 data. An example of data for two agents from the exper-

iments involving two clusters and rotations of  $0^\circ$  and  $180^\circ$  is illustrated in Figure 4. The agent belonging to the first cluster has images without any rotation applied, while the agent from the second cluster has all images rotated by  $180^\circ$ . It is important to note that referencing an agent as part of a cluster solely indicates its data distribution and heterogeneity. The agents’ responsibility remains to autonomously create personalized peer connections among themselves, regardless of the agents’ cluster.

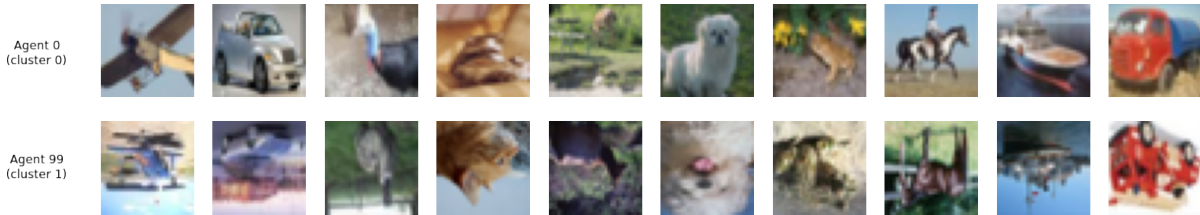


Figure 4: Example datasets for two agents from the experiments involving two clusters with rotations of  $0^\circ$  and  $180^\circ$ . The first agent’s dataset includes images with no rotation, while the second agent’s dataset consists of images rotated by  $180^\circ$ .

Each experiment was conducted with two different numbers of training examples  $T_s$  available to each agent. Specifically,  $T_s$  was set to 100 for experiments involving agents with limited local data, while  $T_s$  was set to 400 for experiments involving agents with a large volume of local data. It is important to note that all experiments involved 100 agents, with 50 agents per cluster in two-cluster experiments and 25 agents per cluster in four-cluster experiments.

**Image rotation.** The image rotation experiment aims to evaluate and compare studied methods

Table 2: Results on the CIFAR-10 dataset, with rotations  $\{0^\circ, 180^\circ\}$  applied in the two-cluster environment and rotations  $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$  applied in the four-cluster environment. Highlighted results indicate performance surpassing the *Sparse* baseline. Methods that outperformed the *Sparse* baseline in all scenarios are also emphasized.

| Method    | $\{0^\circ, 180^\circ\}$ |              | $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$ |              |
|-----------|--------------------------|--------------|---|--------------|
|           | $T_s=100$                | $T_s=400$    | $T_s=100$                                     | $T_s=400$    |
| Oracle    | 43.80                    | 49.44        | 41.52   | 46.79        |
| Sparse    | 40.94                    | 46.16        | 36.41   | 41.96        |
| AUCCCR    | <b>41.33</b>             | <b>49.22</b> | 33.85   | <b>42.61</b> |
| DAC       | 40.61                    | <b>47.98</b> | 36.32   | <b>42.61</b> |
| D-Cliques | 33.69                    | 27.16        | 28.97   | 16.77        |
| DiPLe     | 39.51                    | <b>49.35</b> | 35.50   | <b>44.80</b> |
| L2C       | 33.05                    | 42.83        | 29.11   | <b>42.12</b> |
| PANMGrad  | <b>41.12</b>             | <b>49.62</b> | <b>36.56</b>                                  | <b>45.48</b> |
| PANMLoss  | <b>41.11</b>             | <b>49.67</b> | <b>36.51</b>                                  | <b>46.17</b> |
| PENS      | <b>41.29</b>             | <b>48.06</b> | <b>36.46</b>                                  | <b>42.74</b> |

in an environment where agents own different features (rotation) associated with the same label [2]. In the experiments involving two distinct clusters, rotations  $\{0^\circ, 180^\circ\}$  were used to alter each cluster’s data. Agents belonging to the first cluster were allocated the unaltered data, while agents within the second cluster were provided images subjected to a  $180^\circ$  rotational transformation. For the experiments involving four clusters, rotations  $\{0^\circ, 90^\circ, 180^\circ, 270^\circ\}$  were used to alter agents’ data within each cluster. Table 2 summarizes the results for all evaluated methods. As expected, agents achieved higher accuracy when using the *oracle* communication topology compared to the *sparse* topology. We can conclude that randomly forming connections between peers is not beneficial in the context of image rotation experiments. Therefore, evaluated methods should surpass results obtained in the *sparse* experiments to be deemed effective. In the two-cluster experiments, AUCCCR, PANMGrad, PANMLoss, and PENS outperformed the *sparse* baseline in both limited ( $T_S = 100$ ) and extensive ( $T_S = 400$ ) training set scenarios. DAC and DiPLe achieved superior performance only in the extensive training set scenario, while D-Cliques and L2C performed substantially worse than the *sparse* baseline. PANMGrad, PANMLoss, and PENS methodologies performed better than the *sparse* baseline across both limited and extensive training set scenarios in the four-cluster experimental settings. Conversely, AUCCR, DAC, DiPLe, and L2C demonstrated superiority over the baseline solely in the scenario with a larger training set. Notably, D-Cliques consistently exhibited markedly inferior performance in both scenarios. Overall, PANMGrad, PANMLoss, and PENS emerged as consistently effective methodologies across all examined scenarios in this experiment.

Table 3: Results on the CIFAR-10 dataset, with label swaps  $\{\text{None}, [0, 2]\}$  applied in the two-cluster environment and label swaps  $\{\text{None}, [0, 1], [2, 3], [4, 5]\}$  applied in the four-cluster environment. Highlighted results indicate performance surpassing the *Sparse* baseline.

| Method    | <b>{None, [0, 2]}</b>    |                          | <b>{None, [0, 1], [2, 3], [4, 5]}</b> |                          |
|-----------|--------------------------|--------------------------|---------------------------------------|--------------------------|
|           | <b>T<sub>s</sub>=100</b> | <b>T<sub>s</sub>=400</b> | <b>T<sub>s</sub>=100</b>              | <b>T<sub>s</sub>=400</b> |
| Oracle    | 43.98                    | 49.90                    | 40.72                                 | 46.83                    |
| Sparse    | 45.51                    | 50.77                    | 42.10                                 | 48.53                    |
| AUCCCR    | 42.21                    | 49.83                    | 37.58                                 | 45.21                    |
| DAC       | 44.82                    | 49.41                    | 42.02                                 | 46.28                    |
| D-Cliques | 36.59                    | 30.46                    | 32.36                                 | 26.49                    |
| DiPLe     | 44.73                    | <b>54.92</b>             | 40.57                                 | <b>51.86</b>             |
| L2C       | 36.29                    | 43.68                    | 33.20                                 | 43.13                    |
| PANMGrad  | 45.34                    | <b>52.10</b>             | <b>42.40</b>                          | <b>49.77</b>             |
| PANMLoss  | 45.47                    | 49.92                    | <b>42.17</b>                          | 46.32                    |
| PENS      | <b>45.52</b>             | <b>52.17</b>             | <b>42.35</b>                          | 48.52                    |

**Label swap.** The primary aim of the label swap experiment is to assess the performance of analyzed methods within a scenario in which agents possess identical features (images) that are associated with distinctly different labels [2]. Similarly to the previous experiment, agents’ data within each cluster was modified by performing label swaps. Transformation  $\{\text{None}, [0, 2]\}$  was used in the two-cluster setting in which the agents belonging to the first cluster were allocated the unaltered data, while the second cluster’s agents were provided with data where class index 0 was switched with class index 2, and vice versa. The described approach was also followed in the four-cluster setting using the following label swaps  $\{\text{None}, [0, 1], [2, 3], [4, 5]\}$ . As Table 3 shows, agents achieved higher accuracy when using the *sparse* communication topology as compared to the *oracle* topology. Contrary to the image rotation experiments, within the context of label swap experiments, randomly forming connections between peers enabled agents to achieve better results. DiPLe demonstrated consistent superior performance in scenarios with a large training dataset. Conversely, PANMGrad exhibited enhanced performance compared to the baseline in all scenarios except for the two-cluster setting with limited training data. PENS performed well overall but exhibited slightly lower performance, specifically in the four-cluster scenario with a large training dataset. PANMLoss showcased marginal performance improvements over the baseline solely in the scenario with a limited training dataset in the four-cluster setting. In contrast, all other methods, including AUCCCR, DAC, D-Cliques, and L2C, consistently exhibited lower performance across all scenarios.

Table 4: Results on the CIFAR-10 dataset with different non-IID data partitioning methods applied in the two-cluster and five-cluster environments. Highlighted results indicate performance surpassing the *Sparse* baseline. Methods that outperformed the *Sparse* baseline in all scenarios are emphasized.

| Method    | {Vehicles, Animals} |              | Pathological non-IID |              | Practical non-IID |              |
|-----------|---------------------|--------------|----------------------|--------------|-------------------|--------------|
|           | $T_s=100$           | $T_s=400$    | $T_s=100$            | $T_s=400$    | $T_s=100$         | $T_s=400$    |
| Oracle    | 55.18               | 61.22        | 83.73                | 84.83        | 40.83             | 67.70        |
| Sparse    | 49.70               | 55.87        | 76.68                | 57.68        | 48.21             | 67.50        |
| AUCCCR    | <b>55.07</b>        | <b>61.45</b> | <b>83.50</b>         | <b>83.82</b> | 42.38             | 67.48        |
| DAC       | <b>52.11</b>        | <b>56.84</b> | <b>79.10</b>         | <b>75.54</b> | <b>47.74</b>      | 64.80        |
| D-Cliques | <b>49.79</b>        | 53.76        | 34.29                | 29.48        | 42.31             | 35.10        |
| DiPLe     | <b>52.74</b>        | <b>61.18</b> | <b>77.14</b>         | 54.13        | <b>48.22</b>      | <b>71.09</b> |
| L2C       | 46.59               | <b>55.36</b> | <b>77.70</b>         | <b>76.28</b> | 38.70             | <b>70.65</b> |
| PANMGrad  | <b>55.85</b>        | <b>62.43</b> | <b>79.70</b>         | <b>78.49</b> | <b>48.98</b>      | <b>69.37</b> |
| PANMLoss  | <b>55.27</b>        | <b>61.89</b> | <b>83.68</b>         | <b>86.84</b> | <b>48.93</b>      | <b>68.95</b> |
| PENS      | <b>52.25</b>        | <b>58.84</b> | <b>77.81</b>         | <b>72.33</b> | <b>48.72</b>      | <b>68.94</b> |

**Partition data.** Experiments involving partitioned data assess the performance of studied methods under label distribution skew [2]. The evaluated methods were assessed across three distinct scenarios: 1) a two-cluster setting with one exclusively containing animal samples and the other exclusively containing vehicle classes [37]; 2) *pathological non-IID* data setting [58] where the dataset is partitioned such that each agent is allocated with samples of only two out of ten possible classes; 3) *practical non-IID* data setting [32] where the data is distributed among agents in a way that ensures each agent possesses data from all classes, albeit with varying class distributions, resulting in some classes having higher probabilities than others. Five distinct clusters are formed in both *pathological* and *practical non-IID* scenarios. Table 4 presents the results of the experiments. In all scenarios except *practical non-IID* with limited training data, agents demonstrated superior accuracy when employing the *oracle* communication topology as compared to the *sparse* topology, suggesting that random communication between agents is not beneficial under disjoint label distribution skew. The AUCCCR method exhibited superior performance compared to the baseline in the two-cluster and *pathological non-IID* settings. However, it yielded unsatisfactory results in the *pathological non-IID* scenario. DAC demonstrated enhanced performance over the baseline across all scenarios except for the *pathological non-IID* environment in the scenario with a large training dataset. Similarly, DiPLe outperformed the baseline in all scenarios, except for the *pathological non-IID* environment in the scenario with a large training dataset. The L2C method displayed lower performance in the limited training set scenarios within the two-cluster and *pathological non-IID* environments. The D-Cliques method failed to surpass the baseline in most scenarios, barring the two-cluster setting with limited data, potentially due to the restricted size of the clique groups ( $M = 10$ ). Notably, PANMGrad, PANMLoss, and PENS consistently outperformed the baseline in all examined scenarios, exhibiting effectiveness across the experiment’s diverse settings.

**Communication analysis.** Communication analysis covers all experiments conducted in the synthetic non-IID environment. The results were categorized by distinct training set scenarios ( $T_S$ ) due to the closely aligned outcomes observed across various cluster counts. The Gini coefficient values for the topologies employed in *oracle*, *sparse*, and AUCCCR experiments yielded a Gini coefficient 0, indicating perfect communication balance. Notably, methods D-Cliques, DiPLe, PANMGrad, PANMLoss, and PENS exhibit Gini coefficients below 0.1 in both limited ( $T_S = 100$ ) and large ( $T_S = 400$ ) training set scenarios. This signifies an equitable distribu-

tion of communication load among agents, providing reassurance about the fairness of these methods. High Gini coefficients were obtained for DAC ( $gini > 0.2$ ) and L2C ( $gini > 0.4$ ) for both training set scenarios, implying centralized communication tendencies as agents predominantly source messages from a restricted peer set. All methods were adjusted to maintain communication with approximately the same number of peers, as previously detailed in Section 5. Compared to the baseline *sparse* topology, AUCCCR, DAC, PANMGrad, and PANMLoss methods exhibit similar message communication frequencies. In contrast, the PENS method shows a 15% increase, while the L2C method demonstrates a 48% increase in message communication. D-Cliques and DiPLe methods display significant inefficiencies in communication load, with D-Cliques experiencing a 50-fold increase and DiPLe demonstrating a 6-fold increase in message communication.

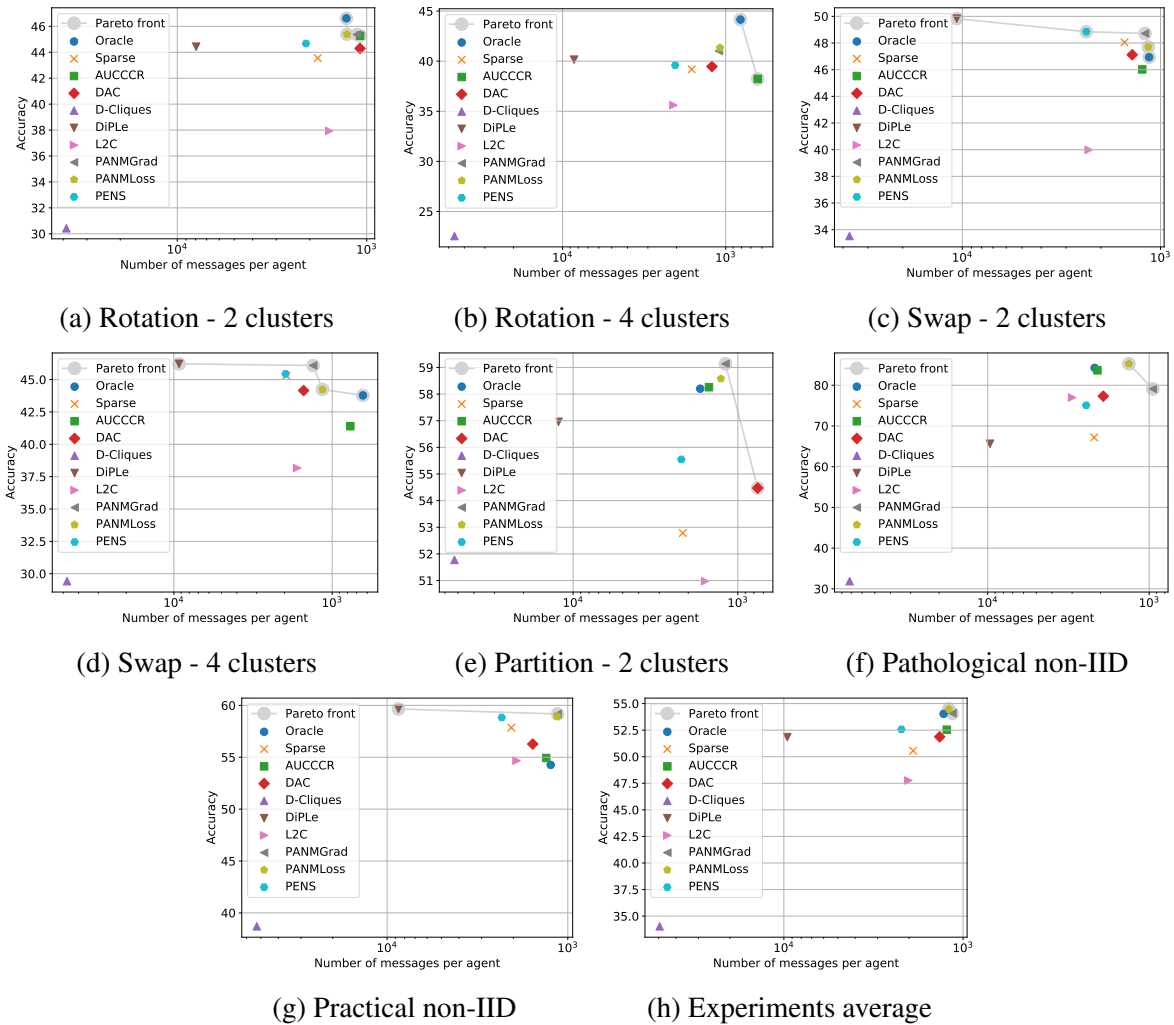


Figure 5: Methods on the Pareto front for all synthetic non-IID experiments.



**Pareto analysis.** Optimal solutions in each experiment were identified by applying the Pareto method. Scenarios within each experiment were grouped based on the number of clusters by taking the mean of the scenarios with limited and large set of training data. The optimal solutions residing on the Pareto front were discerned through an analysis of the top UA and the mean number of agent messages needed to achieve these outcomes. Figure 5 illustrates the Pareto optimal solutions. In the context of rotation experiments, optimal solutions in the two-cluster scenario included *oracle*, AUCCCR, PANMGrad, and PANMLoss, whereas in the four-cluster scenario, only *oracle* and AUCCCR were identified as optimal. In both two-cluster and four-cluster scenarios involving label swap experiments, *oracle*, PANMGrad, PANMLoss, and DiPLe emerged as optimal solutions. Additionally, PENS demonstrated noteworthy performance by residing on the Pareto front in the two-cluster experiment. PANMGrad emerged as the optimal solution in the partition experiments across all scenarios. Furthermore, additional solutions residing on the Pareto front include DAC in the two-cluster scenario, PANMLoss in the *pathological non-IID* scenario, and DiPLe in the *practical non-IID* scenario. Throughout all experiments, PANMGrad and PANMLoss consistently emerged as the optimal solutions on average, providing a clear conclusion to our research. The PANM method is the most effective in synthetic non-IID environments involving an image classification task.

## 6.2. Results in a realistic non-IID environment

As previously mentioned, agents were selected based on the subreddit topic to which they exclusively contributed, resulting in four distinct agent clusters whose members share similar interests. Experiments were performed for each pair of clusters alongside an experiment involving all four clusters combined, resulting in seven distinct scenarios. Like the CIFAR-10 assessments, the learning methodologies were assessed across limited and extensive quantities of local agent data. In the Reddit dataset, variations in sample sizes per agent led to limited local data experiments encompassing user data with sample numbers ( $T_s$ ) between 300 and 700. In contrast, large local data experiments involved users with sample numbers ( $T_s$ ) ranging from 1000 to 5000. Experiments with two clusters comprised 100 agents, whereas experiments with four clusters comprised 200 agents (50 per cluster in all scenarios).

The AUCCCR method was excluded from the subsequent experiments due to its demand

Table 5: Results on the Reddit dataset with two and four cluster experiments under small and large local data quantities scenarios. Highlighted results indicate performance surpassing the *Sparse* baseline. Methods that outperformed the *Sparse* baseline in all scenarios are emphasized.

| Method          | 2-cluster average |                     | 4-cluster         |                     |
|-----------------|-------------------|---------------------|-------------------|---------------------|
|                 | $300 < T_s < 700$ | $1000 < T_s < 5000$ | $300 < T_s < 700$ | $1000 < T_s < 5000$ |
| Oracle          | 6.48              | 8.11                | 6.40              | 8.10                |
| Sparse          | 6.72              | 8.35                | 6.76              | 8.65                |
| <b>DAC</b>      | <b>7.20</b>       | <b>8.94</b>         | <b>7.33</b>       | <b>9.20</b>         |
| D-Cliques       | 5.60              | <b>8.92</b>         | <b>8.58</b>       | <b>10.70</b>        |
| <b>DiPLe</b>    | <b>7.37</b>       | <b>9.98</b>         | <b>7.65</b>       | <b>10.50</b>        |
| L2C             | 6.12              | 7.64                | <b>6.85</b>       | <b>10.50</b>        |
| <b>PANMGrad</b> | <b>7.10</b>       | <b>8.56</b>         | <b>7.22</b>       | <b>8.82</b>         |
| PANMLoss        | 5.86              | 7.13                | 6.02              | 7.39                |
| <b>PENS</b>     | <b>7.14</b>       | <b>8.52</b>         | <b>7.31</b>       | <b>8.82</b>         |

for vector sizes proportional to the square of the number of classes. For the NWP task involving 10,000 classes, this resulted in excessively large vectors that exceeded the storage capacity of the available 64GB of RAM.

**Results analysis.** Table 5 displays the results obtained from all experiments using the Reddit dataset. Experiments with two clusters were consolidated as they produced comparable accuracies. A comparison between the baseline *sparse* and the *oracle* topology reveals that higher accuracies are attained when connections are formed randomly, consistent with the findings from synthetic experiments involving label swaps. DAC, DiPLe, PANMGrad, and PENS consistently outperformed the *sparse* baseline in all scenarios. Furthermore, D-Cliques underperformed only in the two-cluster scenario with limited data. L2C exhibited superior performance over the baseline solely in the four-cluster scenarios, while PANMLoss consistently underperformed across all scenarios.

**Communication analysis.** The communication analysis encompasses all experiments conducted in the realistic non-IID environment, grouped by distinct training set scenarios ( $T_s$ ) due to observed closely aligned outcomes across various cluster counts. Fixed topologies utilized in *oracle* and *sparse* experiments yielded a Gini coefficient 0, indicating perfect communication balance. Methods DAC, D-Cliques, and DiPLe demonstrated Gini coefficients below 0.1 in both limited ( $300 < T_s < 700$ ) and large ( $1000 < T_s < 5000$ ) training set scenarios, suggesting equitable communication load distribution among agents. On the other hand, PENS ( $gini > 0.2$ ) and L2C ( $gini > 0.3$ ) consistently exhibited high Gini coefficients across both

training set scenarios, indicating a tendency towards centralized communication. This suggests potential challenges in maintaining a balanced communication load in these methods. PANMLoss exhibited a Gini coefficient exceeding 0.1 in both training set scenarios. At the same time, PANMGrad showed a Gini coefficient below 0.1 in the limited training set scenario but demonstrated higher centralization properties ( $gini > 0.2$ ) in the large training set scenario.

Compared to the baseline *sparse* topology, both PANMGrad and PANMLoss methods reduced message communication frequencies, whereas DAC and PENS methods displayed similar communication frequencies. D-Cliques exhibited an average 80-fold increase over the *sparse* baseline, while L2C experienced an average 12-fold increase. Additionally, DiPLE demonstrated an average 6-fold increase in message communication.

**Pareto analysis.** Optimal solutions in the realistic non-IID environment were identified through the application of the Pareto method. Scenarios were analyzed separately for different cluster numbers and distinct training set scenarios. The solutions residing on the Pareto front were discerned through an analysis of the top UA and the mean number of agent messages needed to achieve these outcomes. Figure 6 illustrates the analysis of Pareto front optimal solutions. In all experiments, DAC and DiPLE consistently emerged as solutions located on the Pareto

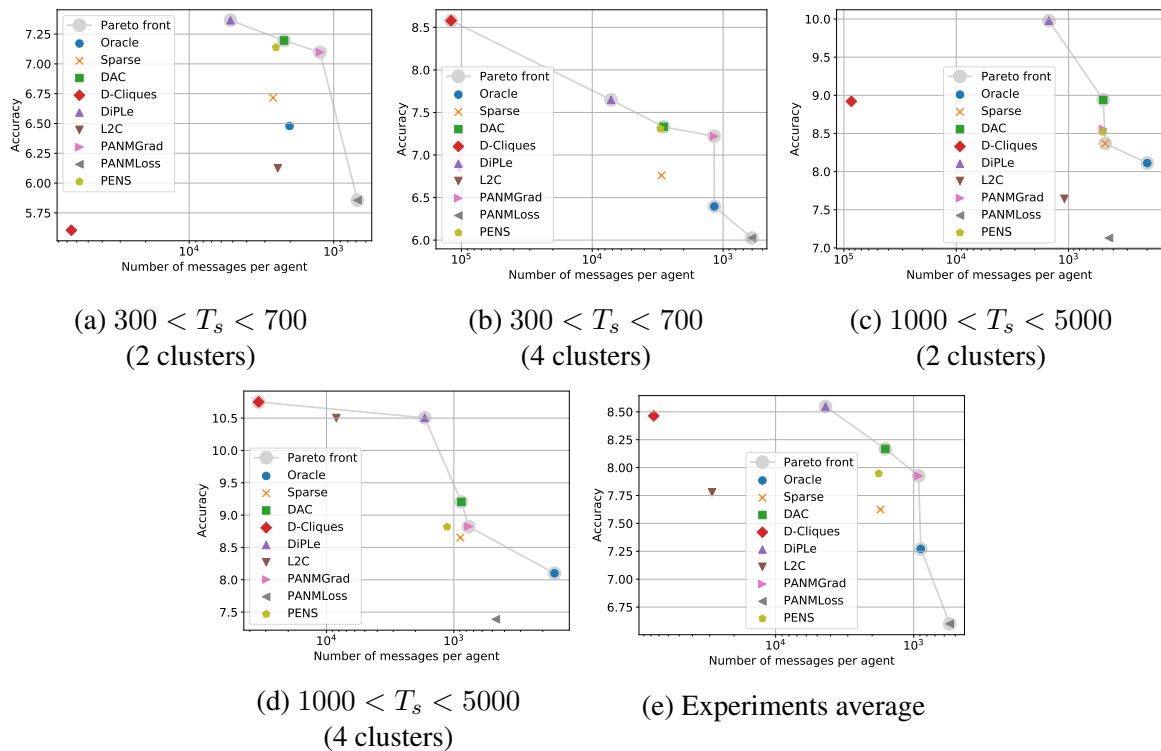


Figure 6: Methods on the Pareto front for all realistic non-IID experiments.

front. In the two-cluster scenario with limited data, additional solutions included L2C, PANMGrad, and PANMLoss. In the two-cluster scenario with a large dataset, additional solutions were *oracle* and *sparse*. For the four-cluster scenario with limited data, additional solutions comprised *oracle*, D-Cliques, PANMGrad, and PANMLoss. In the four-cluster scenario with a large dataset, additional solutions were *oracle*, D-Cliques, and PANMGrad. Overall, *oracle*, DAC, DiPLe, PANMGrad, and PANMLoss were identified as residing on the Pareto front for the realistic non-IID environment involving the Reddit dataset.

### 6.3. Results summary

The following analysis encompasses synthetic and realistic non-IID experiment results to answer our research questions.

Research question **RQ2** investigates the communication efficiency of the analyzed methods. All methods were adjusted to maintain communication with approximately the same number of peers in terms of the number of exchanged messages. Compared to the baseline *sparse* topology, AUCCCR, DAC, PANMGrad, and PANMLoss methods exhibit similar message communication frequencies, while the PENS method shows a 15% increase. The L2C method demonstrates a 48% increase in message communication in synthetic non-IID environments and a 12-fold increase in realistic non-IID environments. D-Cliques and DiPLe methods display significant inefficiencies in communication load, with D-Cliques experiencing a minimum 50-fold increase and DiPLe demonstrating a 6-fold increase in message communication.

Regarding **RQ3**, which addresses the centralization tendencies of the analyzed methods, the Gini coefficient values for the topologies utilized in the *oracle*, *sparse*, and AUCCCR experiments unsurprisingly yielded a coefficient of 0, indicating perfect communication balance. Methods D-Cliques and DiPLe demonstrated Gini coefficients below 0.1 in synthetic and realistic non-IID environments, indicating an equitable distribution of communication load among agents. Furthermore, PANMGrad, PANMLoss, and PENS exhibited Gini coefficients below 0.1 in the synthetic non-IID environment but showed increased coefficients (Gini  $\geq$  0.3) in the realistic non-IID environment, suggesting higher centralization tendencies. While DAC demonstrated a Gini coefficient below 0.1 in the realistic non-IID environment, it exhibited a higher coefficient for synthetic non-IID environments. Conversely, the L2C method showed high cen-

tralization tendencies in synthetic and realistic non-IID environments.

We have the following insights to answer **RQ4**, which inquires about the best balance between communication efficiency and overall improved learning outcomes for agents. Overall, in experiments regarding the synthetic non-IID environment, PANMGrad and PANMLoss methods were identified as the optimal solutions, suggesting their effectiveness in image classification. The low communication requirements per agent and low centralization properties deemed the PANM method a superior approach in synthetic non-IID environments compared to the *sparse* baseline, even outperforming the *oracle* reference. In the experiments involving the next-word prediction task on a realistic non-IID dataset, the *oracle*, DAC, DiPLe, PANMGrad, and PANMLoss methods were identified as residing on the Pareto front. DiPLe was found to be very inefficient regarding the communication load for agents, which significantly diminishes its applicability in realistic scenarios. DAC, PANMGrad, and PANMLoss demonstrated comparable communication loads per agent as the *sparse* baseline, with DAC displaying low centralization tendencies, while PANMGrad and PANMLoss exhibited higher centralization properties. Given the communication load on agents and the observed centralization tendencies, the PANM method, in both its gradient-based (PANMGrad) and loss-based (PANMLoss) variants, emerges as the optimal choice in the conducted experiments.

## 7. Conclusion

This study investigated communication-efficient peer-to-peer learning methodologies within the context of non-IID data distributions, focusing on the autonomous creation of connections between agents. Study methods were analyzed and compared through a series of experiments varying in cluster counts and dataset sizes. Findings obtained through the experiments identified several methods as optimal while also identifying shortcomings of other methods. Methods PANMGrad and PANMLoss emerged as optimal solutions under synthetic and realistic non-IID environments, while DAC and DiPLe emerged as the optimal solutions in the realistic non-IID environment.

Communication analysis showed that the number of messages sent varied significantly be-

tween the different methods. For example, D-Clique, DiPLe, and L2C always required the most messages to reach convergence. Additionally, the calculation of the Gini coefficient revealed distinctive communication patterns. Specifically, L2C exhibited high centralization tendencies in synthetic and realistic non-IID environments. DAC displayed heightened centralization tendencies solely in the synthetic environment, while PENS showed similar tendencies exclusively in the realistic environment.

Given the communication load on agents and the observed centralization tendencies, the PANM method, in its gradient-based (PANMGrad) and loss-based (PANMLoss) variants, emerges as the optimal choice in conducted experiments.

Future investigations in this domain aim to refine and extend the methodologies explored in this study. One potential direction for further exploration involves enhancing existing techniques, such as the PANM method, to mitigate any shortcomings identified during experimentation. Researchers could focus on developing modifications that maintain or enhance communication efficiency and avoid exacerbating centralization tendencies. All examined methods necessitate access to the complete network, encompassing all agents, particularly in the initial phases. However, this assumption could be more practical for large, volatile, decentralized systems with a high churn ratio. Hence, future investigations should develop communication-efficient methods that enable agents to identify similar peers without communicating with the entire network.

In terms of communication efficiency, methodologies reliant on measuring agent similarity based on model loss demonstrate superior efficiency compared to the PANMGrad method. The latter exchanges current gradients and accumulated weight deltas, substantially increasing communication. An additional avenue for exploration involves investigating whether the loss similarity metric can be substituted with the similarity between model predictions. Measures such as cosine similarity or Euclidean distance could be applied, as in the AUCCCR method.

Another promising direction for future research involves investigating peer-to-peer learning methodologies in more challenging non-IID scenarios, such as those involving dynamic or adversarial data distributions. The challenge posed by adversarial agents persists as an active area of research, extending to Federated Learning, where diverse methods have been proposed to address potential attacks [59–61]. Adversarial attackers' objective is to inject triggers that cause targeted misclassifications without compromising model accuracy or disrupting convergence, which is quite challenging. In classification applications, adversary attacks can involve adding

extra patterns to benign images for vision tasks or appending trigger strings for NLP tasks, deliberately causing the classifier to misclassify. These attacks could have severe consequences in scenarios where accurate classification is paramount. Current approaches are unlikely to detect such attacks with loss similarity metrics in place. Understanding how different methodologies perform under these dynamic and challenging conditions could provide valuable insights into their robustness and adaptability limits.

## References

- [1] Rong Yu and Peichun Li. “Toward Resource-Efficient Federated Learning in Mobile Edge Computing”. In: *IEEE Network* 35.1 (2021), pp. 148–155. DOI: 10.1109/MNET.011.2000295.
- [2] Xiaodong Ma et al. “A state-of-the-art survey on solving non-IID data in Federated Learning”. In: *Future Generation Computer Systems* 135 (2022), pp. 244–258. ISSN: 0167-739X. DOI: <https://doi.org/10.1016/j.future.2022.05.003>. URL: <https://www.sciencedirect.com/science/article/pii/S0167739X22001686>.
- [3] Yuwei Sun and Hideya Ochiai. “Homogeneous Learning: Self-Attention Decentralized Deep Learning”. In: *IEEE Access* 10 (2022), pp. 7695–7703. DOI: 10.1109/ACCESS.2022.3142899.
- [4] Robert Šajina, Nikola Tanković, and Ivo Ipšić. “Peer-to-peer deep learning with non-IID data”. In: *Expert Systems with Applications* 214 (2023), p. 119159. ISSN: 0957-4174. DOI: <https://doi.org/10.1016/j.eswa.2022.119159>. URL: <https://www.sciencedirect.com/science/article/pii/S0957417422021777>.
- [5] Avishek Ghosh et al. “An Efficient Framework for Clustered Federated Learning”. In: *Proceedings of the 34th International Conference on Neural Information Processing Systems*. Ed. by Larochelle H. et al. NIPS’20. Vancouver, BC, Canada: Curran Associates Inc., 2020. ISBN: 9781713829546.

- [6] Guodong Long et al. “Multi-center federated learning: clients clustering for better personalization”. In: *World Wide Web* 26 (June 2022), pp. 1–20. DOI: 10.1007/s11280-022-01046-x.
- [7] Felix Sattler, Klaus-Robert Müller, and Wojciech Samek. “Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints”. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.8 (2021), pp. 3710–3722. DOI: 10.1109/TNNLS.2020.3015958.
- [8] Moming Duan et al. “Flexible Clustered Federated Learning for Client-Level Data Distribution Shift”. In: *IEEE Transactions on Parallel and Distributed Systems* PP (Dec. 2021), pp. 1–1. DOI: 10.1109/TPDS.2021.3134263.
- [9] Weiwei She et al. “pFedLN: Personalized Federated Learning Framework With Layer-Wised and Neighbor-Based Aggregation for QoS Prediction”. In: *IEEE Access* 12 (2024), pp. 10135–10145. DOI: 10.1109/ACCESS.2024.3353617.
- [10] Xiang Li et al. “On the Convergence of FedAvg on Non-IID Data”. In: *International Conference on Learning Representations*. 2020.
- [11] Kevin Hsieh et al. “The non-IID data quagmire of decentralized machine learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. ICML’20. JMLR.org, 2020.
- [12] Abraham Woubie, Enoch Solomon, and Joseph Attieh. “Maintaining Privacy in Face Recognition Using Federated Learning Method”. In: *IEEE Access* 12 (2024), pp. 39603–39613. DOI: 10.1109/ACCESS.2024.3373691.
- [13] Aurélien Bellet et al. “Personalized and Private Peer-to-Peer Machine Learning”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Ed. by Amos Storkey and Fernando Perez-Cruz. Vol. 84. Proceedings of Machine Learning Research. PMLR, Apr. 2018, pp. 473–481. URL: <https://proceedings.mlr.press/v84/bellet18a.html>.
- [14] Xiangru Lian et al. “Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Asso-



- ciates, Inc., 2017. URL: <https://proceedings.neurips.cc/paper/2017/file/f75526659f31040afeb61cb7133e4e6d-Paper.pdf>.
- [15] Hanlin Tang et al. “D<sup>2</sup>: Decentralized Training over Decentralized Data”. In: *Proceedings of the 35th International Conference on Machine Learning* 80.1 (2018), pp. 4848–4856. URL: <http://proceedings.mlr.press/v80/tang18a.html>.
- [16] Michael Blot et al. “Distributed optimization for deep learning with gossip exchange”. In: *Neurocomputing* 330 (2019), pp. 287–296. ISSN: 0925-2312. DOI: <https://doi.org/10.1016/j.neucom.2018.11.002>. URL: <https://www.sciencedirect.com/science/article/pii/S0925231218313195>.
- [17] Hideaki Oguni and Kazuyuki Shudo. “Communication Scheduling for Gossip SGD in a Wide Area Network”. In: *IEEE Access* 9 (2021), pp. 77873–77881. DOI: 10.1109/ACCESS.2021.3083639.
- [18] Chaoyang He et al. “Central Server Free Federated Learning over Single-sided Trust Social Networks”. In: *ArXiv abs/1910.04956* (2019). URL: <http://arxiv.org/abs/1910.04956>.
- [19] Robert Šajina, Nikola Tanković, and Darko Etinger. “Decentralized trustless gossip training of deep neural networks”. In: *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*. 2020, pp. 1080–1084. DOI: 10.23919/MIPRO48935.2020.9245248.
- [20] Karim Boubouh et al. “Robust P2P Personalized Learning”. In: *2020 International Symposium on Reliable Distributed Systems (SRDS)*. 2020, pp. 299–308. DOI: 10.1109/SRDS51746.2020.00037.
- [21] Yue Gao et al. “Topology Measurement and Analysis on Ethereum P2P Network”. In: *2019 IEEE Symposium on Computers and Communications (ISCC)*. 2019, pp. 1–7. DOI: 10.1109/ISCC47284.2019.8969695.
- [22] Taotao Wang et al. “Ethna: Analyzing the Underlying Peer-to-Peer Network of Ethereum Blockchain”. In: *IEEE Transactions on Network Science and Engineering* 8.3 (2021), pp. 2131–2146. DOI: 10.1109/TNSE.2021.3078181.

- [23] Andrew Howell, Takfarinas Saber, and Malika Bendeche. “Measuring node decentralisation in blockchain peer to peer networks”. In: *Blockchain: Research and Applications* 4.1 (2023), p. 100109. ISSN: 2096-7209. DOI: <https://doi.org/10.1016/j.bcra.2022.100109>. URL: <https://www.sciencedirect.com/science/article/pii/S2096720922000501>.
- [24] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. “Fully Decentralized Joint Learning of Personalized Models and Collaboration Graphs”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 864–874. URL: <https://proceedings.mlr.press/v108/zantedeschi20a.html>.
- [25] Peter H. Jin et al. “How to scale distributed deep learning?” In: *CoRR* abs/1611.04581 (2016). arXiv: 1611.04581. URL: <http://arxiv.org/abs/1611.04581>.
- [26] Peter Kairouz et al. “Advances and Open Problems in Federated Learning”. In: *Foundations and Trends® in Machine Learning* 14.1–2 (2021), pp. 1–210. ISSN: 1935-8237. DOI: 10.1561/22000000083. URL: <http://dx.doi.org/10.1561/22000000083>.
- [27] Mohammed Aledhari et al. “Federated Learning: A Survey on Enabling Technologies, Protocols, and Applications”. In: *IEEE Access* 8 (2020), pp. 140699–140725. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3013541.
- [28] Carlos Poncinelli Filho et al. “A Systematic Literature Review on Distributed Machine Learning in Edge Computing”. In: *Sensors* 22.7 (2022). ISSN: 1424-8220. DOI: 10.3390/s22072665. URL: <https://www.mdpi.com/1424-8220/22/7/2665>.
- [29] Leon Witt et al. “Decentral and Incentivized Federated Learning Frameworks: A Systematic Literature Review”. In: *IEEE Internet of Things Journal* 10.4 (2023), pp. 3642–3663. DOI: 10.1109/JIOT.2022.3231363. URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85146240820&doi=10.1109%2fJIOT.2022.3231363&partnerID=40&md5=71a522729f21395e61f3565678e4ab2e>

- [30] Mohammad Shoab and Saad Al Jubayrin. “Intelligent neighbor selection for efficient query routing in unstructured P2P networks using Q-learning”. In: *Applied Intelligence* 52.6 (Apr. 2022), pp. 6306–6315. ISSN: 0924-669X. DOI: 10.1007/s10489-021-02793-6. URL: <https://doi.org/10.1007/s10489-021-02793-6>.
- [31] Sara Taylor et al. “Personalized Multitask Learning for Predicting Tomorrow’s Mood, Stress, and Health”. In: *IEEE Transactions on Affective Computing* 11.2 (2020), pp. 200–213. DOI: 10.1109/TAFFC.2017.2784832.
- [32] Yutao Huang et al. “Personalized Federated Learning: An Attentive Collaboration Approach”. In: *CoRR* abs/2007.03797 (2020). URL: <https://arxiv.org/abs/2007.03797>.
- [33] Moming Duan et al. “FedGroup: Ternary Cosine Similarity-based Clustered Federated Learning Framework toward High Accuracy in Heterogeneous Data”. In: *CoRR* abs/2010.06870 (2020). arXiv: 2010.06870. URL: <https://arxiv.org/abs/2010.06870>.
- [34] Minh N. H. Nguyen et al. “Self-organizing Democratized Learning: Towards Large-scale Distributed Learning Systems”. In: *IEEE transactions on neural networks and learning systems* PP (2020).
- [35] Christopher Briggs, Zhong Fan, and Péter András. “Federated learning with hierarchical clustering of local updates to improve training on non-IID data”. In: *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), pp. 1–9.
- [36] Noa Onoszko et al. “Decentralized federated learning of deep neural networks on non-iid data”. In: *International Workshop on Federated Learning for User Privacy and Data Confidentiality in Conjunction with ICML*. Vol. abs/2107.08517. 2021.
- [37] Edvin Listo Zec et al. “Decentralized Adaptive Clustering of Deep Nets is Beneficial for Client Collaboration”. In: *Trustworthy Federated Learning*. Ed. by Randy Goebel et al. Springer International Publishing, 2023, pp. 59–71. ISBN: 978-3-031-28996-5.
- [38] Zexi Li et al. “Towards Effective Clustered Federated Learning: A Peer-to-peer Framework with Adaptive Neighbor Matching”. In: *IEEE Transactions on Big Data* (2022), pp. 1–16. DOI: 10.1109/TBDATA.2022.3222971.

- [39] Yi Sui et al. “Find Your Friends: Personalized Federated Learning with the Right Collaborators”. In: *Workshop on Federated Learning: Recent Advances and New Challenges, in Conjunction with NeurIPS 2022 (FL-NeurIPS’22)*. 2022. DOI: 10.48550/ARXIV.2210.06597. URL: <https://arxiv.org/abs/2210.06597>.
- [40] Aurélien Bellet, Anne-Marie Kermarrec, and Erick Lavoie. “D-Cliques: Compensating for Data Heterogeneity with Topology in Decentralized Federated Learning”. In: *2022 41st International Symposium on Reliable Distributed Systems (SRDS) (2021)*, pp. 1–11.
- [41] Mohammad Rostami et al. “Multi-Agent Distributed Lifelong Learning for Collective Knowledge Acquisition”. In: *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*. Ed. by Andre Elisabeth et al. AAMAS ’18. Stockholm, Sweden: International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 712–720.
- [42] Xue Zheng, Parinaz Naghizadeh, and Aylin Yener. “DiPLE: Learning Directed Collaboration Graphs for Peer-to-Peer Personalized Learning”. In: *2022 IEEE Information Theory Workshop (ITW)*. Ed. by Institute of Electrical and Electronics Engineers. 2022, pp. 446–451. DOI: 10.1109/ITW54588.2022.9965838.
- [43] Shuangtong Li et al. “Learning to Collaborate in Decentralized Learning of Personalized Models”. In: *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 9756–9765. DOI: 10.1109/CVPR52688.2022.00954.
- [44] Amaury Bouchra Pilet, Davide Frey, and François Taïani. “AUCCCR: Agent Utility Centered Clustering for Cooperation Recommendation”. In: *Networked Systems*. Ed. by Karima Echihabi and Roland Meyer. Cham: Springer International Publishing, 2021, pp. 111–125.
- [45] Li Deng. “The mnist database of handwritten digit images for machine learning research”. In: *IEEE Signal Processing Magazine* 29.6 (2012), pp. 141–142.
- [46] Alex Krizhevsky and Geoffrey Hinton. *Learning multiple layers of features from tiny images*. Tech. rep. 0. Toronto, Ontario: University of Toronto, 2009. URL: <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
- [47] Sebastian Caldas et al. “LEAF: A Benchmark for Federated Settings”. In: *ArXiv abs/1812.01097* (2018).

- [48] Andrew Hard et al. “Federated Learning for Mobile Keyboard Prediction”. In: *CoRR* abs/1811.03604 (2018). URL: <https://arxiv.org/abs/1811.03604>.
- [49] Sai Praneeth Karimireddy et al. “SCAFFOLD: Stochastic Controlled Averaging for Federated Learning”. In: *Proceedings of the 37th International Conference on Machine Learning*. Ed. by Hal Daumé III and Aarti Singh. Vol. 119. Proceedings of Machine Learning Research. PMLR, July 2020, pp. 5132–5143. URL: <https://proceedings.mlr.press/v119/karimireddy20a.html>.
- [50] Kangkang Wang et al. “Federated Evaluation of On-device Personalization”. In: *ArXiv* abs/1910.10252 (2019). URL: <http://arxiv.org/abs/1910.10252>.
- [51] Sashank J. Reddi et al. “Adaptive Federated Optimization”. In: *International Conference on Learning Representations*. 2021.
- [52] Joel Stremmel and Arjun Singh. “Pretraining Federated Text Models for Next Word Prediction”. In: *Advances in Information and Communication*. Ed. by Kohei Arai. Springer International Publishing, 2021, pp. 477–488. ISBN: 978-3-030-73103-8.
- [53] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. Ed. by Yoshua Bengio and Yann LeCun. 2015. URL: <http://arxiv.org/abs/1412.6980>.
- [54] J. Mills, J. Hu, and G. Min. “Multi-Task Federated Learning for Personalised Deep Neural Networks in Edge Computing”. In: *IEEE Transactions on Parallel and Distributed Systems* 33.03 (Mar. 2022), pp. 630–641. ISSN: 1558-2183. DOI: 10.1109/TPDS.2021.3098467.
- [55] Theoni Pitoura and Peter Triantafillou. “Load Distribution Fairness in P2P Data Management Systems”. In: *2007 IEEE 23rd International Conference on Data Engineering*. 2007, pp. 396–405. DOI: 10.1109/ICDE.2007.367885.
- [56] Fabio Caccioli, Giacomo Livan, and Tomaso Aste. “Scalability and Egalitarianism in Peer-to-Peer Networks”. In: *Banking Beyond Banks and Money: A Guide to Banking Services in the Twenty-First Century*. Cham: Springer International Publishing, 2016, pp. 197–212. ISBN: 978-3-319-42448-4. DOI: 10.1007/978-3-319-42448-4\_11. URL: [https://doi.org/10.1007/978-3-319-42448-4\\_11](https://doi.org/10.1007/978-3-319-42448-4_11).

- [57] Bartosz Kusmierz and Roman Overko. “How centralized is decentralized? Comparison of wealth distribution in coins and tokens”. In: *2022 IEEE International Conference on Omni-layer Intelligent Systems (COINS)*. Los Alamitos, CA, USA: IEEE Computer Society, Aug. 2022, pp. 1–6. DOI: 10.1109/COINS54846.2022.9854972. URL: <https://doi.ieeecomputersociety.org/10.1109/COINS54846.2022.9854972>.
- [58] Brendan McMahan et al. “Communication-Efficient Learning of Deep Networks from Decentralized Data”. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. Ed. by Aarti Singh and Jerry Zhu. Vol. 54. Proceedings of Machine Learning Research. PMLR, Apr. 2017, pp. 1273–1282. URL: <https://proceedings.mlr.press/v54/mcmahan17a.html>.
- [59] Eugene Bagdasaryan et al. “How To Backdoor Federated Learning”. In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, Aug. 2020, pp. 2938–2948. URL: <https://proceedings.mlr.press/v108/bagdasaryan20a.html>.
- [60] Avishek Ghosh et al. “Robust Federated Learning in a Heterogeneous Environment”. In: *ArXiv abs/1906.06629 (2019)*. URL: <https://api.semanticscholar.org/CorpusID:189927938>.
- [61] Syed Zawad and Ahsan Ali and Chen, Pin Yu and Ali Anwar and Yi Zhou and Nathalie Baracaldo and Yuan Tian and Feng Yan. “Curse or Redemption? How Data Heterogeneity Affects the Robustness of Federated Learning”. English (US). In: *35th AAAI Conference on Artificial Intelligence, AAAI 2021*. 35th AAAI Conference on Artificial Intelligence, AAAI 2021 (2021), pp. 10807–10814.

## **D. Supplemental Materials**

This appendix contains supplementary materials in the form of a link to a public GitHub repository. The repository includes instructions and reproducible code for all experiments conducted in this doctoral thesis and published articles listed in Appendixes A, B, and C: [https://github.com/fipu-lab/p2p\\_bn](https://github.com/fipu-lab/p2p_bn).