École Polytechnique Fédérale de Lausanne

Do Opposites Attract? Optimizing Decentralized Learning: Enhancing Node Performance through Model Distance in Non-IID Settings

by Hugo Lanfranchi

Master Thesis

Approved by the Examining Committee:

Prof. Anne-Marie Kermarrec
Thesis Advisor

Erwan Le Merrer
External Expert

Sayan Biswas
Martijn de Vos
Rishi Sharma
Thesis Supervisor

Scalable Computing Systems
BC 160 (Bâtiment BC)
Station 14
CH-1015 Lausanne

September 13, 2024

Dedicated to my EPFL journey

Acknowledgments

First and foremost, I would like to express my gratitude to my supervisors, Sayan Biswas and Martijn de Vos, for their unwavering support and commitment to the project. Their guidance during our weekly meetings was invaluable, steering me in the right direction and offering assistance whenever needed. I am also thankful to the SaCS laboratory for providing the server and nodes to be able to run the experiments. I would also like to express my gratitude to the HexHive laboratory for providing this template.

Lausanne, September 13, 2024 Hugo Lanfranchi

Abstract

Decentralized Learning (DL) is a collaborative machine learning approach allowing multiple nodes to train a global model without sharing private data. However, data heterogeneity across nodes in Non-Independent and Identically Distributed (Non-IID) settings presents significant challenges, leading to suboptimal model performance. This work introduces the Distance Based Communication Algorithm (DBCA), which optimizes node communication based on model similarity. By adjusting the probability of communication between nodes with dissimilar data distributions, DBCA improves accuracy in decentralized learning environments.

We explore both clustered and non-clustered data settings to analyze the effectiveness of DBCA. Results indicate that communication between nodes with dissimilar models enhances performance, particularly in moderately heterogeneous environments. Conversely, in highly heterogeneous or homogeneous settings, DBCA's advantages diminish. The algorithm shows good performance, outperforming baselines like Epidemic Learning and Decentralized Adaptive Clustering (DAC). Our findings demonstrate the potential of DBCA to generalize across diverse tasks while offering new insights into optimizing communication strategies in decentralized systems.

Contents

Chapter 1

Introduction

Decentralized Learning (DL) [8] has emerged as a powerful approach in collaborative machine learning (ML), enabling multiple nodes to train a global model without the need to share their private datasets and a central entity. This method leverages a communication topology where nodes exchange their locally trained models with neighboring nodes and other participants in the network without revealing their underlying data. In each training round, nodes update their models based on their private datasets, share these updates with neighbors, and aggregate the received models. This aggregated model then serves as the foundation for subsequent training rounds, with the process continuing until the model reaches convergence. Popular DL algorithms such as Decentralized Parallel Stochastic Gradient Descent (D-PSGD) [7] and Epidemic Learning (EL) [9] have been widely tested and used across various fields.

A significant challenge in DL arises from data heterogeneity among nodes[1]. This heterogeneity can manifest in different forms, such as variations in data quantity, label distributions, or feature distributions across nodes. Such diversity exists in multiple domains like animal detection, healthcare or weather, where institutions may collect data using different devices, methodologies, or from data sources with varying diversity. In this work, we focus on scenarios where nodes exhibit heterogeneous feature distributions.

In cases where nodes share similar data characteristics, a clustered distribution often emerges, grouping nodes into distinct subclusters [12]. This phenomenon is common in real-world applications, for instance depending on the geographical distribution of the nodes and thus the underlying data. However, standard DL approaches like D-PSGD or EL typically optimize the model towards the average data distribution across all nodes. This approach can lead to performance issues, particularly for nodes whose data characteristics are very isolated or differ significantly.

For instance, decentralized learning offers clear advantages in tasks like animal detection from images. Consider two nodes: one representing Australia, having a good performance at recognizing

kangaroos, and another representing Europe, better at identifying deer. When the European node encounters a kangaroo, it may misidentify it due to limited exposure. However, by sharing information, both nodes can enhance their detection capabilities, allowing Europe to correctly identify the kangaroo. This illustrates how decentralized learning improves overall accuracy and knowledge through collaboration.

This example highlights the facts that nodes can aim for different training objectives, models in decentralized learning will then likely be very different from node to node in the network. This personalization approach, adjusts the trained model to account for the unique characteristics of each node's dataset.

In a non-identically independently distributed (non-IID) setting, nodes are distinguished by their diverse data distributions. Each node, therefore, pursues a personalized objective, aiming to enhance its accuracy on local test data that mirrors its specific data distribution. This leads to an important question: *can inter-node communication be structured in a way that collectively improves overall performance?* Specifically, does selective node communication, such as prioritizing certain nodes over others yield better results in a non-IID environment where label distribution shifts are present? We will explore this matter through the scope of altering the communication topology of the distributed system, by introducing a probabilistic model of communication based on similarity criteria between nodes data.

To achieve this, in this work we present the Distance Based Communication Algorithm (DBCA), a probabilistic approach to node communication in the non-IDD setting that relies on model similarity to exchange information. This project successfully implemented the described algorithm and showed that it outperforms the state-of-the-art methods for a given data distribution among the nodes.

In summary, the primary contributions of this project include:

1. Showing that communication between nodes with dissimilar models enhances accuracy.
2. Comparing performance in this setting to a clustered environment, where nodes within each cluster share highly similar data, as often observed in real-world scenarios
3. Implementing the DBCA algorithm which is responsible for the probabilistic method to this research approach

Chapter 2

Preliminaries and related work

In this chapter, we first introduce the problem setting, for the non-cluster and cluster data scenarios, then review related work in the domain of personalized learning and decentralized learning.

2.0.1 Problem Formulation

Decentralized Learning (DL) has emerged as a powerful approach in collaborative machine learning (ML), enabling multiple nodes to train a global model without the need to share their private datasets and a central entity. This method leverages a communication topology where nodes exchange their locally trained models with neighboring nodes. A key element in this setup is the gossip matrix, a matrix that dictates how information is shared between nodes. Each element in the matrix represents the weight assigned to a node's model during the aggregation process, ensuring that each node's update is a weighted average of its neighbors' models. This matrix plays a crucial role in achieving consensus among the nodes by controlling how much influence each neighbor has during the update phase, and it helps in the efficient distribution of information across the entire network.

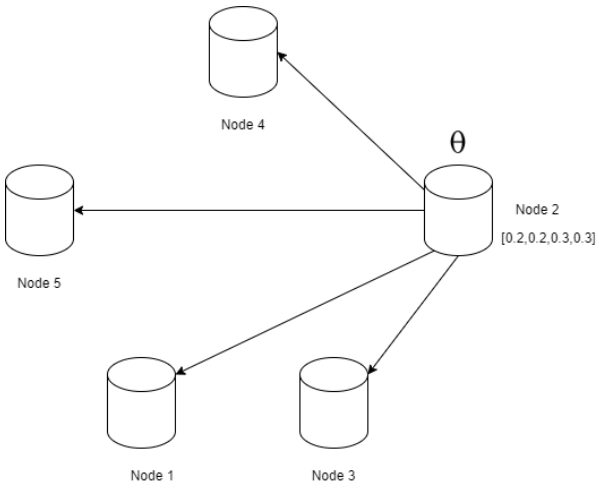

Figure 2.1: Plot from the perspective of node 2 in Decentralized Learning. After the local training, based on a probability matrix [0.2,0.2,0.3,0.3] (a row of the gossip matrix) the node shares the model weights θ to other nodes in the network without going through a central server

In this project, we examined two main types of settings. The first case, which corresponds to each node having its local dataset sampled from the chosen global dataset. Secondly, the case where we form clusters, and then derive nodes with distributions similar to the clusters, such that intra-cluster similarity is high, but outside-cluster similarity is low. These two settings provide a solid layer to investigate the main question of our work, which is to research if it is better to communicate with more similar or not nodes, we will show through our analysis afterwards, that one of the two possibilities is indeed better than the other.

We note here, that we are presented with a personalization problem, as each node n_i looks forward to improving their loss on their unique test dataset T_i.

Notations

Consider a set $\mathcal{N}$ of n nodes that can communicate to improve the local model for their personalized task. Consider $n_i \in \mathcal{N}$, for $i = 1, ..., n$ correspond to the i-th node.

In the non-cluster case, for each node, define S_i as the local training dataset and T_i as the local test dataset for node $i \in [n]$.

For the cluster case, we define $\mathcal{C}$ as a set of clusters, that each has a different distribution sampled from the original dataset. Then, we derive nodes to the cluster, by sampling their own data distributions from their assigned cluster, define S_{ic} as the local training dataset and T_{ic} as the local test dataset, for node $i \in [n]$ and cluster $c \in [c]$.

Let $\mathscr{L} : \Theta \rightarrow \mathbb{R}$ be the loss function associated with a sample z where $\Theta \subseteq \mathbb{R}^{\mathbb{P}}$ represents the parameter space of the model we wish to train.

2.0.2 Non-cluster case

In the first problem setting, we explore the non-IID data issue, specifically, label shift distributions, where each node has a dataset sampled from the global distribution.

This means that the distribution of labels among the nodes is skewed. The objective of each node n_i is to minimize each own data error function F^i:

$$F^i(\theta) = \mathbb{E}_{z \sim S_i}[\mathbb{L}(\theta, z)]$$

In particular, we aim to find the optimal model for each node such that:

$$\theta_i^* = argmin_{\theta \in \Theta} F^i(\theta), \forall i \in [n]$$

A graphical overview is shown in 2.2, we observe that the nodes are spread out over the 2 principal components of a PCA decomposition in a case of heavy heterogeneity. Therefore, each will have a different objective to optimize for in the network. In the case described afterward, the nodes will be quite closer, such as forming clusters in this decomposition.

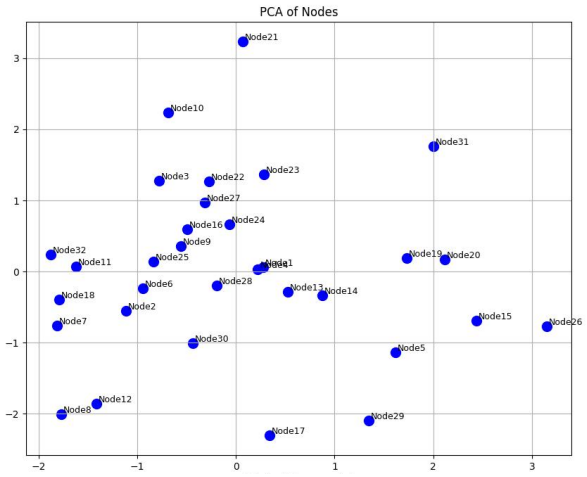

Figure 2.2: Projection of the nodes over the two principal components of a PCA done over the label distribution in a setting of high heterogeneity

2.0.3 Cluster case

In the second problem setting, clusters are sampled from the global distribution, and then nodes from clusters.

The objective of each node n_i is to minimize each own data error function F^{ic}:

$$F^{ic}(\theta) = \mathbb{E}_{z \sim S_{ic}}[\mathbb{L}(\theta, z)]$$

In particular, we aim to find the optimal model for each node such that:

$$\theta^*_{ic} = argmin_{\theta \in \Theta} F^{ic}(\theta), \forall i \in [n]$$

2.0.4 Personalized learning

Customizing a machine learning model involves adapting it to better align with the specific data it will be applied to. This is especially important when working with non-i.i.d. (independent and identically distributed) data, as personalization helps maintain the model's performance across different clients or nodes [10]. Early work on personalization within federated learning (FL) [2] focused on maintaining a single central model, aiming to strike a balance between the diverse data distributions of the nodes involved. However, this created a challenge of balancing global consistency with the need for local adaptation.

Some approaches address this by enabling each node to have its own personalized model, while still contributing to the training of a shared global model. When nodes' data naturally forms clusters, certain FL techniques group similar nodes together to take advantage of these commonalities [14].

In DL, initial efforts to personalize models relied on communication networks that represented the similarity between nodes, often requiring a predefined similarity matrix [3]. However, obtaining such a matrix can be difficult, and in privacy-sensitive environments, sharing data to establish these similarities is usually not possible [5].

Recent approaches have shifted towards using dynamic networks with adaptable communication structures to improve personalization. This work follows that trend, first focusing on personalizing models at the node level, and then exploring how similarities among nodes can be utilized when dealing with clustered data.

2.0.5 Related work in Decentralized Learning

Decentralized learning (DL) in non-IID settings introduces challenges, particularly in optimizing collaboration among nodes with varying data distributions. When nodes possess skewed or non-IID data, this heterogeneity can degrade model performance, making communication strategies critical.

Hsieh et al. [4] highlight how skewed label distributions across nodes negatively affect DL accuracy. They propose SkewScout, which adapts communication based on accuracy loss, a strategy similar to our goal of improving communication between nodes based on data characteristics. Unlike their optimization approach, DBCA algorithm does not consider communication costs and implements a dynamic approach to communication.

Other approaches, like Smart Sampling by Wang et al. [11], dynamically sample data to improve communication efficiency in large-scale DL. Our work expands on this concept by introducing probabilistic communication based on model similarity, which reduces communication overhead while enhancing accuracy. Additionally, we explore whether nodes should communicate with similar or dissimilar models, a question not directly addressed in Smart Sampling.

The Decentralized Adaptive Clustering (DAC) algorithm by Zec et al. [13] forms soft clusters of nodes with similar data. While DAC dynamically adjusts communication between similar nodes, our DBCA algorithm explores a more different approach, focusing on the label skewness wether than feature heterogeneity, also considering both intra- and inter-cluster communication. By allowing nodes to communicate with either similar or dissimilar models, our method generalizes beyond clustering.

Li et al.'s L2C framework [6], which learns collaboration strategies dynamically, further aligns with our goal of improving communication patterns. However, while L2C optimizes collaboration weights, DBCA introduces a probabilistic mechanism based on model similarity to drive communication, providing an alternative method for improving node-to-node collaboration in non-IID environments.

Through these methods, we aim to answer a fundamental question in decentralized learning: can optimizing communication based on model similarity (or dissimilarity) yield better performance in non-IID settings? Our DBCA algorithm builds on these previous studies by systematically adjusting communication probabilities and demonstrating that prioritizing certain node interactions leads to a better accuracy.

Chapter 3

The DBCA Algorithm

In this section, we begin by presenting an overview of our algorithm's design. After, I provide a detailed, formal description of each step. Then, I present how I achieved the result and finally how I modified the algorithm to adapt it in the clustered setting.

3.0.1 Design

In the Distance Based Communication Algorithm (DBCA), we aim at increasing the accuracy of the nodes in a distributed settings in the presence of non-IID. In the network of nodes, each node has a probability of communicating with another node. At each communication round, which happens in a directed graph setting, each node knows to whom it is sending the current local model but is not aware from whom it will receive information. The idea behind this approach is to strategically control which nodes communicate and how frequently they exchange information, based on a notion of "distance", it also takes into account the notion of asymmetry, as in real-world scenarios.

Let's define $P(n_i, n_j)$ the probability to send a model from node i to node j in a given communication round. This probability is defined to be proportional to the distance between the model parameters of the two nodes:

$$P(n_i, n_j)\alpha \frac{1}{||\theta_i^t - \theta_j^{t'}||^p}$$

Our implementation computes the distance to the last received model, therefore, in order to achieves this efficiently, each nodes keeps in memory the last model version of each node it has received from. Once, we have computed all the probabilities, we normalize our matrix for the sum to be 1.

$$P(n_i, n_j) = \frac{P(n_i, n_j)}{\Sigma_{n=0, n!=i}^{N} P(n_i, n_n)}, \forall\, j,\ s.t,\ j! = i$$

For the first iteration, we start with the unbiased case, where each node has an equal probability to communicate with any other node. After, for the following rounds, for nodes that have not yet received a model from every node, they average all other probabilities as a proxy for missing node models.

As observed, the communication between nodes is decided by how we decide to choose the distance metric and the implementation of the proportionality factor.

- First of all, the distance metrics usually tried are the L1 (Manhattan distance) and L2 (euclidean distance) distances, corresponding to p=1 and p=2, in our formula. With L1 distance, models will be closer overall, and with the L2, models further apart will be more distant. Though, in our experiences, we find out that this distance, did not had a significant impact on our accuracy improvement.
- Secondly, when we set our probability to be inversely proportional to the distance of the models, it implies that the more the models are similar, the more the probability will be higher. And, inversely, when we set our probability to be proportional to the distance of the models, it implies that the more the models are dissimilar, the more the probability will be higher. Therefore, this a means of choosing if we would like to optimize for nodes that are more or less similar to the local node.

Figure 3.1: A communication round after the local training from the point of view of a node n_i. In DBCA, each node maintains a probability matrix containing the probabilities to communicate with some node at each round. When the training is finished, the local node n_i sends its local model to the chosen nodes in the network, after that the node updates its local probability matrix if it receives new models

Algorithm 1 DBCA Algorithm from point of view of node n_i

Parameters: number of communication rounds T; outdegree k; number of nodes N; learning rate η; model parameters θ; probability vector γ; distance parameter p; weighting factor wf
for t in $0, 1, ..., T$ communication rounds **do**
  $\theta_i^{t+1} \leftarrow gradient\_descent(\widehat{\theta}_i^t, \eta)$, in local training
  send θ_i^{t+1} to k sampled nodes chosen with respect to γ^{t+1} probabilities
  receive $\theta_{received}^{t+1}$ from nodes that chose n_i as target for this round t and update local received models
  for j in $0, 1, ..., N$ nodes, $j != i$ **do**
    $d_{ij} \leftarrow ||\theta_i^{t+1} - \theta_j^t||^p$
    $\gamma_{ij} \leftarrow e^{-wf * d_{ij}}$
  end for
  $\gamma^{t+1} \leftarrow normalize(\gamma^{t+1})$
  $\widehat{\theta}_i^{t+1} \leftarrow average\_models(\theta_i^{t+1}, [\theta_{received_1}^{t+1}, \theta_{received_2}^{t+1} ...])$
end for

3.0.2 Achieving node convergence

One of the important parts of the research in the setting was to find an innovative way to make the nodes converge to some neighbors. Indeed, in our first implementations we spend quite some time researching methods to improve the accuracy of the nodes with the previous settings, but we were unable to make any significant progress. One of the main drivers for that inability of improvement, was that the nodes communicated like the epidemic learning algorithm, they randomly sent their local model to all the nodes, instead of a selected subgroup, the probability matrix had more or less an equal weighting for all nodes. But, if the node setting would like to learn from other nodes that are significantly similar or dissimilar and not be submerged by the noise of unwanted models, they need to be able in the end, to have a sort of sub-network by sending models to the selected nodes, as they are different or similar they will or not increase the probability to communicate back. However, by only computing the probabilities as we did in the beginning by plainly making them proportional to the distance, as mentioned, no significant result was achieved.

Nevertheless, if we add an exponential to our computed distance, we find that nodes will tend to converge to a subgroup of the nodes, equivalent to the outdegree of each node. The probabilities are then written as:

$$P(N_i, N_j)\alpha \frac{1}{e^{||\theta_i^t - \theta_j^{t'}||^p}}$$

In 3.2, the plot shows how a node will gradually only communicate with a subgroup of the network, in the beginning, the node communicates with everyone, such that it sends its local model to all the nodes in the network, though, over time it will only send the local model to some nodes in the network as observed by only having 5 different bars.

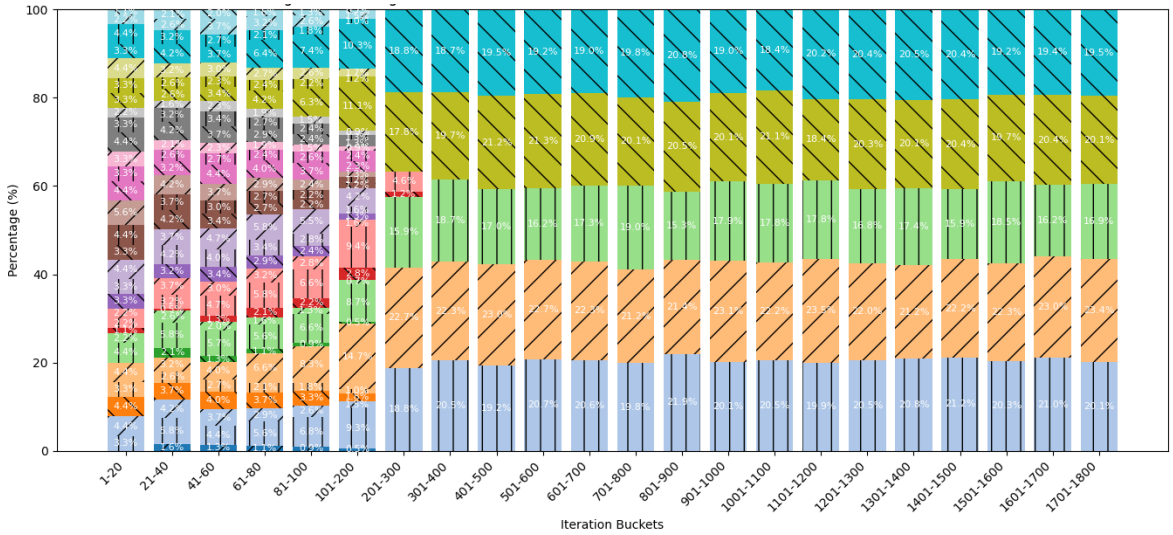

Figure 3.2: Plot from the perspective of a node showing the percentage of models sent to each node in the network over buckets of 100 communication rounds. Each chunk of the bar represents the percentage of models sent to some node, 20% meaning that 20% of the models sent in the 100 rounds communication bucket were sent to the same node.

3.0.3 Cluster Design

Another part of the research was to adapt the network such that it could be divided by clusters. We form clusters of nodes where the data distributions are very different from cluster to cluster, but relatively similar for nodes inside of a cluster.

The algorithm that is described here is a choice among other possible ones, as they yield similar results. In this adaption of DBCA to the cluster case, we note an important change, in some communication rounds, the node will communicate only within the cluster and in others with nodes outside the cluster. The implementation choice made here, is to communicate with the outside nodes every predefined number of communication rounds. To choose the nodes to communicate within the cluster, the node uses the DBCA in the cluster, and to choose the outside nodes, it randomly chooses

nodes outside the cluster similarly to Epidemic Learning.

Figure 3.3: A communication round after the local training from the point of view of a node n_i belonging to cluster c. In DBCA cluster setting, each node maintains a probability matrix containing the probabilities of communication with every node inside the cluster at each round, to communicate outside the cluster, the local node chooses a foreign node randomly. The node then sends its local model to the chosen nodes in the network and updates the local model storage in case it receives new ones

Algorithm 2 DBCA Algorithm from point of view of node n_i in the cluster setting

Parameters: number of communication rounds T; outdegree k; number of nodes N; learning rate η; model parameters θ; probability vector inside the cluster γ; distance parameter p; weighting factor wf; cluster c; communication modulo m
for t in $0, 1, ..., T$ communication rounds **do**
  $\theta_i^{t+1} \leftarrow gradient\_descent(\widehat{\theta}_i^t, \eta)$, in local training
  if $t\%m == 0$ **then**
    send θ_i^{t+1} to k sampled nodes chosen nodes outside the cluster c
    receive θ^{t+1} from nodes that chose n_i as target for this round t
  else
    send θ_i^{t+1} to k sampled nodes chosen with respect to γ^{t+1}
    receive $\theta_{received}^{t+1}$ from nodes that chose n_i as target for this round t and update local received models
    for j in $0, 1, ..., N$ nodes inside the cluster c **do**
      $d_{ij} \leftarrow ||\theta_i^{t+1} - \theta_j^t||^p$
      $\gamma_{ij} \leftarrow e^{-wf * d_{ij}}$
    end for
    $\gamma^{t+1} \leftarrow normalize(\gamma^{t+1})$
  end if
  $\widehat{\theta}_i^{t+1} \leftarrow average\_models(\theta_i^{t+1}, [\theta_{received_1}^{t+1}, \theta_{received_2}^{t+1}...])$
end for

An observation that we can mention, is that the non-clustered case is just a specific case of the clustered case, where each node is its own cluster. Therefore, the cluster case is a generalization of our methodology.

Note that other approaches were also explored, such as having a proxy node to imitate the label distribution inside the cluster, but the underlying results were similar, so we omit their description as they yield the same conclusion as the chosen algorithm for the cluster communication setting.

Chapter 4

Evaluation

4.1 Experimental setup

4.1.1 Datasets and model

The CIFAR-10 dataset is a widely-used benchmark in computer vision, consisting of 60,000 color images across 10 mutually exclusive classes (e.g., airplane, automobile, bird). Each image is 32x32 pixels in size. The dataset is evenly divided into 50,000 training and 10,000 test images. It is commonly used to evaluate image classification models, particularly in deep learning, due to its diverse and balanced classes, making it a standard choice for model benchmarking and educational purposes. The CIFAR-100 dataset, similar to CIFAR-10, is another popular benchmark in computer vision but with a larger number of classes, therefore offering a more complex task.

LeNet is a pioneering convolutional neural network (CNN) architecture developed by Yann LeCun and colleagues in the late 1980s. Originally designed for handwritten digit recognition in the MNIST dataset, LeNet consists of a series of convolutional and subsampling (pooling) layers, followed by fully connected layers leading to a softmax classifier. The architecture is relatively simple, with seven layers (including convolutional, pooling, and fully connected layers) and demonstrates the power of CNNs in extracting spatial hierarchies from images. LeNet laid the foundation for modern deep learning architectures and remains influential in the development of more complex CNNs.

4.1.2 Baselines

We chose to compare against two main baselines:

- DAC, which, to the best of our knowledge, is one the latest and top-performing approach for personalized decentralized learning on clustered non-IID data. DAC employs a dynamic communication topology and has been evaluated in environments with clusters resembling our setup. This approach adjusts communication weights between nodes according to their data distributions, thereby improving learning efficiency and performance in clustered scenarios.
- Epidemic learning approach, which is recognized as an effective method for decentralized learning, particularly in large-scale, distributed environments. This approach is particularly well-suited for non-IID data scenarios, as it facilitates robust information sharing across nodes, ensuring that diverse data distributions are adequately captured. Its adaptability and efficiency make it a strong benchmark for evaluating decentralized learning methods.

4.1.3 Experiment settings

The principal research is about cluster and non-cluster accuracy in a non-IID setting. The main research we have done was realized with the following parameters: 32 Nodes, 1600 communication rounds and outdegree of 5 (log of the number of nodes) for the non-cluster case, outdegree of 3 for the cluster communication (log of the size of the cluster). Then 10 local training rounds, a learning rate of 0.05, euclidean distance and a weighting factor of 10.

In our setting, we obtain niid through dirichlet partitioning. It is a technique used to simulate non-IID data distributions by dividing a dataset into subsets according to a Dirichlet distribution. The method allows control over the degree of data skewness across partitions via the α parameter, enabling realistic modeling of heterogeneous data distributions in decentralized learning environments. This approach is particularly useful in decentralized learning to create diverse, non-overlapping data subsets that better reflect real-world scenarios. To create the cluster,s the experiments set up 2 dirichlet partitionings, the first one to create the clusters, and the second one to create the node datasets from the clusters data. Note again, that we focus on labels skew in the setting, such that each node has a significantly different label distribution.

In the presented results, we chose to run Epidemic Learning for 40 communications rounds before DBCA, it makes the information flow efficiently and improves learning capacity compared to the vanilla version without it.

4.1.4 Results

The main focus of our research was to find out if by communicating with nodes that are more similar than others, the accuracy would improve or not.

No cluster case

One of the highlight of our work is that we proved that in some heterogeneity levels, communicating with nodes that are different, data wise (i.e different label distributions), to the local node ends ups improving the overall accuracy of the setting.

To begin, observe the plot 4.1, it shows that our method outperforms our baselines in the setting where alpha is 0.5, therefore in an environment where heterogeneity is present but not extreme. The believed explanation behind this, is that nodes improve their prediction power on class samples present in low percentages by sharing their model with nodes that are far away data related. As the node receiving the model, if very different will likely communicate back with the original node sender. By receiving information of models trained on various data, it will improve the local generalization power of the local node on samples present in low quantity in the training data.

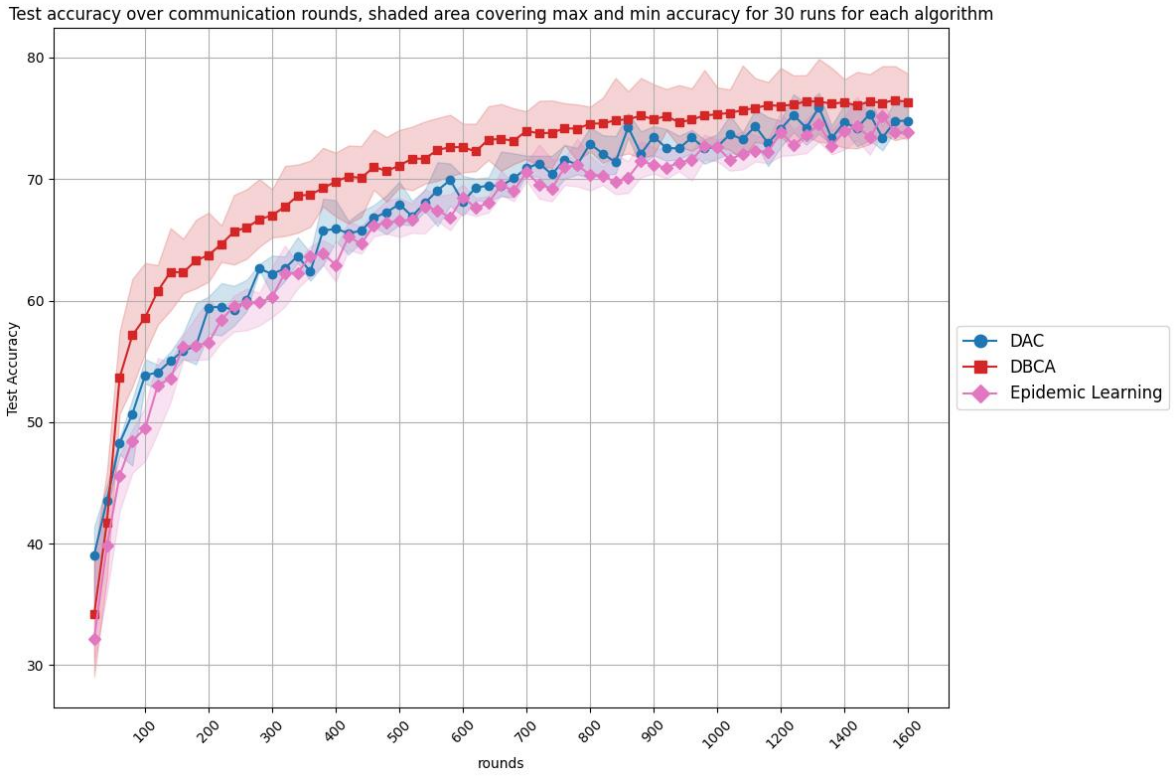

Figure 4.1: Performances of the baselines compared to DBCA in a setting with alpha=0.5

To illustrate this, take weighting factors of 10/-10 and observe the accuracy results presented in plot 4.2. In this case, we have the following communication probabilities for nodes n_i and n_j at time step t (t' is the time step of the round where we last received a model from node n_j):

- For a weighting factor of 10, we have the following communication strategy:

$$P(n_i, n_j) = \frac{1}{e^{-10*||\theta_i^t - \theta_j^{t'}||^2}} = e^{10*||\theta_i^t - \theta_j^{t'}||^2}$$

- For a weighting factor of -10, we have the following communication strategy:

$$P(n_i, n_j) = \frac{1}{e^{10*||\theta_i^t - \theta_j^{t'}||^2}}$$

As a result, we can deduce the following:

- For a weighting factor of 10, the higher the distance between the model parameters, the higher the probability of communicating between the nodes. Therefore, the more dissimilar the nodes, the more likely they're going to communicate.
- For a weighting factor of -10, the higher the distance between the model parameters, the lower the probability of communicating between the nodes. Therefore, the more dissimilar the nodes, the less likely their going to communicate.

This dissimilar and similar communication methods are compared in the plot 4.2. We indeed observe that in the case of some heterogeneity, with alpha being 0.5, we obtain a better accuracy when communicating with dissimilar nodes.

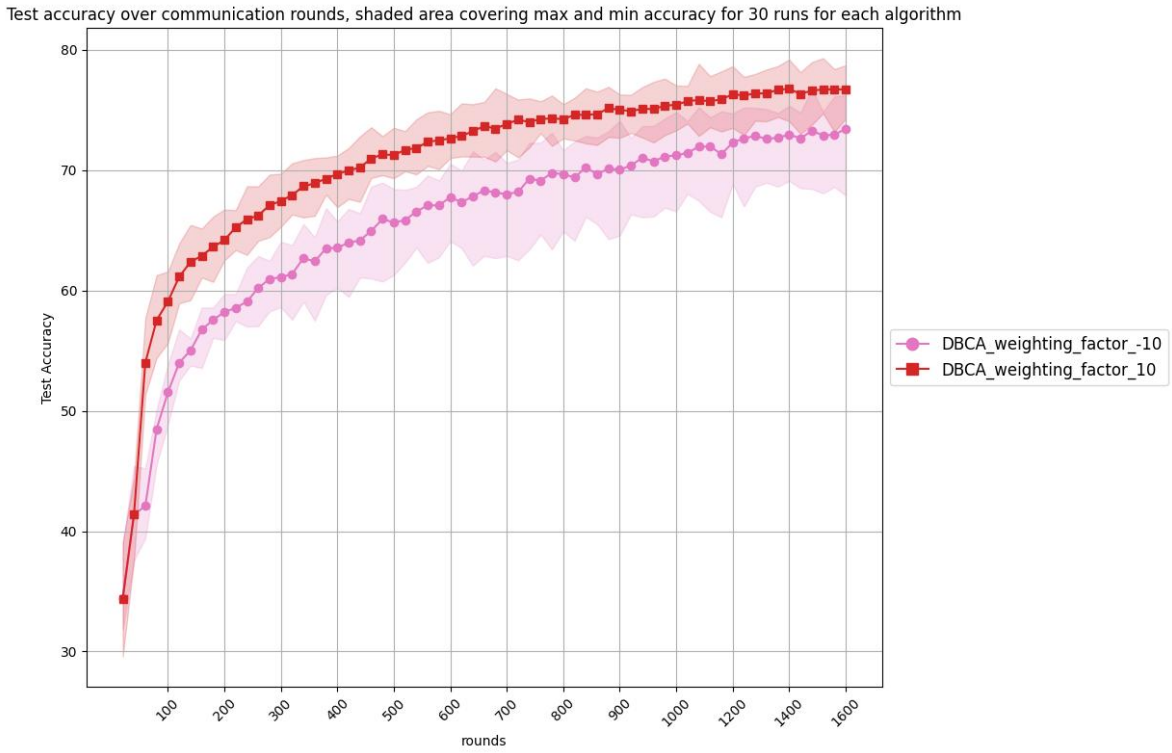

Figure 4.2: Performance of DBCA with different weighting factor in a setting with alpha=0.5

Nevertheless, the plot 4.3 shows that our method does worse in scenarios where heterogeneity is extreme. As indicated, when alpha is 0.1, the setting scenario implies a quite heavy non-IDD presence. The nodes are so skewed at label distribution, that, the vast majority of the train dataset is made of a single label. In this case then, communication with other nodes ends up decreasing the local accuracy compared to not communication at all with other nodes, as observed in the plot.

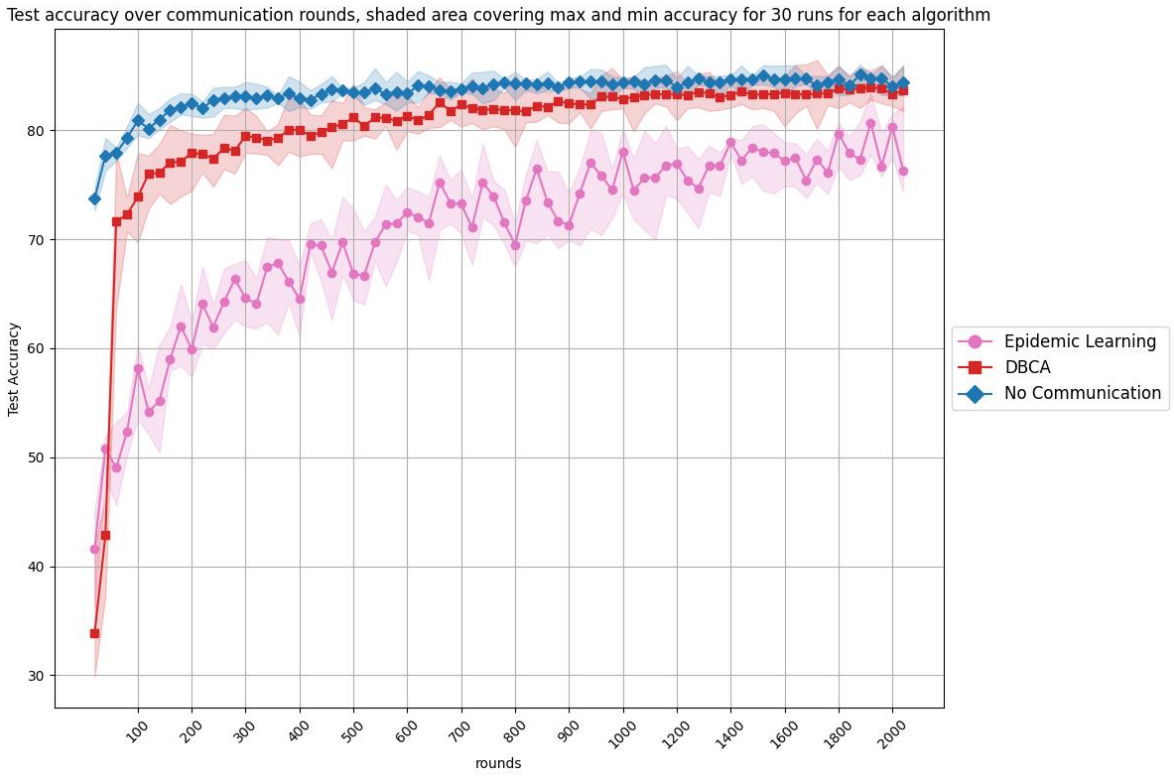

Figure 4.3: Performance of DBCA compare to Epidemic Learning and DBCA without communication (local training only) in a setting with alpha=0.1

However too, when there is no heterogeneity, such as when alpha is close to 1, our algorithm under-performs epidemic learning, as one can observe in the plot 4.4. A behavior that was observed during the experiments, is that DBCA will not converge to communicate to some restricted nodes if there is not enough non-IDD in the setting, and thus will behave like Epidemic Learning.

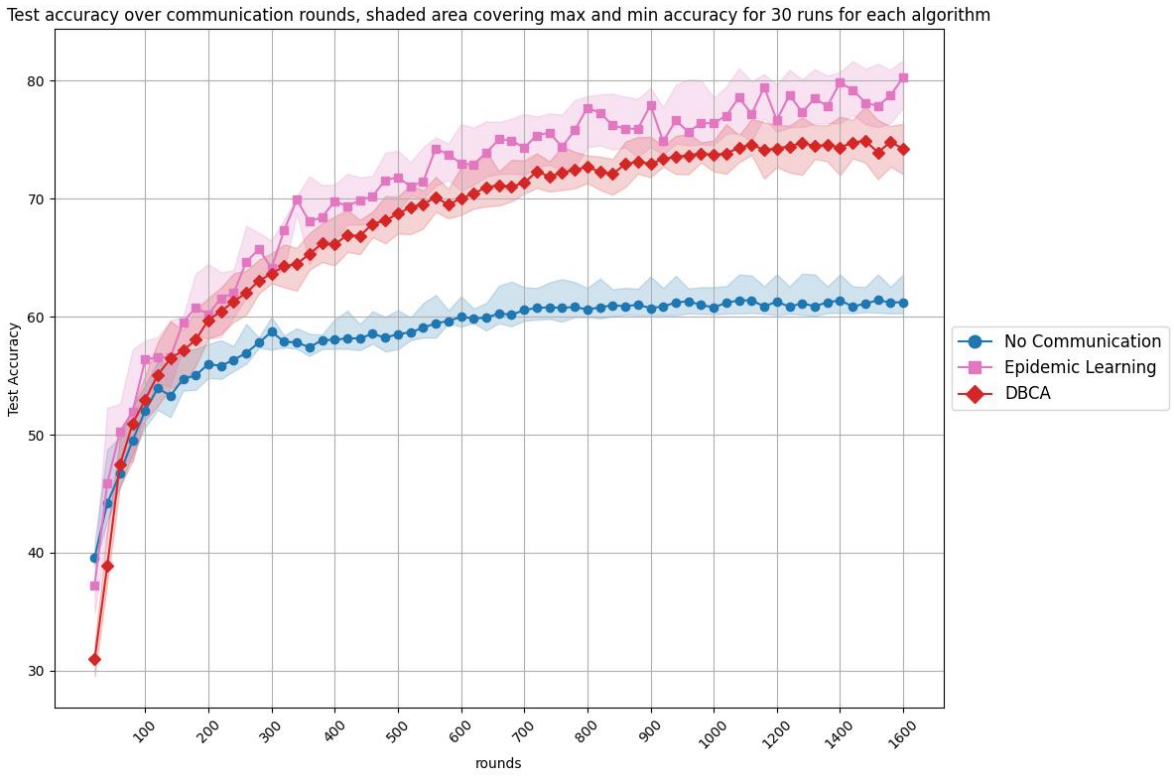

Figure 4.4: Performance of DBCA compared to Epidemic Learning and DBCA without communication (local training only) in a setting with alpha close to 1

The algorithm also performs well on other more diverse datasets, take for instance CIFAR-100, in 4.5 we have the performance of DBCA which still outperforms Epidemic Learning in more complex tasks.

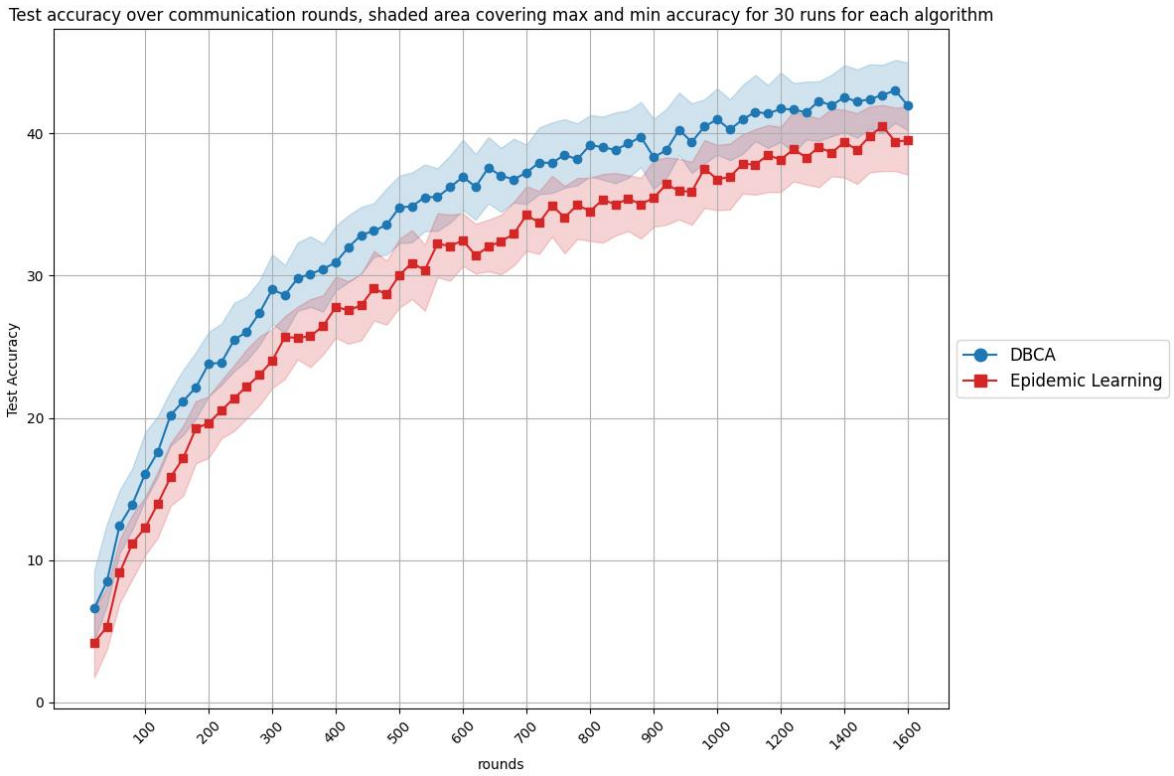

Figure 4.5: Performance of DBCA compared to Epidemic Learning on CIFAR-100 for an alpha of 0.5

Clustered case

In the clustered case, communicating with non-similar nodes also improves accuracy.

To show this research result, we present this setup: create 4 clusters, nodes communicate mostly within the cluster, but from time to time they randomly choose nodes outside of their cluster, and send their model. We plot one of these experiments in 4.6, as observed, during rounds where nodes communicate with nodes outside of their cluster the global accuracy of the setting improves as seen in the spikes.

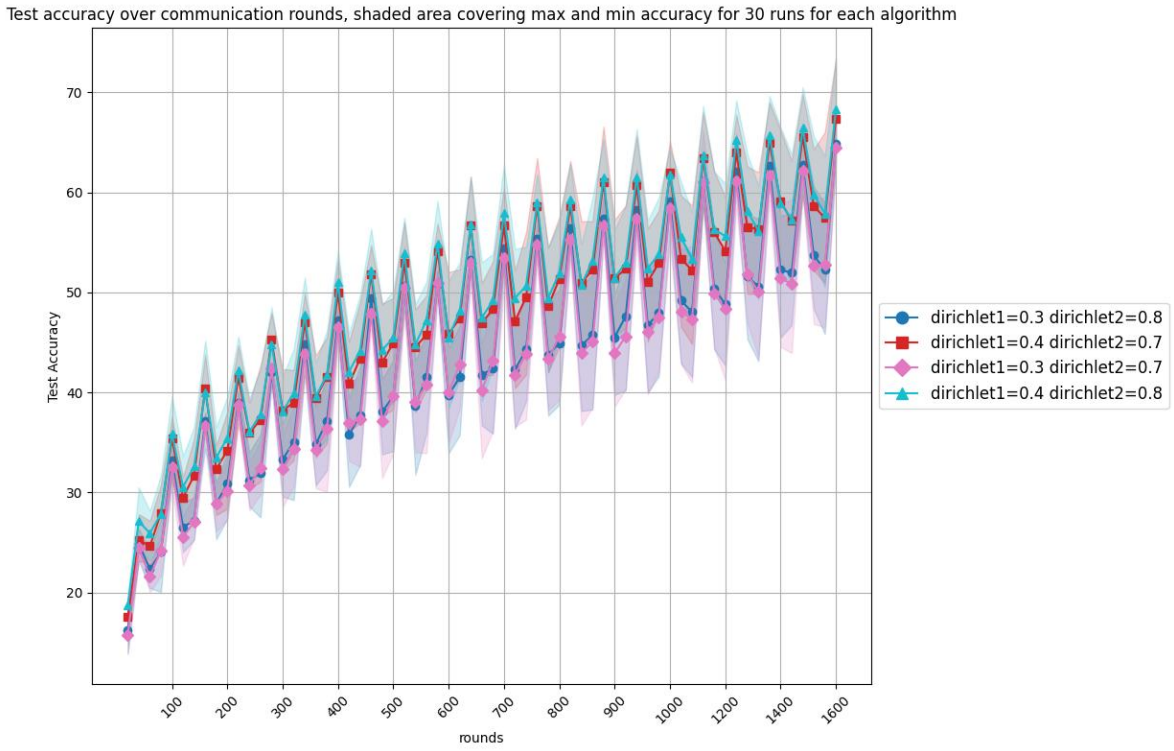

Figure 4.6: Performance of DBCA on different cluster setups, dirichlet 1 representing the first dirichlet split in the data to isolate the cluster, and dirichlet 2 to sample nodes from the clusters

If we take the case with the first split equal to alpha_1=0.3 and after alpha_2=0.7, and compare it to other possible communication methods for the clusters in 4.7 we observe that for this cluster setup we actually underperform Epidemic Learning. However, we also observe that, by communicating only with nodes outside the cluster we obtain better results than by limiting the communication inside of it. As we have made our setting such that nodes belonging to the same cluster are very similar, we showed differently that by communicating with nodes that are dissimilar we can improve the overall accuracy in decentralized learning. Another similar example can be found here 7.1.

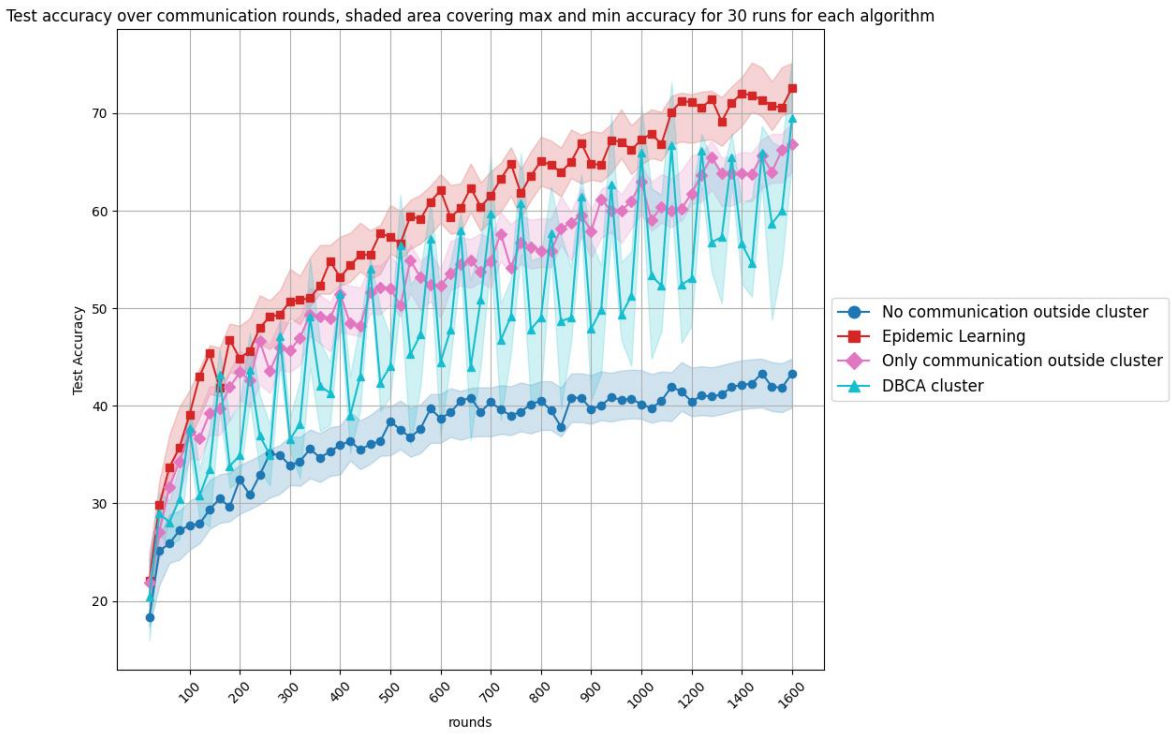

Figure 4.7: Performance of DBCA cluster, compared to other possible communication methods in a split with alpha_1=0.3 and alpha_2=0.7

4.1.5 Sensitivity to hyperparameters

During the experiments done in our research, we tried out more parameters in our setting. To mention a few:

- Learning rate tuning: 0.025, 0.05, 0.1, adding learning rate decay
- Scaling 8 nodes, 16 nodes, 32 nodes, 64 nodes
- Distance: L1 distance, KL distance
- Number of rounds
- Knowledge distillation

Nevertheless, none of these parameter tunings had a significant impact on our results, which highlights that the parameter adjustment mentioned and done in our experiments were the principal drivers of the results we managed to obtain.

Chapter 5

Conclusion

This work has made a contribution to the field of decentralized learning (DL) in non-IID settings, particularly through the development and evaluation of the Distance Based Communication Algorithm (DBCA). By leveraging model distance to guide inter-node communication, DBCA introduces a novel probabilistic approach that facilitates both intra and inter cluster exchanges. Our findings show that communicating with dissimilar nodes can significantly enhance overall accuracy in decentralized learning systems. This discovery highlights the importance of diversity in information sharing, especially in heterogeneous data environments.

The effectiveness of DBCA is particularly evident in scenarios with moderate data heterogeneity. In these cases, DBCA consistently outperformed existing methods, including Epidemic Learning and Decentralized Adaptive Clustering (DAC), demonstrating its robustness and adaptability. However, we also observed that in extreme cases of heterogeneity or near-homogeneity, the benefits of DBCA are diminished, emphasizing the need to tailor communication strategies based on the degree of data skewness.

In clustered settings, we showed that by allowing nodes to occasionally communicate with those outside their cluster, it would improve overall model performance. This suggests that even in highly similar environments, introducing diversity in communication can enhance learning outcomes. Additionally, our experiments on more complex datasets, such as CIFAR-100, confirmed the potential of DBCA to generalize across a wide range of tasks, further showcasing its versatility in decentralized learning scenarios.

Moreover, our experiments demonstrated that DBCA is robust to hyperparameter tuning, with its core benefits largely unaffected by adjustments in learning rates, node scaling, or communication rounds. This robustness makes DBCA an interesting tool for decentralized systems where practical constraints may limit the ability to fine-tune models across various nodes.

In conclusion, the DBCA algorithm advances decentralized learning by strategically balancing

model similarity and diversity in node communication, providing a novel method for optimizing performance in non-IID environments. The results of this work could gear the design of new decentralized learning systems ideas, particularly in real-world applications where data heterogeneity is the norm.

Chapter 6

Discussion

Looking forward, there are several promising avenues for future research. One direction is to explore dynamic adaptations of DBCA that can respond to varying degrees of heterogeneity over time, allowing the algorithm to adjust its communication probabilities as data distributions evolve. Additionally, extending DBCA to handle more complex real-world scenarios—such as those involving time-varying networks or highly imbalanced data—could unlock further potential in fields like healthcare, finance, and distributed sensor networks. Another compelling direction would be to investigate hybrid approaches that combine DBCA with other advanced decentralized learning techniques, potentially creating more robust and efficient systems capable of handling even more diverse and extreme data environments.

By continuing to refine and expand upon DBCA, future work can address the limitations observed in extreme heterogeneity and homogeneity, pushing the boundaries of decentralized learning in increasingly complex and challenging settings.

Also, for the clustered case, new algorithm implementations inspired by DBCA can by tried, as using cluster proxies, PCA decomposition of the data nodes to then communicate and further ideas.

Chapter 7

Appendix

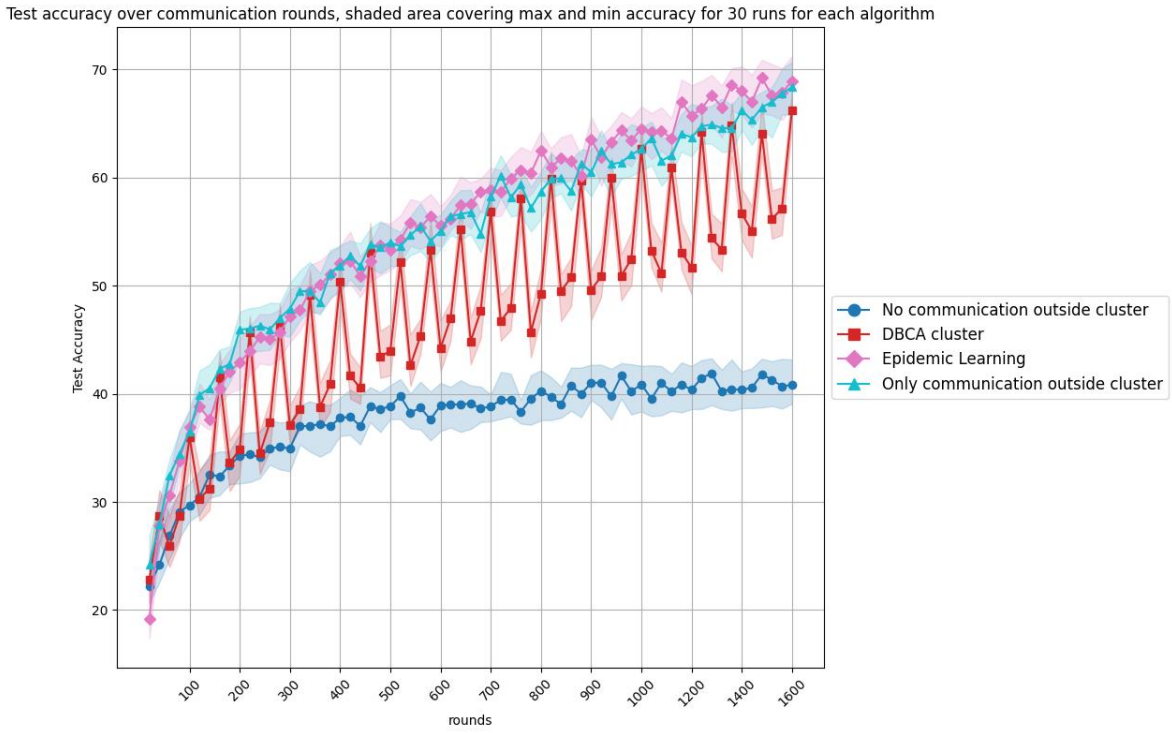

Figure 7.1: Performance of DBCA cluster, compared to other possible communication methods in a split with alpha_1=0.5 and alpha_2=0.5

Bibliography

[1] Yatin Dandi, Anastasia Koloskova, Martin Jaggi, and Sebastian U. Stich. *Data-heterogeneity-aware Mixing for Decentralized Learning*. 2022. arXiv: 2204.06477 [cs.LG]. URL: https://arxiv.org/abs/2204.06477.

[2] Alireza Fallah, Aryan Mokhtari, and Asuman E. Ozdaglar. "Personalized Federated Learning with Theoretical Guarantees: A Model-Agnostic Meta-Learning Approach". In: *Neural Information Processing Systems*. 2020. URL: https://api.semanticscholar.org/CorpusID:227276412.

[3] Maya Gunawardena, Penny Bishop, and Kithmini Aviruppola. "Personalized learning: The simple, the complicated, the complex and the chaotic". In: *Teaching and Teacher Education* 139 (Mar. 2024), p. 104429. DOI: 10.1016/j.tate.2023.104429.

[4] Kevin Hsieh, Amar Phanishayee, Onur Mutlu, and Phillip B. Gibbons. *The Non-IID Data Quagmire of Decentralized Machine Learning*. 2020. arXiv: 1910.00189 [cs.LG]. URL: https://arxiv.org/abs/1910.00189.

[5] Judy Hughey. "Individual Personalized Learning". In: *Educational Considerations* 46 (Jan. 2020). DOI: 10.4148/0146-9282.2237.

[6] Shuangtong Li, Tianyi Zhou, Xinmei Tian, and Dacheng Tao. "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.

[7] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. *Can Decentralized Algorithms Outperform Centralized Algorithms? A Case Study for Decentralized Parallel Stochastic Gradient Descent*. 2017. arXiv: 1705.09056 [math.OC]. URL: https://arxiv.org/abs/1705.09056.

[8] H. Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Agüera y Arcas. *Communication-Efficient Learning of Deep Networks from Decentralized Data*. 2023. arXiv: 1602.05629 [cs.LG]. URL: https://arxiv.org/abs/1602.05629.

[9] Martijn de Vos, Sadegh Farhadkhani, Rachid Guerraoui, Anne-Marie Kermarrec, Rafael Pires, and Rishi Sharma. *Epidemic Learning: Boosting Decentralized Learning with Randomized Communication*. 2023. arXiv: 2310.01972 [cs.LG]. URL: https://arxiv.org/abs/2310.01972.

[10] Candace Walkington and Matthew Bernacki. "Appraising research on personalized learning: Definitions, theoretical alignment, advancements, and future directions". In: *Journal of Research on Technology in Education* 52 (July 2020), pp. 235–252. DOI: 10.1080/15391523.2020.1747757.

[11] Lin Wang, Yang Chen, Yongxin Guo, and Xiaoying Tang. *Smart Sampling: Helping from Friendly Neighbors for Decentralized Federated Learning*. 2024. arXiv: 2407.04460 [cs.LG]. URL: https://arxiv.org/abs/2407.04460.

[12] Lu Yu, Wenjing Nie, Lun Xin, and Manxue Guo. "Clustered Federated Learning Based on Data Distribution". In: Nov. 2021, pp. 1–5. DOI: 10.1145/3503047.3503102.

[13] Edvin Listo Zec, Ebba Ekblom, Martin Willbo, Olof Mogren, and Sarunas Girdzijauskas. *Decentralized adaptive clustering of deep nets is beneficial for client collaboration*. 2022. arXiv: 2206.08839 [cs.LG]. URL: https://arxiv.org/abs/2206.08839.

[14] Ling Zhang, James D. Basham, and Sohyun Yang. "Understanding the implementation of personalized learning: A research synthesis". In: *Educational Research Review* 31 (2020), p. 100339. ISSN: 1747-938X. DOI: https://doi.org/10.1016/j.edurev.2020.100339. URL: https://www.sciencedirect.com/science/article/pii/S1747938X19306487.