An Autoencoder-based Decentralized Clustering Leveraging Model Aggregation Fusion Strategy

by

Nahid Hasan
22366036

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
M.Sc. in Computer Science

Department of Computer Science and Engineering
Brac University
May 2024

Declaration

It is hereby declared that

1. The thesis submitted is my own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Nahid Hasan
22366036
nahid.hasan1@g.bracu.ac.bd

Approval

The thesis titled "An Autoencoder-based Decentralized Clustering Leveraging Model Aggregation Fusion Strategy" submitted by

1. Nahid Hasan (22366036)

Of Summer, 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of M.Sc. in Computer Science on May 25, 2024.

Examining Committee:

Examiner:
(External)

Dr. Ashis Talukder, Ph.D.
Associate Professor
Department of Management Information Systems (MIS)
University of Dhaka

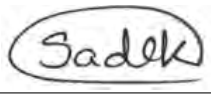
Examiner:
(Internal)

Dr. Md. Ashraful Alam
Associate Professor
Department of Computer Science and Engineering
Brac University

Supervisor:
(Internal)

Dr. Md. Golam Rabiul Alam
Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)



Dr. Md Sadek Ferdous
Associate Professor
Department of Computer Science and Engineering
Brac University

Chairperson:
(Chair)

Sadia Hamid Kazi, Ph.D.
Associate Professor
Department of Computer Science and Engineering
Brac University

Abstract

Unsupervised clustering plays a crucial role in various real-life applications. It works by grouping similar data points together based on certain features or characteristics, without the use of predefined labels. The process generally starts with gathering data in a centralized system that are to be clustered. This data could be in the form of numerical features, text, images, or any other type of information. The exponential expansion of digital transformation, the Internet of Things (IoT), social media, and online platforms has precipitated an unprecedented surge in data generation. This proliferation is characterized by an incessant stream of information flowing from various sources, encompassing user interactions, sensor readings, online transactions, and more. This deluge of data poses both challenges and opportunities for businesses, governments, and individuals alike. The ever-increasing amount of data poses both opportunities and challenges. So, gathering, managing, processing this amount of data in a centralized system requires time and is a very tough process. Additionally, concerns related to data privacy, security, and ethical considerations become more prominent as data volumes continue to grow. Moreover, it's important to respect individuals' privacy rights and adhere to relevant data protection laws and regulations. Federated learning addresses concerns about data volume and privacy by leaving user data on devices. Federated unsupervised representation learning is an architecture that pre-traines deep neural networks utilizing unlabeled input in a federated fashion via unsupervised representation learning. In centralized settings, model-based clustering approaches demonstrate significant effectiveness. These methods rely on statistical models to identify underlying patterns and group data points accordingly. By leveraging sophisticated algorithms, model-based clustering can efficiently handle complex data structures and accurately partition datasets into meaningful clusters. This approach enables centralized systems to efficiently organize and analyze large volumes of data, facilitating insights and decision-making processes across various domains. Moreover, model-based clustering offers flexibility in accommodating different data distributions and can adapt to diverse clustering requirements, making it a versatile tool for centralized data analysis tasks. In contrast to the centralized setup, this way of clustering in federated settings is still relatively unexplored, maybe because training models in a highly diversified context using the FedAvg method is more difficult. The normalizing flow model is used by the recently announced Unsupervised Iterative Federated Clustering (UIFCA) Algorithm to perform clustering on unlabeled datasets in federated environments. The IFCA framework, which tackles the problem of very varied settings, is the foundation of UIFCA. A novel approach for decentralized clustering utilizing proposed model parameter aggregation strategy FednadamN in conjunction with the deep generative model autoencoder is introduced. FednadamN combines the benefits of two cutting-edge optimization methods for federated learning: Adam and Nadam. Adam optimization offers quick convergence and resilience to noisy data by using adaptive learning rates based on the first and second moments of gradients. Adam is expanded by Nadam with the use of Nesterov accelerated gradients, hence increasing the stability and speed of convergence. The method addresses the challenge of clustering in decentralized settings by leveraging the collective intelligence of distributed nodes while preserving data privacy and minimizing communication overhead. By aggregating model parameters across decentralized nodes and em-

ploying Autoencoder-based representations, efficient clustering is enabled efficient clustering without the need for central data storage or coordination. This approach promises to enhance scalability, privacy, and performance in decentralized clustering tasks across various domains. Additionally, a comparison between the tailored approach and the current technique using benchmark datasets is offered. The following four benchmark datasets were used: image segmentation, protein localization, letter image recognition, and vowel deterrence. The suggested technique for clustering letter image recognition data has produced the greatest mutual information score of 1.192 and highest v measure score of 0.373 using the kmeans algorithm. However, FedAvg's fuzzy k means algorithm yields the highest rand index score of 0.925. The proposed approach for clustering Deterding Vowel Recognition Data has the highest v measure score of 0.264 and the highest rand index score of 0.850 when using the kmeans algorithm; however, it performs less well than FedAdam, which uses the minibatch kmeans algorithm to show a v measure score of 0.258. The proposed approach for clustering Protein Localization Data yields the greatest rand score 0.774 , highest mutual info score 0.908 , and highest v measure score 0.527 while utilizing the minibatch kmeans algorithm. The proposed method for clustering Image Segmentation data yields the greatest mutual information score of 1.084, the highest rand score of 0.849, and the highest v measure score of 0.565 when utilizing the minibatch kmeans algorithm. This result demonstrates the suggested approach's improved performance and its potential applicability for various clustering goals. The enhanced efficiency of this method makes it a valuable tool for diverse clustering tasks. Its robustness and adaptability underscore its utility in different contexts. Moreover, the approach's superior outcomes suggest broader relevance across multiple domains.

Keywords: Federated Learning, Clustering, Unsupervised Learning, Auto-encoder, Strategy.

Acknowledgement

Firstly, all praise to the Great Allah for whom my thesis have been completed without any major interruption.
Secondly, to my supervisor Dr. Md. Golam Rabiul Alam sir for his kind support and advice in my work. He helped me whenever I needed help.
And finally to my parents without their throughout support it may not be possible. With their kind support and prayer I am now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

The capacity to glean important insights from massive volumes of data is essential in today's data-driven environment. Conventional clustering methods have long been recognized as indispensable instruments for arranging data into logical groupings, facilitating effective analysis and decision-making. In the field of data clustering, however, the emergence of decentralized systems brings with it both new potential and difficulties. Decentralization has several benefits over centralized methods. It is defined by the dispersion of data and computational power among numerous nodes or devices. It increases scalability, strengthens resilience against single points of failure, and protects data privacy. Within this framework, a viable paradigm for data organization and analysis in remote systems is decentralized clustering. In this research, novel decentralized clustering techniques based on autoencoder models is introduced, complemented by our proposed FednadamN strategy. Leveraging the power of autoencoders, which are deep learning architectures capable of learning efficient representations of input data, our approach aims to partition distributed data into meaningful clusters without compromising individual privacy or data integrity.

1.1 The Motivation behind decentralized clustering

These days, the construction of machine learning models heavily relies on aggregated user data within centralized systems. Existing methodologies encounter a formidable hurdle in accommodating the exponential growth of data generated by Internet of Things (IoT) devices[1]. The sheer volume and diversity of data streams pose a challenge to conventional aggregation techniques. As IoT devices proliferate, the scalability and adaptability of data processing methods become increasingly critical. Addressing this challenge requires innovative approaches capable of efficiently handling the dynamic nature of IoT-generated data. Developing robust frameworks for integrating and analyzing IoT data remains a pressing concern in contemporary machine learning research. Clustering is indispensable when dealing with vast and ever-expanding datasets due to its versatility and effectiveness in data analysis and organization. As data proliferates, the imperative to handle and analyze it effectively intensifies. Efficient management of large datasets becomes paramount for extracting valuable insights. Proper handling ensures accurate decision-making and meaningful outcomes in various domains. Robust analytical techniques are essen-

tial for navigating the complexities of abundant data sources. The ability to derive actionable insights from vast datasets distinguishes successful endeavors in today's data-driven landscape. It serves as a vital tool for dissecting enormous and intricate datasets, enabling the extraction of invaluable insights[2]. Clustering aids in uncovering meaningful patterns and structures within the data, empowering decision-making processes. Additionally, it plays a pivotal role in enhancing data-driven applications across diverse domains. By grouping similar data points together, clustering enables efficient data exploration and interpretation. Its versatility makes it indispensable for tackling the challenges posed by the ever-growing volumes of data in modern times. While clustering predominantly occurs in centralized environments, consolidating user data in one location raises notable privacy apprehensions [3]. Centralized storage poses risks of unauthorized access and potential data breaches, compromising user confidentiality. The concentration of sensitive information heightens the vulnerability to security breaches and misuse. Moreover, concerns regarding data ownership and control emerge when data is centralized. As a result, there's a growing call for decentralized or privacy-preserving clustering techniques to mitigate these risks. Because centralized storage keeps all sensitive user data in one place, it is more susceptible to security breaches, unauthorized access, and hacking. As a result, Federated Machine Learning (FML)[4], a distributed machine learning framework designed to safeguard privacy, is experiencing a surge in popularity. FML enables collaborative model training across decentralized devices while ensuring data privacy is maintained. This approach addresses concerns related to centralized data storage and privacy breaches. By allowing data to remain on local devices, FML enhances user privacy while still facilitating model improvement. Its rising adoption reflects a growing emphasis on privacy-preserving techniques in machine learning. With each passing day, the volume of data expands, leading to a proliferation of raw features. This influx presents both opportunities and challenges for data analysis and interpretation. Managing the increasing complexity of raw features becomes essential for effective data processing. The fundamental attributes of the data often lack suitability for clustering, necessitating a more abstract or meaningful representation. Transforming the data into a more interpretable form enhances clustering performance and facilitates the discovery of underlying patterns. This abstraction process involves extracting higher-level features that capture relevant information and reduce noise. Representation learning enables the capture of complex links and structures within the data, thereby enhancing the performance of clustering algorithms. By extracting meaningful features, representation learning provides a more comprehensive understanding of the underlying data patterns. It may also be used for a range of data types, such as picture, text, and numeric data. The clustering algorithm leverages representation learning by organizing the acquired representations into groups. This integration enhances the algorithm's ability to identify patterns and structures within the data. Unsupervised representation learning plays a crucial role in this process, enabling the algorithm to extract meaningful features without labeled data. By utilizing unsupervised techniques, the algorithm autonomously learns relevant representations, contributing to more accurate and efficient clustering outcomes. This approach underscores the importance of leveraging unsupervised learning for effective data analysis and clustering tasks.Two well-liked techniques exist for obtaining unsupervised representations. Employing generative models, such adversarial [5] and augmented One technique is autoencoder[6] , which mimics the

actual data distribution to discover the latent representation. Another approach involves combining contrastive learning with discriminative models[7, 8, 9] . This method aims to learn representations that are both discriminative and semantically meaningful. By leveraging contrastive learning, the model can distinguish between similar and dissimilar instances, enhancing the quality of learned representations. Integrating discriminative models further refines the representations, enabling the algorithm to capture intricate data relationships. Several deep generative model-based clustering algorithms have been proposed for the centralized environment and have been effective in getting relevant cluster information [10, 11]. Applying these methods to obtain high-quality cluster data in a federated setting is a reasonable consideration.In order to do unsupervised clustering in federated datasets, the Iterative Federated Clustering Algorithm (UIFCA) [12] was recently introduced. It combines the FedAvg method with the normalizing flow model. While normalizing flow models excel at capturing intricate data distributions, they often introduce complexity to the latent space. This increased complexity can make interpretation and analysis more challenging, hindering their usability in certain contexts. Despite their effectiveness in capturing subtle distributions, the intricate structure of the latent space may pose difficulties for downstream tasks like clustering. The need for invertibility may provide challenges in producing a latent space that is both smooth and easily interpreted. On the other hand, autoencoders often offer a continuous and smooth representation of latent space. The encoder-decoder design helps the model develop a succinct and meaningful representation of the input data. The use of autoencoder-based unsupervised representation learning in federated environments serves as a driving force behind the effort.

1.2 Contribution towards decentralized clustering

In this research, a proposed parameter aggregation technique is employed alongside an autoencoder for clustering within a federated environment. By leveraging parameter aggregation, the distributed model parameters are effectively combined, enhancing the clustering process. The autoencoder serves to learn meaningful representations of the data, facilitating accurate clustering across decentralized sources. The primary contributions and proposals of this work are, in essence, as follows:

1. For federated systems, A novel autoencoder-based clustering approach is provided. Autoencoders are used to learn condensed representations of data properties locally on each client device, without sending raw data. It is demonstrated how it is used. Since different datasets require different architectures to attain good performance, A different autoencoder architecture is employed for each dataset used in the experiment .

2. FednadamN, a unique model parameter aggregation technique is proposed. By adding the Adam and Nadam optimizer, this method improves on previously developed federated learning techniques with the goal of improving model performance between decentralized nodes. When compared to conventional aggregation algorithms, FednadamN offers potential gains in both efficiency and

effectiveness, making it a promising innovation in federated learning methodologies.

3. Three well-known federated learning strategies—FedAdam, FedAvg, and FedMedian—as well as our suggested FednadamN technique are compared in this study. In the context of federated learning, this comparison provides a baseline evaluation for assessing the efficacy and efficiency of our unique technique.

4. A thorough simulation was run using four benchmark data sets.The datasets consist of image segmentation, protein localization, vowel discrimination, and letter image recognition. The results of the simulation show that our model performs better than the baseline.

To the best of our knowledge, this work is the first to use a customized FednadamN approach for autoencoder-based decentralized clustering on unsupervised data in a heterogeneous federated environment.

1.3 Organization of the Report

This report is formatted as follows: Chapter 2 provided details on this project's background investigation. Methodologies are discussed and quickly reviewed in Chapter 3. The experiment, observations, and result analysis are covered in detail in Chapter 4. Chapter 5 presents the main conclusion of the thesis.

Chapter 2

Literature Review

Federated learning is a cooperative approach to machine learning in which a centralized model is created by coordinated correspondence with several clients. Clients transmit model changes to a central server rather than exchanging local data as in the case of previous approaches. In the process, anonymity is maintained and a variety of data sources may be used to develop the model. Various techniques for unsupervised federated learning have been put forth, each with a distinct approach. These methods seek to eliminate the requirement for labeled data by enabling cooperative model training across distributed devices. Unsupervised federated learning approaches leverage methods like clustering, generative modeling, or self-supervised learning to build strong models while protecting data confidentiality and privacy over decentralized networks. Several unsupervised federated learning techniques with distinct methodologies have been proposed. Fengda Zhang et al.[13] demonstrated Federated Unsupervised Representation Learning (FURL) as a solution to address data privacy concerns, making it a promising technique. FURL enables distributed edge devices to adopt a shared representation paradigm while preserving user privacy. This approach allows devices to collaboratively learn representations without sharing raw data, ensuring privacy compliance. Chen Zhao et al.[14] enhanced this method for application in Internet of Things (IoT) contexts, as detailed in their research. Their adaptation incorporates a dynamic update mechanism tailored to address the challenges posed by non-IID (non-independent and identically distributed) data commonly found in IoT environments. This dynamic update mechanism ensures the model remains adaptable and effective in capturing the evolving data patterns.Bram van Berlo et al.[15] showcased the potential of FURL in human activity detection, as highlighted in their study. By pre-training with unlabeled data collected from a diverse range of users, FURL achieved competitive or even superior performance compared to supervised deep learning approaches. This demonstration underscores the effectiveness of unsupervised representation learning in capturing nuanced human activity patterns. FedOnce [16] is a communication-efficient method developed to address privacy concerns in real-world vertical federated learning scenarios. It integrates unsupervised representation learning with prioritization and cost management strategies to optimize performance. The approach aims to enhance collaboration among vertically partitioned datasets while safeguarding sensitive information. Sungwon Han et al. introduced FedX[17], a novel approach in federated learning. FedX leverages contrastive learning and knowledge distillation techniques to extract unbiased representations from decentralized data sources. By

incorporating these methods, FedX aims to mitigate bias and enhance the quality of learned representations. Mykola Servetnyk et al.[18] propose weight calculation strategies to address the challenge of unbalanced data in federated learning. These strategies aim to mitigate bias and improve the fairness of model training across heterogeneous datasets. By dynamically adjusting weights based on data distribution, the approach ensures that each participant's contribution is appropriately accounted for. E. S. Lubana et al. introduced the Orchestra [19] method, which employs distributed clustering to achieve a globally consistent segmentation of customers' data. Tiandi Ye et al.[20] expand the adaptive risk reduction strategy to the realm of unsupervised personalized federated learning, aiming to tackle the challenge of delivering tailored models for new clients. Yeongwoo Kim et al.[21] present a dynamic clustering method designed to address privacy issues, accommodate shifting environments, and explore various cluster configurations. By considering privacy concerns, the method prioritizes data protection while achieving effective clustering outcomes. The ability to adapt to changing settings enhances its applicability in dynamic data environments.FML was introduced by McGahan et al. [4] in 2016. According to McMahan et al.[4], a global server model and a loose federation of scattered clients help FML resolve model training. The global model parameters are provided by the clients for training. by the global model server utilizing client data that is locally saved. The global model server receives an update to the global model parameter from the client. The global model server aggregates all modifications. and chooses new parameters. This process is carried out again for a specific number of communication cycles needed for the global model to converge. Federated learning clustering has been the subject of several works. Techniques for user grouping in non-IID federated learning are provided by Jianfei Zhang et al. [22] and Lucas Pacheco et al.[23]. Pacheco concentrates on a neural network-based approach, while Zhang introduces the FedLabCluster method. Avishek Ghosh et al. introduce the Iterative Federated Clustering Algorithm (IFCA)[24] for clustered federated learning, demonstrating its efficacy in addressing non-convex problems. IFCA iteratively refines cluster assignments across distributed data sources, fostering collaboration while preserving data privacy. By leveraging iterative optimization techniques, IFCA effectively navigates complex data distributions inherent in non-convex problems. Chengxi Li et al.[25] expand on the concept by introducing Federated Learning with Soft Clustering (FLSC), which takes advantage of overlapping clusters to improve learning performance. The combined findings of these studies help to develop more effective and efficient federated learning clustering algorithms. Several clustering methods have been proposed for use in federated learning. Junshen Su et al.[26] present a cosine similarity-based method that effectively groups clients and improves test accuracy. With a focus on communication efficiency, La-izhong Cui et al.[27] present ClusterGrad, a novel clustering-based quantization technique and a K-means algorithm for compressing gradients. The performance of federated learning is improved by the combined influence of various methods.

2.1 A Systematic Review on Federated Unsupervised Clustering

A comprehensive literature review technique [28] was used in this study. Key purposes of the review research can be found in table 2.1 . The methodology comprised an exhaustive and methodical examination of extant academic literature pertinent to the subject matter being studied. This strategy made it easier to identify, assess, and synthesize relevant literature by using strict inclusion criteria and search tactics. As a result, the study's results and conclusions were well-founded.

2.1.1 Search Strategy

A systematic search in several electronic databases is conducted, including IEEE Xplore, ACM Digital Library, arXiv, and Google Scholar. This comprehensive approach ensured the inclusion of a wide range of scholarly articles, conference papers, preprints, and other relevant literature sources. we have have searched for federated unsupervised learning (eg: ("federated learning" OR "federated machine learning" OR "distributed machine learning") AND ("unsupervised learning" OR "unsupervised machine learning")) The articles were searched using several keywords related to autoencoder or clustering(eg: ("autoencoder" OR "autoencoders" OR "auto-encoder" OR "auto-encoders" OR "neural network") AND ("clustering" OR "cluster analysis" OR "segmentation")).This query searches for articles containing any variation of the term "autoencoder" and any variation of the term "clustering" or "cluster analysis". we have searched using keyword related to generative model and unsupervised representation learning (eg : ("unsupervised representation learning" OR "unsupervised feature learning" OR "unsupervised feature extraction") AND ("generative model" OR "generative models" OR "generative neural network" OR "generative adversarial network" OR "variational autoencoder"))

2.1.2 Inclusion and Exclusion Criteria

Peer-reviewed journal articles is included, conference proceedings, and preprints that focused on the development, analysis, or application of federated decentralized clustering techniques using autoencoders. Studies that did not explicitly address federated or decentralized settings or did not utilize autoencoders were excluded. In selecting an article, adherence to the following criteria was paramount: (a) The essay must be crafted in English, facilitating comprehension and accessibility across diverse readerships; (b) The article should demonstrate the incorporation of federated learning methodologies, showcasing collaborative machine learning techniques; (c) The chosen paper should illustrate the utilization of either decentralized clustering or federated unsupervised representation learning methods, indicating innovative approaches to data analysis and model training within distributed frameworks; (d) Additionally, the paper should utilize generative models for unsupervised clustering, highlighting advancements in clustering techniques. In addition to the inclusion criteria, the article selection process involved the application of the following exclusion criteria : (a) Any duplicate articles identified through searches across multiple academic databases were excluded to maintain uniqueness and prevent redundancy in the review process; (b) Articles not directly pertinent to our research objectives were

excluded, ensuring focus and relevance to the intended scope of inquiry; (c) Non-English-language articles were excluded to maintain consistency with the language criterion and ensure accessibility and comprehension for the intended audience; and (d) Previous iterations of articles published on the same dataset and exploring identical objectives were excluded to prioritize fresh perspectives and novel insights in the selected literature.

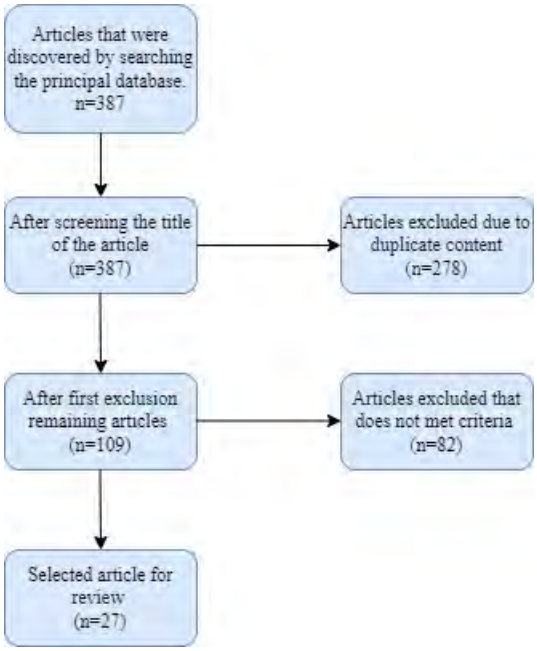


Figure 2.1: Article inclusion and exclusion process flowchart.

2.1.3 Study Selection

A comprehensive search across specified databases yielded a total of 387 articles for consideration. The article selection process was meticulously executed in adherence to predefined exclusion-inclusion criteria, as illustrated in various stages through the Prisma flowchart presented in Figure (2.1). The Prisma flowchart delineates the sequential stages of the selection procedure, from initial database search to final inclusion of relevant articles. At each stage, articles were meticulously screened based on predetermined criteria to ensure rigorous selection and maintain methodological rigor. In the initial phase of the research selection process, the pool of 384 publications underwent scrutiny to eliminate duplicate papers. Through this rigorous screening, a total of 275 articles were identified and removed as duplicates, thereby streamlining the dataset and ensuring the integrity and uniqueness of the remaining literature.This meticulous culling of duplicate papers underscores the commitment to methodological rigor and quality assurance in the research selection process. Following the removal of duplicate papers, a total of 88 items were further eliminated in the previous round based on strict adherence to our predefined inclusion and exclusion criteria. This meticulous screening process ensures that only articles closely aligned with our research objectives and methodological standards are retained for subsequent analysis.

2.1.4 Data Extraction

A meticulous data extraction process was undertaken to delve into various aspects of federated model aggregation strategies, algorithms, machine learning (ML) techniques, and unsupervised clustering within federated settings. This extraction effort involved mining pertinent information from selected literature to elucidate the diverse approaches and methodologies employed in federated learning paradigms. Specifically, the focus was on understanding the intricacies of federated model aggregation, encompassing strategies for aggregating distributed model updates across heterogeneous devices and also on understanding the use of generative models in federated settings. The selected publications underwent a meticulous and comprehensive review process to gather and extract pertinent research material, laying the foundation for the subsequent literature review. This review involved a thorough examination of each chosen publication, scrutinizing the content to extract valuable insights, methodologies, findings, and discussions relevant to the research objectives. Emphasis was placed on identifying key concepts, methodologies, and empirical evidence related to the topic of interest, ensuring a comprehensive understanding of the subject matter. Two senior researchers oversaw and kept an eye on the whole research procedure.researchers with proficiency in machine learning, artificial intelligence, and health informatics to guarantee the accuracy and caliber of this review article. In the present study, a meticulous data extraction process was conducted to gather a diverse array of information from each selected article. Key data points were meticulously extracted, including:Identifying the nature of each article, such as research papers, reviews, or theoretical analyses, the date of publication to contextualize the research within its temporal framework, Understanding the specific objectives and goals outlined by the authors, Detailing the methodologies, algorithms, and techniques utilized in the study and Identifying the datasets employed for experimentation or analysis. Ultimately, the retrieved data were combined and examined to provide an overview of the current literature and suggest possible directions for further investigation.
4.2

Table 2.1: Key Purposes Of The Reviewed Studies

Purposes	Brief description	Ref	Frequency
Federated Unsupervised Learning	In order to address the problems of data distribution shift and representation misalignment in distributed edge devices, the research presented FedCA, a federated unsupervised representation learning technique.	[13]	8

	proposed a completely new federated unsupervised learning method with a dynamic updating mechanism to address non-IID decentralized data issues in Internet of Things picture classification.	[14]	
	introduced federated unsupervised representation learning, which uses unsupervised representation learning in a federated environment to pre-train deep neural networks with unlabeled input. Experiments on human activity detection demonstrate competitive or better performance than supervised deep learning, and thus enables discriminative feature extraction.	[15]	
	presented FedOnce, a vertical federated learning algorithm that is communication-efficient and achieves competitive performance with lower communication costs than state-of-the-art approaches by utilizing unsupervised learning representations and privacy-preserving mechanisms.	[16]	
	proposed FedX, an unsupervised federated learning system that improves performance across five unsupervised algorithms without needing data feature sharing between decentralized and diverse local data sources. FedX uses two-sided knowledge distillation with contrastive learning.	[17]	
	In order to solve accuracy issues in nonuniformly dispersed data, this research proposed an unsupervised learning algorithm within the federated learning framework. Dual Averaging (DA) with two weight calculation techniques, fixed size bin and self-organizing maps (SOM), are used.	[18]	
	In order to guarantee globally consistent data partitioning across clients, the study suggested Orchestra, a unique unsupervised federated learning approach that orchestrates distributed clustering activities.	[19]	
	The paper introduced FedTTA, an unsupervised personalized federated learning method for accommodating new clients joining trained and deployed models, enhanced with adaptive risk minimization, proxy regularization, early-stopping adaptation, and knowledge distillation	[20]	

Federated supervised learning	Introduced a practical federated learning method based on iterative model averaging named FederatedAveraging alogorithm, validated through extensive empirical evaluations across various model architectures and datasets.	[4]	2
	The paper introduced federated versions of adaptive optimizers (Adagrad, Adam, and Yogi) to address convergence challenges in Federated Learning (FL) due to data heterogeneity.	[29]	
Federated Data Clustering	In order to overcome statistical variability between clients, the article presented UIFCA, a federated clustering technique that makes use of generative models within the IFCA framework.	[12]	2
	By utilizing cluster division, cluster calibration, and generative adversarial network-based clustering, the study presented a three-phased data clustering algorithm for federated learning in wireless network management, overcoming the drawbacks of conventional clustering techniques.	[21]	
Federated Client Clustering	FedLabCluster, a unique clustered federated learning technique that clusters clients based on sample labels to handle Non-IID data, was introduced in the study.	[22]	6
	Introduced a Neural Network-based Federated user Clustering mechanism to address the requirement of Independent Identically Distributed (IID) data in Federated Learning (FL).	[23]	
	The paper introduced IFCA, an Iterative Federated Clustering Algorithm for clustered federated learning.	[24]	
	In order to handle non-IID data, the paper presented FLSC, a Federated Learning algorithm with soft clustering that combines the advantages of IFCA and soft clustering.	[25]	
	In order to handle data heterogeneity in Federated Learning (FL), the research presented a cosine similarity-based clustering approach for Clustered Federated Learning (CFL).	[26]	
	In order to dramatically lower transmission volume, the study introduced ClusterGrad, a gradient compression approach for Federated Learning (FL), which makes use of gradient clustering.	[27]	

Generative Model Based Clustering	Introduced ClusterGAN, a unique GAN-based clustering method that combines an inverse network that was simultaneously trained using a clustering-specific loss function with continuous and one-hot encoded latent variables.	[10]	2
	Presented DeepCluster, an unsupervised visual feature learning technique that uses iterative k-means clustering to simultaneously learn neural network parameters and cluster assignments.	[11]	
Unsupervised Representation Learning	Demonstrated state-of-the-art performance across many benchmarks and workloads for unsupervised semantic feature learning using ConvNets trained to detect 2D rotations applied to input images, greatly narrowing the gap with supervised feature learning techniques.	[5]	5
	Introduced deep convolutional generative adversarial networks (DCGANs), demonstrating their generalizability as picture representations across multiple tasks and datasets, and highlighting its potential for unsupervised learning by learning hierarchical representations.	[6]	
	Presented RUC, a novel model based on robust learning that uses pseudo-labels from other image clustering models to address overconfidence and incorrect predictions. It shows better robustness and calibration across a variety of datasets and can be used as a flexible add-on module to improve clustering performance.	[7]	
	A novel self-supervised learning technique called Super-AND is presented. It builds upon the memory-based pretraining method AND model by including unique losses to improve anchor selection and adversarial training to improve embedding learning.	[8]	

	Aimed to develop distributed statistical learning algorithms that are provably robust against Byzantine failures.	[9]	
Privacy Issues	Forecasts 50 billion IoT devices by 2030, noting benefits but also challenges of unique security threats, reviewing existing literature, discussing IoT threat detection research, and outlining future research directions.	[1]	2
	Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach.	[3]	

This research is primarily concerned with data clustering in federated environments. Since autoencoders provide a smooth and continuous latent space representation, An autoencoder-based clustering technique have been employed for the representation of federated environments.To the best of our knowledge, this study is the first to try decentralized clustering using autoencoders.Additionally, the FednadamN model aggregation technique has been put forth.The proposed FednadamN, a combination of Adam and Nadam optimization, takes into account the unique weights of each client to make sure that the global model weights remain closer to the optimal solution during training.

Chapter 3

Methodology

In a centralized system, the goal of clustering is to create distinct and meaningful clusters that aid in better data organization, interpretation, and analysis. Data clustering issues are common in centralized environments, when data is kept on a single computer or client. These issues arise due to the challenges of handling large volumes of data efficiently within a centralized infrastructure. Additionally, the computational complexity of clustering algorithms can strain the resources of a single machine or client. Communication overhead becomes a critical concern when transferring data between the central server and client devices, impacting the performance of clustering algorithms. In a distributed setting with n client computers and one central server, a common data clustering problem is examined.. This scenario introduces complexities in coordinating the clustering process across multiple devices. The challenge lies in efficiently aggregating local clustering results from individual clients to form a cohesive global clustering solution. Communication overhead becomes a significant concern in transmitting data between the central server and numerous client computers. The goal is to determine which of the total D data samples are thought to spontaneously partition into K distinct clusters, $D_1 \ldots\ldots D_k$. A random shuffling strategy was utilized to disperse the data samples among the n clients, ensuring the preservation of heterogeneity. Each client dataset is denoted as D_i, representing a distinct subset of the overall data. This approach helps maintain diversity within individual client datasets, contributing to more representative clustering results. Using the flwr Python library, A federated system with nn client devices is replicated . and a solitary central server. This setup enables us to simulate distributed data processing and coordination, mirroring real-world scenarios encountered in federated learning and decentralized data analysis. The flwr library provides essential tools for orchestrating communication, synchronization, and aggregation between the central server and the client devices, facilitating the development and evaluation of federated learning algorithms and distributed data clustering techniques.For each client and the server, autoencoder models are used to predict cluster information.. Autoencoders are neural network architectures capable of learning efficient representations of data by reconstructing inputs from compressed representations. By utilizing autoencoders in both the server and client models, In order to facilitate accurate cluster information estimate, the goal is to extract and encode the data's underlying structure. Using a connectivity technique, the client computers establish communication with the central server. In this setup, the model residing on the server is termed "global,"

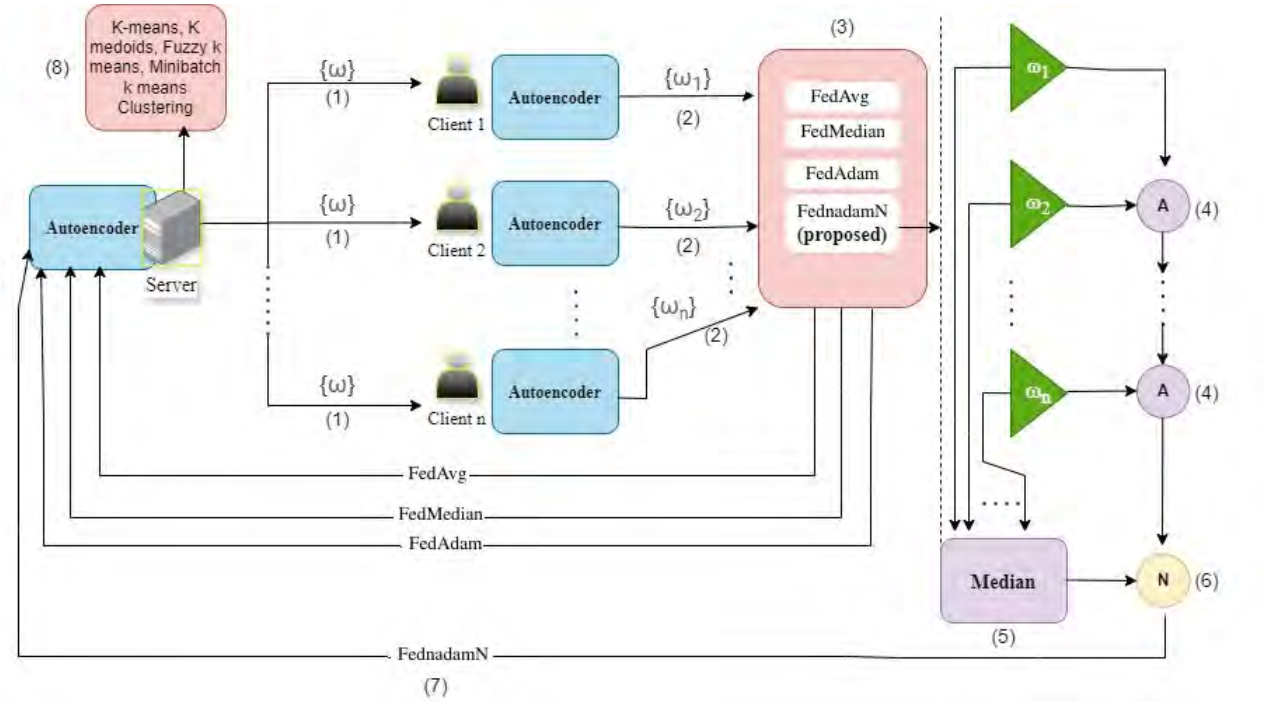


Figure 3.1: Overview of our methodology.(1) Central server broadcast global weight to n clients.(2) Each local client update local autoencoder and send the updated weight to the strategies. (3) Four different strategies are used. (4) Each clients updated weights are updated with the next client upto n clients and combine all the weights. (5) Median weight is calculated considering all clients weights. (6) Median weight is updated with the combined weight. (7) Indiviadual weights from each strategy are send to central server. (8) Server apply clustering algorithms on the output of the global autoencoders latent layer.

whereas the model residing on each client is referred to as "local." This distinction between global and local models allows for decentralized processing and collaborative learning, where the central server coordinates the aggregation of information from individual clients to update the global model iteratively. The core concept revolves around training the global model collaboratively without centralizing the data. By distributing the training process across multiple client devices, each holding its local data, the global model is updated iteratively through communication with these clients.The construction of the global model is initiated by generating random parameters ω. These randomly initialized parameters serve as the starting point for the global model's development. The local models will undergo frequent modifications utilizing local data. Each client device refines its respective local model iteratively by training on its own data subset. These frequent updates ensure that the local models accurately capture patterns and nuances present within the client's data without necessitating data transmission to the central server. That helps the global model undergo refinement and optimization to learn from the distributed data effectively. Federated setting is replicated for R communication rounds, where R

denotes the number of iterations or communication cycles between the central server and the client devices. During each communication round t, updates are exchanged between the global model on the server and the local models on the client devices . Every local autoencoder modifies its local parameter ω using the global parameters during a communication round. In every training epoch, each client computes the Kullback-Leibler divergence loss function using its local dataset D_i. This loss function, denoted as $L_i(\omega)$ (3.1), quantifies the discrepancy between the probability distributions of the reconstructed data and the original input data. By evaluating $L_i(\omega)$ for each client's local dataset, The performance of the autoencoder model in capturing the underlying structure of the data and minimizing information loss during reconstruction is assessed.

$$L_i(\omega) = \frac{1}{|D_i|} \sum_{(x,y)\in D_i} \text{Loss}(f_\omega(x), y) \tag{3.1}$$

The model prediction for input x is denoted by $f_\omega(x)$. For every autoencoder, The Kullback-Leibler divergence loss function is employeed. Subsequently, for every client, the gradient of the local loss $\nabla_\omega L_i(\omega)$ is computed with respect to the model parameters ω_i^t. This gradient calculation quantifies the direction and magnitude of the change required in the model parameters to minimize the loss function. Based on the computed gradient, each local model utilizes an Adam optimizer to update the local model parameters, denoted as ω_i^t. This optimization algorithm adjusts the parameters in the direction that minimizes the local loss function, thereby enhancing the model's performance on the specific dataset. By employing Adam optimization, which combines adaptive learning rates and momentum, the local models iteratively refine their parameters to converge towards optimal solutions while accounting for variations in the loss landscape.

$$\omega_i^{(t+1)} = \text{Adam}(\omega_i^t, \nabla_\omega L_i(\omega)) \tag{3.2}$$

Equation (3.2) uses the updated local model parameters $\omega_i^{(t+1)}$ to reflect how the model has been adjusted to fit the local dataset D_i. After a specified number of epochs of local training, each client transmits the updated local model parameters to the central server. This communication step enables the central server to aggregate the parameter updates from all client devices, facilitating the synchronization and consolidation of knowledge across the distributed system. By periodically exchanging model parameters, the central server maintains an up-to-date global model that reflects the collective insights gleaned from the diverse local datasets. In the server, combine the local model updates that have been received to derive the new global model parameters $\omega^{(t+1)}$ in (3.3). The local update may be further explained by using Algorithm 1.

$$\omega^{(t+1)} = \text{Aggregate}(\omega_1^t, \omega_2^t, \ldots, \omega_n^t) \tag{3.3}$$

Four strategies have been used for parameter aggregation. Among the various strategies employed, FednadamN stands out as a proposed approach tailored to our specific needs and objectives. This strategy, named FednadamN, is uniquely crafted to address the challenges and requirements of our project or scenario. It offers a specialized solution designed to optimize performance, enhance efficiency, or achieve particular goals within the given context. Alongside the customized strategy of FednadamN, FedAvg, FedMedian, and FedAdam are incorporated as baseline methods for parameter aggregation. These established approaches provide benchmarks

against which the performance of FednadamN can be evaluated and compared. By leveraging a combination of customized and standardized methods, A comprehensive assessment is ensured for different aggregation techniques, thereby facilitating informed decision-making and optimization of our federated learning framework. These baseline tactics act as the fundamental frameworks that measure the efficacy and performance of our tailored strategy. FednadamN is comparable and evaluable. This decision is made to employ the global parameter in the global model after conducting R communication rounds. This decision marks a turning point in our iterative process, as the ideas and collective knowledge gained across R communication cycles are combined to produce the global model. including the worldwide parameter. We fully supplied the global model with data, enabling a thorough comprehension of the subtleties and intricacies of the dataset. Thus, the latent layer's output out of the global autoencoder model is taken. This deliberate approach allows to retrieve the key underlying characteristics and patterns from the data that the latent layer autoencoder is looking for.

To thoroughly evaluate our approach, The central server is equipped with a wide range of clustering methods. K-means, fuzzy k-means, minibatch k-means, and k-medoids are a few of them. This comprehensive implementation allows us to assess the effectiveness and suitability of different clustering techniques for our specific dataset and objectives. By exploring a wide variety of algorithms, It allows to learn more about performance, strengths, and limitations, thus informing our decision-making process and guiding further optimizations in our clustering strategy. Each algorithm brings its own unique strengths and characteristics to the table, enriching our exploration of the dataset and enabling us to extract valuable insights from diverse perspectives. By leveraging a variety of clustering techniques, within the data hidden patterns, relationships, and structures can be uncovered , providing a holistic understanding of its underlying properties. Our clustering methods utilized the output of the latent layer as an input. This approach leverages the representations learned by the autoencoder model in the latent space, which captures essential features and patterns within the data. By employing the latent layer's output for clustering, this work get benefit from the enhanced abstraction and dimensionality reduction achieved during the autoencoder's training process. Our goal is to more compactly and meaningfully describe the underlying structures and patterns contained in the data by utilizing the encoded representations that our model's latent layer has learnt. A summary of our research is provided in Figure (3.1).

Algorithm 1 Client Update

Input: Client data C, initial parameters ω
Output: Updated parameters ω
for $i = 1$ to E **do**
    $\omega \leftarrow \text{Adam}(\omega, \nabla L(\omega; D))$
end for
return ω

3.1 Algorithms Behind Clustering

In our experiment, Four clustering algorithms alongside autoencoder-based models to achieve decentralized clustering have been emploeed. This multifaceted approach allowed us to explore various methodologies for clustering while leveraging the capabilities of autoencoder models to extract meaningful features from the data. By combining different clustering algorithms with autoencoder-based models, the objective is to improve the decentralized clustering framework's resilience and efficacy.

1. K-means: A well-liked clustering technique called kmeans divides a dataset into k clusters, each of which has a single data point that belongs to the cluster with the closest mean. Selecting the number of clusters, k, is the first step. Next, initialize the centroids of each of the k clusters at random. Subsequently, each data point is assigned to the cluster with the closest centroid (often determined by Euclidean distance) . The assignment step is often formulated as follows in (3.8)

$$argmin_j ||\mathbf{x}_i - \mathbf{C}_{n_j}||^2 \tag{3.4}$$

   Where x_i is a data point, Cn_j is the centroid of cluster j and $||.||$ represents euclidian distance.Then It takes the mean of all the data points allocated to each cluster to recalculate the centroids of the clusters. The update step is often formulated as follows in (3.9)

$$\mathbf{C}_{n_j} = \frac{1}{|S_j|} \sum_{i \in S_j} \mathbf{x}_i \tag{3.5}$$

   here S_j is the set of data points assigned to cluster j. Up to convergence, repeat the assignment and recalculation. Convergence happens after a certain number of repetitions, or when the centroids no longer exhibit substantial variation

2. K-medoids: A variant of the k-means algorithm, the kmedoids method use the medoid as the representation of a cluster rather than the mean (average). The data point inside a cluster that minimizes the total distances to all other points in the same cluster is known as the medoid. The initial step is to choose k , the number of clusters. The centroids of each of the k clusters should then be randomly initialized. Each data point is assigned to the cluster whose medoid (often determined using a distance metric like Euclidean distance) has the least dissimilarity. The assignment step is often formulated as follows in (3.10)

$$\text{argmin}_j \sum_{i \in S_j} d(\mathbf{x}_i, \mathbf{m}_j) \tag{3.6}$$

   Here x_i is data point , m_j is the medoid of cluster and $d(.,.)$is a distance metric. Subsequently The data point that minimizes the total distances to other points in the same cluster as the new medoid should be chosen for each cluster The update step is often formulated as follows in (3.11)

$$\mathbf{m}_j = \text{argmin}_{\mathbf{x} \in S_j} \sum_{i \in S_j} d(\mathbf{x}, \mathbf{x}_i) \tag{3.7}$$

Here m_j is the medoids of cluster jj andS_j Sj is the of data points assigned to cluster J.Until convergence, repeat the assignment and updating steps. Convergence happens after a certain number of repetitions, or when the medoids no longer exhibit significant changes.

3.2 Autoencoder

In the realm of artificial intelligence and machine learning, autoencoders stand as a fundamental and versatile tool, wielding the power to unravel the intricate tapestry of data. With their capacity to learn compact representations of input data and reconstruct it with remarkable fidelity, autoencoders have found widespread application across diverse domains, from image processing to natural language understanding. This page delves into the essence of autoencoders, exploring their architecture, training process, and manifold applications. At the heart of an autoencoder lies a symphony of neural network layers meticulously orchestrated to compress and decompress data. The architecture typically comprises two primary components: an encoder and a decoder. The encoder, akin to a sculptor, transforms the high-dimensional input data into a lower-dimensional representation, capturing its essence in a latent space. Conversely, the decoder acts as an artist, skillfully reconstructing the original data from the latent representation. Together, these components form a seamless pipeline, orchestrating the intricate dance of information compression and decompression. The journey of an autoencoder begins with the quest to minimize the chasm between the original input and its reconstructed counterpart. Through an iterative process of training, the autoencoder traverses the landscape of data, fine-tuning its parameters to minimize the reconstruction error. This journey unfolds in the realm of unsupervised learning, where the autoencoder navigates the labyrinth of data without the guiding hand of labeled examples. Armed with optimization algorithms such as stochastic gradient descent, the autoencoder embarks on a quest for efficiency, striving to distill the essence of the data into a compact and meaningful representation. primarily Its objective is to acquire efficient data representations, typically for dimensionality reduction or feature learning. The encoder of the autoencoder is its initial part. The encoder component of an autoencoder translates the input data to a representation that is lower dimensional. This process involves capturing essential features and patterns while reducing the dimensionality of the data. By compressing the input into a compact latent space, the encoder learns to distill the most salient information for subsequent reconstruction. This lower-dimensional representation serves as a compressed encoding of the input data, facilitating efficient storage and computation in downstream tasks. The encoder function may be expressed mathematically as $h = f(x)$, where x represents the input data and h represents the encoded representation. Then comes the latent space, where the representation of the supplied data is condensed into a lower-dimensional form. This latent space serves as a compact encoding of the input data, capturing essential features and patterns while reducing dimensionality. Within this compressed representation, the autoencoder encapsulates the essence of the input data, facilitating efficient storage and analysis. The latent space acts as a bottleneck in the autoencoder architecture, enforcing a compressed representation that encourages the extraction of salient information. In the lower-dimensional space of the autoencoder's latent representation, each point corresponds to a potential encod-

ing of the input data. These points encapsulate the essential features and patterns extracted by the autoencoder during the encoding process. By mapping the input data to this compressed space, the autoencoder captures the underlying structure of the data in a more concise form. This representation facilitates efficient storage and analysis, enabling downstream tasks such as reconstruction and clustering.The final component of the autoencoder is the decoder, tasked with reconstructing the original input data from the encoded representation obtained from the latent space. Employing the learned features from the latent space, the decoder aims to faithfully recreate the input data. Through a process of decoding, the autoencoder endeavors to minimize the reconstruction error, striving to produce an output that closely resembles the original input. This reconstruction process completes the cycle of the autoencoder, enabling it to capture and recreate the essential characteristics of the input data. The goal is to produce an output that closely matches the input. The decoder function may be expressed mathematically as $r = g(h)$, where r represents the reconstructed output and h represents the encoded representation. Reconstruction error minimization stands as the primary training objective of the autoencoder. By striving to minimize the discrepancy between the original input data and its reconstructed counterpart, the autoencoder learns to capture the essential features and patterns within the data. This iterative optimization process fine-tunes the autoencoder's parameters, enhancing its ability to faithfully recreate the input data. Through the pursuit of reconstruction error minimization, the autoencoder endeavors to distill the essence of the data into a compact and meaningful representation.

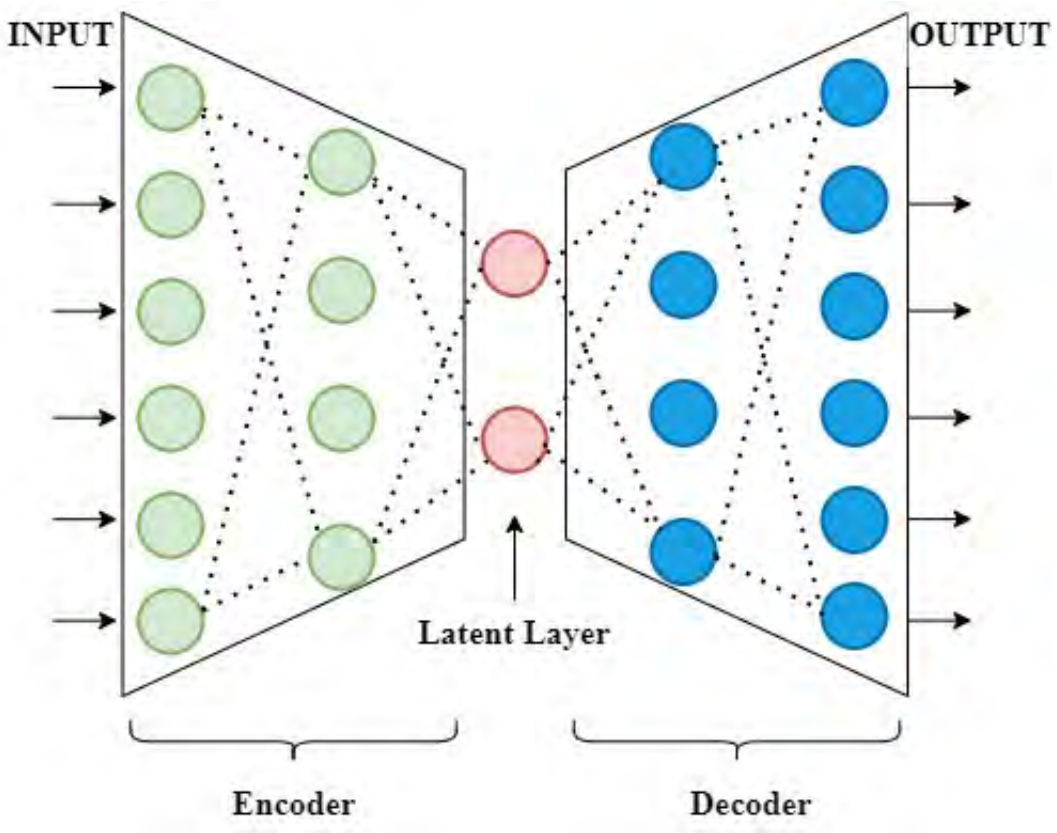


Figure 3.2: Sample Auto encoder

Reconstruction error minimization stands as the primary training objective of the autoencoder. By striving to minimize the discrepancy between the original input data and its reconstructed counterpart, the autoencoder learns to capture the essential features and patterns within the data. This iterative optimization process fine-tunes the autoencoder's parameters, enhancing its ability to faithfully recreate the input data. Through the pursuit of reconstruction error minimization, the autoencoder endeavors to distill the essence of the data into a compact and meaningful representation. The discrepancy between the input data and the reconstructed output is quantified by the loss function. This loss function evaluates the difference between the original input and its reconstructed counterpart, providing a metric for the reconstruction accuracy of the autoencoder. By minimizing the loss function during training, the autoencoder learns to generate reconstructions that closely match the original input data. This iterative optimization process enables the autoencoder to refine its parameters and improve its reconstruction capabilities over time. Depending on the nature of the input data, autoencoders commonly employ Mean Squared Error (MSE) or Binary Crossentropy as loss functions. MSE is suitable for continuous data, measuring the average squared difference between the original input and the reconstructed output. On the other hand, Binary Crossentropy is preferred for binary data, quantifying the discrepancy between the binary input and its reconstruction. The choice of loss function depends on the specific characteristics of the input data and the objectives of the autoencoder training. Typically, the loss function is specified as $L(x, g(f(x)))$ in where x represents the input data, f denotes the encoder, and g denotes the decoder. During training, the autoencoder adjusts the settings of both the encoder and decoder to minimize the reconstruction error. By iteratively fine-tuning these parameters, the autoencoder learns to extract and represent the most salient features of the input data. This optimization process enables the autoencoder to capture essential patterns and structures while reducing redundancy in the latent representation. Through continuous refinement, the autoencoder gradually improves its ability to reconstruct the input data with greater fidelity. An example of an autoencoder is shown in figure (3.2). The phases of decentralized clustering using autoencoder are represented by algorithm 2.

Algorithm 2 Autoencoder based decentralized clustering

Initialize server autoencoder with ω^0, parameters β_1, β_2, δ
for each round $t = 1, 2, \ldots$ **do**
    $S_t \leftarrow$ random set of m clients
    Initialize client autoencoder with ω^t
    for each client $C \in S_t$ in parallel **do**
        $\omega_C^{(t+1)} \leftarrow \text{ClientUpdate}(C, \omega_C^t)$
    end for
    $\omega^{(t+1)} \leftarrow \text{Strategy}(\omega_1^t, \omega_2^t, \omega_3^t, \ldots, \omega_C^t)$
    Update server autoencoder with $\omega^{(t+1)}$
end for
Calculate latent representation, $h = f(x)$, with input data x
Apply clustering algorithms on h

3.3 Strategy Behind Model Aggregation

In this section, It is started by presenting a straightforward extension of the existing parameter aggregation technique, which serves as our baseline approach. This initial expansion provides a foundational framework upon which advanced strategies can be build and refined more . By establishing a baseline, It is aimed to assess the effectiveness of our proposed modifications and innovations in comparison to this foundational technique. This baseline serves as a reference point for evaluating the performance and efficacy of alternative parameter aggregation methods in our experimentation. Then, we delve into the specifics of our proposed strategy, outlining its key components and operational mechanisms. This section offers a comprehensive overview of our innovative approach, detailing how it differs from the baseline technique and the rationale behind its design. By elucidating the intricacies of our proposed strategy, It is aimed to provide clarity and insight into its potential benefits and implications for parameter aggregation in our context.

3.3.1 FedAvg

FedAvg [4] technique A new global model is created by averaging the model updates from each client that takes part in the training process.Devices with larger datasets typically have a greater influence on the global model during the aggregation stage, as this process is often weighted based on each device's data volume. This weighting scheme ensures that devices with more extensive datasets contribute proportionally more information to the global model's updates. By accounting for variations in data volume across devices, Authors maintain a balanced and representative aggregation process, fostering collaboration and maximizing the utility of available data resources. After receiving the updated models from each client device, the server calculates a weighted average of these models. This weighted averaging process ensures that contributions from each client are appropriately accounted for, with devices carrying more significant data volumes exerting greater influence on the global model. By aggregating the models in this manner, the server synthesizes a comprehensive representation of the collective knowledge gleaned from the distributed data sources. The weight for every client may be expressed as $\frac{n_i}{n}$ in where n is the total number of data samples and n_i is the number of local data samples for each customer in ith. The formula for updating the global model is represented as follows in (3.8).

$$\omega^{(t+1)} = \sum_{i \in C} \frac{n_i}{n} \omega_i^{(t+1)} \tag{3.8}$$

Weighted averaging is commonly employed to achieve this, where models from clients demonstrating superior performance are assigned higher weights. This ensures that contributions from clients with more accurate or relevant data are given greater consideration during aggregation. By incorporating performance metrics into the weighting scheme, such as accuracy or loss reduction, the influence of each client's model on the global update can be adjusted dynamically. This adaptive approach promotes collaborative learning, allowing the global model to benefit from the expertise of individual clients while mitigating the impact of noisy or less informative

data sources.

3.3.2 FedMedian

[9] FedMedian As an alternative to FedAvg [4], aggregation uses the median rather than the mean to aggregate local models across decentralized devices. In contrast to calculating the average, the objective is to obtain the median of the model parameters. This alternative aggregation method aims to mitigate the influence of outliers or skewed data distributions, providing a more robust estimate of the global model. By prioritizing the median, the aggregation process becomes less sensitive to extreme values, enhancing stability and resilience in the face of variability across client devices. This approach is particularly beneficial in scenarios where data distributions are non-normal or prone to outliers, ensuring a more representative aggregation outcome. Through the utilization of the median, authors strive to foster greater consistency and reliability in the global model's parameters. The formula for updating the global model is represented as follows in (3.9).

$$\omega^{(t+1)} = \sum_{i \in C} \frac{n_i}{n} \omega_i^{(t+1)} \tag{3.9}$$

Here, the median is computed individually for each parameter of the model, resulting in a parameter-wise aggregation approach. This parameter-wise computation ensures that each element of the model's parameters undergoes median calculation independently. By treating each parameter separately, the aggregation process maintains granularity and preserves the unique characteristics of the model's architecture. This allows for nuanced adjustments and fine-grained aggregation, enhancing the fidelity and accuracy of the resulting global model. Calculating the median for each parameter individually offers a concise summary of the distribution of values, revealing insights into their unique characteristics. This method allow to assess the central tendency of each parameter independently, untethered from the broader context of the entire model. By focusing on individual parameters, this strategy gain a granular understanding of distributional properties and variability, enabling targeted adjustments and optimizations. This parameter-wise analysis enhances our ability to identify outliers or anomalies specific to certain parameters, facilitating robust model interpretation and refinement.

3.3.3 FedAdam

FedAdam [29] technique begins by initializing the first and second moment estimates m and v to zero for each parameter before averaging the model updates from each client that participates in training.Every communication round t, ∇^t is calculated by taking the difference between the weights that are averaged and the weights that are now in the global model. Using exponential decay, it modifies the first moment estimate m^t and the second moment estimate ∇^t at each communication round t. It is possible to represent ∇^t and m^t as in (3.10) and (3.11).

$$m^t = \beta_1 \cdot m^{(t-1)} + (1 - \beta_1) \cdot \nabla^t \tag{3.10}$$

$$v^t = \beta_2 \cdot v^{(t-1)} + (1 - \beta_2) \cdot (\nabla^t)^2 \tag{3.11}$$

After each round of local training, the global model parameters are updated using the Adam update rule. This adaptive optimization algorithm adjusts the model parameters based on the gradients computed from the local training process. By employing the Adam update rule, the global model undergoes iterative refinement, dynamically adapting to the evolving characteristics of the data. This facilitates efficient convergence towards optimal solutions while mitigating the impact of noisy or sparse gradients. Through the application of the Adam update rule, the global model harnesses the collective insights gleaned from distributed client devices, enhancing its performance and generalization capabilities. it can be viewed in (3.12).

$$\omega^{(t+1)} = \omega^t + \frac{\eta}{\sqrt{v^t} + \delta} \cdot m^t \tag{3.12}$$

In this case, the decay rates are δ, β_1, β_2, and η. After a t number of communication cycles, the global model update process is repeated until the global model converges.

3.3.4 Proposed FednadamN

In our FednadamN (Proposed) strategy, instead of computing the average or mean of the local weights, each client's weights is treated as individually during the global model weight updation process. This approach ensures that each client's contribution is accounted for independently, without being influenced by the aggregate behavior of other clients. By considering all client weights individually, the unique characteristics and contributions of each device is preserved to the global model. This personalized approach to weight updation allows for fine-grained adjustments and optimization, tailored to the specific characteristics of each client's data. Through the FednadamN strategy, it is aimed to enhance the fairness, efficiency, and effectiveness of parameter aggregation in federated learning scenarios. Algorithm 3 can further clarify .

The first step in this procedure is to set the starting values of the first and second moment estimates for each parameter, m, nm, and v, nv. Following the completion of all local client updates, it initializes ω^{temp} with the weight update of the first client, ω_1^t, and ω^{med} with the weight update median of all clients, $\omega_{S_t}^t$. The weight update of each updated client's model, ∇^t, is calculated using the difference between each local model weights and the ω^{temp}. Using exponential decay, it updates each client's first moment estimate (m^t and second moment estimate (v^t. One by one, it modifies the temporary weight variable ω^{temp} for every client. ω^{temp} The update formula is comparable to the Adam optimization, which has the formula (3.13).

$$\omega^{\text{temp}} = \omega^{\text{temp}} + \frac{\eta}{\sqrt{v^t} + \delta} \cdot m^t \tag{3.13}$$

The weight update formula for Nadam optimization extends the update rule employed in the Adam optimizer. Nadam combines elements of Adam with Nesterov momentum to enhance convergence and stability during optimization. This hybrid

Algorithm 3 Proposed FednadamN

Server executes
Initialize ω^0, parameters β_1, β_2, δ
for each round $t = 1, 2, \ldots$ **do**
    $S_t \leftarrow$ random set of m clients
    for each client $C \in S_t$ in parallel **do**
        $\omega_C^t \leftarrow \text{ClientUpdate}(C, \omega_C^{t-1})$
    end for
    $\omega^{\text{temp}} \leftarrow \omega_1^t$
    $\omega^{\text{med}} \leftarrow \text{median}(\omega_{S_t}^t)$
    for each client $C \in S_t$ and $C \neq 1$ **do**
        $\nabla^t \leftarrow [\omega_C^t - \omega^{\text{temp}}]$
        $m^t \leftarrow \beta_1 \cdot m^{t-1} + (1 - \beta_1) \cdot \nabla^t$
        $v^t \leftarrow \beta_2 \cdot v^{t-1} + (1 - \beta_2) \cdot (\nabla^t)^2$
        $\omega^{\text{temp}} \leftarrow \omega^{\text{temp}} + \frac{\eta}{\sqrt{v^t}+\delta} \cdot m^t$
    end for
    $n\nabla^t \leftarrow [\omega^{\text{med}} - \omega^{\text{temp}}]$
    $nm^t \leftarrow \beta_1 \cdot nm^{t-1} + (1 - \beta_1) \cdot n\nabla^t$
    $nv^t \leftarrow \beta_2 \cdot nv^{t-1} + (1 - \beta_2) \cdot (n\nabla^t)^2$
    $\omega^{(t+1)} \leftarrow \omega^{\text{med}} + \frac{\beta_1 \cdot nm^t + (1-\beta_1) \cdot n\nabla^t}{\sqrt{v^t}+\delta} \cdot \eta$
end for

approach incorporates the benefits of both algorithms, offering improved performance and robustness. The Nadam optimizer adjusts the learning rate dynamically, allowing for adaptive optimization in different regions of the parameter space. By leveraging Nesterov momentum, Nadam accelerates convergence towards optimal solutions while reducing oscillations. This makes Nadam particularly effective for training deep neural networks and handling non-convex optimization problems. Through its adaptive learning rate and momentum updates, Nadam enhances the efficiency and effectiveness of the optimization process, contributing to improved model performance in various machine learning tasks. Nadam seamlessly integrates the adaptive moment estimation (Adam) algorithm with Nesterov accelerated gradient (NAG) descent. This combination allows Nadam to optimize its learning rate dynamically, adapting to the varying gradients encountered during training. By incorporating NAG, Nadam enhances its ability to anticipate and adjust for momentum, facilitating smoother convergence towards optimal solutions. This adaptive approach enables Nadam to navigate complex optimization landscapes more efficiently, resulting in improved training performance and convergence speed. Through its fusion of Adam and NAG, Nadam stands as a powerful optimizer capable of handling diverse optimization challenges with agility and effectiveness. Following the updating of ω^{temp} for every client, ∇^t is calculated by taking the difference between the updated temporary weight variable ω^{temp} and ω^{med}. It uses exponential decay to update the first moment estimate nm^t and the second moment estimate nv^t. Next, it applies the Nadam update rule to change the global model parameters, as seen in (3.14). Flow of the proposed strategy can be viewed in figure (3.3)

$$\omega^{(t+1)} = \omega^{\text{med}} + \frac{\beta_1 \cdot nm^t + (1 - \beta_1) \cdot n\nabla^t}{\sqrt{v^t} + \delta} \cdot \eta \tag{3.14}$$

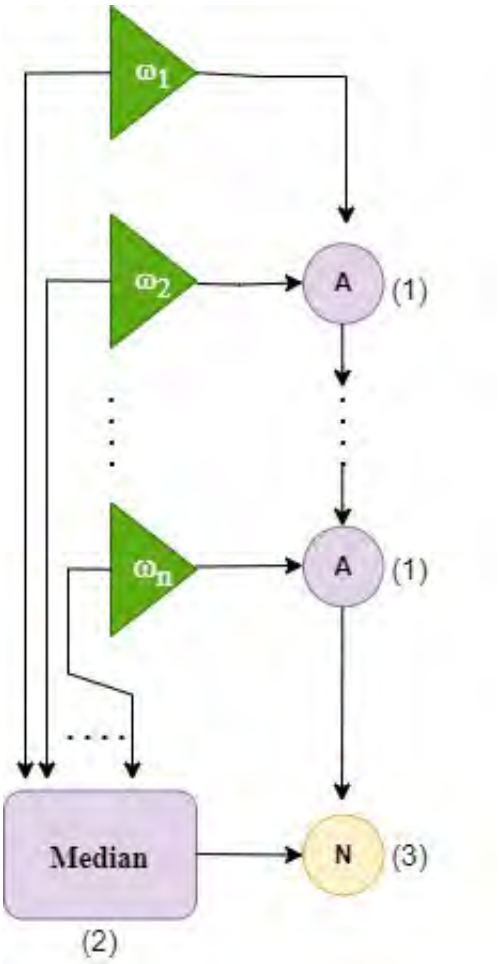


Figure 3.3: Overview of our proposed strategy. (1) Each clients updated weights are updated with the next client upto n clients and combine all the weights. (2) Median weight is calculated considering all clients weights. (3) Median weight is updated with the combined weight.

Chapter 4

EXPERIMENTAL RESULTS AND DISCUSSION

In a federated learning environment, the configuration and parameters used for conducting trials or experiments are referred to as federated settings. This covers the configuration in which the model is dispersed among several servers or client devices, each of which is keeping samples of local data. Since only model updates are shared and raw data stays on the client devices in federated setups, data privacy is protected.Research in federated environments is carried out using simulation functionality. The behavior and features of federated client devices are imitated using simulations as opposed to actual client devices. With this method, researchers can manage and adjust several elements of the experiment in a regulated setting, including the number of client devices, data distributions, and communication methods.The utilization of simulation functionality is required due to the limited number of federated client devices that are accessible throughout the execution of the experiment. Obtaining a large number of client devices for tests may be difficult in real-world situations because of logistical issues or privacy concerns. As a result, using simulations to mimic federated learning environments with a scalable number of virtual client devices is a useful alternative.Three metrics are used in the study to assess the experimental results. These metrics are used as quantitative indicators to evaluate the efficacy and performance of the federated learning strategy. Accuracy, convergence speed, communication overhead, privacy assurances, and any other pertinent performance indicators are common evaluation metrics in federated learning investigations, contingent on the particular goals of the investigation.this sample emphasizes the approach taken to carry out experiments in federated learning environments, stressing the significance of utilizing suitable metrics to assess findings and the usage of simulation features to get around real-world deployment constraints.

4.1 Dataset

Since clustering is a type of unsupervised learning that involves grouping similar data points together based on their inherent characteristics, the dataset plays a crucial role in the clustering process. The dataset serves as the foundation upon which clustering algorithms operate, providing the raw material from which clusters are formed. The quality, size, and diversity of the dataset significantly impact the effi-

cacy and performance of clustering algorithms. A well-curated dataset with diverse and representative samples enhances the clustering process, enabling the algorithms to discern meaningful patterns and structures within the data. Conversely, a dataset lacking in diversity or containing noisy or irrelevant data may hinder the clustering process, leading to suboptimal results. Therefore, careful preprocessing and selection of the dataset are essential steps in preparing for the clustering task. The dataset's dimensionality and scale also influence the clustering process, as algorithms may perform differently depending on the number of features and their magnitude. Additionally, the dataset's distribution and density affect the clustering outcomes, with algorithms exhibiting varying degrees of sensitivity to skewed or sparse data distributions. Moreover, the dataset's completeness and consistency are critical factors in ensuring the reliability and validity of clustering results. Incomplete or inconsistent data may introduce biases or distortions, leading to inaccurate cluster assignments. Furthermore, the dataset's size and computational complexity impact the scalability and efficiency of clustering algorithms, with larger datasets requiring more computational resources and potentially longer processing times. Overall, the dataset serves as the bedrock of the clustering process, influencing every stage from algorithm selection to evaluation and interpretation. Therefore, meticulous attention to dataset quality, relevance, and preprocessing is essential for achieving meaningful and actionable clustering results. The software looks for patterns and structures in the data without specific instructions. It provides the foundation for clustering algorithms and provides a place of departure for identifying patterns and grouping objects.The quality and type of the dataset have a significant impact on the efficacy and usability of clustering findings in real-world applications. In the experiment, simulations are conduced within a federated environment using benchmark datasets, including the Letter Image Recognition Data, Deterding Vowel Recognition Data, Protein Localization Data, and Image Segmentation data. These datasets were chosen to represent a diverse range of domains and data types, allowing for comprehensive evaluation of our proposed approach. The Letter Image Recognition Data [30] comprises images of handwritten letters, commonly used for character recognition tasks. The Deterding Vowel Recognition Data [31] consists of speech recordings for vowel recognition applications. The Protein Localization Data [32] includes sequences of amino acids for predicting protein subcellular localization. Lastly, the Image Segmentation data [33] contains images segmented into distinct regions for image analysis tasks. By leveraging these benchmark datasets, it is aimed aimed to assess the performance and generalizability of our federated clustering approach across various domains and data modalities. The Letter Image Recognition Dataset consists of 20,000 samples, with each sample representing one of 26 distinct classes corresponding to the English alphabet. Each sample in the dataset is an image depicting a handwritten letter, with variations in handwriting styles and shapes. This dataset is commonly used for character recognition tasks and provides a comprehensive representation of handwritten letters. The dataset's large size and diversity facilitate robust training and evaluation of clustering algorithms. Each sample in the Letter Image Recognition Dataset is described by 16 attributes, which capture various features of the letter images. These attributes includes characteristics such as width, height, and pixel intensity values, symmetry, diagonal intensity, and vertical and horizontal position within a specific pixel grid. By representing each sample with these descriptive attributes, the dataset provides

a structured representation of the handwritten letters. These attributes serve as input features for clustering algorithms, enabling the algorithms to discern patterns and similarities among the letter images. Letter Image Recognition Dataset can be visulaized in figure (4.1)

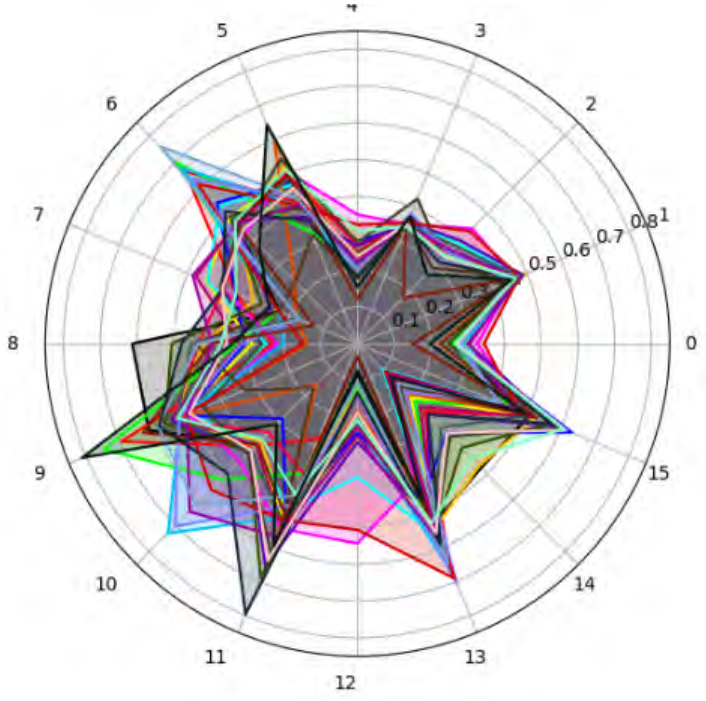


Figure 4.1: Visualization of the Letter Image Recognition Data using radar chart, where Each feature or attribute is represented by a different axis that radiates outward from the center, and the data values are plotted along these axes.Diffrent color represnts diffrernt class

The Deterding Vowel Recognition Dataset comprises 991 samples, with each sample characterized by 10 attributes relevant to vowel sounds. These attributes capture various acoustic features, such as pitch, formant frequencies, and duration, essential for distinguishing between different vowel sounds. The dataset is commonly used in speech recognition and phonetics research for vowel classification tasks. Each sample represents a recorded instance of a spoken vowel sound, providing a comprehensive representation of vowel articulation. The Deterding Vowel Recognition Dataset comprises 11 distinct classes, each representing a different vowel sound. These classes encompass a range of vowel articulations found in various languages and dialects. Each class corresponds to a specific vowel sound, such as "a," "e," "i," "o," and "u," among others. The dataset provides a comprehensive representation of vowel phonemes, enabling researchers to study the acoustic characteristics and variability of vowel sounds. The Deterding Vowel Recognition Dataset can be visulaized in figure (4.2)
The Protein Localization Dataset consists of eight classes, seven characteristics, and 336 samples. Each sample represents a sequence of amino acids, and the dataset aims to predict the subcellular localization of proteins based on these sequences. The eight classes correspond to different subcellular locations, such as nucleus, cytoplasm, mitochondrion, and others. The dataset's characteristics may include attributes related to amino acid composition, sequence motifs, or physicochemical

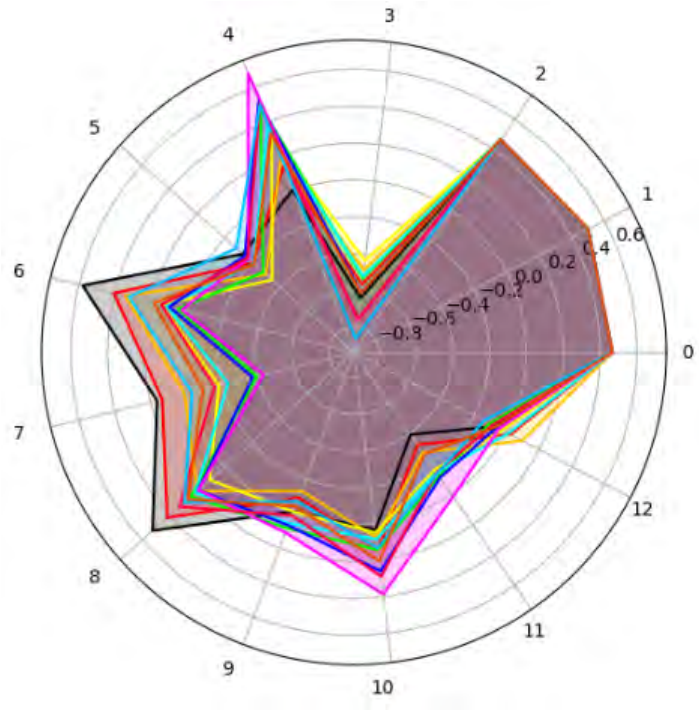


Figure 4.2: Visualization of the Vowel Recognition Data using radar chart, where Each feature or attribute is represented by a different axis that radiates outward from the center, and the data values are plotted along these axes.Diffrent color represnts diffrernt class

properties.The Protein Localization Dataset can be visulaized in figure (4.3)
The Image Segmentation dataset comprises 2,311 samples, 19 characteristics, and 7 classes. Each sample represents an image segmented into distinct regions, with the goal of classifying each pixel into one of the seven predefined classes. The dataset's characteristics may include pixel intensity values, texture features, and spatial relationships among neighboring pixels. The seven classes correspond to different semantic regions within the images, such as sky, vegetation, buildings, and roads. The Image Segmentation dataset can be visulaized in figure (4.4)
In preprocessing our dataset, which comprises both categorical and numerical values, pandas factorize function is utilized to obtain a numeric representation of categorical data. This conversion enables machine learning algorithms to process categorical features effectively by assigning unique numerical codes to each category. By converting categorical variables into numerical representations, we ensure compatibility with algorithms that require numeric input. The factorize function assigns a unique integer to each distinct category, facilitating subsequent analysis and modeling tasks. This approach preserves the inherent structure of categorical variables while enabling seamless integration with numerical features. Through this preprocessing step, the dataset's suitability is enhanced To ensure the quality and reliability of our dataset, Samples containing null values have been removed. Null values, also known as missing data, can significantly impair the performance of machine learning models, as many algorithms struggle to handle incomplete or inconsistent data. By eliminating samples with null values, we mitigate the risk of introducing biases or inaccuracies into our analysis. This preprocessing step enhances the robustness and reliability of our dataset, enabling more accurate and effective model training. Additionally,

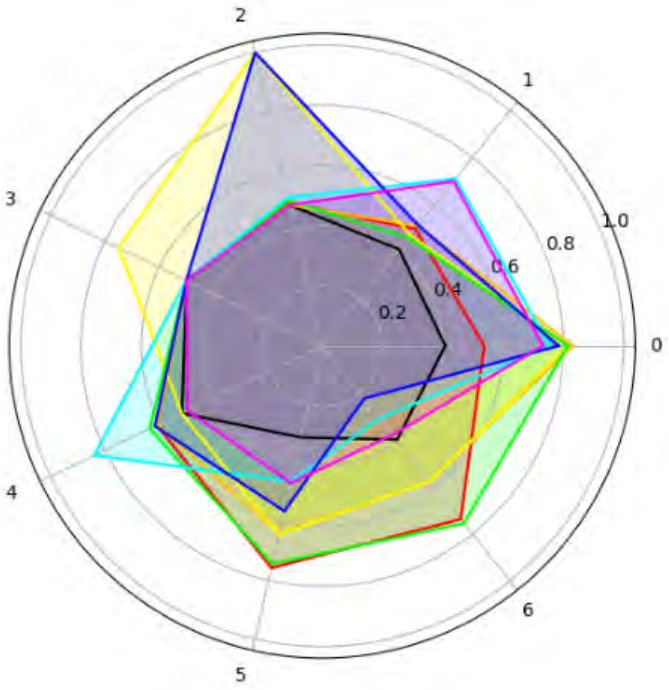


Figure 4.3: Visualization of the Protein Localization Data using radar chart, where Each feature or attribute is represented by a different axis that radiates outward from the center, and the data values are plotted along these axes.Diffrent color represnts diffrernt class

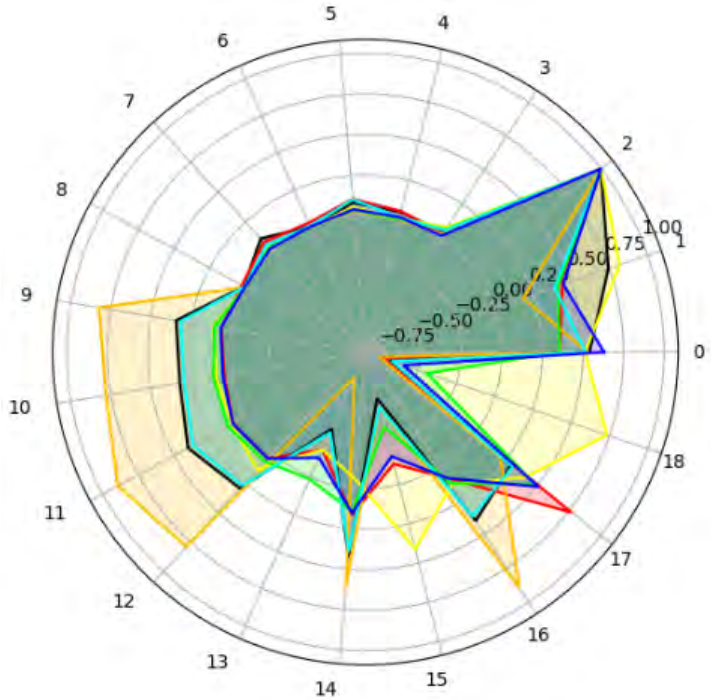


Figure 4.4: Visualization of the Image Segmentation Data using radar chart, where Each feature or attribute is represented by a different axis that radiates outward from the center, and the data values are plotted along these axes.Diffrent color represnts diffrernt class

removing null values simplifies the data cleaning process and improves the interpretability of our results. Through this approach, It is upholded rigorous standards of data integrity and quality assurance in our analysis pipeline. Dataset's can be visulaized in figure (4.1) , (4.2), (4.3) and (4.4).

4.2 Autoencoder architectures behind the experiment using different dataset

This approach to addressing the Letter Image Recognition Dataset involves leveraging a fully connected autoencoder network. This architecture enables us to capture the intricate features and patterns present in the handwritten letter images. By employing a fully connected structure, the autoencoder network can effectively encode and decode the input data, preserving essential information during the reconstruction process. In this case Our network architecture consists of five fully connected (FC) layers, meticulously crafted to optimize the encoding and decoding processes. Each FC layer plays a critical role in transforming the input data into a compressed representation and reconstructing it back to its original form. These layers are strategically designed to capture intricate features and patterns inherent in the data. By cascading multiple FC layers, hierarchical abstraction and representation learning within the network is enabled. This hierarchical structure facilitates the extraction of increasingly abstract features as the data progresses through the layers. Through careful design and optimization of the FC layers, it is aimed to enhance the network's capacity to learn meaningful representations of the input data. With layer configurations of 16, 12, 10, 12, and 16 neurons respectively, the network architecture is tailored to balance complexity and representational capacity. This architecture can be visulaized in figure (4.8) These configurations are optimized to capture essential features and patterns within the data while minimizing redundancy. By adjusting the number of neurons in each layer, efficient information processing and representation learning is ensured throughout the network. For the Deterding Vowel Recognition Dataset, our primary architectural framework involves the utilization of a fully connected autoencoder network. This network architecture is chosen for its ability to capture the essential acoustic features relevant to vowel recognition tasks. By employing a fully connected structure, it is aimed to extract and encode informative representations of the vowel sounds present in the dataset. Five fully connected (FC) layers make up this network architecture, which is thoughtfully structured to enable efficient encoding and decoding of the dataset's complex properties. This architecture can be visulaized in figure (4.8) Specifically, the fully connected (FC) layers in our autoencoder network for the Deterding Vowel Recognition Dataset are structured with 10, 8, 6, 8, and 10 neurons respectively. This configuration is meticulously designed to balance complexity and representational capacity, ensuring efficient information processing and feature extraction. By adjusting the number of neurons in each layer, it is aimed to capture the essential acoustic features relevant to vowel recognition tasks while minimizing redundancy. For the Protein Localization Dataset, a fully connected autoencoder network comprising three FC layers is employeed. These FC layers are structured with 7, 4, and 7 neurons respectively. This architecture is tailored to effectively capture the diverse

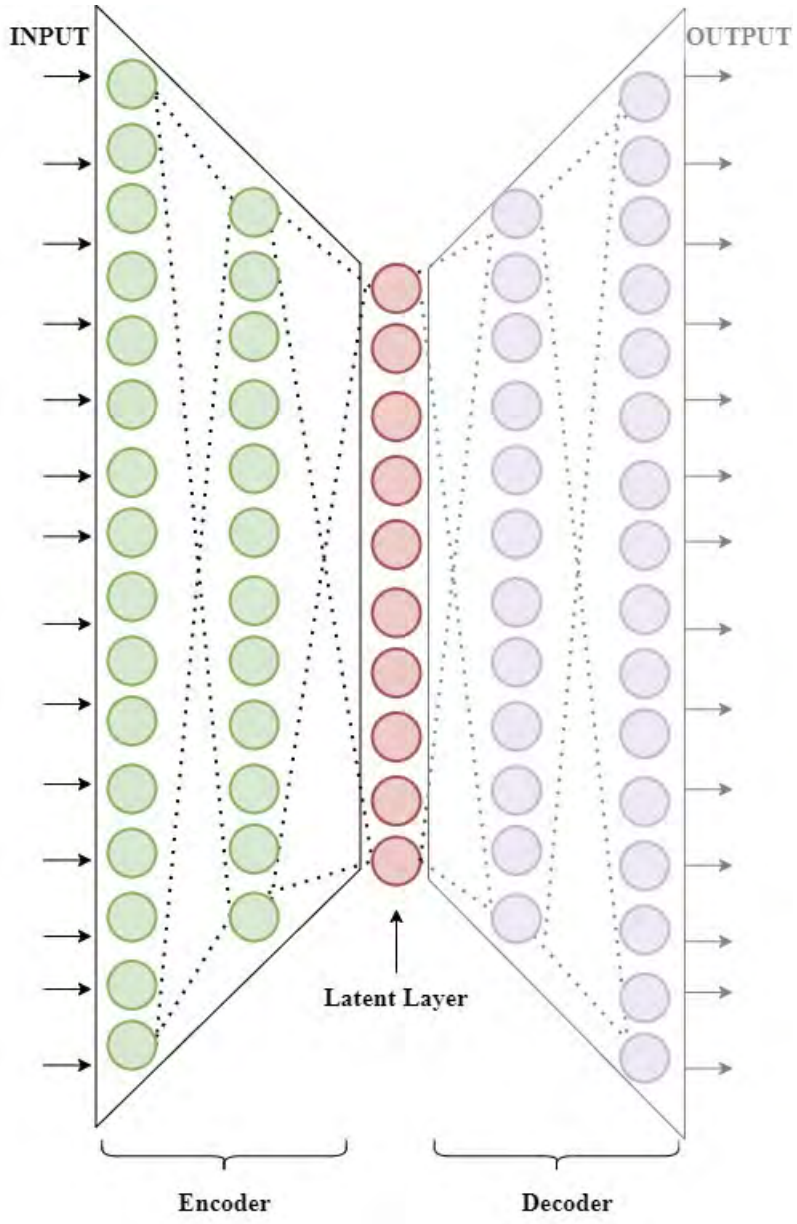


Figure 4.5: Visualization of the autoencoder architecture used for Letter Image Recognition Data

characteristics and spatial relationships present in protein sequences. By utilizing a fully connected structure, it is aimed to extract informative representations of the amino acid sequences relevant to protein subcellular localization prediction.This architecture can be visulaized in figure (4.8)
For the Image Segmentation dataset, we have employed a fully connected autoencoder network. Each FC layer within this architecture is tailored with specific neuron counts, structured as 19, 16, 14, 12, 10, 12, 14, 16, and 19 neurons respectively. This carefully designed configuration enables effective encoding and decoding processes, facilitating the extraction of meaningful features from the segmented image data. By utilizing a fully connected structure, we aim to capture the spatial relationships and semantic information present in the image regions. This architecture can be visulaized in figure (4.8) In our autoencoder architectures, we integrate Rectified Linear

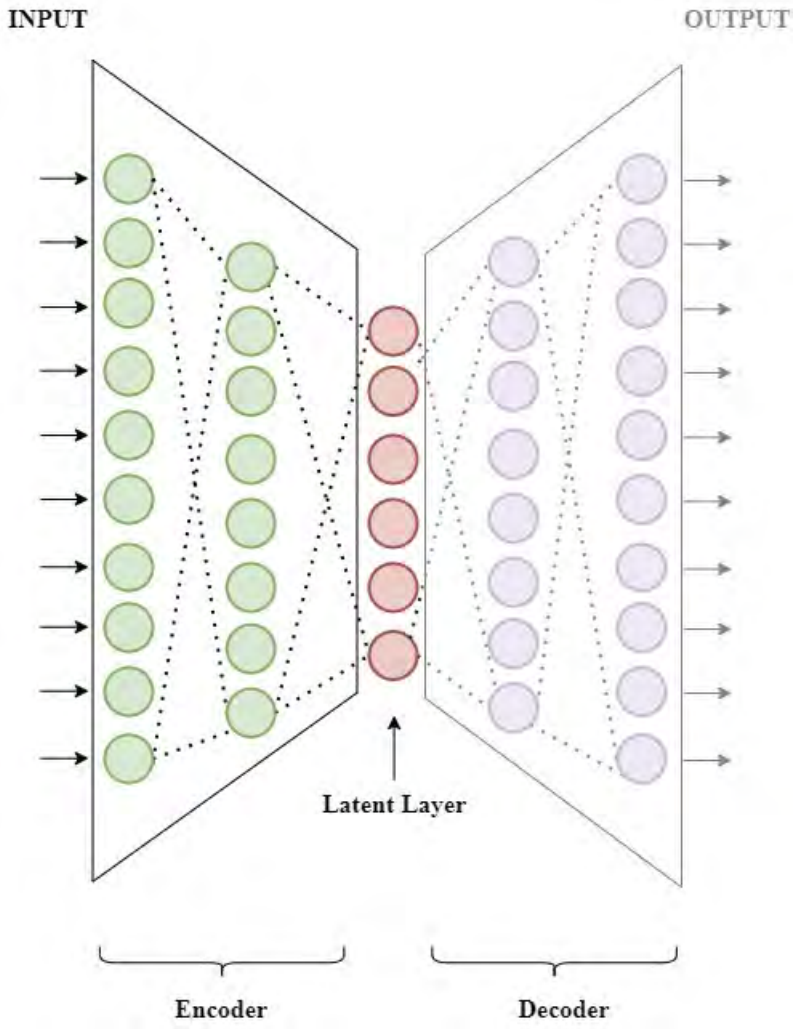


Figure 4.6: Visualization of the autoencoder architecture used for Detarding Vowel Data

Unit (ReLU) activation functions within the hidden layers to introduce non-linearity and facilitate feature learning. ReLU activation functions are widely used for their simplicity and effectiveness in overcoming the vanishing gradient problem. These functions enable efficient propagation of signals through the network, promoting faster convergence during training. Additionally, we employ the softmax activation function in the output layer to produce probabilistic outputs for classification tasks. Softmax activation ensures that the output values are normalized to represent probabilities, making it suitable for multi-class classification problems. ReLU activation is chosen for its ability to introduce non-linearity while efficiently mitigating the vanishing gradient problem, enhancing the model's capacity to capture complex patterns and features within the data. This activation function allows for faster convergence during training by addressing the issue of diminishing gradients encountered in deep neural networks. By enabling the network to learn complex representations of the input data, ReLU activation promotes better model performance and generalization across various tasks and datasets. Its simplicity and effectiveness make it a popular choice in modern neural network architectures, contributing to improved training efficiency and effectiveness. Meanwhile, softmax activation is em-

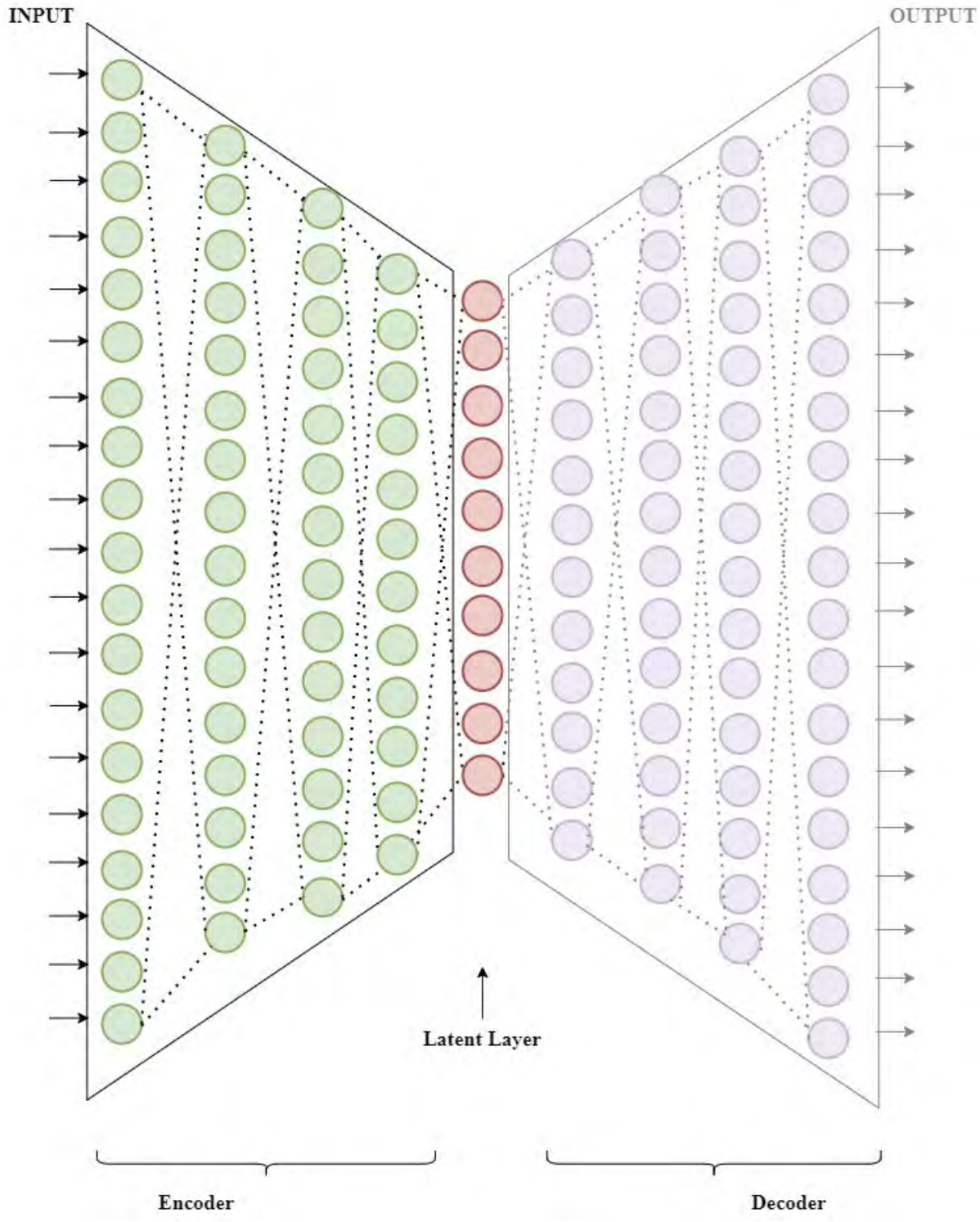


Figure 4.7: Visualization of the autoencoder architecture used for Image Segmentation Data

ployed in the output layer to produce probability distributions over multiple classes. This activation function ensures that the output values are normalized to represent probabilities, making it suitable for multi-class classification problems. By transforming the output into a probability distribution, softmax activation facilitates the interpretation of model predictions and enables probabilistic reasoning. It allows for

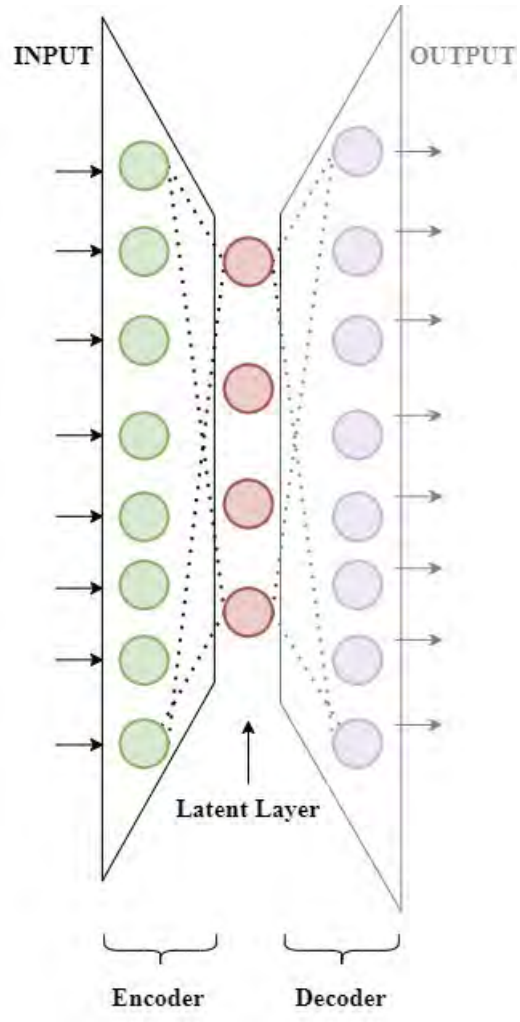


Figure 4.8: Visualization of the autoencoder architecture used for Protein Localization Data

clear and intuitive understanding of the model's confidence in assigning each class label, aiding in decision-making processes. The utilization of softmax activation enhances the interpretability and reliability of the model's predictions, contributing to its effectiveness in classification tasks. By combining ReLU and softmax activations, we aim to enhance the expressive power and accuracy of our autoencoder models for various datasets and tasks. We utilize Kullback-Leibler (KL) divergence as the loss function in our model training process. This choice of loss function enables us to measure the discrepancy between the predicted probability distribution and the true distribution of the target variable. By minimizing the KL divergence, we aim to align the model's predictions more closely with the actual data distribution. KL divergence serves as a metric for assessing the dissimilarity between probability distributions, guiding the optimization process towards better model performance. Its use in our training process facilitates effective learning and convergence towards accurate predictions, enhancing the overall quality of our models. Our models aim to learn latent representations that closely resemble the intended distribution by reducing KL divergence during training. This approach enables efficient data compression and reconstruction, as the model learns to encode essential information while minimizing information loss. By optimizing for lower KL divergence, our models strive to

capture the underlying structure and patterns within the data, facilitating accurate reconstruction of the input. This process of minimizing KL divergence promotes effective learning of meaningful representations, enhancing the model's ability to capture complex relationships within the data. Through this framework, our models achieve efficient data compression and reconstruction while preserving essential information, contributing to their overall effectiveness in various tasks.

4.3 Configuration behind the experiments using different dataset

In this experiment, four distinct datasets have been employed, resulting in four alternative federated setup configurations. Each dataset presents unique characteristics and challenges, necessitating tailored federated learning setups to optimize performance. By configuring federated setups specific to each dataset, we ensure that the learning process is optimized to accommodate the data's diversity and complexity. These alternative configurations allow for flexibility in adapting federated learning methodologies to different data domains and tasks. Through careful consideration of dataset-specific requirements, we aim to maximize the effectiveness and efficiency of federated learning across various experimental scenarios. In the first setup, ten clients are engaged to participate in a federated learning task, utilizing The Letter Image Recognition Dataset. This configuration allows for distributed learning across multiple client devices, each contributing its local data to the collaborative training process. By involving ten clients, we aim to leverage a diverse range of data sources to enhance the robustness and generalization of the trained model. Of the 20,000 samples in The Letter Image Recognition Dataset [30], 18,000 are utilized for federated learning tasks, serving as the training data distributed across the ten participating clients. The remaining 2,000 samples are reserved for evaluating the clustering performance of the trained model. This division allows for a comprehensive assessment of the model's efficacy in clustering handwritten letters while ensuring a robust evaluation on unseen data. In the second configuration, Deterding Vowel Recognition Data[31] is utilized to engage five clients in a federated learning assignment. While federated learning is employed with the remaining 800 data, the 191 examples are utilized to evaluate the clustering performance.This setup involves a smaller number of clients compared to the first configuration, allowing for a more focused and streamlined collaborative training process. By distributing the dataset across five clients, we aim to leverage their collective data while minimizing communication overhead and computational burden. This configuration enables efficient model training while still benefiting from the diversity of data sources available within the dataset. In the third setup, five clients are engaged to participate in a federated learning task, utilizing the Protein Localization Dataset. This configuration focuses on leveraging the distributed nature of the federated learning framework to train models on the Protein Localization Dataset. By involving five clients, we aim to harness diverse data sources while maintaining scalability and efficiency in the federated learning process. In this setup, three hundred and sixty-six samples from the Protein Localization Dataset[32] are allocated for federated learning tasks, serving as the training data distributed across the five participating clients. The remaining seventy-six samples are reserved for assessing the clustering performance of

the trained model. This division ensures a comprehensive evaluation of the model's effectiveness in predicting protein subcellular localization while providing a robust assessment on unseen data. In the fourth arrangement, ten clients are engaged in a federated learning job, utilizing Image Segmentation data[33]. This configuration enables distributed model training across a larger number of clients, leveraging their collective data for collaborative learning. By involving ten clients, we aim to capture a diverse range of image segmentation patterns and characteristics present within the dataset. In this configuration, while the remaining 2,000 samples from the Image Segmentation dataset are utilized for federated learning across the ten participating clients, the clustering performance is evaluated using 311 samples. This division ensures that a significant portion of the dataset is dedicated to collaborative model training, allowing for effective learning across distributed client devices. Meanwhile, the reserved samples for evaluation provide a robust assessment of the trained model's clustering performance on unseen data. In each arrangement, data samples are distributed equally among the participating clients. This equal distribution ensures fairness and balance in the federated learning process, allowing each client to contribute an equal share of data for collaborative model training. By distributing the samples evenly, we aim to prevent bias and promote representativeness across the training dataset for each client. This approach facilitates effective collaboration and model convergence while maintaining consistency in the learning process across all clients. Through the equitable distribution of data samples, we optimize the efficiency and effectiveness of federated learning across diverse experimental setups and datasets. After training the local autoencoder for 100, 10, 10, and 60 epochs in each respective configuration, its performance is assessed using a local dataset in each communication round. This verification step ensures that the local models have adequately learned the underlying patterns and features present within the data. By evaluating the trained autoencoder models using local datasets, we validate their effectiveness in capturing the essential characteristics of the data while ensuring consistency and reliability across different training epochs. This iterative process of model verification helps monitor the progress and performance of the local models throughout the federated learning process, facilitating robust and reliable model training across distributed client devices. After every round, the local training global encoder weights are adjusted based on the aggregated updates from the participating clients. This adjustment ensures that the global model parameters are continuously refined to reflect the collective knowledge learned from the distributed data sources. Through this iterative process, the global encoder adapts to the evolving patterns and characteristics present within the federated dataset. For the purpose of updating the weights, we employed various techniques including FedAvg, FedMedian, FedAdam, and FednadamN (proposed). Each technique offers distinct approaches to aggregating model updates from distributed clients. FedAvg computes the average of model parameters, FedMedian selects the median, FedAdam adapts the Adam optimizer, while FednadamN introduces a novel optimization method. This comprehensive evaluation aimed to determine the most effective strategy for collaborative weight updates in federated learning setups. We conducted only one communication round of the simulation in each configuration with every client. This single round allowed for a preliminary evaluation of the federated learning setup and its performance using the specified techniques. In federated environments, we utilize the output of the latent layer as the input for the cluster-

ing algorithms following only one communication loop. This approach enables rapid integration of federated learning outputs into downstream clustering tasks, allowing for timely analysis and interpretation of model performance. By leveraging the latent representations learned by the federated models, we aim to facilitate efficient clustering without the need for extensive communication rounds or data sharing, thus preserving privacy and minimizing computational overhead. This streamlined process enhances the agility and effectiveness of federated learning frameworks in generating actionable insights from distributed data sources.

4.4 Performance metrics

Once clustering is completed, its performance can be assessed using various metrics. Ideal clustering is characterized by minimal intra-cluster distance, meaning that data points within the same cluster are similar, and maximal inter-cluster distance, indicating that data points in different clusters are dissimilar. Quantifying these characteristics helps evaluate the quality of the clustering solution. Metrics such as silhouette score, Davies-Bouldin index, and Calinski-Harabasz index provide insights into the cohesion within clusters and the separation between clusters. By assessing these metrics, the effectiveness of the clustering algorithm can be determined in partitioning the data into meaningful groups. Ultimately, the goal is to achieve clusters that are both internally cohesive and well-separated from each other, indicating a high-quality clustering solution. As dataset possess ground truth labels, extrinsic measures for evaluating clustering performance have been employeed. Extrinsic measures rely on known class labels to assess the alignment between clusters and true classes. By comparing clustering results to ground truth, metrics such as accuracy, purity, and F-measure can be evaluated quantitatively, providing insights into the clustering algorithm's effectiveness in capturing underlying structures and separating distinct classes. Leveraging extrinsic measures facilitates objective evaluation and comparison of clustering methods based on their alignment with known ground truth, enhancing our understanding of their performance in real-world applications.

4.4.1 V-measure score

One metric used to assess the quality of clustering in machine learning is the V-measure score. The V-measure combines homogeneity and completeness measures to offer a comprehensive evaluation of clustering performance. Homogeneity assesses the extent to which each cluster contains only data points belonging to a single class, while completeness measures the degree to which all data points belonging to a given class are assigned to the same cluster. By combining these two measures, the V-measure provides a holistic view of clustering quality, capturing both the accuracy and completeness of the clustering solution. This metric is particularly useful for evaluating the effectiveness of clustering algorithms in capturing the true structure of the data and producing meaningful cluster assignments. Regarding N data samples, C distinct class labels, k clusters, and the ack number of data points associated with the class c (as well as the cluster k. Next, the following in (4.1) yields the homogeneity h.

$$h = 1 - \frac{H(C,K)}{H(C)} \tag{4.1}$$

$$H(C,K) = -\sum_{k=1}^{K}\sum_{c=1}^{C}\frac{a_{ck}}{N}\log\left(\frac{a_{ck}}{\sum_{c=1}^{C}a_{ck}}\right) \tag{4.2}$$

$$H(C) = -\sum_{c=1}^{C}\left(\frac{\sum_{k=1}^{K}a_{ck}}{C}\log\left(\frac{\sum_{k=1}^{K}a_{ck}}{C}\right)\right) \tag{4.3}$$

the completeness c is given by the equation (4.4).

$$c = 1 - \frac{H(K,C)}{H(K)} \tag{4.4}$$

$$H(K,C) = -\sum_{c=1}^{C}\sum_{k=1}^{K}\frac{a_{ck}}{N}\log\left(\frac{a_{ck}}{\sum_{k=1}^{K}a_{ck}}\right) \tag{4.5}$$

$$H(K) = -\sum_{k=1}^{K}\left(\frac{\sum_{c=1}^{C}a_{ck}}{C}\log\left(\frac{\sum_{c=1}^{C}a_{ck}}{C}\right)\right) \tag{4.6}$$

These two metrics are combined to provide the V-measure score, which is a single number that represents the clustering algorithm's overall efficacy. Better clustering performance is indicated by a higher V-measure score. The formula for the V-measure score, V, may be obtained as follows (4.7)

$$V = \frac{2hc}{h+c} \tag{4.7}$$

4.4.2 Mutual Info Score

Regardless of permutations, it assesses the degree of agreement between the genuine labels and the projected cluster assignments. In the context of clustering evaluation, the Mutual Information score quantifies the amount of information shared between the true labels and the predicted clusters. This metric measures the level of agreement between the two sets of labels, providing insights into the consistency and accuracy of the clustering solution. By evaluating the mutual information, we can assess the degree to which the clustering algorithm captures the underlying structure of the data and produces meaningful cluster assignments. The Mutual Information score is widely used in clustering analysis to evaluate the effectiveness of different algorithms and parameter configurations in achieving accurate clustering results. A higher Mutual Information score indicates better agreement between the true labels and the predicted clusters. This metric quantifies the level of similarity between the two sets of labels, reflecting the accuracy and consistency of the clustering solution. By assessing the Mutual Information score, we can gauge the effectiveness of the clustering algorithm in capturing the underlying structure of the data and producing meaningful cluster assignments. A higher score signifies a stronger correspondence

between the true labels and the clustering results, indicating a more accurate representation of the data's intrinsic patterns. Equation (4.8) may be used to calculate the Mutual Information score given a collection of predicted clusters V and a set of true labels U.

$$\mathrm{MI}(U,V) = \sum_{u\in U}\sum_{v\in V} p(u,v)\log\left(\frac{p(u,v)}{p(u)p(v)}\right)) \tag{4.8}$$

The combined probability of the genuine label u and the anticipated cluster v is represented here by $p(u,v)$. The probability of the real label u is represented by $p(u)$ and that of the projected cluster v by $p(v)$.

4.4.3 Rand Index Score

In order to determine the comparability of two clusterings, the Rand Index (RI) examines pairs of data points to determine if they are assigned to the same or different clusters in both the true and predicted clusterings. This index quantifies the level of agreement between the two clusterings, providing insights into their similarity and consistency. By analyzing point pairs, the Rand Index offers a robust evaluation of clustering performance, measuring the degree to which the predicted clustering aligns with the ground truth. This metric is valuable for assessing the accuracy and reliability of clustering algorithms in capturing the underlying structure of the data and producing meaningful cluster assignments. Given a set of data points and two clustering solutions, the Rand Index evaluates all possible combinations of data points and determines how their relationships change between the two clusterings. This index quantifies the level of agreement between the two clustering solutions by comparing the assignments of data points to clusters. By examining the consistency of clustering assignments across different solutions, the Rand Index provides insights into the similarity and correspondence between the two clusterings. This metric is valuable for assessing the accuracy and reliability of clustering algorithms in capturing the underlying structure of the data and producing meaningful cluster assignments. Perfect agreement between two clusterings is represented by a Rand Index (RI) value of 1, indicating that all data point pairs are assigned to the same clusters in both clusterings. Conversely, no agreement, as would occur in random clustering, is indicated by an RI value of 0, signifying that the clustering solutions bear no resemblance to each other. The RI value provides a clear measure of the similarity and consistency between two clustering solutions, with higher values indicating greater agreement and alignment between the clusterings. formula for the Unadjusted Rand Index (RI) is in (4.9).

$$RI = \frac{a+b}{a+b+c+d} \tag{4.9}$$

The numbers a and b in this case indicate the number of point pairs that are in the same cluster in the true and predicted clusterings, c the number of point pairs that are in the same cluster in the true clustering but in different clusters in the predicted clustering, and d the number of point pairs that are in different clusters in the true clustering but in the same cluster in the predicted clustering.

4.5 Results and Analysis

The findings from the study are summarized in table 4.1, table 4.2 and table 4.3. To facilitate identification and comparison, the higher performance approach is denoted in these tables with underlining. This visual cue highlights the superior performance or preference among different approaches, making it easier for users to discern the most effective solution. By visually emphasizing the better-performing option, the tables provide a clear and intuitive means of comparison, aiding decision-making and analysis in various contexts such as experimental results, model evaluations, or algorithm comparisons. From the table 4.1 , 4.2 and 4.3 we observed that

Table 4.1: V measure score on Letter Image Recognition Data, Deterding Vowel Recognition Data, Protein Localization Data, Image Segmentation Data under different strategy and clustering algorithms.A high V Measure score indicates strong agreement between the clustering result and the true class labels. The underline highlight the best-performing compared approach.

Dataset	Model aggregation strategy	Kmeans	Kmedoids	Fuzzy kmeans	Minibatch kmeans
Letter Image Recognition Data	FedAvg	0.259	0.262	0.218	0.243
	FedMedian	0.353	0.337	0.214	0.345
	FedAdam	0.293	0.287	0.232	0.297
	FednadamN (proposed)	0.373	0.355	0.260	0.370
Deterding Vowel Recognition Data	FedAvg	0.238	0.213	0.244	0.242
	FedMedian	0.243	0.248	0.239	0.254
	FedAdam	0.261	0.242	0.243	0.258
	FednadamN (proposed)	0.264	0.259	0.257	0.255
Protein Localization Data	FedAvg	0.357	0.418	0.367	0.399
	FedMedian	0.270	0.153	0.290	0.230
	FedAdam	0.407	0.414	0.381	0.386
	FednadamN (proposed)	0.512	0.527	0.508	0.513
Image Segmentation Data	FedAvg	0.508	0.455	0.477	0.453
	FedMedian	0.442	0.462	0.436	0.475
	FedAdam	0.517	0.466	0.516	0.516
	FednadamN (proposed)	0.561	0.512	0.539	0.565

Based on the V-measure score and mutual information score, our analysis of the first setup utilizing the Letter Image Recognition dataset indicates that the clustering algorithms outperformed other methods when employing our model aggregation strategy. However, the Rand score indicated relatively low results. This suggests that while our model aggregation strategy effectively improves clustering performance in terms of capturing cluster homogeneity and completeness, it may not consistently align with the ground truth labels represented by the Rand score. Further investigation and refinement may be necessary to address this discrepancy and enhance the overall clustering accuracy in future iterations. The performance of clustering algorithms utilizing the FedAvg, FedMedian, FedAdam, and FednadamN (proposed) methods is depicted in Figures (4.9), (4.10), and (4.11).

Table 4.2: Rand Index score on Letter Image Recognition Data, Deterding Vowel Recognition Data, Protein Localization Data, Image Segmentation Data under different strategy and clustering algorithms.A high V Measure score indicates strong agreement between the clustering result and the true class labels. The underline highlight the best-performing compared approach.

Dataset	Model aggregation strategy	Kmeans	Kmedoids	Fuzzy kmeans	Minibatch kmeans
Letter Image Recognition Data	FedAvg	0.921	0.911	0.925	0.923
	FedMedian	0.929	0.919	0.915	0.927
	FedAdam	0.928	0.926	0.923	0.929
	FednadamN (proposed)	0.927	0.932	0.919	0.925
Deterding Vowel Recognition Data	FedAvg	0.833	0.836	0.8428	0.832
	FedMedian	0.834	0.834	0.837	0.841
	FedAdam	0.836	0.827	0.838	0.825
	FednadamN (proposed)	0.847	0.841	0.850	0.832
Protein Localization Data	FedAvg	0.726	0.735	0.722	0.731
	FedMedian	0.695	0.671	0.711	0.692
	FedAdam	0.747	0.745	0.742	0.746
	FednadamN (proposed)	0.772	0.761	0.769	0.774
Image Segmentation Data	FedAvg	0.802	0.821	0.819	0.813
	FedMedian	0.814	0.829	0.821	0.816
	FedAdam	0.840	0.834	0.8427	0.843
	FednadamN (proposed)	0.844	0.846	0.843	0.849

Table 4.3: Mutual Info Score on Letter Image Recognition Data, Deterding Vowel Recognition Data, Protein Localization Data, Image Segmentation Data under different strategy and clustering algorithms.A high V Measure score indicates strong agreement between the clustering result and the true class labels. The underline highlight the best-performing compared approach.

Dataset	Model aggregation strategy	Kmeans	Kmedoids	Fuzzy kmeans	Minibatch kmeans
Letter Image Recognition Data	Fedavg	0.825	0.818	0.686	0.776
	Fedmedian	1.141	1.067	0.674	1.106
	Fedadam	0.946	0.923	0.714	0.961
	Fednadamn (proposed)	<u>1.192</u>	<u>1.148</u>	<u>0.817</u>	<u>1.177</u>
Deterding Vowel Recognition Data	Fedavg	0.553	0.501	0.577	0.565
	Fedmedian	0.569	0.584	0.564	<u>0.601</u>
	Fedadam	0.610	0.557	0.573	0.594
	Fednadamn (proposed)	<u>0.628</u>	<u>0.611</u>	<u>0.615</u>	0.595
Protein Localization Data	Fedavg	0.63	0.723	0.642	0.701
	Fedmedian	0.469	0.264	0.514	0.402
	Fedadam	0.710	0.714	0.681	0.690
	Fednadamn (proposed)	<u>0.900</u>	<u>0.908</u>	<u>0.908</u>	<u>0.904</u>
Image Segmentation Data	Fedavg	0.944	0.870	0.901	0.858
	Fedmedian	0.836	0.891	0.837	0.896
	Fedadam	1.001	0.897	0.999	0.998
	Fednadamn (proposed)	<u>1.068</u>	<u>0.989</u>	<u>1.034</u>	<u>1.084</u>

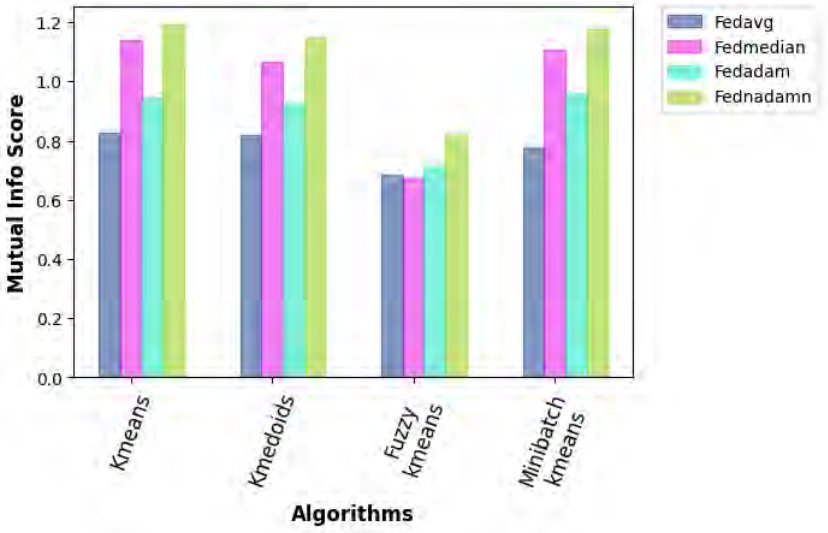


Figure 4.9: Comparing Mutual Info scores for different algorithms using different strategy

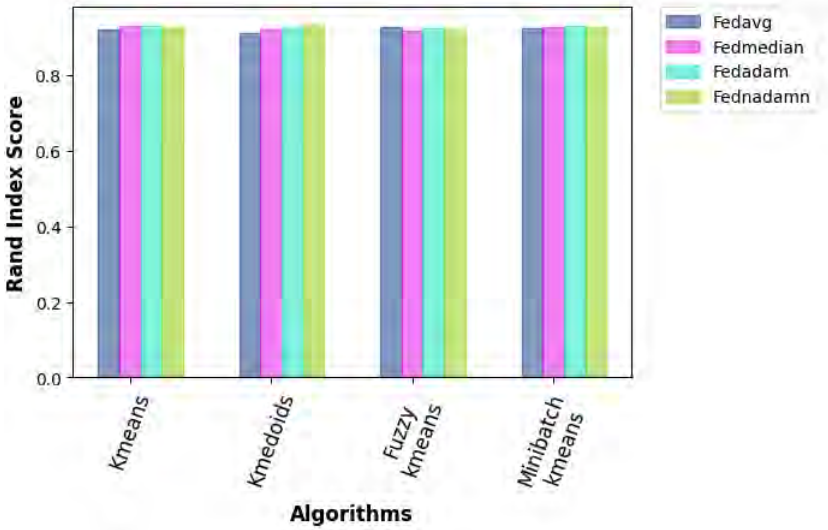


Figure 4.10: Comparing Rand Index scores for different algorithms using different strategy

These figures provide visual insights into the comparative effectiveness of different model aggregation techniques in improving clustering performance. This visual representation facilitates the evaluation and selection of the most suitable model aggregation approach for specific clustering tasks, aiding in the optimization of federated learning frameworks for clustering applications.

The second setup makes use of the Vowel Recognition Dataset. In this situation, using our strategy outperformed previous approaches employing fuzzy k means, kmediods, and kmeans. However, based on the V-measure score, the FedAdam method exhibited superior performance in the minibatch k-means scenario compared to the other techniques. This indicates that the FedAdam approach effectively enhanced clustering accuracy and quality in this particular setting. In terms of the Rand

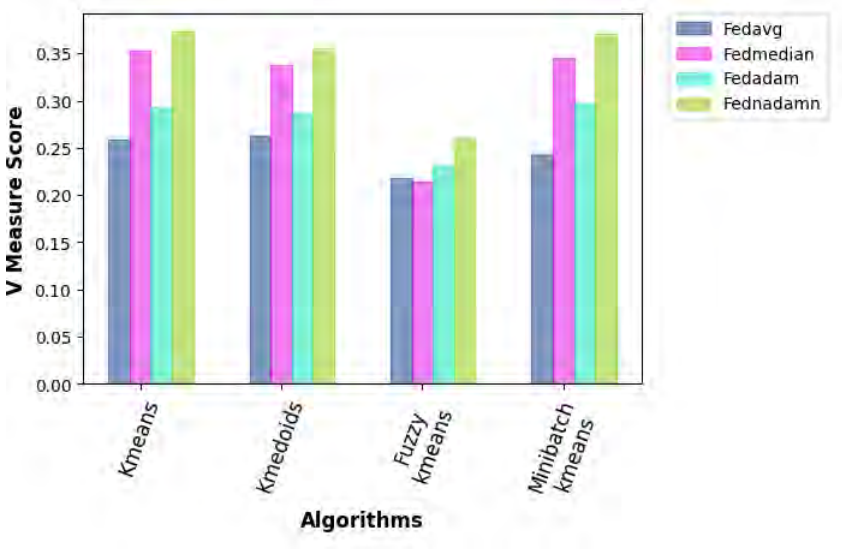


Figure 4.11: Comparing V Measure scores for different algorithms using different strategy

Index score, our approach surpassed previous tactics in this setup utilizing k-means, k-medoids, and fuzzy k-means algorithms. This suggests that our method achieved higher consistency and agreement with the ground truth labels compared to alternative approaches. The superior performance of our approach in capturing cluster similarity and alignment underscores its effectiveness in producing accurate clustering results. Nevertheless, the FedMedian strategy outperformed the other methods in terms of minibatch k means. FednadamN (proposed) has demonstrated success in our evaluation according to Mutual Info Score. Compared to other methods, k-means, k-medoids, and fuzzy k-means achieved favorable results. However, when utilizing the FedMedian technique, minibatch k-means outperformed the other methods. These observations suggest that different model aggregation techniques may yield varying results depending on the clustering algorithm and dataset characteristics. The effectiveness of each method should be carefully assessed based on specific clustering objectives and requirements. The performance of clustering algorithms utilizing the FedAvg, FedMedian, FedAdam, and FednadamN (proposed) methods is depicted in Figures (4.12), (4.13), and (4.14) in this setup.
Our analysis indicates that in the third setting utilizing the Protein Localization Dataset, clustering algorithms employing our model aggregation approach outperformed other strategies by a significant margin. This superior performance is evident across multiple evaluation metrics, including the V-measure score, Rand Index score, and Mutual Information score. These results underscore the effectiveness of our model aggregation technique in enhancing clustering accuracy and quality, highlighting its potential as a valuable tool for clustering tasks, particularly in scenarios involving complex datasets like the Protein Localization Dataset. Figure (4.15), (4.16) and (4.17) illustrates the superior results of our technique .
Furthermore, in the fourth setting utilizing Image Segmentation Data, our method exhibited superior performance compared to alternative approaches based on the Mutual Information score, the V-measure score, and the Rand Index score. These findings underscore the effectiveness and reliability of our method in achieving higher clustering accuracy and quality in diverse clustering scenarios. By outperforming

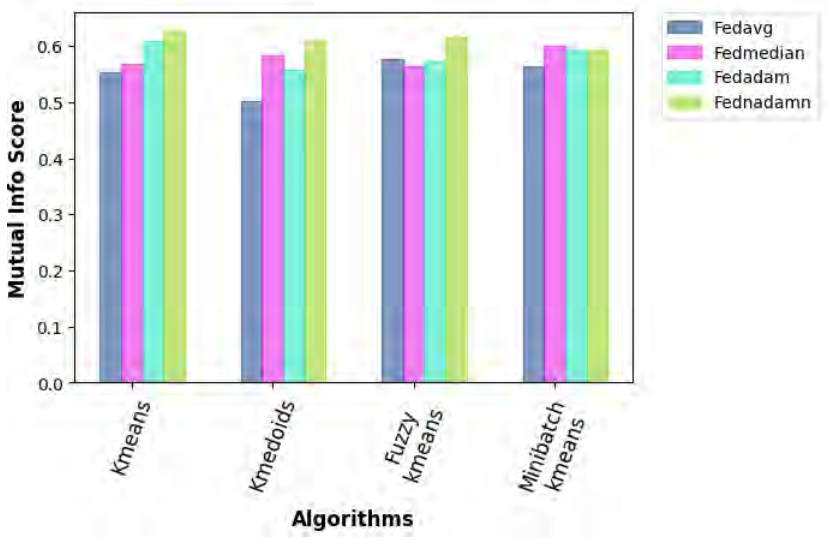


Figure 4.12: Comparing Mutual Info scores for different algorithms using different strategy

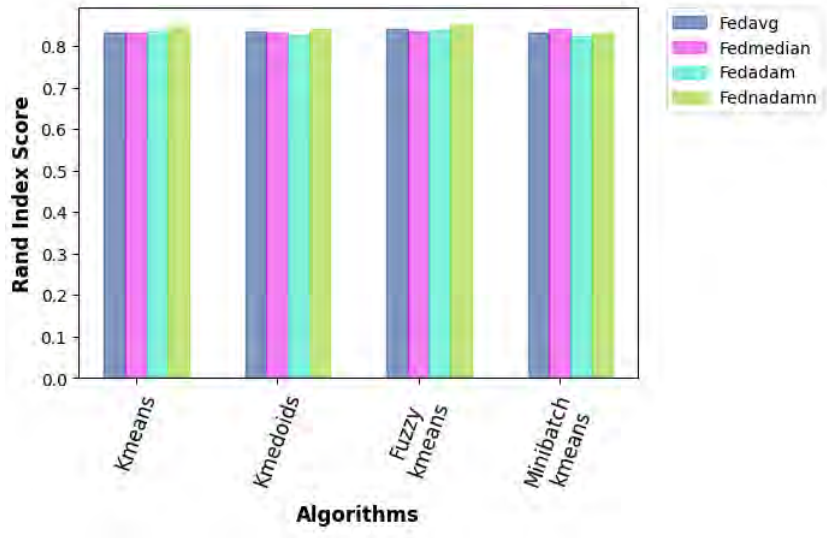


Figure 4.13: Comparing Rand Index scores for different algorithms using different strategy

other methods across multiple evaluation metrics, our approach demonstrates its robustness and suitability for clustering tasks, particularly in complex datasets such as Image Segmentation Data. Figure (4.18),(4.19) and (4.20) show how our method produces improved outcomes.
This comparison suggests that our model performed better in creating clusters that closely resemble the real class labels and in capturing the underlying structure of the data. This result highlights our method's resilience and efficacy in the specified scenario.The federated learning method used in FedAvg entails calculating the weighted average of local model weights among dispersed clients. The contribution of each device is weighted according to the size of its dataset or other predetermined parameters. Conversely, FedMedian adopts a different strategy by taking the median of the local model weights into account. The federated learning procedure

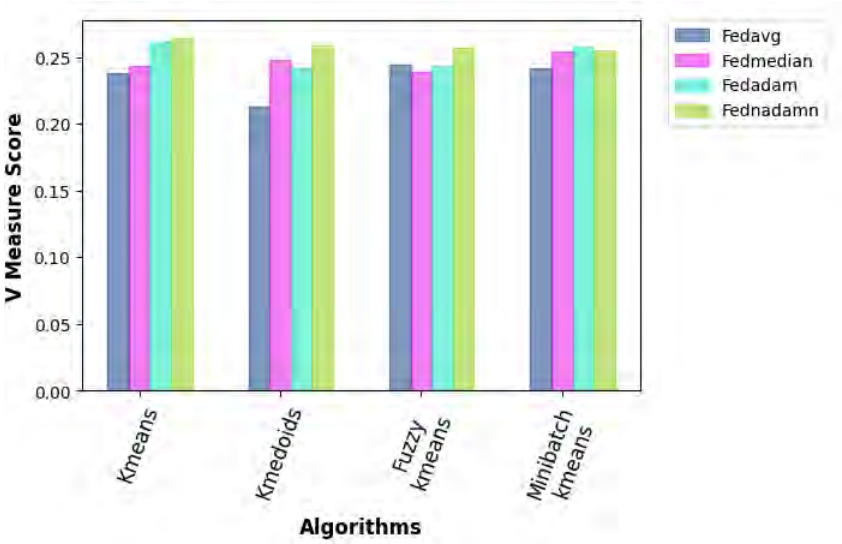


Figure 4.14: Comparing V Measure scores for different algorithms using different strategy

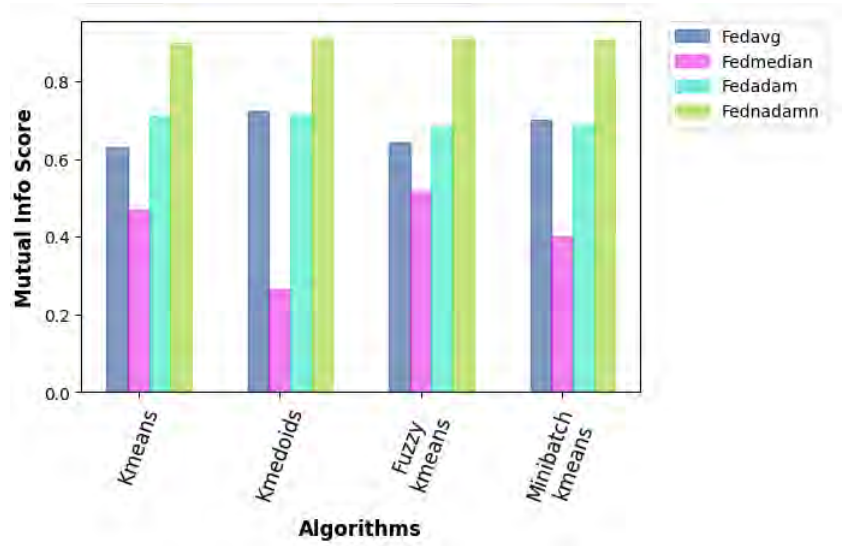


Figure 4.15: Comparing Mutual Info scores for different algorithms using different strategy

in FedAdam is done in two steps. It first calculates the average of the dispersed devices' local model weights. The calculated average is then used to update the global model via the Adam optimization process. This tactic combines the idea of model averaging with the Adam algorithm's capacity for optimization. These three approaches take into account the mean or average, which is why the global server's autoencoder's latent space was unable to adequately capture the data's underlying structure. Our unique Fednadamn approach takes into account each client separately rather than averaging or medianting the data, which enables it to efficiently extract the fundamental structure of the data and eliminate extraneous or noisy aspects. In several datasets, our model demonstrated a statistical advantage over these commonly-used methodologies, indicating its efficacy in attaining superior convergence and performance in federated learning settings.

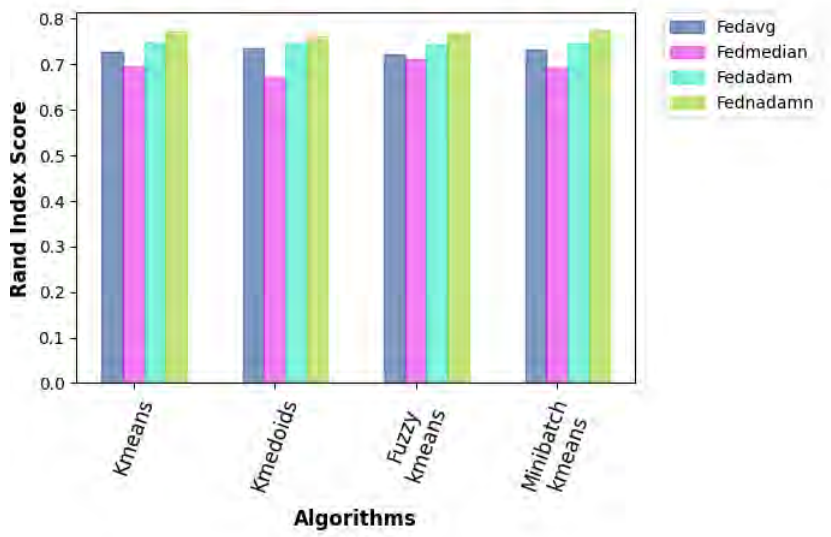


Figure 4.16: Comparing Rand Index scores for different algorithms using different strategy

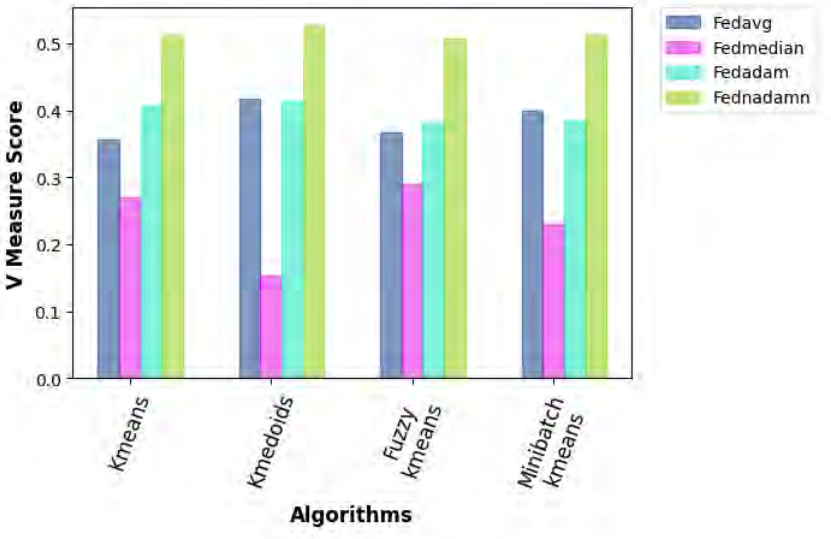


Figure 4.17: Comparing V Measure scores for different algorithms using different strategy

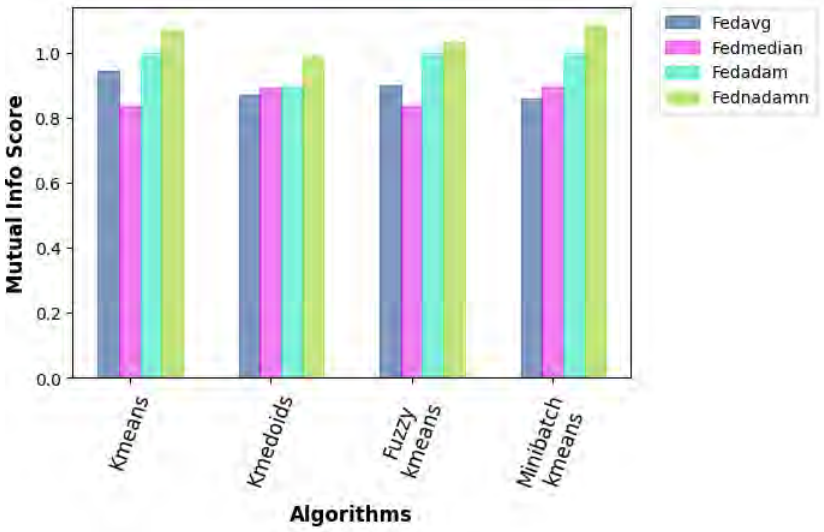


Figure 4.18: Comparing Mutual Info scores for different algorithms using different strategy

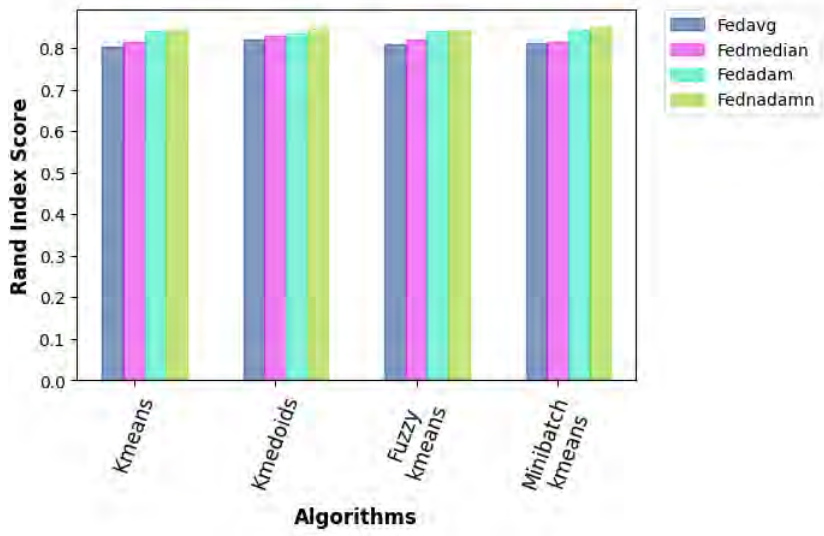


Figure 4.19: Comparing Rand Index scores for different algorithms using different strategy

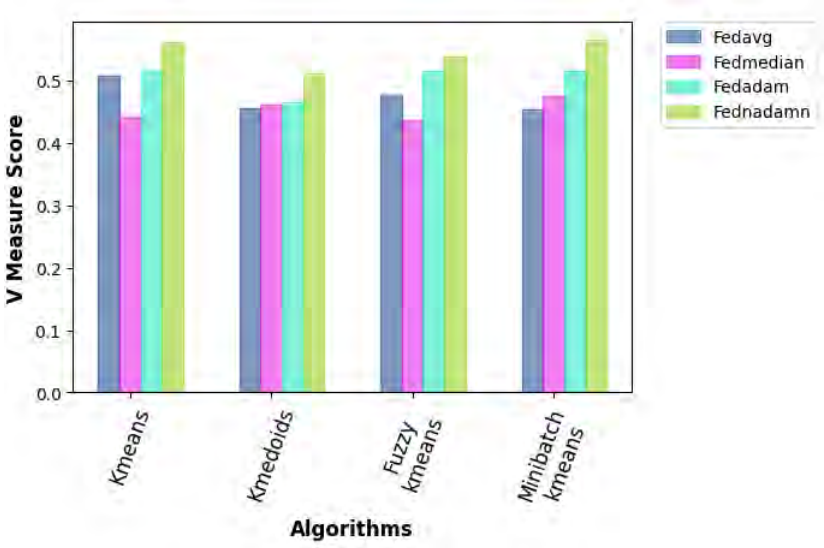


Figure 4.20: Comparing V Measure scores for different algorithms using different strategy

Chapter 5

Conclusion

Clustering is critical to many real-world applications across a wide range of industries. Marketing professionals use clustering as a method to identify customer segments with comparable buying trends. This information may be useful for targeted marketing campaigns, customized recommendations, and optimizing product offers to certain customer segments. In medical research, clustering can be used to find patient groups based on genetic, clinical, or demographic data. This is in favor of customized medicine, which tailors treatments to specific patient populations in an effort to enhance healthcare outcomes. Clustering is a technique used in social network analysis to identify communities or groups of individuals with similar connections or interests. Targeted advertising, social dynamics study, and trend prediction are all done with this data. Every day, the amount of digital data increases. Sophisticated techniques for organizing, storing, and evaluating data are increasingly necessary due to its growth. Security and privacy are other important considerations. It's getting more and harder to collect data in a centralized system and use clustering for different applications. In this study, Proposed parameter aggregation mechanism in conjunction with a generative model is used to describe a decentralized clustering technique.Clustering issue inside a heterogeneous federated learning environment is tackled, where clients may have access to data from many clusters. When applied to an unstructured real-world dataset, our method outperforms the baselines. To extract latent representations, An autoencoder structure is used that is simple to understand yet highly efficient. Utilizing autoencoders' built-in features, the procedure and guarantee simplicity is expedited without sacrificing effectiveness or performance. It is intended to investigate more complex autoencoder topologies as further advancements pursed in latent representation extraction. Although the simplicity of our current technique is emphasized, It is acknowledged that adopting more complicated designs has the potential to improve speed and feature extraction. A fascinating avenue for more investigation would be to combine our suggested approach with other generative models. Both the diversity and quality of learnt representations can be improved by fusing the advantages of many methods.

Bibliography

[1] Nickson M Karie, Nor Masri Sahri, and Paul Haskell-Dowland. "IoT threat detection advances, challenges and future directions". In: *2020 workshop on emerging technologies for security in IoT (ETSecIoT)*. IEEE. 2020, pp. 22–29.

[2] Pascal Vincent et al. "Extracting and composing robust features with denoising autoencoders". In: *Proceedings of the 25th international conference on Machine learning*. 2008, pp. 1096–1103.

[3] Carole Cadwalladr and Emma Graham-Harrison. "Revealed: 50 million Facebook profiles harvested for Cambridge Analytica in major data breach". In: *The guardian* 17.1 (2018), p. 22.

[4] Brendan McMahan et al. "Communication-efficient learning of deep networks from decentralized data". In: *Artificial intelligence and statistics*. PMLR. 2017, pp. 1273–1282.

[5] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. "Unsupervised representation learning by predicting image rotations". In: *arXiv preprint arXiv:1803.07728* (2018).

[6] Alec Radford, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks". In: *arXiv preprint arXiv:1511.06434* (2015).

[7] Sungwon Park et al. "Improving unsupervised image clustering with robust learning". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 12278–12287.

[8] Yi-Zhan Xu et al. "A comprehensive and adversarial approach to self-supervised representation learning". In: *2020 IEEE International Conference on Big Data (Big Data)*. IEEE. 2020, pp. 709–717.

[9] Dong Yin et al. "Byzantine-robust distributed learning: Towards optimal statistical rates". In: *International Conference on Machine Learning*. Pmlr. 2018, pp. 5650–5659.

[10] Sudipto Mukherjee et al. "Clustergan: Latent space clustering in generative adversarial networks". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 4610–4617.

[11] Mathilde Caron et al. "Deep clustering for unsupervised learning of visual features". In: *Proceedings of the European conference on computer vision (ECCV)*. 2018, pp. 132–149.

[12] Jichan Chung, Kangwook Lee, and Kannan Ramchandran. "Federated unsupervised clustering with generative models". In: *AAAI 2022 International Workshop on Trustable, Verifiable and Auditable Federated Learning*. Vol. 4. 2022.

[13] Fengda Zhang et al. "Federated unsupervised representation learning". In: *Frontiers of Information Technology & Electronic Engineering* 24.8 (2023), pp. 1181–1193.

[14] Chen Zhao et al. "FedUSC: Collaborative Unsupervised Representation Learning from Decentralized Data for Internet of Things". In: *IEEE Internet of Things Journal* (2023).

[15] Bram van Berlo, Aaqib Saeed, and Tanir Ozcelebi. "Towards federated unsupervised representation learning". In: *Proceedings of the third ACM international workshop on edge systems, analytics and networking*. 2020, pp. 31–36.

[16] Zhaomin Wu, Qinbin Li, and Bingsheng He. "Practical vertical federated learning with unsupervised representation learning". In: *IEEE Transactions on Big Data* (2022).

[17] Sungwon Han et al. "Fedx: Unsupervised federated learning with cross knowledge distillation". In: *European Conference on Computer Vision*. Springer. 2022, pp. 691–707.

[18] Mykola Servetnyk, Carrson C Fung, and Zhu Han. "Unsupervised federated learning for unbalanced data". In: *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE. 2020, pp. 1–6.

[19] Ekdeep Singh Lubana et al. "Orchestra: Unsupervised federated learning via globally consistent clustering". In: *arXiv preprint arXiv:2205.11506* (2022).

[20] Tiandi Ye et al. "UPFL: Unsupervised Personalized Federated Learning towards New Clients". In: *arXiv preprint arXiv:2307.15994* (2023).

[21] Yeongwoo Kim et al. "Dynamic clustering in federated learning". In: *ICC 2021-IEEE International Conference on Communications*. IEEE. 2021, pp. 1–6.

[22] Jianfei Zhang and Shuaishuai Lv. "Fedlabcluster: A clustered federated learning algorithm based on data sample label". In: *2021 International Conference on Electronic Information Engineering and Computer Science (EIECS)*. IEEE. 2021, pp. 423–428.

[23] Lucas de Sousa Pacheco et al. "Federated user clustering for non-iid federated learning". In: *Electronic Communications of the EASST* 80 (2021).

[24] Avishek Ghosh et al. "An efficient framework for clustered federated learning". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 19586–19597.

[25] Chengxi Li, Gang Li, and Pramod K Varshney. "Federated learning with soft clustering". In: *IEEE Internet of Things Journal* 9.10 (2021), pp. 7773–7782.

[26] Junshen Su, Xijun Wang, and Xiang Chen. "An Efficient Client Clustering Algorithm for Clustered Federated Learning". In: *2022 IEEE/CIC International Conference on Communications in China (ICCC)*. IEEE. 2022, pp. 902–907.

[27] Laizhong Cui et al. “ClusterGrad: Adaptive gradient compression by clustering in federated learning”. In: *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE. 2020, pp. 1–7.

[28] Feng Shi et al. “Review of artificial intelligence techniques in imaging data acquisition, segmentation, and diagnosis for COVID-19”. In: *IEEE reviews in biomedical engineering* 14 (2020), pp. 4–15.

[29] Sashank Reddi et al. “Adaptive federated optimization”. In: *arXiv preprint arXiv:2003.00295* (2020).

[30] Peter W Frey and David J Slate. “Letter recognition using Holland-style adaptive classifiers”. In: *Machine learning* 6 (1991), pp. 161–182.

[31] Peter D Turney. “Robust classification with context-sensitive features”. In: *arXiv preprint cs/0212041* (2002).

[32] Kenta Nakai and Minoru Kanehisa. “Expert system for predicting protein localization sites in gram-negative bacteria”. In: *Proteins: Structure, Function, and Bioinformatics* 11.2 (1991), pp. 95–110.

[33] PK Srimani and Mrs Shanthi Mahesh. “A comparative study of different classifiers on image-segmentation data”. In: (1990).