TACKLING BIAS, PRIVACY, AND SCARCITY CHALLENGES IN HEALTH DATA ANALYTICS

by

Shuwen Wang

A Dissertation Submitted to the Faculty of

The College of Engineering and Computer Science

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

Florida Atlantic University

Boca Raton, FL

December 2023

TACKLING BIAS, PRIVACY AND SCARCITY IN HEALTH DATA

ANALYTICS

by

Shuwen Wang

This dissertation was prepared under the direction of the candidate's dissertation advisor, Dr. Xingquan Zhu, Department of Electrical Engineering and Computer Science, and has been approved by all members of the supervisory committee. It was submitted to the faculty of the College of Engineering and Computer Science and was accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

SUPERVISORY COMMITTEE:

Xingquan Zhu, PhD.
Dissertation Advisor

Ankur Agarwal, PhD.

Raquel Assis, PhD.

Hari Kalva, PhD.
Interim Chair, Department of Electrical Engineering and Computer Science

Oge Marques, PhD.

Xiangnan Zhong, PhD.

Stella Batalama, Ph.D.
Dean, College of Engineering & Computer Science

Robert W. Stackman Jr., Ph.D.
Dean, Graduate College

Date

ACKNOWLEDGEMENTS

I would like to express my deep gratitude to everyone who has contributed to the successful completion of my doctoral dissertation. This journey has been filled with challenges, growth, and moments of inspiration, and I am profoundly thankful for the support and encouragement I have received along the way.

First and foremost, I extend my heartfelt appreciation to my dissertation advisor, Dr. Xingquan Zhu, for his unwavering guidance, patience, and expertise throughout this research endeavor. His mentorship has been instrumental in shaping the direction of my work, and I am truly fortunate to have had the privilege of learning from him.

I am also grateful to the members of my dissertation committee, Dr. Oge Marques, Dr. Raquel Assis, Dr. Ankur Agarwal and Dr. Xiangnan Zhong, for their valuable insights, constructive feedback, and commitment to helping me refine my research. Their diverse perspectives have enriched the quality of this dissertation.

To my colleagues and fellow graduate students, I appreciate the countless discussions, brainstorming sessions, and collaborative efforts that have enriched this dissertation.

Last but not least, I appreciate my beloved families. Your love, understanding, and patience have been my anchor during the challenging moments of this academic pursuit. You have made this journey all the more meaningful.

This research is sponsored by the National Science Foundation under grant Nos. IIS-1763452 and IIS-2302786. Any opinions, findings, and conclusions or recommendations expressed in this research are those of the author and do not necessarily reflect the views of the National Science Foundation.

ABSTRACT

Author:	Shuwen Wang
Title:	Tackling Bias, Privacy, and Scarcity Challenges in Health Data Analytics
Institution:	Florida Atlantic University
Dissertation Advisor:	Dr. Xingquan Zhu
Degree:	Doctor of Philosophy
Year:	2023

Health data analysis has emerged as a critical domain with immense potential to revolutionize healthcare delivery, disease management, and medical research. However, it is confronted by formidable challenges, including sample bias, data privacy concerns, and the cost and scarcity of labeled data. These challenges collectively impede the development of accurate and robust machine learning models for various healthcare applications, from disease diagnosis to treatment recommendations.

Sample bias and specificity refer to the inherent challenges in working with health datasets that may not be representative of the broader population or may exhibit disparities in their distributions. These biases can significantly impact the generalizability and effectiveness of machine learning models in healthcare, potentially leading to suboptimal outcomes for certain patient groups. Data privacy and locality are paramount concerns in the era of digital health records and wearable devices. The need to protect sensitive patient information while still extracting valuable insights from these data sources poses a delicate balancing act. Moreover, the geographic and jurisdictional differences in data regulations further complicate the use of health data in a global context. Label cost and scarcity pertain to the often labor-intensive and

expensive process of obtaining ground-truth labels for supervised learning tasks in healthcare. The limited availability of labeled data can hinder the development and deployment of machine learning models, particularly in specialized medical domains.

This dissertation mainly focuses on health data analysis and explores approaches to tackle the above challenges. More specifically, the following three problems will be studied from different perspective: (1) Sample bias and specificity in health data. (2) Data privacy and locality in health data. (3) Label cost and scarcity in health data.

In summary, our major contributions in this dissertation are demonstrated in the following aspects:

- To tackle data bias and specificity problem, we first conduct a comprehensive exploration of the twin phenomena in health data with National Readmission Databases (NRD). We first propose to use imbalanced learning for 30-day hospital readmission prediction. The main goal is to predict, at the time of a hospital discharge, whether the patient may return in 30 days or not in the future. We created a set of features, using simple patient demographics, ICD-10 clinical modification (CM), and Clinical Classification Software Refined (CCSR) conversion, to represent each hospital visit. Because patient readmission is only a small portion of all patient visits, the machine learning task is severely challenged by the imbalanced class distributions. To solve the challenge, we used random under sampling (RUS) to create different copies of balanced sample sets. Ensemble classifiers were trained from balanced sample sets to build classifiers for readmission prediction. conversion.

  Secondly, we carry out systematic studies to understand data statistics for United States nationwide hospital admission, and further designs a machine learning framework for disease-specific 30-day hospital readmission prediction. We identified factors related to three key party of the hospital remissions: pa-

tient, disease, and hospitals, and reported national scale hospital admission statistic. Based on the data statistics, we created 526 features with five major types, including demographics features, admission and discharge features, clinical features, disease features, and hospital features. We collected six disease specific readmission datasets, which reflect the top six leading diseases of death. By using random under sampling and ensemble learning, combined with soft *vs.* hard voting and four types of machine learning methods, including gradient boosting, decision tree, logistic regress, and random forests, our experiments validate three major type of settings: (1) hard voting *vs.* soft voting, (2) random under sampling, and (3) disease specific readmission prediction..

- We introduce a unique federated learning approach designed specifically to address the data privacy and locality of health data. We propose a dynamic node matching method for federated learning. We argued that neural networks are inherently non-transparent and unstable, and the same network structure may end up with very different weight values, even with the same training data and same parameter settings. Traditionally, existing methods, such as FedAvg, force neurons across sites to be matched with predefined order, and use fixed matching nodes during the FL learning process. Alternatively, we proposed a dynamic node alignment, FedDNA, approach which dynamically finds matching nodes across sites, and uses matched nodes to calculate weight for FL learning. FedDNA represents each neuron as a vector, using their weight values, and calculate distances between neurons to find matching nodes. Meanwhile, because finding marching nodes are computationally expensive, we proposed a minimum spanning tree (MST) based approach to speed up the matching, with matched nodes across all sites being linked by using an MST tree. So the matching process is simply the MST tree growing process.

- In order to address the labeling challenges in health care data analysis, we pro-

pose a locality-customized GSA Federated Active Learning (LG-FAL) method for federated active learning. LG-FAL combines locality-customized active learning and Gravitational Search Algorithm (GSA) in a collaborative and effective way. In locality-customized active learning, both the local model as well as the global model are taken into consideration when annotating local samples, in which each data's overall uncertainty is a combination of both the local model's prediction entropy and the global model's prediction entropy. In GSA federated learning, global model parameter aggregations are achieved by GSA which is empowered with higher adaptability with a set of parameters to allow clients to move freely towards areas of high fitness calculated based on their masses (accuracy).

TACKLING BIAS, PRIVACY, AND SCARCITY CHALLENGES IN HEALTH DATA ANALYTICS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

In recent years, the field of health data analytics has emerged as a crucial and rapidly evolving domain in healthcare research and practice. This surge of interest in health data analytics can be attributed to several key factors, each of which underscores its growing significance in the healthcare landscape. Firstly, there has been a remarkable increase in the popularity and accessibility of electronic health data. The digitization of health records, diagnostic tests, patient histories, and treatment outcomes has transformed healthcare into a data-rich industry. This shift has been facilitated by the widespread adoption of electronic health record (EHR) systems in healthcare institutions worldwide [28, 56, 104]. The wealth of digital health data generated on a daily basis provides a unique opportunity to harness the power of data analytics for improving patient care, optimizing healthcare operations, and advancing medical research.

Secondly, the escalating costs of healthcare systems, particularly in the United States and across the globe, have spurred a critical need for more efficient, cost-effective, and data-driven approaches to healthcare delivery. The unsustainable rise in healthcare expenditures, driven by factors such as an aging population, chronic diseases, and expensive medical technologies, has necessitated innovative strategies to reduce costs while maintaining or even enhancing the quality of care. Health data analytics offers promising solutions to address these challenges by enabling predictive modeling, resource allocation optimization, and the identification of cost-saving opportunities [64, 86, 91]. Thirdly, the advent of advanced techniques and systems empowered by data science and analytics has ushered in a new era of possibilities for

healthcare. Numerous companies, both established healthcare providers and innovative startups, have heavily invested in healthcare data science technologies. These investments have given rise to cutting-edge tools and platforms for health analytics, ranging from artificial intelligence and machine learning algorithms to data-driven decision support systems, as shown in Fig. 1.1. This flourishing ecosystem of data-driven solutions holds immense potential to revolutionize healthcare delivery, enhance patient outcomes, and drive evidence-based medical advancements.

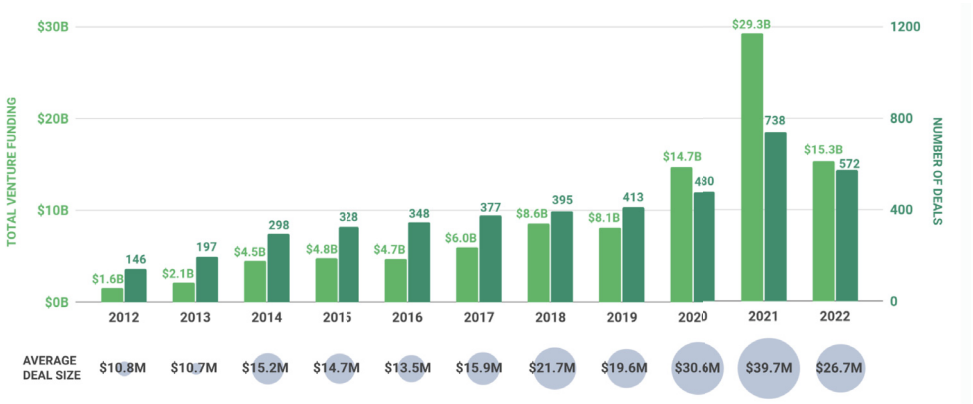


Figure 1.1: U.S. Digital Health Funding and Deal Size [18]

The successful analysis and utilization of healthcare data are contingent upon the integrity and quality of the data itself. Data bias, stemming from factors such as uneven data collection, skewed representation, and systemic inequalities in healthcare access, can introduce distortions into the analysis, leading to inaccurate predictions and biased decision-making [10, 74, 102]. Simultaneously, the sensitive nature of healthcare data, laden with personal and confidential information, necessitates stringent privacy safeguards to protect patient rights. The imperative to strike a delicate balance between data utility and individual privacy adds complexity to the already intricate landscape of healthcare data analysis [17, 70, 70, 79]. Furthermore, data scarcity remains a significant hurdle in healthcare analytics. While the potential for data-driven advancements is vast, healthcare datasets are often limited in size, diversity, and quality. This scarcity poses a substantial obstacle to the development

and validation of robust predictive models and data-driven interventions.

Learning tasks regarding with data bias, data privacy as well as data scarcity in health care data analysis are fundamental, but challenging, which have also received continuous attention in the research field. This thesis focuses on health data analysis and explores approaches to tackle the above challenges. Many researches also proposed to use a variety of prediction models, such as support vector machines [21] and neural networks [13] to enable better health data analysis. While numerous existing methods or models excel in this area, they may overlook at least one of the following challenges or scenarios.

1.1 SAMPLE BIAS AND SPECIFICITY IN HEALTH DATA

Sample bias and specificity within health data have emerged as pivotal concerns that demand rigorous investigation [87]. Sample bias refers to a phenomenon where datasets used to train a predictive model have a biased class distribution. In many health care cases, one type of samples (*i.e.* positive class) are significantly less than other types of samples. This is partially caused by the reality that disease samples are only a small percentage of the whole population, and naturally results in the class imbalance. Learning models with imbalanced class distributions is defined challenge, because most algorithms are affected by frequency bias and pay more attention to majority class samples [43]. Sample bias tends to force the classifier to classify all samples as normal, in order to satisfy the defined objective function, such as minimizing the classification errors [16]. Common solutions are to re-balance samples in different classes, by manipulating data populations (sampling approaches) or classification outcomes (cost-sensitive learning). Sample specificity in health data, on the other hand, is associated to the sample distributions (or independent variables). At population level, data analysis might be collected from a local/regional hospital, where the demographics of the patient body naturally introduce bias. At individual

level, when collecting data for each patient, the hospital visits used to characterize the patient may also introduce bias. At the ministration level, regularization also impose restriction for data sharing across hospitals, making it difficult to learn good models from local data [68, 96]. The landscape of health data, both in its richness and complexity, is characterized by the interplay of various factors such as patient demographics, healthcare access, geographic disparities, and clinical practices. Consequently, the inherent biases and limitations within these datasets can significantly influence the validity and generalizability of findings derived from them.

While many methods exist to tackle the above challenges in health data analysis, some major questions in the field still need to be better addressed. (1) First of all, the the positive class are a small portion of the total hospital visits, representing a severe class imbalance problem for learning; (2) Secondly, for each type of disease, their causes are different, leading to variance in disease characteristics. Such distinctions can further result in patient, in-hospital treatment and discharge gap, reflected by unique patient features for each disease. How will demographic information, such as gender, age, geographic, impact on disease prediction? Is the prevalence of certain diseases or conditions vary widely across populations? Many methods are available for prediction, but no existing research has provided clear answer to the above questions. (3) Thirdly, health data analysis is a compound outcome of many factors, including patient, disease, care providers *etc*. Datasets that do not account for these factors may fail to capture the full picture of health disparities. (4) Fourthly, health data analysis is essentially data driven, where features and samples are the key to ensure model performance. While many methods have been using a wide variety of patient treatment data, such as patient blood tests, nutritional factors [31], treatment *etc*, the data privacy and the Health Insurance Portability and Accountability Act (HIPAA) [22] limit sensitive features to be used in general analysis setting.

Motivated by the above challenges, in this research, we conduct a comprehensive

exploration of the twin phenomena of sample bias and specificity in health data . We first design an imbalanced learning strategy to predict patient readmission possibility with National Readmission Databases (NRD). We use a random sampling approach to balance the sample distributions in the training set. By implementing random sampling, it helps ensure that our training dataset is representative of the overall population, which is crucial because biased or non-representative samples can lead to poor model performance. By randomly selecting data points, we are able to reduce the risk of introducing bias into our training set. In addition, overfitting can be reduced which occurs when a model learns to perform well on the training data but fails to generalize to unseen data. We create features from patient hospital visit, by combining patient demographic information, ICD-10 clinical modification (CM) and procedure codes (PCS), and Clinical Classification Software Refined (CCSR) conversion. Instead of directly using ICD-10- CM/PCS code to characterize patients, we convert each patient's visit to CCSR code space with a smaller feature space. To better capture the sample bias and specificity problem, we use National Readmission Databases (NRD), with over 15 million hospital visits, as our testbed, and report national scale hospital admission statistics, including readmission rate differences with respect to different demographic and hospital factors, such as gender, age, payment type, hospital profile, and disease types. After that, we create six disease specific readmission tasks for Cancer, Heart disease, Chronic obstructive pulmonary disease (COPD), Diabetes, Pneumonia, and Stroke. Random under sampling and ensemble learning, including hard-voting and soft-voting, are used to train models for disease-specific readmission prediction.

1.2 DATA PRIVACY AND LOCALITY IN HEALTH DATA

As the healthcare industry reaps the benefits of data revolution from the digitization of health records, the proliferation of wearable devices, and the emergence of

telemedicine *etc*, it simultaneously faces profound challenges in safeguarding the privacy, confidentiality and data locality of sensitive health information. Data privacy and locality in health data refers to the protection of individuals' personal health information from unauthorized access, disclosure, or misuse. It encompasses the principles and practices that ensure the confidentiality, integrity, and security of sensitive health-related data. This type of data includes but not limited to personal identifiable information, medical records, biometric data. The adoption of electronic health records enables health care professionals to disseminate knowledge across all sectors of health care, which in turn helps to reduce medical errors and improve patient care and satisfaction. However, as mentioned previously, adequate medical data sets are difficult to obtain. However, in order to capture the subtle relationships between disease patterns, socioeconomic and genetic factors, and complex and rare cases, exposing the model to different cases is critical. FL is able to address this issue by enabling the distributed training of machine learning models using remotely hosted datasets without the need to accumulate data and therefore compromise the data privacy [10, 17, 70, 70, 74, 79, 102].

FL enables devices to collaboratively learn shared predictive models while keeping all training data on-device, decoupling the power of machine learning from the need to store data in the cloud. This goes beyond using native models to make predictions on mobile devices and also brings model training to the device. Recently, other weight aggregation methods have also been proposed in FL. For example, anomaly score of each client is taken into consideration to detect abnormal client behavior, thus, clients will not contribute equally when global model updates the weight values, the majority of those novel methods are still based on FedAvg [36, 59]. Even though this method is widely used and has been proved with good prediction performance [34, 90], due to the nature of hidden layers in deep learning neural networks, we can clearly observe that this method manually forces weight aggregations between neurons located at

the exact same location (*i.e.*, same layer and same node index) of two networks. However, when training two same-structured deep learning networks N_A and N_B, even they are given the same input, neurons at the same location of the two networks do not always give the same update. In other words, certain property of the input (or the same instance) may trigger the most significant activation to the i-th node of N_A, but same instance may triger the most significant activation to the j-th node of N_B. Meaning that same instance responds differently for the same lactation nodes between two networks.

In our research, we present a novel federated learning method specifically tailored to the intricacies of health data. We delineate the pressing need for innovative solutions in health data analytics, underscore the limitations of traditional data-sharing approaches, and provide a glimpse into the potential benefits of our proposed novel federated learning method. We aim to design a dynamic node matching method, FedDNA, to aggregate weight values in each round based on a neuron-distance method, in which neuron distances across all the clients are calculated after each client completes training the model parameters with their own data. After that, the closest neurons are matched to calculate their average weight values as new parameter for the global model.

1.3 LABEL COST AND SCARCITY IN HEALTH DATA

To harness the full potential of health data for tasks like disease prediction, drug discovery, and personalized treatment, robust machine learning models are essential. However, these models typically demand vast quantities of labeled data for training. In the field of healthcare, acquiring accurate and high-quality labels is a complex, resource-intensive, and sometimes elusive endeavor. Label cost and scarcity in health data refer to the challenges associated with obtaining and using labeled data for machine learning and data analysis tasks in the healthcare domain. These challenges

are particularly prominent in the field of healthcare due to the unique characteristics of health data and the stringent requirements for data quality and privacy. Label cost refers to the resources, time, and effort required to manually annotate or label data points. In the context of health data, this process often involves healthcare professionals or domain experts reviewing and providing accurate labels for medical records, images, or other health-related information [7, 80, 99]. Labeling can be expensive, as it may require the expertise of qualified professionals and rigorous quality control measures to ensure the accuracy and relevance of data. The cost of labeling can also include considerations related to data privacy and compliance with regulations such as HIPAA (Health Insurance Portability and Accountability Act) in the United States. In addition, label scarcity arises when there is a limited availability of labeled data for a specific healthcare task or research problem. This scarcity can stem from various factors, including the difficulty of obtaining consent from patients for data usage, the need for domain expertise in labeling, or the sheer volume of data required for training machine learning models effectively. In many healthcare applications, labeled data is a precious resource, and there may be insufficient quantities of labeled examples to build robust and accurate models, especially for rare medical conditions or emerging diseases [7, 82, 85, 88].

Recently, Active Learning (AL) has emerged as a machine learning method that can effectively address data annotation workloads [78, 84]. Its main strategy is to iteratively find the most informative data points to annotate. The annotated data are then used as part of the training data in the next iteration. With more and more iterations, the machine learning model's performance can be more and more improved. This strategy has been integrated into federated learning and generated a new paradigm called Federated Active Learning (FAL) [8, 55, 72, 100]. The FAL framework consists of several clients and one central server. Each client holds one labeled dataset and one unlabelled dataset, which can not be shared with others.

The server holds a test dataset that can be shared with all clients. The goal of FAL is to train a globally optimized model at the server by annotating informative data samples at the clients. FAL framework is trained in an iterative manner. In one iteration, each client first trains a local model with annotated data. And then, the local parameters are transmitted to the server. The server synthesizes local parameters into a global model. With more and more iterations, more and more data will be annotated and the global model can be improved. However, in most FAL, local unlabelled samples are annotated by the aggregated global model's parameters [50]. Its global model parameter updating is limited to one method, which is called Federated average (FedAvg) [7, 8, 67, 69].

In order to address the labeling challenges in health care data analysis, we propose a locality-customized GSA federated active learning (LG-FAL) method which strategically selects the most informative data points for labeling, holds the potential to mitigate label cost and scarcity challenges. In this design, the algorithm takes the local model into consideration aside from the global model when annotating. As for model parameters update, global model parameter aggregations are achieved by GSA which draws inspiration from the law of gravity and the interactions between celestial bodies.

1.4 THESIS ORGANIZATION

Fig. 1.2 shows the thesis organization. Three health data analysis problems are studied in this thesis which are elaborated in chapters 3, 4, 5 and 6, respectively. For each research problem, respective data are experimentally studied and experimental results demonstrate our proposed methods.

More specifically, we organize the thesis as follows. Chapter 2 briefly describes the background including related work about health data analysis and preliminary knowledge for our proposed methods. Chapter 3 emphasizes the problem of the sample bias

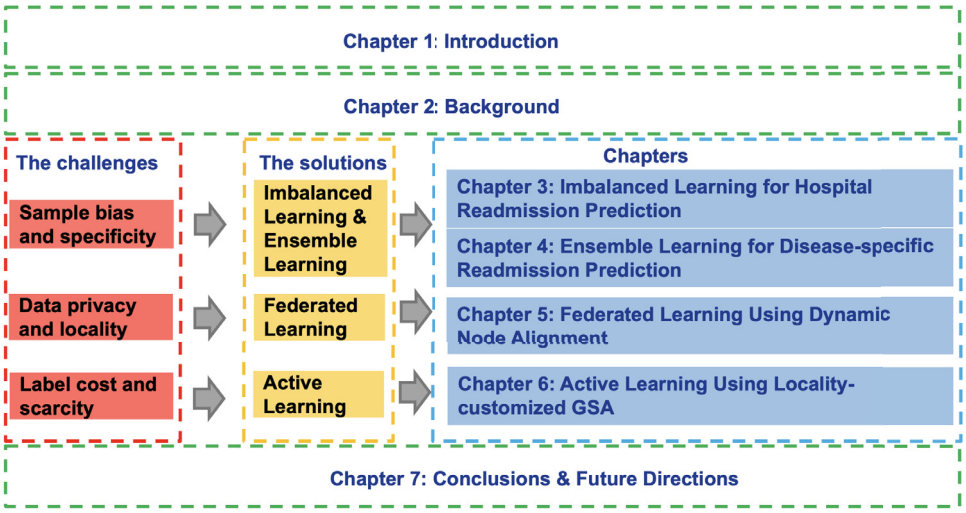


Figure 1.2: The studied learning problems and their organization. 1) Chapter 1 gives a general description of the studied problems with highlighted contributions. 2) Charter 2 introduce relevant backgrounds about related work and preliminary knowledge. 3) Chapter 3 studies the imbalanced learning for hospital readmission prediction problem. 4) Chapter 4 studies the ensemble learning for disease-specific readmission prediction problem. 5) Chapter 5 studies federated learning using dynamic node alignment . 6) Chapter 6 studies active learning using locality-customized GSA. 7) Finally, chapter 7 concludes the thesis and discusses future directions.

problem for hospital readmission prediction, and proposes a imbalanced learning algorithm as a solution. Chapter 4 emphasizes the sample specificity, formulates the nationwide hospital disease-specific readmission prediction problem and proposes an ensemble learning approach. In chapter 5, we focus on the data privacy and locality challenge and propose a novel federated learning node alignment approach. In chapter 6, we consider the data label cost and scarcity and propose a localized federated learning method. Finally, we conclude the contribution and discuss future directions in charter 7.

Chapter 2: Background

This chapter first presents related work about imbalanced learning for hospital readmission prediction. Then, we also present related work about hospital disease-specific 30-day readmission prediction. Finally, preliminary knowledge are briefly described about federated learning using dynamic node alignment and federated active learning respectively.

Chapter 3: Imbalanced Learning for Hospital Readmission Prediction

In this chapter, we propose to use imbalanced learning for hospital readmission prediction. The goal is to predict whether a patient, based on his/her current hospital visit records, is likely going to be re-admitted or not within 30-days after being discharged from the current hospital visit. We design an imbalanced learning strategy to create features from patient hospital visit, by combining patient demographic information, ICD-10 clinical modification (CM) and procedure codes (PCS), and Clinical Classification Software Refined (CCSR) conversion. Instead of directly using ICD-10-CM/PCS code to characterize patients, we convert each patient's visit to CCSR code space with a smaller feature space. By using random sampling approach to balance the sample distributions in the training set, our method achieves good performance to predict patient readmission.

Chapter 4: Ensemble Learning for Disease-specific Readmission Prediction

In this chapter, we use National Readmission Databases (NRD), with over 15 million hospital visits, as our testbed, and report national scale hospital admission statistics, including readmission rate differences with respect to different demographic and hospital factors, such as gender, age, payment type, hospital profile, and disease types. After that, we create six disease specific readmission tasks for Cancer, Heart disease, Chronic obstructive pulmonary disease (COPD), Diabetes, Pneumonia, and Stroke. Random under sampling and ensemble learning, including hard-voting and

soft-voting, are used to train models for disease-specific readmission prediction.

Chapter 5: Federated Learning Using Dynamic Node Alignment

In this chapter, we aim to design a dynamic node matching method, FedDNA, to aggregate weight values in each round based on a neuron-distance method, in which neuron distances across all the clients are calculated after each client completes training the model parameters with their own data. After that, the closest neurons are matched to calculate their average weight values as new parameter for the global model. The results show that dynamic node matching provides much smaller weight variance across all nodes of different networks. The advantage of reducing variance is that it allows nodes with similar behaviors to be aggregated for weight averaging. This potentially results in stable and improved federated learning performance.

Chapter 6: Active Learning Using Locality-customized GSA

In this chapter, we propose a locality-customized GSA federated active learning (LG-FAL) method. We propose a new annotating strategy that considers both local and global optimization. By doing so, the localization of samples and models can be considered. We propose to update the global model parameters with GSA, in which the model is updated in a more interactive and adaptable way. We design extensive experiments to validate the proposed methods with different parameter settings and comparisons.

Chapter 7: Conclusion and Future Directions

In this chapter, we summary our contributions for health care data analysis and their applications. We also discuss some future research directions.

1.5 DISSERTATION CONTRIBUTION

In summary, our major contributions in this dissertation are demonstrated in the following aspects:

- To tackle data bias and specificity problem, we first conduct a comprehensive exploration of the twin phenomena in health data with National Readmission Databases (NRD). We first propose to use imbalanced learning for 30-day hospital readmission prediction. The main goal is to predict, at the time of a hospital discharge, whether the patient may return in 30 days or not in the future. We created a set of features, using simple patient demographics, ICD-10 clinical modification (CM), and Clinical Classification Software Refined (CCSR) conversion, to represent each hospital visit. Because patient readmission is only a small portion of all patient visits, the machine learning task is severely challenged by the imbalanced class distributions. To solve the challenge, we used random under sampling (RUS) to create different copies of balanced sample sets. Ensemble classifiers were trained from balanced sample sets to build classifiers for readmission prediction. conversion.

  Secondly, we carry out systematic studies to understand data statistics for United States nationwide hospital admission, and further designs a machine learning framework for disease-specific 30-day hospital readmission prediction. We identified factors related to three key party of the hospital remissions: patient, disease, and hospitals, and reported national scale hospital admission statistic. Based on the data statistics, we created 526 features with five major types, including demographics features, admission and discharge features, clinical features, disease features, and hospital features. We collected six disease specific readmission datasets, which reflect the top six leading diseases of death. By using random under sampling and ensemble learning, combined with

soft *vs.* hard voting and four types of machine learning methods, including gradient boosting, decision tree, logistic regress, and random forests, our experiments validate three major type of settings: (1) hard voting *vs.* soft voting, (2) random under sampling, and (3) disease specific readmission prediction..

- We introduce a unique federated learning approach designed specifically to address the data privacy and locality of health data. We propose a dynamic node matching method for federated learning. We argued that neural networks are inherently non-transparent and unstable, and the same network structure may end up with very different weight values, even with the same training data and same parameter settings. Traditionally, existing methods, such as FedAvg, force neurons across sites to be matched with predefined order, and use fixed matching nodes during the FL learning process. Alternatively, we proposed a dynamic node alignment, FedDNA, approach which dynamically finds matching nodes across sites, and uses matched nodes to calculate weight for FL learning. FedDNA represents each neuron as a vector, using their weight values, and calculate distances between neurons to find matching nodes. Meanwhile, because finding marching nodes are computationally expensive, we proposed a minimum spanning tree (MST) based approach to speed up the matching, with matched nodes across all sites being linked by using an MST tree. So the matching process is simply the MST tree growing process.

- In order to address the labeling challenges in health care data analysis, we propose a locality-customized GSA Federated Active Learning (LG-FAL) method for federated active learning. LG-FAL combines locality-customized active learning and Gravitational Search Algorithm (GSA) in a collaborative and effective way. In locality-customized active learning, both the local model as well as the global model are taken into consideration when annotating local samples, in which each data's overall uncertainty is a combination of both the local

model's prediction entropy and the global model's prediction entropy. In GSA federated learning, global model parameter aggregations are achieved by GSA which is empowered with higher adaptability with a set of parameters to allow clients to move freely towards areas of high fitness calculated based on their masses (accuracy).

The following eight papers have been published & completed in relation to the dissertation study:

1. Shuwen Wang, Magdalyn E. Elkin, and Xingquan Zhu. "Imbalanced learning for hospital readmission prediction using national readmission database." In 2020 IEEE International Conference on Knowledge Graph (ICKG), pp. 116-122. IEEE, 2020.

2. Shuwen Wang, and Xingquan Zhu. "Nationwide hospital admission data statistics and disease-specific 30-day readmission prediction." Health Information Science and Systems 10, no. 1 (2022): 25.

3. Shuwen Wang, and Xingquan Zhu. "FedDNA: Federated learning using dynamic node alignment." Plos one 18, no. 7 (2023): e0288157.

4. Shuwen Wang, and Xingquan Zhu. "LG-FAL: Locality-customized GSA Federated Active Learning." Under Review.

5. Shuwen Wang, and Xingquan Zhu. "Predictive modeling of hospital readmission: challenges and solutions." IEEE/ACM Transactions on Computational Biology and Bioinformatics 19, no. 5 (2021): 2975-2995.

6. Shuwen Wang, Xingquan Zhu, Weiping Ding, and Amir Alipour Yengejeh. "Cyberbullying and cyberviolence detection: A triangular user-activity-content view." IEEE/CAA Journal of Automatica Sinica 9, no. 8 (2022): 1384-1405.

7. Man Wu, Shuwen Wang, Shirui Pan, Andrew C. Terentis, John Strasswimmer, and Xingquan Zhu. "Deep learning data augmentation for Raman spectroscopy cancer tissue classification." Scientific reports 11, no. 1 (2021): 23842.

8. Divya Gangwani, Qianxin Liang, Shuwen Wang, and Xingquan Zhu. "An empirical study of deep learning frameworks for melanoma cancer detection using transfer learning and data augmentation." In 2021 IEEE International Conference on Big Knowledge (ICBK), pp. 38-45. IEEE, 2021.

CHAPTER 2
RELATED WORK

This chapter first presents related work about sample bias and specificity in health data. Then, we also present related work about health data privacy and locality. Finally, preliminary knowledge are briefly described about label cost and scarcity for heath data.

2.1 SAMPLE BIAS AND SPECIFICITY IN HEALTH DATA

Recently there have been many models built through machine learning methods to tackle data bias as well as specificity and provide corresponding results and suggestions [42, 45, 103]. Logistic Regression is a popular model in medical prediction fields [87]. In addition, many researches also proposed to use a variety of prediction models, such as support vector machines [21] and neural networks [13]. Sampling approaches change data distributions to balance samples in different groups in order to tackle the data imbalance challenge. Common sampling solutions are to either drop majority class samples, repeat samples from minority class, or create synthetic samples for minority class [16, 43]. A research study [52] using Medical Information Mart for Intensive Care III (MIMIC-III) database [51] shows that, by using undersampling, their model achieves 0.642 AUC score for ICU patient readmission. Another study [9] investigates RUS sampling and five supervised learning methods, decision trees, naive bayes, logistic regression, neural networks, and support vector machines (SVM) for risk modality and hospital readmission prediction. The results show that, overall, neural networks achieve best performance for both risk modality and hospital

readmission prediction. A novel approach is introduced to mitigate bias in electronic health records (EHR) research. The study proposes a cohort design methodology combined with natural language processing techniques to improve the quality and reliability of research conducted using EHR data. By systematically addressing issues related to data bias, missing information, and data quality, this approach aims to enhance the accuracy of insights drawn from EHRs, ultimately contributing to more robust and unbiased healthcare research, which is crucial for informed decision-making and improved patient care [54]. Authors examine the impact of COVID-19 victimization distress and racial bias on the mental health of young adults from diverse racial backgrounds, including American Indian/Alaska Native (AIAN), Asian, Black, and Latinx individuals. The study explores how the experiences of distress related to COVID-19 and racial bias contribute to variations in mental health outcomes within these communities. The findings of this research provide insights into the complex interplay between pandemic-related stressors and racial bias, shedding light on the unique mental health challenges faced by young adults from different racial and ethnic backgrounds in the context of the COVID-19 pandemic [37].

Racial bias in the patient descriptors within electronic health records (EHRs)is investigated [89]. The study focuses on identifying the presence of negative descriptors related to race or ethnicity in these records. The authors analyze a range of patient characteristics documented in EHRs, highlighting instances where racial bias is evident. Their findings shed light on the persistence of racial biases in healthcare documentation, which can contribute to disparities in patient care and outcomes. This study underscores the need for greater awareness and efforts to address racial bias in healthcare data to promote equity and fairness in healthcare delivery. In addition, using AdaBoost to change the weight of instances for learning results in 3% and 6% improvement for readmission and mortality predictions, respectively. Using SMOTE to generate synthetic instances to balance positive and negative samples for 30 day

readmission prediction has been studied [83] by using a UCI hospital readmission dataset [93]. The experiments show exceptionally higher AUC values (0.974) than results from other studies (normally around 0.7 AUC range). One possible reason is that UCI readmission dataset has a relatively balanced sample distributions because 11.2% samples belong to positive class (readmission), whereas in other dataset, such as National Readmission Database [5], the positive ratio is much smaller. By using different sampling approaches, including RUS, ROS, and ROSE, a method [30] comparatively studies the three methods using UCI readmission dataset [93], using different classifiers, such as SVM, random forest, gradient boosting, and regression and partition trees. The results show that ROSE is significantly worse than other approaches (including original data without any sampling). In addition, RUS and ROS have comparable performance, and both frequently outperform models trained from original imbalanced dataset. A readmission prediction algorithm Joint Imbalanced Classification and Feature Selection (JICFS) is proposed to construct the loss function and applied sample weight to handle class-imbalance problem [35]. Researchers propose a novel approach for bearing fault diagnosis. They introduce a multitasking intelligent system that utilizes representation learning techniques to address the challenges posed by imbalanced datasets common in real-world bearing fault data. By integrating multiple tasks into their model, they achieve improved diagnostic accuracy and robustness, even in scenarios where data samples for different fault classes are unevenly distributed. The paper presents a promising advancement in the field of intelligent fault diagnosis, particularly in situations where imbalanced data is a significant concern, offering potential benefits for various industrial applications reliant on machinery health monitoring and maintenance [106].

2.2 DATA PRIVACY AND LOCALITY IN HEALTH DATA

Common approaches to tackle data privacy and locality are to employ ensemble learning or federated learning [60]. The former trains multiple models from local datasets and combine them for prediction, whereas the latter trains one model from multiple decentralized/localized datasets. Ensemble learning combines multiple base models for prediction. Typical approaches include bagging, boosting, and stacking [71]. Bagging trains base models separately (often in parallel), and then combines them using weighted (or unweighted) majority voting. Boosting, on the other hand, trains base models in a sequential and progressive manner, so a later trained base model is improved based on an earlier trained base model. Stacking is a meta learning approach, which uses base classifiers to generate outputs, and then retrains another model from the outputs for prediction. In [108], a localized sampling approach is proposed to allow sampling process to focus on instances difficult to classify. By using localized sampling to generate balanced datasets, this approach is validated using data collected from several South Florida regional hospitals. A joint ensemble-learning model [105] combines weight boosting algorithm with stacking algorithm, and compares three major baseline (1) the LACE index, (2) RandomForest-Lasso-SMOTE, and (3) SMOTE (which uses SMOTE to replace bagging for data samping) on national Hosptial Quality Monitoring System (HQMS) database (including 651,816 records after data processing). The results show that LACE (which is commonly hospital score systems) has the least performance, confirming that machine learning is useful for hospital readmission prediction. Meanwhile, bagging with weight boosting and stacking shows clear benefits on high dimensional medical data with imbalance class distributions and imbalanced misclassification costs.

A research [17] proposes to use federated learning to build a global model to predict hospitalizations due to heart diseases using patient electronic health records (whether a patient will be hospitalized within one or two years, prior to the time

of prediction). A FL-based privacy-aware and resource-saving collaborative learning protocol was introduced in [40] for an EHR analysis management system working with multiple hospital institutions and cloud servers, where each hospital runs neural network models with its own EHR with the help of cloud computing. In addition, an FL-based approach was proposed to predict hospitalizations in patients diagnosed with heart disease using their historical EHR. More specifically, health data from an EHR system consisting of patient smartphones and distributed hospitals is trained locally on demographic information such as age, gender and physical characteristics [17]. [29] proposed a FLT scheme for wearable health monitoring, in which smartphones and cloud servers cooperated to train and share CNN model for the identification of privacy-conscious human activities. A disease prediction method using FL with a large national health insurance data set of 99 medical sites (such as hospitals and clinical laboratories) distributed across 34 states in the United States is studied [61]. The data included EHR for diabetes, psychological disorders, and ischemic heart disease. The FL approach achieves competitive performance in terms of high accuracy and privacy by comparing with traditional methods such as centralized learning and local training without federation. also builds a FL-based health monitoring solution for analyzing patient outcomes from distributed hospital networks. Interestingly, each hospital created an entity called the Personalized Treatment Effect Estimator. Each estimator can be classified in each subgroup, where individual treatment outcomes include outcomes of patient characteristics, and site indicators are used to estimate overall treatment outcomes coordination sites [92]. LoAdaBoost FedAvg is proposed to achieve higher model prediction accuracy on distributed intensive care data, in which local models with a high cross-entropy loss were further optimized before model averaging on the server [48]. Federated-Autonomous Deep Learning (FADL) is designed to update global model by training part of the model using all data sources in a distributed manner while the rest of the model is trained

with data from specific data sources [62]. When it comes to IID data, Haddadpour and his colleagues introduce a framework called federated averaging with compression (FedCOM), which the global model is decided not only by the update by the average of all clients' training results, but also determined by the previous global model [39].

2.3 LABEL COST AND SCARCITY IN HEALTH DATA

Recently, Active Learning (AL) has emerged as a machine learning method that can effectively address data labeling workloads and label scarcity [78, 84]. Its main strategy is to iteratively find the most informative data points to annotate. The annotated data are then used as part of the training data in the next iteration. With more and more iterations, the machine learning model's performance can be more and more improved. This strategy has been integrated into federated learning and generated a new paradigm called Federated Active Learning (FAL) [8,55,72,100]. The FAL framework consists of several clients and one central server. Each client holds one labeled dataset and one unlabelled dataset, which can not be shared with others. The server holds a test dataset that can be shared with all clients. The goal of FAL is to train a globally optimized model at the server by annotating informative data samples at the clients. FAL framework is trained in an iterative manner. In one iteration, each client first trains a local model with annotated data. And then, the local parameters are transmitted to the server. The server synthesizes local parameters into a global model. After that, the global model is sent to each client to annotate several unlabeled data with the highest informativeness. The labeled dataset is extended by merging the previously labeled dataset and the newly annotated dataset. In the next iteration, new local models are trained with the new labeled dataset. With more and more iterations, more and more data will be annotated and the global model can be improved.

A novel approach is designed to improve the classification accuracy of waste and natural disaster images using a combination of Active learning and Federated learn-

ing techniques. The approach utilizes Active learning to select the most informative and relevant data samples for labeling, reducing the labeling workload. These labeled samples are then utilized in a Federated learning setting, where multiple devices collaborate to train a shared model without sharing raw data centrally, which effectiveness has been demonstrated in achieving higher classification accuracy compared to traditional federated learning approaches [7]. A novel annotation strategy to enhance Federated Learning (FL) by leveraging the concept of active learning, F-AL, is proposed. to address the challenge of limited annotated data in FL scenarios. By incorporating active learning techniques, F-AL aims to intelligently select and query the most informative data samples from each client's local dataset, reducing the annotation burden and improving the performance of the global model. The paper presents the evaluation of F-AL, highlighting its potential benefits in promoting more effective and privacy-preserving FL implementations [8]. A semi-supervised and personalized framework that combines active learning and label propagation techniques is proposed. In this method, leverages unlabelled data from individual clients in the federated environment to enhance the activity recognition process. Active learning is used to intelligently select the most informative samples for labeling, reducing the labeling effort while improving the model's accuracy. Label propagation is then employed to propagate the labeled data across clients, allowing the global model to be personalized for each client's unique activity recognition requirements. The results demonstrate the superiority of the semi-supervised and personalized approach, highlighting its potential to achieve more accurate and personalized activity recognition in a federated setting [73].

Federated Active Learning with a focus on inter-class diversity is explored by introducing novel methodologies to improve the performance of Active learning in a Federated learning setting. By taking into account the diversity among different classes of data, the authors propose innovative techniques that enhance the selection

of informative samples for labeling during the active learning process. This approach is aimed at improving the overall performance of the federated learning model while reducing the labeling effort required from individual clients [55]. A novel framework for enhancing intrusion detection in Zero-Trust Security Models (ZSM) using federated learning and semi-supervised active learning techniques is created, which incorporates semi-supervised active learning to optimize the model by selectively labeling the most informative data samples, thus reducing the reliance on fully labeled data. The paper highlights the effectiveness of this combined approach in improving intrusion detection performance and addresses challenges related to data privacy and isolation in ZSM environments [69]. Wu Xing, et al, propose a framework that combines Federated Learning and Active Learning to improve disease diagnosis accuracy while preserving data privacy in a multi-center scenario. Federated Learning enables multiple medical centers to collaborate and train a shared model without sharing raw patient data. Active Learning is incorporated to intelligently select the most informative and relevant data samples from each center for labeling, reducing the need for extensive labeled data. It is evaluated on a multi-center dataset, showcasing its effectiveness in achieving higher diagnostic accuracy compared to traditional methods [100].

CHAPTER 3
IMBALANCED LEARNING FOR HOSPITAL READMISSION PREDICTION

A hospital readmission is defined as an admission where a patient previously discharged from a hospital is being admitted to the same or a different hospital, within a specific time interval such as 30 days or 90 days. The reasons behind a hospital readmission often differ from patient to patient [57] and the readmission rates between different medical institutions also vary significantly [107]. A readmission implies extra costs to the stakeholders, adds financial burden to the patients and deteriorates their life quality [19,94]. Hospital readmissions are also related to unsatisfying patient outcomes and heavy financial burden to the healthcare system [11, 41, 58]. Preventable readmissions can lead to almost $17 billion annual cost reduction [49]. Therefore, in 2012, a national Hospital Readmissions Reduction Program (HRRP) initiative started to link the health care payment to the quality of hospital care, by reducing payments to hospitals with excess readmissions and providing hospitals an incentive to improve their care coordination in post-discharge planning. HRRP is established to penalize hospitals with readmission rates exceeding the national average by a drop in their payments. It is expected that by implementing such a penalization, an improvement in post-discharge communication and care to patients can be implemented by hospitals and a reduction in readmissions can be expected [66].

Since 2012, many efforts have been taken, by hospitals, caregivers, and academics [2, 108], to reduce readmission. But unfortunately, after eight years, it is observed that the "needle has not moved very far" [77]. In 2019, Medicare, under the HRRP plan, cut payments to 2,853 hospitals. Among the 3,129 general hospitals

which were evaluated in the HRRP program, 83% of them received a penalty [77].

The reduction of hospital readmission rate is of great significance to Medicare system and the effective usage of health care resources. It is meaningful and important to predict preventable hospital readmission earlier than it really happens. Intuitively, this problem is equivalent to predicting the likelihood of a patient being readmitted again in the defined time-frame, using patient's current information, including demographics, diagnose, treatment, *etc.*

In order to promote research and analysis of national readmission rates for all patients, a Federal-State-Industry partnership sponsored by the Agency for Healthcare Research and Quality (AHRQ) published National Readmission Database (NRD) [4]; including patient level admission information from 2010 to 2017, regardless of the expected payer for the hospital stay. The NRD database provides a powerful public data source for readmission analysis, using all cause national scale patient level data with demographics, hospital, and treatment/procedure information.

Motivated by the above observation, we propose to use imbalanced learning for hospital readmission prediction. The goal is to predict whether a patient, based on his/her current hospital visit records, is likely going to be re-admitted or not within 30-days after being discharged from the current hospital visit. The main challenge of hospital readmission prediction is twofold:

- *Challenge 1*: The readmission visits (i.e., the positive class) are a small portion of the total hospital visits, representing a severe class imbalance problem for learning

- *Challenge 2*: Due to privacy and health regulation, the information available for patient characterization is limited; and is often only limited to the payment level information. However, there are over 80,000 procedures code, representing a high dimensionality and high sparsity problem for learning.

Motivated by the above challenges, in our research, we design an imbalanced learning strategy to predict patient readmission possibility. For *Challenge 1*, we use a random sampling approach to balance the sample distributions in the training set. By implementing random sampling, it helps ensure that our training dataset is representative of the overall population, which is crucial because biased or non-representative samples can lead to poor model performance. By randomly selecting data points, we are able to reduce the risk of introducing bias into our training set. In addition, overfitting can be reduced which occurs when a model learns to perform well on the training data but fails to generalize to unseen data. For *Challenge 2*, we create features from patient hospital visit, by combining patient demographic information, ICD-10 clinical modification (CM) and procedure codes (PCS), and Clinical Classification Software Refined (CCSR) conversion. Instead of directly using ICD-10- CM/PCS code to characterize patients, we convert each patient's visit to CCSR code space with a smaller feature space. Experimental results on NRD validate the effectiveness of our method. Our contributions can be summarized below:

- We first implement feature engineering on National Readmission Database (NRD) to create 16 features as a solution.
- We propose a imbalanced learning strategy, to tackle the data imbalance challenge in 30-day hospital readmission prediction.
- Experiments on NRD datasets demonstrate that our imbalanced learning approach achieves better performance.

3.1 THE PROPOSED METHOD

In this section, we present our proposed imbalanced learning algorithm for 30-day hospital readmission prediction. Our learning objectives are to (1) capture the most relevant and informative input feature by creating new features, modifying existing

ones, and (2) enable imbalanced learning for 30-day hospital readmission prediction. We will first present our feature engineering for hospital visit. Then we present our imbalanced learning for readmission predictions.

3.1.1 Feature Engineering for Hospital Visit

National Readmission Database National Readmission Database (NRD) was first created by the Agency for Healthcare Research and Quality (AHRQ) in 2015 to provide data support for analyses of national readmission rates and further promote the quality of health care [45]. AHRQ is in the family of Healthcare Cost and Utilization Project (HCUP); where a collection of longitudinal healthcare databases combined with professional data analysis tools are provided in order to facilitate healthcare-related policies improvement. The database contains both clinical and nonclinical elements and collects around 18 million discharges in a year. In order to protect patient privacy, no patient's is recorded in a NRD file. The actual admitted date, discharged date or any other content that may reveal personal information are coded in a special format for the derivation of the gap between two visits of the same patient. Both single and repeated visits for patients are captured in the NRD database, and patient revisits are linked through the "VLink" filed, as shown in Table 3.1. In 2016, the NRD database replaced the International Classification of Diseases, Ninth Revision, Clinical Modification (ICD-9-CM) applied in version 2015 with the tenth revision (ICD-10-CM/PCS) codes to represent clinical diagnosed and inpatient procedures [4]. ICD-10-CM/PCS codes are an American adopted version modified by Centers for Medicare and Medical Services (CMS) and the National Center for Health Statistics (NCHS), based on ICD-10, the statistical classification of disease published by the World Health Organization (WHO). 'CM' in ICD-10-CM codes stands for 'Clinical Modification'. There are more than 72,000 ICD-10-CM codes in the 2016 NRD database. Each ICD-10-CM code consists of 3 to 7 characters and the main

purpose is to enable healthcare institutions to have a better understanding on a patient's medical conditions so that a more comprehensive and efficient treatment can be provided to patients. ICD-10-PCS stands for an inpatient procedural system. The intention of ICD-10-PCS codes is to provide insurance companies, healthcare providers with specific and accurate patient medical records.

We chose to use the 2016 NRD database as the data resource for our research. There are three files in the database. The first file is a Core file, in which every patient is represented by a unique NRD-Visitlink. Each row encodes visit information for every single patient visit including patient demographics. The second file, severity file, contains supplementary data information for condition severity identification and hospital. The third file, the level file, represents the information about hospitals to which patients in Core file were admitted. For this paper, we mainly focus on data analysis using the Core file. There are total 17,197,683 number of visits recorded in the Core file, with each visit including 103 data elements recorded in 103 columns.

Table 3.1: Example to calculate readmission days

Visit	Patient Visitlink	LOS	NRD_DaysToEvent
1	112233	2 days	2679
2	112233	5 days	2691
3	112233	3 days	2789

3.1.1.1 Feature Engineering for NRD Database

The most important steps for successful data analysis are pre-processing data and extracting critical features [63]. In the clinical field, these steps are especially significant because medical data are inherently complex and contain a variety of data fields with different ranges. For this reason, we first removed patients visit records with outliers, which are marked as a special value in the database. After that, we

normalized columns with large range, such as total charges, to a fixed range. This is helpful to improve the performance of the final result.

In order to extra features for patient readmission prediction, we consider three types of features, including (1) patient demographics, (2) patient admission and discharge information, and (3) patient clinical information. Table 3.2 summarizes the features created for readmission prediction.

Table 3.2: Features chosen for prediction

Feature Type	Feature	Description
Demographics	AGE	Patient's age
	FEMALE	Patient's gender (binary, '1' is female)
	PAY1	Payment method
	PL_NCHS	Patient's location (based on NCHS Urban-Rural Code
	ZIPINC_QRL	Estimated median house income in the patient's zip code
	RESIDENT	Patient's local (binary, '1' is the patient comes from same state as hospital)
Admission and Discharge Information	AWEEKEND	Patient's admission Day (binary, '1' means the admission day is a weekend)
	MONTH	Patient's discharge month
	QUARTER	Patient's discharge quarter
	DISPUNIFORM	Disposition of patients
	LOS	Length of the hospital stay
	ELECTIVE	Binary, '1' represents elective admission
	REHABTRANSFER	Binary, '1' is rehab transfer
	WEIGHT	Weight to discharges in AHA universe
	TOTAL CHARGES	Patient's inpatient total charges
	1^{st} HOSPITAL VISIT	Binary,'1' means the first hospital visit
Clinical Information	CCSR Code	Clinical categories

Patient Demographics and Admission Information For research purposes, the patient demographics and medical records during patients hospitalization and discharge information are key for readmission prediction. Data provided by demographics information about each participant, such as age and gender, are crucial in helping us determine whether individuals in this study are representative samples of the target population. Analyzing demographics characteristics is a catalyst for exhaustive medical policy enhancement.

In addition, patient admission and discharge information also play important roles in determining the likelihood of a readmission visit in the future. For example, the length of stay (LOS) of the current visit may imply the degree of illness (or severity of the disease) with respect to the current visit. Take feature 'DISPUNIFORM' as another example. It refers to the place where a patient is discharged, such as a family with home care or a nursing center. This feature plays an important role in readmission prevention.

Patient Clinical Information In addition to the patient demographics and admission information, we also consider patient clinical information which is encoded as the ICD-10-CM code in the NRD database. For each patient visit, the ICD-10-CM codes detail the diagnose and treatment carried out during the patient visit. One essential challenge is that because ICD-10-CM are used for payment purposes and include all disease types, the total number of unique ICD-10-CM is very large. There are over 72,000 unique ICD-10-CM codes in the 2016 NRD database, making it highly ineffective to directly use ICD-10-CM codes as features for learning.

In order to reduce the number of features reflecting the patient clinical information, we convert the ICD-10-CM codes into Clinical Classification Software Refined (CCSR) codes. CCSR is an aggregation version for ICD-10-CM and it can improve the specificity of ICD-10-CM codes. Its utilization greatly improves the analysis on health models including healthcare cost, efficiency, outcomes [3]. Figure 3.1 shows

the distribution of total number of ICD-10-CM codes for each patient visit. The result indicates that the total amount of ICD-10-CM codes for per visit is concentrated between 5 and 20. Figure 3.2 and Figure 3.3 represent the frequency distributions of ICD-10-CM codes and CCSR codes respectively. Where the frequency of the codes in the dataset are sorted in a log 10 scale descending order and the x-axis stands for the rank order of the corresponding code. From these two figures, we can tell that the frequency of both kinds of codes follows a negative exponential function.

After converting ICD-10-CM codes to CCSR codes, the number of features used for patient clinical information is denoted by less than 500 unique CCSR codes, as we will soon explain in Section Experiments.

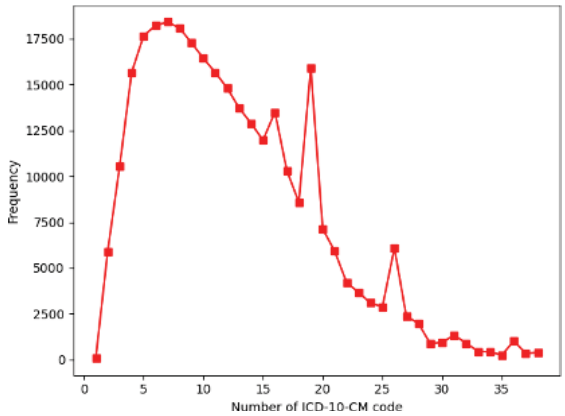


Figure 3.1: Distributions of the number of ICD-10-CM codes in each visit. The $x-$axis denotes the number of ICD-10-CM codes in a patient visit. The $y-$axis denotes that for each $x-$axis value, the number of patient visits (frequency) with the specified number of ICD-10-CM codes.

Readmission Labeling Protocol In order to generate class label for each patient visit, we label each patient visit as a readmission or not a readmission, by using 30-day as the criterion. Because our objective is to predict the possibility of a readmission in the future, we employ the following labeling protocol. For two visits (V_a and V_b) of the same patient, if the admission of V_b happens within 30-day (inclusive) after the

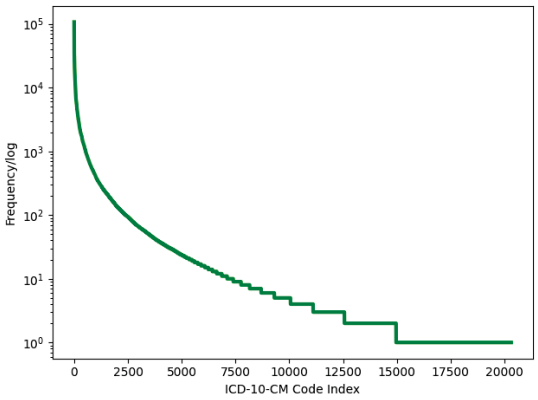


Figure 3.2: Distributions of ICD-10-CM codes across all patient visits in log-scale. The $x-$axis denotes the ICD-10-CM codes ranked in a descending order according to their frequency. The $y-$axis denotes the frequency of each code in log-scale.

discharge of V_a, we label V_a as a readmission visit (denoted by 1). Otherwise, V_a is labeled as not a readmission (denoted by 0). If the patient only have two visits V_a and V_b, then V_b will be labeled as not a readmission, because there is no succeeding visit following V_b. Intuitively, if the prediction is accurate, for each current patient visit, we will be able to estimate his/her readmission possibility in the future, when discharging the patient from the current visit.

Because there is no exact date information for the admission and discharge date of patient admissions, we need to calculate the gap (time period) between two admitted dates before labelling. In the NRD database, they use NRD_VisitLink to represent patient, thus, privacy can be protected through this de-identified patient record. Another feature used for privacy protection is NRD_DaysToEvent, where the actual patient admission date is substituted to a randomly chosen number (the main purpose is to hide the actual admission/discharge date of each visit for privacy protection). LOS stands for time duration a patient stays in the hospital after admission. Using these three features we are able to label which visit is a readmission.

An example to calculate gaps between hospital visits and the corresponding labels

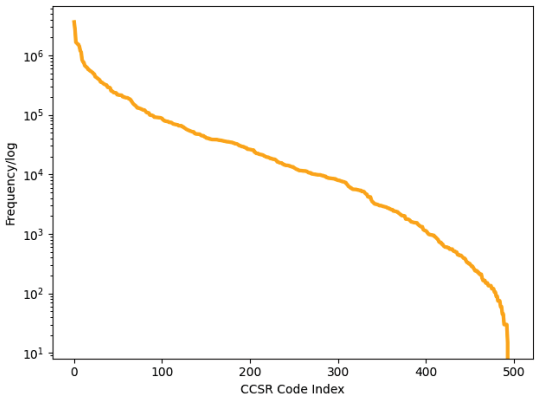


Figure 3.3: Distributions of CCSR codes across all patient visits in log-scale. The $x-$axis denotes the CCSR code ranked in a descending order according to their frequency. The $y-$axis denotes the frequency of each CCSR code in log-scale.

are shown in Table 3.1 and Figure 3.4. In Table 3.1, a patient has three visit records in the dataset. The time interval between visit 2 and visit 1 equals to the second NRD_DaysToEvent minus the first NRD_DaysToEvent minus the first LOS. This is 2691 - 2679 - 2 = 10. For visit 3 and visit 2, the calculation is 2789 - 2691 - 5 = 93. For visit links that appear more than once, if the time interval between two visits is less than 30 days (inclusive), the earlier visit is label as '1', which represents a readmission. Therefore, we should label the first time visit as readmission and the second as well as the third visits are labeled as not a readmission as showed in Figure 3.4. The reason why we do not label the second time as readmission is that the purpose of our research is to predict whether there will be a possibility that a patient will return to hospital in 30 days or not after being discharged. For those visit links only appear once in the dataset meaning there exists no readmission for the patients, the time interval is infinite and they are labelled as '0'.

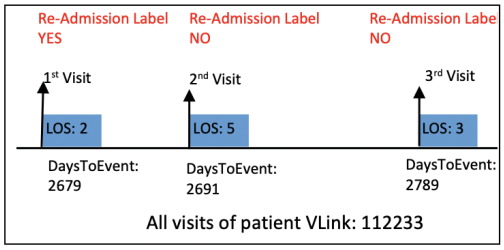


Figure 3.4: Temporal arrangement of patient visits for re-admission labeling (Based on visits showing in Table II).

3.1.2 Imbalanced Learning for Readmission Predictions

Using feature engineering and labeling process, we are able to create a training dataset with both features and labels, where each instance in the dataset represents a hospital visit. This is a typical supervised learning task. Many leaning algorithms can be applied to learn classifiers for prediction.

3.1.2.1 Class Imbalance

The final dataset for our research includes 300,000 rows representing 300,000 patient visits, 498 columns of patient clinical features (CCSR code), 16 columns of patient admission features, and one additional column denotes the label of the visit. Although the number of features in this dataset is not particularly large, the data is actually severely imbalanced. There are only 2,926 patients who conducted multiple visits to hospitals, in which 2,851 patients were admitted into hospitals twice, 74 visited hospitals three times and only 1 patient visited 4 times. With respect to the label part, only 881 visits are labelled as readmission and the rest 299,119 visits are not readmission. Figure 3.5 and Figure 3.6 show the sample distributions of the dataset. As a result, the ratio between readmission visits *vs.* not readmission visits is around 1:340, meaning that positive samples (readmission visits) are less than 0.3% of the whole training samples. This represents a well-known imbalanced learning challenge, because majority learning algorithms prefer an equal percentage of positive *vs.*

negative samples for learning accurate classifiers.

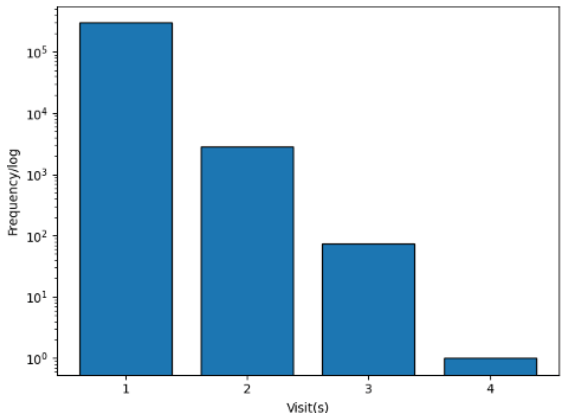


Figure 3.5: Distributions of the number of hospital visit(s) of all patients. Out of all 300,000 hospital visits, only 2,851 patients have two more more visits. If a patient only has one visit, the visit will be labeled as "no a readmission" (0).

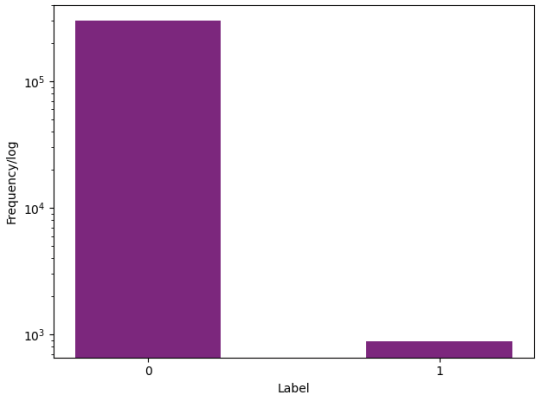


Figure 3.6: Class distributions between 30-day re-admission visits (labeled as "1") *vs*. non 30-day re-admission visits (labeled as "0". Overall, the re-admission visits are less than 0.3% of the total hospital visits.

3.1.2.2 Imbalanced Learning

Severe class imbalance will deteriorate the performance of the learning algorithms, as a result, the learning tends to be biased to the majority (negative) class samples, and

neglects the minority (positive) class. In our case, the positive samples (readmission visits) are less than 0.3% of the whole population, so a classifier can predict all instances to be negative and achieves 99.3% accuracy. This is, unfortunately, not useful for readmission prediction.

To tackle the class imbalance, we employ a random under sampling based approach to generate different versions of relatively balanced training set, where each training set contains a higher percentage of positive samples, compared to the positive/negative ratio in the original training set. More specifically, we applied a repeated k-fold cross-validation data frame in which re-sampling technique Random Under Sampling was used. Repeated k-fold cross-validation is a re-sampling method that repeatedly splits the dataset into k groups, and it is usually used to estimate the general performance of a model. In each fold, a bagging approach combined with three learning methods is implemented to combine results from multiple sampling. By doing so, the bias can be lowered and can demonstrate a better estimation in terms of statistics. The overall imbalanced learning algorithm is presented in Table 3.3.

Table 3.3: Imbalanced Learning Algorithm

Algorithm: Imbalanced Learning for Hospital Readmission Prediction

Input NRD database;

Output Prediction of a new visit: *Test*

For features in NRD database:

$\mathcal{F}$ ←Extract features as shown in Table I

For each visit ν in NRD database:

Label v as first visit or not

$\mathcal{F}_\nu$ ← Extract features from visit v using selected features $\mathcal{F}$

Label v as Readmission(1) or Not(0)

D ← Created traning set of NRD database

For each sampling repetition i:

$\mathcal{S}_i$ ← random under sampling to D to create a balanced training set

$\mathcal{C}_j$ ← Train a classifier from $\mathcal{S}_i$

Test result $[j]$← Predict using classifier ($\mathcal{C}_j$, *Test*)

End

Test Final prediction← Combine results from all sampling repetitions to make final prediction

3.2 EXPERIMENTS

3.2.1 Experimental Settings

We randomly extracted 300,000 patients visit records from the overall 17,197,683 patient visits and created 16 demographic and admission features, and 498 clinical features (CCSR codes) as shown in Table I to evaluate the algorithm performance for readmission prediction. In our experiments, the values in column AGE and TOTAL CHARGES are normalized through divided by the maximum value in the column to

range [0,1]. Due to the large number of ICD-10- CM codes in 2016 NRD, instead of directly using them, we converted them into manageable number of clinical categories. The CCSR enables a way to identify specific clinical conditions using ICD-10-CM codes and this helps reduce the number to 498 but still keep the clinical information of each patient visit. In the experiments, we count the number of each CCSR code for each visit, and use the numerical values as features for learning. So in total, our training set contain 300,000 instances (visits), where each instance is represented by 516 features and a class label.

For all experiments, we used a 10 times 10-fold cross validation. Making multiple 10-fold cross validation repeatedly divided the data into 10 blocks for ten times where every block has equal size. As a result, it will generate 100 re-samples that with averaged data. For each fold in cross validation, we implemented Random Under Sampling with different sampling ratios, where the proportion of positive and negative classes are designed as 1:1, 1:2, 1:5, 1:10. Three learning algorithms are used in the experiments, including Decision Tree, Random Forest with 500 trees, and Random Forest with 1000 trees.

3.2.2 Experimental Results

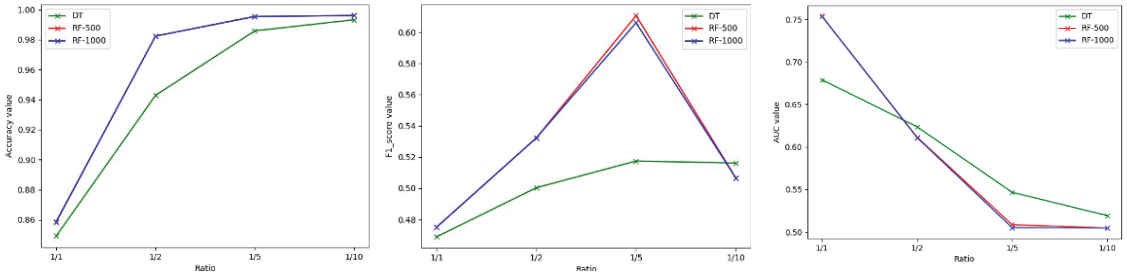
The detailed performance including accuracy, F1_score and Area Under the ROC Curve (AUC) values for learning method Decision Tree (DT), Random Forest with 500 trees (RF-500), and Random Forest with 1000 trees (RF-1000) using four sampling ratios are reported in Table 3.4.

Table 3.4: Performance of imbalanced learning algorithm

Learning Method	Performance	Positive:Negative Sampling Ration			
		1:1	1:2	1:5	1:10
DT	Accuracy	0.8491	0.9429	0.9859	0.9933
	F1_score	0.4688	0.5003	0.5174	0.5161
	AUC	0.6789	0.6236	0.5466	0.5191
RF-500	Accuracy	0.858	0.9824	0.9955	0.9961
	F1_score	0.4751	0.5322	0.6106	0.5066
	AUC	0.7538	0.6114	0.5085	0.5046
RF-1000	Accuracy	0.8585	0.9824	0.9955	0.9961
	F1_score	0.4749	0.5322	0.5060	0.5065
	AUC	0.7535	0.6109	0.5080	0.5046

The three line graphs in Figure 3.7 indicate the change trend of three performance values with respect to different sampling ratios. For accuracy performance, as showed in Figure 3.7a, the results of RF- 500 and RF-1000 are almost the same except the value under sampling ratio 1:1. All of the three methods show improved accuracy using 1:5 or more balanced sampling ratios (such as 1:1 or 1:2). When using more imbalanced sampling ratios (such as 1:5 or higher), the accuracy will remain stable. This is possible because that when data are imbalanced in the sampled set, using 1:5 or 1:10 sampling ratios, all positive samples will be misclassified as negative samples. Therefore, the accuracy will become stable (approaching to the percentage of negative samples in the test set).

As for the F1_scores, shown in Figure 3.7b, the change shows two opposite trends at the point of ratio 1:5 for three methods. Overall, RF-500 and RF-1000 demonstrate a more significant rate of descent than DT. This is, in fact, consistent with the accuracy showing in Figure 3.7a, where the accuracy remain stable when using 1:10



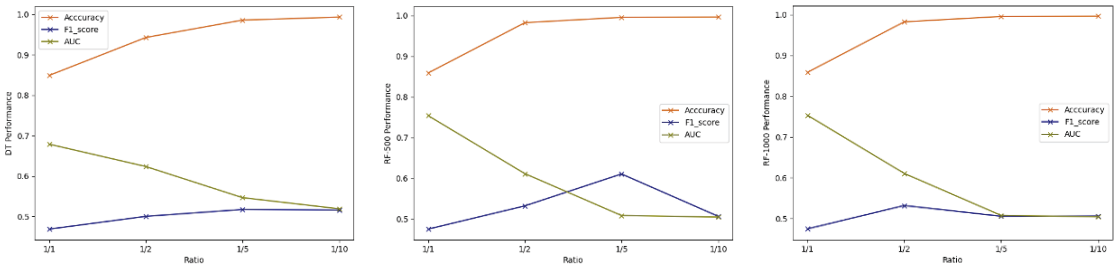
(a) Classification Accuracy (b) F1_score (c) AUC values

Figure 3.7: Performance comparisons using different class sampling ratios 1:1, 1:2, 1:5, 1:10

sampling ratio.

Figure 3.7c reports the AUC scores of all three methods with respect to different sampling ratios. Comparing to the accuracy and F1_score, AUC is much more accurate in evaluating the performance of the classifier with respect to both positive and negative samples. The results in Figure 3.7c show that as the sampling ratio is becoming more imbalanced (from 1:1 to 1:5), the performance of all methods deteriorate in their AUC scores. After the sampling ratio reach 1:5, using more imbalanced sampling, such as 1:10, does not deteriorate the algorithm performance further, because all positive samples are classified as negative samples, resulting in 0.5 AUC values.

Figure 3.8 reports performance of three learning methods using different sampling ratios. For DT, Figure 3.8a, its accuracy and f1_score keep climbing before ratio 1:5 and after it the ascent scope becomes smooth. However, the AUC score decreases for all the four ratios. RF-500, Figure 3.8b, is consistent with RF-1000, Figure 3.8c, in respect to accuracy and AUC, which is also the same as DT. The peak for RF-500 is the point at ratio 1:5 whereas it reaches the maximum at ration 1:2 for RF-1000.



(a) Decision Tree (b) RF(500 trees) (c) RF (1000 trees)

Figure 3.8: Performance comparisons between decision trees (a), and random forest with 500 trees (b), and 1,000 trees (c)

3.3 CONCLUSIONS

In this research, we proposed to use imbalanced learning for 30-day hospital readmission prediction. The main goal is to predict, at the time of a hospital discharge, whether the patient may return in 30 days or not in the future. To build a machine learning task, we used National Readmission Databases (NRD) to extract features from patient visits. We created a set of features, using simple patient demographics, ICD-10 clinical modification (CM), and Clinical Classification Software Refined (CCSR) conversion, to represent each hospital visit. Because patient readmission is only a small portion of all patient visits, the machine learning task is severely challenged by the imbalanced class distributions. To solve the challenge, we used random under sampling (RUS) to create different copies of balanced sample sets. Ensemble classifiers were trained from balanced sample sets to build classifiers for readmission prediction. Experiments on the NRD databases confirm that Random Forests, with 1,000 trees, deliver the best AUC scores for 30-day hospital readmission prediction.

CHAPTER 4

ENSEMBLE LEARNING FOR DISEASE-SPECIFIC READMISSION PREDICTION

Hospital readmission is a process or episode when a patient discharged from a hospital is readmitted within a specific time interval, say 30 or 90 days, since the previous discharge [98]. With annual costs reaching $41.3 billion for patients readmitted within 30 days after discharge, readmission is one of the costliest episodes to treat in the United States [46]. The large annual costs not only imply unsatisfactory hospital quality, but also hinder resources available for other attention-required government programs and erode US industrial competitiveness [14]. To minimize the negative impact of high readmission rate, since 2012, a Hospital Readmissions Reduction Program (HRRP) has been developed by Centers for Medicare & Medicaid Services (CMS) aiming to improve the quality of patient care and reduce healthcare expenditures by imposing fines on hospitals with higher readmission rates than expected rate [26]. Hospitals across the US are under scrutiny of this program and have increased the investment in order to enhance their discharge process, resulting in the drop of readmission rate from 21.5% to 17.5%, from 2007 to 2015 [109]. Despite of this encouraging drop, the expenses on developing an effective discharge procedure including better medication prescription, patient education, discharge follow-up and so on is extremely high and time consuming [65]. Development of readmission risk analysis tools has increased dramatically for accurate identification of high-risk patients. Nevertheless, the complexity of in-patient care and discharge process hinders the progress of building high-sensitivity and precise risk models, which stimulates growing research focusing on finding potential patterns of readmission and aiming to prevent avoidable

readmissions.

Machine learning, supervised learning in particular, has the unique strength to learn patterns from historical data for prediction. Accordingly, many methods have been proposed to train predictive models to assess readmission risk of individual patients, using their past visit records combined with other information [42, 68, 103]. For example, logistic regression is a commonly used model, due to its simplicity and transparency for prediction. In addition, studies also propose to use more advanced models, such as support vector machines and neural networks, for readmission analysis [21,81]. Our previous study [98] has systematically reviewed major research challenges for hospital readmission.

The main contribution of our work, compared to existing research in the field, is fourfold.

Answers to important questions: With over 15 million hospital visits in national readmission databases (NRD), we are able to carry out data statistics analysis and conclude answers for several important questions regarding hospital readmission. To find out the impact from demographics on hospital readmission, we explored the readmission percentage between gender and various age groups, from which an apparent readmission difference between male and female can be observed with male having higher readmission rate than female. Also, patients aged over 56 usually have larger risk to be readmitted into hospital. The second aspect we conclude is that patients suffering from diseases vary significantly regarding to their readmission rates. For example, patients with heart diseases have much more readmission rate than patients with pneumonia. As for hospital, private-owned non-profit hospitals discharged much more patients than government-owned hospitals and private-own hospitals.

Nationwide Admission Data Statistics: Using National Readmission Databases (NRD), with over 15 million hospital visits, as our testbed, we summarize nationwide patient admission data statistics, in related to to demographic, disease types, and hospital

factors. By separating patient visits into different cohorts, our study directly answers how demographic, socioeconomic, and diseases are reflected in the readmission. The data statistics can not only be useful for designing features for readmission prediction, but are also useful for policy and other purposes. For example, our study found that, even in the same disease group, patients with low incomes do not go/return to the hospital as the same as populations with higher incomes. These observations can help design policy to help patients vulnerable to high readmission risk in specific geographic locations or service areas.

Feature Engineering for Readmission Prediction: In order to design HIPAA compliant features to characterize patients, diseases, and hospitals, we use feature engineering to design 526 representative features to model each patient visit. The six demographic features, ten admission and discharge features, 498 clinical features, three disease features, and nine hospital features are fully compliant with the HIPAA standard to support disease-specific readmission prediction.

Disease Specific Readmission Prediction: Our studies found that readmission rates vary significantly from diseases to diseases. For six diseases studied in our research, their readmission rates vary from 1.832% (Pneumonia) to 8.761% (Diabetes). The high variance makes it inaccurate to use one model for all prediction. In addition, the readmission visits are a small portion of the patient visits, presenting a data imbalance issue for learning. Accordingly, we propose to use random under sampling, combined with hard-voting and soft-voting based ensemble leaning. By training different ensemble models using disease specific datasets, and comparing their performance using Friedman test and Nemenyi post-hoc test, our study shows the most accurate models for disease-specific readmission prediction.

4.1 THE PROPOSED METHOD

4.1.1 US National Readmission Databases Overview

Due to HIPAA regulations [22], patient data cannot be shared between researches. This creates a barrier for researchers to obtain hospital data for research study and designs. Nationwide Readmission Databases (NRD) provide an alternative public data source for readmission analysis, using all cause national scale patient level data. The NRD databases were first created by the Agency for Medical Research and Quality (AHRQ) in 2015 to provide data support for the analysis of national readmission rates and further improve the quality of medical care. AHRQ belongs to the "Health Care Cost and Utilization Project (HCUP)" family, which provides a collection of longitudinal healthcare databases combined with professional data analysis tools to promote the improvement of healthcare-related policies. The NRD database contains clinical and non-clinical elements and collects about 18 million unweighted discharges each year with more than 100 clinical and non-clinical variables per hospitalization. NRD is a unique and powerful database designed to support various types of analysis of national readmission rates for all payers and uninsured. The database addresses a huge gap in healthcare data: the lack of nationally representative information on hospital readmission rates for all age groups [5].

4.1.1.1 NRD Database Descriptions

The NRD database has three major tables, each includes information about patient, hospital, and disease, respectively. Each row of the core table represents a hospital visit, and table has 103 fields, including admission, diagnose, and discharge information. The 103 fields in the table can be separated into three main categories: Demographics, Admission and Discharge information, and Clinical information [97]. Patients' privacy are protected with de-identified KEY_NRD element and the dates

related to their in-patient treatment are replaced by sequential numbers. For clinical information, ICD-10-CM code is applied for medical diagnoses (the next subsection details the ICD diagnose code descriptions).

The hospital table in the NRD databases includes information about hospitals involved in the core table. The hospitals are across the whole country, with different types of ownership and teaching status, such as non-profit, government owned, or for-profit hospitals. In addition, hospitals are also categorized based on their bed sizes which reflect the scale/capacity of the hospital.

The disease severity table in the NRD databases includes diseases associated to each hospital visits. The disease information is based on the main reason of each admission. In addition, the risk of mortality and severity of illness are also encoded in the disease severity table. The code in the disease severity table is based on APRDRG (All Patients Refined Diagnosis Related Groups) code associated to each visit.

4.1.1.2 ICD Diagnose Code

In the NRD database, the diagnose and treatment with respect to each hospital visit are recorded using ICD-10-CM (International Classification of Diseases) code. The standardized coding allow stakeholders, including physicians, hospitals, and care givers, to classify and code all diagnoses, symptoms and procedures, with details necessary for diagnostic specificity and morbidity classification. For each visit, a number of ICD-10-CM and ICD-10-PCS (procedure coding system) codes are recorded to represent diagnose and procedures carried out during patient's visit. ICD-10-CM is the Clinical Modification of World Health Organization's International Classification of Diseases (ICD) 10^{th} version and it is used for medical diagnoses. An example of the ICD-10-CM code structure is shown in Fig. 4.1. In order to sufficiently serve health care needs, U.S. made the transition from ICD-9-CM to ICD-10-CM codes [24].

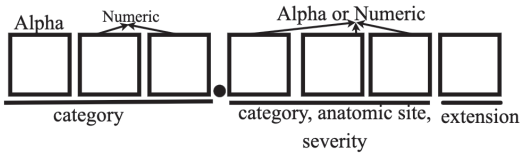


Figure 4.1: ICD-10-CM code structure. For example, S06.0X1A code means "Concussion with loss of consciousness of 30 minutes or less, initial encounter".

As shown in Table 4.1, ICD-10-CM codes are very different from ICD-9-CM codes with nearly 5 times as many diagnoses codes as in ICD-9-CM and it has alphanumeric categories instead of numeric ones. ICD-10-CM code sets provide more precise identification and conditions tracking by including laterality, severity, and complexity of disease conditions [24, 44]. The ICD-10-CM code specification has 21 chapters and it has a much longer index and tabular list. It uses an indented format for both the index and tabular list. Categories, subcategories, and codes are contained in the tabular list [12, 27]. ICD-10-CM codes can consist of up to 7 characters with the seventh digit extensions representing visit encounter or sequela for injuries and external causes compared to five digits in ICD-9-CM codes. Fig.4.1 shows the meanings of the seven characters: characters 1-3 indicate the category of diagnoses, characters 4-6 indicate etiology, anatomic site, severity, or other clinical detail, and character 7 is the extension. All ICD-10-CM codes begin with one of the alphas and they are not case sensitive. Although in the original version, alpha U was excluded, CDC released COVID-19-guidelines from April 1 2020 to September 30 2020 in which U07.1 is used to defined a positive COVID-19 test result, or a presumptive positive COVID-19 test result [23].

4.1.1.3 Readmission Label

In the NRD database, the core table only records each hospital visits (from admission to discharge). There is no readmission label associated to the visits. Therefore, we need to derive label to determine whether a visit is a readmission visit or not. For this

Table 4.1: Comparison between ICD-9-CM and ICD-10-CM Diagnosis Code Sets

ICD-9-CM	ICD-10-CM
14,025 codes	69,823 codes
3-5 characters	3-7 characters
First character is alpha or numeric	First character is alpha, second character is numeric
Characters 2-5 are numeric	Characters 3-7 can be alpha or numeric
Decimal placed after the first three characters	Decimal placed after the first three characters
Lacks detail and laterality	Very specific and has laterality

purpose, we need to leverage *NRD_DaysToEvent* (a timing variable specifies a number of days from a random "start date" to the current admission) and *LOS* (Length of stay) two fields in each record.

Each hospital visit record in NRD is kept in de-identified format in order to comply to the HIPAA regulations. As a result, not only patient's name is represented using *NRD-VisitLink*, the exact admission/discharge date are also adjusted using a specific random number for each patient. For each patient, a random "start date" is first selected. The admission time (*NRD_DaysToEvent*) of the patient is then calculated by using difference from the "start date" to the admission day. Starting from 2009, Centers for Medicare & Medicaid has been reporting each hospital's 30-day risk-standardized readmission rate (RSRR) across the U.S to measure unplanned readmissions that happen within 30 days of discharge from patients' admission, which has formed a 30-day readmission rule as a standard for hospital evaluation [57]. Thus, in our research, we use 30-day criterion for readmission labeling. For two visits (V_a

and V_b), if the interval between V_b admission and V_a discharge is less than 30 days, then visit V_a will be labeled as readmission [97]. One example to label patient visit is demonstrated in Table 4.2, in which the patient has three visits in total. The time interval between two visit is calculated as the second *NRD_DaysToEvent* minus the first *NRD_DaysToEvent* and minus the LOS. For visit 2 and visit 1, the result is 1053 - 1034 - 3 = 16, which is less than 30 days, therefore, we label the first visit as 1, indicating that this is a readmission visit. For visit 3 and visit 2, their difference is 1097 - 1053 - 2 = 42, so visit 2 is labelled as 0, meaning not a readmission. Visit 3 is also labelled as 0 because there is no more records showing the patient returning to the hospital after the third visit.

Table 4.2: Example to label patient visit

Patient Visitlink	Visit	NRD_Days ToEvent	LOS	Readmission Label
863245	1	1034	3 days	1
863245	2	1053	2 days	0
863245	3	1097	4 days	0

By using the above labeling approach, if two consecutive visits are within the defined interval (30-days in our setting), the first visit is labeled as the readmission visit. We do not label the second visit as readmission because we want to predict the possibility of a patient returning back to the hospital after being discharged from the current visit. By doing so, we can implement the hospital readmission prediction at the time of patient discharge.

4.1.2 NRD Data Statistics

4.1.2.1 Demographics Related Statics

Table 4.3 reports the NRD patient admission statistics. The total number of readmission in NRD is 17,197,683 in which the effective admissions is 15,722,444 excluding outliers. The number of effective admissions does not equal to the number of unique patients, because each patient has a unique *NRD-VisitLin* (global ID) and some patients will return back to the hospitals for multiple times. Table 4.3 shows that about 80% of patients only have a single visit, so readmissions happen to the rest 20% of patients. In Fig. 4.2, we further report the readmission percentages between gender and different age groups. Combining Table 4.3 and Fig. 4.2, we can find that although female patients are the majority part of hospital visits, the readmission rates of male population exceed that of female across all the age groups, especially for age group [18, 35], where the readmission rate of male is more than twice the rate of female.

Table 4.3: A summary of NRD patient admission

Categories	Number(%)
Effective Admission Total	15,722,444
30-Day Readmission	1,834,786 (11.67%)
Not 30-Day Readmission	13,887,658 (88.33%)
Unique Patient Total	11,691,620
Patient with single visit	9,335,277 (79.85%)
Patient with multiple visits	2,356,343 (20.15%)
Patient Visit Total	15,722,444
Male patient visits	6,630,005 (42.17%)
Female patient visits	9,092,439 (57.83%)

The NRD databases have three main payment types, Medicare, Medicaid, and Pri-

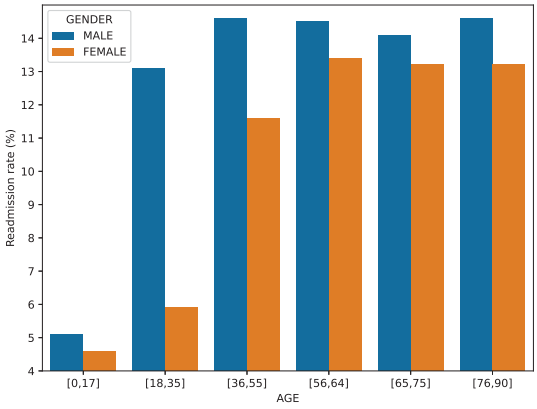


Figure 4.2: Gender readmission rate difference with respect to different age groups

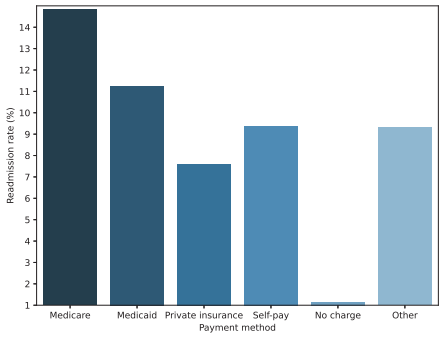


Figure 4.3: Readmission rate comparison with respect to different payment methods

vate insurance, which cover 43.40%, 21.80%, and 28.08% of payments in the database, respectively. In Fig 4.3, we report the readmission rates comparison between different payment groups. The results show that the top two highest readmission rates are from the Medicare and Medicaid patients, respectively. Fig. 4.2 shows that the readmission rates increase for older age groups, this partially explains why Fig 4.3 medicare and medicaid patients have higher readmission rates than patients from other payment groups.

4.1.2.2 Hospital Related Statistics

NRD hospital table includes information, such as ownership and teaching status, from about 2,355 hospitals across the US. In our analysis, we categorize hospitals based on their bed size and ownership. Hospital bed size are presented as numbers 1 to 3, indicating small, medium, and large respectively (this number indicates the capacity of the hospital). Fig. 4.4 reports the total admissions/discharges in 2016 from hospitals under different ownership. The results show that private-owned non-profit hospitals discharged much more patients than government-owned hospitals and private-own hospitals. Overall, as the hospital capacity increase (from 1 to 3), the mean admission/discharge numbers also increase. This is quite understandable because large capacity hospitals can accommodate more patient visits. In order to validate whether hospital ownership plays any significant roles in readmission, we report the readmission rates of different types of hospitals on admissions with five and more days during the visits. The results in Fig. 4.5 show that despite of the large difference in the total discharge, only a small variance is observed when comparing the percentage of admissions with Length of Stay (LOS) >= 5 days.

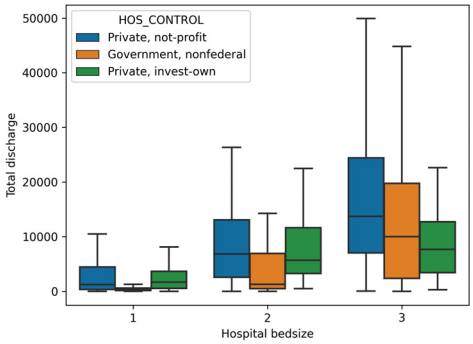


Figure 4.4: Total annual hospital discharge

In order to understand whether hospital ownership and capacity introduce significant variance to the diagnose and procedures carried out during the patient visits,

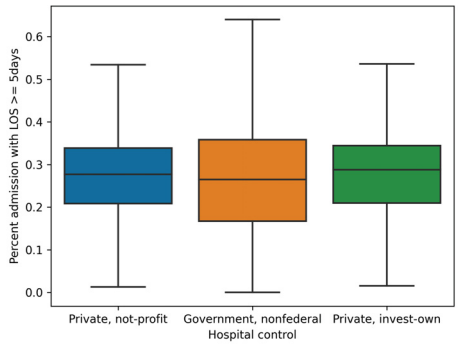


Figure 4.5: Percentage of admission with LOS >= 5 days

we report the average number of ICD-10-CM codes and ICD-10-PCS codes for each visit in Fig. 4.6 and Fig. 4.7, respectively. The results show that, in general, patients admitted to non-federal government-owned hospitals have less amount of averaged ICD-10-CM/PCS codes for their in-patient treatment, compared with patients admitted to private-owned not-profit hospitals and private-owned investment hospitals. Meanwhile, hospital bed size (or capacity) also play significant roles, especially in terms of the ICD-10-PCS. The results show an explicit rising trend, as the bed size increases for all kinds of hospitals. This is possibly because that large scale hospitals frequently accommodate patients with more complicated (or severe) disease conditions, and therefore more diagnoses and procedures are carried out on those patients.

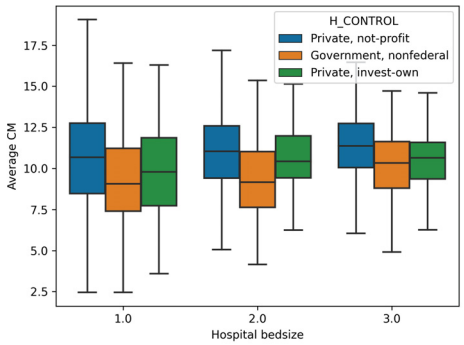


Figure 4.6: Average number of ICD-10-CM codes in each visit

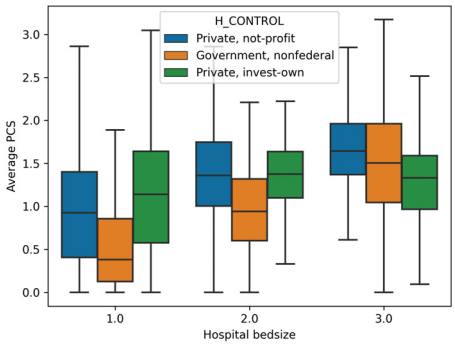


Figure 4.7: Average number of ICD-10-PCS codes in each visit

4.1.2.3 Disease Related Statistics

Disease and the level of severity are the two important factors associated to readmission. The disease severity table in the NRD database records the illness measurement of each patient in the core table, where each row is the description of patient's classification according to their admission reason, risk of mortality and severity of illness. One major disease is identified for each admission. The coding is based on the APRDRG (All Patients Refined Diagnosis Related Groups) code.

In order to understand the readmission difference between different disease specific patient cohorts, we comparatively study top leading disease to death as well as the top diseases for admission. There are 320 APRDRG code in total and 38% patients are diagnosed as "Moderate loss of function". We extracted the top 10 most frequent reasons for hospital admission based on the APRDRG code for each visit. In addition, we also report the top seven leading diseases to death according to CDC [25] to analyse the readmission rate and revisit rate. Table 4.4 and Table 4.5 report the statistics of top 10 APRDRG coded diseases/reasons and top seven leading disease of death, respectively.

The results from Tables 4.4 and 4.5 show that readmission rates of patients suffering from different diseases vary significantly in their readmission rates. For example,

Table 4.4: Readmission distributions for the top 10 APRDRG in NRD

Admission reason	Readmission rate	Revisit rate
Vaginal delivery	0.048%	0.168%
Septicemia & disseminated infections	3.983%	9.184%
Neonate birthwt > 2499g, normal newborn or neonate w other problem	0.848%	0.847%
Cesarean delivery	0.013%	0.062%
Heart disease	8.696%	19.500%
Knee joint replacement	0.392%	5.775%
Other pneumonia	1.800%	4.654%
Chronic obstructive pulmonary disease(COPD)	6.990%	16.684%
Hip joint replacement	1.088%	5.222%
Cardiac arrhythmia & conduction disorders	3.662%	7.868%

vaginal delivery and cesarean delivery are the two APRDRG coded top reasons for admissions, but these visits have very small readmission rates. For the top seven leading diseases to death, their readmission rates also vary significantly, where diabetes have the highest readmission rates (8.761%) and pneumonia has the lowest readmission rates (1.832%). Overall, readmission rates and revisit rates for leading diseases to death are much higher than the 10 most common admissions. This is due to the nature of the diseases and their complications.

In order to study the readmission rate variance with respect to socioeconomic factors, we report the readmission rates of the seven leading diseases of death with respect to the family incomes, which are coded by ZIP 1 to Zip 4 meaning low to

Table 4.5: Readmission distributions for the top seven leading diseases of death

Leading diseases	Readmission rate	Revisit rate
Heart disease	8.092%	17.873%
Stroke	2.448%	3.770%
Pneumonia	1.832%	4.738%
COPD	6.990%	16.684%
Cancer	6.823%	12.275%
Diabetes	8.761%	14.372%
Nephritis & nephrosis	7.019%	10.595%

high incomes. Readmission rates for four ZIP code areas categorized by the estimated median household income of residents in the patient's residence for the seven leading disease are shown in Fig. 4.8. The results show that area gap can be observed explicitly: for every disease, readmission rates for patients from lower income families (ZIP 1 and ZIP 2) are higher than those from high-income families (ZIP 3 and ZIP 4). Table. 4.6 summarizes factors of interest analyzed in this paper as for demographic, hospital and disease respectively.

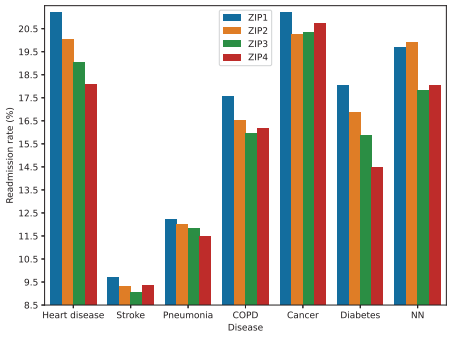


Figure 4.8: Readmission rate for leading diseases of death with respect median household incomes (ZIP 1 to 4 denotes an increasing level of incomes)

Table 4.6: Factors of interest analyzed in NRD database

Aspect	Factors of interest
Demographic	Gender; Age; Payment (Insurance)
Hospital	Bed size; Ownership
Disease	Disease type; ZIP code (Household income)

4.1.3 Feature Engineering for Disease specific hospital readmission prediction

Based on the nationwide hospital admission data statistics, we design five types of features, demographics features, admission and discharge features, clinical features, disease features, and hospital features, and use ensemble learning, combined with under random sampling, for disease specific readmission prediction.

Table 4.7 lists five types of features created using feature engineering to capture patient, disease, and hospital information. In the following, we briefly describe each type of features, and explain why they were chosen for readmission prediction.

4.1.3.1 Demographics Features

Demographic is a combination of population demography and socioeconomic information, which includes patient gender, age, average income of the community, patient medical record and so on. A generalization of a specific geography's population can be concluded based on a sampling of people in that geography and profoundly affect how important decisions are made. In medical institution, statistical results obtained from the patient allow for the identification of a future patient and the categorization, such analysis will enhance the development of high pertinence medical policy.

Table 4.7: Features created for disease specific hospital readmission prediction

Feature Type	Feature	Description	Feature size and domain
Demographics Feature	AGE	Patient's age	$\mathbb{R}^1 \in \mathbb{R}_+$
	FEMALE	Patient's gender (binary, '1' is female)	$\mathbb{N}^1 \in \{0, 1\}$
	PAY1	Payment method	$\mathbb{N}^1 \in \{1, 2, 3, 4, 5, 6\}$
	PL_NCHS	Patient's location (based on NCHS Urban-Rural Code	$\mathbb{N}^1 \in \{1, 2, 3, 4, 5, 6\}$
	ZIPINC_QRL	Estimated median house income in the patient's zip code	$\mathbb{N}^1 \in \{1, 2, 3, 4\}$
	RESIDENT	Patient's location ('1': the patient is from same state as hospital)	$\mathbb{N}^1 \in \{0, 1\}$
Admission and Discharge Feature	AWEEKEND	Admission Day ('1': the admission day is a weekend)	$\mathbb{N}^1 \in \{0, 1\}$
	MONTH	Patient's discharge month	$\mathbb{N}^1 \in \{1, 2, 3, \cdots, 12\}$
	QUARTER	Patient's discharge quarter	$\mathbb{N}^1 \in \{1, 2, 3, 4\}$
	DISPUNIFORM	Disposition of patients	$\mathbb{N}^1 \in \{1, \cdots, 7, 20, 21, 99\}$
	LOS	Length of the hospital stay	$\mathbb{N}^1 \in \mathbb{N}$
	ELECTIVE	Binary, '1' represents elective admission	$\mathbb{N}^1 \in \{0, 1\}$
	REHAB	Binary, '1' is rehab transfer	$\mathbb{N}^1 \in \{0, 1\}$
	WEIGHT	Weight to discharges in AHA universe	$\mathbb{R}^1 \in \mathbb{R}_+$
	CHARGES	Patient's inpatient total charges	$\mathbb{R}^1 \in \mathbb{R}_+$
	1^{st} VISIT	Binary,'1' means the first hospital visit	$\mathbb{N}^1 \in \{0, 1\}$
Clinical Feature	CCSR Code	Clinical categories	$\mathbb{N}^{498} \in \mathbb{N}$
Disease Feature	APR−DRG	Patient admission reason	$\mathbb{N}^1 \in \mathbb{N}$
	RISK	The mortality risk	$\mathbb{N}^1 \in \{0, 1, 2, 3, 4\}$
	SEVERITY	The severity of illness	$\mathbb{N}^1 \in \{0, 1, 2, 3, 4\}$
Hospital Feature	BEDSIZE	Hospital bed size	$\mathbb{N}^1 \in \{1, 2, 3\}$
	CONTROL	Hospital ownership	$\mathbb{N}^1 \in \{1, 2, 3\}$
	URU	Hospital urban−rural designation	$\mathbb{N}^1 \in \{1, \cdots, 9\}$
	AVE_CHARGE	Average charge amount per patient visit of the hospital	$\mathbb{R}^1 \in \mathbb{R}_+$
	AVE_CM	Average number of ICD-CM per patient visit of the hospital	$\mathbb{R}^1 \in \mathbb{R}_+$
	AVE_PCS	Average number of ICD-PCS per patient visit of the hospital	$\mathbb{R}^1 \in \mathbb{R}_+$
	PER_LOS	Percentage admission with LOS larger than 5 days	$\mathbb{R}^1 \in \mathbb{R}_+$
	DIS/UNI	Sample discharges/Universe discharges in NRD_STRATUM	$\mathbb{R}^1 \in \mathbb{R}_+$
	DIS/BED	Total hospital discharges/num bed size of hospital	$\mathbb{R}^1 \in \mathbb{R}_+$

4.1.3.2 Admission and Discharge Features

Informative materials about patient in-hospital activities can be obtained from admission and discharge information. There are time-related message indicating the exact time of the patient admission and length of stay (LOS) for treatment, admission nature-related information such as whether the patient was hospitalized through

emergency or not and so on. This kind of information offers a comprehensive view of the procedures a patient received from the healthcare providers, how patient's condition improve, and whether the treatment is adequate and effective to prevent readmission.

4.1.3.3 Clinical Features

Clinical features are used to characterize diagnoses and treatments patient received during the hospital visit. Because each patient's medical condition varies and there are tens of thousands of subcategory disease types, medical treatments, procedures *etc.*, finding good clinical features to represent patients is a significant challenge.

An essential challenge of using ICD-10-CM codes as clinical features to represent patients is that the total number of unique ICD-10-CM codes is very large (about 70,000), making it ineffective and computationally expensive for learning. Accordingly, we employ ICD-CCSR transformation [97] to convert ICD-CM code to CCSR code. CCSR stands for Clinical Classification Software Refined, which is used to aggregate ICD-10-CM/PCS codes into clinically meaningful categories. Fig. 4.9 shows CCSR code structure, where the first three letters mean the body system category and the last three numbers are CCSR categories numeric sequence of individual CCSR category starting at "001" within each body system [6]. In the code assignment, each CCSR code is designed to match to at least one or multiple ICD-10-CM code categories. Table 4.8 shows an example of many-to-one CCSR mapping, where multiple ICD-10-CM codes, corresponding to "displaced fracture of shaft of left clavicle", are mapped into one CCSR code [6]. The alphabetic correspondence between ICD-10-CM code and CCSR code is listed in Table 4.9, where the alphabetic conversion follows defined rules, and the numeric part also follows the user guide [6]. In Fig. 4.10a and Fig. 4.10b, we report the ICD-10-CM code distributions for Pneumonia disease and the mapped CCSR code distributions. In the figure, the y-axis shows the logarithm of

the code frequency sorted in a descending order, and the index of the corresponding code is shown in the x-axis. For ICD-10-CM codes, the log scale of the code frequency still follows a negative exponential function, meaning that ICD-10-CM code frequency follows an exponential to the power of exponential decay, and a few ICD-10-CM codes have very high frequency. The converted CCSR code frequency follows an exponential decay (so the logarithm function is close to a linear line). The ICD-10-CM to CCSR conversion not only preserves similar node frequency patterns, but also reduces the clinical feature dimension in our experiments from about 70,000 to around 498 as shown in "Feature size and domain" in Clinical Feature in Table 4.7. As a result, the clinically meaningful categories, with respect to each disease, are provided to detail diagnoses and treatments implemented during patient in-hospital visit.

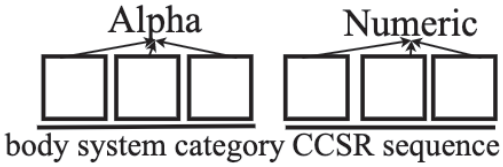


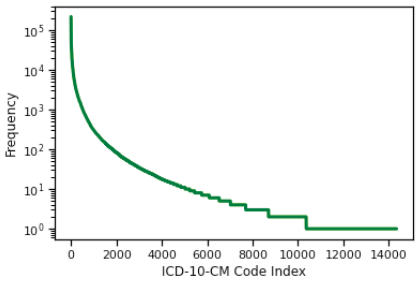
Figure 4.9: CCSR (Clinical Classification Software Refined) code structure. For example, INJ008 code indicates Traumatic brain injury (TBI); concussion, initial encounter.

4.1.3.4 Disease Features

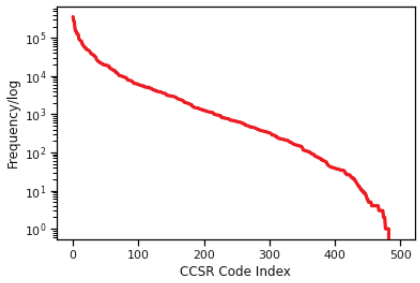
In addition to the CCSR code specified clinical features, three disease-level features are also added. The first feature is called APR−DRG, which represents the patient admission reason. Because a disease may include multiple subgroups, we select all APR-DRG codes related to one disease, and then use a numeral number to encode the feature value. Table 4.10 lists the APR-DRG codes selected for all six diseases in our study. For example, "Heart Disease" has six sub-groups (each has one APR-DRG code). We then use six integers, 10, 11, 12, 13, 14, 15, to encode them. By doing so,

Table 4.8: An example of ICD-10-CM to CCSR mapping

ICD-10-CM code	ICD-10-CM code description	CCSR category	CCSR description
S42022D	Displaced fracture of shaft of left clavicle, subsequent encounter for fracture with routine healing	INJ041	Fracture of the upper limb; subsequent encounter
S42022G	Displaced fracture of shaft of left clavicle, subsequent encounter for fracture with delayed healing	INJ041	Fracture of the upper limb, subsequent encounter
S42022K	Displaced fracture of shaft of left clavicle, subsequent encounter for fracture with nonunion	INJ041	Fracture of the upper limb, subsequent encounter
S42022P	Displaced fracture of shaft of left clavicle, subsequent encounter for fracture with malunion	INJ041	Fracture of the upper limb, subsequent encounter



(a) ICD-10-CM (Pneumonia)



(b) CCSR (Pneumonia)

Figure 4.10: (a) Distributions of ICD-10-CM code of all Pneumonia disease patient visits. The x-axis denotes the ICD-10-CM codes ranked in a descending order according to their frequency. The y-axis denotes the frequency of each code in log scale. (b) Distributions of CCSR codes converted from ICD-10-CM codes in (a). The x-axis shows the CCSR code ranked in a descending order according to their frequency. The y-axis denotes the frequency in log-scale.

we are encoding APR-DRG codes as numerical values within similar range, allowing some learning algorithms, such as logistic regression to better leverage the code value.

RISK is the second extracted disease-level feature representing the risk of patient

Table 4.9: Correspondence between ICD-10-CM and CCSR Categories by Body System

ICD-10-CM	Body System Description	CCSR
A, B	Infectious and Parasitic Diseases	INF
C	Neoplasma	NEO
D	Neoplasms, Blood,Blood-forming Organs	BLD
E	Endocrine, Nutritional, Metabolic	END
F	Mental and Behavioral Disorders	MBD
G	Nervous System	NVS
H	Eye and Adnexa, Ear and Mastoid Process	EYE/EAR
I	Circulatory System	CIR
J	Respiratory System	RSP
K	Digestive System	DIG
L	Skin and Subcutaneous Tissue	SKN
M	Musculoskeletal and Connective Tissue	MUS
N	Genitourinary System	GEN
O	Pregnancy, Childbirth and the Puerperium	PRG
P	Certain Conditions Originating in the Perinatal Period	PNL
Q	Congenital Malformations, Deformations and Chromosomal Abnormalities	MAL
R	Symptoms, Signs and Abnormal Clinical and Lab Findings	SYM
S/T	Injury, Poisoning, Certain Other Consequences of External Causes	INJ
U	no codes listed, will be used for emergency code additions	
V, W, X, Y	External Causes of Morbidity (home-care will only have to code how patient was hurt; other settings will also code where injury occurred, what activity patient was doing)	EXT
Z	Factors Influencing Health Status and Contact with Health Services (similar to current "V-codes")	FAC

mortality. There are five different levels (0 to 4) indicating patient's likelihood of dying where level 4 mortality means the highest risk. The last feature is SEVERITY standing for the severity of illness and the degree of loss of function. Similar to RISK, degree zero to extreme severity is represented by number 0 to 4.

Table 4.10: APR-DRG codes selected for the six studied diseases

Disease	Components	APR-DRG	Feature
Heart Disease	Heart &/ lung transplant	2	10
	Major cardiothoracic repair of heart anomaly	160	11
	Cardiac defibrillator & heart assist implant	161	12
	Permanent cardiac pacemaker implant w AMI, heart failure or shock	170	13
	Perm cardiac pacemaker implant w/o AMI, heart failure or shock	171	14
	Heart failure	194	15
Cancer	Nervous system malignancy	41	20
	Respiratory malignancy	136	21
	Digestive malignancy	240	22
	Malignancy of hepatobiliary system & pancreas	281	23
	Musculoskeletal malignancy & pathol fracture d/t muscskel malig	343	24
	Kidney & urinary tract malignancy	461	25
	Malignancy, male reproductive system	500	26
	Uterine & adnexa procedures for ovarian & adnexal malignancy	511	27
	Female reproductive system malignancy	530	28
Stroke	Intracranial hemorrhage	44	44
	CVA & precerebral occlusion w infarct	45	45
	Nonspecific CVA & precerebral occlusion w/o infarct	46	46
Pneumonia	Bronchiolitis & RSV pneumonia	138	138
	Other pneumonia	139	139
Diabetes	Diabetes	420	420
COPD	COPD	130	30

4.1.3.5 Hospital Features

Hospital features are created to characterize hospital ownership, bed size (capacity), locations, and patient body admitted to the hospitals. For example, hospital bed size tells us the hospital scale, the ownership represents the control of the hospital, and the geographic locations of the hospitals specify the patient demographic. In addition to simple statistics, we also create several statistics features, such as the average charge amount and the average number of ICD-CM codes for each visit. For feature DIS/UNI, the universe discharge is the total number of inpatient discharges in the universe of American Hospital Association (AHA) excluding non-rehabilitation and Long-Term Acute Care Hospitals (LTAC) for the stratum. These features provide specific understanding of patient in-hospital treatment in order to discover the effect of different treatment provided by hospitals towards hospitalized patients' recovery.

4.1.4 Prediction Framework

Six disease-specific datasets are extracted (we focus on the leading diseases of death as given in Table 4.11), including cancer, heart disease, chronic obstructive pulmonary disease (COPD), diabetes, pneumonia, and stroke. All six datasets are imbalanced due to the nature of the readmission [97].

In the six datasets, the ratios of non-readmission visits (negative samples) to readmission visits (positive samples) all exceed 10 (with the largest value 53). This imbalanced distribution causes the machine learning model to be more biased towards majority (negative) samples, which in our case, non-readmission samples and causes poor classification of minority (positive) classes. As a result, the model will give a high false negative value, which means a patient is not considered that he will be readmitted to the hospital but actually he is. Such classification performance will not only hinder the application of machine learning models but also will not be able to detect potential illness in advance, which goes against our intent, because one of the

Table 4.11: Readmission distributions for the top seven leading diseases of death

Leading diseases	Readmission rate	Revisit rate
Heart disease	8.092%	17.873%
Stroke	2.448%	3.770%
Pneumonia	1.832%	4.738%
COPD	6.990%	16.684%
Cancer	6.823%	12.275%
Diabetes	8.761%	14.372%
Nephritis & nephrosis	7.019%	10.595%

reasons AI models are applied to healthcare is to anticipate potential risks, to prevent patients suffering from pain, to reduce the burden on patients and the burden on the healthcare system [15].

In order to tackle the class imbalance, Random Under Sampling (RUS) is applied to balance the ratio between positive and negative samples. RUS is employed to generate various versions of relatively balanced training sets, in which positive samples have a higher percentage than the original dataset. During this process, the sampling radio applied to the data is critical, and will impact on the algorithm performance. In addition, RUS changes the sample distributions, and inevitably introduces bias to the training data. In order to address the above challenges, we propose to employ three solutions as follows:

- *Sampling Ratios:* We will employ different sampling ratios to the random under sampling to balance the positive *vs.* negative samples, valid the algorithm performance, and choose the best sampling ratios for readmission prediction.

- *Ensembles:* We will carry out random under sampling for multiple times on the training data. The classifiers trained from each copy of the sampled data are combined to form an ensemble for prediction. This will alleviate the bias and

improve the overall performance.

- *Soft vs. hard voting:* We will validate two ways to combine classifiers trained from random under sampled data, hard voting *vs.* soft voting. Assume $\hbar_t()$ denotes a trained classifier in a classifier ensemble $\mathbb{E}$, Eq. (4.1) defines the binary prediction of the classifier on a test instance x, where $\Pr_{\hbar_t}(c|x)$ define the class distribution (*i.e.*, conditional probability) of the classifier predicting instance x to class c. Hard voting predicts the final class label with the most agreed votes by summing the predictions for each class label from models, as shown in Eq. (4.2), where $I(\hbar_t(x) = c)$ returns 1 if classifier $\hbar_t(x)$ predicts instance x to be class c, or 0 otherwise. Soft voting, defined in Eq. (4.3), summarizes the predicted class probabilities for each class from models and predict the classes with the largest summed probability.

$$\hbar_t(x) = \arg \max_{c \in \{P,N\}} \Pr_{\hbar_t}(c|x) \tag{4.1}$$

$$\hat{y}_x = \arg \max_{c \in \{P,N\}} \sum_{t=1}^{|\mathbb{E}|} |I(\hbar_t(x) = c)| \tag{4.2}$$

$$\hat{y}_x = \arg \max_{c \in \{P,N\}} \sum_{t=1}^{|\mathbb{E}|} \Pr_{\hbar_t}(c|x) \tag{4.3}$$

4.2 EXPERIMENTS

4.2.1 Experimental settings

We create six disease-specific readmission datasets from NRD databases (2016 version). The datasets and their simple statistics are reported in Table 4.12. Using feature engineering approaches, we created 526 features for each instance (which represents a hospital visit). The list of features are summarized in in Table 4.7. Among

Algorithm 1 Disease Specific Hospital Readmission Prediction

Input: (1) Disease specific database: $\mathcal{D}$;

(2) Under Sampling Ratio: r;

(3) Ensemble size: K

Output: Prediction on a test hospital visit: x

$\{\mathcal{D}^+, \mathcal{D}^-\} \leftarrow$ Label pos. (+) and neg. (-) visits in $\mathcal{D}$

$\mathbb{F}^{\mathcal{D}} \leftarrow$ Create features from $\mathcal{D}$

$\mathbb{E} \leftarrow \emptyset$

for each random under sampling (RUS) round $t \in K$

$[\hat{\mathcal{D}}^-, \hat{\mathcal{D}}^-] \leftarrow$ RUS with ratio r on $\mathcal{D}^-$ and $\mathcal{D}^+$

$\hat{\mathcal{D}} \leftarrow \{\mathcal{D}^+ \cup \hat{\mathcal{D}}^-\}$. Create balanced training set

$\hbar_t(\cdot) \leftarrow$ Train classifier from $\hat{\mathcal{D}}$ using features $\mathbb{F}^{\mathcal{D}}$

$\mathbb{E} \leftarrow \mathbb{E} \cup \hbar_t(\cdot)$

end for

$\hat{y}_x \leftarrow$ Apply hard voting or soft voting for prediction.

return $\hat{y}_x$.

all features, AGE, TOTAL CHARGES, and AVE_CHARGE are normalized to range [0, 1] by dividing each value by the maximum value in the column.

In order to evaluate the performance between different random under sampling ratios and different voting approaches, including hard voting *vs.* soft voting, for disease-specific readmission prediction, we will need to repeat experiments for a large number of times. Therefore, for three large datasets (COPD, Heart Disease, and Pneumonia), we randomly sample 300,000 records from each of them, and use the sampled datasets to validate the parameter settings. For the remaining experiments, the whole datasets are used for each disease.

All experiments use 10-fold cross validation. For each fold, RUS is applied to the training data, using different sampling ratios, where the ratios between negative

vs. positive classes vary from 0.5:1, 0.7:1, 0.8:1, 0.9:1, 1:1, 1.1:1, 1.2:1, 1.5:1, 2:1, 3:1, 4:1, to 5:1. Instead of using 1:1 balanced sampling, like most existing methods do, we intentionally vary the class ratios to a large range, to study how will class distributions impact on the readmission prediction results.

Four learning algorithms are used in the experiments, including Decision Tree, Random Forest with 500 trees, Logistic Regression. and Gradient Boosting.

Table 4.12: Total sample number and sample ratio in six disease datasets

Datasets	Total sample number	Negative:positive sample ratio
COPD	327,269	10.88
Heart disease	582,058	10.16
Cancer	171,495	12.3
Diabetes	183,726	10.4
Pneumonia	358,001	7.38
Stroke	273,395	45

4.2.1.1 Performance Metrics and Statistical Test

Four performance metrics, Accuracy, Balance Accuracy, F1-score, and AUC, are used in our experiments. The purpose of using other three measures, in addition to accuracy, is to take class imbalance into consideration for validation.

We use Friedman test [32] to validate statistical difference between four models trained on the six datasets. For each measurement, the classifiers are ranked according to their performance in a descending order. The classifier with the best score is ranked as 1 and the one with the lowest is ranked as 4. Two classifiers present the same measurement performance score are ranked with the average rank.

Assume that R_j denotes the average rank of a classifier j and r_i^j is the rank of classifier j on dataset i, Eq. (4.4) defines the average ranking.

$$R_j = \frac{1}{N}\sum_{i=1}^{N} r_i^j \tag{4.4}$$

The average rankings of the algorithms are compared by the Friedman test. The Friedman statistic is defined as χ_F^2 as shown in Eq. (4.5) where N means the number of datasets and k is the number of classifiers. After the calculation of the Friedamn test statistic, the χ_F^2 value is used to calculate the p-value, and decide whether the null-hypothesis is valid, where the null-hypothesis states that all algorithms are equal, meaning there is no statistical difference between their ranking R_j.

$$\chi_F^2 = \frac{12N}{k(k+1)}\left[\sum_j R_j^2 - \frac{k(k+1)^2}{4}\right] \tag{4.5}$$

A Nemenyi post-hoc test will be performed for performance pairwise comparisons if the null-hypothesis is rejected. Critical difference (CD) is used to determine the classifiers' average ranking difference and Eq. (4.6), in which q_α is the Studentized range statistic divided by $\sqrt{2}$ [32]. In this study, with four classifiers and α =0.05, q_α =2.569, therefore, CD =1.9148. The performance difference between classifiers is plotted using CD diagrams (detailed in the experiments).

$$CD = q_\alpha\sqrt{\frac{k(k+1)}{6N}} \tag{4.6}$$

4.2.2 Experimental Results

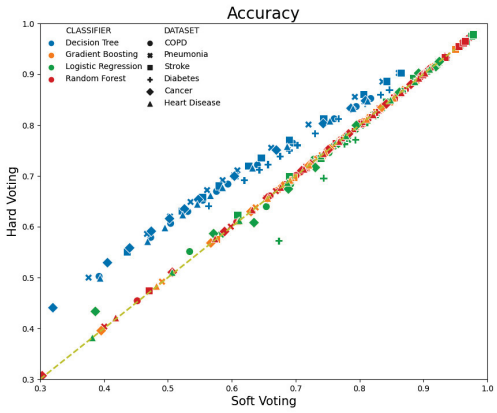
4.2.2.1 Hard Voting vs. Soft Voting Results

Fig. 4.11 compare the performance between hard voting and soft voting, with respect to four measurements, Accuracy, F1-socre, AUC, and Balanced Accuracy, on all six disease specific datasets. For each plot, the $x-$axis and $y-$axis represent the mea-

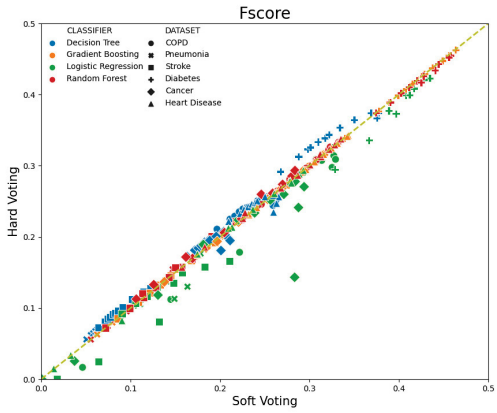
surement values of a classifier, trained using one sampling ratio and using soft voting *vs.* hard voting, respectively, on all six datasets. Because there are 12 different sampling ratios (from 0.5:1 to 5:1), four classifiers, and six disease datasets, each plot has 12×4×6=288 points. Points below the $y = x$ line are those performing better with soft voting and points above the line means hard voting outperforming soft voting. The head-to-head comparison plots allow us to directly compare soft voting *vs.* hard voting on all experimental settings and benchmark data.

The Accuracy comparisons in Fig. 4.11a show that the number of data points above and below the $y = x$ line are 167 and 121, respectively, meaning hard voting achieves better performance than soft voting, but majority of achievements are from using Decision Tree classifier. There is no obvious performance difference between soft voting *vs.* hard voting with respect to other three classifiers, Gradient Boosting classifier, Logistic Regression, and Random Forest classifier, in terms of accuracy. Ensemble models are know to benefit from unstable base classifiers, such as decision trees. Since decision trees are much more unstable than other three classifiers, the results in Fig. 4.11a confirm that using decision trees combined with hard voting can boost the classification accuracy.

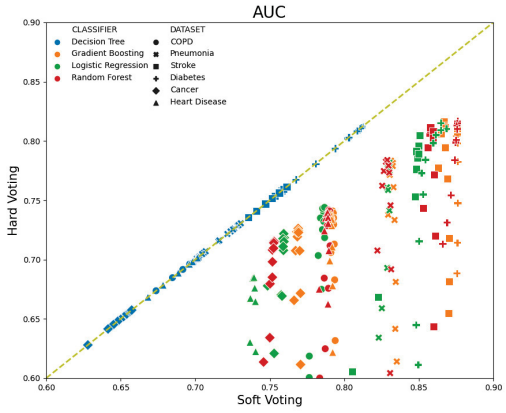
The AUC value comparisons in Fig. 4.11c show that majority points (217 points) are below the $y = x$ line, and additional 68 points are right located on the $y = x$ line (points on the $y = x$ line mean that soft voting and hard voting deliver the same prediction performance). There are only three points (288-217-68=3) that hard voting outperforms soft voting in terms of AUC values. In addition, the point color in Fig. 4.11c also show that decision trees using soft voting and hard voting have similar performance, whereas there is a significant AUC performance gain using soft voting for gradient boosting, logistic regression, and random forest. AUC is calculated by using posterior probability values of the ensemble classifier on a given test instance. Hard voting uses 0/1 frequency count to calculates final posterior probability of the



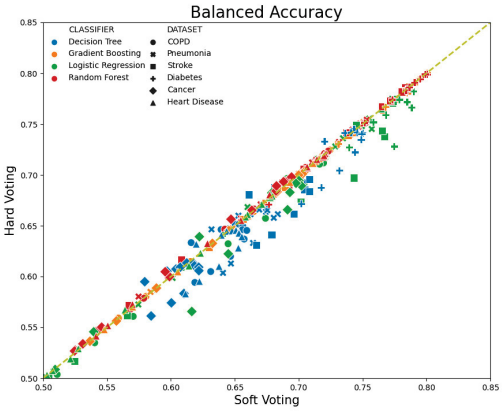
(a) Accuracy



(b) F1-score



(c) AUC



(d) Balanced Accuracy

Figure 4.11: Hard voting vs. soft voting performance on all six disease-specific datasets and 12 sampling ratios. Points are color coded by different classifiers, and shape coded by different datasets. Points above $y = x$ diagonal lines denote hard voting outperforming soft voting, and vice versa.

ensemble, whereas soft voting uses average of the base classifier's posterior probability as the ensemble classifier's posterior probability. This observation shows that for 0/1 loss based measures, such as accuracy, hard voting may outperform soft voting, whereas for continuously loss based measures, soft voting frequently outperforms hard

voting.

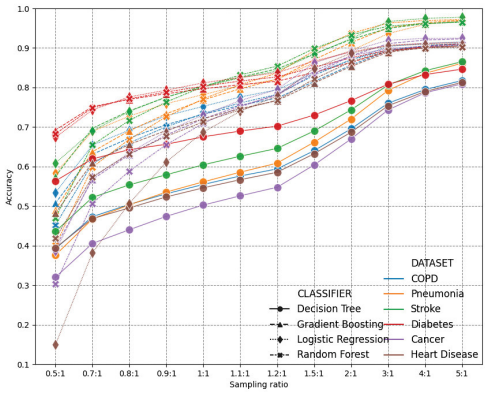
For F1-score and balance accuracy in Fig. 4.11b and Fig. 4.11d, the performance of soft voting and hard voting do not differ significantly. For F1-score, there are 137 points below the $y = x$ line, 14 less than that above the line. For balanced accuracy, 173 points are below the $y = x$ line, 58 points more than points above the $y = x$ line. Because soft voting shows better performance majority of times, and for imbalanced datasets, AUC and balanced accuracy are more objective measures, we choose soft voting in all remaining experiments.

4.2.2.2 Imbalanced Learning Results

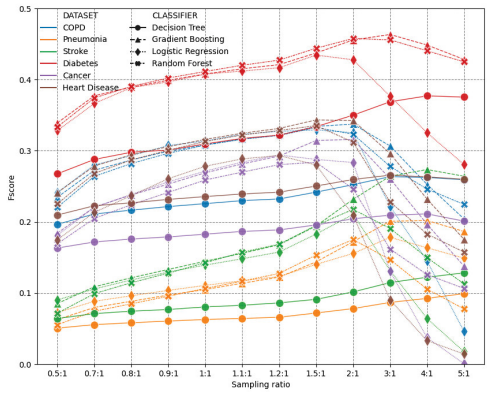
Fig. 4.12 reports the performance of all four classifiers on six disease specific datasets, using soft voting and different sampling ratios. Each plot in Fig. 4.12 reports performance measure ($y-$axis) of four classifiers on six datasets (so there are $4\times6=24$ curves in each plot), by using different sampling ratios ($x-$axis).

In the accuracy measure plot in Fig.4.12a, the larger the sampling ratio, the higher the classification accuracy each classifier achieves. This partially demonstrates the class imbalance challenge. Because sampling ratio denotes the ratio between negative *vs.* positive samples, the larger the sampling ratio (*e.g.* 5:1), the more negative samples the training set has (the ratio in the original datasets are all more than 10:1, as show in Table 4.5). Fig.4.12a shows that as negative samples gradually dominate training set, the trained classifier intends to classify more samples to be negative, in order to achieve a higher accuracy. The higher accuracy, however, does not assure useful classification results, as shown in F1-score, AUC, and balance accuracy, where all three plots show a downward/decreasing trend, after sampling ratios pass certain ratio values.

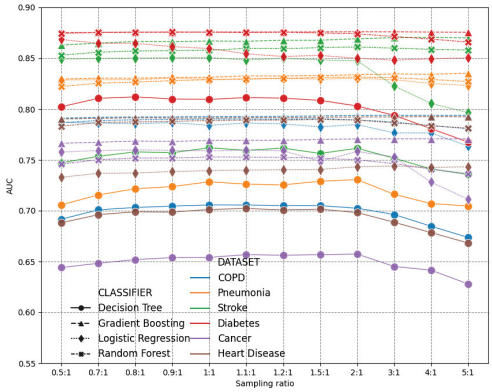
Because plots in Fig. 4.12 are color coded by different datasets, and shape coded by different classifiers, this helps understand the performance trend of each classifiers.



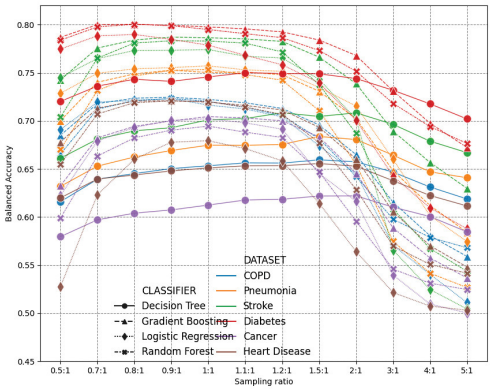
(a) Accuracy



(b) F1-score



(c) AUC



(d) Balanced Accuracy

Figure 4.12: Performance comparisons using soft voting and different sampling ratios. Points are color coded by different datasets, and shape coded by different classifiers. Each curve denote one classifier's performance on a specific dataset, using different sampling ratios.

Overall, decision trees have the worst performance in terms of all four measures. Random forest, Logistic regression and Gradient boosting are comparable with relatively small value variance, and gradient boosting shows relative better performance among the three classifiers. When comparing results of all six disease types, Diabetes (red-

colored) receive best prediction results in terms of all four performance measures. While diabetes also have the highest readmission rates among all six disease types (meaning less severe class imbalance), stroke (green-colored) has the second lowest readmission rate (Pneumonia has the lowest readmission rate). The AUC and balanced accuracy in Figs. 4.12c and 4.12d show that they both receive the best and second best prediction results. This observation indicates that the prediction results are not directly tied to the class imbalance rate. Our sampling and ensemble learning framework is effective to tackle the class imbalance. Meanwhile, the readmission prediction performance of each disease critically depends on the nature and characteristics of the diseases.

Overall, the aforementioned observations for the four measures lead to the conclusion that sampling ratio 1.1:1 presents the best performance of all classifiers on the six disease datasets. Therefore, we use 1.1:1 sampling ratio in the remaining experiments.

4.2.2.3 Readmission Prediction Results & Statistical Analysis

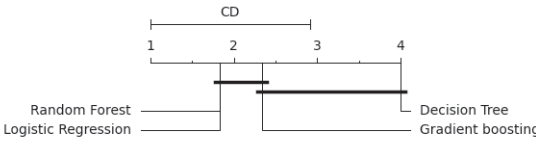
Table 4.13 reports the hospital readmission prediction results using all samples in Table 4.12, including four classifiers' average performances on the six disease specific datasets. The bold-text denotes the best result for each measure-disease combination. Overall, the results show that gradient boosting achieves the best performance.

In order to fully understand the four classifiers' performance, we carry out Friedman test for each measure, and report the critical difference diagram plots in Fig. 4.13. For all measures, we use $\alpha = 0.05$, the χ_F^2 and p values corresponding to each measure are reported as (χ_F^2, p) value pair underneath each plot. For ease of comparisons, in each plot, a horizontal bar is used to group classifiers that are not significantly different, meaning that their average ranks do not differ by CD).

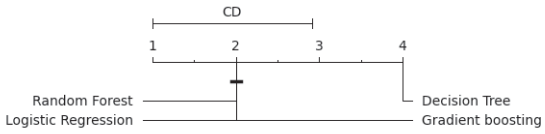
Fig. 4.13 shows that for all four measures, the largest p value is 0.0129 (which cor-

Table 4.13: Readmission prediction performance comparisons using all samples (using soft voting and 1.1:1 sampling ratio). Bold-text denotes best performance on each measure-disease combination (*i.e.* each row)

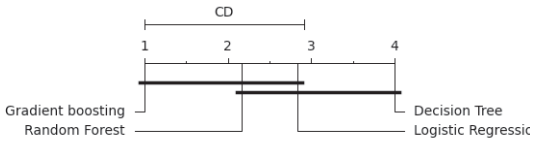
Measure	Disease	Decision Tree	Random Forest	Logistic Regression	Gradient Boosting
Accuracy	COPD	0.4659	0.7301	**0.7317**	0.7300
	Cancer	0.5260	0.7509	**0.7670**	0.7536
	Diabetes	0.6898	0.8163	**0.8249**	0.8070
	Heart Disease	0.4631	0.6983	**0.7194**	0.7025
	Pneumonia	0.5705	0.6964	**0.7262**	0.7192
	Stroke	0.6261	**0.8318**	0.8244	0.8263
F1 score	COPD	0.1791	0.2414	0.2376	**0.2415**
	Cancer	0.1866	0.2700	**0.2841**	0.2814
	Diabetes	0.3173	**0.4201**	0.4119	0.4152
	Heart Disease	0.1889	0.2350	0.2209	**0.2370**
	Pneumonia	0.2941	0.3607	0.3585	**0.3648**
	Stroke	0.0828	**0.1574**	0.1482	0.1558
AUC	COPD	0.5957	0.6767	0.6604	**0.6793**
	Cancer	0.6568	0.7527	0.7596	**0.7692**
	Diabetes	0.8113	0.8753	0.8543	**0.8758**
	Heart Disease	0.5958	0.6732	0.6406	**0.6768**
	Pneumonia	0.6919	**0.7678**	0.7542	0.7645
	Stroke	0.7594	0.8597	0.8484	**0.8667**
Balanced Accuracy	COPD	0.5687	0.6303	0.6250	**0.6304**
	Cancer	0.6176	0.6882	0.6979	**0.7030**
	Diabetes	0.7500	0.7906	0.7682	**0.7956**
	Heart Disease	0.5691	0.6168	0.5954	**0.6184**
	Pneumonia	0.6481	**0.7057**	0.6894	0.7003
	Stroke	0.7023	0.7808	0.7672	**0.7852**



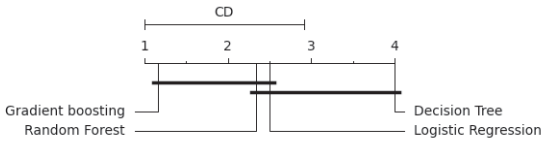
(a) Accuracy (11.4, 0.0097)



(b) F1-score (10.8, 0.0129)



(c) AUC (17.0, 0.0007)



(d) Balanced Accuracy (14.6 0.0022)

Figure 4.13: Critical difference diagram of classifiers on the six disease specific hospital readmission prediction tasks (Based on results from Table 4.9). All plots use $\alpha = 0.05$. The two numerical numbers inside the parentheses denote the χ_F^2 and p values for each plot, *i.e.,* (χ_F^2, p). Classifiers not significantly different, (*i.e.* their average ranks do not differ by CD), are grouped together with a horizontal bar.

responds to the F1-score). Because all p values are less than 0.05, the null-hypothesis (which states that all algorithms are equal and there is no statistical difference between their ranking) is rejected. This concludes that there is a statistical difference between different methods in terms of their performance ranking. Meanwhile, the χ_F^2 value shows the spread of the classifier performance. The higher the χ_F^2 value, the larger the variance of all classifiers (with respect to the current measure) is. For AUC and balanced accuracy (which are the two measures most frequently used to assess classifier performance under class imbalance), the gradient boosting outperforms, random forest and logistic regression, with random forest outperforming logistic regression, in terms of their mean rankings. Also, although these three classifiers have different mean rankings, their performance are not statistically different. In summary, the critical difference diagrams in Fig. 4.13 concludes that gradient boosting achieves the best average ranking among all models, whereas decision tree has the

lowest ranking.

4.3 CONCLUSIONS

This research carries out systematic studies to understand data statistics for United States nationwide hospital admission, and further designs a machine learning framework for disease-specific 30-day hospital readmission prediction. We argued that although many methods exist for hospital readmission prediction, answers to some key questions, such as demographic, disease, and hospital characteristics with respect to admissions, still remain open. Accordingly, we employed national readmission databases (NRD), with over 15 million hospital visits, to carry out data statistics analysis. We identified factors related to three key party of the hospital remissions: patient, disease, and hospitals, and reported national scale hospital admission statistic. Based on the data statistics, we created 526 features with five major types, including demographics features, admission and discharge features, clinical features, disease features, and hospital features. We collected six disease specific readmission datasets, which reflect the top six leading diseases of death.

By using random under sampling and ensemble learning, combined with soft *vs.* hard voting and four types of machine learning methods, including gradient boosting, decision tree, logistic regress, and random forests, our experiments validate three major type of settings: (1) hard voting *vs.* soft voting, (2) random under sampling, and (3) disease specific readmission prediction. Experiments and statistical test results show that soft voting outperforms hard voting on majority results, especially for AUC and balanced accuracy which are the main measures for imbalanced data. Random under sampling using 1.1:1 for negative:positive ratio achieves the best performance for AUC, balanced accuracy, and F1-score. Gradient boosting achieves the best performance for disease specific hospital readmission prediction, and decision trees have the worst performance.

CHAPTER 5

FEDERATED LEARNING USING DYNAMIC NODE ALIGNMENT

Federated Learning (FL), originally proposed in 2016 [67], is a learning paradigm which builds machine learning models based on datasets distributed across multiple sites/devices in order to protect privacy and prevent data leakage. While traditional machine learning methods are typically trained based on centralized data, using FL provides a feasible way to develop models that can keep all the training data on distributed devices and update model parameters using immediate aggregation.

As data collection and analytics are becoming increasingly popular, protecting data privacy and safety is becoming a major concern for business, government, and nearly all sections of human society. By deploying FL, each participant in the model training process can build one model together without sharing data, naturally results in data privacy protection. Traditional machine learning methods need to concentrate training data in a certain machine or a single data center, which means in order to meet the gradually increasing data level, it is necessary to continuously add machines and build infrastructure. Such method not only greatly increases the cost but also hinders the efficiency building models. In contrast, FL allows all the needed data stay in their local places without the need to build specific data center to aggregate them, at the same time, each part of the data will be used to develop the model. Such efficient characteristic enables Federated Learning to be widely used in multiple areas especially in the healthcare domain.

The shift from written health records to electronic health records has been instrumental in driving the use of patient data to improve the healthcare industry. The adoption of electronic health records enables health care professionals to disseminate

knowledge across all sectors of health care, which in turn helps to reduce medical errors and improve patient care and satisfaction. However, as mentioned previously, adequate medical data sets are difficult to obtain. However, in order to capture the subtle relationships between disease patterns, socioeconomic and genetic factors, and complex and rare cases, exposing the model to different cases is critical. FL is able to address this issue by enabling the distributed training of machine learning models using remotely hosted datasets without the need to accumulate data and therefore compromise the data privacy [10, 17, 70, 70, 74, 79, 102].

While FL is capable of making use of data across different sites/institutions, there are still several data acquisition issues which can cause bias during model develop process. First of all, due to data privacy limitation, the Health Insurance Portability and Accountability Act (HIPAA) has set up regulations for healthcare organizations to manage and safeguard personal information and address their risks and legal responsibilities in relation to processing personal patients data [22]. This leads to strict data share policies of each healthcare provider, which, limits the amount of available data source. Another issue is that there exist hospital speciality gaps between different hospitals, in other words, healthcare providers might focus on several particular diseases treatment instead of performing general hospitalization. In this case, there are big chances where FL models trained across all different disease focus datasets will perform predictions with certain disease-specific bias. In addition, biases also exist when patients demographic characteristics differ. Different income groups, age groups, genders, and geographical locations and living environments will all affect the overall patient characteristics that admitted to different regional hospitals, thus, data bias can also be observed in such kind of dissimilarity. Therefore, it is essential to reduce all the above biases when we try to develop a federated learning model to make crucial medical predictions. We aim to design a novel federated learning model that can take this kind of bias into consideration at the first step where node weight

aggregation takes place.

Despite the fact that use of traditional machine learning techniques (TML) in combination with electronic health records (EHR) is gaining popularity as a means to extract knowledge that can improve decision-making processes in healthcare, they usually require the training of high-quality learning models based on diverse and comprehensive datasets that are difficult to obtain due to the sensitivity of medical data from patients. Meanwhile, although distributed machine learning [95] has addressed parallel computing in handling large scale data, these methods are often designed to tackle the data volumes using frequent data exchange. In addition, switch learning models are often prohibitively expensive/inconvenient, making it difficult for end users to try/implement different learning algorithms. On the contrast, FL enables devices to collaboratively learn shared predictive models while keeping all training data on-device, decoupling the power of machine learning from the need to store data in the cloud. This goes beyond using native models to make predictions on mobile devices and also brings model training to the device.

Table 5.1 summarizes the main difference between federated learning, traditional machine learning methods, and distributed machine learning methods. In summary, the inherent advantage of federated learning is that is allows flexible modeling training and continuous learning on end-user devices while ensuring no end-user data leaves the device.

Fig. 5.1 shows how FL works. Global model M is downloaded from the central server to each client when it comes to training the model, after which the downloaded model is trained by each client using their own dataset. Once the training process is completed, each client needs to update their updated training parameters to the central server and the central server would aggregate the learnt parameters (parameter aggregation) and pass the aggregation results to the global model, therefore, one update for the global model is accomplished and this process is called Global update.

Table 5.1: Comparison between Federated Learning (FL), traditional machine learning (TML), and distributed machine learning (DML) algorithms. DML methods are commonly data driven (DML_d) or computing driven (DML_c). Data driven methods (DML_d) mainly try to learn from large volume distributed data, whereas computing driven methods (DML_c) aim to parallelize computing in learning from centralized data. *Computing framework* refers to the whole eco-system for learning, and *model switch* refers to easiness of switching a new learning model.

Method	Data location	Main computing	Computing framework	Data exchange	Main Challenge	Privacy protection	Model switch
TML	Centralized	Data center	Very restrictive	Yes	Model performance	Low	Very restrictive
DML_c	Centralized	Data center	Restrictive	Yes	Data volume	Low	Very restrictive
DML_d	Distributed	Local	Flexible	Yes	Data volume	Medium	Restrictive
FL	Distributed	Local	Very flexible	Prohibited	Data protection	High	Flexible

Once global update is finished, model parameters will be passed from the global model to each local model for Local update, where clients' model parameter will be updated with the new aggregated model weights to start a new round training. [74].

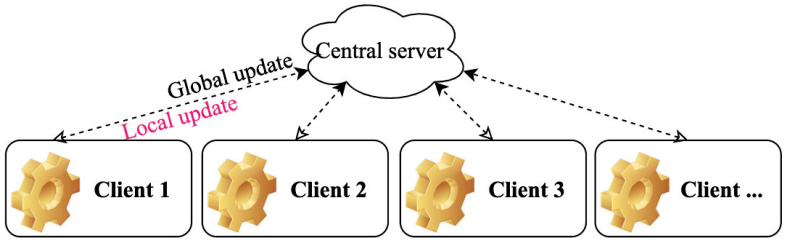


Figure 5.1: A conceptual view of the FL Framework. The local update (downstream) and global update (upstream) are carried out iteratively to ensure models trained using local data are aggregated at central server, and then dispatched to distributed sites.

Parameter aggregation is one of the most important steps of the federated learning. Among all existing methods, Federated Averaging (FedAvg) is the most commonly used method. Eq. 6.13 summarizes the global weight values $\mathbf{w}$ updating of FedAvg in each training round t, in which k is the client index, K means the total number of

clients, n is the total number of instances and n_k is the local data examples for each client [67]. Overall, Eq. 6.13 indicates that the global weight vector $\mathbf{w}$ is the weighted average of weight values obtained from local clients. A bold-faced symbol denotes a vector or a high dimensional vectors (*e.g.* a matrix).

$$\mathbf{w}_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} \mathbf{w}_t^k \tag{5.1}$$

Recently, other weight aggregation methods have also been proposed in FL. For example, anomaly score of each client is taken into consideration to detect abnormal client behavior, thus, clients will not contribute equally when global model updates the weight values, the majority of those novel methods are still based on FedAvg [36, 59]. Even though this method is widely used and has been proved with good prediction performance [34, 90], due to the nature of hidden layers in deep learning neural networks, we can clearly observe that this method manually forces weight aggregations between neurons located at the exact same location (*i.e.*, same layer and same node index) of two networks. However, when training two same-structured deep learning networks N_A and N_B, even they are given the same input, neurons at the same location of the two networks do not always give the same update. In other words, certain property of the input (or the same instance) may trigger the most significant activation to the i-th node of N_A, but same instance may triger the most significant activation to the j-th node of N_B. Meaning that same instance responds differently for the same lactation nodes between two networks.

In order to demonstrate the above hypothesis, we create a simple dense neural network N_D with one input layer, two hidden layers and one output layer. One dataset with 10 features is fed into N_D. For the i^{th} node in the first hidden layer N_D^1, there will be 10 weight values $\{w_{i,0}^1, w_{i,1}^1, ..., w_{i,9}^1\}$ corresponding to the 10 input features (the superscript denotes the first trained network). After we train N_D from scratch for five times with the exactly same dataset, a node e is randomly chosen from all five networks (with the same node index), from which we will get 10 weight

vectors of $\{\mathbf{w}_{e,0}, \mathbf{w}_{e,1}, \cdots, \mathbf{w}_{e,9}\}$ in which $\mathbf{w}_{e,0} = [w^1_{i,0}, w^2_{i,0}, w^3_{i,0}, w^4_{i,0}, w^5_{i,0}]$ represents all five trained weight values corresponding all five networks' first indexed node and first feature dimension as shown in Fig. 5.2. After that, we calculate the variance of $\mathbf{w}_{e,0}$, and repeat the same for all 10 nodes. Fig. 5.3 reports the variance of the weight values across all five trained network. The high weight variance in Fig. 5.3 concludes that weight aggregation by static node matching will not only add uncertainty to model performance, but also will hinder the practical application of Federated Learning in industry.

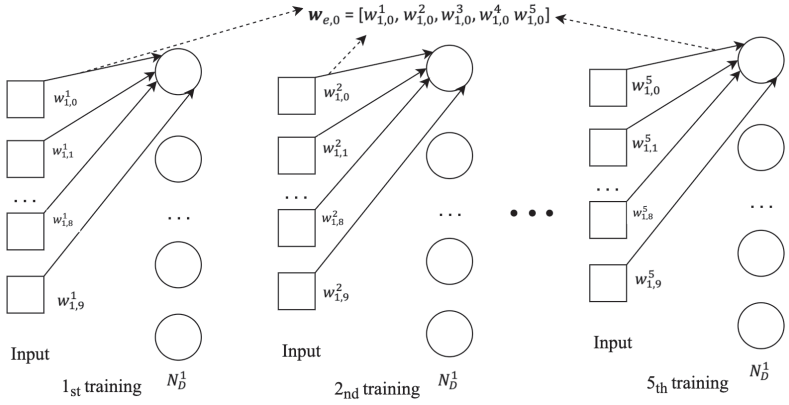


Figure 5.2: A conceptual view of node weight variance calculation. Five neural networks with the same architecture are trained using same training sample. The first hidden layer nodes are trained with the same input features and the first node is chosen to calculate the node variance.

In this research, we aim to design a dynamic node matching method, FedDNA, to aggregate weight values in each round based on a neuron-distance method, in which neuron distances across all the clients are calculated after each client completes training the model parameters with their own data. After that, the closest neurons are matched to calculate their average weight values as new parameter for the global model. Fig. 5.3 reports weight variance of the matched nodes trained using same setting as the static node matching. The results show that dynamic node matching provides much smaller weight variance across all nodes of different networks. The

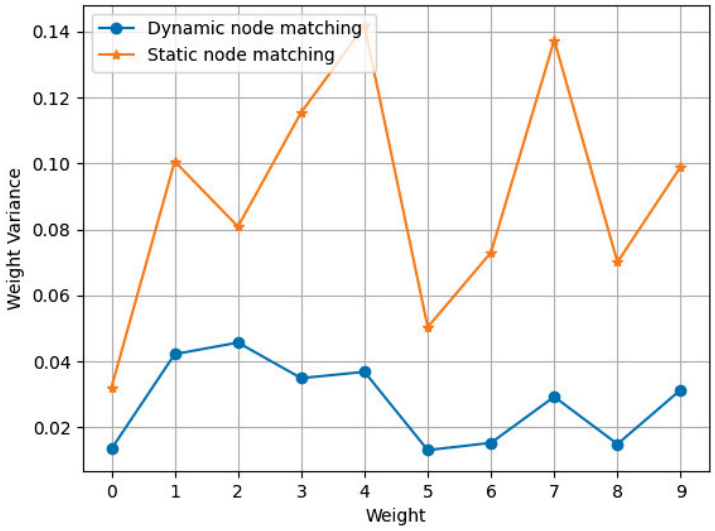


Figure 5.3: Comparisons of weight variance between two weight matching methods, *Static node matching* vs. *Dynamic node matching* (proposed). The x-axis denotes the neuron node ID of the first hidden layer, consisting of 10 neurons, of a neural network. The network was trained five times till convergence, using same training data. The y-axis denotes the variance of the weight values of each of the hidden nodes (Larger variance mean the neuron weights are more unstable across different training times, even for the same feature dimension of the same neuron).

advantage of reducing variance is that it allows nodes with similar behaviors to be aggregated for weight averaging. This potentially results in stable and improved federated learning performance.

In summary, the main contribution of the proposed research is summarized as follows:

- **Dynamic node alignment:** We propose a new dynamic node alignment framework, `FedDNA`, for weight aggregation in federated learning. Instead of using fixed node index to match nodes across different sites, `FedDNA` finds the best matching nodes based on node weight values, such that nodes, of the same

layer, with the most similar response to the input are considered as one new node for next round training.

- **Fast node alignment:** To increase node aliment speed, we propose a Minimum Spanning Tree (MST) based method to find global optimal matching nodes across all sites.

- **Alignment and frozen:** In each training process, after finding the matching nodes at the very beginning, node matching will be frozen and federated average will be used for the rest of training rounds. By doing this, we can ensure the matching nodes orders which will not be disturbed by subsequent training.

5.1 THE PROPOSED METHOD

Instead of using fixed node matching, like FedAvg does, we propose to use dynamic node matching to find matching node between different sites, and then aggregate weights of matched nodes to calculate weight values of the global model. During the FL process, the sites will pass their local weight values to the center, so the center will carry out node matching before aggregating site weight values. Our idea is to use weight values of each node as a feature vector to find matching nodes. Because weight values of a neuron are associated to each features, for nodes at the same hidden layer, they will have same input space. This allows us to use weight values to find distance/similarity between nodes for matching.

To make sure weight values are aggregated from the most similar nodes crossing all clients C, at the first step, nodes distances are calculated across all clients as shown in the distance matrix in Table. 5.2, from which Minimum Spanning Tree (MST) as shown in Fig. 5.4 is used to ensure that the matching are across all clients. A minimum spanning tree (MST) or minimum weighting tree is a subset of edges of a connected edge-weighted undirected graph that joins all vertices together without any

loops and with the smallest possible total edge weights. That is, it is a spanning tree with the smallest possible sum of edge weights. More generally, any edge-weighted undirected graph (not necessarily connected) has a minimum spanning forest, which is the union of the minimum spanning trees of its connected components [38]. In our example in Table 5.2, a distance mapping is plotted to demonstrate how the matching process works. At first node distances are calculated across all sites, in this case, 3 sites. we start the matching process from node a because it has the smallest distance 0.11 across all the nodes. we can observe that node B has the smallest distance with it, therefore, B will be matched to a. For the next step, we are using MST to find the next matching node for $\{a, B\}$, which in this case, will be node α. This MST matching process will continue until all the nodes are matched across all clients as shown in Fig. 5.4.

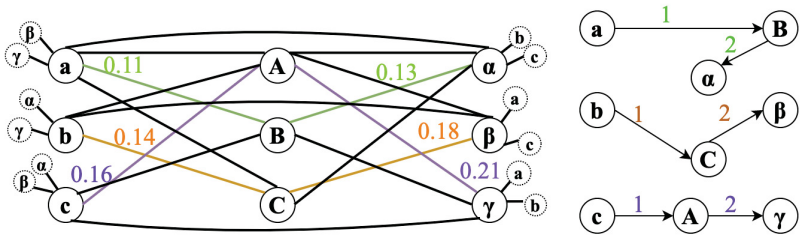


Figure 5.4: Node matching using MST. Node a is the starting point since it has the smallest distance 0.11 with node B, therefore, B will be matched to a. Node α will be matched with a, B with MST. This MST matching process will continue for node b and c.

5.1.1 Dynamic Neural Network Node Matching

In the proposed method, one key step is to find the closest nodes based on distance calculation in each round. This step is carried out at the center, and the aggregated weights are then dispatched to the federated learning site for the next round. The node matching is applied to one specific hidden layer of all networks, one at a time.

Table 5.2: An example of pairwise distance tables between three sites where each site has three nodes: Site$_1$=$\{A, B, C\}$, Site$_2$=$\{a, b, c\}$, and Site$_3$=$\{\alpha, \beta, \gamma\}$. Each value in the table denotes distance between two nodes across two sites.

	a	b	c
A	0.12	0.15	0.16
B	**0.11**	0.13	0.17
C	0.16	0.14	0.18

	A	B	C
α	0.27	0.13	0.19
β	0.23	0.14	0.18
γ	0.21	0.16	0.21

	a	b	c
α	0.13	0.24	0.18
β	0.14	0.19	0.21
γ	0.21	0.21	0.25

By default, we are referring to nodes in the first hidden layer for ease of explanation. The same matching process is applicable to any other hidden layers as well. Algorithm 4 outlines the main steps of FedDNA for matching nodes across networks. Overall definition of the symbols used in our node matching is shown in Table. 5.3.

Table 5.3: Definition for symbols used in node matching

Symbol	Definition
$\mathbf{S}$	Global model (server)
$\mathbf{v}_i^s$	Weight vector of the i-th node in global model first hidden layer
$\mathcal{C}$	Set of clients
$\mathbf{c}^\alpha$	Nodes weight vector of client $\mathbf{c}$
$\mathbf{v}_i^\alpha$	Node weight vector of client $\mathbf{c}^\alpha$'s i-th node
$w_{i,d}^\alpha$	Weight values of node $\mathbf{v}_i^\alpha$
$d(a, b)$	Distance between node a and node b
$\mathcal{T}$	Minimum spanning tree
$d(\mathbf{v}, \mathcal{T})$	Distance between node $\mathbf{v}$ and tree $\mathcal{T}$

Denote $\mathbf{S} = \{\mathbf{v}_1^s, \mathbf{v}_2^s, \cdots, \mathbf{v}_n^s\}$ the global model (server) in which $\mathbf{v}_i^s = [w_{i,0}^s, w_{i,1}^s, \cdots, w_{i,m}^s]$ is the weight vector of the i-th node in its first hidden layer. $\mathcal{C} = \{\mathbf{c}_1, \mathbf{c}_2, \cdots, \mathbf{c}_\Sigma\}$

means the set of clients $\mathbf{c}$ and $\mathbf{c}^\alpha = \{\mathbf{v}_1^\alpha, \mathbf{v}_2^\alpha, \cdots, \mathbf{v}_n^\alpha\}$ is nodes weight vector of client $\mathbf{c}^\alpha$. Node weight vector of client $\mathbf{c}^\alpha$'s i-th node is denoted by $\mathbf{v}_i^\alpha = [w_{i,0}^\alpha, w_{i,1}^\alpha, \cdots, w_{i,m}^\alpha]$.

5.1.1.1 Neuron Matching Distance Calculation

Given two neurons $\mathbf{v}_i^\alpha$ and $\mathbf{v}_j^\beta$ at the same layer, because they have the same input dimensions (In this paper, we are using dense network architecture, so neurons at the same layer are connecting to all inputs/nodes of the preceding layer), we can represent each nuron as a vector, and calculate distance/similarity between neurons using the vectors.

Assume for any particular layer, the input dimension is m, and the weight values of neuron $\mathbf{v}_i^\alpha = [w_{i,0}^\alpha, w_{i,1}^\alpha, \cdots, w_{i,m}^\alpha]$, weight values of neuron $\mathbf{v}_j^\beta = [w_{j,0}^\beta, w_{j,1}^\beta, \cdots, w_{j,m}^\beta]$, respectively. Node distance between $\mathbf{v}_i^\alpha$ and $\mathbf{v}_i^k$ can be calculated with Euclidean distance defined in Eq. 5.2 or using Manhattan distance defined Eq. 5.3. The Euclidean distance between two points in Euclidean space is defined as the length of the line segment between the two points, which essentially represents the shortest distance between two points. Manhattan distance is a distance measure between two points in an m-dimensional vector space. It is the sum of the projected lengths of the line segments between the points on the coordinate axes. In simple terms, it is the sum of the absolute differences of two points measured in all dimensions.

$$d_{Euclidean}(\mathbf{v}_i^\alpha, \mathbf{v}_j^\beta) = \sqrt{\sum_{d=1}^{m}(w_{i,d}^\alpha - w_{j,d}^\beta)^2} \tag{5.2}$$

$$d_{Mahattan}(\mathbf{v}_i^\alpha, \mathbf{v}_j^\beta) = \sum_{d=1}^{m}|w_{i,d}^\alpha - w_{j,d}^\beta| \tag{5.3}$$

During the node matching process, we will be growing a tree (*i.e.* a minimum spanning tree MST) to link matched/aligned nodes across all sites. In this case, a tree $\mathcal{T}$ consists of a set of neurons, *i.e.,* $\mathcal{T} = \{\mathbf{v}_i^\alpha, \cdots, \mathbf{v}_j^\beta, \cdots\}$ where $\alpha \neq \beta$. We enforce $\alpha \neq \beta$ such that an MST tree only contains one node from each site (because we are trying to find

Algorithm 2 FedDNA: Federated learning with dynamic node alignment

Input: The index of the layer to apply node alignment (default first hidden layer); Client models' node weight values; Chosen node set $\mathcal{N}$={}; Matched node set $\mathcal{M}$={}; Chosen Client set $\mathcal{C}$={}

Output: Aggregated weight values of the global model **S**

$\mathcal{C} \leftarrow$ Set of federated learning participating clients

$\mathcal{N} \leftarrow$ Node set of the l^{th} hidden layer of participating clients ($\mathcal{C}$)

$\mathbf{W} \leftarrow$ Obtain l^{th} layer's weight values from participating clients ($\mathcal{C}$)

$\mathbf{c}^{\alpha} \leftarrow$ Randomly select one client from the client set $\mathcal{C}$

$\{\mathbf{v}_1^{\alpha}, \mathbf{v}_2^{\alpha}, \cdots, \mathbf{v}_n^{\alpha}\} \leftarrow$ Obtain client $\mathbf{c}^{\alpha}$'s layer l node weight vectors

$\mathcal{M} \leftarrow \emptyset$; $\mathcal{R} \leftarrow \mathcal{N}$ Initialize matched node set ($\mathcal{M}$) and remaining unmatched node set ($\mathcal{R}$)

$\overline{\mathcal{W}} \leftarrow \emptyset$ Initialize set ($\overline{\mathcal{W}}$) storing mean weight values of matched nodes across all sites

while $\mathcal{R}$ is not empty loop until all nodes are matched **do**

    $\mathcal{C}' \leftarrow \mathcal{C}$ A temporary set to ensure that each site has one node being matched, one at a time

    $\mathbf{c}^{\alpha} \leftarrow$ Randomly select one client from the client set $\mathcal{C}$

    $\mathbf{v}_i^{\alpha} \leftarrow$ Randomly select on neuron of $\mathbf{c}^{\alpha}$ from remaining node set $\mathcal{R}$

    $\mathcal{T} \leftarrow \{\mathbf{v}_i^{\alpha}\}$ Initialize MST tree for matching

    $\mathcal{R} \leftarrow \mathcal{R} \setminus \mathbf{v}_i^{\alpha}$ Exclude $\mathbf{v}_i^{\alpha}$ from remaining node set $\mathcal{R}$

    $\mathcal{C}' \leftarrow \mathcal{C} \setminus \mathbf{c}^{\alpha}$ Exclude selected site $\mathbf{c}^{\alpha}$, because its node already in the tree $\mathcal{T}$

    while $\mathcal{C}'$ is not empty *loop until all sites are matched **do**

        $[v^*,\ c^*] \leftarrow \underset{v* \in c^*;\ c^* \in \mathcal{C}'}{\arg\min}\ d(v^*, \mathcal{T})$ find node v^* most closest to the MST tree $\mathcal{T}$

        $\mathcal{T} \leftarrow \mathcal{T} \cup v^*$ Include matched node v^* to the tree $\mathcal{T}$

        $\mathcal{C}' \leftarrow \mathcal{C}' \setminus c^*$ Exclude site c^*

        $\mathcal{R} \leftarrow \mathcal{R} \setminus v^*$ Exclude v^* from remaining node set $\mathcal{R}$

    end while

    $\overline{\mathbf{w}} \leftarrow \text{Average}(\mathbf{v}_k); \forall v_k \in \mathcal{T}$ Calculate average weight values of matched notes in $\mathcal{T}$

    $\overline{\mathcal{W}} \leftarrow \overline{\mathcal{W}} \cup \overline{\mathbf{w}}$ Center collects average weights of matched nodes across all sites

    $\mathcal{M} \leftarrow \mathcal{M} \cup \mathcal{T}$ Include all MST tree nodes to the matched set $\mathcal{M}$

end while

for each client $\mathbf{c}^{\alpha} \in \mathcal{C}$ **do**

    $\mathbf{W} \leftarrow \text{ClientUpdate}(\mathbf{c}^{\alpha}, \overline{\mathcal{W}})$ Dispatch mean weight values to each site for next round federated learning (Alg. 3)

end for

Algorithm 3 ClientUpdate(c,**w**): Local client weight updating

Input: **w**: trainable model parameters; D_c: local data at site c; (2) b: batch size; (3) e: Number of epochs; (4) η: learning rate

Output: **w**: updated local model parameters

$\mathcal{B} \leftarrow$ split local data D_c into batches of size b
for each epoch from 1 to e **do**
    for batch $\mathbf{b} \in \mathcal{B}$ **do**
        $\mathbf{w} \leftarrow \mathbf{w} - \eta \bigtriangledown \iota(\mathbf{w}; \mathbf{b})$
    end for
end for
Return **w**

matching nodes across all sites. It does not make sense to have a neuron to match a node of the same network). The number of nodes in the tree $\mathcal{T}$ varies, as the tree is growing dynamically. However, after the matching, each node only belongs to one MST tree, and the final number of nodes in the MST tree equals to the number of sites of the FL learning framework. We do not record edges connecting nodes in the tree, because our goal is to find matching nodes as a group, and then use their weights to update center's node weights. In this case, the pairwise relationship between sites is not important to us. Also, each tree $\mathcal{T}$ records its membership nodes and will use their weights to calculate the average weights, which will be pass to respective members of the tree $\mathcal{T}$ for next round FL learning.

During node matching, we need to expand the tree $\mathcal{T}$ and include matching node to the tree. Therefore, we define the distance between a node $\mathbf{v}$ and Minimum Spanning Tree $\mathcal{T}$ as in Eq. 5.4. The distance from a node to a Minimum Spanning Tree tree $\arg\min\ d(\mathbf{v}, \mathcal{T})$ equals to its distance to its closest node in the tree.

$$d(\mathbf{v}, \mathcal{T}) = \underset{\mathbf{v}^\alpha \in \mathcal{T}}{\arg\min}\, d(\mathbf{v}, \mathbf{v}^\alpha) \tag{5.4}$$

5.1.1.2 Minimum Spanning Tree for Neuron Alignment across Sites

At the first step, each client downloads the model from central server and train it with its local data, after which client $\mathbf{c}^{\alpha}$ is randomly chosen from $\mathcal{C}$. One node $\mathbf{v}_i^{\alpha}$ will be randomly selected among all the nodes in the first hidden layer of client $\mathbf{c}^{\alpha}$'s local model. In the second step, another client $\mathbf{c}^k$ will be chosen at random from $\{\mathcal{C}$ - $\mathbf{c}^{\alpha}\}$. A distance function explained previously will be used to calculate the distance $d(\mathbf{v}_j^k, \mathbf{v}_i^{\alpha})$ between all nodes in the first hidden layer of client $\mathbf{c}^k$ model and node $\mathbf{v}_i^{\alpha}$. We can get two nodes matched $(\mathbf{v}_j^k, \mathbf{v}_i^{\alpha})$ based on the smallest distance.

Now we have two nodes, which are also the start of our MST tree $\mathcal{T} = \{\mathbf{v}_j^k, \mathbf{v}_i^{\alpha}\}$, from which we will start to grow the tree. MST is the one whose cumulative edge weights have the smallest value, and in our proposed method, it means the one whose cumulative node distances have the smallest value. In each matching step, we will randomly pick one client from $\{\mathcal{C}$ - $\{\mathbf{c}^{\alpha}, \mathbf{c}^k\}\}$. Node to tree distance Eq. 5.4 will be applied to find the subsequent matching nodes to join the MST tree $\mathcal{T}$. The MST tree $\mathcal{T}$ will continue growing until $\{\mathcal{C}$ - $\{\mathbf{c}^{\alpha}, \mathbf{c}^k$,...$\}\}$ is empty and at the same time, a complete tree $\mathcal{T}$ with new node set $\{\mathbf{v}_i^{\alpha}, \mathbf{v}_j^k, \mathbf{v}_t^{\beta}$,...$\}$ will be formed to aggregate their averaged weight values as a new node weight $\mathbf{v}_i^s$ for the global model. To illustrated the above description, for example, one client is randomly chosen in Fig. 5.5, then in Fig. 5.6, after the first calculation, node $\{a, B\}$ are matching node, then we calculate distance $d(a, \alpha)$, $d(a, \beta)$, $d(a, \gamma)$, $d(a, \theta)$, $d(B, \alpha)$, $d(B, \beta)$, $d(B, \gamma)$, $d(B, \theta)$, then choose node α with the smallest distance and node $\{a, B, \alpha\}$ are the matching nodes. Weight values $\{\mathbf{v}_a^1, \mathbf{v}_B^2, \mathbf{v}_{\alpha}^3\}$ will be averaged to be considered as a new node value for the global model.

5.1.1.3 Dynamic Node Alignment vs. Frozen

In our proposed method, frozen means instead of using dynamic node alignment through the entire training process, we choose to train the federated learning model

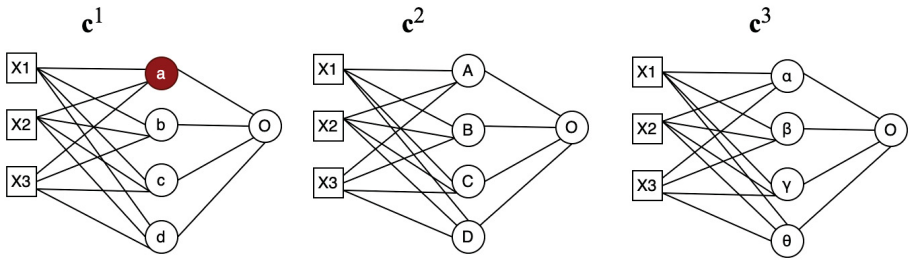


Figure 5.5: Node matching step 1. after each client finishing training its local model, client $\mathbf{c}^{\alpha}$ is randomly chosen from $\mathcal{C}$ and node $\mathbf{v}_i^{\alpha}$ will be randomly selected among all the nodes in the first hidden layer of client $\mathbf{c}^{\alpha}$ local mode.

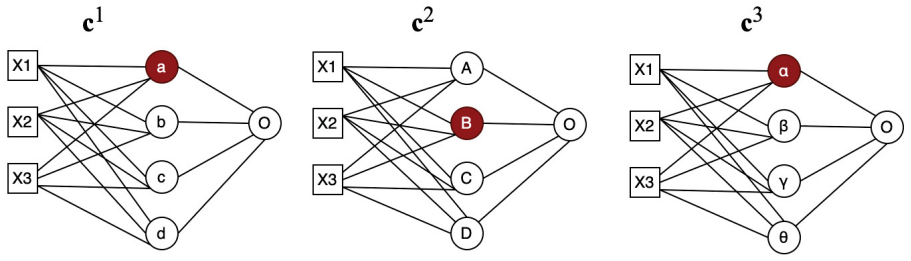


Figure 5.6: Node matching result. node $\{a, B, \alpha\}$ are matching nodes.

with dynamic node alignment for certain rounds at the very first beginning, then static node alignment will be applied for the rest training part. By doing so, nodes with similar response will be paired right after the training process starts and once all the neurons are matched during the first certain rounds, we believe that the node pair pattern will discovered and fixed to a certain extend, therefore, using static node alignment will prevent the pattern from being disturbed from subsequent training process.

5.1.1.4 Theoretical Analysis

In this subsection, we analyze the time complexity of `FedDNA`, and compare its complexity with simple global optimal matching search. Denote Σ the number of sites, n the number of first layer nodes at each site, and m the number of features for each neuron. Because all sites in FL setting have same network structure, we only focus on

first layer, and the same analysis applies to other layers, if dynamic node alignment is carried out beyond the first layer.

Finding global optimal matching (*i.e.*, the sum of matching distances of all nodes, across all sites) is a combinatorial problem, because it requires comparisons of all nodes against all other nodes, across all sites. For two sites, each having n nodes, the matching complexity is $Øn \times n \times m$, because it needs to cross check all pairs (and each pairs involve m feature dimension comparisons). Adding a third site would result in $Øn \times n \times n \times m$ complexity because all node pairs between three sites need to be checked. As a result, for Σ sites the total complexity is $Øn^{\Sigma} \times m$, which grows exponentially with respect to the number of sizes.

For `FedDNA`, finding matching nodes across all sites for one node requires $Ø(\Sigma - 1) \times n \times m$ complexity because a node needs to search all nodes from other sites and it does not need to search nodes from the same site. Once the first node is matched (across all sites) and matched nodes are added to the minimum spanning tree (MST), the next node matching requires $Ø(\Sigma - 1) \times (n - 1) \times m$ complexity because there are $n - 1$ unmatched nodes remain for each site. As a result, total time complexity for all nodes (across all sites) is the sum of all individual nodes' complexity: $Ø(\Sigma - 1) \times n \times m + Ø(\Sigma - 1) \times (n - 1) \times m + \cdots + Ø(\Sigma - 1) \times 1 \times m = Ø\Sigma \times n^2 \times m$. By growing minimum spanning tree (MST) to support the matching, `FedDNA` reduces the exponential complexity from $Øn^{\Sigma} \times m$ (for global optimal matching) to quadratic $Ø\Sigma \times n^2 \times m$.

In summary, `FedDNA`'s complexity is linear with respect to the number of sites, and quadratic with respect to the number of nodes at each site.

5.2 EXPERIMENTS

5.2.1 Experimental settings

5.2.1.1 Datasets

We used four benchmark datasets in the experiments. The first one is Diabetes Data Set which data source are obtained from two main aspects, an automatic electronic recording device and paper records to predict whether a patient has diabetes or not. For the automated electronic recording devices, they have an internal clock to mark events, whereas paper records only provide periods of "logical time" (breakfast, lunch, dinner, bedtime). For paper records, fixed times are assigned to breakfast (08:00), lunch (12:00), dinner (18:00) and bedtime (22:00). Therefore, paper records have a fictional uniform time of recording, while electronic records have a more real time stamp [53]. The second dataset we used is Spam_base Data Set from UCI which was created by spam emails from postmaster and individuals and non-spam emails from filed work and personal e-mails in order to construct a personalized spam filter. In this dataset, The last column indicates whether the email is considered spam (1) or not (0), that is, unsolicited commercial email. Most properties indicate whether a particular word or character occurs frequently in emails [47]. Another data set used in this paper is called Patient Survival Prediction Dataset. It uses knowledge about patient chronic conditions from Intensive Care Units (ICUs) to inform clinical decisions about patient care and ultimately predict patient's survival outcomes [75]. Occupancy Detection Data Set is the last data set we used to verify our model's performance. It is a dataset for predicting room occupancy using environmental factors such as Temperature, Humidity, Light and CO2. Ground-truth occupancy obtained from time stamped pictures that were taken every minute [20].

Basic descriptions about these four datasets are shown in Table 5.4 from which we can observe the number of samples in each dataset, diabetes database has 1150

samples and there are 4601, 91714 instances in the spambase data set and Patient Survival Prediction data set separately. Patient Survival Prediction set has the most samples and also the largest attributes while Occupancy detection data set has the medium size samples with the least number of attributes. Apart from that, data dimensions of those four are also different with various feature types such as categorical features, numerical features. One same point is that there are only two classes in all the datasets, which means binary classification will be performed in our proposed model.

Table 5.4: Summary of the benchmark datasets used in the experiments, including sample amount, attributes amount, data characteristics and class distribution.

Dataset	# of instances	# of attributes	Attribute Characteristics	Class	Class Distribution	Class set up
Diabetes Database	1,150	19	Categorical, Integer	Binary	0.89	0.2; 0.4; 0.5; 0.6; 0.7
Spambase Data Set	4,601	56	Integer, Real	Binary	1.54	0.8; 0.6; 0.5; 0.4; 0.2
Patient Survival Prediction	91,714	186	Categorical, Real	Binary	11.26	0.5; 1; 3; 5; 7
Occupancy Detection	20,560	7	Real	Binary	3.33	0.5; 1; 1.5; 2; 2.5

5.2.1.2 Baseline Methods

To validate the performance of the proposed method, we use deep neural networks as the training models and employ four baselines for our comparisons. One is plain neural network (Plain_NN) model which has the same structure as our proposed model which has one input layer, two hidden layers and one output layer.

FedAvg The second baseline is called Federated Average (FedAvg), which also share the same network structure with our proposed method and use static node matching to aggregated node weight values for the global model. In FedAvg, each client downloads the current model from a central server, improves it by learning from its own local data, and then aggregates the changes into a small centralized update. Only updates to the model are sent to the server/cloud using encrypted communication

and immediately averaged with other user updates to improve the shared model based on Eq. 6.13. All training data is kept locally and no individual updates are stored in the cloud.

Federated Average (FedAvg) is a generalization of FedSGD that allows local nodes to perform multiple batch updates to local data and swap updated weights instead of gradients. The basic principle behind this generalization is that in FedSGD, if all local nodes start from the same initialization, the mean gradient is strictly equivalent to the mean weight itself. Furthermore, averaging adjustment weights from the same initialization does not necessarily harm the performance of the resulting averaging model [67, 79].

FedDyn Next baseline is call FedDyn, in which each client learns a unique model with its own regularization parameter [1]. In this method, each client in the federated learning system learns a unique model with its own regularization parameter. The regularization parameter is updated dynamically during the training process based on the client's local model performance. This means that clients with more difficult data can have a higher regularization, while clients with easier data can have a lower regularization, which improves the convergence speed and accuracy of the federated learning process.

The objective of FedDyn is to solve Eq. 5.5, where $k \in [m]$ consists of N_k training instances, $L_k(\theta)$ is the empirical loss of the k_{th} device and θ are the parameters of the neural network.

$$arg \min_{\theta \in \mathbb{R}^d} [l(\theta)] \triangleq \frac{1}{m} \sum_{k \in [m]} L_k(\theta) \tag{5.5}$$

FedDNA$_{\text{fixed}}$ Baseline 4 (FedDNA$_{\text{fixed}}$) calculates nodes' distance based on a fixed node. This baseline is created because we want to confirm whether the node matching pattern in dynamic node alignment improve compared with when the node used for

matching remains the same. At the first step, after each client finishing training its local model, client $\mathbf{c}^{\alpha}$ is randomly chosen from $\mathcal{C}$. Then one node $\mathbf{v}_i^{\alpha}$ will be randomly selected among all the nodes in the first hidden layer of client $\mathbf{c}^{\alpha}$'s local model. In the next step, one client will be randomly $\mathbf{c}^{k}$ picked from $\{\mathcal{C}$ - $\mathbf{c}^{\alpha}\}$, a distance function explained previously will be used to calculate the distance $d(\mathbf{v}_j^k, \mathbf{v}_i^{\alpha})$ between all nodes in the first hidden layer of client $\mathbf{c}^{k}$ model and node $\mathbf{v}_i^{\alpha}$. We can get two nodes matched $(\mathbf{v}_j^k, \mathbf{v}_i^{\alpha})$ based on the smallest distance.

Now we have two nodes, which are also the start of our MST tree $\mathcal{T} = \{\mathbf{v}_j^k, \mathbf{v}_i^{\alpha}\}$, from which we will start to grow the tree. In each matching step, we will randomly pick one client from $\{\mathcal{C}$ - $\{\mathbf{c}^{\alpha}, \mathbf{c}^{k}\}\}$. Unlike `FedDNA`, in this baseline, the distance between a node to the tree will be calculated with Eq. 5.6, which means that only $\mathbf{v}_i^{\alpha}$ will be used in MST tree $\mathcal{T}$ to do the node matching. The MST tree $\mathcal{T}$ will continue growing until $\{\mathcal{C}$ - $\{\mathbf{c}^{\alpha}, \mathbf{c}^{k}$,...$\}\}$ is empty and at the same time, a complete tree $\mathcal{T}$ with new node set $\{\mathbf{v}_i^{\alpha}, \mathbf{v}_j^{k}, \mathbf{v}_t^{\beta}$,...$\}$ will be formed to aggregate their averaged weight values as a new node weight $\mathbf{v}_i^{s}$ for the global model.

$$d(\mathbf{v}, \mathcal{T}) = d_{\text{Euclidean/Manhattan}}(\mathbf{v}, \mathbf{v}_i^{\alpha}) \tag{5.6}$$

For example, in Fig. 5.5, distance for $\mathbf{c}^2$ will be $d(a, A)$, $d(a, B)$, $d(a, C)$, $d(a, D)$ and for $\mathbf{c}^3$ the distance will be $d(a, \alpha)$, $d(a, \beta)$, $d(a, \gamma)$, $d(a, \theta)$. Assume for $\mathbf{c}^2$, the smallest distance is $d(a, B)$ and $d(a, \alpha)$for $\mathbf{c}^3$, then node $\{\mathbf{v}_a^1, \mathbf{v}_B^2, \mathbf{v}_{\alpha}^3\}$ are the matching nodes and their weight values will be averaged as one new node weight values for the global model.

FedDNA$_{\text{random}}$ The last baseline is a modification based on both `FedDNA` and FedDNA$_{\text{fixed}}$. Instead of being too static or too dynamic with the node matching, we cant to confirm the feasibility when the matching node is neither 100% percent fixed nor using the entire MST tree as a matching node. Settings for baseline 4 (FedDNA$_{\text{random}}$) is as follows: At the first step, after each client finishing training its

local model, client $\mathbf{c}^\alpha$ is randomly chosen from $\mathcal{C}$. Then one node $\mathbf{v}_i^\alpha$ will be randomly selected among all the nodes in the first hidden layer of client $\mathbf{c}^\alpha$'s local model. In the next step, one client will be randomly $\mathbf{c}^k$ picked from $\{\mathcal{C}$ - $\mathbf{c}^\alpha\}$, a distance function explained previously will be used to calculate the distance $d(\mathbf{v}_j^k, \mathbf{v}_i^\alpha)$ between all nodes in the first hidden layer of client $\mathbf{c}^k$ model and node $\mathbf{v}_i^\alpha$. We can get two nodes matched $(\mathbf{v}_j^k, \mathbf{v}_i^\alpha)$ based on the smallest distance.

Now we have two nodes, which are also the start of our MST tree $\mathcal{T} = \{\mathbf{v}_j^k, \mathbf{v}_i^\alpha\}$, from which we will start to grow the tree. In our third step, one node will be randomly chosen from $\{\mathbf{v}_i^\alpha, \mathbf{v}_j^k\}$ which will be used to match nodes of client $\{\mathcal{C}$ - $\{\mathbf{c}^\alpha, \mathbf{c}^k\}\}$ using Eq. 5.4. Step 3 will be repeated until $\{\mathcal{C}$ - $\{\mathbf{c}^\alpha, \mathbf{c}^k$,...$\}\}$is empty and at the same time, a new node set $\{\mathbf{v}_i^\alpha, \mathbf{v}_j^k, \mathbf{v}_t^\beta$,...$\}$ will be formed to aggregate their averaged weight values as a new node weight value for the global model. Assume we randomly choose $\mathbf{c}^2$ in Fig. 5.5 to do the first match, node $\{a, B\}$ are the matching nodes, then one node will be randomly chosen from node $\{a, B\}$ to calculate distance for $\mathbf{c}^3$. If node B is chosen, distance $d(B, \alpha)$, $d(B, \beta)$, $d(B, \gamma)$, $d(B, \theta)$, will be calculated to choose the next matching node.

Our overall experiment setting is shown in Table 5.5. We use 10-fold cross validation, under which there will be 10 training rounds for each model to train. For each dataset, our aim is to predict the corresponding target and 10-fold cross validation is applied to reduce both bias and variance. Under each cross validation fold K, same weight values are initialized for all both baseline models and our proposed models, $Plain\_NN$, $FedAvg$, $FedDyn$, $Baseline_3$, $Baseline_4$ and $FedDNA$. For methods under FL setting, model parameters will be passed to each clients at the very beginning of training. Training data will be randomly split into 5 sites and distributed to 5 clients, which is able to training the local model using their own data, after which weight values will be aggregated based on different FL method and then send back to the global models. Global models will pass the new calculated parame-

ters to their local clients to start new round training until the convergence. For our proposed method FedDNA, there are two experiment settings in this paper. One is called no-freezing weight update setting, in which weight values of the global model will be aggregated using FedDNA method for all the 10 rounds while the second type of setting is called freezing, in other word, we will choose to update the global model parameters with FedDNA at the first i round and after that FedAvg will be used to aggregate clients' model weight values for the rest of rounds. We design this type of setting because we think the first several rounds of distance calculation will give use the answer of the closest matching nodes then we can use that match to directly aggregated the node weight values.

For our experiment dataset settings, we first run our model based on the original class distributions across all clients in all datasets which is $negative : positive = 1.54$ in spam database, $negative : positive = 0.89$ in diabetes data base, $negative : positive = 11.26$ in Patient Survival Prediction data set and $negative : positive = 3.33$ in Occupancy Detection data set . In the second experiment setting, for each training process, 2 clients are randomly chosen to exchange 2/3 of their data while the rest 3 clients keep their own data, in this case, our model will be verified on non-IID datasets. Calculated overall node distance, Accuracy, AUC, F1_score, Balanced accuracy and Loss are used as performance measurement metrics.

Apart from randomly selecting 2 clients to exchange their data, we decide to evaluate our proposed model under different class distribution settings. The original class distribution (negative:positive) of the four datasets are as shown in Table 5.4. A set of class distributions is set up for the original four data sets to check the model performance. Since all the datasets have different original class distributions, the assigned class distributions of the four datasets in this paper are different from each other.

Table 5.5: The pseudo code of the experiment settings and comparisons (all methods are compared based based on same training/test data. The initial network weights of each site are the same for different methods to avoid discrepancy due to random weight initialization.

Experiment setting: Node_matching Federated Learning

Input (1) Datasets $\mathcal{D}$; (2) Node matching setting: Freeze; No freeze

Output Prediction of the target y

For each cross validation fold K:

  Initialize same weight values for all the global models:

    $Plain\_NN$, $FedAvg$, $FedDyn$, $(FedDNA_{fixed})$, $(FedDNA_{random})$ and $FedDNA$

  Split training data into 5 sites for 5 clients

  For $Baseline_1$:

    Train model using all the training data

  For federated learning models:

    Client train their own model using their own data

    Match nodes with their distance calculation principle

    Aggregated weight values pass to each global model

End For

Evaluate each global models

5.2.2 Experimental Results

Table 5.6 to Table 5.9 show the results for Diabetes dataset, Spam dataset, Occupancy data set and Patient survival data set respectively in our first dataset setting. Due to page limitation, only the best model performance results are presented in this paper. For Diabetes database, we can observe that FedDNA, which uses our proposed method FedDNA is able to find nodes combinations where the total node distance is

the smallest with value 42.1352 compared with other methods whose final distance results are greater than 50. At the same time, FedDNA presents better metrics performance. Similarly, the smallest overall node distance and better metric performance are realized by our FedDNA method for spam database. However, we can also come to the conclusion that smaller overall node distance and better metrics performance does not always come with the smallest training loss, especially for FedDNA. For occupancy and patient survival prediction datasets, FedDNA shows similar performance as for the previous two datasets. Its overall classification performance outperforms all the baselines with the smallest node distance 5.7316 and 57.4096 respectively after matching, which indicates FedDNA is able to pair closest nodes together. We can tell that overall, for all the four datasets, when class distributions are the same across all clients, our proposed method performs the best in the freezing setting when the first two rounds using Manhattan distance to find the matching nodes and the rest using FedAvg with the smallest overall distance under freeze first two rounds experiment setting with 42.1352 for diabetes dataset, 730.3930 for the spam data set, 5.7316 for Occupancy Detection and 57.4096 for Patient Survival Prediction data set.

Table 5.6: Experimental results from Diabetes dataset using Manhattan distance based matching. For FedDNA, the matching freezes after first two rounds of dynamic node alignment.

	Distance	Accuracy	AUC	F1_score	Balanced accuracy	Loss
$Plain\_NN$		0.6755	0.7529	0.6361	0.6744	3.3221
$FedAvg$	51.7421	0.7165	0.8044	0.7029	0.7219	1.9016
$FedDyn$	53.0637	0.7016	0.8012	0.7147	0.7078	1.8932
$FedDNA_{fixed}$	63.7216	0.7264	0.7961	0.7120	0.7341	1.8354
$FedDNA_{random}$	56.3497	0.7298	0.8003	0.7280	0.7396	1.8274
$FedDNA$	**42.1352**	**0.7381**	**0.8230**	**0.7290**	**0.7434**	**1.9253**

Table 5.7: Experimental results from Spam dataset using Manhattan distance based matching. For FedDNA, the matching freezes after first two rounds of dynamic node alignment.

	Distance	Accuracy	AUC	F1_score	Balanced accuracy	Loss
Plain_NN		0.9165	0.9536	0.8905	0.9095	0.5110
FedAvg	824.3761	0.9320	0.9717	0.9120	0.9294	0.4823
FedDyn	792.1406	0.9316	0.9719	0.9136	0.9117	0.4431
$FedDNA_{fixed}$	798.0362	0.9350	0.9762	0.9187	0.9343	0.6352
$FedDNA_{random}$	911.6532	0.9351	0.9772	0.9178	0.9337	0.4521
FedDNA	**730.3930**	**0.9376**	**0.9781**	**0.9210**	**0.9357**	**0.4841**

Table 5.8: Experimental results from Occupancy detection dataset using Manhattan distance based matching. For FedDNA, the matching freezes after first two rounds of dynamic node alignment.

	Distance	Accuracy	AUC	F1_score	Balanced accuracy	Loss
Plain_NN		0.8327	0.9578	0.5018	0.7021	1.4599
FedAvg	7.8736	0.9225	0.9715	0.8236	0.8910	1.1592
FedDyn	8.5130	0.9135	0.9713	0.8137	0.8862	1.0927
$FedDNA_{fixed}$	10.4510	0.9346	0.9751	0.8681	0.9179	0.9031
$FedDNA_{random}$	9.6437	0.9306	0.9762	0.8699	0.9083	0.8621
FedDNA	**5.7316**	**0.9402**	**0.9788**	**0.8710**	**0.9140**	**0.8711**

5.2.2.1 FedDNA vs. FedAvg with respect to Different Class Distributions

Fig. 5.7 is a box-plot for our second experiment setting's results. Instead of showing all the results of all models across all the datasets, since results from the first set-

Table 5.9: Experimental results from Patient survival prediction dataset using Manhattan distance based matching. For FedDNA, the matching freezes after first two rounds of dynamic node alignment.

	Distance	Accuracy	AUC	F1_score	Balanced accuracy	Loss
$Plain\_NN$		0.8201	0.5309	0.0618	0.4998	1.7521
$FedAvg$	63.2566	0.7913	0.6185	0.0676	0.4942	1.1150
$FedDyn$	67.9825	0.7740	0.6098	0.0635	0.5099	1.0047
$FedDNA_{fixed}$	71.2609	0.9083	0.6422	0.0302	0.5024	1.0670
$FedDNA_{random}$	75.6094	0.8633	0.6251	0.0232	0.5075	1.0427
$FedDNA$	**57.4096**	**0.8898**	**0.6485**	**0.0297**	**0.5079**	**1.4629**

ting highlights that FedDNA outperforms FedAvg and FedDyn overall across all the datasets, only comparisons between FedDNA and FedAvg, FedDyn with the combined results across all the datasets are shown in Fig. 5.7, in which outliers can be observed for three models but overall we can come to the conclusion that when data is not evenly distributed across all clients, FedDNA performs the best in the freezing setting when the first two rounds using Manhattan distance to find the matching nodes and the rest using FedAvg.

Since under this experiment setting, FedDyn does not deliver better overall performance than FedAvg according to above tables and figure, its detailed comparison with FedDNA is not demonstrated. Fig. 5.8 to Fig. 5.11 report the performance of FedDNA and FedAvg, with respt to different class distributions (the class distributions were adjusted to assess the algorithm performance under different conditions). The y-axis is the values of each measurement and x-axis is different class distribution set ups for each dataset as shown in Table 5.4.

For Diabetes Dataset, FedDNA and FedAvg has the largest gap for all the measurements when sampling rate is 0.5 and both models' performance fluctuate a lot

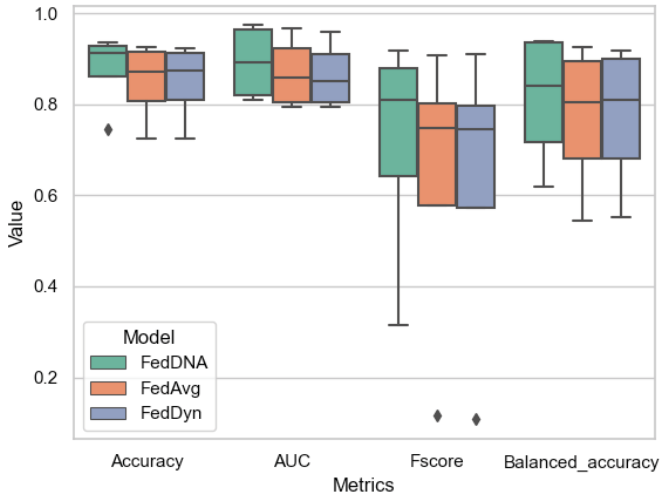


Figure 5.7: Overall performance comparisons between FedDNA and FedAvg, FedDyn.Outliers ca be observed from both methods, overall FedDNA outperforms FedAvg and FedAvg performs similarly as FedDyn.

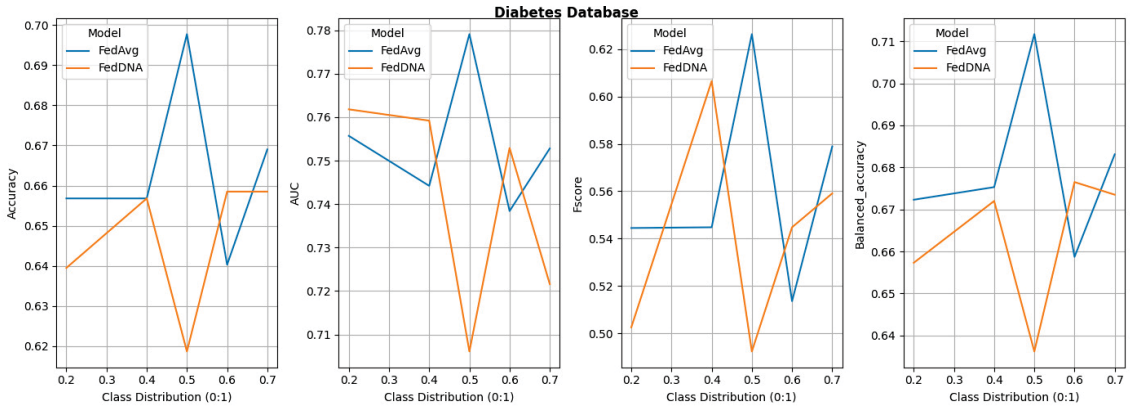


Figure 5.8: Performance comparisons between FedDNA and FedAvg with respect to different class distributions for diabetes dataset.

with the change of class distributions. For Spam Dataset, even though FedDNA and FedAvg perform similarly when class distribution is smaller, as more and more negative samples shown in the datasets, FedDNA starts to show more advantages than FedAvg, especially when negative instances take up more than 40% of the dataset, the gap between both models become larger with a better performance from FedDNA.

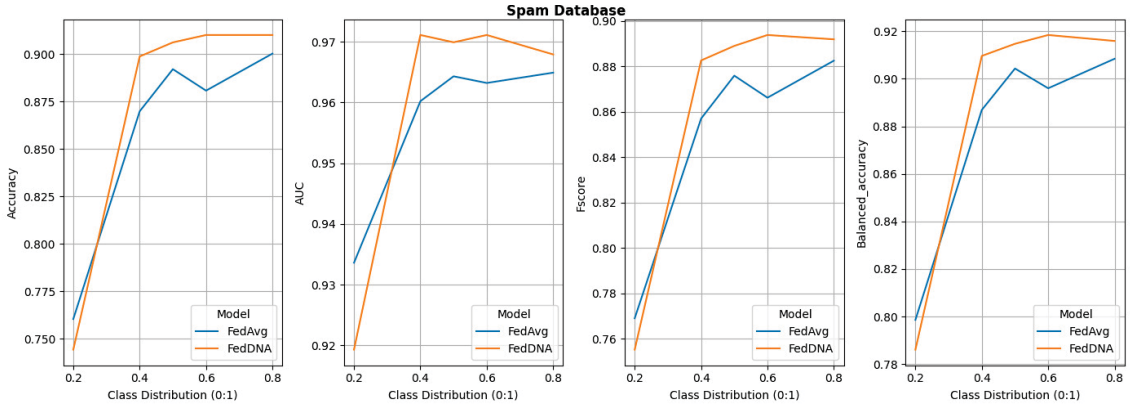


Figure 5.9: Performance comparisons between FedDNA and FedAvg with respect to different class distributions for spam dataset.

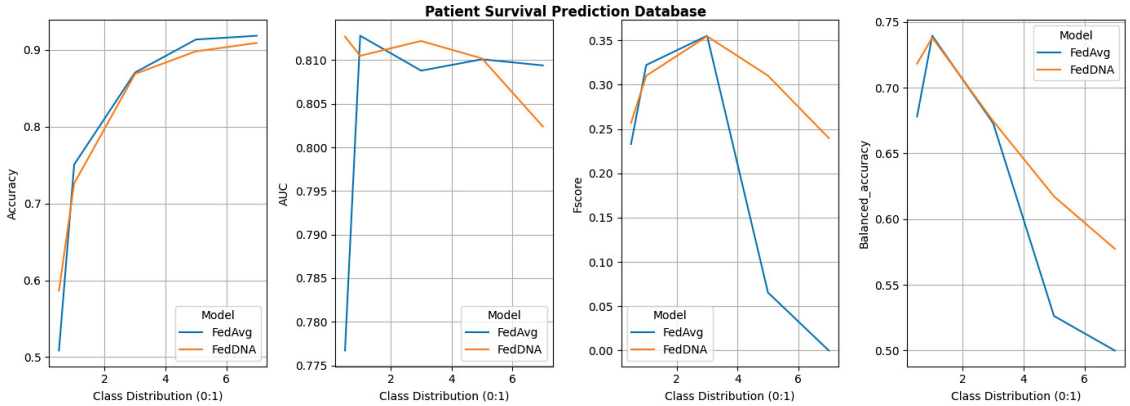


Figure 5.10: Performance comparisons between FedDNA and FedAvg with respect to different class distributions for patient survival prediction dataset.

Similarly, for Patient Survival Prediction Dataset, the larger the sampling rate is, the better FedDNA outperforms FedAvg especially in terms of Fscore and Balanced accuracy. While for Occupancy dataset, FedDNA does not show much better results than FedAvg when class distribution is less than 2, after which both models the performance of the two models tends to be consistent.

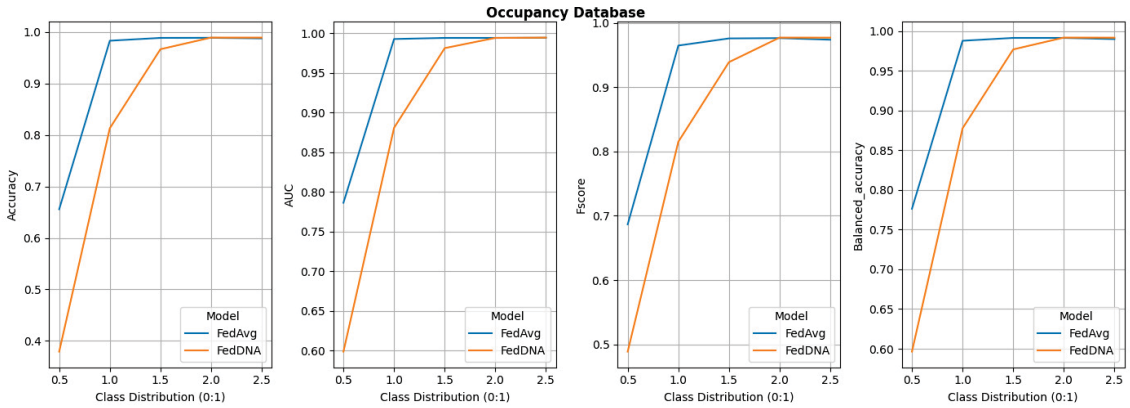


Figure 5.11: Performance comparisons between FedDNA and FedAvg with respect to different class distributions for occupancy dataset.

5.3 CONCLUSIONS

In this research, we propose a dynamic node matching method for federated learning. We argued that neural networks are inherently non-transparent and unstable, and the same network structure may end up with very different weight values, even with the same training data and same parameter settings. Traditionally, existing methods, such as FedAvg, force neurons across sites to be matched with predefined order, and use fixed matching nodes during the FL learning process. Alternatively, we proposed a dynamic node alignment, FedDNA, approach which dynamically finds matching nodes across sites, and uses matched nodes to calculate weight for FL learning. FedDNA represents each neuron as a vector, using their weight values, and calculate distances between neurons to find matching nodes. Meanwhile, because finding marching nodes are computationally expensive, we proposed a minimum spanning tree (MST) based approach to speed up the matching, with matched nodes across all sites being linked by using an MST tree. So the matching process is simply the MST tree growing process. Experiments and comparisons, including biased sample distributions, validate the performance of the FedDNA, compared to other baseline.

Future study can emphasize on the following three directions. First, we only

studied dense networks and verified its performance using FedDNA. Future study can try to explore node matching between different types of network architectures, such as convectional nueral networks. Second, we only studied the proposed design using binary classification problems. In the future, multi-class classification problem will be explored using our proposed FedDNA method. For the last direction, we will use non-IID datasets to further adjust our model so that it can be applied not only to binary classification problem but also can achieve good results for datasets with different settings.

CHAPTER 6

ACTIVE LEARNING USING LOCALITY-CUSTOMIZED GSA

Federated Learning (FL), represents a novel learning approach that constructs machine learning models using decentralized datasets distributed across numerous sites/devices. The primary purpose of this paradigm is to safeguard privacy and avert data leakage risks [67]. The feasibility of Federated Learning (FL) as a decentralized machine learning approach heavily relies on the proficiency of local models in both training and inference tasks. These local models' effectiveness is contingent upon the availability of meaningful and annotated data, which is essential for their successful training [7, 80, 99]. However, obtaining such data involves a laborious and time-consuming annotation process, necessitating manual analysis of the training samples. In the realm of machine learning, data annotation plays a pivotal role in empowering models with the capacity to generalize effectively and achieve high-performance levels. However, The data annotation process presents two significant challenges that researchers and practitioners must confront. First, it demands meticulous and time-consuming analysis for each sample, rendering it a laborious endeavor. Second, and perhaps more critically, the selection of appropriate samples is not always guaranteed, resulting in potential negative impacts on the overall performance of the model [7, 82, 85, 88].

Recently, Active Learning (AL) has emerged as a machine learning method that can effectively address data annotation workloads [78, 84]. Its main strategy is to iteratively find the most informative data points to annotate. The annotated data are then used as part of the training data in the next iteration. With more and more iterations, the machine learning model's performance can be more and more improved. This strategy has been integrated into federated learning and generated a

new paradigm called Federated Active Learning (FAL) [8, 55, 72, 100].

The FAL framework consists of several clients and one central server. Each client holds one labeled dataset and one unlabelled dataset, which can not be shared with others. The server holds a test dataset that can be shared with all clients. The goal of FAL is to train a globally optimized model at the server by annotating informative data samples at the clients. FAL framework is trained in an iterative manner. In one iteration, each client first trains a local model with annotated data. And then, the local parameters are transmitted to the server. The server synthesizes local parameters into a global model. After that, the global model is sent to each client to annotate several unlabeled data with the highest informativeness. The labeled dataset is extended by merging the previously labeled dataset and the newly annotated dataset. In the next iteration, new local models are trained with the new labeled dataset. With more and more iterations, more and more data will be annotated and the global model can be improved.

However, current FAL has two significant weaknesses: (i) In most FAL, local unlabelled samples are annotated by the aggregated global model's parameters, which totally ignores the localization of the samples, furthermore, the importance of local models for local sample annotation is completely ignored [50]. (ii) Its global model parameter updating is limited to one method, which is called Federated average (FedAvg) [7,8,67,69]. FedAvg relies on a strong assumption that the corresponding nodes in local neural networks share the same importance when averaging, while different local models should have different average weight [99].

To tackle the first weakness, we propose a locality-customized annotation strategy, which takes the local model into consideration aside from the global model when annotating. There are two reasons to pay attention to the local model: (i) local models compose the global model; (ii) the annotated data are directly used to train local models. Specifically, we first predict all unlabeled data's labels by the local

model and the global model separately. Then, we calculate the uncertainty of the prediction by the metric of entropy. Each data's overall uncertainty is a combination of both the local model's prediction entropy and the global model's prediction entropy. Finally, we annotate the top K data with the highest informativeness.

To tackle the second weakness, we propose a Gravitational Search Algorithm (GSA) based FAL framework. Different from FedAvg, global model parameter aggregations are achieved by GSA which draws inspiration from the law of gravity and the interactions between celestial bodies. GSA allows population diversity as well as global exploration, which means FL clients can interact with each other based on their masses (accuracy) and positions (local model parameters), at the same time, GSA is capable of exploring the solution space globally by allowing clients to move freely towards areas of high fitness calculated based on their masses (accuracy). Moreover, it is empowered with higher adaptability with a set of parameters that control the interaction between clients. Essentially, the GSA method can be viewed as a weighted averaging strategy where the mass plays the role of weight.

To summarize, in this research, we propose a locality-customized GSA federated active learning (LG-FAL) method. The main contributions of the proposed research are: (i) We propose a new annotating strategy that considers both local and global optimization. By doing so, the localization of samples and models can be considered; (ii) We propose to update the global model parameters with GSA, in which the model is updated in a more interactive and adaptable way; (iii) We design extensive experiments to validate the proposed methods with different parameter settings and comparisons.

6.1 THE PROPOSED METHOD

We propose to combine Active Learning (AL) and Gravitational Search Algorithm (GSA) in a federated and collaborative framework to select a small enough subset

of informative local data and provide an improved trade-off between communication costs and learning accuracy.

6.1.1 Framework

Our proposed locality-customized GSA Federated Active Learning (LG-FAL) framework includes two phases: local phase and global phase. Inspired by the original version of FL algorithm to train the global model in a central server, locality-customized annotation AL is executed in clients and sends the local parameters to the central server. Then globally, GSA continues the process in the central server to aggregate the local model parameters and update the global model. Once the GSA is finished, new model parameters will be passed down to clients for the next iteration. We summarize the description of the framework in Fig. 6.1.

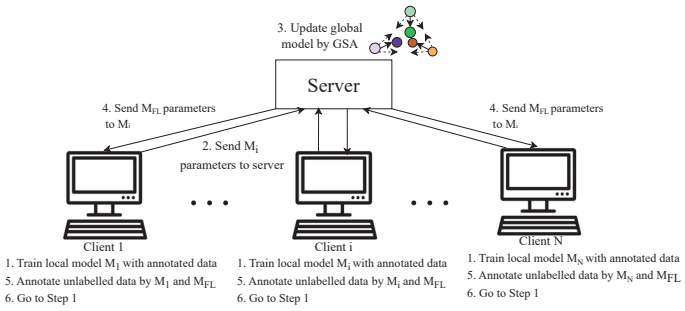


Figure 6.1: Framework of LG-FAL. Clients train local models and send them to the server. The server synthesizes the models and gets a global model. The global model is sent to each client to help annotate local data.

6.1.2 Locality-customized Annotating

Inspired by the classic AL methods, we introduce a locality-customized annotation function $\mathcal{AL}$, which is able to find the most informative samples to be labelled for each local dataset with the combined informativeness from both local model and global model.

Local Model For each local client i, model trained using its own local data is defined as $\mathcal{M}_i$. Local model $\mathcal{M}_i$ enables customization and adaptation to specific local device characteristics and local data patterns to better fit unique local data distributions, which is able to capture different facets of the local data distribution and localization.

Global Model Global model $\mathcal{M}_{FL}$ is the central model that is shared and iteratively updated across a network of decentralized $\mathcal{M}_i$ in FL. During the training process, locally trained models $\mathcal{M}_i$ send back their parameter updates to a central server, which aggregates these updates to refine the global model $\mathcal{M}_{FL}$. By combining data informativeness from both $\mathcal{M}_i$ and $\mathcal{M}_{FL}$, we are able to capture the data generalization while maintain its localization at the same time.

Locality-customized Active Learning In this work, for each local client i, both local labelled dataset $\mathcal{D}_i$ as well as newly-annotated dataset A_i by $\mathcal{AL}$ will be set up for the training. We design a score function $S(x)$ to evaluate unlabeled samples. The strategy is to annotate data samples with the highest score in the unlabelled data as shown in Eq. 6.1, where z is the sampling number and $S(x)$ is the score function of x.

$$\mathcal{AL}_i = \underset{|A_i|=z, x \in \mathcal{U}_i}{\operatorname{argmax}} \; S(x) \tag{6.1}$$

To make sure that the score function is able to reflect the localization and potential informativeness of local unlabelled instances, we introduce the score function as shown in Eq. 6.2. Global model $\mathcal{M}_{FL}$ and local model $\mathcal{M}_i$ are allowed to predict on the labelling possibilities of samples. The most informative query is considered to be the instances about which they most agree. The sample informativeness from both global model $\mathcal{M}_{FL}$ and local model $\mathcal{M}_i$ are integrated to find the average score of an sample.

$$S(x) = w_1 * \text{Entr}(Dis(x|\mathcal{M}_i)) + w_2 * \text{Entr}(Dis(x|\mathcal{M}_{FL})) \tag{6.2}$$

where $Dis(x|M)$ denotes the prediction distribution of x under model M; $Entr(Dis(x|M))$ denotes the entropy of the distribution. w_1 and w_2 are weights between 0 and 1.

The higher the entropy is, the more uncertain the sample under the model will be. Thus, the active learning strategy prefers to annotate samples with high uncertainty. AL adopts multiple rounds as the FL goes on for sampling and gradually adds samples to the labeled local dataset.

6.1.3 GSA Federated Learning

We integrate Gravitational Search Algorithm (GSA) with federated learning to obtain a globally optimized model from local models. Within GSA's iterative framework, each local model is viewed as one object with mass, while its parameters are viewed as position coordinates. These objects attract each other due to gravity, prompting their movement towards heavier masses, which correspond to favorable solutions [76]. Fig.6.2 shows the movement of the object.

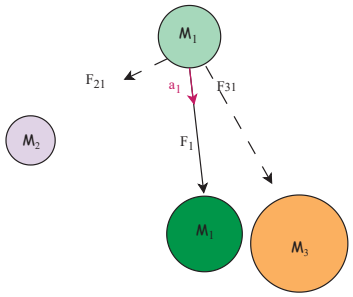


Figure 6.2: Demonstration of object movement with GSA. Object M_1 is attracted by M_2 and M_3, with gravity force F_{21} and F_{31}. The total force F_1 results in acceleration a_1 to update the position of M_1, which equals to update the parameter vector X_1.

Assume there are N clients participating FL, each of whose local model has a D dimensional parameter vector denoted as Eq. 6.3, where x_i^d is the parameter of the ith agent in dimension d.

$$X_i = (x_i^1, ..., x_i^d, ..., x_i^D) \qquad i = 1, ..., N \tag{6.3}$$

First, the gravitational mass of each object using the fitness values is calculated as in Eq. 6.4 and Eq. 6.5. The gravitational mass is denoted as $M_i(t)$ and $fit_i(t)$ indicates the fitness value of the ith object at iteration t, which, in our method, client's predictive accuracy on the test dataset is used as the fitness value.

$$m_i(t) = \frac{\text{fit}_i(t) - \text{worst}(t)}{\text{best}(t) - \text{worst}(t)} \tag{6.4}$$

$$M_i(t) = \frac{m_i(t)}{\sum_{j=1}^{N} m_j(t)} \tag{6.5}$$

In addition, $worst(t)$ and $best(t)$ are the worst and best fitness values obtained in the collection of objects at t which are defined for maximization problem as in Eq. 6.6 and Eq.6.7 respectively.

$$\text{worst}(t) = \min fit_j(t) \qquad j \in \{1, ..., N\} \tag{6.6}$$

$$\text{best}(t) = \max fit_j(t) \qquad j \in \{1, ..., N\} \tag{6.7}$$

The total force that is applied on the ith object from other objects is computed following the gravity law in Eq. 6.8

$$F_i^d(t) = \sum_{j \in Kbest, j \neq i} \text{rand}_j^d G(t) \frac{M_j(t) \times M_i(t)}{R_{ij} \times \epsilon} (x_j^d(t) - x_i^d(t)) \tag{6.8}$$

in which rand_j^d is a random number with uniform distribution in the interval [0, 1]. ϵ denotes a small number close to 0, $R_{ij}(t)$ denotes the Euclidean distance between clients i and j, and $Kbest$ is a set consisting of the first K objects with the best fitness values (the largest masses).

K is set as N at the very beginning and reduces linearly with time until it reaches to 1 in the end. The gravitational constant at iteration t is denoted as $G(t)$ which is initialized at the first iteration by G_0 and decreased by time according to Eq.6.9, where T is the total number of iterations.

$$G = G_0 \exp^{-\alpha \frac{t}{T}} \tag{6.9}$$

Then, the obtained force is used to calculate the acceleration of the object using the law of motion as in Eq.6.10.

$$\begin{aligned} a_i^d(t) &= \frac{F_i^d(t)}{M_i(t)} \\ &= \sum_{j \in Kbest, j \neq i} \text{rand}_j G(t) \frac{M_j(t)}{R_{ij}(t) \times \epsilon} (x_j^d(t) - x_i^d(t)) \end{aligned} \tag{6.10}$$

After the above steps, the next movement for ith object can be computed based on the change of its acceleration as in Eq. 6.12 and this is the end of one GSA iteration.

$$v_i^d(t+1) = v_i^d(t) + a_i^d(t), \qquad v_i^d(0) = 0 \tag{6.11}$$

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{6.12}$$

After a certain number of iterations, all parameter vectors X_i are updated with other vectors' information. In other words, all X_i can be viewed as candidates of the aggregated global model parameters. We test them on the test dataset, and consider the parameter with the best performance as the parameter of the global model $\mathcal{M}_{FL}$.

Training of a global GSA algorithm is performed in an iterative fashion. It communicates with local ones iteratively since the stopping criterion is reached. Each client initially starts with a randomized model that is the exact same structure as the central model. The pseudo-code of the LG-FAL is shown in Algorithm 4.

6.2 EXPERIMENTS

6.2.1 Experimental Settings

6.2.1.1 Datasets

We use two benchmark datasets in the experiments. The first one is MNIST Dataset (Modified National Institute of Standards and Technology database) [33], which consists of a collection of handwritten digits. It contains 60,000 training images and 10,000 testing images. Each image is a grayscale image of size 28x28 pixels, representing a single digit (0-9). The second dataset is called Fashion MNIST [101]. It is a

Algorithm 4 Locality-customized GSA Federated Active Learning (LG-FAL)

Input: Number of clients N, number of FAL iteration T, test dataset $\mathcal{T}$, initially labelled dataset $\{\mathcal{D}_i(0)\}_{i=1}^{N}$, number of annotated data in each iteration z, initially unlabelled dataset $\{\mathcal{U}_i(0)\}_{i=1}^{N}$

Output: Optimized global model $\mathcal{M}_{FL}$

for $t = 1$ to T **do**
  for each client i, $i = 1, ..., N$ **do**
    Train local model $\mathcal{M}_i$ with annotated data
    Send $\mathcal{M}_i$ parameters to the server
  end for
  At the server:
  Calculate gravitational mass $M_i(t)$ for each client by Eq.6.5
  Calculate total force $F_i^d(t)$ on each client by Eq. 6.8
  Calculate the acceleration $a_i^d(t)$ of each client by Eq. 6.10
  Update local model parameters by Eq. 6.12
  Evaluate all updated local models with test dataset $\mathcal{T}$, define
  $\mathcal{M}_{FL}$ as the best local model
  Send $\mathcal{M}_{FL}$ to each client
  for each client i, $i = 1, ..., N$ **do**
    Annotate z unlabelled data $\mathcal{A}_i$ from $\mathcal{U}_i(t)$
    Update $\mathcal{D}_i(t+1) = \mathcal{D}_i(t) + \mathcal{A}_i$
    Update $\mathcal{U}_i(t+1) = \mathcal{U}_i(t) - \mathcal{A}_i$
  end for
end for

variation of the original MNIST dataset, but instead of containing handwritten digits, it consists of images of various types of clothing and fashion items. This dataset has 10 different categories, which include items like T-shirts, trousers, pullovers, dresses, coats, sandals, shirts, sneakers, bags, and ankle boots with 60,000 training images

and 10,000 testing images and each image being a grayscale 28x28 pixel.

6.2.1.2 Baselines

To validate the performance of the proposed method, we use deep neural networks as the training models and employ two baselines for our comparisons.

The first baseline is called Federated Average (FedAvg) Active Learning, FedAvg-FAL, which also shares the same network structure with our proposed method. In FedAvg, each client downloads the current model from a central server, improves it by learning from its own local data, and then aggregates the changes into a small centralized update. Eq. 6.13 summarizes the global weight values $\mathbf{w}$ updating of FedAvg in each training round t, in which i is the client index, N means the total number of clients, $\mathcal{D}$ is the total number of instances and $\mathcal{D}_i$ is the local data examples for each client [67].

$$\mathbf{w}_{t+1} = \sum_{i=1}^{N} \frac{\mathcal{D}_i}{i} \mathbf{w}_t^i \tag{6.13}$$

The second baseline S-FAL annotates samples in a single AL way while keep the same GSA FL model parameter update approach. In this setting, the local instance informativeness score is only computed based on the updated local model $\mathcal{M}_{FL}$ with local dataset D_i as shown in Eq. 6.14.

$$S(x) = Entr(Dis(x|\mathcal{M}_{FL})) \tag{6.14}$$

6.2.1.3 Experiment Settings

Our overall experiment setting is as follows. For each dataset, our aim is to predict the corresponding target. Model parameters will be passed to each clients at the very beginning of training. Training data will be randomly split into 3 sites and distributed to 3 clients, which is able to training the local model using their own data. As for AL part, by default, $w_1 = w_2 = 0.5$. For each round, 32 unlabelled samples will be annotated by different AL approaches and added into local dataset for training.

For FL part, weight values will be aggregated based on different FL methods and then send back to the global models. Global models will pass the new calculated parameters to their local clients to start new round training until the convergence.

In order to explore the effect of various GSA parameter settings, we decided to evaluate our proposed method with different α and G_0 combinations. Additionally, we conduct experiments to explore the impact of w_1 and w_2 on out proposed method.

6.2.2 Experimental Results

Table 6.1 and Table 6.2 show the model results for MNIST dataset and Fashion MNIST dataset respectively. Due to page limitation, only the best model performance results are presented in this paper. For MNIST dataset, overall, GSA based FAL methods do a better classification job than FedAvg based FAL approach. For all the methods, an obvious increase in their predictive accuracy can be observed with gradually increased samples annotated by AL regardless the GSA parameter settings, however, we have to admit that there are small accuracy fluctuations for all methods especially with annotated samples larger than 192. When $\alpha = 30$ and $G_0 = 20$, LG-FAL still outperforms FedAvg-AL with a 0.835 accuracy when 320 samples are annotated, while S-FAL performs the best with a 0.849 accuracy. The superiority of our proposed method is more significant under $\alpha = 30$ and $G_0 = 10$ setting and $\alpha = 30$ and $G_0 = 50$ setting, with 0.842 and 0.855 accuracy respectively. LG-FAL deliver the highest final predictive accuracy 0.855 when $\alpha = 30$, $G_0 = 50$.

For Fashion MNIST dataset, similar performance as in MNIST dataset can be observed for all three methods. S-FAL and LG-FAL demonstrate better model performance with higher predictive accuracy than FedAvg-AL under all three parameter settings. The advantage of LG-FAL over S-FAL does not show when $\alpha = 30$, $G_0 = 10$ since both methods achieve 0.73 around accuracy. However, as G_0 goes up and the more annotated samples, the gap between S-FAL and LG-FAL gets more obvious

Table 6.1: Model accuracy *w.r.t* different parameter settings for MNIST with gradually increasing annotated samples: FedAvg-AL is the first baseline, S-FAL is the second baseline, LG-FAL is our proposed method.

GSA Parameter setting	Model	32	64	96	128	160	192	224	256	288	320
	FedAvg-AL	0.795	0.786	0.819	0.809	0.813	0.816	0.809	0.813	0.809	0.806
$\alpha = 30, G_0 = 10$	S-FAL	0.757	0.793	0.808	0.803	0.814	0.822	0.824	0.82	0.833	0.823
	LG-FAL	0.817	0.82	0.823	0.831	0.842	0.846	0.843	0.844	0.842	0.842
	FedAvg-AL	0.765	0.812	0.82	0.828	0.829	0.801	0.81	0.821	0.823	0.832
$\alpha = 30, G_0 = 20$	S-FAL	0.783	0.811	0.829	0.814	0.834	0.829	0.836	0.837	0.847	0.849
	LG-FAL	0.79	0.8	0.793	0.811	0.821	0.831	0.821	0.829	0.825	0.835
	FedAvg-AL	0.756	0.798	0.812	0.8	0.799	0.803	0.808	0.813	0.813	0.814
$\alpha = 30, G_0 = 50$	S-FAL	0.769	0.815	0.823	0.82	0.838	0.834	0.837	0.83	0.822	0.84
	LG-FAL	0.816	0.848	0.844	0.854	0.853	0.857	0.861	0.852	0.857	0.855

Table 6.2: Model accuracy *w.r.t* different parameter settings for Fashion MNIST with gradually increasing annotated samples: FedAvg-AL is the first baseline, S-FAL is the second baseline, LG-FAL is our proposed method.

Parameter setting	Model	32	64	96	128	160	192	224	256	288	320
	FedAvg-AL	0.669	0.677	0.677	0.692	0.694	0.697	0.675	0.708	0.699	0.7
$\alpha = 30, G_0 = 10$	S-FAL	0.692	0.708	0.722	0.715	0.725	0.722	0.714	0.712	0.727	0.733
	LG-FAL	0.668	0.7	0.701	0.726	0.713	0.729	0.7	0.718	0.722	0.731
	FedAvg-AL	0.602	0.604	0.615	0.598	0.612	0.609	0.602	0.601	0.603	0.612
$\alpha = 30, G_0 = 20$	S-FAL	0.698	0.698	0.698	0.692	0.701	0.68	0.703	0.7	0.703	0.717
	LG-FAL	0.718	0.713	0.742	0.717	0.729	0.747	0.749	0.75	0.752	0.757
	FedAvg-AL	0.453	0.553	0.601	0.6	0.62	0.67	0.684	0.695	0.699	0.701
$\alpha = 30, G_0 = 50$	S-FAL	0.711	0.715	0.717	0.73	0.734	0.731	0.723	0.724	0.725	0.726
	LG-FAL	0.732	0.748	0.755	0.756	0.755	0.736	0.74	0.743	0.762	0.762

with LG-FAL showing a higher accuracy.

Since LG-FAL and S-FAL outperforms FedAvg-AL, we create a figure to further demonstrate model comparison between LG-FAL and S-FAL with the results from both MNIST dataset and Fashion MNIST dataset. Regardless the parameter setting, the average accuracy of two models for both datasets are calculated. Fig. 6.3 report

the performance of LG-FAL and S-FAL as the increasing of annotated samples. The y-axis is the values of model accuracy and x-axis shows the increase of annotated instances. As the number of labeled samples gradually increases, the overall performance of the two models also shows an upward trend. Overall, LG-FAL always outperforms S-FAL, increasing from 0.755 to 0.795, which is much higher than the accuracy of S-FAL.

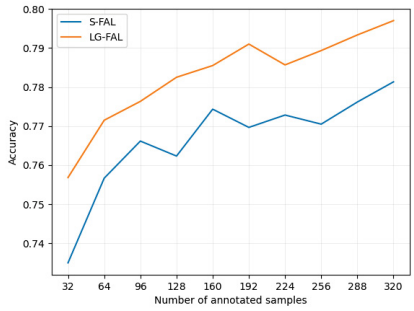
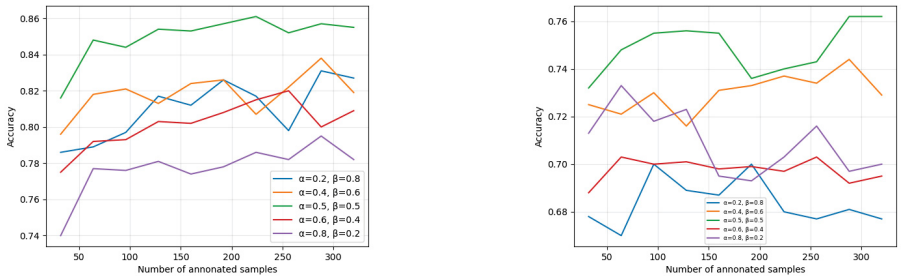


Figure 6.3: Overall performance comparison between LG-FAL and S-FAL with the increase of annotated samples: y-axis is the model averaged accuracy from MNIST dataset and Fashion MNIST dataset; x-axis is the gradually increased number of annotated samples.

The advantages of LG-FAL are able to be verified with the previously shown results. With the confirmation that LG-FAL is able to outperform our baselines, especially when $\alpha = 30$, and $G_0 = 50$, we further conduct a series of experiments aiming to figure out how the change of proportion of $\mathcal{M}_i$ and $\mathcal{M}_{FL}$ in Eq. 6.2 effects the performance of LG-FAL. By default, the values of w_1 and w_2 are set as 0.5 and 0.5 respectively. Different combinations of w_1 and w_2 are designed in order to check how our proposed method will react as follows: $w_1 = 0.2$, $w_2 = 0.8$; $w_1 = 0.4$, $w_2 = 0.6$; $w_1 = 0.5$, $w_2 = 0.5$; $w_1 = 0.6$, $w_2 = 0.4$; $w_1 = 0.8$, $w_2 = 0.2$.

Fig. 6.4 reports the overall predict accuracy trend of LG-FAL on our benchmark datasets under different w_1 and w_2 settings. For MNIST dataset, the prediction



(a) LG-FAL performance on MNIST dataset

(b) LG-FAL performance on Fashion MNIST dataset

Figure 6.4: Performance of LG-FAL with different w_1 and w_2 settings on MNIST dataset and Fashion MNIST dataset with the increase of annotated samples: y-axis is LG-FAL accuracy; x-axis is the gradually increased number of annotated samples; legends are w_1 and w_2 settings.

accuracy difference of LG-FAL with different w_1 and w_2 combinations is relatively clear. We can observe that when prefer the predictive uncertainty from local model over global model, the performance of LG-FAL drops especially when $w_1 = 0.8$ and $w_2 = 0.2$. However, the gradual increase of model performance with the increasing annotated samples can still be validated. LG-FAl demonstrates the best predictive accuracy on MNIST dataset when local model and global model are equally considered for annotating the instances. As for Fashion MNIST dataset, similarly, LG-FAL performs the best when the entropy from local model and global model contribute equally for the calculation of sample informativeness.

6.3 CONCLUSIONS

In this research, we propose a locality-customized GSA Federated Active Learning (LG-FAL) method for federated active learning.We argued that in most federated active learning frameworks, local unlabelled samples are annotated by the aggregated global model's parameters, which totally ignores the localization of the samples, lead-

ing to the neglect over the importance of local models for local sample annotation. In addition, current federated active learning approaches usually are limited to one method, Federated averaging (FedAvg) to update global model parameter. Alternatively, we propose a locality-customized GSA Federated Active Learning method, LG-FAL, to tackle the aforementioned limitations. LG-FAL combines locality-customized active learning and Gravitational Search Algorithm (GSA) in a collaborative and effective way. In locality-customized active learning, both the local model as well as the global model are taken into consideration when annotating local samples, in which each data's overall uncertainty is a combination of both the local model's prediction entropy and the global model's prediction entropy. In GSA federated learning, global model parameter aggregations are achieved by GSA which is empowered with higher adaptability with a set of parameters to allow clients to move freely towards areas of high fitness calculated based on their masses (accuracy). Experiments and comparisons validate the performance of the LG-FAL, compared to other baselines.

Future study can focus on the following directions. First, we only verified the performance of LG-FAL with dense neural network. Future study can try to explore LG-FAL between different types of network architectures. Second, we only studied the proposed design with image datasets. In the future, more data formats will be explored using our proposed method, such as tabular data.

CHAPTER 7
CONCLUSION & FUTURE DIRECTIONS

In the rapidly evolving landscape of healthcare, data-driven approaches have emerged as a cornerstone for advancing medical research, diagnosis, treatment, and overall healthcare delivery. The potential benefits of harnessing the power of health data are vast, ranging from early disease detection to personalized treatment recommendations and improved patient outcomes. However, as we embark on this data-driven healthcare journey, we are confronted with a triad of formidable challenges: data bias, privacy concerns, and data scarcity.

In this thesis, in order to promote effective and robust health data analysis algorithms, we propose imbalanced learning, ensemble learning to address sample bias and specificity. We design novel federated learning approaches to better utilize limited health data resources and protect patient privacy. We implement federated active learning as a solution for label cost and scarcity in health data. The contributions and evaluation results of the proposed models are summarized as below.

7.1 CONCLUSION

1. Sample Bias and Specificity in Health Data.

- **Contributions.** We conduct a comprehensive exploration of the twin phenomena of sample bias and specificity in health data . First, we proposed to use imbalanced learning for 30-day hospital readmission prediction with National Readmission Databases (NRD). The main goal is to predict, at the time of a hospital discharge, whether the patient may return in 30 days or not in the fu-

ture. We created a set of features, using simple patient demographics, ICD-10 clinical modification (CM), and Clinical Classification Software Refined (CCSR) conversion, to represent each hospital visit. Because patient readmission is only a small portion of all patient visits, the machine learning task is severely challenged by the imbalanced class distributions. To solve the challenge, we used random under sampling (RUS) to create different copies of balanced sample sets. Ensemble classifiers were trained from balanced sample sets to build classifiers for readmission prediction. Secondly, we carry out systematic studies to understand data statistics for United States nationwide hospital admission, and further designs a machine learning framework for disease-specific 30-day hospital readmission prediction. We identified factors related to three key party of the hospital remissions: patient, disease, and hospitals, and reported national scale hospital admission statistic. Based on the data statistics, we created 526 features with five major types, including demographics features, admission and discharge features, clinical features, disease features, and hospital features. We collected six disease specific readmission datasets, which reflect the top six leading diseases of death. By using random under sampling and ensemble learning, combined with soft *vs.* hard voting and four types of machine learning methods, including gradient boosting, decision tree, logistic regress, and random forests, our experiments validate three major type of settings: (1) hard voting *vs.* soft voting, (2) random under sampling, and (3) disease specific readmission prediction.

2. Data Privacy and Locality in Health Data.

- **Contributions.** We introduce a unique federated learning approach designed specifically to address the data privacy and locality of health data. We propose a dynamic node matching method for federated learning. We argued that neural

networks are inherently non-transparent and unstable, and the same network structure may end up with very different weight values, even with the same training data and same parameter settings. Traditionally, existing methods, such as FedAvg, force neurons across sites to be matched with predefined order, and use fixed matching nodes during the FL learning process. Alternatively, we proposed a dynamic node alignment, FedDNA, approach which dynamically finds matching nodes across sites, and uses matched nodes to calculate weight for FL learning. FedDNA represents each neuron as a vector, using their weight values, and calculate distances between neurons to find matching nodes. Meanwhile, because finding marching nodes are computationally expensive, we proposed a minimum spanning tree (MST) based approach to speed up the matching, with matched nodes across all sites being linked by using an MST tree. So the matching process is simply the MST tree growing process.

3. Label Cost and Scarcity in Health Data.

- **Contributions.** In order to address the labeling challenges in health care data analysis, we propose a locality-customized GSA Federated Active Learning (LG-FAL) method for federated active learning. LG-FAL combines locality-customized active learning and Gravitational Search Algorithm (GSA) in a collaborative and effective way. In locality-customized active learning, both the local model as well as the global model are taken into consideration when annotating local samples, in which each data's overall uncertainty is a combination of both the local model's prediction entropy and the global model's prediction entropy. In GSA federated learning, global model parameter aggregations are achieved by GSA which is empowered with higher adaptability with a set of parameters to allow clients to move freely towards areas of high fitness calculated based on their masses (accuracy).

7.2 FUTURE DIRECTIONS

For imbalanced learning in health data analytics, the majority of the data used in our study are historical. Future research should explore the integration of real-time patient data, including wearable devices, electronic health records, and social determinants of health, to make predictions more timely and precise. Combining diverse data modalities such as medical images, free-text clinical notes, and structured patient records will be considered in order to lead to more comprehensive and accurate predictions. Future study can emphasize on the following directions for data privacy and locality in health data analytics. First, future study can try to explore node matching between different types of network architectures, such as convolutional neural networks. Second, more practical multi-class classification problems will be explored using our proposed methods. For the last direction, future study will aim to explore the locality-customized GSA using healthcare data. With the validated feasibility of our proposed method using MNIST dataset, we will explore and fine-tune it using healthcare data with more completed experiments.

BIBLIOGRAPHY

[1] D. A. E. Acar, Y. Zhao, R. M. Navarro, M. Mattina, P. N. Whatmough, and V. Saligrama. Federated learning based on dynamic regularization. *arXiv preprint arXiv:2111.04263*, Last accessed: September, 2023.

[2] A. Agarwal, C. Baechle, R. Behara, and X. Zhu. A natural language processing framework for assessing hospital readmissions for patients with copd. *IEEE journal of biomedical and health informatics*, 22(2):588–596, 2017.

[3] Agency for Healthcare Research and Quality. Clinical classifications software refined (ccsr) for icd-10-cm diagnoses. `https://hcup-us.ahrq.gov/toolssoftware/ccsr/dxccsr.jsp`, Last accessed: September, 2023.

[4] Agency for Healthcare Research and Quality. Nrd database documentation. `https://hcup-us.ahrq.gov/db/nation/nrd/nrddbdocumentation.jsp`, Last accessed: September, 2023.

[5] Agency for Healthcare Research and Quality. Overview of the nationwide readmissions database (nrd). `https://www.hcup-us.ahrq.gov/nrdoverview.jsp`, Last accessed: September, 2023.

[6] Agency for Healthcare Research and Quality Healthcare Cost and Utilization Project (HCUP). User guide:clinical classifications software refined (ccsr) version 2019.1. `https://hcup-us.ahrq.gov/toolssoftware/ccsr/DXCCSR-User-Guide-v2019-1.pdf`, Last accessed: September, 2023.

[7] L. Ahmed, K. Ahmad, N. Said, B. Qolomany, J. Qadir, and A. Al-Fuqaha. Active learning based federated learning for waste and natural disaster image classification. *IEEE Access*, 8:208518–208531, 2020.

[8] J.-H. Ahn, K. Kim, J. Koh, and Q. Li. Federated active learning (f-al): an efficient annotation strategy for federated learning. *arXiv preprint arXiv:2202.00195*, Last accessed: September, 2023.

[9] M. Almardini and Z. Ras. A supervised model for predicting the risk of mortality and hospital readmissions for newly admitted patients. *23rd Intl. Sym. on Foundations of Intel. Sys. (ISMIS)*, 2017.

[10] R. S. Antunes, C. André da Costa, A. Küderle, I. A. Yari, and B. Eskofier. Federated learning for healthcare: Systematic review and architecture proposal. *ACM TIST*, 13(4):1–23, 2022.

[11] R. N. Axon and M. V. Williams. Hospital readmission as an accountability measure. *Jama*, 305(5):504–505, 2011.

[12] A. Barta, G. C. McNeill, P. L. Meli, K. E. Wall, and A. M. Zeisset. Icd-10-cm primer. *Journal of AHIMA*, 79(5):64–66, 2008.

[13] S. Basu Roy, A. Teredesai, K. Zolfaghar, R. Liu, D. Hazel, S. Newman, and A. Marinez. Dynamic hierarchical classification for patient risk-of-readmission. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1691–1700, 2015.

[14] D. Berwick and A. Hackbarth. Eliminating Waste in US Health Care. *JAMA*, 307(14):1513–1516, 2012.

[15] S. Borofsky, A. K. George, S. Gaur, M. Bernardo, M. D. Greer, F. V. Mertan, M. Taffel, V. Moreno, M. J. Merino, B. J. Wood, et al. What are we missing? false-negative cancers at multiparametric mr imaging of the prostate. *Radiology*, 286(1):186, 2018.

[16] P. Branco et al. A survey of predictive modelling under imbalanced distributions. *CoRR*, abs/1505.01658, 2015.

[17] T. S. Brisimi, R. Chen, T. Mela, A. Olshevsky, I. C. Paschalidis, and W. Shi. Federated learning of predictive models from federated electronic health records. *International journal of medical informatics*, 112:59–67, 2018.

[18] K. Bryant, M. Knowles, and K. Adriana. 2022 year-end digital health funding: Lessons at the end of a funding cycle. `https://rockhealth.com/insights/2022-year-end-digital-health-funding-lessons-at-the-end-of-a-funding-cycle/`, Last accessed: September, 2023.

[19] H. Bueno, J. S. Ross, Y. Wang, J. Chen, M. T. Vidán, S.-L. T. Normand, J. P. Curtis, E. E. Drye, J. H. Lichtman, P. S. Keenan, et al. Trends in length of stay and short-term outcomes among medicare patients hospitalized for heart failure, 1993-2006. *Jama*, 303(21):2141–2147, 2010.

[20] L. M. Candanedo and V. Feldheim. Accurate occupancy detection of an office room from light, temperature, humidity and co2 measurements using statistical learning models. *Energy and Buildings*, 112:28–39, 2016.

[21] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. Intelligible models for healthcare: Predicting pneumonia risk and hospital 30-day readmission. *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1721–1730, 2015.

[22] Centers for Disease Control and Prevention. Health insurance portability and accountability act of 1996 (hipaa). `cdc.gov/phlp/publications/topic/hipaa.html`, Last accessed: September, 2023.

[23] Centers for Disease Control and Prevention. Icd-10-cm official coding and reporting guidelines. https://www.cdc.gov/nchs/data/icd/COVID-19-guidelines-final.pdf, Last accessed: September, 2023.

[24] Centers for Disease Control and Prevention. International classification of diseases (icd-10-cm/pcs) transition - background. https://www.cdc.gov/nchs/icd/icd10cm_pcs_background.htm, Last accessed: September, 2023.

[25] Centers for Disease Control and Prevention. Leading causes of death. https://www.cdc.gov/nchs/fastats/leading-causes-of-death.htm, Last accessed: September, 2023.

[26] Centers for Medicare & Medicaid Services. Hospital readmissions reduction program (hrrp). https://www.cms.gov/Medicare/Medicare-Fee-for-Service-Payment/AcuteInpatientPPS/Readmissions-Reduction-Program, Last accessed: September, 2023.

[27] Centers for Medicare and Medicaid Services. Icd-10-cm official guidelines for coding and reporting. https://www.cms.gov/Medicare/Coding/ICD10/Downloads/2020-Coding-Guidelines.pdf, Last accessed: September, 2023.

[28] A. Chang. The role of artificial intelligence in digital health. In *Digital health entrepreneurship*, pages 75–85. Springer, 2023.

[29] Y. Chen, X. Qin, J. Wang, C. Yu, and W. Gao. Fedhealth: A federated transfer learning framework for wearable healthcare. *IEEE Intelligent Systems*, 35(4):83–93, 2020.

[30] A. Choudhury and C. Greene. Evaluating Patient Readmission Risk: A Predictive Analytics Approach. *American Journal of Engineering and Applied Sciences*, 11(4):1320–1331, 2018.

[31] P. Cruz, B. Soares, J. da Silva, et al. Clinical and nutritional predictors of hospital readmission within 30 days. *European Journal of Clinical Nutrition*, 76(2):244–250, 2021.

[32] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[33] L. Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE signal processing magazine*, 29(6):141–142, 2012.

[34] Y. Deng, M. M. Kamani, and M. Mahdavi. Distributionally robust federated averaging. *Adv Neural Inf Process Syst*, 33:15111–15122, 2020.

[35] G. Du, J. Zhang, Z. Luo, F. Ma, L. Ma, and S. Li. Joint imbalanced classification and feature selection for hospital readmissions. *Knowl. Based Syst.*, 200:106020, 2020.

[36] D. Enthoven and Z. Al-Ars. Fidel: Reconstructing private training samples from weight updates in federated learning. *arXiv preprint arXiv:2101.00159*, Last accessed: September, 2023.

[37] C. B. Fisher, X. Tao, and T. Yip. The effects of covid-19 victimization distress and racial bias on mental health among aian, asian, black, and latinx young adults. *Cultural diversity and ethnic minority psychology*, 29(2):119, 2023.

[38] R. Graham and P. Hell. On the history of the minimum spanning tree problem. *Annals of the History of Computing*, 7(1):43–57, 1985.

[39] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi. Federated learning with compression: Unified analysis and sharp guarantees. In *AISTATS*, pages 2350–2358, 2021.

[40] M. Hao, H. Li, G. Xu, Z. Liu, and Z. Chen. Privacy-aware and resource-saving collaborative learning for healthcare in cloud computing. In *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pages 1–6, 2020.

[41] J. Harkey and R. Vraciu. Quality of health care and financial performance: is there a link? *Health Care Management Review*, pages 55–63, 1992.

[42] D. He, S. C. Mathews, A. N. Kalloo, and S. Hutfless. Mining high-dimensional administrative claims data to predict early hospital readmissions. *Journal of the American Medical Informatics Association*, 21(2):272–279, 2014.

[43] H. He and E. Garcia. Learning from imbalanced data. *IEEE Trans. on Knwl. & Data Eng.*, 21:1263 – 1284, 10 2009.

[44] Health Network Solutions. Understanding the icd-10 code structure. `https://www.healthnetworksolutions.net/index.php/understanding-the-icd-10-code-structure#Structure`, Last accessed: September, 2023.

[45] Healthcare Cost and Utilization Project and others. Nrd overview. `https://hcup-us.ahrq.gov/nrdoverview.jsp`, Last accessed: September, 2023.

[46] A. L. Hines, M. L. Barrett, H. J. Jiang, and C. A. Steiner. Conditions with the largest number of adult hospital readmissions by payer, 2011: statistical brief# 172. *Healthcare Cost and Utilization Project (HCUP) Statistical Briefs [Internet]*, 2014.

[47] M. Hopkins, E. Reeber, G. Forman, and J. Suermondt. Spambase data set. `https://archive.ics.uci.edu/ml/datasets/spambase`, 1999.

[48] L. Huang, Y. Yin, Z. Fu, S. Zhang, H. Deng, and D. Liu. Loadaboost: Loss-based adaboost federated machine learning with reduced computational complexity on iid and non-iid intensive care data. *Plos one*, 15(4):e0230706, 2020.

[49] S. F. Jencks, M. V. Williams, and E. A. Coleman. Rehospitalizations among patients in the medicare fee-for-service program. *New England Journal of Medicine*, 360(14):1418–1428, 2009.

[50] Y. Jin, X. Wei, Y. Liu, and Q. Yang. Towards utilizing unlabeled data in federated learning: A survey and prospective. *arXiv preprint arXiv:2002.11545*, Last accessed: September, 2023.

[51] A. Johnson, T. Pollard, et al. Mimic-iii clinical database (version 1.4). `https://doi.org/10.13026/C2XW26`, Last accessed: Semptember, 2023.

[52] A. Junqueira et al. A machine learning model for predicting icu readmissions and key risk factors: analysis from a longitudinal health records. *Health and Technology*, 9:297–309, 2019.

[53] M. Kahn. Diabetes data set. `https://archive.ics.uci.edu/ml/datasets/diabetes`, Last accessed: September, 2023.

[54] S. Khurshid, C. Reeder, L. X. Harrington, P. Singh, G. Sarma, S. F. Friedman, P. Di Achille, N. Diamant, J. W. Cunningham, A. C. Turner, et al. Cohort design and natural language processing to reduce bias in electronic health records research. *NPJ Digital Medicine*, 5(1):47, 2022.

[55] S. Kim, S. Bae, H. Song, and S.-Y. Yun. Re-thinking federated active learning based on inter-class diversity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3944–3953, 2023.

[56] R. Knevel and K. P. Liao. From real-world electronic health record data to real-world results using artificial intelligence. *Annals of the Rheumatic Diseases*, 82(3):306–311, 2023.

[57] H. M. Krumholz, A. R. Merrill, E. M. Schone, G. C. Schreiner, J. Chen, E. H. Bradley, Y. Wang, Y. Wang, Z. Lin, B. M. Straube, et al. Patterns of hospital performance in acute myocardial infarction and heart failure 30-day mortality and readmission. *Circulation: Cardiovascular Quality and Outcomes*, 2(5):407–413, 2009.

[58] E. W. Lee. Selecting the best prediction model for readmission. *Journal of Preventive Medicine and Public Health*, 45(4):259, 2012.

[59] S. Li, Y. Cheng, Y. Liu, W. Wang, and T. Chen. Abnormal client behavior detection in federated learning. *arXiv preprint arXiv:1910.09933*, Last accessed: September, 2023.

[60] Y. Li, X. Wang, R. Zeng, P. K. Donta, I. Murturi, M. Huang, and S. Dustdar. Federated domain generalization: A survey. *arXiv preprint arXiv:2306.01334*, Last accessed: September, 2023.

[61] D. Liu, K. Fox, G. Weber, and T. Miller. Confederated machine learning on horizontally and vertically separated medical data for large-scale health system intelligence. *arXiv preprint arXiv:1910.02109*, Last accessed: September, 2023.

[62] D. Liu, T. Miller, R. Sayeed, and K. D. Mandl. Fadl: Federated-autonomous deep learning for distributed electronic health record. *arXiv preprint arXiv:1811.11400*, Last accessed: September, 2023.

[63] K. Liu, Y. Fu, L. Wu, X. Li, C. Aggarwal, and H. Xiong. Automated feature selection: A reinforcement learning perspective. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

[64] K. Liu, X. Li, C. C. Zou, H. Huang, and Y. Fu. Ambulance dispatch via deep reinforcement learning. In *Proceedings of the 28th International Conference on Advances in Geographic Information Systems*, pages 123–126, 2020.

[65] E. Mahmoudi, N. Kamdar, N. Kim, G. Gonzales, K. Singh, and A. Waljee. Use of electronic medical records in development and validation of risk prediction models of hospital readmission: systematic review. *BMJ*, 369:m958, 2020.

[66] C. K. McIlvennan, Z. J. Eapen, and L. A. Allen. Hospital readmissions reduction program. *Circulation*, 131(20):1796–1803, 2015.

[67] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR, 2017.

[68] X. Min, B. Yu, and F. Wang. Predictive modeling of the hospital readmission risk from patients' claims data using machine learning: a case study on copd. *Scientific reports*, 9(1):2362, 2019.

[69] F. Naeem, M. Ali, and G. Kaddoum. Federated-learning-empowered semi-supervised active learning framework for intrusion detection in zsm. *IEEE Communications Magazine*, 61(2):88–94, 2023.

[70] D. C. Nguyen, Q.-V. Pham, P. N. Pathirana, M. Ding, A. Seneviratne, Z. Lin, O. Dobre, and W.-J. Hwang. Federated learning for smart healthcare: A survey. *ACM Computing Surveys (CSUR)*, 55(3):1–37, 2022.

[71] D. Opitz and R. Marlin. Popular ensemble methods: An empirical study. *J. of Artif. Intell. Research*, 11:168–198, 1999.

[72] S. Pandya, G. Srivastava, R. Jhaveri, M. R. Babu, S. Bhattacharya, P. K. R. Maddikunta, S. Mastorakis, M. J. Piran, and T. R. Gadekallu. Federated learning for smart cities: A comprehensive survey. *Sustainable Energy Technologies and Assessments*, 55:102987, 2023.

[73] R. Presotto, G. Civitarese, and C. Bettini. Semi-supervised and personalized federated activity recognition based on active learning and label propagation. *Personal and Ubiquitous Computing*, 26(5):1281–1298, 2022.

[74] A. Qayyum, K. Ahmad, M. A. Ahsan, A. Al-Fuqaha, and J. Qadir. Collaborative federated learning for healthcare: Multi-modal covid-19 diagnosis at the edge. *IEEE Open Journal of the Computer Society*, 3:172–184, 2022.

[75] J. D. Raffa, A. E. Johnson, Z. O'Brien, T. J. Pollard, R. G. Mark, L. A. Celi, D. Pilcher, and O. Badawi. The global open source severity of illness score (gossis). *Crit. Care Med.*, 50(7):1040–1050, 2022.

[76] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi. Gsa: a gravitational search algorithm. *Information sciences*, 179(13):2232–2248, 2009.

[77] J. Rau. New round of medicare readmission penalties hits 2,583 hospitals. *Kaiser Health News*, 2019.

[78] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, and X. Wang. A survey of deep active learning. *ACM computing surveys (CSUR)*, 54(9):1–40, 2021.

[79] N. Rieke, J. Hancox, W. Li, F. Milletari, H. R. Roth, S. Albarqouni, S. Bakas, M. N. Galtier, B. A. Landman, K. Maier-Hein, et al. The future of digital health with federated learning. *NPJ digital medicine*, 3(1):1–7, 2020.

[80] Y. Roh, G. Heo, and S. E. Whang. A survey on data collection for machine learning: a big data-ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, 33(4):1328–1347, 2019.

[81] B. S. Roy, A. Teredesai, K. Zolfaghar, R. Liu, D. Hazel, S. Newman, and A. Marinez. Dynamic hierarchical classification for patient risk-of-readmission. *ACM Knowledge Discovery and Data Mining*, pages 1691–1700, 2015.

[82] S. Russo, M. D. Besmer, F. Blumensaat, D. Bouffard, A. Disch, F. Hammes, A. Hess, M. Lürig, B. Matthews, C. Minaudo, et al. The value of human data annotation for machine learning based anomaly detection in environmental systems. *Water Research*, 206:117695, 2021.

[83] Sarthak, S. Shukla, and S. Prakash Tripathi. Embpred30: Assessing 30-days readmission for diabetic patients using categorical embeddings. In *Smart Innovations in Communication and Computational Sciences: Proceedings of ICSICCS 2020*, pages 81–90. Springer, 2021.

[84] B. Settles. Active learning literature survey. 2009.

[85] S. H. Shetty, S. Shetty, C. Singh, and A. Rao. Supervised machine learning: Algorithms and applications. *Fundamentals and Methods of Machine and Deep Learning: Algorithms, Tools and Applications*, pages 1–16, 2022.

[86] L. Shi and D. A. Singh. *Essentials of the US health care system*. Jones & Bartlett Learning, 2022.

[87] A. G. Singal, R. S. Rahimi, C. Clark, Y. Ma, J. A. Cuthbert, D. C. Rockey, and R. Amarasingham. An automated model using electronic medical record data identifies patients with cirrhosis at high risk for readmission. *Clinical Gastroenterology and Hepatology*, 11(10):1335–1341, 2013.

[88] I. Spasic, G. Nenadic, et al. Clinical text data in machine learning: systematic review. *JMIR medical informatics*, 8(3):e17984, 2020.

[89] M. Sun, T. Oliwa, M. E. Peek, and E. L. Tung. Negative patient descriptors: Documenting racial bias in the electronic health record: Study examines racial bias in the patient descriptors used in the electronic health record. *Health Affairs*, 41(2):203–211, 2022.

[90] T. Sun, D. Li, and B. Wang. Decentralized federated averaging. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[91] J. Sundström, J. Bodegard, A. Bollmann, M. G. Vervloet, P. B. Mark, A. Karasik, T. Taveira-Gomes, M. Botana, K. I. Birkeland, M. Thuresson, et al. Prevalence, outcomes, and cost of chronic kidney disease in a contemporary population of 2· 4 million patients from 11 countries: The careme ckd study. *The Lancet Regional Health–Europe*, 20, 2022.

[92] X. Tan, C.-C. H. Chang, L. Zhou, and L. Tang. A tree-based model averaging approach for personalized treatment effect estimation from heterogeneous data sources. In *International Conference on Machine Learning*, pages 21013–21036. PMLR, 2022.

[93] UCI Machine Learning Repository. Diabetes 130-us hospitals for years 1999-2008 data set. `https://archive.ics.uci.edu/dataset/296/diabetes+130-us+hospitals+for+years+1999-2008`, Last accessed: September, 2023.

[94] M. P. A. C. (US). *Report to the congress, Medicare payment policy*. Medicare Payment Advisory Commission, 2019.

[95] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer. A survey on distributed machine learning. *Acm computing surveys (csur)*, 53(2):1–33, 2020.

[96] H. Wang, Z. Cui, Y. Chen, M. Avidan, A. B. Abdallah, and A. Kronzer. Cost-sensitive deep learning for early readmission prediction at a major hospital. *Canada Proc. BIOKDD*, (17), 2017.

[97] S. Wang, M. E. Elkin, and X. Zhu. Imbalanced learning for hospital readmission prediction using national readmission database. In *2020 IEEE International Conference on Knowledge Graph (ICKG)*, pages 116–122, 2020.

[98] S. Wang and X. Zhu. Predictive modeling of hospital readmission: Challenges and solutions. *IEEE/ACM Transactions on Computational Biology and Bioinformatics, DOI: 10.1109/TCBB.2021.3089682*, 2021.

[99] S. Wang and X. Zhu. Feddna: Federated learning using dynamic node alignment. *Plos one*, 18(7):e0288157, 2023.

[100] X. Wu, J. Pei, C. Chen, Y. Zhu, J. Wang, Q. Qian, J. Zhang, Q. Sun, and Y. Guo. Federated active learning for multicenter collaborative disease diagnosis. *IEEE Transactions on Medical Imaging*, 2022.

[101] H. Xiao, K. Rasul, and R. Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, Last accessed: September, 2023.

[102] J. Xu, B. S. Glicksberg, C. Su, P. Walker, J. Bian, and F. Wang. Federated learning for healthcare informatics. *J. Healthc. Inform. Res*, 5(1):1–19, 2021.

[103] C. Yang, C. Delcher, E. Shenkman, and S. Ranka. Predicting 30-day all-cause readmissions from hospital inpatient discharge data. In *2016 IEEE 18th International conference on e-Health networking, applications and services (Healthcom)*, pages 1–6. IEEE, 2016.

[104] S. Yang, P. Varghese, E. Stephenson, K. Tu, and J. Gronsbell. Machine learning approaches for electronic health records phenotyping: a methodical review. *Journal of the American Medical Informatics Association*, 30(2):367–381, 2023.

[105] K. Yu and X. Xie. Predicting hospital readmission: A joint ensemble-learning model. *IEEE Journal of Biomedical and Health Informatics*, 24:447–456, 2020.

[106] J. Zhang, K. Zhang, Y. An, H. Luo, and S. Yin. An integrated multitasking intelligent bearing fault diagnosis scheme based on representation learning under imbalanced sample condition. *IEEE Transactions on Neural Networks and Learning Systems*, 2023.

[107] W. Zhang and S. Watanabe-Galloway. Ten-year secular trends for congestive heart failure hospitalizations: an analysis of regional differences in the united states. *Congestive Heart Failure*, 14(5):266–271, 2008.

[108] X. Zhu, J. Hurtado, and H. Tao. Localized sampling for hospital re-admission prediction with imbalanced sample distributions. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 4571–4578. IEEE, 2017.

[109] R. Zuckerman, S. Sheingold, E. Orav, J. Ruhter, and A. Epstein. Readmissions, observation, and the hospital readmissions reduction program. *New England Journal of Medicine*, 374(16):1543–1551, 2016.