Journal of Information Systems and Telecommunication
Vol.13, No.2, April-June 2025, 154-164
http://jist.acecr.org
ISSN 2322-1437 / EISSN:2345-2773

Credit Risk Prediction: An Application of Federated Learning

Sara Houshmand[1*], Amir Albadvi[1]

[1].Department of Industrial Engineering, Faculty of Engineering, Tarbiat Modares University, Tehran, Iran

Received: 27 Dec 2024/ Revised: 04 Jun 2025/ Accepted: 19 Jul 2025

Abstract

Credit risk is one of the major challenges faced by all financial institutions. Different institutions apply various techniques and models to reduce the risks associated with lending and other financial activities. However, due to the sensitivity of financial data and the diversity of modeling approaches, sharing data among institutions is extremely difficult, often impossible. As a result, improvements in credit risk prediction models typically occur in isolation, hindering collective progress toward higher accuracy and broader effectiveness. Federated learning offers a promising solution by allowing institutions to collaboratively train models without exposing or transferring sensitive data. In this research, we present a federated learning architecture for credit risk prediction that ensures privacy throughout the entire training process. Our results indicate that this approach not only protects data confidentiality but also maintains high predictive accuracy over numerous training rounds, offering a reliable and efficient framework for institutional adoption. The core contribution of this work is the development of a decentralized federated learning (FL) architecture tailored to heterogeneous, non-IID financial data. This framework enhances privacy, scalability, and regulatory compliance, and demonstrates performance advantages over traditional methods. In this article, we demonstrate that using five real-world credit risk datasets, the decentralized FL architecture significantly improves model accuracy (ranging from 71% to 99%) compared to traditional machine learning methods, especially in scenarios where privacy and communication efficiency are essential. While centralized FL achieves the highest average accuracy (up to 83%), the decentralized model provides a strong trade-off between performance and privacy-aware collaboration.

Keywords: Federated Learning (FL); Credit Risks; Financial Institutions; Heterogeneous Data; Decentralized Federated Learning (DFL) Architecture.

1- Introduction

Credit risk is among the major risks which are associated with commercial banks and other financial institutions. It refers to the probability of default in the repayment of the principal amount along with interest on loans, which may adversely affect organizational performance and, in fact, the economy as a whole [22]. The forecast of credit risk has become indispensable in general, and within financial industries in particular, as a result of the insightful support it gives to the organizational decision-making processes enabling the organizations to avoid potential losses [29].

However, despite the importance of credit risk prediction, existing methods often fail to address the unique challenges of financial institutions, such as the need for secure data sharing, managing heterogeneous and non-IID (non-independent and identically distributed) data, and fostering collaboration without compromising privacy. With the development of data-driven technologies, machine learning has attracted increasing interest in credit risk prediction. However, most machine learning models collect and process data in the centralized server, which might bring serious security leakage and privacy violation problems [5]. Moreover, the reluctance of organizations to share sensitive data for model training due to privacy and security concerns further complicates the collaboration needed for accurate predictions.

Therefore, Google, in the year 2016, proposed a more recent AI-based technology known as federated learning [29]. Instead of the sharing of local data, federated learning secures sensitive data by sharing local models. This is

✉ **Corresponding Author**
Sarahoushmand99@gmail.com

helpful when organizations want to train larger datasets but cannot share data due to legal, strategic, or economic reasons. Federated learning can, therefore, enable different organizations to collaborate without sharing data with full security and privacy assurance of the data through decentralized models [29].

Despite the promise of federated learning, existing FL models whether centralized or decentralized often struggle with the specific challenges faced by financial institutions, such as the management of heterogeneous, non-IID data distributions and ensuring robust privacy protections. This article proposes a decentralized architecture for heterogeneous data environments to predict credit risk using federated learning, addressing these challenges more effectively than current methods.

The main contributions of this paper are as follows:

1- A novel decentralized federated learning architecture is proposed for credit risk prediction using heterogeneous, non-IID financial datasets, eliminating reliance on a central server.
2- Implementation of a socket-based communication framework to simulate real-world decentralized environments and facilitate peer-to-peer model aggregation.
3- Experimental evaluation on five real-world credit datasets, comparing traditional machine learning, centralized FL, and decentralized FL approaches across multiple performance metrics.
4- Discussion of accuracy, scalability, and privacy trade-offs, with results showing decentralized FL achieves competitive accuracy (71%–99%) while enhancing data privacy and reducing central dependency.

2- Literature Review

2-1- Credit Risk

Risks might have serious and sometimes unpredictable consequences on organizations, banks, companies, or even the wider economy. Credit scoring is used to evaluate credit risk prediction, which involves quantifying the probability of future default. Credit scoring methods can be divided into two categories: judgmental and operational scoring. While Judgmental scoring systems are based on specific customer attributes, and the scores are assigned accordingly. In contrast, in operational scoring systems, much emphasis is placed on predictive models of financial variables [22].

For instance, a judgmental scoring could be the 5C criterion adopted by banks, which entails [11],[14]:

Character: past activities, personal credit;
Capacity: income capacity;
Capital: the financial statement evaluation of the individual;
Coverage: the assets given to institutions for coverage purposes during credit issuance;
Conditions.

Therefore, the other criteria for judgmental scoring include the LAPP method which includes: Liquidity; Activity; Profitability; Potential [11],[14].

Operational scoring techniques rely more on quantitative analysis for predicting credit risk, employing models and techniques such as mathematical programming, nearest neighbor algorithms, artificial neural networks, genetic algorithms, etc. [27],[15],[17]. The following table, Table 1, summarizes some of these.

Table 1: Credit scoring methods in credit risk

Scoring Methods	Criteria	Sources
judgmental scoring	5C	Rouintan, P., 2006
	LAPP	
Operational scoring	mathematical programming	Rasouli, M., 2022
	nearest neighbor algorithms	
	ANN	
	genetic algorithms	

2-2- Machine Learning & Federated Learning

The most salient uses include finance, health, transportation, and e-commerce. Considering the popularity that machine learning has gained, much attention must be paid to privacy in data and security. Traditional machine learning methods use a centralized approach in model training by collecting training data on a central server. Data gathering on a central server remains one of the biggest challenges in machine learning when sensitive information exists and causes quite several security threats to data privacy [21].

This prompted Google in 2016 to introduce a new technology in artificial intelligence called Federated Learning. As opposed to the sharing of local data, federated learning protects sensitive data by sharing local models. The decentralized approach to model training will be the technique. It will be very useful in cases where various regions want to train models on larger datasets than their own, but due to legal, strategic, or economic reasons, sharing the data with others cannot be done. According to the definition of machine learning. Federated Learning is a framework where a global model is designed in advance to solve collaboration problems among data owners without data exchange. The central server aggregates optimized models from all regions. Since no data is being exchanged, there is no risk of exposure to user privacy. Various sections or regions in a federated learning system send their own trained models to a center server. Further, this training may be iterated until a satisfactory level of accuracy is achieved [29].

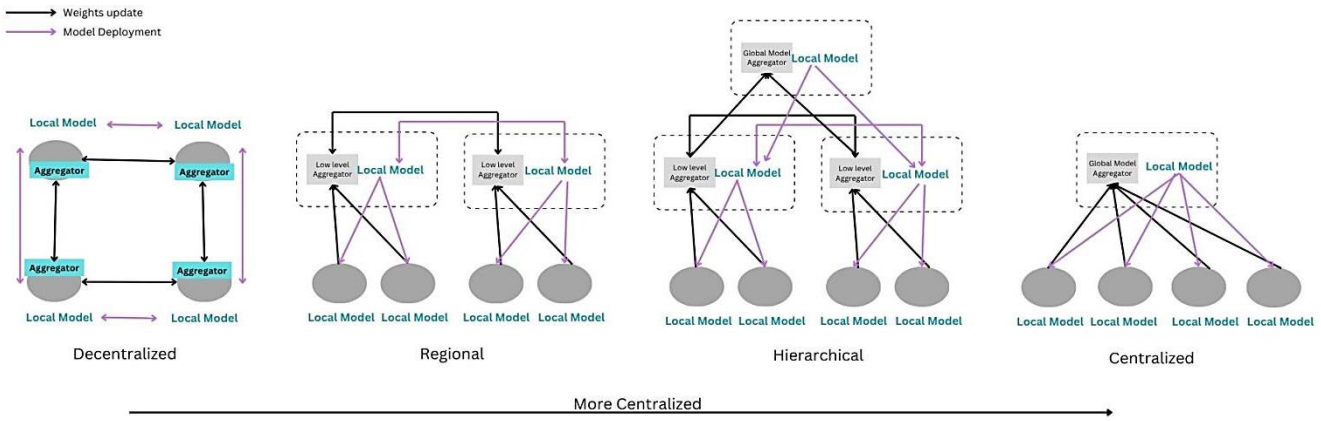

Fig. 1 Application of FL

After having given a view on what Federated Learning is, we go further into the significant benefits of federated learning against traditional centralized machine learning methods in more detail [6]:

Data Security: The training datasets remain on the devices, hence there is no need for a centralized dataset.

Data Diversity: Allows access to heterogeneous data when the sources of data can only communicate their data at certain times.

Continuous Real-time Learning: Models keep improving continuously with customer data without gathering the data for continuous learning.

Hardware Efficiency: It deploys less complex hardware as the models in federated learning do not require a high-end central server for data analysis.

Federated learning is widely applicable in numerous fields such as finance, healthcare, and transportation, positioning it as a transformative technology in our data-driven world [6],[7]. This article will concentrate on one of its primary uses in the financial sector: credit risk predictions, which will be elaborated on in the next section. Figure 1 illustrates the various applications of federated learning.

2-3- Federated Learning in Credit Risk Prediction

Given the various methods for predicting credit risk discussed in previous sections, each technique has its own set of advantages. However, when organizations want to enhance their model performance through data sharing but face challenges due to the sensitive nature of financial information, federated learning emerges as a viable solution. This approach enables two or more financial institutions to improve their model performance without sharing actual data. Federated learning functions as a decentralized method, allowing collaborating organizations to refine their models by exchanging model parameters rather than data on a centralized server [2]. Research studies in this area are summarized in Table 2.

The first article introduces FedKT, a federated learning framework designed to enhance credit scoring while ensuring data privacy. However, the study falls short in providing adequate comparative analysis among different machine learning methods, leaving a gap in understanding its relative effectiveness and does not address how federated learning can handle heterogeneous and non-IID data distributions commonly found in financial contexts [26].

The second article investigates the use of federated learning for mortgage credit risk assessment, with a focus on the Freddie Mac dataset. While there are promising advancements for smaller financial institutions, the research is constrained by its narrow focus on a specific dataset and the examination of only loans with a final status. These limitations hinder the generalizability of the findings to wider contexts. Additionally, the study lacks a comprehensive comparison between centralized, decentralized, and federated learning approaches, which leaves out critical insights on the relative performance and scalability of these methods. There is also a lack of comparisons between centralized and decentralized architectures, as well as other federated learning methods, which could have enriched the research [9].

The third article delves into federated learning paired with the SecureBoost algorithm for credit evaluation among micro and small enterprises (MSEs). This approach effectively tackles privacy issues and data silo challenges, resulting in enhanced accuracy and stability. Furthermore, the study mainly depends on two external data sources (credit and electricity consumption data) and does not consider other potentially relevant datasets, limiting the model's applicability across various contexts. Moreover, this study does not explore the impact of non-IID data and lacks an in-depth analysis of the performance across different federated learning architectures. [27].

All the mentioned articles share a common limitation: they lack a thorough comparison of the three approaches centralized, decentralized, and federated learning.

Furthermore, they depend on a narrow range of machine learning methods for their comparisons, and some studies use smaller, more homogeneous datasets. These gaps highlight the need for a more comprehensive and scalable solution that can handle heterogeneous, non-IID data, and provide a better understanding of the comparative performance across different architectures. In general, research in this area is limited and still in its early stages, with no substantial work done in Iran so far.

Table 2: Papers in credit risk prediction using federated learning

Title	Source
A novel federated learning approach with knowledge transfer for credit scoring	Zhongyi Wang et al., 2024
Federated Learning for Credit Risk Assessment	Chul Min Lee et al., 2023
MSEs Credit Risk Assessment Model Based on Federated Learning and Feature Selection	Zhanyang Xu et al., 2023

3- Methodology

3-1- Dataset

In credit risk assessment and prediction, several factors can affect a borrowers' ability to repay their debts. These factors help financial institutions estimate the likelihood of repayment. Common features used to evaluate credit risk include:

Demographic Characteristics: Information such as age, gender, marital status, and education level, which aid in analyzing the borrower's profile.

Financial Characteristics: This encompasses income, employment status and history, debt-to-income ratio, current financial obligations, and levels of savings and assets.

Loan-Specific Features: Factors such as loan amount, term, interest rate, and purpose relate to the specific conditions of the loan granted.

These features may vary based on the policies of different institutions and are sometimes combined to create a more accurate evaluation of the borrower's overall risk. In this Fig. 1 Applications of FL study, open-source data from various institutions will be utilized to develop an effective model in the field of credit risk:

The Univ.AI Hackathon dataset includes demographic details of loan applicants, such as age, income, job experience, and marital status. It is a binary-class dataset(0 and 1) with no missing values, comprising around 252,000 records [19].

The Loan Data from 2007 to 2015 features issued loans along with financial attributes like loan amount and interest rate. This dataset contains 73 features and approximately 855,000 records, though some values are missing [12].

The German Credit Card dataset emphasizes credit history and personal information, consisting of 21 features and 1,000 records, with no missing values [1].

The Credit Risk dataset encompasses features such as age, income, loan amount, and loan status. It is a binary-class dataset with 12 features and 32,000 records [8].

The Credit Risks dataset includes payment history and credit-related details, such as payment delays and the number of bank accounts. It has 28 features and around 100,000 records, is categorized into three classes (good, bad, standard), and contains some missing values [16].

Finally, it has been tried to use different datasets and implement this architecture on these datasets. A summary of the dataset is given in Table 3.

Table 3: Summary of Dataset Features

	Dataset	Features	Records
1	Univ.AI Hackathon Dataset	13	252,000
2	Loan Data (2007 - 2015)	73	855,000
3	Credit Risk Dataset	12	32,000
4	German Credit Card Dataset	21	1,000
5	Credit Risks Dataset	28	100,000

3-2- Research Method

In our research, we collected datasets for credit risk prediction from various sources, each featuring distinct characteristics and exhibiting non-IID (non-independent and non-identically distributed) properties. This indicates that the data originates from different regions, customer segments, or financial contexts, with each dataset presenting its own unique distribution. For instance, one dataset may concentrate on user behavior, while another might focus on economic conditions or credit history, leading to diverse data distributions across the datasets [3]. The non-IID nature poses challenges in federated learning, as the model needs to manage data with differing statistical properties. These variations in data distributions can complicate the federated model's ability to converge effectively or perform well across all datasets. Although these differences mirror the diversity found in real-world credit risk scenarios and can enhance model adaptability, they also bring about complexities such as slower convergence, inconsistent performance, and inefficiencies in resource use. In this study, we have implemented various data preprocessing techniques to mitigate the effects of non-IID data [24],[25]. Since the data across institutions have different characteristics and cannot be combined due to their diverse nature, federated learning can be employed. After identifying the datasets, the next step is to outline the various stages necessary for implementing credit risk prediction using federated learning. The first step involves determining the type of architecture to be used in federated learning. Below, the main types of federated learning architectures are introduced, along with an explanation of the architecture utilized in this research.

3-2-1- Federated Learning Architecture

Federated learning employs various architectures to aggregate and update models. Here, four commonly used architectures are presented [18],[28].

Centralized Architecture
In a centralized architecture, a central federated server manages the training process. Clients send their local model updates to this central server, which aggregates the updates to form a global model. This setup is effective when a limited number of organizations are involved and a high level of trust exists among them.

Decentralized Architecture
The decentralized architecture does not depend on a central federated server. Instead, participating organizations communicate directly with each other or through a decentralized network to share model updates and coordinate training. This architecture is ideal when there is a need to minimize reliance on a central authority or when concerns about the availability or security of a central server arise.

Regional Architecture
In a regional architecture, participating organizations are grouped into zones where local models are trained collaboratively. Model updates from different regions can then be aggregated at a higher level (e.g., a central node for regional updates). This approach is suitable for balancing regional collaboration with national-level cooperation and helps address regional variations in data and needs.

Hierarchical Architecture
In a hierarchical architecture, there are several layers of model aggregation. Local models are combined within subgroups or nodes to produce intermediate-layer models, which are then further combined to create a global model. This structure is ideal for complex organizational frameworks or situations that require multiple layers of collaboration and data sharing.
Figure 2 illustrates these federated learning architectures.
A decentralized architecture is the most appropriate choice for this application. In this model, organizations and institutions function parallel and non-hierarchically while still fostering collaboration. Other architectures depend on a central server, which does not fit the ecosystem of financial institutions. Moreover, the selected datasets indicate that the data cannot be grouped under a central server. Due to the diverse nature of the data, a decentralized architecture seems to be the right option.

3-2-2- Step by Step Procedure

After defining the architecture, the next step involves outlining the procedures for designing a decentralized framework. Each client, represented by different datasets and referred to as clients in federated learning, may employ various methods to train their data, including different machine learning techniques or neural networks.
The general steps include [23],[13]:

1- Local Data Preprocessing
2- Training Local Models
3- Peer-to-Peer Client Communication: Clients exchange model coefficients (gradients of the learning model) rather than raw data.
4- Applying Decentralized Aggregation Methods: This step utilizes decentralized aggregation methods such as averaging or other combination techniques.
5- Establishing Consensus in the Decentralized Learning Process: This involves determining when and among which clients consensus should take place. During the initial phase of training local models, consensus starts with random numbers.

The steps of decentralized architecture illustrated in Figure 3 are as follows:

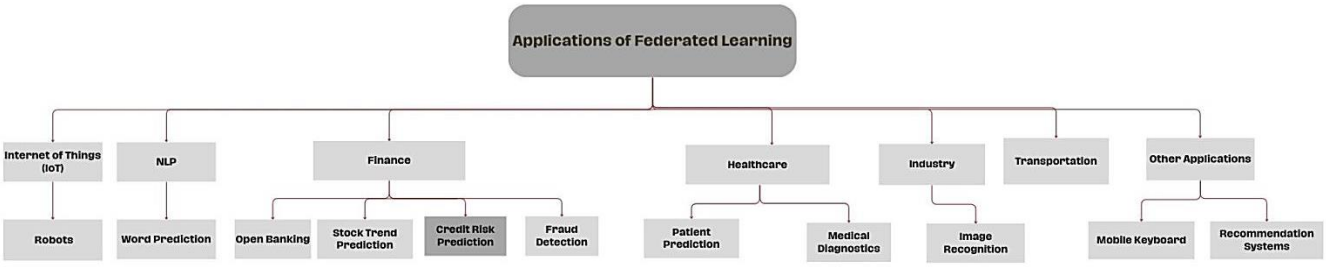

Fig. 2 Federated Learning Architecture [Adapted from 28]

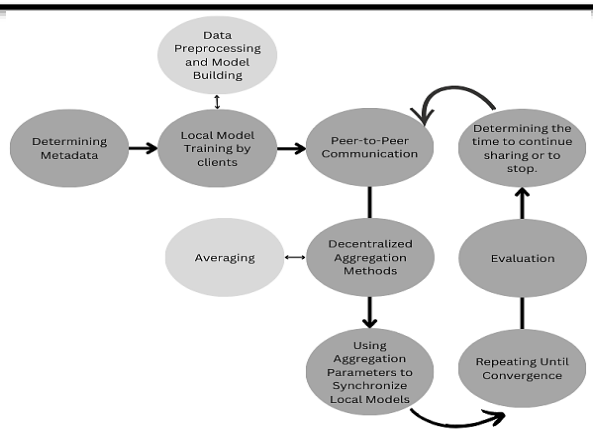

Fig. 3 Steps for designing a decentralized architecture

4- Implementation

The implementation of federated learning is the next significant stage after defining the datasets and architecture. Although federated learning can be accomplished using Python, several specialized frameworks and tools have been developed to assist in the effective execution of federated learning algorithms. These frameworks provide vital building blocks for creating decentralized, privacy-preserving machine-learning models, including federated computation, model aggregation, and privacy-preserving techniques, along with compatibility with established machine-learning libraries. For this study, TensorFlow Federated (TFF) [20] was chosen due to its robust support for federated learning, extensive documentation, and seamless integration with TensorFlow, one of the most widely used machine learning libraries. TFF offers a comprehensive and flexible platform for building federated learning systems, providing key features such as federated computation, model aggregation, and privacy-preserving techniques. These capabilities are essential for implementing decentralized, privacy-preserving machine learning models, particularly in environments where sensitive data cannot be shared.The choice of TensorFlow Federated is also motivated by its close integration with TensorFlow, which allows for a smooth transition from traditional machine learning workflows to federated learning without the need to learn a new framework.Furthermore, TFF's modular design makes it easier to experiment with various federated learning algorithms, which is beneficial for research and practical applications, especially in financial institutions where privacy and data heterogeneity are critical concerns.

However, there are certain limitations faced when using TensorFlow Federated. One challenge is the learning curve associated with setting up the federated learning system, particularly in terms of managing communication between clients and the server, as well as handling the distribution of data. Additionally, although TFF is well-documented, it is still evolving, and certain advanced features may require customization or additional workarounds. Performance optimization for federated learning algorithms in TFF can be complex, particularly when working with large-scale datasets and ensuring efficient resource utilization across distributed environments. Despite these challenges, TensorFlow Federated's comprehensive nature makes it a strong and suitable choice for implementing federated learning in this study. The implementation will follow the process illustrated in Figure 4.

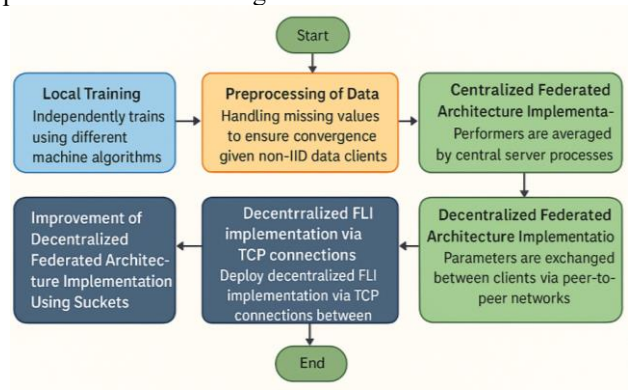

Fig. 4 Steps of Implementing a Decentralized Federated Architecture

4-1- Local Training of Datasets

To initiate our research, we begin by preparing each dataset. This preparation step involves eliminating outliers and handling any incomplete values to enhance the precision and performance of our learning models. We also convert categorical data into numerical values, making it suitable for machine learning algorithms and neural networks. Each dataset undergoes analysis through various methods, incorporating machine learning techniques and neural networks. These different machine learning methods were selected to enable a comprehensive comparison between federated learning and various individual approaches. We implement thorough verification procedures to guarantee the models' reliability and precision. This verification process consists of dividing the data into training and testing sets, developing the models with the training data, and evaluating their effectiveness with the testing data. Following the preparation and individual training phases, we present the models' results in Table 4. This table demonstrates that multiple methods have been implemented across the datasets, enabling a comprehensive comparison between federated learning and various individual steps.

✉ **Corresponding Author**
Sarahoushmand99@gmail.com

Table 4: A summary of local dataset models

Dataset	Models	Accuracy
Univ.AI Hackathon Dataset	Gradient Boosting	59 %
Loan Data (2007 - 2015)	Random Forest	99 %
Credit Risk Dataset	MLP	87 %
German Credit Card Dataset	Adaboost	70 %
Credit Risks Dataset	DecisionTree	75 %

These results show significant variation in model performance across datasets, with Random Forest achieving 99% accuracy on Loan Data while Gradient Boosting only reached 59% on the Hackathon Dataset. This variation highlights the importance of selecting appropriate algorithms for specific data characteristics.

4-2- Preprocessing of Datasets

In this section, we selected TensorFlow Federated (TFF) as the platform for implementing federated learning. TFF offers various installation options, encompassing local setups and cloud-based environments like Google Colab. For this research, we selected a local installation, and after encountering some compatibility issues with specific versions, we established Python 3.8.10 as our development environment. Since TFF necessitates datasets to maintain identical numbers of features for model participation, the non-IID (Non-Independent and Identically Distributed) nature of the data required special attention during preprocessing. The datasets were inherently heterogeneous, containing different distributions across clients (organizations), and were often unbalanced with features exhibiting varying importance across clients. To address these challenges, we employed feature selection as a critical preprocessing step. The goal was to eliminate features exhibiting minimal correlation with the target label, as they were deemed insignificant for prediction purposes, especially considering the non-IID nature where some features may be more relevant to certain subsets of data than others.We also incorporated techniques like normalization using MinMaxScaler to ensure consistent feature scaling across all clients, which is crucial for federated learning models to ensure proper model aggregation and convergence. Additionally, we divided the data into training and validation sets, ensuring that the data distribution across these sets remained representative of the real-world non-IID nature. By reducing the feature set to 11 relevant features, we achieved multiple advantages: reduced model complexity, accelerated training times, a decreased risk of overfitting, and improved model interpretability, making the model more manageable and reliable. These 11 features were selected based on their Pearson correlation with the target label, aiming to remove attributes with minimal predictive value. The Pearson correlation coefficient measures the strength and direction of the linear relationship between two variables, making it a suitable choice for identifying features most closely associated with the output. Given the non-IID nature of the data across clients, feature relevance varied, and this method allowed us to retain the most informative features per dataset. Upon completion of preprocessing, the data was transformed into TensorFlow format to be compatible with TFF and was then divided into the necessary training and validation sets. We subsequently designed the federated learning model using Keras, incorporating a Sequential architecture with four Dense layers. The intermediate layers consisted of 32 and 64 nodes with ReLU activation, while the output layer contained a single node with sigmoid activation for binary classification. Finally, we transformed the model into a federated format using TFF, establishing it for implementation in a federated learning environment [4],[10].

4-3- Centralized FL Implementation

It is crucial to define the model and articulate the type of algorithm discussed in previous sections. For this purpose, TFF's federated averaging method within a centralized framework (consistent with the centralized architecture described in Section 2-2) is implemented, which incorporates updates from client models and updates the global model on the server. This process is executed over a fixed number of epochs, established at 50. Each of the five datasets undergoes training, and subsequently, the model is evaluated using test data. The results are presented in Table 5. The assessment results demonstrate that the federated averaging model functions effectively, and in some cases, it even exceeds the performance of models trained independently.

Table 5: The accuracy of centralized federated learning

Accuracy	83 %
Precision	43 %
Recall	49 %
Loss	45 %

The centralized FL achieved a solid 83% accuracy, demonstrating effective model aggregation across diverse datasets. While precision and recall values indicate room for optimization, the model successfully established a baseline for federated learning implementation that can be further improved through hyperparameter tuning.

4-4- Decentralized FL Implementation and Improvement

In the previous centralized architecture, all datasets, formatted in TensorFlow, initiated the training process using the same model definition and random parameters, ultimately leading to convergence. In contrast, the decentralized architecture (as defined in Section 2-2) processes each dataset successively, transferring refined parameters to the next dataset until the model converges, ensuring that each dataset's parameters achieve an optimal learning level. For

decentralized implementation, data preprocessing must be completed first. Then, analogous to the centralized architecture, a collaborative learning algorithm particularly distributed averaging is implemented.

The first step involves generating a random parameter using TensorFlow Federated (TFF), which is applied to the first client. Updated parameters are then sequentially passed to the fifth client. Every possible permutation of the five clients is considered, with parameters recalculated for different configurations. However, this method has several drawbacks: It does not converge, and the model's convergence cannot be detected since each round is treated independently.

This step is time-consuming, as it evaluates all potential configurations, with a single execution taking up to four hours. Due to these limitations, this method was deemed unreliable, prompting the implementation of a second method. The pseudocode of the first step is provided in Algorithm 1.

Algorithm 1: Pseudocode of Decentralized FL in the First Approach

- Generate a random initial model parameter using TFF.
- For each permutation of 5 clients:
  - Set initial state = random parameter.
  - For each client in the permutation:
    - Use client's data to update the model.
    - Pass updated model to the next client.
- Repeat the above for all permutations (5! = 120 total).
- After all updates, evaluate the final model.
- Note: This method does not ensure convergence and is time-consuming, as it explores all possible client orders.

The Second step tackles the convergence issues identified in the first method, we calculate the loss function for all regions after updating each parameter. In the subsequent learning epoch, we select the region with the lowest loss to initiate the process from the strongest region. At each step, we continue to choose the region with the minimum loss until we achieve convergence. For each client, convergence means the model's accuracy stops improving after multiple training rounds and reaches a stable level. However, this step has a limitation: the region with the minimum loss remains unchanged at each step, which
hinders the model's ability to converge. The results are presented in Table 6.

According to the table, all regions demonstrate satisfactory learning, except for the fifth dataset. Consequently, a third method has been introduced.

Table 6: Accuracy of Decentralized FL in the Second Approach

Dataset	Accuracy
Univ.AI Hackathon Dataset	63 %
Loan Data (2007 - 2015)	86 %
Credit Risk Dataset	87 %
German Credit Card Dataset	99 %
Credit Risks Dataset	28 %

The third step: After adjusting each parameter for the regions, the loss function for all regions is computed. During the next learning cycle, the minimum loss is chosen to initiate the process from the strongest region. The key difference from Method 2 is that this method begins with the strongest region, and the parameters for each region are refined until convergence is achieved. The results can be found in Table 7 and The pseudocode of the third step is provided in Algorithm 2.

Algorithm 2: Pseudocode of Decentralized FL in the Third Approach

- Initialize the evaluator and model state.
- Define a list to store the loss values for each region.
- Set the round counter to zero.
- While convergence is not reached and the round limit is not exceeded:
  - If it is the first round:
    - For each of the 5 regions:
      - Train the model using the region's training data.
      - Evaluate the model on the region's validation data.
      - Record the loss value.
  - Else:
    - Identify the region with the lowest loss.
    - Train the model on that region's data to update parameters.
    - For each of the remaining regions:
      - Train the model on the region's data.
      - Evaluate and update the corresponding loss value.
  - Increase the round counter by one.
- Once the stopping condition is met (e.g., 10 rounds):
  - Output the final loss values.
  - Output the final evaluation metrics for all regions.

Table 7: Accuracy of Decentralized FL in the Third Approach

Dataset	**Accuracy**
Univ.AI Hackathon Dataset	63 %
Loan Data (2007 - 2015)	87 %
Credit Risk Dataset	87 %
German Credit Card Dataset	99 %
Credit Risks Dataset	28 %

Notably, the fifth dataset performed poorly during the initial training rounds. Instead of using Federated Averaging, this study adopted stochastic gradient descent (SGD) to explore whether it could enhance the learning process in the decentralized architecture. However, the use of SGD did not significantly impact the final model accuracy. Given this, adjustments were made specifically for the fifth dataset.

While the first four datasets represented binary classification problems and learned effectively, the fifth dataset originally contained three classes. To align it with the others and address convergence issues, it was preprocessed into a binary format by combining the "standard" and "good" credit risk categories into a single "good" label. Training was resumed for several rounds under this revised configuration, and the updated results are reported in Table 8.

Table 8: Accuracy of Decentralized FL in the Third Approach after the corrections

Dataset	Accuracy
Univ.AI Hackathon Dataset	71 %
Loan Data (2007 - 2015)	89 %
Credit Risk Dataset	87 %
German Credit Card Dataset	99 %
Credit Risks Dataset	79 %

As can be seen from the table, the results have been considerably improved, proving the effectiveness of Method 3 after the corrections. A new concept is introduced in the next section, more in line with real world applications to which the decentralized architecture must be fitted.

4-5- Decentralized Federated Implementation Using Sockets

After the decentralized architecture of federated learning is implemented, one should know that since all the datasets are kept locally in one environment, this setting is not very realistic. In real-world scenarios, clients are usually not on the same machine and can be distributed across different machines. This way, the problem of communication overhead between the clients arises. The idea of sockets in Python was used to assess whether the usage of more than one machine has any impact on execution time. Sockets serve as an interface for communication where messages can be sent and received across different regions. Each region in this architecture works independently, and after each training phase, it sends its model updates to other regions using sockets. These are then aggregated, using certain algorithms, to enhance the overall model. This cycle continues until the model achieves satisfactory convergence. This step, which leverages the robust capabilities of sockets for managing concurrent and distributed communications, proves to be an effective and efficient method to implement decentralized federated learning. The above steps were first performed for the implementation: all the preprocessing steps were carried out earlier. After that, a sixth intermediary client was used to collect the trained parameters from the regions and send them further to the next regions until convergence. The decentralized federated learning architecture, implemented using sockets, requires approximately 120 seconds. Running on a single machine takes around 90 seconds. In other words, running decentralized federated learning across multiple machines takes 30 more seconds, which is fully expected. Because the socket implementation manages the communication load.

4-6- Results and Discussion

In the previous section, we explored the implementation and examination of federated learning. After choosing the platform and architecture type, the steps involved included data preprocessing, feature selection, and converting the data into TensorFlow format. Initially, we defined the Keras model and adapted it into a federated model. Following that, we implemented centralized federated learning (Section 2-2) using federated averaging and evaluated the results. In the next phase, we tackled the convergence issue by applying three different methods for decentralized federated learning. The third method yielded improved results through specific enhancements. Finally, to simulate real-world conditions and measure communication time, we executed a decentralized implementation using sockets, which facilitated effective communication across various regions. It is important to note that this implementation is not exhaustive. The data volume was not fully representative, and in some cases, the model was not optimized or did not achieve significant progress. While certain methods did not result in improvement, the overall performance was better compared to traditional machine learning approaches. Additionally, further exploration of techniques like averaging could have been beneficial. A summary of the results is provided in Table 9. As summarized in Table 9, traditional machine learning achieves the highest accuracy in certain scenarios, such as loan datasets, but its performance can vary sometimes it even falls short compared to federated learning. On the other hand, decentralized federated learning shows significant improvements over traditional machine learning for specific datasets, like the Hackathon dataset and the German credit card dataset. Generally, centralized federated learning tends to provide the best performance in most situations, primarily due to enhanced data coordination. In some cases, decentralized federated learning can surpass traditional machine learning. This comparison indicates that adopting a federated learning approach can enhance model accuracy in many instances. The best method should be chosen based on the type of data and the model being utilized. One other way to compare is by using confidence interval tests. The confidence intervals used to assess the accuracy of the models are based on the statistical method known as "Interval Estimation." This method typically relies on either the t-student distribution or the normal distribution for its calculations. It determines a range where we expect the true model accuracy to lie. Initially, the models' accuracy is calculated, and then the confidence interval is established using relevant statistical formulas that take into account the data and sample size. The primary metric for these confidence intervals is accuracy, which reflects the proportion of correct predictions to the total predictions.

Depending on the data characteristics, either the normal distribution or the t-student distribution may be applied in this process. In this study, confidence intervals were computed for three distinct models: the Federated Centralized Model, the Federated Decentralized Model, and the Machine Learning Model. The findings reveal that The Federated Centralized Model exhibits a narrow confidence interval (0.83–0.83), indicating high stability and accuracy. In contrast, the Federated Decentralized Model exhibits a broader confidence interval ranging from 0.72 to 0.98, indicating greater variability in its accuracy, with the potential for this model to outperform the others in certain scenarios. The Machine Learning Model also presents a wider confidence interval, spanning from 0.59 to 0.97, which implies that its performance may be less stable. Overall, the results suggest that while centralized federated learning models offer stable and consistent performance, decentralized federated models, despite their higher variability, may still offer potential benefits in certain contexts. Traditional machine learning models, however, may need further refinement to achieve performance stability comparable to federated learning approaches. A summary of the results is provided in Table 10.

Table 9: Comparison of accuracy between MLand FL

Dataset	accuracy	accuracy	accuracy
	Centralized FL	Decentralized FL	ML
Univ.AI Hackathon Dataset	83 %	71 %	59 %
			Gradient Boosting
Loan Data (2007 - 2015)		89 %	99 %
			Random Forest
Credit Risk Dataset		87 %	87 %
			MLP
German Credit Card Dataset		99 %	70 %
			Adaboost
Credit Risks Dataset		79 %	75 %
			DecisionTree

Table 10: Federated vs. Machine Learning: Confidence Intervals and Accuracy

Model	Confidence Interval	Mean Accuracy	Range of Confidence Interval
FL(Centralized)	(0.83, 0.83)	0.830	0.000
FL(Decentralized)	(0.71, 0.99)	0.850	0.280
ML	(0.59, 0.99)	0.790	0.400

5- Conclusion and Research Contribution

This research demonstrates that decentralized federated learning effectively enhances credit risk prediction while preserving data privacy. Centralized FL achieved 83% accuracy with high stability, while decentralized FL showed competitive performance (71%-99%, mean 85%) compared to traditional ML (59%-99%). This study successfully implemented the first decentralized FL framework for credit risk in Iran, with the third iterative approach proving most effective. Socket-based implementation showed practical feasibility with only 30 seconds of communication overhead. Results indicate financial institutions can collaborate to improve model accuracy without sharing sensitive data, enabling better risk management while maintaining compliance and customer trust. This approach provides a viable solution, balancing performance with privacy for enhanced credit risk prediction systems.

My contribution to this research involves the step-by-step implementation and refinement of a decentralized federated learning architecture, making it a practical and accessible tool for various applications. In contrast to many previous studies that often lacked comparative analyses or relied on limited machine learning methods and datasets, this research bridges the gap by providing a more comprehensive and practical approach.This research opens up several significant directions for future work.

In the current implementation, a uniform model architecture was used for all clients in the decentralized federated learning setup. While this approach ensures consistency and simplifies model management, it does not account for the diverse characteristics of each client's data. Future research can explore model diversification, where each client employs a tailored model adapted to its own data distribution and complexity. For instance, clients with significantly different risk profiles, data volumes, or feature distributions may benefit from locally optimized models. Comparing these personalized models to a shared global model may provide valuable insights into the trade-offs between consistency and performance.

Another important direction is communication optimization. The current socket-based architecture introduces noticeable latency and communication overhead, especially as the number of clients grows. To address this, future work could incorporate techniques such as adaptive model compression, quantization, or sparse updates to minimize transmitted data volume. Additionally, exploring asynchronous update strategies and adopting more efficient protocols (e.g., gRPC, MQTT) could further reduce communication delay and improve scalability.

Lastly, deploying the framework in real-world financial environments would offer opportunities to evaluate operational constraints, legal compliance (e.g., with privacy laws), and performance in production settings. These extensions would help mature the decentralized FL approach and accelerate its adoption in sensitive, data-restricted domains such as banking and credit risk management.

References

[1] A. Jangir, "German Credit Card Data" [Data set], Kaggle. Available:https://www.kaggle.com/datasets/arunjangir245/german-credit-card. Accessed: Dec. 6, 2024.

[2] A. Oualid, Y. Maleh, and L. Moumoun, "FEDERATED LEARNING TECHNIQUES APPLIED TO CREDIT RISK MANAGEMENT: A SYSTEMATIC LITERATURE REVIEW," EDPACS, vol. 68, no. 1, pp. 42–56, Jul. 2023, doi: 10.1080/07366981.2023.2241647.

[3] D. Gao, C. Ju, X. Wei, Y. Liu, T. Chen, and Q. Yang, "HHHFL: Hierarchical Heterogeneous Horizontal Federated Learning for Electroencephalography," arXiv.org, Sep. 11, 2019. http://arxiv.org/abs/1909.05784

[4] F. Mozaffari, I. Raeesi Vanani, P. Mahmoudian, and B. Sohrabi, "Application of Machine Learning in the Telecommunications Industry: Partial Churn Prediction by using a Hybrid Feature Selection Approach," Journal of Information Systems and Telecommunication (JIST), vol. 11, no. 44, pp. 331–346, Dec. 2023, doi: https://doi.org/10.61186/jist.38419.11.44.331.

[5] J. Ding, E. Tramel, A. K. Sahu, S. Wu, S. Avestimehr, and T. Zhang, "Federated Learning Challenges and Opportunities: An outlook," arXiv.org, Feb. 01, 2022. http://arxiv.org/abs/2202.00807

[6] J. Zhou et al., "A Survey on Federated Learning and its Applications for Accelerating Industrial Internet of Things," arXiv.org, Apr. 21, 2021. http://arxiv.org/abs/2104.10501.

[7] L. Li, Y. Fan, M. Tse, and K.-Y. Lin, "A review of applications in federated learning," Computers & Industrial Engineering, vol. 149, p. 106854, Sep. 2020, doi: 10.1016/j.cie.2020.106854.

[8] LaoTse, "Credit Risk Dataset" [Data set], Kaggle. Available: https://www.kaggle.com/datasets/laotse/credit-risk-dataset. Accessed: December 6, 2024.

[9] M. Goutier, C. Diebel, M. Adam, and A. Benlian, "Federated Learning for credit risk assessment," Proceedings of the ... Annual Hawaii International Conference on System Sciences/Proceedings of the Annual Hawaii International Conference on System Sciences, Jan. 2024, doi: 10.24251/hicss.2023.048.

[10] M. Loukili, F. Messaoudi, and R. El Youbi, "Implementation of Machine Learning Algorithms for Customer Churn Prediction," Journal of Information Systems and Telecommunication (JIST), vol. 11, no. 43, pp. 196–208, Aug. 2023, doi: https://doi.org/10.61186/jist.34208.11.43.196.

[11] M. Rasouli, "Implementation and comparison of machine learning methods in the credit risk assessment of financial institution customers," 8th International Conference on Industrial Engineering and Systems, 2022.

[12] R. Mehta, "Credit Risk Analysis" [Data set], Kaggle. Available: https://www.kaggle.com/datasets/rameshmehta/credit-risk-analysis?resource=download. Accessed: Dec. 6, 2024.

[13] N. Mohammadi, A. Rezakhani, H. Haj Seyyed Javadi, and P. Asghari, "FLHB-AC: Federated Learning History-Based Access Control Using Deep Neural Networks in Healthcare System," Journal of Information Systems and Telecommunication (JIST), vol. 12, no. 46, pp. 90–104, Jun. 2024, doi: https://doi.org/10.61186/jist.44500.12.46.90.

[14] P. Rouintan, "Factors affecting credit risk: A case study of bank customers; Keshavarzi Bank," 2006.

[15] P. Sharifi, V. Jain, M. A. Poshtkohi, E. Seyyedi, and V. Aghapour, "Banks Credit Risk Prediction with Optimized ANN Based on Improved Owl Search Algorithm," Mathematical Problems in Engineering, vol. 2021, pp. 1–10, Dec. 2021, doi: 10.1155/2021/8458501.

[16] Parisrohan, "Credit Score Classification" [Data set], Kaggle. Available: https://www.kaggle.com/datasets/parisrohan/credit-score-classification?resource=download. Accessed: December 6, 2024.

[17] M. Rosuli, "Implementation and comparison of machine learning methods in assessing the credit risk of customers in financial and credit institutions," 8th International Conference on Industrial Engineering and Systems, 2022.

[18] S. Bharati, M. R. H. Mondal, P. Podder, and V. B. S. Prasath, "Federated learning: Applications, challenges and future directions," International Journal of Hybrid Intelligent Systems, vol. 18, no. 1–2, pp. 19–35, Apr. 2022, doi: 10.3233/his-220006.

[19] S. Jain, "Loan Prediction Based on Customer Behavior" [Data set], Kaggle. Available: https://www.kaggle.com/datasets/subhamjain/loanprediction-based-on-customer-behavior?select=Training+Data.csv. Accessed: Dec. 6, 2024.

[20] TensorFlow, "TensorFlow: An end-to-end open-source machine learning platform." Available: https://www.tensorflow.org/. Accessed: December 6, 2024.

[21] Wst, "Neural network credit scoring models," Computers & Operations Research, vol. 27, no. 11–12, pp. 1131–1152, 2000.

[22] Y. Li, "Credit risk prediction based on machine learning methods," in Proc. 14th Int. Conf. Comput. Sci. Educ. (ICCSE), Aug. 2019, pp. 1011–1013, doi: 10.1109/ICCSE.2019.8845525.

[23] Y. Shastri, "A step-by-step guide to federated learning in computer vision," V7labs.com, V7, Apr. 21, 2023. Available: https://www.v7labs.com/blog/federated-learning-guide. Accessed: June 30, 2023.

[24] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated Learning with Non-IID Data," arXiv (Cornell University), Jan. 2018, doi: 10.48550/arxiv.1806.00582.

[25] Z. Iqbal and H. Y. Chan, "Concepts, key challenges and open problems of federated learning," Int. J. Eng. (IJE), doi: 10.5829/ije.20..a.11.

[26] Z. Wang, J. Xiao, L. Wang, and J. Yao, "A novel federated learning approach with knowledge transfer for credit scoring," Decision Support Systems, vol. 177, p. 114084, Sep. 2023, doi: 10.1016/j.dss.2023.114084.

[27] Z. Xu, J. Cheng, L. Cheng, X. Xu, and M. Bilal, "MSES credit Risk Assessment model based on federated learning and feature selection," Computers, Materials & Continua/Computers, Materials & Continua (Print), vol. 75, no. 3, pp. 5573–5595, Jan. 2023, doi: 10.32604/cmc.2023.037287.

[28] H. Zhang, J. Bosch, and H. H. Olsson, "Federated Learning Systems: Architecture Alternatives," in Proc. 27th Asia-Pacific Software Engineering Conf. (APSEC), 2020, pp. 385–394, doi: 10.1109/APSEC51365.2020.00047.

[29] Y. Zhang, H. Xie, B. Bai, W. Yu, L. Li, and Y. Gao, "A survey on federated learning," Knowledge-Based Systems, vol. (Volume not provided).