Hierarchical Federated Deep Learning System

Advanced Diabetes Prediction with Privacy-Preserving Machine Learning

Project Type	Hierarchical Federated Learning Platform
Primary Application	Diabetes Risk Prediction
Architecture	3-Tier Federation (Patient → Fog → Global)
Security Features	Differential Privacy + Committee Validation
Interface	Streamlit Web Application
Language Support	English and French
Documentation Date	June 12, 2025

Table of Contents

1. System Requirements and Dependencies

2. Installation Guide

3. Methodology and Architecture

4. Communication Protocols

5. Program Description and Components

6. Step-by-Step Usage Guide

7. User Interface Tabs Explanation

8. Advanced Features and Analytics

9. Security and Privacy Features

10. Troubleshooting and Best Practices

1. System Requirements and Dependencies

1.1 Hardware Requirements

- Minimum 4GB RAM (8GB recommended for optimal performance)
- Multi-core processor (4+ cores recommended)
- 2GB available disk space
- Stable internet connection for federated communication
- Browser with JavaScript support for web interface

1.2 Software Dependencies

Package	**Version**	**Purpose**
streamlit	Latest	Web interface framework
numpy	≥1.21.0	Numerical computations
pandas	≥1.3.0	Data manipulation
scikit-learn	≥1.0.0	Machine learning algorithms
plotly	≥5.0.0	Interactive visualizations
matplotlib	≥3.4.0	Static plotting
seaborn	≥0.11.0	Statistical visualizations
reportlab	≥3.6.0	PDF generation
networkx	≥2.6.0	Network graph analysis
trafilatura	Latest	Web content extraction

2. Installation Guide

2.1 Environment Setup

Step 1: Clone or download the project repository

Step 2: Ensure Python 3.8+ is installed on your system

Step 3: Install required dependencies using the package manager

Step 4: Verify all dependencies are correctly installed

Step 5: Configure the Streamlit server settings

2.2 Configuration Files

The system requires specific configuration for optimal performance: Streamlit Configuration (.streamlit/config.toml): [server] headless = true address = "0.0.0.0" port = 5000 This configuration ensures the application runs properly in both development and deployment environments.

2.3 Running the Application

```
Command: streamlit run app.py --server.port 5000
```

3. Methodology and Architecture

3.1 Hierarchical Federation Overview

The system implements a 3-tier hierarchical federated learning architecture: TIER 1 - Medical Facilities (Edge Nodes): • Local model training on patient data • Privacy-preserving data processing • Local differential privacy implementation • Committee-based validation participation TIER 2 - Fog Nodes (Regional Aggregators): • Regional model aggregation • Intermediate privacy protection • Load balancing and coordination • Regional performance optimization TIER 3 - Global Server (Central Coordinator): • Global model orchestration • Final model aggregation using FedProx algorithm • System-wide performance monitoring • Security protocol enforcement

3.2 Machine Learning Methodology

The system employs multiple machine learning approaches: Primary Algorithm: Logistic Regression • Interpretable predictions for medical applications • Efficient training in federated environments • Robust performance with limited data Alternative Algorithms: • Random Forest: For complex feature interactions • Neural Networks: For deep learning capabilities • Support Vector Machines: For high-dimensional data Feature Engineering: • Standardization and normalization • Missing value imputation • Feature selection and importance analysis

4. Communication Protocols

4.1 Federation Communication Flow

Phase	Source	Destination	Data Transmitted
Initialization	Global Server	All Clients	Initial model parameters
Local Training	Medical Facilities	Local Storage	Training progress
Model Upload	Medical Facilities	Fog Nodes	Encrypted model updates
Regional Aggregation	Fog Nodes	Global Server	Aggregated updates
Global Update	Global Server	All Clients	Updated global model
Validation	Committee Members	All Participants	Validation results

4.2 Security Protocols

The system implements multiple layers of security: Differential Privacy: • Gaussian and Laplace noise mechanisms • Privacy budget management (ε-δ privacy) • Adaptive noise scaling based on sensitivity Committee-Based Validation: • Multi-party consensus for model updates • Anomaly detection for malicious participants • Reputation scoring system Secret Sharing: • Polynomial-based secret sharing scheme • Distributed weight reconstruction • Protection against single points of failure Communication Security: • Encrypted parameter transmission • Secure aggregation protocols • Authentication and authorization

5. Program Description and Components

5.1 Core Components

Component	File	Primary Function
Main Application	app.py	Streamlit interface and coordination
Federated Learning Manager	federated_learning.py	FL orchestration and training
Client Simulator	client_simulator.py	Medical facility simulation
Data Preprocessing	data_preprocessing.py	Data cleaning and preparation
Aggregation Algorithms	aggregation_algorithms.py	FedAvg, FedProx implementation
Differential Privacy	differential_privacy.py	Privacy protection mechanisms
Advanced Analytics	advanced_client_analytics.py	Performance monitoring
Data Distribution	data_distribution.py	IID/Non-IID data handling
Secret Sharing	training_secret_sharing.py	Cryptographic protocols

5.2 Data Flow Architecture

The system processes data through multiple stages: 1. Data Ingestion: • Diabetes dataset loading and validation • Missing value detection and handling • Feature standardization and encoding 2. Data Distribution: • Client data partitioning (IID/Non-IID) • Privacy-preserving data allocation • Quality assessment and validation 3. Local Training: • Individual client model training • Local performance evaluation • Privacy noise injection 4. Aggregation: • Secure parameter collection • Weighted model averaging • Global model reconstruction 5. Evaluation: • Performance metric calculation • Convergence analysis • Security validation

6. Step-by-Step Usage Guide

6.1 Initial Setup

1. Launch the application using the streamlit run command
2. Access the web interface through your browser
3. Select your preferred language (English/French)
4. Review the system overview and architecture
5. Navigate to the Training Configuration tab

6.2 Training Configuration

1. Set the number of medical facilities (clients): 3-10 recommended
2. Configure maximum training rounds: 20-50 for optimal results
3. Set target accuracy threshold: 0.85 for high performance
4. Choose aggregation algorithm: FedProx for robustness
5. Enable differential privacy with appropriate epsilon values
6. Configure committee size for security validation
7. Select machine learning model type
8. Set early stopping parameters for efficiency

6.3 Training Execution

1. Click 'Start Federated Training' to begin the process
2. Monitor real-time progress through the progress bars
3. Observe accuracy and loss metrics during training
4. Watch for convergence indicators and early stopping
5. Review final training results and model performance

7. User Interface Tabs Explanation

7.1 ■ Training Configuration

Purpose: Configure federated learning parameters and start training

Key Features:

- Number of medical facilities configuration
- Training rounds and target accuracy settings
- Aggregation algorithm selection
- Differential privacy parameters
- Committee security settings
- Model type selection
- Early stopping configuration

7.2 ■ Medical Station Monitoring

Purpose: Real-time monitoring of training progress and facility performance

Key Features:

- Live training progress visualization
- Individual facility performance metrics
- Real-time accuracy and loss tracking
- Training round completion status
- Performance comparison across facilities

7.3 ■ Interactive Journey Visualization

Purpose: Visual representation of the federated learning process

Key Features:

- Network topology visualization
- Data flow diagrams
- Hierarchical architecture display
- Interactive facility selection
- Communication pattern analysis

7.4 ■ Performance Analysis

Purpose: Comprehensive analysis of training results and model performance

Key Features:

- Accuracy and loss progression charts
- Training metrics summary
- Performance improvement tracking
- Convergence analysis
- Final model evaluation

7.5 ■ Patient Risk Prediction Explainer

Purpose: Individual patient diabetes risk assessment and prediction

Key Features:

- Patient information input form
- Real-time risk prediction
- Feature importance analysis
- Clinical interpretation
- Risk factor explanations

7.6 ■ Advanced Medical Analytics

Purpose: Deep dive into medical facility performance and correlation analysis

Key Features:

- Correlation matrix analysis
- Feature relationship visualization
- Clinical insights and recommendations
- Medical facility performance dashboard
- Advanced statistical analysis

7.7 ■ Network Visualization

Purpose: Interactive network topology and communication visualization

Key Features:

- Network topology graphs
- Data flow visualization
- Hierarchical architecture display
- Performance-based node coloring
- Interactive network exploration

7.8 ■ Advanced Analytics Dashboard

Purpose: Comprehensive analytics with confusion matrices and performance comparisons

Key Features:

- Confusion matrix analysis
- Accuracy vs clients optimization
- Fog node performance analysis
- Comprehensive performance comparison
- Medical facility grading

8. Advanced Features and Analytics

8.1 Performance Optimization

The system includes several performance optimization features: Early Stopping: • Monitors training progress for convergence • Prevents overfitting and reduces training time • Automatically restores best performing model • Configurable patience and improvement thresholds Adaptive Learning: • Dynamic learning rate adjustment • Performance-based parameter tuning • Convergence detection algorithms • Resource usage optimization Model Selection: • Multiple algorithm support • Automatic best model selection • Cross-validation integration • Performance comparison tools

8.2 Analytics and Visualization

Comprehensive analytics capabilities include: Real-time Monitoring: • Live training progress tracking • Performance metric visualization • Resource utilization monitoring • Error detection and reporting Post-training Analysis: • Confusion matrix analysis • ROC curve generation • Feature importance ranking • Model interpretability tools Comparative Analysis: • Multi-model performance comparison • Client performance benchmarking • Aggregation algorithm evaluation • Privacy-utility trade-off analysis

9. Security and Privacy Features

9.1 Differential Privacy Implementation

The system implements state-of-the-art differential privacy: Noise Mechanisms: • Gaussian mechanism for numerical data • Laplace mechanism for counting queries • Exponential mechanism for categorical data • Adaptive noise scaling based on sensitivity Privacy Budget Management: • ε-δ privacy guarantees • Composition theorem application • Budget allocation optimization • Privacy accounting across rounds Advanced Features: • Local differential privacy options • Privacy-utility optimization • Moment accountant for tight bounds • Personalized privacy levels

9.2 Committee-Based Security

Multi-party validation ensures system integrity: Validation Process: • Random committee selection • Consensus-based model validation • Anomaly detection algorithms • Malicious participant identification Reputation System: • Historical performance tracking • Trust score computation • Weighted voting mechanisms • Adaptive committee size Security Measures: • Byzantine fault tolerance • Sybil attack prevention • Model poisoning detection • Gradient leakage protection

10. Troubleshooting and Best Practices

10.1 Common Issues and Solutions

Issue	Possible Cause	Solution
Training fails to start	Missing dependencies	Install all required packages
Low accuracy results	Insufficient training data	Increase client data or rounds
Slow convergence	High privacy noise	Adjust epsilon parameters
Memory errors	Large dataset/many clients	Reduce batch size or clients
Connection timeouts	Network instability	Check internet connection
Analytics not showing	Training not completed	Complete training first

10.2 Best Practices

Recommendations for optimal system performance: Configuration: • Start with 5-10 medical facilities for balanced training • Use 20-50 training rounds for convergence • Set epsilon between 0.1-2.0 for privacy-utility balance • Enable early stopping with patience of 5-10 rounds Data Management: • Ensure balanced data distribution across clients • Validate data quality before training • Monitor for missing or corrupted data • Use appropriate preprocessing techniques Performance: • Monitor system resources during training • Use appropriate hardware for large-scale experiments • Consider distributed computing for very large deployments • Regular backup of training results and models Security: • Regularly update privacy parameters • Monitor for unusual participant behavior • Validate committee consensus results • Keep audit logs of all training activities

---

Documentation generated on June 12, 2025 at 09:25