

# Sequential Deep Learning for Credit Risk Modeling in Data-Constrained Environments

---

A research project investigating temporal feature engineering and sequential deep learning for credit risk modeling using mobile money (MoMo) transaction data from Ghana.

## Project Overview

This project develops and evaluates temporal feature engineering frameworks and sequential deep learning models for credit risk prediction in data-constrained environments. Key contributions include:

- **Temporal Feature Engineering Framework:** Extracts 113 features from transaction sequences across 8 categories
- **Calibrated Synthetic Data Generator:** Generates realistic mobile money data calibrated to real Ghana transaction patterns
- **Model Comparison:** Compares sequential models (LSTM) vs. static models (Logistic Regression, XGBoost)
- **Real Data Analysis:** Analysis of 482 real mobile money transactions over 200 days

## Research Focus

- **Paper A:** Feature Engineering Framework for temporal transaction data
- **Paper B:** Static vs. Sequential modeling comparison for fraud detection

## Getting Started

### Prerequisites

- Python 3.8 or higher
- Jupyter Notebook support (or [Google Colab](#))

### Setup

#### 1. Clone the repository:

```
git clone https://github.com/Attabeezy/sequential-crm-for-dce.git
cd sequential-crm-for-dce
```

#### 2. Install required packages:

```
pip install pandas numpy scikit-learn tensorflow xgboost matplotlib
seaborn ctgan
```

# Usage

## Python Modules

The project includes reusable Python modules for feature engineering and data generation.

### Feature Engineering

```
from src.feature_engineering.real_temporal_feature_engineering import
TemporalTransactionFeatureEngineer

engineer = TemporalTransactionFeatureEngineer()

# Extract all 113 temporal features
df_features = engineer.extract_all_features(df, windows=[3, 7, 14, 30])

# Create user-level summary
user_summary = engineer.create_user_level_summary(df)
```






### Synthetic Data Generation

```
from src.data_generation.calibrated_synthetic_generator import
CalibratedMoMoDataGenerator

generator = CalibratedMoMoDataGenerator(
    n_users=2000,
    avg_transactions_per_user=15,
    fraud_rate=0.05,
    start_date='2024-01-01',
    duration_days=180
)

transactions_df, user_profiles_df = generator.generate_dataset()
```

## Jupyter Notebooks

Notebook	Description	Colab
<b>credit_risk_prediction_v1c.ipynb</b>	Latest credit risk prediction workflow	 <a href="#">Open in Colab</a>
<b>credit_risk_prediction_v1b.ipynb</b>	Enhanced version with additional metrics	 <a href="#">Open in Colab</a>
<b>credit_risk_prediction_v1a.ipynb</b>	Initial credit risk prediction workflow	 <a href="#">Open in Colab</a>
<b>syn_data_gen.ipynb</b>	Basic synthetic MoMo data generator	 <a href="#">Open in Colab</a>
<b>ctgan_syn_data_gen.ipynb</b>	CTGAN-based synthetic data generation	 <a href="#">Open in Colab</a>

## Running Notebooks

**Google Colab (Recommended):** Click the "Open in Colab" badges above.

### Local Jupyter:

```
pip install jupyter
jupyter notebook
# Navigate to notebooks/ folder
```

## Model Training Workflow

The credit risk prediction notebooks:

1. Load and preprocess transaction data
2. Apply temporal feature engineering (113 features)
3. Train and compare models: ANN, Logistic Regression, XGBoost
4. Evaluate with metrics: Accuracy, MSE, Macro-F1, ROC-AUC, Sensitivity/Precision
5. Generate ROC curves and classification reports

## Repository Structure

```
sequential-crm-for-dce/
├── data/
│   ├── real/                                # Real transaction data from Ghana
│   │   ├── transactions.xlsx - Table 1.csv
│   │   ├── transactions.xlsx - Table 5.csv # 482 transactions
│   │   ├── engineered_features_real_data.csv
│   │   └── user_level_summary.csv
│   └── synthetic/                          # Calibrated synthetic datasets
│       ├── synthetic_momo_calibrated.csv    # 29,994 transactions
│       ├── synthetic_user_profiles.csv      # 2,000 user profiles
│       ├── synthetic-momo-data.csv
│       └── real_data_calibration.json
├── docs/
│   ├── COMPLETE_ANALYSIS.md                # Feature engineering documentation
│   └── SYNTHETIC_DATA_GUIDE.md              # Synthetic dataset guide
├── notebooks/
│   ├── credit_risk_prediction_v1a.ipynb
│   ├── credit_risk_prediction_v1b.ipynb
│   ├── credit_risk_prediction_v1c.ipynb
│   ├── ctgan_syn_data_gen.ipynb
│   └── syn_data_gen.ipynb
├── src/
│   ├── data_generation/
│   │   ├── __init__.py
│   │   └── calibrated_synthetic_generator.py
│   └── feature_engineering/
│       ├── __init__.py
```

```

├── real_temporal_feature_engineering.py
├── .gitignore
├── LICENSE # MIT License
├── README.md
└── SESSION_LOG.md # Research session documentation

```

## Feature Engineering Framework

The `TemporalTransactionFeatureEngineer` class extracts 113 features across 8 categories:

Category	Features	Examples
Transaction-level static	10	log_amount, sqrt_amount, is_micro_txn, is_large_txn
Categorical encodings	8	is_transfer, is_debit, is_payment, is_cash_out
Temporal extraction	17	hour, day_of_week, is_weekend, hour_sin, hour_cos
Balance dynamics	23	balance_change, balance_pct_change, amount_to_balance_ratio
Sequence features	31	last_N_avg_amount, cumulative_volume, amount_vs_last_N_avg
Rolling windows	28	rolling_Nd_count, rolling_Nd_mean (3/7/14/30-day windows)
Behavioral patterns	4	unique_recipients, repeated_recipient, self_transfer
Risk indicators	8	unusual_hour, rapid_transaction, rapid_balance_drop

## Data

### Real Data

- **Source:** Ghana mobile money account (Feb-Sep 2024)
- **Transactions:** 482 over 200 days
- **Frequency:** 2.41 transactions/day
- **Volume:** GHS 15,810.40 total

### Synthetic Data

- **Users:** 2,000 (1,900 legitimate, 100 fraudulent)
- **Transactions:** 29,994
- **Fraud types:** account\_takeover, social\_engineering, sim\_swap
- **Calibrated to:** Real Ghana MoMo transaction patterns

## Dependencies

- **pandas** - Data manipulation
- **numpy** - Numerical operations
- **scikit-learn** - ML baselines and preprocessing
- **tensorflow/keras** - Neural networks (ANN, LSTM)
- **xgboost** - Gradient boosting

- **matplotlib/seaborn** - Visualization
- **ctgan** - Conditional Tabular GAN for synthetic data

## Contributing

Contributions are welcome! Please feel free to submit a Pull Request.

## License

This project is licensed under the MIT License - see the [LICENSE](#) file for details.