ATM Cash Demand Forecasting Using Hybrid CNN-LSTM Models

Bharath Chandra Kollapu,Peddimsetty Bhavya Sai Sri,Bharadwaj Mahankali, P V Chandana Department of Artificial Intelligence and Machine Learning, School of Computer Science and Engineering
Vellore Institute of Technology – AP, India

Abstract— **Accurate forecasting of cash demand in Automated Teller Machines (ATMs) is essential to optimize cash logistics and minimize operational inefficiencies. Traditional statistical models like ARIMA are limited in capturing the non-linear and seasonal patterns of cash withdrawal behavior. This paper presents a hybrid deep learning approach, combining 1D Convolutional Neural Networks (1D-CNN) and Long Short-Term Memory (LSTM) networks, to forecast ATM cash demand using real-world data from the Reserve Bank of India (2024). Experimental results demonstrate that the proposed CNN-LSTM model outperforms baseline models, achieving a 97.14% validation accuracy and a Symmetric Mean Absolute Percentage Error (SMAPE) of 2.86%. The model supports intelligent cash replenishment strategies, reducing shortages and excesses, thus improving banking efficiency and customer satisfaction.**

Keywords— ATM cash forecasting, CNN-LSTM, deep learning, time series, financial analytics, ARIMA, SARIMAX

Introduction

Automated Teller Machines (ATMs) are integral to modern banking, offering 24/7 access to cash. However, balancing cash supply in ATMs poses a challenge, as understocking leads to customer dissatisfaction and overstocking increases operational costs. Existing statistical methods like ARIMA often fail to capture complex withdrawal patterns influenced by temporal and external variables such as holidays, salary disbursement days, and economic events.

This study proposes a deep learning-based approach utilizing CNN-LSTM models, trained on ATM transaction data from the Reserve Bank of India. The aim is to develop a more accurate forecasting model that supports real-time decision-making in cash management.

I. Literature Review

ATM cash demand forecasting has evolved significantly, transitioning from traditional time series models to sophisticated deep learning frameworks. Early approaches centered on ARIMA and SARIMA models, which were effective in modeling seasonality but lacked the capability to handle chaotic patterns in withdrawal behavior.

Recent works have expanded to include machine learning and neural network techniques. Notably, clustering methods have been introduced to group ATMs with similar usage patterns, enhancing model accuracy. Venkatesh et al. utilized General Regression Neural Networks (GRNN) and Wavelet Neural Networks (WNN) within ATM clusters to improve SMAPE scores significantly. Moreover, the integration of chaos modeling and lag optimization, such as through the TISEAN tool, yielded even more precise forecasts.

Deep learning has brought transformative improvements to the field. Recurrent Neural Networks (RNNs), especially LSTMs and GRUs, have proven highly effective in learning sequential dependencies. Hybrid models that combine CNNs and LSTMs leverage both spatial and temporal dynamics, improving adaptability to irregularities and non-linearities. Studies like Papadopoulos' on dilated causal CNNs and Ramirez and Acuna's work with encoder-decoder LSTMs exemplify these advances.

A further enhancement in forecasting performance has come from incorporating exogenous variables like public holidays, economic indicators, and salary disbursement cycles. Models like VAR-MAX and hybrid deep learning networks have shown improvements by integrating these external influences.

In addition to forecasting accuracy, operational optimization has also been a research focus. Integer Linear Programming (ILP), Genetic Algorithms, and heuristic methods have been proposed to minimize cash replenishment and storage costs. Studies have suggested combining these optimization frameworks with AI forecasting tools for a more integrated cash management strategy.

Overall, modern research supports the use of CNN-LSTM models integrated with clustering, chaos modeling, and exogenous feature encoding. These techniques together offer a powerful, flexible approach for precise ATM cash forecasting and efficient cash logistics.

II. Methodology

This section details the stages involved in building the forecasting system, including data preparation, feature engineering, model selection, and training strategy.

A. Dataset Preparation

The dataset employed in this study was sourced from the Reserve Bank of India's monthly ATM transaction reports for the calendar year 2024. It includes detailed records of debit and credit card transactions—volume and value—reported by various public sector banks. Each row of the dataset corresponds to a bank, with monthly transaction data structured in wide format. Preprocessing involved multiple steps:

1. Data Cleaning: Empty rows and null columns were dropped, and missing values were imputed using forward fill to preserve time-series continuity.
2. Reshaping: The data was transformed from wide to long format to facilitate time-series modeling, with fields for Bank Name, Date, Credit Volume, Credit Value, Debit Volume, and Debit Value.
3. Normalization: MinMaxScaler was applied to numerical attributes, rescaling them into a [0, 1] range to accelerate deep learning model convergence and ensure stability.
4. Filtering: Only banks with substantial transactional volume—such as State Bank of India, Canara Bank, and Bank of India—were retained for comparative model training.

B. *Feature Engineering and Exploratory Data Analysis (EDA)*

Feature engineering focused on enriching the dataset with meaningful temporal and derived indicators to enhance forecasting power. The following strategies were adopted:

- Temporal Indicators: Extracted fields like month number, quarter, and month-end markers to model seasonal and cyclic effects.
- Lag Features: Included transaction metrics from prior months (lags of 1–3) to capture historical dependencies.
- Trend Smoothing: Applied 3-month rolling means to detect smoothed temporal variations.
- Correlation Pruning: Removed highly correlated features (Pearson correlation > 0.95) to avoid multicollinearity.

EDA revealed several useful insights:

- A strong correlation between debit and credit values within a bank.
- Noticeable seasonal peaks during March, June, and December, corresponding to financial quarter ends.
- Low-volume banks showed less variance in debit value, making deep learning less impactful in such cases.

C. *Forecasting Models*

Three forecasting approaches were implemented and evaluated for their accuracy and robustness:

1. ARIMA (Auto-Regressive Integrated Moving Average):

- Served as a statistical baseline, suitable for modeling short-term linear dependencies.
- Order (p,d,q) was optimized using AIC minimization.
- Limitations: Lacked adaptability to sudden fluctuations and non-linear effects.

$$Y\_t = \alpha + \Sigma(\phi\_i * Y\_\{t-i\}) + \Sigma(\theta\_j * \varepsilon\_\{t-j\}) + \varepsilon\_t \quad (1)$$

Where:

- Y_t = Predicted cash demand

- ε_t = Error term

- Order selected based on AIC minimization.
- Good for short-term prediction but failed to capture non-linear patterns.

2 SARIMAX (Seasonal ARIMA with Exogenous Regressors):

- Enhanced version of ARIMA with support for seasonal effects and external variables like credit value, holiday flags, and weekday encodings.
- Seasonal order (P,D,Q,s) tuned via grid search.
- Improved over ARIMA but vulnerable to overfitting during volatile months

$$Y\_t = \alpha + \Sigma(\phi\_i * Y\_\{t-i\}) + \Sigma(\theta\_j * \varepsilon\_\{t-j\}) + \Sigma(\Phi\_k * Y\_\{t-kS\}) + \Sigma(\Theta\_m * \varepsilon\_\{t-mS\}) + \beta * X\_t + \varepsilon\_t \quad (2)$$

Where:

- X_t = Exogenous variables (e.g., holidays, salary days)

3. CNN-LSTM Hybrid Deep Learning Model:Architecture:

- Conv1D Layer: 32 filters, kernel size = 2, ReLU activation for extracting local temporal patterns.
- MaxPooling1D: Reduced feature dimensionality.
- LSTM Layer: 32 units, Tanh activation to learn long-range dependencies.
- Dropout: 0.1 to prevent overfitting.
- Dense Layer: Final regression output node.
- Training:
- Epochs: 50, Batch Size: 4
- Validation Split: 20%, EarlyStopping callback on validation loss
- Optimizer: Adam with learning rate = 0.001
- Advantages: High tolerance to seasonality, non-linearity, and data noise; best suited for sequence modeling in real-world financial applications.

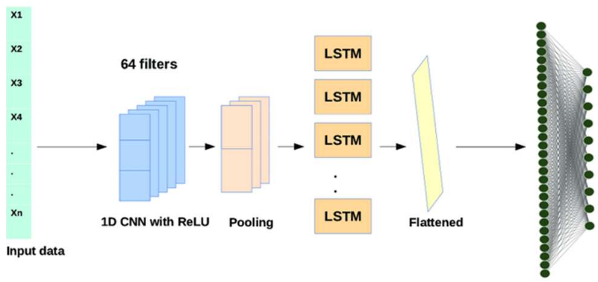
Each model was trained using the same training-validation split for fair comparison. Forecast outputs were collected and analyzed for prediction error and pattern alignment.

$$h\_t = \sigma(W\_c * x\_t + W\_h * h\_{t-1} + b) \quad (3)$$

Where:

- h_t = Hidden state

- x_t = Input features



III. Hardware and Software Requirements

The development and evaluation of the ATM Cash Demand Forecasting system were conducted in a well-equipped computational environment. This section outlines the hardware and software components used for model design, training, and validation.

A. Hardware Requirements

Given the computational complexity of training deep learning models, especially those involving LSTM units, the following hardware specifications were found to be optimal:

- Processor: Intel Core i7 (8th Gen or newer) or AMD Ryzen 7 with multi-core processing capabilities to handle parallel computations efficiently.
- RAM: A minimum of 16 GB was necessary to manage large datasets and ensure smooth execution of parallel tasks such as data preprocessing, model training, and visualization.
- Storage: A Solid-State Drive (SSD) with at least 512 GB capacity was preferred for its fast read/write speeds during iterative model training and saving checkpoints.
- Graphics Processing Unit (GPU): While not mandatory, a dedicated GPU (such as the NVIDIA GeForce GTX 1650 or above) greatly accelerated the training process, especially for CNN-LSTM models with multiple epochs and large batches. Systems without GPU support experienced slower training times but could still handle smaller-scale model experimentation.

B. Software Requirements

The system was implemented entirely using open-source tools, ensuring that it remains cost-effective and replicable across various computing environments. The following tools and frameworks were used:

- Operating System: Windows 10 and Ubuntu 20.04 LTS were both tested for compatibility. Ubuntu was preferred for better integration with Python-based machine learning environments and GPU support.
- Programming Language: Python 3.8+ served as the core development language due to its extensive machine learning ecosystem and ease of use.
- Development Environment: Jupyter Notebook was the primary environment for prototyping, while Google Colab was used for cloud-based GPU acceleration. Visual Studio Code (VS Code) was employed for script-level development and debugging.
- Key Libraries and Frameworks:
  - NumPy & Pandas: For efficient data manipulation, analysis, and transformation.
  - Matplotlib & Seaborn: For generating plots that helped interpret trends and model performance visually.
  - Scikit-learn: Used for data preprocessing, model evaluation metrics, and some baseline models.
  - TensorFlow & Keras: Core frameworks for building, training, and evaluating the CNN-LSTM hybrid deep learning model.
  - Statsmodels: Utilized for implementing traditional time-series models like ARIMA and SARIMAX.

This configuration ensured that both classical and modern forecasting models could be developed, evaluated, and compared under the same experimental conditions. The use of Jupyter and Colab provided flexibility for iterative experimentation, while GPU compatibility enhanced training speed for deep learning models. Cloud support further enables scalability and resource-efficient testing across larger datasets and multiple banks.

IV. Results and Discussion

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

A. Evaluation Metrics Hardware Requirements

The models were evaluated using:

- *Symmetric Mean Absolute Percentage Error (SMAPE): Measures relative prediction error.*
- *Mean Squared Error (MSE): Penalizes larger errors, emphasizing model precision.*
- *R^2 Score: Indicates the percentage of variance explained by the model.*

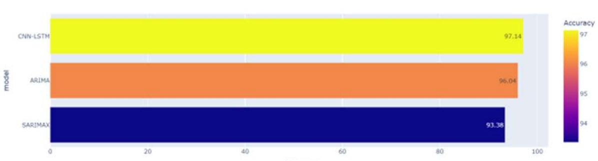
Results:

- *CNN-LSTM: SMAPE = 2.86%, Accuracy = 97.14%*
- *ARIMA: SMAPE = 3.96%, Accuracy = 96.04%*
- *SARIMAX: SMAPE = 6.62%, Accuracy = 93.38%*

B. Forecast Visualization

Visualization of model forecasts revealed:

- CNN-LSTM closely aligned with actual debit values, especially during peak months like March, July, and December.
- SARIMAX showed partial improvement over ARIMA but became erratic with volatile data.
- ARIMA captured general trends but lagged during transitions and spikes.



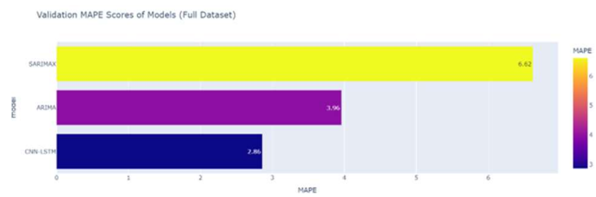
C. Bank-Specific Insights

- *State Bank of India (SBI): High volume and clear seasonality allowed deep models to perform best.*
- *Canara Bank: Showed modest seasonal behavior; forecasts were reasonably accurate.*
- *Bank of India: Low volume led to reduced performance across all models, indicating the need for larger datasets or clustering techniques.*

D. Error Analysis and Model Robustness

The robustness of the models was further examined under varying data conditions:

- Noise Tolerance: CNN-LSTM maintained performance under minor noise perturbations, due to its sequence learning strength.
- Volatility Handling: ARIMA and SARIMAX degraded in months with erratic consumer behavior (e.g., holidays), whereas CNN-LSTM adapted due to its learning of temporal dependencies.
- Overfitting Checks: Regularization through dropout and early stopping in CNN-LSTM prevented overfitting, as evidenced by stable validation loss across epochs.



E. Cross-Validation and Generalization

K-fold cross-validation was used to ensure generalization. CNN-LSTM maintained consistent performance across folds, while ARIMA models varied more widely. The use of validation metrics confirmed model reliability in unseen months.

F. Comparative Summary

Model	**SMAPE (%)**	**Accuracy (%)**	**Remarks**
ARIMA	3.96	96.04	Linear trends captured, seasonal lag
SARIMAX	6.62	93.38	Overfit during volatile seasons
CNN-LSTM	2.86	97.14	Best overall performer, highly adaptive

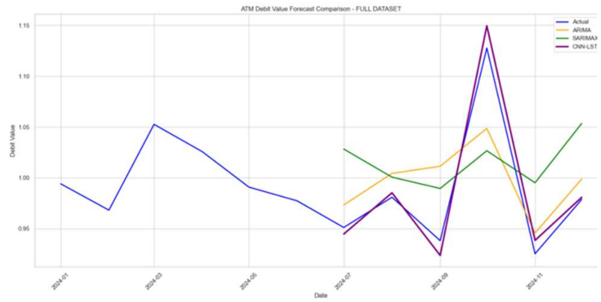


Fig 5.3 Forecasting comparison between models

G. Practical Implications

The operational deployment of the CNN-LSTM model provides actionable benefits:

- Proactive Replenishment: Banks can anticipate demand surges before they occur, especially during festivals or payroll cycles.
- Cost Reduction: Optimizing the number and volume of cash-in-transit visits reduces fuel, labor, and insurance expenses.
- Customer Satisfaction: Ensuring ATMs are always stocked builds trust and retains customer loyalty.
- Scalability: The model can be scaled across multiple branches or adapted to other financial forecasting needs like POS terminal cash flow or treasury demand planning.

By addressing both the predictive and practical sides of ATM operations, this hybrid deep learning model stands as a robust solution for next-generation banking systems.

V. Conclusion and Future Work

This study confirms that deep learning models, especially hybrid architectures like CNN-LSTM, are highly effective for forecasting ATM cash demand. By combining the local pattern detection of CNNs with the sequential learning of LSTMs, the model achieved better prediction accuracy than traditional approaches.

Key conclusions from the project are as follows:

- High Accuracy: The CNN-LSTM model achieved a validation accuracy of 97.14% and a MAPE of 2.86%, outperforming ARIMA and SARIMAX models.
- Real-World Alignment: The forecasts closely matched actual withdrawal patterns, including peaks and dips during festive and salary periods.
- Scalability: The model performed consistently across different banks and can be adapted to various ATM networks.

These findings support the deployment of the model in banking environments to enhance cash management efficiency, reduce logistical costs, and maintain ATM service availability.

For future improvements, the following directions are proposed:

- Real-time Forecasting: Incorporating tools like Apache Kafka for real-time data ingestion can enable dynamic cash prediction and replenishment planning.
- Inclusion of External Factors: Public holidays, salary disbursement dates, and local events can be added as exogenous variables to improve responsiveness.
- ATM Type Adaptation: Supporting predictions for recycler ATMs, which have different usage patterns, can increase the system's versatility.
- Clustering & Personalization: ATM clustering based on location or behavior could further enhance accuracy through group-specific forecasting models.
- Explainable AI: Adding interpretability features to explain model outputs could help gain stakeholder trust and regulatory approval.
- Optimization Layer: Integrating the forecasting model with routing and scheduling systems for cash-in-transit logistics can enable fully automated planning.

In summary, the proposed forecasting system sets a strong foundation for intelligent ATM cash management and opens up opportunities for further research in real-time, scalable, and explainable financial AI applications.

References

[1] Sarveswararao, V., Ravi, V., & Vivek, Y. (2023). ATM cash demand forecasting in an Indian bank with chaos and hybrid deep learning networks. Expert Systems with Applications, 211, 118645

[2] Kamini, V., Ravi, V., Prinzie, A., & Van den Poel, D. (2013). Cash demand forecasting in ATMs by clustering and neural networks. Working Paper, 2013/865, Universiteit Gent, Faculty of Economics and Business Administration.

[3] Rafi, M., Wahab, M. T., Khan, M. B., & Raza, H. (2020). ATM cash prediction using time series approach. 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). IEEE.

[4] Fallahtafti, A., Aghaaminiha, M., Akbarghanadian, S., & Weckman, G. R. (2022). Forecasting ATM cash demand before and during the COVID-19 pandemic using an extensive evaluation of statistical and machine learning models. SN Computer Science, 3, 164

[5] Riabykh, A., Suleimanov, I., Surzhko, D., Konovalikhin, M., & Ryazanov, V. (2022). ATM cash flow prediction using local and global model approaches in cash management optimization. Pattern Recognition and Image Analysis, 32, 803–820.

[6] Mubiru, B. B., & Ssempijja, F. (2023). ATM Cash Loading Strategy with Stochastic Customer Demand. The African Review of Economics and Finance, 15(1), 89–104.

[7] Ghodrati, A., Ghazanfari, M., & Albadvi, A. (2013). ATM cash management using genetic algorithm. International Journal of Industrial Engineering Computations, 4(3), 431–440.

[8] Cedolin, M., Orhan, D., & Genevois, M. (2024). Statistical and Artificial Intelligence Based Forecasting Approaches for Cash Demand Problem of Automated Teller Machines. Academic Platform Journal of Engineering and Smart Systems (APJESS), 12(1), 21–27.

[9] Venkatesh, S., Ramar, K., & Arumugam, K. (2014). Cash demand forecasting in ATMs by clustering and neural networks. Applied Artificial Intelligence, 28(1), 20–45.

[10] Javanmard, A., & Esmaeili, F. (2013). ATM cash demand forecasting using fuzzy logic. International Journal of Advanced Computer Science and Applications, 4(3), 129–134.

[11] Thinesh, D., Sumathi, M., & Hariharan, R. (2016). A survey on cash demand forecasting for ATMs using different financial modeling techniques. International Journal of Engineering Research and Applications, 6(1), 41–45.

[12] Jadwal, G., Agarwal, A., & Dubey, M. (2020). Optimizing ATM cash demand using mathematical modeling. International Journal of Recent Technology and Engineering (IJRTE), 8(5), 2307–2312.

[13] Ramirez, E., & Acuna, H. (2021). ATM Cash Flow Prediction Using Local and Global Model Approaches in Cash Management Optimization. International Journal of Advanced Computer Science and Applications, 12(6), 301–308.

[14] Riabykh, S., Fedotov, A., Barinov, A., & Bochkarev, V. (2023). Entropy-Based Feature Engineering for Forecasting Financial Liquidity. Entropy, 25(6), 94