Federated Blood Supply Chain Demand Forecasting: A Case Study

Anonymous Author(s)

ABSTRACT

Blood transfusion is a commonly used, life-saving medical therapeutic worldwide. A significant challenge is the high variability of supply and demand in blood products, making it difficult to maintain a balance between preventing shortages of blood products and preventing wastage. Recent studies used data-driven methods on demand forecasting for blood products from regional and centralized databases due to regulatory restrictions, which lack the panorama view of the national blood supply and demand picture. Motivated by achieving better policy-making through national blood supply chain demand forecasting, in this paper, we propose to use federated learning (FL) to forecast the demand for platelets through a case study with simulated scenarios considering a national demand and supply network. Our solution facilitates FL with a Long-Short-Term Memory (LSTM) model to make collaborative predictions for future decision-making processes from distributed and regional time-series data. Empirical studies show that FL brings additional performance improvement in various settings, especially for regions with scarcer and shorter data histories.

ACM Reference Format:

1 INTRODUCTION

Powered by blood supply chain management (BSCM), blood transfusion saves over 4.5 million lives annually in the US alone [3]. As an essential transfusion medicine, the supply and demand of blood products can be highly variable over time [19]. The supply of blood products is dominantly from donors with planned quantities, while the demand for blood products can vary due to seasonality, holidays, weekend patterns, and so forth [6]. As such, blood suppliers worldwide face significant challenges in maintaining a balance between preventing shortages and wastage of blood products, as well as reducing the overall cost of maintaining an efficient blood supply chain, including routine and urgent deliveries costs, holding costs, and wastage due to the short shelf lives of products such as red blood cells (RBC) and platelets [8]. Addressing these conflicting requirements for orders of blood products requires the blood products inventory to be carefully managed to keep the overall cost low, along with prudently-designed healthcare policies to compensate for the unavoidable costs to ensure the universal accessibility of the blood products.

Data-driven methods provide a promising way to address the aforementioned challenges by forecasting blood products based on historical usage, enabling the calibration of the short- and long-term demands at a finer granularity and leading to leaner while sufficient inventories and reduced operating costs. While several studies have been conducted on demand forecasting for blood products from regional and centralized databases [14], a national understanding of multiple inventories across jurisdictions would leverage their synergies. Therefore, collectively modeling from multiple databases greatly helps in accurate forecasting and policy-making processes, such as subsidizing the costs from a nationwide perspective. However, predictive models generated from state-of-the-art methods are primarily from a regional database covering data serving only one jurisdiction. Healthcare regulatory burdens prohibit a national model from being generated following the traditional methods because health data cannot be shared outside of the jurisdiction by law, mainly due to confidentiality concerns.

Federated learning (FL) offers a decentralized approach to model training, allowing multiple parties to collaboratively train a predictive model without sharing data, therefore keeping data confidentiality. This privacy-aware approach utilizes collective intelligence from distributed datasets and can bring additional benefits in blood supply chain demand forecasting, inventory management, and future policy making. Despite its potential, there has been no study of blood product demand forecasting with FL.
Inspired by the criticality of blood supply chain demand forecasting and the research gaps identified above, and thanks to the ARIMA models proposed in [14] based on the TRUST database in Hamilton, ON, Canada, in this paper, we explore a federated approach to forecasting the demand for platelets through a case study using simulated data based on the model parameters from [14] without the use of clinical data, for the simplified data accessibility and reproducibility of this study. We identify patterns and trends by considering key factors that affect demands, such as seasonality parameters. We use a Long-Short-Term Memory (LSTM) model that handles sequential information and learns long-term relationships [1], with FL to generate reliable predictions to guide future decision-making processes. Our proposed privacy-aware and collaborative FL approach, shown in Fig. 1, provides the enabling capability for national blood suppliers to manage supply-demand dynamics effectively among local health institutions. To our best knowledge, this is the first study that adopts FL for demand forecasting of blood products. Our solution can potentially transform the regulatory landscape of blood chain supply data and greatly benefit the public with such demand forecasting capability. We summarize the following contributions to this study:

- We propose to adopt FL to the unique scenarios of blood supply chain settings where jurisdictional data holders keep the data locally. In a national demand and supply network, data remains at the regional distribution center level across different provinces. A global model is aggregated nationally based on the local models learned from regional distribution centers nationwide. We believe the shared global model will foster a national view of the demand forecast, promoting an informed decision-making process for stakeholders.
- We build a time series platelets demand dataset using the ARIMA model with parameters extracted from a large-scale dataset.
- We apply LSTM through FL against the dataset to verify the effectiveness of FL for each local site holding regional data. The experimental results demonstrate the effectiveness of FL in enhancing demand forecasting accuracy, reducing model bias, and improving scalability across heterogeneous datasets and geographically dispersed supply chains.

2 RELATED WORK

Statistical techniques, e.g., autoregressive integrated moving averages (ARIMA), have been extensively used to capture time-series dynamics, becoming widely accepted for supply chain demand forecasting. However, such traditional methods often rely on a single centralized source, leading to privacy regulatory concerns [5, 17]. Motamedi et al. [14] highlighted the significance of integrating clinical predictors into demand forecasting models for improving accuracy. Schilling et al. [16] employed a statistical model and a deep neural network to forecast platelet demand within a hospital setting. Their models showed a potential decrease in platelet waste and shortage and substantially reduced financial implications.

Initially proposed by McMahan et al. [13], FL offers a promising solution for training models collaboratively across decentralized data resources. FedProx [9] was proposed to address statistical and system heterogeneity. Due to the wide availability of time series data, FL has been extensively studied with time series data in electrical load forecasting [7, 18] and traffic forecasting [11, 20]. Specifically, the use of LSTM with FL was investigated in [4, 10, 12, 15] for various purposes, including anomaly detection, intrusion detection, and behavior analysis. No study has used LSTM for FL in blood supply chain management.

3 PROPOSED METHOD

Given suitable parameters, ARIMA models could generate high-quality time series data that reflects the actual blood product demand in a given period with low errors [14]. Extensive research was conducted considering different ARIMA parameters to generate different demand scenarios faced at different hospitals in Canada.

3.1 Synthetic data of platelets demand

In our study, we use the three parameters fitted from the study in [14]. In addition, we generate datasets split in a heterogeneous fashion under various settings. Fig. 2 shows the time series data generated regarding demand and time range. We list the categories of the synthetic data below with the consideration of if the data is independent and identically distributed (i.i.d.):

- Balanced i.i.d. data. Each client generates equal amount of data using the same \(p, d, q\) values in the ARIMA model.
- Imbalanced i.i.d. data. Each client generates different volumes of data with the same parameters. Here we simulate the imbalance of data, commonly found in healthcare, as certain institutions have little or no historical data.
- Noisy data with zeros: A portion of clients have random zeros in their datasets, creating noise in the system. Considering our previous understanding of FL, we should expect this noise to have a negative impact on the performance of FL, possibly leading to worse prediction results compared to the local models for clients without random zeros. This simulates the situation in hospitals where there are missing or incomplete records.
- Data with inconsistent ARIMA parameters (non-i.i.d.): Data is generated using different ARIMA parameters since institutions may have different demand patterns.

3.2 The LSTM model

With the synthetic data, we adopt the LSTM model to forecast the demands. Our LSTM model consists of an input layer with 11 - 50 dimensions (previous 7 - 46 days’ values, day of month, month of year, day of week, and week of year), three hidden 64 \(\times\) 64 layers, with one fully connected layer with an output dimension of 1. For hyperparameter settings, we use 64 for the mini-batch size and 50 for the number of epochs in each round. We used ADAM optimizer, with the initial learning rate being 0.001. The total number of trainable parameters is 86,337 - 96,321 for the LSTM model.

4 EXPERIMENTAL RESULTS

4.1 Experimental settings

Our experiments use an FL system based on the Flower framework [2]. The experiments are conducted on a Windows PC (AMD Ryzen 7 5800X, 32GM RAM, RTX 4070 with 12GB GDDR RAM), simulating 1 centralized server and 5 local clients.
Figure 2: Data generated using ARIMA. First row: balanced i.i.d. data. Second row: imbalanced i.i.d. data. Third row: noisy data with zeros. Fourth row: data with various ARIMA parameters (one set of ARIMA parameters per client). Fifth row: data with various ARIMA parameters (two sets of ARIMA parameters per client).

Figure 3: The proposed data pipeline first generates synthetic time series blood product demand data using the ARIMA model learned in [14]. The data is preprocessed before it is split across multiple RDC clients. An LSTM model is then used to learn a federated model.

We apply two popular FL algorithms: FedAvg [13] and FedProx [9]. Fig. 1 shows the overall FL framework, where a client is a regional distribution center (RDC) holding that region’s blood supply and demand data. Each client trains its local data based on the global model maintained by the server. We train an LSTM model with the synthetic dataset under the following 5 scenarios in the FL setting. For each scenario, 5 clients are participating in training.

- For the balanced i.i.d. data scenario, each client carries 10 years of data (3650 data points).
- For the imbalanced i.i.d. data scenario, each client holds a different amount of data, carrying 20 years (7300 data points), 10 years (3650 data points), 1 year (365 data points), 5 years (1825 data points), and 40 years (14600 data points).
- For the noisy data scenario with zeros, 3 out of 5 clients have random zeros in their data, with Client 1 having 20% of data being random zeros, Client 3 having 30% of data being random zeros, and Client 5 having 40% of data being random zeros.
- For the heterogeneous scenario with one set of ARIMA parameters per client, two clients share one set of ARIMA parameters while three other clients share another set.
- For the heterogeneous scenario with two sets of ARIMA parameters per client, the five clients use a series combinations of ARIMA parameters for their local data.

4.2 Numerical Results

We compare the following three methods:

- **Federated**: A federated model is learned with FL with LSTM.
- **Local**: A model is learned from a single client with LSTM.
- **Centralized**: A centralized model is learned from all clients using a centralized LSTM. This can be considered the ideal learning method that defines the upper bound of learning.

We use the performance metrics of root mean square error (RMSE), mean absolute error (MAE), mean absolute percentage error (MAPE), and symmetric mean absolute percentage error (SMAPE).

**Balanced i.i.d. data.** When data is balanced across all the clients, Federated is expected to perform moderately better than Local trained with only local data, as the federated model learns knowledge from more clients. As illustrated by Fig. 4 and Table 1, Federated outperforms Local with RMSE by 3.21, MAE by 2.88, MAPE by 2.20, and SMAPE by 2.30.
Algorithm 1 Synthetic data generation with ARIMA. \(v\) is the autoregressive term, \(\phi\) is the moving average term, \(p\) is the length of \(v\), \(q\) is the length of \(\phi\), \(\mu\) is the mean of the error term, \(\sigma\) is the standard deviation of the error term, \(d\) is the number of unit root trends. \(n\) is the time series length, and \(burn\) is the number of discarded values used to start the data generation.

1: Initialization: Initialize an empty dataframe. Define the AR and MA terms \(\phi, v\). Define \(d, t, \mu, \sigma, n,\) and \(burn\).
2: Error terms: Create an array size of \((n + \max(p, q) + burn)\) with error terms following \(\mathcal{N}(\mu, \sigma^2)\).
3: for \(i\) in range(length of error terms array) do
   4:     obtain the AR term by \(\tilde{v} \cdot \hat{p}_{i-1}\)
   5:     obtain the MA term by current error term + \(\tilde{v} \cdot \hat{q}_{i-1}\)
   6:     current value is \(AR + MA + t\)
4: end for
5: Unit Root: If \(d! = 0\), introduce the unit root to the time series.
6: Return the last \(n\) values of the simulated ARIMA process.

Table 1: Errors for Local and Federated approaches.

<table>
<thead>
<tr>
<th>Method</th>
<th>RMSE</th>
<th>MAE</th>
<th>MAPE</th>
<th>SMAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balanced local</td>
<td>15.66</td>
<td>12.79</td>
<td>16.56%</td>
<td>14.12%</td>
</tr>
<tr>
<td>Balanced federated</td>
<td>12.45</td>
<td>9.91</td>
<td>12.55%</td>
<td>11.16%</td>
</tr>
<tr>
<td>Imbalanced local</td>
<td>34.99</td>
<td>28.73</td>
<td>14.63%</td>
<td>13.66%</td>
</tr>
<tr>
<td>Imbalanced federated</td>
<td>20.29</td>
<td>15.86</td>
<td>7.51%</td>
<td>7.50%</td>
</tr>
<tr>
<td>Noisy local</td>
<td>31.58</td>
<td>26.08</td>
<td>21.89%</td>
<td>17.97%</td>
</tr>
<tr>
<td>Noisy federated</td>
<td>27.62</td>
<td>22.61</td>
<td>19.09%</td>
<td>15.88%</td>
</tr>
<tr>
<td>Hetero. local</td>
<td>2.46</td>
<td>2.02</td>
<td>1.31%</td>
<td>1.20%</td>
</tr>
<tr>
<td>Hetero. federated</td>
<td>6.29</td>
<td>5.97</td>
<td>3.57%</td>
<td>3.50%</td>
</tr>
<tr>
<td>Hetero. M. local</td>
<td>22.16</td>
<td>17.47</td>
<td>10.25%</td>
<td>9.86%</td>
</tr>
<tr>
<td>Hetero. M. federated</td>
<td>21.67</td>
<td>17.02</td>
<td>9.99%</td>
<td>9.62%</td>
</tr>
</tbody>
</table>

by 4.01%, and SMAPE by 2.96%. The errors of Federated converge towards the upper bound with the Centralized. The results indicate that for RDCs with similar data distributions and comparable data volumes, FL can enhance the global model performance by weighing in more knowledge from nationwide.

Imbalanced i.i.d. data. When clients have various amounts of data, those with fewer data samples should see a significant improvement with the FL model. This resonates with the case that certain RDCs may have a shorter data history than their counterparts in other regions. Fig. 5 depicts the results of the local and global models on a client with 1 year’s worth of data. Compared to the previous setting with balanced data, the clients with fewer data histories greatly benefit from FL. At Round 7, Federated outperforms Local with RMSE by 14.7, MAE by 12.87, MAPE by 7.12%, and SMAPE by 6.16%. For RDCs with imbalanced data, FL proves capable of transferring the knowledge from established RDCs to new RDCs.

Noisy dataset with zeros. RDCs may be missing data rows from certain days for various reasons. We explore how FL can improve a noisy dataset with zeros. When the datasets contain zeros, preprocessing was done to replace the zeros with the preceding non-zero value. It is expected to see the federated model performing better than local models in the cases where more data were being random zeros. The evaluation results in Fig. 6 show Federated outperforms Local with RMSE by 3.96, MAE by 3.47, MAPE by 2.8%, and SMAPE by 2.09%. The results confirm that FL can improve demand forecasting accuracy in noisy datasets with missing rows.

Heterogeneous data with single ARIMA model per client. When ARIMA parameters are different, given that the data is non-i.i.d. time series data, it is expected that FL would perform worse since FL tends not to perform as well under heterogeneous settings. For time series data, if two clients have drastically different dataset trends, performing a federated model will give a model that does not do well on either of the clients because federated learning aims for generalization. Given the data heterogeneity, FedProx [9] was used...
This paper explored the potential of leveraging FL to enhance blood supply chain demand forecasting. By generating synthetic data with ARIMA models fitted from the real blood supply and demand data, we conducted extensive numerical studies on the benefits of FL with various data distribution settings. When clients have imbalanced i.i.d. data, FL can significantly boost the performance for clients with fewer data points. This is particularly meaningful for newly involved hospitals and RDCs in BSCM. We also learned that FL can still moderately improve the overall prediction errors with balanced i.i.d. data, justifying the need for a federated framework. Additionally, FL proved to mitigate the learning accuracy loss caused by missing data. Lastly, FL reduced the errors with heterogeneous data using multiple ARIMA models per client. Future work includes verifying the effectiveness of FL using actual data from BSCM.

REFERENCES


APPENDIX

A THE ARIMA MODEL

Effective in practical applications, the AutoRegressive Integrated Moving Average (ARIMA) model is a popular forecasting method for time series data, which captures a suite of different standard temporal structures in time series data, stationary, with a trend, or with a seasonal component. Let \( y_1, y_2, \ldots, y_t \) be the demand values over time period \( t \); the time series data can be written as:

\[
y_t = f(y_{t-1}, y_{t-2}, y_{t-3}, \ldots, y_{t-n}) + \epsilon_t. \tag{1}
\]

An ARIMA model assumes that the value of demand is a linear function of several previous past demand values and previous error values. Thus, the ARIMA model can be written as:

\[
\hat{y}_t = \mu + v_1 y_{t-1} + v_2 y_{t-2} + \cdots + v_p y_{t-p} + \epsilon_t - \phi_1 \epsilon_{t-1} - \phi_2 \epsilon_{t-2} - \cdots - \phi_q \epsilon_{t-q}. \tag{2}
\]

where \( \hat{y}_t \) is the response variable (the predicted demand), \( \mu \) is a constant, \( v_i \) and \( \phi_j \) are model parameters in which \( i \in [1, p] \) and \( j \in [1, q] \). Here \( p \) defines the number of autoregressive terms, and \( q \) is the number of lagged forecast errors in the prediction equation. For a nonseasonal ARIMA model, define \( d \) as the number of nonseasonal differences needed for stationarity. We can write the model as a function \( ARIMA(p, d, q) \) with 3 parameters.