Mobile Networks and Applications
https://doi.org/10.1007/s11036-020-01645-w

Predicting Systolic Blood Pressure in Real-Time Using Streaming Data and Deep Learning

Hager Saleh[1] · Eman M. G. Younis[2] · Radhya Sahal[3,4] · Abdelmgeid A. Ali[5]

Abstract
High systolic blood pressure causes many problems, including stroke, brain attack, and others. Therefore, examining blood pressure and discovering issues related to it at the right time can help prevent the occurrence of health problems. Nowadays, health-based data brings a new dimension to healthcare by exploiting the real-time patients' data to early detect systolic blood pressure (SBP). Furthermore, technologies typically associated with smart and real-time data processing add value in the healthcaredomain, including artificial intelligence, data analytic technologies, and stream processing technologies. Thus, this paper introduces a systolic blood pressure prediction system that can predict SBP in real-time and, therefore, can avoid health problems that may stem from sudden high blood pressure. The proposed system works through two components, namely, developing an offline model and an online prediction pipeline. The aim of developing an offline model module is to develop the model using investigate different deep learning models to achieve the smallest root mean square error. It has been developed using Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM), Bidirectional Short-Term Memory (BI-LSTM), Gated Recurrent Units (GRU) models andMedical Information Mart for Intensive Care (MIMC II) SBP time-series dataset. The online prediction pipeline module is using Apache Kafka and Apache Spark to predict the near future of SBP in real-time using the best deep learning model and SBP streaming time-series data. The experimental results indicate that the BI-LSTM model has achieved the best performance using three hidden layers, and it is used to predict the near future of SBP in real-time.

Keywords Blood pressure prediction · Time-series forecasting · Stream processing · RNN · LSTM · BI-LSTM · GRU · Apache Kafka and Apache Spark

✉ Hager Saleh
hager.saleh.fci@gmail.com

Eman M. G. Younis
eman.younas@mu.edu.eg

Radhya Sahal
radhya.sahal.dsi@gmail.com

Abdelmgeid A. Ali
abdelmgeid@yahoo.com

[1] Faculty of Computers and Information, Minia University, Minia, Egypt

[2] Information Systems Department, Faculty of Computers and Information, Minia University, Minia, Egypt

[3] Data Science Institute, National University of Ireland, Galway, Ireland

[4] Faculty of Computer Science and Engineering, Hodeidah University, Hodeidah, Yemen

[5] Computer Science Department, Faculty of Computers and Information, Minia University, Minia, Egypt

1 Introduction

Blood Pressure (BP) is among the most acute symptoms of human health quality. This is because high blood pressure affects humans' health, and various diseases may follow it later [2]. In this respect, studies demonstrated the risk of having many conditions for hypertension, including stroke, heart failure, memory loss, and damage to eyes, and many more [23, 26]. According to the World Health Organization (WHO) [21], elevated blood pressure is expected to lead to 7.5 million deaths, which counts for about 12.8% of the total deaths. High blood pressure is a factor that raises the risk of cardiovascular diseases and strokes [20] Therefore, hypertension detection is essential for adequate and timely treatment [6]. The highest level of the artery pressure is called systolic blood pressure (SBP), while the lowest level of the artery pressure is called diastolic blood pressure [17]. According to the American Heart Association [1], SBP is classified into three categories: value less than 120 for systolic is normal blood pressure. High SBP is when the readings mostly range

Published online: 04 September 2020

between 120 and 129. The third category is hypertension, which is SBP that consistently varies between 130 and 139. Therefore, there is a persistent need for a highly accurate system that can detect early high blood pressure before it occurs. This is expected to lead to preventing stroke, myocardialinfarction, sudden death, and renal failure.

Recently, machine learning algorithms are playing a significant role in the medical field. Some researchers used machine learning methods to predict stroke [13], heart disease [9], and Alzheimer's disease citesmola2004tutorial12, random forest model [26], and linear regression model [7]) have been utilized to predict blood pressure. Nowadays, the capability to foresee the future, relying solely on past data, results in strategic merits, which may be considered the opener of organizations success [20]. Consequently, the Time Series Forecasting (TSF) system signifies the future value prediction, which is based on information about thepresent and past condition of the system [4]. Currently, researchers have used artificial neural network algorithms and deep neural network algorithms to analyze and extract meaningful statistics from time-series data to predict blood pressure. For example, Koshimizu, Hiroshi, et al. [14] have used deep neural networks with loss function to predict blood pressure. Lee, Joon, and R. G. Mark [15] have developed an artificial neural network using time series data, which are collected from the Medical Information Mart for Intensive Care database to predict blood pressure. Similarly, Li, Xiaohanet al. [16] used long short-term memory with a contextual layer to predict the trend of users' blood pressure. Girkar, Uma M., et al. [8] have also applied logistic regression algorithms with regularization, long short-term memory networks (LSTM), and the gated recurrent units network to predict blood pressure. Su, Peng, et al. [32] have used deep recurrent neural networks (RNN), which includes a multilayer of Long Short-Term Memory network to predict blood pressure. Zhao, Qingxiang, et al. [38] have proposed a model based on Long Short Term Memory Networks to predict the SBP and diastolic blood pressure. Su, Peng, et al. [33] proposed a novel deep recurrent neural network to predict SBP and diastolic blood pressure. It includes multi-layered Long Short-Term Memory, which is a bidirectional structure and residual connections. The bidirectional structure is used to access larger-scale context information of input sequences. The residual connections are used to allow gradients in deep RNN to propagate.

At the present time, healthcare data management is the process of storing and analyzing data generated from different sources. Mining the healthcare data gives the healthcare systems ability to build patients' profiles to personalize their treatments, improve communication, and enhance health outcomes. Therefore, the industry has evolved by utilizing the explosion in data coming from new data sources that have not been widely used within healthcare digitalization. Wearable devices are one such example that potentially brings a new dimension to healthcare data management as it enables further smart data management among patients' data. That motivates healthcare marketers to make predictions about which patients may have a propensity toward certain conditions. In doing so, more advanced data processing technologies are adopted in the healthcare industry to support real-time data processing, giving health systems a higher level of accuracy with time-sensitive data processing. Stream processing technologies such as Apache Spark [29], Apache Kafka [12], and Apache Storm [31] are used to create nearly instantaneous personalization opportunities and profoundly informed decisions by performing big data analytics tasks, i.e., collecting and analyzing the health-based streaming data (i.e., data generated from sensors) on-the-fly. This has led academia, health care organizations, and practitioners to invest more in streaming technologies. For example, streaming technologies can aggregate timely health-based streaming data from wearable devices, clean them, normalize, and enrich them for pattern recognition and complex event processing. These technologies have added some useful new forms of solutions when moving large-scale data around for real-time applications. For example, Zhang, Fan, et al. [37] have developed a task-level adaptive MapReduce framework to apply real-time streaming data in healthcare applications. Veeravalli [36] et al. have developed an anomaly detection system for streaming data, which is generated from wearable healthcare devices. Hager, et al. [40] have developed a system using machine learning algorithms to predict heart disease from streaming Twitter data. On the other hand, artificial intelligence has played an essential role in advancing hypertension care. In particular, health-based streaming data integrated with AI technologies can detect early signs of hypertension by analyzing data produced from wearable devices. The previous studies of SBP prediction have focused only on predicting SBP depends on developing machine learning models on stored historical data. These studies haven't used real-time data to predict SBP in the future (i.e., streaming data generated from SBP sensors). That motivates us to predict near future of SBP from a sequence of time-series SBP in real-time using deep learning techniques and stream processing technologies.

To the best of the authors' knowledge, there is no previous work considering the use of streaming data for predicting high blood pressure in real-time. Therefore, this paper aims to build a prediction system that can work in real-time through receiving SBP time series from the simulated sensors and then send it to the developed model to predict the five minutes of SBP in advance. In particular, the proposed prediction system aims to avoid future potential risks resulting from high blood pressure by predicting it in real-time. The proposed system has two components: 1) Developing an offline model, and 2) the online prediction pipeline. The first component is developing the

optimal offline prediction model based on SBP history from Medical Information Martfor Intensive Care (MIMC II) [24]. Four time series forecasting techniques, namely Recurrent Neural Network, Long short-term memory, Bidirectional long short term memory, and Gated recurrent units, are used to develop the offline model. The second component is using Apache Kafka and Apache Spark to evaluate the proposed system in real-time. The paper contributions can be summarized as:

- Developing a generator to generate time-series data for SBP (i.e., the generated data is simulated as readings of Blood Pressure Sensor data)
- Developing a real-time system that can help avoid the risk of high SBP in the near future.
- The offline models are trained and tested by using a real-time series dataset from Medical Information Mart for Intensive Care.
- Comparing four deep learning models to identify the best model that will be used to predict the SBP in real-time, namely RNN, LSTM, BI-LSTM, and GRU.

Moreover, the prediction and forecasting terms will be used interchangeably throughout the paper. The remainder of this paper is organized as follows. Section 2 presents the proposed SBP prediction system. Section 3 discusses the experimental results and model evaluation. Finally, Section 4 concludes the paper.

2 Systolic blood pressure prediction system

Systolic Blood pressure prediction system consists of two main components, which are developing offline model and an online prediction pipeline as shown in Fig. 1 Each component will be described in detail in the following subsections.

2.1 Developing an offline model

The aim of developing an offline model component is to develop the model using different deep learning models to achieve the smallest root mean square error throughout experimentation. Deep learning models, including Recurrent Neural Network, Long Short-Term Memory, Bidirectional Short-Term Memory, and Gated Recurrent Units, are used to train and test the models. Besides, data pre-processing has played an essential role in achieving the smallest root mean square error. Figure 1 presents the main stages of this component, which are data collection, data pre-processing, data splitting, optimization/training models, and evaluating the models. Each stage is described as follow:

2.1.1 Data collection

For developing an offline model, we extracted the SBP time series (univariate dataset) on a minute by minute basis for one patient from Medical Information Mart for Intensive Care (MIMC-II) [24] database. The patientsâ€™ data of MIMC-II were gathered from a difference of ICUs in Beth Israel Deaconess Medical Center in Boston, Massachusetts. The reason for using the MIMIC II database is because it is a free critical care database and open to researchers. Besides, it has been applied by different research works [19, 25, 34].

2.1.2 Data pre-processing

The pre-processing of SBP time series dataset has the following steps:

Transforming raw data into stationary data Basically, the stationarity data time series means that the statistical properties of a process generating a time series data do not change over time. It does not mean that the series does not change over time, just that the way it changes does not itselfchange over time, which means that the way of change is a linear function, not a constant. In particular, the statistical measures such as the mean, standard deviation, covariance, and autocorrelation are somewhat similar and constant over time. Furthermore, the stationary processes are considered a subclass of a wider family of possible models in reality. The main feature of this sub-class is much easier to model, investigate, and possible to predict, as the way of change is predictable.

On the other hand, the non-stationary time series data has a trend as a rule, which makes it unpredictable and cannot be modeled or forecasted. The observations from a non-stationary time series data show seasonal effects, trends, and other structures that depend on the time index. Furthermore, the statistics of the non-stationary time-series data like the mean and variance do change over time, providing a drift in the concepts a model may try to capture. According to this work, the SBP time-series dataset (i.e., raw data) is considered a non-stationary time-series data. Substantially, making the time-series stationary is critical if we want the prediction model to work well. To avoid the unskillful forecasts due to the trends in the SBP time series dataset, we have applied a transformation method (i.e., differencing) forthe dataset. In doing so, we have computed the difference of consecutive terms in the series differencing, which is typically performed to get rid of the varying means. In particular, differencing is the observation from the previous time step (t-1) which is subtracted from the current observation (t). In the current preprocessing step,We have applied the order differencing which is a trend that can be removed by subtracting the previous value from each value in the series. The order differencing has been implemented using custom function which is called different.

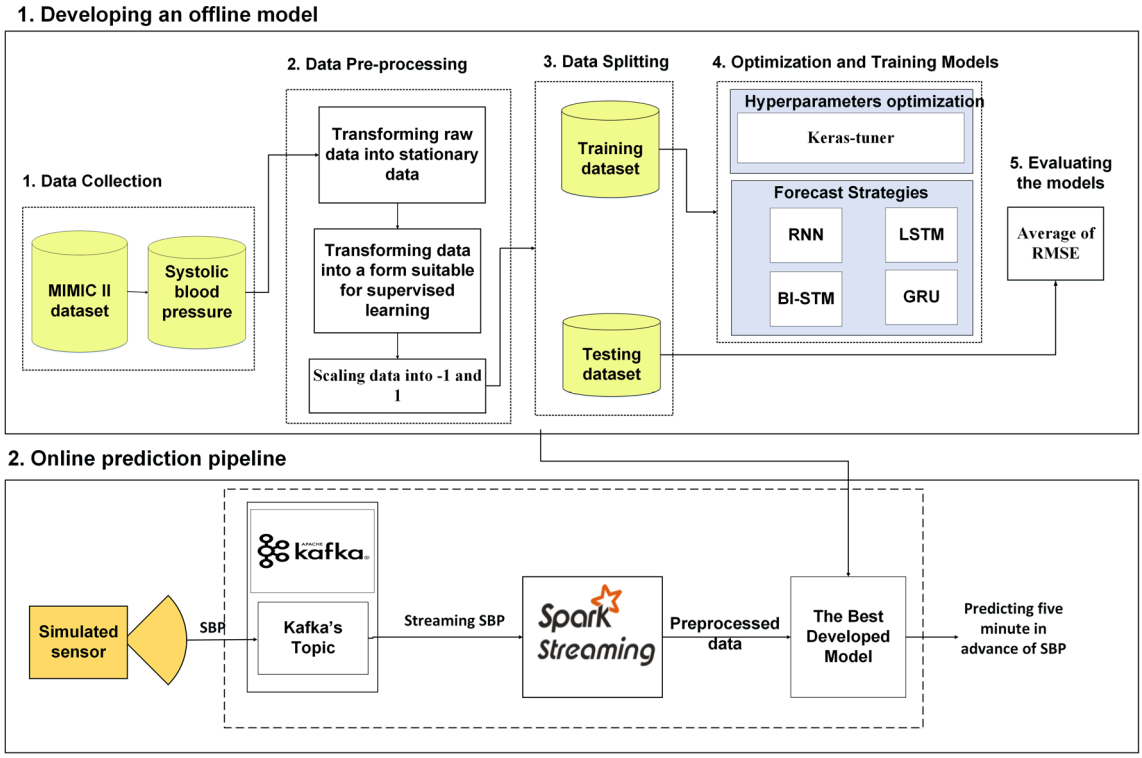

Fig. 1 The architecture of SBP prediction system

The function takes the time series and the interval for the difference calculation, e.g. 1 for a trend difference. For reverse different, we have used custom method which is called *inverse_difference*.

Transforming data into a form suitable for supervised learning For any supervised learning model, a machine learning algorithm is used to learn the association function between the input variables (X) and output variables (Y). For prediction problems, the model is trained using the input data (X). Then, the model can predict the output variables (Y) for the new input data (X). Most prediction problems involve a time component, so time series prediction can be framed as a supervised learning problem. In doing so, time series data must be transformed into a supervised learning problem. Also, the non-stationary data should be processed in the form that can be used in a supervised training process by removing temporal structure like trends and seasonality [18].

In this work, we have extracted time-series data to create samples for the prediction models. Each sample consisted of two-time intervals as: 1) observation window (X), and 2) target window (Y) achieved through the sliding window method. The observation window is known as the input and its size in the sample depends on the user-defined sequence where the target window is known as the output and its size in the sample depends on the prediction models. Regarding this work, we have considered the observation window as 6 min while the target window is considered as 5 min.

To configure target window size, we have calculated the correlation for time series observations with observations with previous time steps, called lags. Basically, there are three types of correlation which are; positive correlation, negative correlation, and zero correlation. Regarding this work, the AutoCorrelation Function(ACF) is computed which is also sometimes called lagged correlation or serial correlation. The AFC function describes the autocorrelation between an observation and another observation at a prior time step that includes direct and indirect dependence information. According to the ACF function of the historical data of SBP which plotted in Fig. 2, we have chosen lag as 6.The main goal of our work is to predict SBP in real-time, which helps avoid the risk of high blood pressure in the near future. In our work, we achieve this goal we used the open-source dataset,

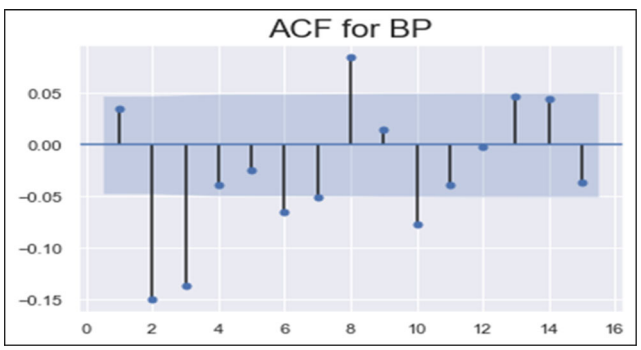

Fig. 2 The Autocorrelation Function (AFC) for Blood Pressure Dataset

and we extract the SBP time series for one patient. Experimentally, when we tried to predict SBP in 10 min in advance, the RMSE was higher due to the limitation of the dataset which collected for one patient. Obviously, if the dataset is a larger, the RMSE will be smiler and allow as to predict SBP for a larger number of minutes in advanced such as 20 and 30 min. Therefore, we have considered the observation window as 6 min while the target window is considered as 5 min.

Scaling data into −1 and 1 We have done a normalization which is a rescaling of data from the original range to a new range between −1 and 1. In particular, we have scaled the data to be within the range of −1 and 1; and the reason for doing so is that RNN modelsprefer to work on data within the range of its activation function. We have used *MinMaxScaler(feature_range = (−1, 1))* [27] that is library in Python language to scaling dataset between −1 and 1. Also, we have used RNN, LSTM, BI-LSTM,and GRU models with hyperbolic tangent (*tanh*) as activation function. In the evaluation phase (i.e., model testing), we have transformed the predictions back into the original scale, in order to correctly evaluate the model performance. For reverse scaling, we used *inverse_transform* function that is exist in *MinMaxScaler* library in Python. It inverse scaling of data row to original values.

2.1.3 Data splitting

In this step, the dataset is divided into a (80%) training set and a (20%) testing set. The training set is used to optimize and train the models, while the testing set is used to evaluate the models.

2.1.4 Optimization and training models

We used Recurrent Neural Networks (RNN) [22], Long Short-Term Memory (LSTM) [11],Bidirectional Short-Term Memory (BI-LSTM) [40], and Gated Recurrent Units (GRU) [3] as four candidates forecasting models for our SBP forecasting problem. For each model, we applied a different number of hidden layers, specifically from one hidden layer up to three hidden layers and used a dropout layer [10].The output layer has 5 neurons because we predict 5 min in advance for SBP. And the activation function for input layer is relu. We used the ADAM optimizer [5] and the Mean Square Error (MSE) as a loss function. For Hyper-parameter optimization, the keras-tuner library [35] is used to select the best value for each parameter of deep neural network models. Table 1 shows the value for each parameter as tuned for each model.

Table 1 The value for each parameters in models

Parameter	Value
Dropout rate	Within the range of 0.1 rate to 0.5 rate
The number of neurons	Within the range of 10 neuron to 200 neurons

2.1.5 Evaluating the models

For measuring modelsâ€™ performance, we calculated average of Root Mean Square Error (RMSE) for each models, which can be given as follows:

$$\mathrm{RMSE} = \sqrt{\left(\frac{1}{n}\right)\sum_{i=1}^{n}\left(y_i^{obs} - y_i^{Pred}\right)^2}$$

2.2 Online prediction pipeline

The online prediction pipeline component which is implemented as data pipeline analytics contains two steps including data generating and collection, and data streaming analysis/prediction. Each phase can be described as follows.

2.2.1 Data generating and collection

In this phase, we have used Apache Kafka and Apache Spark, distributed streaming technologies to build real-time streaming data pipelines that reliably get data from SPB sensors. We have chosen Apache Kafka [12] as it is the state-of-the-art distributed large-scale real-time data application. It has high delivery and ordering guarantees of the data stream, which makes it reliable to collect critical time-series data such as health data (e.g., blood pressure, a heart pulse), climate data, industrial data, and many more. Also, Apache Kafka is responsible for sending input streams to the stream processing system (i.e., Apache Spark). In this work, we have developed a simulated sensor to generate time-series SBP data as JSON format. Each one minute includes two parameters which are SBP value and timestamp. A snapshot of generated data SBP streaming and timestamp is presented in Fig. 3 Also, we have used a Producer API, library in Kafka to push SBP as streaming to Kafka topic.

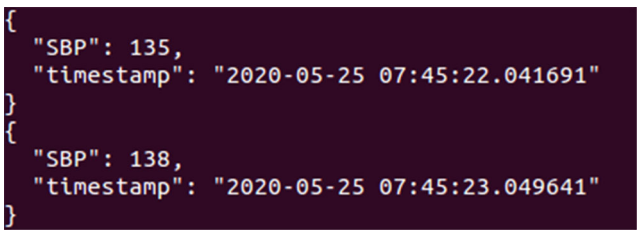

Fig. 3 JSON-like generated time-series streaming SBP

2.2.2 Streaming analysis and prediction

Apache Spark [29] is an open-source Big Data processing, an in-memory, and streaming enabled engine. Apache Spark uses the micro batching procedure to perform stream processing by dividing the incoming stream of events into small batches andkeeping the latency of stream processing under control. Therefore, it claims to be faster than Apache Hadoop by achieving better performance thanks to its micro-batch processing. A strength point for using Apache Spark is its capacity to enable batch andstreaming analysis in the same platform and its package streaming, which is capable of processing streaming data from different sources, including those medical sensor devices. Apache Spark includes Spark Streaming API [30] that can readdata from Apache Kafka, and then it can process data using complex algorithms like a map, reduce, join, and window.

In this work, Apache Spark streaming is used to consume SBP streams from Kafkaâ€™s topic. Substantially, Spark streaming leverages the advantage of windowed computations in Apache Spark. Window functions are used to do operations over a set of rows. Sparkstreaming allows applying transformations over a sliding window of data. Basically, sliding windows assign rows to windows of fixed length; hence, sliding windows can be overlapping with the slide, which is smaller than the window size. Any Spark slidingwindow operation requires specifying two parameters as 1) window length, which defines the duration of the window, 2) sliding interval, which represents the interval at which the window operation is performed. More details about the Spark streaming configurations will be introduced later in the experimental results section.

3 Results analysis and discussion

3.1 Experiment setup

The proposed SBP prediction system has been implemented using deep learning models and streaming data technologies. The experiments of the developing offline model component are done to find the optimal model that has the smallest RMSE. The RNN, LSTM, BI-LSTM, and GRU models with different layers are implemented using the Keras in Python 3.7. The dataset is split into an 80% training set and a 20% testing set. The training set is used to optimize models using different hyperparameters by the Keras-tuner library. The testing set is used to evaluate the models using RMSE.

For the online prediction pipeline component, the candidate model, which has obtained the performance of the best-developed model in the offline component, is used as a real-time blood pressure predictor to predict the next 5 min SBP in the near future using 6 min back of SBP streaming. In doing so, we have set lag to 6 (i.e., Lagging means going some steps back in time). Furthermore, the RNN, LSTM, BILSTM, and GRU models are primarily suited for multi-step-ahead prediction. To assess the prediction model performance, average Root Mean Square (RMSE) is used.

Also, streaming data is generated from a simulated sensor, which is used to evaluate the system in real-time. The optimal model is used to predict SBP in real-time, which helps avoid the risk of high blood pressure in the near future. We have implemented a producer using Python, which is a simulated sensor that generates SBP time-series data and sends them to Kafka's topic.

The Apache Kafka receives SBP streaming from the simulated sensor. Spark Streaming (consumer) is used to read SBP time-series streaming from the Kafka topic that is implemented by Pyspark. The online prediction pipeline component is executed on a Spark cluster, which consists of one master node and two worker nodes. We have used Ubuntu virtual machines that run Java (JVM) to build the cluster. The name node and worker nodes have 20GB of RAM, 7 cores, and 100GB disk.

The following subsections show the offline component's experimental results, where RNN, LSTM, BI-LSTM, and GRU prediction models have been used. Then, one candidate model which has obtained the performance of the best-developed model in the offline component is used as a real-time blood pressure predictor.

3.2 Results of an offline phase

The results of applying deep learning models, RNN, LSTM, BI-LSTM, and GRU on the historical MIMC II dataset are explained in this section. Difference constant hyperparameters for each model are configurated, such as lag: 6 and a learning_rate: 0.0001, batch size: 1, and epochs: 10.

3.2.1 Results of RNN

Table 2 shows the results of RNN models with one hidden layer up to 3 hidden layers and their corresponding hyperparameters. It is noted that the averages of RMSE are 3.025, 3.065, and 3.055 using one, two, and three hidden layers, respectively. It can be noticed that RNN with one hidden layer has the smallest RMSE, whereas RNN with three hidden

Table 2 The result of RNN model

#. layers	#. hidden neurons	dropout	Average of RMSE
1	[480]	.6	3.025
2	[140,320]	[.3,.4]	3.065
3	[160,300,440]	[.5,.4,.2]	3.055

layers has the largest RMSE error. It can be observed that the RNN using one layer appears to exhibit lower RMSE prediction performance.

3.2.2 Results of LSTM model

Table 3 shows the LSTM model results using one, two, and three hidden layers and their corresponding hyper-parameters. It is noted that the averages of RMSE are 3.106, 3.074, and 3.06 using one, two, and three hidden layers, respectively. Ascan be seen, the LSTM model with three hidden layers is the best model, among others, compared to LSTM models, where it achieved the best performance with the smallest values of the three errors.

3.2.3 Results of BI-LSTM model

Table 4 shows the results of the three BI-LSTM model using one, two, and three hidden layers and their corresponding hyper-parameters. It is noted that the averages of RMSE are 3.101, 3.069, and 2.840 using one, two, and three hidden layers, respectively. As can be seen, the BI-LSTM model with three hidden layers is the best model among other BI-LSTM models using one and two hidden layers where it achieved the best performance with the smallest values of the three errors.

3.2.4 The result of GRU model

Table 5 shows the results of the three GRU model using one, two, and three hidden layers and their corresponding hyper-parameters. It is noted that the averages of RMSE are 3.077, 3.084, and 3.061 using one, two, and three hidden layers, respectively. The RMSE results have shown that the GRU model using three hidden layers has the best performance among GRU models using one and two hidden layers.

3.3 Discussion

The best Predicting Model can be considered given these empirical results seen in Table 1-4. The best performance of the RNN is the smallest error using one hidden layer, RSME = 3.025. For LSTM, BI-STM, and GRU, three hidden layers have achieved the best performances, which are 3.06, 2.48, and 3.061, respectively. It is observable in Fig. 4 that the prediction model with the BI-LSTM algorithm outperforms the RNN, LSTM, and GRU model. Thus, it can be tentatively concluded that the BI-LSTM model using three hidden layers will be used to predict the SBP 5 min in advance in real-time.

Table 4 The results of BI-LSTM model

#. layers	#. hidden neurons	dropout	Average of RMSE
1	[100]	[0.4]	3.101
2	[80,20]	[0.4,0.2]	3.069
3	[260,240,60]	[0.1,0.2,0.1]	2.840

3.4 Evaluating the proposed system in real-time using streams of SBP

After testing developing an offline model component, the best model is BI-LSTM using three hidden layers, which is used to predict the SBP in real-time using streaming SBP time-series data. The online prediction pipeline component aims to evaluate the proposed system's ability to work in real-time by receiving streaming SBP time series from a simulated sensor. In addition to assessing the ability of the system to predict 5 min of SBP in advance. These predicated SBP readings could be used to warn thepatients of the risk in advance to take their attention.

In the technical level, as we described earlier in Section 2 (i.e., online prediction pipeline component), SBP streaming time-series data is collected using simulated SBP sensor data. Then, the streaming time series data is sent to Apache Kafka which publishes it. Afterwards, Spark streaming consumes the streaming SBP time-series data. According to the prediction of streaming data in 5 min in advance (i.e., using lagging for steps back in time), we have configured Spark sliding window parametersas 6 for window length and 1 for a sliding interval. In particular, Spark streaming consumes six SBP readings for 6 min in its windowed DStream. The different steps will be applied to SBP returned from the sliding window, such as difference between SBP, and sliding SBP into (−1,1). Then, preprocessed SBP data are sent to the best-developed model to predict SBP in the 5 min in advance. After that, the predicted SBP is inversed different

Table 3 The result of LSTM model

#. layers	#. hidden neurons	dropout	Average of RMSE
1	[30]	0.4	3.106
2	[80,150]	[0.3,0.4]	3.074
3	[170,130,80]	[0.2,0.3,0.4]	3.06

Table 5 The result of GRU model

#. layers	#. hidden neurons	dropout	Average of RMSE
1	[130]	0.4	3.077
2	[120,140]	[0.3,04]	3.084
3	[200,50,50]	[0.4,0.4,0.3]	3.061

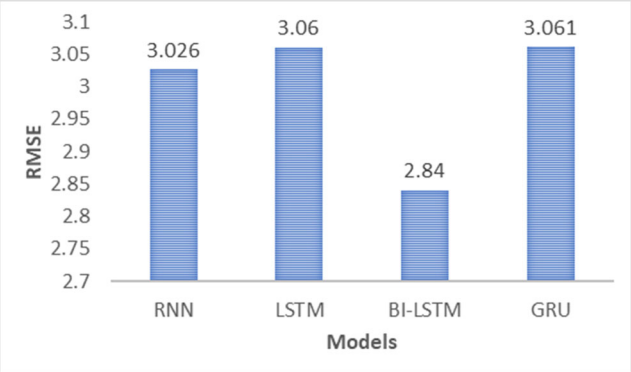

Fig. 4 Comparison of best prediction models using RMSE

and then inversed scaling to get the actual values. The predicted SBP is used to notify the patient in case of hypertension risk. Table 6 shows a sample of generated and predicted SBP. In particular, the simulated sensor sends the generated SBP reading every minute, which is queued in the Kafka's topic. For example, the six published SBP readings for the first 6 min are 120.546, 125.817, 123.911, 130.1501, 133.175, and 134.637. Then, the Spark streaming window consumes the sequences of generated SBP from the Kafka's topic every 5 min with 1 min slide. Afterward, the sequence of consumed SBP within DStream is sent to the best prediction mode (i.e., Bl-LSTM) in real-time to predict 5 min SBP in advance. The Bl-LSTM prediction model uses the 6 min readings to predict 5 min readings in real-time. With time increasing, Spark streaming consumes SBP readings with 1 min shifting for each windowed batch of the source DStream (see the bolded SPB reading in Table 5). For instance, for the 7th min, Spark streaming consumes 6 SBP readings from the 2nd min, such as125.817, 123.911, 130.1501, 133.175, 134.637, and 130.66, and so on.

Table 6 Sample of generated and predicted SBP

Time (min)	Sequences of SBP in Kafka topic	Sequences of SBP in Spark streaming window	Predicted SBP
1–6	120.546	120.546	136.58
	125.817	125.817	138.57
	123.911	123.911	140.49,
	130.1501	130.1501	142.43
	133.175	133.175	144.39
	134.637	134.637	
1–7	120.546	125.817	131.51
	125.817	123.911	132.42
	123.911	130.1501	133.24
	130.1501	133.175	134.09
	133.175	134.637	134.96
	134.637	130.66	
	130.66		
1–8	120.546	123.911	140.41
	125.817	130.1501	142.18
	123.911	133.175	143.85
	130.150	134.63	145.56
	133.175	130.66	147.29
	134.63	138.7	
	130.66		
	138.7		
1–9	120.546	130.1501	144.21
	125.817	133.175	145.980
	123.911	134.63	147.65
	130.1501	130.66	149.36
	133.175	138.7	151.09
	134.63	142.5	
	130.66		
	138.7		
	142.5		

4 Conclusions

In this paper, the core merit of the proposed SBP prediction system is its ability to handle real-time SBP streaming data that contain patientsâ€™ data efficiently and investigate the SBP level to notify the patients of the hypertension risks. The proposed system was developed using Apache Spark and Apache Kafka and deep neural networks. It consists of two components, namely, developing an offline model and online prediction pipelines. The offline model developing component is used to investigate different deep learning models to find the best model that can achieve the best performance (i.e., lower RMSE). In this component, we have compared four deep learning models, which are RNN, LSTM, BI-LSTM, and GRU models using the MIMC-II SBP time-series dataset.The results have proved that the BI-LSTM with three hidden layers has achieved the best performance, with the least error rate. In the online prediction pipeline component, streams of SBP time-series data are generated and then sent to Kafka topic. Sparkstreaming reads the data from the Kafka topic and then forwards it to the best deep learning model, which is BI-LSTM, to predict the near future of SBP in real-time. Overall, the comparison of deep learning models and predicting the future SBP in real-time based on a sequence of SBP time-series data contribute to improving the SBP prediction system in the applied solutions regarding real-time data processing, deep learning, and health-care applications.

For future work, we plan to develop our proposed system to be a healthcare-based cloud service to provide a systolic blood pressure prediction service to the users/patients. We have to identify the combination of the requirements of the systolic blood pressure prediction using streaming data and deep learning. As there are hundreds of cloud providers and AI vendors such as Google, Azure, Amazon, IBM, ASP, and so on, we have to look for a highly scalable platform for analyzing collected massive health-based streaming data which generated from wearable devices, i.e., blood pressure sensors. Also, we plan to deploy the developed cloud service to be used by healthcare application in the industrial setting such as 1) real-time dashboard for healthcare monitoring, 2) alarm systems which connected to the hospital emergency department, 3) a mobile app and web service that allow users/ patients to explore their profile and link them as medical records for their doctors, 5) hospital medical records which stored in big data cloud storage (e.g., HDFS, 189 MongoDB), and 6) hospital social networks considering patientsâ€™ privacy.

References

1. Understanding blood pressure readings. https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings. Accessed 10 Apr 2020.
2. Chiang PH, Dey S (2018) Personalized effect of health behavior on blood pressure: machine learning based prediction and recommendation. In: 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom), pp 1–6. IEEE
3. Chung J, Gulcehre C, Cho K, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling. arXiv preprint arXiv:1412.3555
4. Cortez P, Rocha M, Neves J (2004) Evolving time series forecasting Arma models. J Heuristics 10(4):415–429
5. Kingma DP (2015) Ba j. Adam: a method for stochastic optimization. In: The international conference on learning representations
6. Egbi OG, Ogoina D, Oyeyemi A (2018) Prevalence of hypertension and associated factors in a rural community in Bayelsa state. Int J Res Med Sci 6(4):1106
7. Ghosh S, Banerjee A, Ray N, Wood PW, Boulanger P, Padwal R (2016) Continuous blood pressure prediction from pulse transit time using ecg and ppg signals. In: 2016 IEEE Healthcare Innovation Point-Of-Care Technologies Conference (HI-POCT), pp 188–191. IEEE
8. Girkar UM, Uchimido R, Lehman LwH, Szolovits P, Celi L, Weng WH (2018) Predicting blood pressure response to fluid bolus therapy using attention-based neural networks for clinical interpretability. arXiv preprint arXiv:1812.00699
9. Gupta D, Sharma P, Choudhary K, Gupta K, Chawla R, Khanna A, Albuquerque VHCd (2020) Artificial plant optimization algorithm to detect infected leaves using machine learning. Expert Systems p e12501
10. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, Salakhutdinov RR (2012) Improving neural networks by preventing co-adaptation of feature detectors. arXiv preprint arXiv:1207.0580
11. Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780
12. Apache Kafka. https://kafka.apache.org/. Accessed May 2020.
13. Khosla A, Cao Y, Lin CCY, Chiu HK, Hu J, Lee H (2010) An integrated machine learning approach to stroke prediction. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining, pp 183–192
14. Koshimizu H, Kojima R, Kario K, Okuno Y (2020) Prediction of blood pressure variability using deep neural networks. Intern J Med Inform: 104067
15. Lee J, Mark RG (2010) A hypotensive episode predictor for intensive care based on heart rate and blood pressure time series. In: 2010 Computing in Cardiology, pp 81–84. IEEE
16. Li X, Wu S, Wang L (2017) Blood pressure prediction via recurrent models with contextual layer. In: Proceedings of the 26th International Conference on World Wide Web, pp 685–693
17. Mann JK, Kaffashi F, Vandendriessche B, Jacono FJ, Loparo K (2020) Data collection and analysis in the icu. In: Neurocritical Care Informatics. Springer, pp 111–134
18. Masum S, Chiverton JP, Liu Y, Vuksanovic B (2019) Investigation of machine learning techniques in forecasting of blood pressure time series data. In: International Conference on Innovative Techniques and Applications of Artificial Intelligence. Springer, pp 269–282
19. Mikhno A, Ennett CM (2012) Prediction of extubation failure for neonates with respiratory distress syndrome using the mimic-ii clinical database. In: 2012 Annual international conference of the IEEE Engineering in Medicine and Biology Society. IEEE, pp 5094–5097
20. Mitchell GF (2014) Arterial stiffness and hypertension. Hypertension 64(1):13–18
21. Raised blood pressure https://www.who.int/gho/ncd/risk_factors/blood_pressure_prevalence_text/en/. Accessed May 2020.

22. Prasanth T, Gunasekaran M (2019) Effective big data retrieval using deep learning modified neural networks. Mobile Networks Appl 24(1):282–294
23. Rendle S (2010) Factorization machines. In: 2010 IEEE International Conference on Data Mining. IEEE, pp 995–1000
24. Saeed M, Lieu C, Raber G, Mark RG (2002) Mimic ii: a massive temporal icu patient database to support research in intelligent patient monitoring. In: Computers in cardiology. IEEE, pp 641–644
25. Scott DJ, Lee J, Silva I, Park S, Moody GB, Celi LA, Mark RG (2013) Accessing the public mimic-ii intensive care relational database for clinical research. BMC Med Inf Decision Making 13(1):9
26. Sharma P, Choudhary K, Gupta K, Chawla R, Gupta D, Sharma A (2020) Artificial plant optimization algorithm to detect heart rate & presence of heart disease using machine learning. Artif Intell Med 102:101752
27. sklearn (2020): Minmaxscaler. https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html. Accessed May 2018
28. Smola AJ, Schölkopf B (2004) A tutorial on support vector regression. Stat Comput 14(3):199–222
29. Spark A (2020) Apache spark. https://spark.apache.org/. Accessed May 2020
30. Spark A (2020) Spark streaming. https://spark.apache.org/docs/latest/streaming-programming-guide.html. Accessed May 2020
31. Storm A (2020) Apache storm. https://storm.apache.org/. Accessed May 2020
32. Su P, Ding XR, Zhang YT, Liu J, Miao F, Zhao N (2018) Long-term blood pressure prediction with deep recurrent neural networks. In: 2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). IEEE, pp 323–328
33. Su P, Ding XR, Zhang YT, Liu J, Miao F, Zhao N (2018) Long-term blood pressure prediction with deep recurrent neural networks. In: 2018 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI). IEEE, pp 323–328
34. Sun J, Reisner A, Saeed M, Mark R (2005) Estimating cardiac output from arterial blood pressurewaveforms: a critical evaluation using the mimic ii database. In: Computers in Cardiology, 2005. IEEE, pp 295–298
35. keras-tuner. https://keras-team.github.io/keras-tuner/. Accessed Apr 2020
36. Veeravalli B, Deepu CJ, Ngo D (2017) Real-time, personalized anomaly detection in streaming data for wearable healthcare devices. In: Handbook of Large-Scale Distributed Computing in Smart Healthcare. Springer, pp 403–426
37. Zhang F, Cao J, Khan SU, Li K, Hwang K (2015) A task-level adaptive mapreduce framework for real-time streaming data in healthcare applications. Futur Gener Comput Syst 43:149–160
38. Zhao Q, Hu X, Lin J, Deng X, Li H (2019) A novel short-term blood pressure prediction model based on lstm. In: AIP Conference Proceedings, vol 2058. AIP Publishing LLC, p 020003
39. Zhou P, Shi W, Tian J, Qi Z, Li B, Hao H, Xu B (2016) Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th annual meeting of the association for computational linguistics (volume 2: Short papers), pp 207–212
40. Hager Ahmed, Eman M.G. Younis, Abdeltawab Hendawi, Abdelmgeid A. Ali, (2020) Heart disease identification from patients' social posts, machine learning solution on Spark. Future Generation Computer Systems 111:714–722