IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 28, 2020, accepted June 3, 2020, date of publication June 15, 2020, date of current version June 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3002346

Medicine Expenditure Prediction via a Variance-Based Generative Adversarial Network

SHRUTI KAUSHIK[1], (Member, IEEE), ABHINAV CHOUDHURY[1], SAYEE NATARAJAN[2], LARRY A. PICKETT, JR.[2], AND VARUN DUTT[1], (Senior Member, IEEE)

[1]Applied Cognitive Science Laboratory, Indian Institute of Technology Mandi, Kamand 175075, India
[2]RxDataScience, Inc., Durham, NC 27709, USA

Corresponding author: Shruti Kaushik (shrutikaushik15@gmail.com)

The work of Varun Dutt was supported under Grant IITM/CONS/RxDSI/VD/33.

ABSTRACT Machine learning (ML) offers a wide range of techniques to predict medicine expenditures using historical expenditures data as well as other healthcare variables. For example, researchers have developed multilayer perceptron (MLP), long short-term memory (LSTM), and convolutional neural network (CNN) models for predicting healthcare outcomes. However, recently proposed generative approaches (e.g., generative adversarial networks; GANs) are yet to be explored for time-series prediction of medicine-related expenditures. The primary objective of this research was to develop and test a generative adversarial network model (called ''variance-based GAN or V-GAN'') that specifically minimizes the difference in variance between model and actual data during model training. For our model development, we used patient expenditure data of a popular pain medication in the US. In the V-GAN model, we used an LSTM model as a generator network and a CNN model or an MLP model as a discriminator network. The V-GAN model's performance was compared with other GAN variants and ML models proposed in prior research such as linear regression (LR), gradient boosting regression (GBR), MLP, and LSTM. Results revealed that the V-GAN model using an LSTM generator and a CNN discriminator outperformed other GAN-based prediction models, as well as the LR, GBR, MLP, and LSTM models in correctly predicting medicine expenditures of patients. Through this research, we highlight the utility of developing GAN-based architectures involving variance minimization for predicting patient-related expenditures in the healthcare domain.

INDEX TERMS Generative adversarial network, long short-term memory, medicine expenditures, multilayer perceptron, regression, time-series prediction, variance minimization.

I. INTRODUCTION

According to the Centers for Medicare and Medicaid Services, the National Health Expenditure in the United States reached $3.2 trillion in 2015, which makes it $9,990 per person per year [1]. Such huge expenditures on healthcare may not lead to affordable healthcare to patients [2]. Therefore, it is crucial to predict the likely future patient-related expenditures to help patients better manage their huge healthcare costs. Predicting the healthcare expenditure may also be useful for various other stakeholders that include drug manufacturers, health insurers, pharmacies, and hospitals. Here, developing accurate healthcare expenditure models may help patients to choose appropriate insurance plans and may help healthcare delivery systems in better business planning [3].

The associate editor coordinating the review of this manuscript and approving it for publication was Shiping Wen.

Healthcare datasets mostly contain hundreds of variables such as demographic information, health-plan related information, medicine purchase information, diagnoses, and procedure codes [4]. Therefore, choosing the right set of healthcare variables and predicting the healthcare expenditures accurately is a challenging problem. Prior researchers have performed healthcare expenditure predictions by developing regression [5]–[7] and classification techniques [8], [9]. Most of the prior studies have used classical data mining approaches such as decision tree [10], random forest regression [11], multiple linear regression [11], clustering [8], bagging [10], and gradient boosting [10] for healthcare expenditure predictions. Here, some of the prior studies have relied upon the specific patient population (which may lead to non-generic results), used uncorrelated healthcare variables, or variable with very limited predictive accuracy [12]–[14].

Some of the recent studies in the healthcare domain have addressed prediction problems as time-series forecasting

problems [15]–[18]. Here, researchers have shown that statistical time-series methods like autoregressive integrated moving average (ARIMA) models do not work for predicting healthcare data [18]. A likely reason is the pre-assumption of linearity in the underlying time-series by these methods [19]. Since time-series variables may depict non-stationary and non-linear behavior, neural networks may be considered as a preferred choice for time-series predictions in practical scenarios [19]. That is because neural networks, with different assumptions on activation functions and hidden layers, may account for the non-stationary and non-linear behavior in time-series variables [19]. Recently, certain neural network architectures such as multilayer perceptron (MLP), long short-term memory (LSTM), and convolutional neural network (CNN) have been proposed for predicting time-series involving healthcare expenditures with some success [15]–[18].

Furthermore, a class of generative neural network models called generative adversarial network (GAN) has been proposed in the literature [20]. GANs are trained as a min-max game, where a discriminative neural network learns to distinguish whether the supplied data instance is real or fake, and a generative neural network learns to confuse the discriminator by generating high-quality fake data [20], [21]. One seeks the convergence of the combined generator-discriminator model by minimizing a chosen loss function [21].

Mostly, binary cross-entropy (also called as log loss) has been used as the loss function to train GAN models [21]. However, the binary cross-entropy loss function may not provide a sufficient gradient for the generator model to learn as quickly as the discriminator model [20]. Therefore, alternate loss functions like the least-square loss [22] and the Wasserstein loss [23] have been proposed in the literature. Recent research has shown that the least-square loss may lead to the problem of vanishing gradients when updating the generator model [22] and the Wasserstein loss (calculated as the distance between two probability distributions in terms of the cost of turning one distribution into another) shows better properties of convergence compared to the least-square loss [23].

By relying upon different loss functions for the generator and discriminator models, GANs have been successfully developed for a wide range of applications such as semantic segmentation [24], image inpainting [25], stock market prediction [26], and video prediction [27], [28]. However, to the best of authors' knowledge, the development of GANs and its variants for performing time-series prediction of future healthcare expenditures has not yet been explored in literature. We overcome this literature gap in this research by proposing a novel GAN architecture called variance-based GAN (or V-GAN) for the time-series predictions of healthcare expenditures.

Different from prior studies, we introduce a variance term in the V-GAN's loss function that explicitly minimizes the difference in the variance between patient data and model data during model training. Thus, in the new V-GAN model, we train the generator network using a different and novel combination of loss functions, which includes root mean square error (RMSE) loss, binary cross-entropy loss, variance loss, and their combinations. For the discriminator network in the new V-GAN model, we experiment with binary cross-entropy loss function, Wasserstein distance loss function [23], and a novel combination of both these loss functions.

For comparing the performance of the V-GAN architecture, we use the following machine learning methods: an ordinary least square linear regression (LR) model [29], a gradient boosting regression (GBR) model [29], an MLP model [18], and an LSTM model [15]. Since the V-GAN model can learn the approximate distributions in training data, we expect the V-GAN model to be able to generate accurate time-series predictions of the healthcare expenditures better than other existing models like GANs, regression models, MLPs, and LSTMs.

In what follows, we first provide a brief review of related work involving healthcare expenditure prediction and GAN models. Then, we explain the data used in this research for model development. Next, we explain the methodology of the new V-GAN model, different GAN variants, and the LR, GBR, MLP, and LSTM models. Furthermore, we present our experimental results, and we conclude the paper by discussing the implications of this research and its possible extensions.

II. RELATED WORK

A. HEALTHCARE EXPENDITURE PREDICTION METHODS

Based on prior research, two categories of approaches have been proposed to predict healthcare expenditures: regression models and classification algorithms [5], [8]. The first category involves using classical regression approaches such as ordinary least square (OLS) linear regression to estimate total annual health expenditure of patients in insurance claims data [5], [6], [30], [31]. For example, Moran *et al.* compared generalized linear models and OLS to predict individual patient expenditures in intensive care units [30]. These researchers obtained optimal overall performance from both these models. Marquardt *et al.* used linear regression and regression trees to develop a data-mining framework for scalable prediction of healthcare expenditures [31]. In recent years, researchers have developed neural network architectures such as MLP, LSTM, and CNN models for predicting patients' expenditure data [15]–[18]. Researchers have also compared the neural architectures with statistical time-series methods (e.g., ARIMA model) and found that the neural network models perform better compared to statistical methods for predicting healthcare expenditures [15], [18].

The second category involves the use of classification algorithms, where patients are classified into different expenditure buckets/classes [8], [32], [33]. For example, Bertsimas *et al.* used classification trees and clustering algorithms to classify patients into five different

expenditure classes by using patients' expenditures and clinical information [8]. Lahiri and Agarwal performed expenditure predictions as a binary classification task to predict whether beneficiaries' inpatient claim amounts increased or not between 2008 and 2009 [32]. These researchers achieved good performance by using an ensemble of six different classification approaches, which included conditional inference tree, logistic regression, gradient boosting, neural networks, support vector machines, and naïve Bayes. Similarly, Guo *et al.* performed a predictive modeling approach to predict patients' transitions from one expenditure bucket to another expenditure bucket [33]. Beyond the classical approaches described above, generative adversarial networks (i.e., networks that use generative and discriminative models in a min-max game) could be developed for predicting healthcare expenditures.

B. GENERATIVE ADVERSARIAL NETWORKS

Generative adversarial networks (GANs) rely upon two neural networks (a generator G and a discriminator D) that are trained simultaneously in an adversarial manner [20]. The generator model generates the synthetic or fake samples (that can pass for real data) by estimating the data distribution, and the discriminator model estimates the probability that a sample came from the training data or the generator model. Aim of G's training process is to maximize the probability of D making a mistake. Both G and D are trained against a static adversary. When G is trained, D's values are kept constant and vice-versa [20], [23].

Goodfellow *et al.* used MLP models for training G and D [20]. However, recently, researchers have implemented G and D using an LSTM model [34] and a CNN model [35], respectively, for a number of applications [26], [36], [37]. For example, Nie *et al.* used adversarial training to train a convolutional network for generating computer tomography images (CT) given medical resonance (MR) images [36]. The experimental results showed that the proposed method was accurate and robust for predicting CT images from MR images and could be used for medical image synthesis tasks. Researchers have also proposed a speech enhancement generative adversarial network (SEGAN) architecture for speech enhancement using GAN frameworks [37]. SEGAN architecture worked end-to-end by learning from different speakers and noise conditions such that model parameters were shared across these speakers and different noise conditions. Experimental results showed that the SEGAN model was generalizable and required no explicit assumption about the raw data. Zhou *et al.* have proposed a GAN framework to forecast high-frequency stock market data [26]. The authors trained a GAN model using data provided by a trading software, where G was trained using an LSTM, and D was trained using a CNN model. Experimental results showed that the proposed framework could effectively improve the stock price's direction prediction accuracy and reduce forecast errors [26].

Although the literature has proposed models for healthcare expenditure prediction, to the best of authors' knowledge, this paper is the first to adopt a GAN approach for predicting patients' expenditures on medications. In this paper, we propose a new V-GAN model and compare it with other GAN variants, regression-based models such as LR, GBR, and neural network models such as MLP and LSTM to predict patient-related expenditures on a medication.

III. METHOD

A. DATA

A popular pain medication's purchase data from the Truven MarketScan dataset [4] was used for model evaluations in this paper. The selected pain medication was among the top-ten most prescribed pain medications in the US in 2015 [38].[1] The dataset (uploaded on the IEEE data port, DOI: 10.21227/k0mm-jb74) ranged between 2nd January 2011 and 15th April 2015 (1565 days). The dataset between 2nd January 2011 and 30th July 2014 (1306 days) was used for model training, and the dataset between 31st July 2014 and 15th April 2015 (259 days) was used for model testing. On average, each day, about 1,428 patients refilled the selected medication. The dataset was a multivariate time-series dataset consisting of 21 attributes, which included the daily average expenditures by patients for purchasing the medication. The 20 attributes provided information regarding the number of patients of a particular gender (male, female), the number of patients in a particular age group (0-17, 18-34, 35-44, 45-54, and 55-65), the number of patients from a US region (south, northeast, north-central, west, and unknown), the number of people in a certain health-plan (two types of health plans), and the number of patients belonging to different diagnoses and procedure codes (six ICD-9 codes) who consumed medicine on a particular day. These 6 ICD-9 codes were selected using the frequent pattern mining Apriori algorithm [39], and the selection procedure is reported in a separate publication [40]. The 21st attribute was the average expenditure per patient for the medicine on day t, and it was defined as per the following equation:

$$\textit{Daily Average Expenditure}_t = {}^{i_t}/_{j_t} \tag{1}$$

where i_t was the total amount spent on buying the medicine across all patients on day t and j_t was the total number of patients who refilled the medicine on day t. We checked the stationarity of all 21 attributes by performing the augmented dickey fuller (ADF) test [41]. The ADF test revealed the time-series of all 21 attributes to be stationary without any trend or seasonal patterns (the ADF statistics value for the 21st attribute, i.e., daily average expenditure $= -10.10, p < 0.05$). The time-series data had values across different scales for different attributes. Therefore, we standardized all 21 attributes in the time-series before training different models [42].

[1]The name of the pain medication has not been disclosed due to a non-disclosure agreement.

Next, the daily average medicine expenditure was used to compute the weekly average expenditure. For computing the weekly average medicine expenditure, the daily average medicine expenditure was summed over a 7-day block. This resulted in the weekly average expenditure across 186 blocks of training data (1306 days were used in training), and 37 blocks of test data (259 days were used in testing). Fig. 1 shows the weekly average expenditure (in USD per patient) of the 21st attribute over weekly blocks. The weekly average expenditure (per block) was used to evaluate different models. We used the Root Mean Square Error (RMSE) to evaluate the performance of different models. The RMSE was computed between model predictions and real data at the block-level.

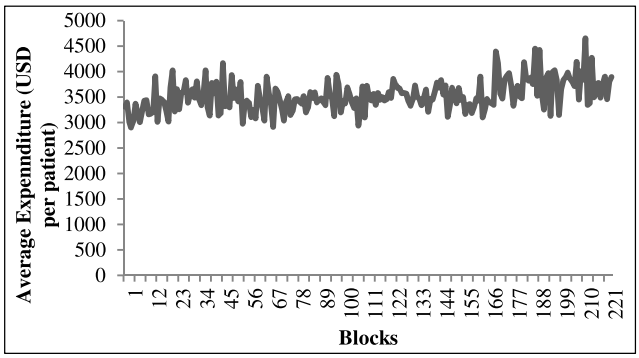


FIGURE 1. The weekly average expenditure (in USD per patient) over blocks. Each block is a 7-day period.

B. PREDICTION MODEL

In this research, we developed a V-GAN model for generating time-series predictions about patient-related expenditures. Since long short-term memory (LSTM) models are widely used for time-series prediction problems [15], we chose LSTM as a generator model (G) to generate predictions based on an input noise. The task of a discriminator model (D) is to estimate the probability of whether a sequence comes from real samples or the generated (fake) samples. The D works as a classifier that has to correctly classify the input samples as real or fake. Since convolutional neural network (CNN) models are mostly used for classification-based tasks [43], [44], they have been used as the D. As the CNN model is used as the D in prior literature for implementing GAN [26], we chose a CNN model as one of the D models in this research. Additionally, we also implemented the V-GAN model with an MLP as the D (since an MLP was used as the D by Goodfellow *et al.* [20] in the original GAN paper). It is important to note that the architectures of G and D could be adjusted based on the specific application and could be fine-tuned based on underlying time-series to enhance predictive performance.

For developing the V-GAN architecture (shown in Fig. 2(A)) for time-series prediction, the G (an LSTM) received an input dimension of 1×21 from the Gaussian distribution. For training the G, we varied the number of hidden layers (1, 2, and 3), the number of neurons (8, 16, 32, and 64), activation function (ReLU, tanh, sigmoid, and leaky ReLU), and the dropout rate for each hidden layer (20% and 30%). For training the D, we used 1×21 dimension input from the real samples and 1×21 dimension input generated by the G. As shown in Fig. 2(A), we passed the Gaussian noise represented as $Z_1, Z_2, \ldots, Z_{21}$, an input dimension of 1×21 in a batch size of 64 to the G, which made $64 \times 1 \times 21$ as the input shape of the G. In the fully trained V-GAN model, the G contained one LSTM hidden layer with 32 neurons and a ReLU activation function [45]. It was followed by a dropout layer with a 20% dropout rate, and finally, a dense layer with neurons equal to the input data dimension, i.e., 21. Thus, the G's output shape was the same as the input shape, i.e., 1×21. From the G, we obtained the fake time-series of 21 dimensions represented as $\hat{y}_1, \hat{y}_{2,\ldots,}\hat{y}_{21}$ in Fig. 2(A). Next, the D was trained using both the real data and fake (generated) data.

When CNN was used as the D, we varied the same range of hyper-parameters as the G along with the number of filters (32, 64, or 128) and different kernel sizes. The D received the generated time-series represented as $\hat{y}_1, \hat{y}_2, \ldots, \hat{y}_{21}$ and actual time-series represented as $Y_1, Y_2, \ldots, Y_{21}$ in a batch size of 64. In the CNN-based D, the architecture was composed of two 2D convolution layers followed by two fully connected (FC) layers. The first convolution layer contained 32 filters of 32×4 kernel size, singe-stride, and the leaky ReLU (LReLU) activation function with slope $= 0.01$ [45]. Second, convolution layer contained 64 filters, 64×4 kernel size, single-stride, and LReLU activation function with slope $= 0.01$. Padding was done in both the convolution layers to keep the input and output shapes the same. Fig. 3 shows the complete convolution operation (inside the D model) in detail. As shown in Fig. 3(A), the first 2D convolution layer received input of shape $64 \times 1 \times 21 \times 1$, where 64 represented the batch size, 1 represented the height, 21 represented the width, and 1 represented the number of channels. For the 2D convolution operation, 32 filters with a kernel size of 32×4 and a single stride were applied. In order to match the kernel size (height and width), 31 rows and 3 columns were padded, making the new data height $= 32$, and data width $= 24$. The output shape of the first convolution layer was $64 \times 1 \times 21 \times 32$ (batch size $\times$ height $\times$ width $\times$ channels). The difference in the input and output shapes was just in the number of channels, as 32 filters were applied in the convolution operation. Fig. 3(B) shows the second convolution layer operation using 64 filters with a kernel size of 64×4 and a single stride. Similarly, to match the kernel size, 63 rows (assuming 32 above the actual data row and 31 below the actual data row) and 3 columns (assuming 1 column in front and 2 columns at the end) were padded to the data. The padding made the new data height $= 64$ and data width $= 24$. The output shape of the second convolution layer was $64 \times 1 \times 21 \times 64$ (batch size $\times$ height $\times$ width $\times$ channels). The output of the second convolution layer was

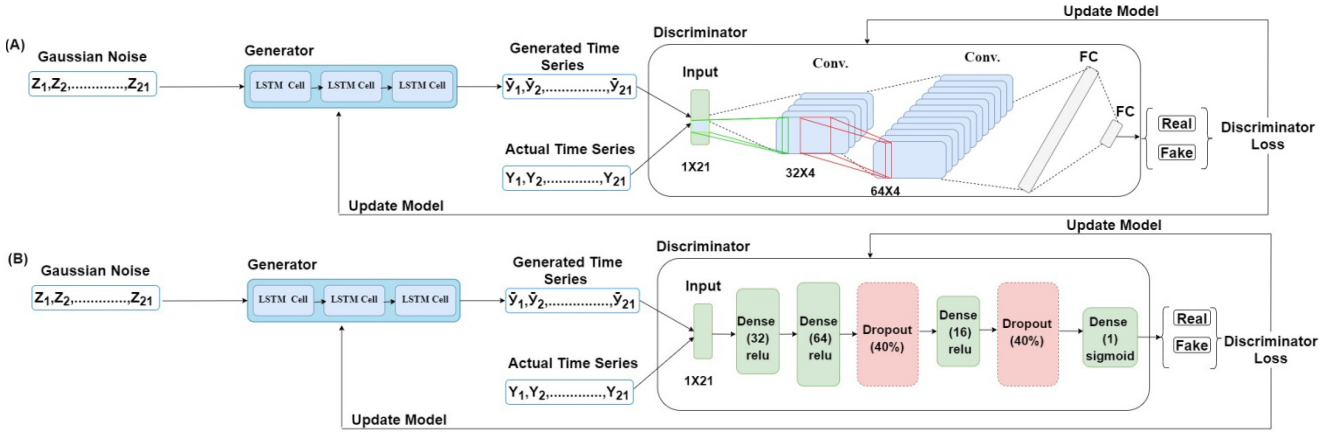


FIGURE 2. The V-GAN architecture. (A) The Generator model (G) was developed using an LSTM model, and the Discriminator model (D) was developed using a CNN model. The "Conv." and "FC" are abbreviations for the convolutional layers and a fully connected layer, respectively. (B) The Generator model (G) was developed using an LSTM model, and the Discriminator model (D) was developed using an MLP model.

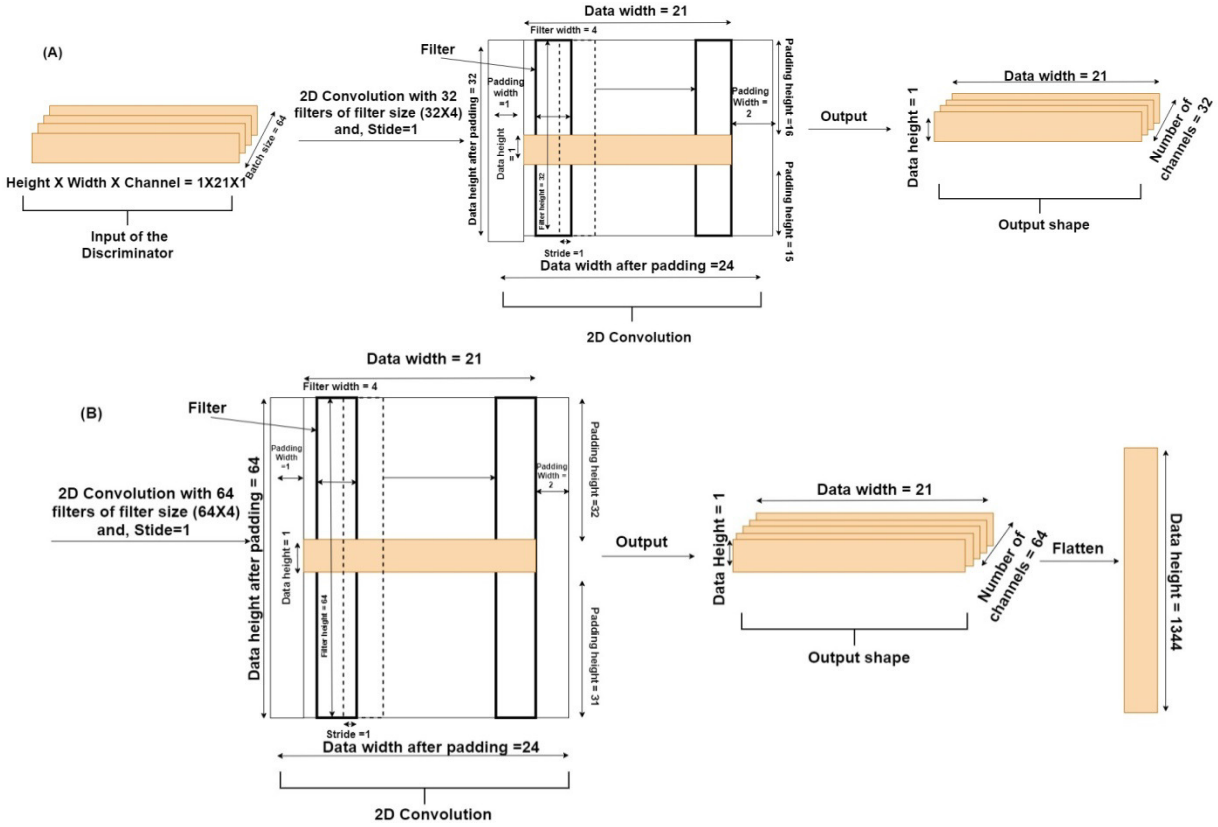


FIGURE 3. The convolution operation in the discriminator model. (A) Shows the input and output of the first 2D convolution layer, and (B) shows the input and output of the second 2D convolution layer.

flattened ($1 \times 21 \times 64 = 1344$; output shape of flatten layer = 64×1344) and passed to the first FC layer which contained 64 neurons and the LReLU activation function with slope = 0.01 (the output shape of first FC layer = 64×64). The first FC layer was followed by a second FC layer with 1 neuron and a sigmoid activation function (output shape of second FC layer = 64×1).

As stated above, we also developed the V-GAN architecture with an MLP as the D. Fig. 2(B) shows the architecture where V-GAN was implemented with LSTM as the G and MLP as the D. For developing this architecture, we kept the G (LSTM) architecture the same as described above in Fig. 2(A). For developing the MLP-based D architecture, we varied the number of hidden layers (1, 2, 3, and 4), the number of neurons (8, 16, 32, and 64), activation function (ReLU, tanh, sigmoid, and leaky ReLU), and the dropout rate for each hidden layer (20%, 30%, and 40%). As shown in Fig. 2(B), the final D was trained with one hidden layer

containing 32 neurons and the ReLU activation function. The second hidden layer contained 64 neurons and the ReLU activation function, followed by a dropout layer with a 40% dropout rate. The third hidden layer contained 16 neurons and the ReLU activation function, followed by a dropout layer with a 40% dropout rate. At last, the output layer contained 1 neuron with a sigmoid activation function.

The D's output indicated whether the input sample was real (supplied from the actual time-series) or fake (supplied from the generated time-series). The G aimed to confuse the D such that it could not guess correctly and would give a 50% probability of a sample being fake. Therefore, in the result section, we have also shown the percentage of real-like samples (i.e., fake samples reported as real) reported by the D. The complete model was trained for 20,000 epochs with 64 batch size and using the stochastic gradient descent (SGD) optimizer [46], initialized with 0.01 learning rate, and 0.9 momentum. Additionally, the GAN-based prediction models generated all the 21 attributes on the daily-level. However, the performance of all models was compared based on the predictions obtained for the average daily expenditure, i.e., the 21st attribute.

C. DIFFERENT LOSS FUNCTIONS

In this section, different loss functions used across different models, including the V-GAN model, are discussed.

1) ADVERSARIAL LOSS (L_A)

In the original GAN paper [20], the adversarial loss was used to evaluate the distance between two probability distributions. Adversarial loss is derived from the cross-entropy between the real and generated distributions, defined as L_A in (2) below:

$$L_A\left(y,\hat{y}\right) = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(p\left(y_i\right)) + \hat{y}_i \log(p\left(\hat{y}_i\right)) \quad (2)$$

where y_i represents real data for an attribute, $\hat{y}_i$ represents predicted/generated output of the same attribute, and N is the total number of generated points (which is equal to the number of points in a batch). The generator model could only affect the second term in distance measure as it reflected the distribution of the fake data. Therefore, during the generator's training, the model dropped the first term in (2), which reflected the distribution of the real data. Whereas, the discriminator model was trained using both the generated samples and the real samples. Thus, both terms in (2) were present in the training of the discriminator model.

2) VARIANCE LOSS (L_B)

Variance is used to measure the deviation from the mean. The variance loss (L_B) minimized explicitly the squared difference of the variances between the generated data and the actual data as defined in the following equation:

$$L_B\left(y,\hat{y}\right) = \left(\sigma\left(y\right) - \sigma\left(\hat{y}\right)\right)^2 \quad (3)$$

where σ signifies the variance in the data, which is calculated as the average of the squared differences from the mean. In other words, we calculated the sum of the squared distances of each term in the distribution from the mean, divided by the number of terms (which is equal to the batch size) in the distribution.

3) FORECAST ERROR LOSS (L_C)

In order to improve the forecasting performance of time-series models, researchers have proposed the forecast error loss or the RMSE loss (L_C) [26]. The idea behind using forecast error loss function is to bring predicted data closer to the actual data. Forecast error is defined as per the following equation:

$$L_C\left(y,\hat{y}\right) = \sqrt{\frac{1}{N}\left(y-\hat{y}\right)^2} \quad (4)$$

where y and $\hat{y}$ represent the real and predicted data, respectively. N represents the total number of points in a batch.

4) WASSERSTEIN DISTANCE LOSS (L_W)

Wasserstein distance is a loss function that measures the distance between the data distribution observed in the training samples and the distribution observed in the generated samples [23]. Arjovsky *et al.* have shown that training the generator may seek a minimization of the distance between the distribution of the data observed in the training dataset and the distribution observed in generated samples for better convergence and stable training [23]. The Wasserstein distance loss function is a measure of the distance between two probability distributions over a region D. The Wasserstein distance between two distributions X and Y is defined as the minimum amounts of work done to match X and Y, normalized by the total weight of the lighter distribution. Assume that the distribution X has m clusters with $X = (x_1, w_{x1}), (x_2, w_{x2}), \ldots, (x_m, w_{xm})$, where x_i is the cluster representative and w_{xi} is the weight of the cluster. Similarly, another distribution $Y = \left(y_1, w_{y1}\right), \left(y_2, w_{y2}\right), \ldots, \left(y_n, w_{yn}\right)$ has n clusters. Let $D = [d_{ij}]$ be the ground distance between clusters x_i and y_j. The objective is to find a flow $F = [f_{i,j}]$, where $f_{i,j}$ is the flow between x_i and y_j, which minimizes the overall work as per the following equation:

$$work = min \sum_{i=1}^{m}\sum_{j=1}^{n} f_{i,j} d_{i,j} \quad (5)$$

subjected to the constraints:

$$\begin{aligned} & f_{i,j} \geq 0, \quad 1 \leq i \leq m,\ 1 \leq j \leq n; \\ & \sum_{j=1}^{n} f_{i,j} \leq w_{xi}, \quad 1 \leq i \leq n; \\ & \sum_{i=1}^{m} f_{i,j} \leq w_{yj}, \quad 1 \leq j \leq m; \\ & \sum_{i=1}^{m}\sum_{j=1}^{n} f_{i,j} = \min \sum_{i=1}^{m} w_{xi}, \quad \sum_{j=1}^{n} w_{yj} \end{aligned} \quad (6)$$

The total flow F is found by solving this linear optimization problem. The Wasserstein distance $W(X, Y)$ is defined as the work normalized by the total flow as per the following equation:

$$W(X, Y) = \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j} d_{i,j}}{\sum_{i=1}^{m} \sum_{j=1}^{n} f_{i,j}} \quad (7)$$

D. DIFFERENT GAN MODELS

In this research, we developed GAN-based time-series prediction models using novel combinations of above-defined loss functions. Table 1 shows the generator and discriminator models' loss functions across different GAN models (including the new V-GAN model).

TABLE 1. Different GAN models.

Model	Generator Model's Loss Function	Discriminator Model's Loss Function
Adversarial GAN (A-GAN)	L_A[1]	L_A
Variance GAN (V-GAN)	$L_A + L_B$[2]	L_A
Forecast Error GAN (FE-GAN)	$L_A + L_C$[3]	L_A
Variance and Forecast Error GAN (VFE-GAN)	$L_A + L_B + L_C$	L_A
Variance and Wasserstein Distance1 GAN (VW1-GAN)	$L_A + L_B$	L_W[4]
Variance and Wasserstein Distance 2 GAN (VW2-GAN)	$L_A + L_B$	$L_A + L_W$

[1]L_A = Binary cross-entropy, [2]L_B = Variance loss, [3]L_C = Root mean square error, [4]L_W = Wasserstein distance

Adversarial GAN: We trained Adversarial GAN (A-GAN) using the adversarial loss function (L_A) for both the G and the D (as shown in Table 1). The adversarial learning helps the G to confuse the D, and it helps the D to classify the input samples as real or fake [20]. Both the G and the D update their cost independently as updating the gradient of both models concurrently cannot guarantee convergence [47].

Variance GAN: The Variance GAN (V-GAN) was developed using a novel combination of L_A and variance loss function (L_B), i.e., $L_A + L_B$, to train the G. The D in V-GAN model was trained using L_A loss function. The reason for trying a combination of L_A and L_B loss functions was that minimizing adversarial loss alone may not guarantee satisfactory predictions. With just adversarial loss, the G could generate samples to confuse the D; however, those samples may not be close to the actual data. To ensure that the G produced samples that confused the D and the generated samples were close to the actual data, we trained the G using the L_B loss function in addition to the L_A loss function. The L_B specifically minimized the difference in variance between generated and actual data. By developing V-GAN, we ensured that the model reduced the adversarial loss, and the L_B part of the loss function in the G ensured that the generated sample means did not deviate much from the actual data means.

Forecast Error GAN: The Forecast Error GAN (FE-GAN) architecture was developed using a novel combination of L_A and forecast error (L_C) loss function, i.e., $L_A + L_C$ for training the G and L_A was used to train the D. The reason for taking this combination was prior work in the time-series domain, where researchers used the forecast error loss function (L_C) [26]. The FE-GAN ensured that the G reduces the adversarial loss as well as the forecast error while generating the fake samples.

Variance and Forecast Error GAN: The Variance and Forecast Error GAN (VFE-GAN) was developed using a novel combination of L_A, L_B, and L_C loss functions ($L_A + L_B + L_C$) and the D was trained with L_A loss function (as shown in Table 1). By developing the VFE-GAN, we aimed at reducing the variance loss and forecast error along with the adversarial loss while generating the samples from G to confuse the D with realistic samples.

Variance and Wasserstein Distance-1 GAN: The Variance and Wasserstein Distance-1 GAN (VW1-GAN) was developed using a combination of L_A and L_B loss functions ($L_A + L_B$) to train the G. The D was trained using the Wasserstein distance loss function (L_W) (as shown in Table 1). The reason behind using a combination of L_A and L_B loss functions for the G was to investigate whether the V-GAN improved its performance by using a different loss function with convergence properties in the D [23]. The use of L_W in the D is also motivated from literature, where it may become challenging to obtain Nash equilibrium in a non-cooperative game when adversarial loss (L_A) is used in the D [47].

Variance and Wasserstein Distance-2 GAN: Building upon VW1-GAN, we also developed a Variance and Wasserstein Distance-2 GAN (VW2-GAN), where the G was trained using a combination of L_A and L_B loss functions ($L_A + L_B$) and the D was trained using a combination of L_A and L_W loss functions ($L_A + L_W$). The reasoning to use VW2-GAN was to explore improvements in the V-GAN by using a combination of L_A and L_W loss functions. Thus, the minimization of the adversarial loss and Wasserstein distance together in the D may help the G in return to produce better predictions.

IV. DIFFERENT MODELS FOR COMPARISON

To evaluate the performance of the proposed V-GAN model and other GAN-based prediction model variants, we developed an LR model [8], [29], a GBR model [29], an MLP model [18], and an LSTM model [15], [18] (see Table 2). All the developed models in Table 2 were evaluated in their ability to perform time-series predictions on the medicine expenditures' data. As shown in Table 2, based upon prior literature [29], we trained the LR and GBR models using the least square loss function. Also as shown in Table 2, the MLP and LSTM models were trained three times using the different combinations of loss functions: variance loss function (L_B), root mean square error loss function (RMSE; L_C), and a combination of L_B and L_C loss functions ($L_B + L_C$), separately. The idea behind using these loss functions for the MLP and LSTM models were to evaluate the variance minimization during model training against the

TABLE 2. Different models for comparison with GAN models.

Model	Loss Function
Linear Regression (LR)	Least Square
Gradient Boosting Regression (GBR)	Least Square
Multilayer Perceptron (MLP)	L_B[1]
	L_C[2]
	$L_B + L_C$
Long Short-Term Memory (LSTM)	L_B
	L_C
	$L_B + L_C$

[1]L_B = Variance loss, [2]L_C = Root mean square error

commonly used forecast error (RMSE) loss minimization (as was also done in the GAN variants, including V-GAN).

Model Training: We used the first 1306 days data to train the models and the last 259 days data for testing the models (the training and test size was kept the same as used in the development of the GAN-based prediction models). Different models were trained on the training dataset. Next, test data was used to evaluate the performance of the fully trained models. After getting the predictions for 259 days (test data), we summed these daily average expenditures on the block of 7 days to get the weekly average expenditures by patients on medicines. We used the block-level data to compute the value of the evaluation metric. The RMSE was used to compare the performance of these models against the prediction of GAN variants (including the V-GAN) on training and test data sets.

A. LR MODEL TRAINING

The ordinary least square linear regression (LR) is the most widely used approach for predictive modeling in healthcare [29]. Using the input variables as described above, we fitted an LR model using the least square loss function to predict patients' future expenditures.

B. GBR MODEL TRAINING

GBR is a successful ML technique used in research that generates an ensemble of decision trees to be used as a predictive model [29]. GBR learns different trees in an additive manner. Each round learns a new tree by optimizing the least square error (the objective function used in this research) between actual and predicted values. For training GBR, 100 decision trees were trained. A grid search was performed to find the optimum tree depth (varied between 2 to 14 in step size of 2) and learning rate (varied between 0.01 to 0.1 in step size of 0.01).

C. MLP AND LSTM MODEL TRAINING

For developing the MLP and LSTM models, we used the prior time-steps of all the 21 attributes together as an input (the number of prior time-steps or the lag value was determined by using the grid search approach) to predict the daily average expenditure (21^{st} attribute) at time step t. We performed a grid search for the following set of hyper-parameters during model training: hidden layers (1, 2, and 3), number of neurons in a layer (8, 16, and 32), batch size (4, 8, 16, and 20), number of epochs (8, 16, 32, 64, 128, 256, and 512), lag/look-back period (2 to 8, with a step size of 1), activation function (tanh, ReLU, and sigmoid), and dropout rate (20% and 30%). The models were trained to generate one-step-ahead time-series predictions of the daily average expenditure by patients. As defined above, we trained the MLP and LSTM models three times using three different loss functions: variance loss, root mean square error (RMSE) loss, and the combined loss of variance and RMSE.

V. RESULTS

Table 3 shows the GAN models' results obtained with different novel combinations of loss functions. We have reported two quantities: 1) the root mean square error (RMSE) on training and test data (on week-level data after summing the daily predictions in the block of 7 days); and, 2) the percentage of real-like samples reported by the discriminator model for different GAN models. As shown in Table 3, from the newly proposed V-GAN model, we obtained an RMSE of USD 330.08 on the training data and USD 321.08 on test data. The test RMSE was the best among all GAN variants. The discriminator model in V-GAN reported 58.97% of the fake/generated samples as real. Additionally, when V-GAN was implemented with MLP as D, V-GAN's performance decreased (as shown in Table 3).

TABLE 3. GAN model results.

GAN Model	Train RMSE	Test RMSE	% of Real-like Samples
A-GAN	304.15	496.77	13.73
V-GAN (with CNN as D)	330.08	**321.01**	58.97
V-GAN (with MLP as D)	337.27	392.71	59.67
FE-GAN	354.26	551.84	47.02
VFE-GAN	296.97	379.53	77.22
VW1-GAN	265.12	399.16	00.00
VW2-GAN	299.51	398.57	11.88

Bold text shows the results for the GAN model with the least RMSE on test data.

Furthermore, the training and test RMSEs of the V-GAN model showed that the model did not overfit the training data, where some amount of overfitting was present among other GAN models. Fig. 4 shows the average expenditure results on test data obtained from the V-GAN model. The x-axis in Fig. 4 depicts a block of 7 days, and the y-axis depicts the average expenditure by patients in USD. Additionally, we found that all GAN models in which variance was part of the loss function of the generator model performed better on test data than GAN models where variance was not a part of the loss function. Moreover, the discriminator model

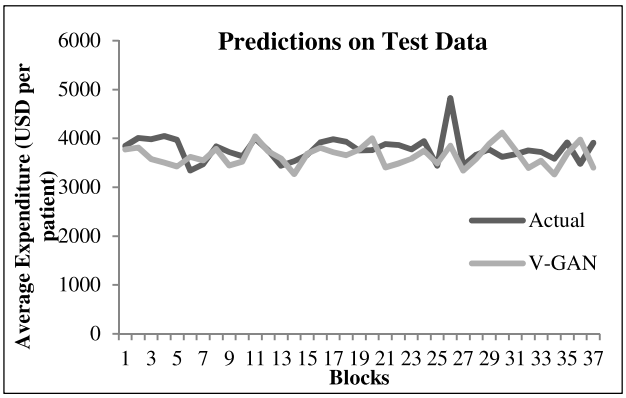


FIGURE 4. Prediction results on test data using V-GAN architecture.

in VW1-GAN reported 0% real-like samples, which meant that the generator model could not confuse the discriminator model with only L_W as the discriminator's loss function.

Next, we compared the ability of the best performing V-GAN model with other ML models. Table 4 shows the results of the different models (LR, GBR, MLP, and LSTM) compared to the V-GAN model for predicting patients' expenditures using different loss functions.

TABLE 4. Comparison of different models with V-GAN.

Model	Loss	Train RMSE	Test RMSE
LR	Least Square	256.16	361.08
GBR	Least Square	255.65	362.53
LSTM	L_B[2]	269.22	330.86
	L_C[3]	181.99	338.04
	$L_B + L_C$	299.51	353.41
MLP	L_B	471.37	335.45
	L_C	180.31	380.38
	$L_B + L_C$	545.15	438.51
V-GAN (with CNN as D)	L_A[1] + L_B for G L_A for D	330.08	321.01

[1]L_A = Binary cross-entropy, [2]L_B = Variance loss, [3]L_C = Root mean square error

As shown in Table 4, the MLP model performed the best in training data; however, the V-GAN model outperformed all other models in correctly predicting the future expenditures in test data. Furthermore, we obtained better performance on test data when the variance loss function was used to train the MLP and LSTM models compared to other loss functions. Among the four ML models, the LSTM performed the best on test data, followed by the MLP, LR, and GBR models. The final LSTM model was trained with a 2-lag period, 128 epochs, 8 batch size, ReLU activation function, and Adam optimizer. The LSTM model contained one hidden layer with 16 neurons, one dropout layer with 20% dropout, and an output layer with 1 neuron. Similarly, the final MLP model was trained with a 2-lag period, 256 epochs, 4 batch size, ReLU activation function, and Adam optimizer. The MLP model contained one hidden layer with 8 neurons, one dropout layer with 20% dropout, and then an output layer with 1 neuron. The final GBR model contained 100 decision tree estimators with 12 as the maximum tree depth and 0.1 as the value of the learning rate. Moreover, as shown in Table 3 and 4, the V-GAN model outperformed other GAN-based prediction models and different ML models used in prior research for correctly predicting the medicine expenditures of patients on test data.

VI. DISCUSSION AND CONCLUSION

The primary objective of this research was to evaluate the potential of generative adversarial networks (GANs) as a time-series model for predicting patients' expenditures on medications. In this research, we experimented with different combinations of loss functions to train the generator and discriminator models in a GAN and proposed a novel variance-based GAN (V-GAN) architecture, which minimized the difference in variance between model and actual data explicitly. The generator model (G) in V-GAN was trained using the binary cross-entropy and the variance loss function, where the discriminator model (D) was trained using the binary cross-entropy loss function. This research systematically evaluated the use of different loss functions in GAN's training for generating time-series dataset related to patients' expenditures on a medication. Moreover, the proposed V-GAN model's performance was also compared against a linear regression (LR), a gradient boosting regression (GBR), a multilayer perceptron (MLP), and a long short-term memory (LSTM) model for predicting the multivariate time-series dataset.

First, we found that the proposed V-GAN model outperformed all the other GAN-based models (developed with different loss functions) for predicting patients' average expenditure on a weekly level. A likely reason for this result is that reducing the difference in variance between model and patients' data during model training helped obtain a lower RMSE on test data. Next, we found that the V-GAN model outperformed LR, GBR, MLP, and LSTM models in this research on test data. Additionally, we obtained better performance from the MLP and LSTM models trained with the variance loss function compared to when these models were trained with the other loss functions. Therefore, we found a consistent behavior of the time-series prediction models, which implies that variance minimization for model and actual data during training helps generate better predictions. Furthermore, among LR, GBR, MLP, and LSTM models, the LSTM outperformed others. LSTM's performance was followed by MLP, LR, and GBR. A likely reason for this result could be that the LSTM model can maintain memory states across sequences and produce superior performance for time-series predictions [34]. Overall, we conclude that

GAN-based prediction models that focus on variance minimization can be developed in the healthcare domain and in other applications for generating time-series datasets. Additionally, developing LSTM based time-series models where loss function aims at reducing the variance difference between model and actual data might be helpful in general.

Second, we found that the Wasserstein distance (L_W) as a discriminator model's loss function in the VW1-GAN model did not help the model generate correct predictions. This result is in contrast to prior results on image datasets, where the use of Wasserstein distance loss function resulted in better performance and stable training [23]. However, in this research, when L_W was used as a discriminator model's loss function, we experienced mode collapse and 0% real-like samples as a result reported by the discriminator. Mode collapse is a situation where the generator generates a limited diversity of samples, or even the same sample, regardless of the input [48]. This happens when the generator fails to model the distribution of the training data well enough. In the case of VW1-GAN, we found that after certain epochs, the model stopped learning, and it was generating the same samples again and again. In general, the generator is always trying to find the one output that seems most plausible to the discriminator. If the generator starts producing the same output (or a small set of outputs) over and over again, the discriminator's best strategy is to learn always to reject that output. However, if the next generation of discriminator gets stuck in a local minimum and does not find the best strategy, then it is too easy for the next generator iteration to find the most plausible output for the current discriminator. Thus, in the VW1-GAN model, each iteration of the generator was over-optimized for a particular discriminator, and the discriminator never managed to find its way out of the trap. As a result, the generator rotated through a small set of output types and encountered mode-collapse. Also, using a combination of L_A and L_W loss functions for the discriminator did not help much in generating real-like samples, and we obtained only 11% real-like samples.

Next, we also experienced a minimal amount of overfitting from the V-GAN model (as shown in the train and test RMSE results reported in Table 3) compared to the other models. Additionally, when the variance loss function was present in the generator across all the GAN-based prediction models, the amount of overfitting reduced compared to when the variance loss function was absent. Moreover, for MLP and LSTM models, the variance loss function helped these models reduce the overfitting (as shown in the train and test RMSE results reported in Table 4) compared to the forecast error loss function. Therefore, the reduction in the difference in variance between model and actual data is in general useful for time-series forecasting models.

From the above results, we imply that the proposed V-GAN model can be utilized as a prediction model to generate/predict time-series datasets in the healthcare domain. The advantage of using V-GAN over existing models in this domain is that the GAN-based models generate future data that will have similar distribution as the actual patients' data. Whereas, methods like regression, bagging, and boosting do not care about the underlying data distributions while predicting future outcomes. Moreover, it is clear from the experiment shown in this research that the V-GAN model has the potential to produce robust and accurate results compared to other popular time-series models such as LSTMs. That is because the V-GAN model minimizes the difference in variance between model and actual data by reducing the gap between generated values and the mean value of the distribution. Additionally, GAN is an unsupervised learning approach. Therefore, we do not need to worry about providing labels or the lag values (i.e., values on prior time-steps) in the V-GAN model, as we need to train other existing time-series prediction models.

A drawback of the proposed method is model training: GAN-based prediction models are harder to train compared to other neural network models like MLP and LSTM. Furthermore, the GAN models may fail to model a multimodal probability distribution of data and can encounter mode collapse. Additionally, GAN models may experience slow convergence due to the internal covariate shift [48]. The internal covariate shift occurs when there is a change in input distribution to the network [48]. Due to changes in the input distribution, hidden layers may try to learn to adapt to the new distribution, which slows down the training process. If the training process slows down, it takes a long time to converge to a global minimum. Batch normalization technique may be one solution to avoid this problem during GAN training [48]. Though there are certain challenges associated with GAN models, these methods have the potential to generate accurate predictions.

In this research, we predicted the patients' expenditure on medication, and our results are likely to hold for other patient-related time-series variables. Therefore, we believe that the V-GAN framework could be used for other medications after some fine-tuning of the proposed structure of generator and discriminator models. Developing GAN-based prediction models may be beneficial, where data is limited, and accuracy is the prime objective. The proposed models may help pharmaceutical companies optimize the medications' manufacturing process and other industries for better inventory management. Apart from the healthcare domain, the proposed method could be used to predict stock market data, weather prediction, earthquake prediction, and for several other applications. As part of our future research, it would be worthwhile to explore V-GAN and other GAN-based prediction models for predicting patients-related expenditures on other medications.

REFERENCES

[1] M. Hartman, A. B. Martin, N. Espinosa, A. Catlin, and National Health Expenditure Accounts Team, "National health care spending in 2016: Spending and enrollment growth slow after initial coverage expansions," *Health Affairs*, vol. 37, no. 1, pp. 150–160, Jan. 2018.

[2] J. Kane. (Oct. 22, 2012). *Health Costs: How the U.S. Compares With Other Countries*. [Online]. Available: https://www.pbs.org/newshour/health/health-costs-how-the-us-compares-with-other-countries

[3] L. A. Winters-Miner, *Seven Ways Predictive Analytics Can Improve Healthcare*. Amsterdam, The Netherlands: Elsevier, 2014. [Online]. Available: https://www.elsevier.com/connect/seven-ways-predictive-analytics-can-improve-healthcare

[4] E. Danielson, *Health Research Data for the Real World: The MarketScan Databases*. Ann Arbor, MI, USA: Truven Health Analytics, 2014.

[5] C. A. Powers, C. M. Meyer, M. C. Roebuck, and B. Vaziri, "Predictive modeling of total healthcare costs using pharmacy claims data: A comparison of alternative econometric cost modeling techniques," *Med. Care*, vol. 43, no. 11, pp. 1065–1072, Nov. 2005.

[6] A. S. Ash, R. P. Ellis, G. C. Pope, J. Z. Ayanian, D. W. Bates, H. Burstin, L. I. Iezzoni, E. MacKay, and W. Yu, "Using diagnoses to describe populations and predict costs," *Health Care Financing Rev.*, vol. 21, no. 3, pp. 7–28, 2000.

[7] Y. Zhao, A. S. Ash, R. P. Ellis, J. Z. Ayanian, G. C. Pope, B. Bowen, and L. Weyuker, "Predicting pharmacy costs and other medical costs using diagnoses and drug claims," *Med. Care*, vol. 43, no. 1, pp. 34–43, Jan. 2005.

[8] D. Bertsimas, M. V. Bjarnadóttir, M. A. Kane, J. C. Kryder, R. Pandey, S. Vempala, and G. Wang, "Algorithmic prediction of health-care costs," *Oper. Res.*, vol. 56, no. 6, pp. 1382–1392, Dec. 2008.

[9] J. W. Robinson, "Regression tree boosting to adjust health care cost predictions for diagnostic mix," *Health Services Res.*, vol. 43, no. 2, pp. 755–772, Apr. 2008.

[10] M. A. Morid, K. Kawamoto, T. Ault, J. Dorius, and S. Abdelrahman, "Supervised learning methods for predicting healthcare costs: Systematic literature review and empirical evaluation," in *Proc. AMIA Annu. Symp.*, 2017, pp. 1312–1321.

[11] S. Sushmita, S. Newman, J. Marquardt, P. Ram, V. Prasad, M. D. Cock, and A. Teredesai, "Population cost prediction on public healthcare datasets," in *Proc. 5th Int. Conf. Digit. Health (DH)*, Florence, Italy, May 2015, pp. 87–94.

[12] J. T. Pacala, C. Boult, C. Urdangarin, and D. McCaffrey, "Using self-reported data to predict expenditures for the health care of older people," *J. Amer. Geriatrics Soc.*, vol. 51, no. 5, pp. 609–614, May 2003.

[13] C. Bourey, W. Williams, E. E. Bernstein, and R. Stephenson, "Systematic review of structural interventions for intimate partner violence in low- and middle-income countries: Organizing evidence for prevention," *BMC Public Health*, vol. 15, no. 1, p. 1165, Nov. 2015, doi: 10.1186/s12889-015-2460-4.

[14] C. A. Huber, S. Schneeweiss, A. Signorell, and O. Reich, "Improved prediction of medical expenditures and health care utilization using an updated chronic disease score and claims data," *J. Clin. Epidemiol.*, vol. 66, no. 10, pp. 1118–1127, Oct. 2013.

[15] S. Kaushik, A. Choudhury, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "Using LSTMs for predicting patient's expenditure on medications," in *Proc. Int. Conf. Mach. Learn. Data Sci. (MLDS)*, New Delhi, India, Dec. 2017, pp. 120–127.

[16] S. Kaushik, A. Choudhury, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "Ensemble of multi-headed machine learning architectures for time-series forecasting of healthcare expenditures," in *Applications of Machine Learning*. Singapore: Springer, 2020, ch. 14, pp. 199–216.

[17] S. Kaushik, A. Choudhury, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "Evaluating single- and multi-headed neural architectures for time-series forecasting of healthcare expenditures," in *Computational Intelligence Theoretical Advances and Advanced Applications*, D. Bisht, Ed. Berlin, Germany: De Gruyter Publisher, 2020.

[18] S. Kaushik, A. Choudhury, P. K. Sheron, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "AI in healthcare: Time-series forecasting using statistical, neural, and ensemble architectures," *Frontiers Big Data*, vol. 3, p. 4, Mar. 2020, doi: 10.3389/fdata.2020.00004.

[19] G. P. Zhang, "Time series forecasting using a hybrid ARIMA and neural network model," *Neurocomputing*, vol. 50, pp. 159–175, Jan. 2003.

[20] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio "Generative adversarial nets," in *Proc. NIPS*, Montreal, QC, Canada, Dec. 2014, pp. 2672–2680.

[21] J. Brownlee. (Sep. 2, 2019). *A Gentle Introduction to Generative Adversarial Network Loss Functions*. [Online]. Available: https://machinelearningmastery.com/generative-adversarial-network-loss-functions/

[22] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, Z. Wang, and S. P. Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2794–2802.

[23] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," Dec. 2017, *arXiv:1701.07875*. [Online]. Available: http://arxiv.org/abs/1701.07875

[24] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," Nov. 2016, *arXiv:1611.08408*. [Online]. Available: http://arxiv.org/abs/1611.08408

[25] S. Iizuka, E. Simo-Serra, and H. Ishikawa, "Globally and locally consistent image completion," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–14, Jul. 2017.

[26] X. Zhou, Z. Pan, G. Hu, S. Tang, and C. Zhao, "Stock market prediction on high-frequency data using generative adversarial nets," *Math. Problems Eng.*, vol. 2018, Apr. 2018, Art. no. 4907423, doi: 10.1155/2018/4907423.

[27] M. Mathieu, C. Couprie, and Y. LeCun, "Deep multi-scale video prediction beyond mean square error," Feb. 2015, *arXiv:1511.05440*. [Online]. Available: http://arxiv.org/abs/1511.05440

[28] S. Wen, W. Liu, Y. Yang, T. Huang, and Z. Zeng, "Generating realistic videos from keyframes with concatenated GANs," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 8, pp. 2337–2348, Aug. 2019.

[29] C. Yang, C. Delcher, E. Shenkman, and S. Ranka, "Machine learning approaches for predicting high cost high need patient expenditures in health care," *Biomed. Eng. OnLine*, vol. 17, no. S1, Nov. 2018, Art. no. 131, doi: 10.1186/s12938-018-0568-3.

[30] J. L. Moran, P. J. Solomon, A. R. Peisach, and J. Martin, "New models for old questions: Generalized linear models for cost prediction," *J. Eval. Clin. Pract.*, vol. 13, no. 3, pp. 381–389, Jun. 2007.

[31] A. Marquardt, S. Newman, D. Hattarki, R. Srinivasan, S. Sushmita, P. Ram, V. Prasad, D. Hazel, A. Ramesh, M. D. Cock, and A. Teredesai, "HealthSCOPE: An interactive distributed data mining framework for scalable prediction of healthcare costs," in *Proc. IEEE Int. Conf. Data Mining Workshop*, Shenzhen, China, Dec. 2014, pp. 1227–1230.

[32] B. Lahiri and N. Agarwal, "Predicting healthcare expenditure increase for an individual from medicare data," in *Proc. ACM SIGKDD Workshop Health Inform.*, New York, NY, USA, Aug. 2014, pp. 1–10.

[33] X. Guo, W. Gandy, C. Coberley, J. Pope, E. Rula, and A. Wells, "Predicting health care cost transitions using a multidimensional adaptive prediction process," *Population Health Manage.*, vol. 18, no. 4, pp. 290–299, Aug. 2015.

[34] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[35] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[36] D. Nie, R. Trullo, J. Lian, C. Petitjean, S. Ruan, Q. Wang, and D. Shen, "Medical image synthesis with context-aware generative adversarial networks," in *Proc. MICCAI*, Montreal, QC, Canada. Cham, Switzerland: Springer, Sep. 2017, pp. 417–425.

[37] S. Pascual, A. Bonafonte, and J. Serrà, "SEGAN: Speech enhancement generative adversarial network," Jun. 2017, *arXiv:1703.09452*. [Online]. Available: http://arxiv.org/abs/1703.09452

[38] G. Scott. (Oct. 6, 2014). *Top 10 Painkillers in the US. MD Magazine*. [Online]. Available: https://www.mdmag.com/medical-news/top-10-painkillers-in-us

[39] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. VLDB*, vol. 1215, Sep. 1994, pp. 487–499.

[40] S. Kaushik, A. Choudhury, N. Dasgupta, S. Natarajan, L. A. Pickett, and V. Dutt, "Evaluating frequent-set mining approaches in machine-learning problems with several attributes: A case study in healthcare," in *Proc. MLDM*, New York, NY, USA. Cham, Switzerland: Springer, Jul. 2018, pp. 244–258.

[41] D. A. Dickey and W. A. Fuller, "Likelihood ratio statistics for autoregressive time series with a unit root," *Econometrica, J. Econ. Soc.*, vol. 49, no. 4, pp. 1057–1072, Jul. 1981.

[42] S. G. K. Patro and K. K. Sahu, "Normalization: A preprocessing stage," Mar. 2015, *arXiv:1503.06462*. [Online]. Available: http://arxiv.org/abs/1503.06462

[43] Z. Li, M. Dong, S. Wen, X. Hu, P. Zhou, and Z. Zeng, "CLU-CNNs: Object detection for medical images," *Neurocomputing*, vol. 350, pp. 53–59, Jul. 2019.

[44] M. Dong, S. Wen, Z. Zeng, Z. Yan, and T. Huang, "Sparse fully convolutional network for face labeling," *Neurocomputing*, vol. 331, pp. 465–472, Feb. 2019.

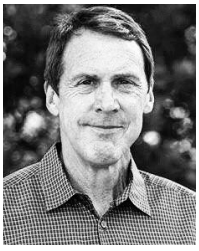
[45] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," Feb. 2018, *arXiv:1803.08375*. [Online]. Available: http://arxiv.org/abs/1803.08375

[46] L. Bottou, "Stochastic gradient descent tricks," in *Neural Networks: Tricks of the Trade*, 2nd ed. Berlin, Germany: Springer, 2012, pp. 421–436, doi: 10.1007/978-3-642-35289-8_25.
[47] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training GANs," in *Proc. NIPS*, Barcelona, Spain, Dec. 2016, pp. 2234–2242.
[48] K. Ahirwar. (2019). Generative adversarial networks projects. Packt, Birmingham, U.K. [Online]. Available: https://subscription.packtpub.com/book/big_data_and_business_intelligence/9781789136678

SHRUTI KAUSHIK (Member, IEEE) is currently pursuing the Ph.D. degree in computer science with IIT Mandi, India. Her research interests include machine learning, data mining and visualization, and deep learning. In her current research, she is working on healthcare datasets to develop time-series forecasting algorithms for predicting patient-related features.

ABHINAV CHOUDHURY is currently pursuing the Ph.D. degree in computer science with IIT Mandi, India. His research interests include social network analysis and reinforcement learning. In his current research, he is working on understanding diffusion of innovation inside social networks.

SAYEE NATARAJAN received the Bachelor of Engineering degree in computer science from the Government College of Technology, Coimbatore, and the M.B.A. degree from the School of Business, University of Connecticut. He is currently the Co-Founder and the Chief Technology Officer of RxDataScience Inc., a healthcare big data analytics startup based in RTP, NYC, and London. He has over 20 years of experience in pharmaceutical industry.



LARRY A. PICKETT, JR., received the bachelor's degree from the University of North Carolina at Chapel Hill and the M.B.A. degree from the University of North Carolina at Greensboro. He is currently the Co-Founder and the Chief Executive Officer of RxDataScience Inc., a healthcare big data analytics startup based in RTP, NYC, and London. He is also the former Vice President and the Chief Information Officer at Purdue Pharma LP, a pharmaceutical company located in Stamford, CT. In addition to the CIO role at Purdue, he held senior IT positions with Merck, Glaxo, and GE.

VARUN DUTT (Senior Member, IEEE) received the Ph.D. degree in engineering and Public Policy from Carnegie Mellon University, in 2011. He currently works as an Associate Professor with the School of Computing and Electrical Engineering and the School of Humanities and Social Sciences, IIT Mandi. He has been serving as an Associate Editor for *Frontiers in Cognitive Science* journal, a Review Editor for *Frontiers in Decision Neuroscience* journal, and a member of the Editorial Board for the *International Journal on Cyber Situational Awareness*.