future internet

MDPI

Article

Machine Learning-Based Patient Load Prediction and IoT Integrated Intelligent Patient Transfer Systems

Kambombo Mtonga [1,*], Santhi Kumaran [1], Chomora Mikeka [2] and Kayalvizhi Jayavel [3,*] and Jimmy Nsenga [1]

1 African Center of Excellence in Internet of Things (ACEIoT), College of Science and Technology, University of Rwanda, Kigali P.O. Box 3900, Rwanda; santhikr@yahoo.com (S.K.); jimmy.nsenga@gmail.com (J.N.)
2 Department of Physics, University of Malawi, Zomba P.O. Box 280, Malawi; cmikeka@cc.ac.mw
3 Department of Information Technology, School of Computing, SRM Institute of Science and Technology, Kattankulathur 603203, India
* Correspondence: kambombomtonga@gmail.com (K.M.); kayalvij@srmist.edu.in (K.J.)

Received: 2 July 2019; Accepted: 15 September 2019; Published: 12 November 2019

Abstract: A mismatch between staffing ratios and service demand leads to overcrowding of patients in waiting rooms of health centers. Overcrowding consequently leads to excessive patient waiting times, incomplete preventive service delivery and disgruntled medical staff. Worse, due to the limited patient load that a health center can handle, patients may leave the clinic before the medical examination is complete. It is true that as one health center may be struggling with an excessive patient load, another facility in the vicinity may have a low patient turn out. A centralized hospital management system, where hospitals are able to timely exchange patient load information would allow excess patient load from an overcrowded health center to be re-assigned in a timely way to the nearest health centers. In this paper, a machine learning-based patient load prediction model for forecasting future patient loads is proposed. Given current and historical patient load data as inputs, the model outputs future predicted patient loads. Furthermore, we propose re-assigning excess patient loads to nearby facilities that have minimal load as a way to control overcrowding and reduce the number of patients that leave health facilities without receiving medical care as a result of overcrowding. The re-assigning of patients will imply a need for transportation for the patient to move from one facility to another. To avoid putting a further strain on the already fragmented ambulatory services, we assume the existence of a scheduled bus system and propose an Internet of Things (IoT) integrated smart bus system. The developed IoT system can be tagged on buses and can be queried by patients through representation state transfer application program interfaces (APIs) to provide them with the position of the buses through web app or SMS relative to their origin and destination stop. The back end of the proposed system is based on message queue telemetry transport, which is lightweight, data efficient and scalable, unlike the traditionally used hypertext transfer protocol.

Keywords: deep convolutional neural networks; time series forecast; patient overcrowding; patient load prediction; smart transport; intelligent patient transfer

1. Introduction

Overcrowding of hospital waiting rooms by patients characterizes the outpatient departments (OPDs) of healthcare centers of developing countries [1,2]. It is true that most of the patients that visit health centers have varying health challenges and some may require prompt medical attention. However, due to overcrowding, these patients are made to endure a lengthy and painful wait for treatment [3,4]. In the developing world, the problem is aggravated by the fact that health facilities are few, scattered and under-staffed. In such countries, it is a common experience for patients visiting an OPD to return home without receiving medical attention. For example, data collected from the patients record book at one general OPD showed that, within a one month period, of the 2211 patients who walked-in into the OPD department, only 1870 (84.58%) patients were seen and 341 (15.42%) patients were not seen [5].

To deal with such excess demand, physicians may opt to work overtime or delegate some work to nurses. Consequently, this may cause the healthcare center to incur extra costs and/or provide reduced quality of service to the patients [6,7]. Reduced quality of service might impact patient's trust in the health facility and/or practitioners [8]. Overcrowding in hospitals also relates to patients' safety. Due to limited space, patients may be placed in inappropriate spaces, which can be a recipe for complications and fatalities. The challenge further result from the fact that, in the developing world, patients' hospital visits (especially to the OPD) are rarely scheduled, which makes it difficult to project daily demand for the services [9,10]. This leads to an inbalance between hospital staffing ratios and health service demand. It is the case in many situations that as one heath center is being pressed with high demand beyond its capacity, the facilities nearby may have few patients. However, there is no mechanism for a timely exchange of information about patient load between health facilities. This leads to patients in overcrowded health facilities having to endure a lengthy and painful wait for treatment which may not even be given.

Patients are usually transferred between health facilities and the purpose for such transfers is to maintain the continuity of medical care. Normally, these are patients that require specialized treatment or procedures and are transferred to facilities with such specialized equipment and personnel, i.e., the benefits of care available at another facility against the potential risks involved. The patient transfer process involves such key elements as the decision to transfer and notification, patient pre-transfer stabilization and preparation, the choice of appropriate mode of transfer (e.g., land or air transport), personnel to accompany the patient, equipment and monitoring required during the transfer and finally the documentation and handover of the patient at the receiving facility [11,12]. In each transfer these key elements are followed so as not to affect patient prognosis.

Since our focus is the outpatient department, we make the following assumptions: (1) Upon arrival, patients are triaged such that the critical patients are served first and then the least critical last. (2) Excess patient loads can only be transferred once, i.e., a patient can only be transferred once. As pointed out above, the patient transfer process starts with a decision to transfer the patient because of the exposure of the patient and the staff to additional risks and additional expenses for the guardians. A senior consultant level doctor is responsible for making the decision to transfer the patient . A patient's guardians are made aware of the benefits and risks involved during such a transfer. Written and informed consent of patients' relatives along with the reason(s) to transfer is mandatory before the transfer. In this work, the role of the senior doctor will be played by the central controller of the system. Since the system provides the patient with alternative healthcare service sources, it is up to the patient to go to the recommended facility or wait in the queue and risk returning home without getting medical care. Hence a patient's consent will be by virtually in accepting to go to the recommended facility. Since patients are triaged upon arrival (with the most critical served first), we assume that the patients being recommended for transfer are stable enough such that pre-transfer stabilization and preparation are not required. The choice of an appropriate mode of transport is key in this work and is covered in Section 6. The consideration of guardian, patient monitoring during transfer and documentation are beyond the scope of this work.

By intelligently re-assigning patients from congested facilities to less congested ones, overcrowding of OPDs can be managed in a timely way. Re-assigning excess patient loads reduces the number of patients that return to their homes without being attended to by physicians and consequently leads to improvement of the quality of the service demanded of medical staff being kept to a minimum. Patient flows in hospitals are usually not continuous and stable; rather, patient flows are complex and may change suddenly [13]. Hospitals experience two kinds of service demands: (1) Event driven demand, i.e., ambulatory arrivals during accidents and natural disasters, and (2) regular demand from the catchment area. However, a lack of real time patient flow information may result in some health facilities straining to satisfy the demand while at the same time facilities in the same vicinity may have minimal demand. While patients may be wary about being transferred to a hospital they know little or nothing about, such transfers may significantly reduce the amount of time a patient must wait to get medical care [14]. Attempts in the literature exist to predict patient traffic, but most of the work focuses on patient flow in the emergency department (ED) [15,16]. Furthermore, independent attempts have been made to predict and schedule traffic flow [17,18]. However, to our knowledge, no literature has explored the integration of the two processes using machine learning techniques.

In this paper, a deep learning-based patient load prediction model is proposed. The model aides in overcoming the effects of overcrowding in hospital waiting rooms, which results from a mismatch between hospital staffing ratios and the demand for healthcare services. Overcrowding of hospital waiting rooms leads to excessive patient waiting times, incomplete service delivery and unhappy medical staff. Worse, due to the limited patient loads that a health facility can handle, patients may leave the facility before the medical examination is complete. However, it is true that, as one health facility may be struggling with excessive patient load, another facility in the vicinity may have low patient turn out. In this work, we assume the existence of a hospital information management system, which enables timely sharing of excess patient load information among hospitals [19]. The proposed machine learning-based patient load prediction model takes current and historical patient load data as inputs and outputs future predicted patient load. Furthermore, in the case of excess patient loads, we propose to re-assign excess loads to nearby facilities that have a minimal load as a way to control overcrowding and reduce the number of patients that leave health facilities without receiving medical care.

The re-assigning of patients will imply a need for transportation for the patient to move from one facility to another. To avoid putting a further strain on the already fragmented ambulatory services, we assume the existence of a scheduled bus transport system which can support the timely movement of patients from the bus stop nearest to the source facility to the destination facility. Building on this assumption we propose an Internet of Things (IoT) integrated smart bus system. We develop an Arduino-based smart bus system that can be tagged on public buses and can be queried by patients through representation state transfer application program interfaces (APIs) to provide them with the position of the buses through web app or SMS relative to their origin and destination stop. The back end of the proposed system is based on message queue telemetry transport (MQTT), which is lightweight, data efficient and scalable, unlike the traditionally used hypertext transfer protocol (HTTP). To ensure the reliability of the smart transport system, our solution makes use of the real time location of the buses to compute the approximated time for the bus to reach a particular destination. By saving a bus's location data on the server together with corresponding timestamps, the system is able to estimate the arrival time of the bus to a particular bus stop. Alternatively, the arrival time can also be approximated using services like Google maps [20].

The Internet of Things (IoT) enables advanced services by interconnecting physical and virtual things based on existing and evolving inter-operable information and communication technologies. The IoT gives immediate access to information about physical objects and leads to innovative services with high efficiency and productivity [21]. Building on the power of IoT, we demonstrate its application in the transportation system by developing an IoT-based smart bus system. Generally, a smart bus system consist of four basic components, namely smart bus depots, smart bus stops, smart buses and interactive citizen interfaces (web portal based and smart phone app based). Each of these components are connected through the Internet. The smart bus depots, smart bus stops and smart buses consist of a number of heterogeneous wireless and embedded sensors. These sensor networks are connected to the city internet backbone through Wi-Fi hotspots in the bus stops, depots and inside buses. These intelligent autonomous devices attached to or embedded into the system senses the user requirements and interacts with them, shares information with other devices and takes decisions without any human intervention. Buses are widely used public transportation in many cities today. Normal buses can be converted into smart buses with the incorporation of intelligent sensors and IoT devices. Our contributions in this paper are three fold:

- Using a deep learning approach, we predict patient traffic flow.
- We combine the deep learning patient load prediction and hospital assignment using the predicted patient load as criterion to perform the intelligent hospital assignment.
- We develop an IoT-based smart bus system. We explore scalable and efficient techniques that allow the system to handle increased requests for data.

The remainder of this paper is organized as follows. In Section 2, a review of the relevant literature and machine learning structures is presented. In Section 3, we model our system architecture and training model. In Section 4, we propose a deep learning-based patient load prediction method. In Section 6, we present our proposed IoT-based smart bus system that allows the estimation of the time for the smart bus to arrive at a bus stop, or the time for it to reach a destination such that transferred patients do not experience unnecessary delays due to transport congestion. Finally, Section 7 concludes the paper.

2. Related Work

A lot of research efforts have been made to deal with the problem of overcrowding with regard to the ED of the hospital [22–25]. To deal with this problem, inter-hospital transfers have been proposed as a possible solution [26]. However, overcrowding is also a problem in the OPD and equally affects patient satisfaction, which is affected by the efficiency of the services rendered. Efficient provision of health services encompasses such issues as waiting time to consultation, duration of consultation, timely response to emergencies, quick drug dispensation, and timely and accurate diagnosis [27,28]. The challenge in dealing with excess demand results from the challenges in predicting patient flow in the OPD. Patient load forecasting involves predicting patient loads for future time periods. Depending on time horizons, patient load prediction can be categorized into: Long-range forecasting, medium-range forecasting, short-range forecasting and real-time or very short-term forecasting. Accurate patient flow predictions may aide in improving hospital management efficiency.

Traditional methods have been applied in forecasting hospital visits. In [29], Dan and Qualls explored the problem of predicting ED patient volume, length of stay and acuity. They studied five models, namely raw observations, moving averages, mean values with moving averages, seasonal indicators with moving averages and auto-regressive integrated moving averages (ARIMAs). It was discovered in this study that simpler models performed best. In [30], Rotstein et al. explore the problem of short-range forecasting of patient volume. A general linear model (GLM) is formulated that can be applied for short-range forecasting of patient volume. In [31], Batal et al. developed equations for predicting daily patient volumes via stepwise linear regression analysis. In [32], Reis and Mandl developed a trimmed mean seasonal model for the expected number of daily patient visits to an ED. In [33], Brillman et al. constructed a first order cyclical regression model with fixed-width sine and cosine harmonics as the seasonal component and a hierarchical model with a scalable Gaussian function as the seasonal component for ED daily respiratory chief complaints. In [34], Flottemesch et al. formulated a mathematical model for forecasting ED censuses. In [35], Boyle et al. investigated the performance of a general linear regression model formed with 11 dummy variables in forecasting monthly patient admissions. In [36], Au-Yeung et al. forecasted patient arrivals to an accident and ED via a structural time series model. In [37], Kam et al. studied the problem of predicting daily patient numbers for a regional medical center via the application of time series analysis. Capan et al. applied time series analyses; specifically, they applied best-fitting models of ARIMA and linear regression to various prediction models and compared the results using error statistics. Their work aimed at forecasting censuses in neonatal intensive care units (NICU) [38]. Other interesting studies relating to predicting patient flow in ED have been reported in [39–42]. While these traditional techniques are popular in forecasting hospital visits, they are not good at dealing with complexity in hospital visits data.

Artificial Intelligence techniques, e.g., artificial neural networks (ANNs), form another approach for hospital visit forecasting [43]. However, ANN is not as prevalent as the traditional methods in this field. In [44], Jones et al. explored the performance comparison of exponential smoothing, seasonal ARIMA, ANN and time series regression in daily ED patient volume prediction with linear regression. Their results show that the former four methods did not perform consistently in forecasting with a sample, although they all performed better in sample fitting. In [45], Aladag and Aladag modeled the number of outpatient visits by ANN using different activation functions. In [46], Xu et al. modeled daily patient arrivals at ED via ANN. Kottalanka et al. developed an artificial intelligence model based on back-propagation Neural Network for predicting patient inflow [47]. Clearly, more research is needed to demonstrate and harness the power of machine learning to improve the provision of healthy services, which can lead to improved quality of care and also efficient utilization of resources.

The patient flow data is basically time series data. In [48], a time series is defined as a vector $X = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \cdots, \mathbf{x}^{(n)}\}$, where each element $\mathbf{x}(t) \in \mathbb{R}^m$ pertaining to X is an array of m values $\{\mathbf{x}_1^{(t)}, \mathbf{x}_2^{(t)}, \cdots, \mathbf{x}_m^{(t)}\}$, with each of the m values corresponding to the input variables measured in the time series. Several traditional techniques of manipulating time series data including traditional ANNs have been exploited in the literature for applications in economics, engineering and medicine [49,50]. Deep learning, however, has shown an exceptional performance in both classification and forecasting applications [51–53]. For classification, the existing methods generally relied on the usage of domain specific features normally crafted manually by human experts. However, getting best features was a daunting task and the performance of the classifier was heavily dependent on their quality. The advantage of deep learning is in its ability to learn such features by itself, reducing the need for human experts [54,55].

Convolutional neural network (CNN) models have continued to gain popularity among the deep learning community in various domains [56]. Due to their efficiency with data that has a topological structure in the features space, CNN models have achieved the best results in computer vision [57]. They have also been applied successfully to sequential data such as sentences, time series and speech [58–63]. CNNs employ sets of shared weights across the whole input, which is efficient both statistically and computationally. CNNs are particularly interesting in domains associated with the processing of large amounts of data [64]. Since patient load data and traffic data can be categorized as big data, the application of deep learning-based solutions is suited to solve the combined problem of patient load predictions and intelligent inter-hospital patient transfers.

We point out that in recent times deep reinforcement learning (Deep-RL) has gained popularity. Deep learning basically models a scenario in which no direct interaction exist between the algorithm and the environment it is operating in. Basically, RL enables a feedback loop between the algorithm and the environment. It allows the algorithm to experience a dataset that varies with time as a result of the interaction with the surrounding environment. RL is applicable to scenarios that can be modeled by a Markov decision process (MDP). Recent works based on RL techniques have shown comparable performance against NNs. In [65], Negnevitsky et al. applied an adaptive neural fuzzy inference system (ANFIS) to study the problem of load forecasting in power systems. They further pointed out the difficulty of load forecasting in power systems resulting from the complexity of the power load series as it exhibits seasonality levels and the fact that power load series have various weather related exogenous variables. In [66], the authors applied principles of RL and game theory to develop an autonomous evacuation process to support distributed and efficient evacuation planning. Deep-RL methods results when deep NNs are used to approximate any of the components of RL, e.g., action-value and/or the policy functions [67]. A combination of NN with RL will enable solving even more complex problems.

Despite the wide range of successes, current state-of-the-art Deep-RL methods still face a number of significant drawbacks [68]. As the training of NNs requires huge amounts of data, Deep-RL demonstrates unsatisfying results in settings where data generation is expensive. Even in cases where interaction is nearly free (e.g., in simulated environments), Deep-RL algorithms tend to require excessive amounts of iterations, which raises their computational and wall-clock time cost. Furthermore, Deep-RL suffers from random initialization and hyperparameter sensitivity, and its optimization process is known to be uncomfortably unstable [69]. An especially embarrassing consequence of these Deep-RL features turned out to be the low reproducibility of empirical observations from different research groups [70]. The main reason for NN becoming so popular lies in its ability to learn complex and nonlinear relationships that are difficult to model with conventional techniques [71,72]. Hence our choice of deep learning is based on its many documented exciting achievements [73,74], which are a function of big data, powerful computation, new algorithmic techniques, mature software packages and architectures and strong financial support.

Deep Learning Architectures

Deep NNs are a part of the broad field of AI. AI is the science and engineering of creating intelligent machines that have the ability to achieve goals like humans do. Figure 1 shows the relationship of deep learning to the entire field of AI [75]. There exist different deep learning structures including the deep Boltzmann machines (DBMs) and deep convolutional neural networks (CNNs). These architectures can be modeled to help control patient flow systems. The chosen deep belief architecture (DBA) has L layers, including one input layer, one visible output layer and $(L-2)$ hidden layers. The units comprising each layer except for the input layer have their own weight value called bias. The units for two adjacent layers are connected with each other via weighted links, while no inner layer connection exists.

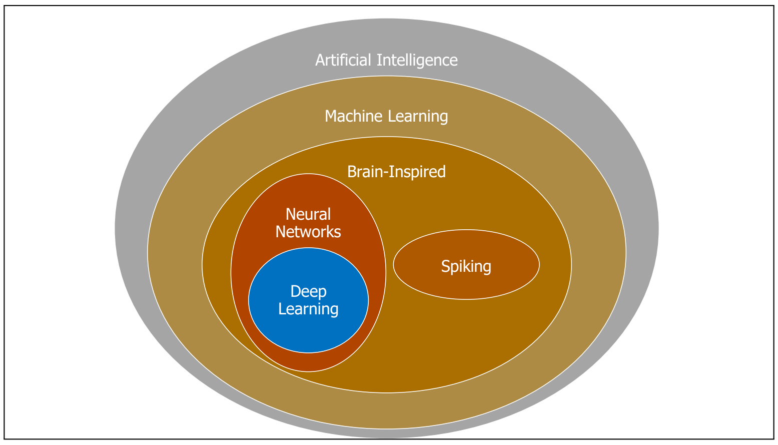


Figure 1. Relationship between deep learning and the rest of AI.

3. System Modeling

In this section we describe our system architecture and the training process. We start by presenting the notations used through out the remainder of this paper in Table 1.

Table 1. Notations.

Symbol	*Meaning*
G	A graph-a network of health facilities
E	Edges of a graph G
P	A set of patients
L	A set of hospitals in a particular region
M	Total number of hospitals in a particular region
R	Average number of patients in the catchment area of hospital L
w_{ij}	Weight of link between i and j
b_i	Bias of unit i
ξ	Learning rate
$f(\cdot)$	Activation function

We model the system as a graph $G = (P \cup L \cup C, E)$, where:

- C is the central controller. The central controller can be played by the big health facility within a defined catchment area.
- $L = \{l_1, l_2, \cdots, l_M\}$ is the set of local hospitals under the control of C.

Given that M is the number of hospitals under consideration, we define the following;
Let $p_m(t)$ be the number of patients at the mth hospital at time t. Then:

$$P(t) = \{p_1(t), p_2(t), \cdots, p_M(t)\}, \tag{1}$$

is the set of patients over the set of hospitals at time t.

Note that, here, M is the total number of hospitals located in a particular region with each hospital serving a population in its catchment area. If we let R be the average number of patients in the catchment area of L, i.e., the total catchment area covering the M hospitals, then $|L| = M \times R$. R is also time variant, since the population of a particular area varies with time. Each local hospital maintains a local database of patient information.

Define E to represent the edges in a graph G. Let an edge $e \in E$ in the graph represent the route connecting the two nearest hospitals. Then the weight, $w(e)$, represents the obstacles that will impact a patient's ability to get from one facility to another. These obstacles include distance, traffic congestion, transport cost and the physical terrain. Here we consider the shortest possible road connecting two facilities. In Figure 2, the arrows indicate information flow between the central controller and each of the facilities in its domain and the solid circular line defines a catchment area while the dotted circular line defines a network of health facilities under the control of the central controller. From now on we abbreviate central controller as Cc.

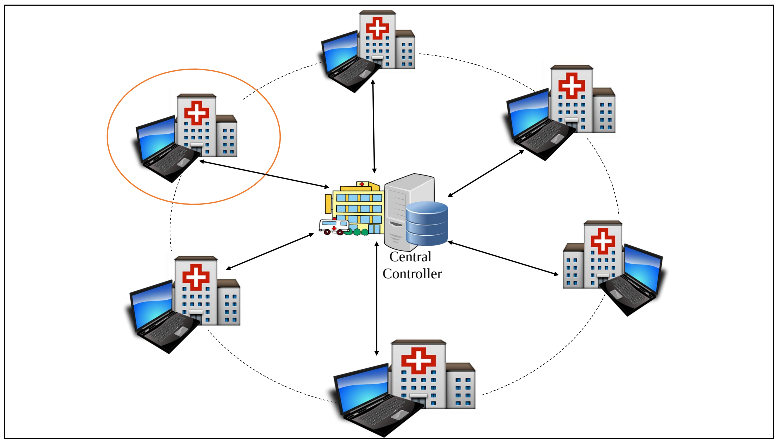


Figure 2. A centralized system architecture—the dotted circle defines a network of hospitals, whilst the solid circle defines a catchment area of a particular hospital.

Training the Model

Let the pair (x_{input}, y_{output}) represent the values of units in the input and output layers, respectively. Let w_{ij} represent the weight of the link between units i and j, and b_i represent the bias of unit i. Let w be the matrix of the weights of all links and b be the matrix of all the bias values; then the training of the deep belief architecture (DBA) comprises two steps, namely forward propagation and back propagation processes. Here, forward propagation is used for constructing the structure and activating the output, whilst back propagation is used for adapting the structure and fine-tuning the values of the weight and bias matrices. The forward propagation process can be modeled as a log-likelihood function and is given as:

$$l\left(w, b, x_{input}, y_{output}\right) = \sum_{t=1}^{m} \log\ p\left(v^{(t)}\right), \tag{2}$$

where $v^{(t)}$ denotes the tth training data. Here, the DBA training can be seen as a log-linear Markov random field (MRF). As such, $p\left(v^{(t)}\right)$ represents the probability of $\left(v^{(t)}\right)$. m is the total amount of training data.

The purpose of the training process is to minimize $l\left(w, b, x_{input}, y_{output}\right)$, in the back propagation process. Back propagation is an efficient way to compute the partial derivatives of the gradient. It is a computation derived from the chain rule of calculus, and it operates by passing values backwards through the network to compute how the loss is affected by each weight. The link weight w and the bias b are adjusted using the gradient descent method. We represent w and b as:

$$w = w + \xi \frac{\partial\left(w, b, x_{input}, y_{output}\right)}{\partial w}, \tag{3}$$

$$b = b + \xi \frac{\partial\left(w, b, x_{input}, y_{output}\right)}{\partial b}, \tag{4}$$

where ξ is the learning rate of the training process. The Gradient descent is a first-order optimization method, this means it uses only information of the first derivative of the error; hence, it can be used in combination with error back propagation. The challenge with this method is the difficulty of choosing the learning rate so as to get fast learning but at the same time avoid oscillation.

As the input layer increases (i.e., gets larger) and becomes less connected in high dimensions, the DBA structure becomes inefficient as it fails to capture the spatial features efficiently. The CNNs tend to be powerful in this regard. The convolution operation extracts features of the input, and the parameters of the convolution operation comprise a set of learnable filters. Let W_{l_1} denote the filters and the kth filter be represented by $W_k^{l_i}$, then the convolution operation outputs a feature map given as:

$$\begin{aligned} u_{i,j,k}^{(l_1)} &= \left(U^{l_1-1} * W_k^{l_1}\right)(i,j) + w_{bk}^{l_1} \\ &= \sum_{p=1}^{P} \sum_{m=1}^{M'} \sum_{n=1}^{N'} w_{m,n,p} a_{i+m,j+n,p}^{(l-1)} + w_{bk}^{(l1)} \\ a_{i,j,k}^{(l_1)} &= f\left(u_{i,j,k}^{l_1}\right), \end{aligned} \tag{5}$$

where $f(\cdot)$ is the activation function and $a_{i,j,k}^{(l_1)}$ is the activated value of the unit in the ith row and jth column of the feature map. Therefore, $u_{i,j,k}^{(l_1)}$ is the value before activation. $w_{bk}^{(l_1)}$ denotes the bias of the kth filter and is usually a single numeric value. $a_{i+m,j+n,p}^{(l-1)}$ is the activated value of the unit in the $(i+m)$th row and $(j+n)$th column. The rectified linear units (ReLU) has gained popularity among activation functions. Introduced by [76], ReLU works by thresholding values at 0, i.e., $f(x) = max(0,x)$ [77].

4. Proposed Deep Learning-Based Patient Load Prediction Model

The proposed deep learning-based system (Figure 3) assumes the existence of a health information exchange (HIE) system that allows health facilities to share relevant health information. HIE systems are typically categorized by how patient health information is stored and how the participants can access patient health information [78]. The common HIE models (from here on, we use the words model and system interchangeably) are:

Decentralized model: In this model, each participating health facility controls its data separately in special "edge servers" at a unique location. Patient-specific data is only shared with other participants upon request. In a strictly decentralized model, every request for patient data must be made to every participating data source.

Centralized model: Here participants agree to share data and the data is normalized in a common format and terminology and are housed together in a central data repository where they can be accessed and used by participants in line with agreed policies and procedures. More than one repository may exist for different kinds of data. This model may offer the best technical performance in terms of data availability and response time.

Hybrid-federated model: This model is similar to the decentralized model, but it adds a "record locator service" to track patient movement.

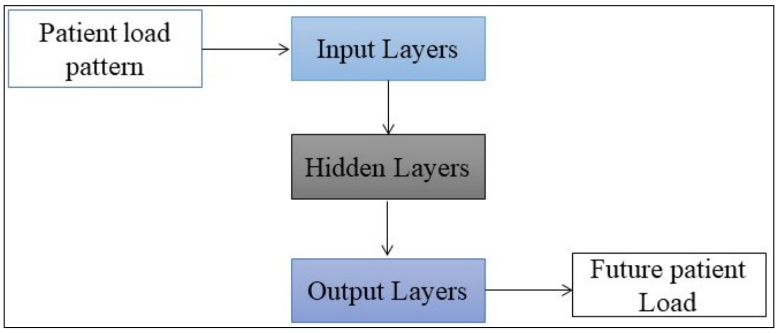


Figure 3. Proposed Deep Learning System.

For our work, the required information to be shared by the health facilities is patient load information. Hence, the proposed model can suit any of the existing HIE models. However, in this work we propose two different system, a totally centralized system in which all control and computation tasks are handled by central controller and a decentralized control system.

The patient load at a health facility is influenced by the patient rate of arrival. We define the total patient load, PL_{total} to be:

$$\begin{aligned} PL_{total} &= \text{patients relayed from other hospitals} \\ &+ \text{patients from the local catchment area (regular patient load)} \\ &= PL_{rel} + PL_{reg}. \end{aligned} \tag{6}$$

4.1. Centralized Patient Load Prediction System

The centralized prediction system involves four phases, namely data collection, training, prediction and online training.

4.1.1. Data Collection Phase

The Cc is responsible for collecting all information of the health facilities. Let PS be the patient load sequence of every facility in the last N time slots (or time intervals) recorded by the Cc. Let $T = \{t_1, t_2, \cdots, t_n\}$ be the collection of time intervals over which a hospital's traffic load can be predicted. We can assume t_i to be hours over which the hospital is operational, say 7:00–17:00. Then let $\triangle$ be the length of each time interval. Let ps_k^i be the recorded patient load of a facility i in the last time interval k. Then the past patient load PS^i of facility i is given by:

$$PS^i = \{ps_k^i, ps_{k-1}^i, \cdots, ps_{k-N+1}^i\} \tag{7}$$

which is just a length-N vector. Where N is the number of considered past time intervals. Note that N depends on the complexity of the input data and is decided according to the training performance. The controller collects all patient load series of every facility and formats them as a patient load matrix:

$$PS = \{PS^1, PS^2, \cdots, PS^M\}. \tag{8}$$

This can also be expressed as:

$$PS = \{ps_k, ps_{k-1}, \cdots, ps_{k-N+1}\}. \tag{9}$$

Which is just a time series expression of the patient loads of all health facilities in the past N time intervals.

The patient load matrix PS is the main output of the data collection phase and is taken as the input of training data. In the next time slot, the Cc records the patient load as real future patient load as:

$$ps_{k+1} = \{ps_{k+1}^1, ps_{k+1}^2, \cdots, ps_{k+1}^M\}. \tag{10}$$

This record is taken as the output of the training data. The process is repeated several times and the Cc gathers all such labeled data for training the deep NN in the training phase.

4.1.2. Training Phase

Consider M deep CNNs, with each deep CNN responsible only for training the patient load of one health facility. With this structure, the central controller takes only the future patient load of a single health facility as the output of the corresponding deep CNN. For example, take the training data of deep CNN, CNN^i, $(x_{input}, y_{output}) = \left(PS, ps_{k+1}^i\right)$. Then, the Cc trains all the deep CNNs separately so as to obtain all the stable weight matrices.

4.1.3. Prediction and Accuracy Calculation Phase

During this phase, the Cc forecasts future patient load and computes the prediction accuracy. The weight matrix of each deep CNN obtained during the the training phase is then adopted for predicting the future patient load. For all the deep CNNs, the output is recorded as:

$$PSF_{k+1} = \{psf^1_{k+1}, psf^2_{k+1}, \cdots, psf^M_{k+1}\}. \tag{11}$$

Recall that the real future patient load of time interval $(k+1)$ is recorded as $PS_{k+1} = \{ps^1_{k+1}, ps^2_{k+1}, \cdots, ps^M_{k+1}\}$. Hence, the prediction accuracy can be computed according to:

$$\frac{1}{K \times M} \sum_{k=0}^{k-1} \sum_{i=1}^{M} \frac{\left| plp^i_{k+1} - pl^i_{k+11} \right|}{pl^i_{\max}}, \tag{12}$$

where K is the total number of time intervals considered and $pl^i_{\max}$ is the maximum patient load of hospital i, i.e., daily patient capacity. See Figure 4.

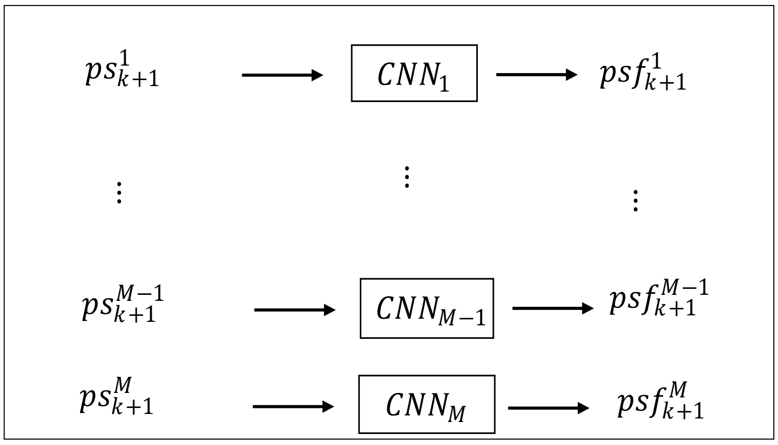


Figure 4. Patient load prediction phase in the centralized system.

4.1.4. Online Training Phase

If the arrival pattern of patient load acts in a consistent pattern/manner, then the training and prediction processes can reasonably only be based on the existing training data. It is possible, however, for the arrival pattern of patients to change because of some reasons. Under such situations, it would be important that the training process be adapted accordingly. This would necessitate the online training phase so as to allow the adjusting of the deep CNNs to adapt to the new pattern. During this phase, each health facility continuously records the patient load data and the training phase is processed periodically with the collected new training data. Hence the weight matrices are also adjusted periodically.

4.2. Decentralized Patient Load Prediction System

Unlike in the centralized system, here the Cc is granted a partial computation ability, and each health facility completes some tasks locally. Each health facility uses local information to make simple pre-forecasting. This lessens the computational burden of the Cc. The final prediction, however, is done by the Cc, but it integrates the global pre-forecasting information from all local health facilities. Below we discuss only the data collection and the training phase. The prediction phase and the online phase are the same as in the centralized control system.

4.2.1. Data Collection

In the centralized system above, the Cc predicts the patient load based on collected patient flow patterns of all health facilities. However, in the decentralized system, the Cc has limited capabilities, such that each facility does not transfer all the raw patient flow data to the Cc. Here, each facility performs some pre-processing of the raw patient flow data and sends limited/less information to the Cc, hence reducing the computational and signaling overheads of the Cc. Each facility i captures the patient load ps_k^i of the previous time interval and also separately captures the relayed patient load PL_{rel}^i and the regular patient load from its catchment area as PL_{reg}^i of the last N time slots. Using PL_{reg}^i as the input, each facility i predicts the future regular patient load $psf_{reg_{k+1}}^i$ of the next time slot. The training and prediction process is conducted in the training phase. Thereafter, each facility forwards the obtained psf_{k+1}^i and the recorded patient load of the previous time slot ps_k^i to the Cc. The received information from all health facilities is then constructed by the Cc as the training data ps_k and ps_{reg_k}.

4.2.2. Training Phase

Here the training phase is split into two steps. The first step involves each health facility training a local NN to forecast its future integrated patient load with its past N time slot regular patient loads, such that, for each facility i, the training data of its local NN can be represented as:

$$\left(x_{input}, y_{output}\right) = \left(PL_{reg}^i, ps_{reg_{k+1}}^i\right). \tag{13}$$

Compared with the input to the deep CNN used in the Cc, here the input is simpler such that the training can be treated as a function fitting process between inputs and outputs. Hence, the DBN can be used to perform the training process. As pointed out above (i.e., in the data collection section), it is the trained DBN that will be used to forecast the future integrated patient load that is denoted as $psf_{reg_{k+1}}^i$. The outcome of this process is periodically sent to the Cc.

Upon each facility completing self prediction and sending the result to the Cc, the Cc performs the final prediction with the last time interval's patient load ps_k and the predicted regular patient load $psf_{reg_{k+1}}$ of all health facilities. Since the patient load and regular load are different system features, they can be considered as two separate pieces of input data. Hence, the training data input can be formed as a matrix:

$$\left(ps_k, ps_{reg_{k+1}}\right) = \left(\{ps_k^i, ps_k^2, \cdots, ps_k^M\}, \{psf_{reg_{k+1}}^1, psf_{reg_{k+1}}^2, \cdots, psf_{reg_{k+1}}^M\}\right). \tag{14}$$

As pointed out above, the deep learning structures in the Cc are used for predicting the future patient loads of all the health facilities. Just as with the central-control-based prediction, we make use of M deep CNNs to make the prediction to reduce the computational burden and guarantee the accuracy. Hence, for CNN^i, its labeled training data is formed as (see Figure 5):

$$\left(x_{inpu}, y_{output}\right) = \left(\left(ps_k, psf_{reg_{k+1}}\right), ps_{k+1}^i\right). \tag{15}$$

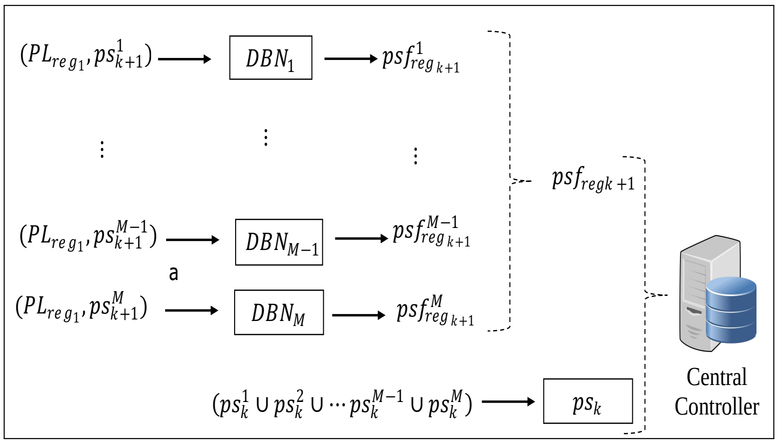

(PL_{reg_1}, ps^1_{k+1})
DBN_1
$psf^1_{reg\,k+1}$
$(PL_{reg_1}, ps^{M-1}_{k+1})$
DBN_{M-1}
$psf^{M-1}_{reg\,k+1}$
a
(PL_{reg_1}, ps^M_{k+1})
DBN_M
$psf^M_{reg\,k+1}$
psf_{regk+1}
$(ps^1_k \cup ps^2_k \cup \cdots ps^{M-1}_k \cup ps^M_k)$
ps_k
Central Controller

Figure 5. Training phase in the decentralized system.

5. Computational Considerations

A common challenge in applying machine learning techniques to problems is the exponential increase of the size of training data and the growing model complexity. It is true that the availability of large sized data requires the use of more complex machine learning models so as to discover finer structures in the data. However, more complex models are often associated with higher computational costs. Furthermore, deeper NNs also bring in more complex computational patterns [79]. Techniques such as, shrinking the model, data parallelism and model parallelism are being explored as possible solutions to accelerate the performance of deep learning models. In [80] it was shown that about 80% of the computational cost in NNs results from the convolution operation [81]. Hence, recent studies on NN model designs have focused on how to configure convolutional layers [82]. Thus to control the execution overhead of an NN model, in [83], the authors mathematically formulate the convolutional overhead. Without such a formal overhead formulation, an NN model structure is mainly configured based on a designer's experience with balancing the execution overhead and the model's accuracy. The designer manually selects a configuration (based on experience or some public models) and then trains the model. In the case of low training accuracy, a different configuration will be tested. Clearly this approach lacks a systematic way to configure the model. On the contrary, with the execution overhead formulation, this configuration can be quantitative and effective. Denote $\mathcal{O}$ as the convolutional overhead, such that:

$$\mathcal{O} \leq \alpha \times \mathcal{B}, \tag{16}$$

where $\mathcal{B}$ is the preferred or predefined resource budget and α is the percentage of computations due to convolution. Hence, the execution overhead of layer k, denoted $\mathcal{O}_k$, needs to satisfy:

$$\mathcal{O}_k = \mathcal{O} \times \beta_j \leq \alpha \times \mathcal{B} \times \beta_j, \tag{17}$$

where β_j indicates the computation percentage of each layer, $j = 1, 2, 3, \cdots, m$. For a detailed discussion on this method we refer the reader to [83].

6. Patient Transfer Logistics

This section presents the development of the IoT-based smart bus system which can support the timely movement of reassigned patients from a bus stop near the source facility to the destination facility. This can help prevent putting a further strain on the already fragmented ambulatory services in many clinics. Since upon arrival patients are triaged and get treated in order of urgency, then in order of arrival, the patients that are re-assigned are the least critical, such that they can manage to carry themselves to the nearest bus station.

Hospital logistics generally differentiate between three main flows: Patient, information and material flows. The patient flows run in three directions: Towards the hospital or inbound, within the hospital or internal and away from the hospital or outbound. As pointed out above, the goals of the above proposed system are to predict future patient loads. Furthermore, in the case of excess patient loads, we propose that excess loads should be re-assigned to nearby facilities that have a minimal load as a way to control overcrowding and reduce the number of patients that leave health facilities without receiving medical care. However, the re-assigning of patients will imply a need for transportation for the patient to move from one facility to another. The two most commonly employed modes of transfer of patients are ground transport, with the inclusion of ambulances, and mobile intensive care units (MICUs) [84]. Here we assume the existence of a scheduled bus transport system. A bus journey involves such stages as waiting, queuing and transferring from the origin point to the final destination. These stages are impacted by the services rendered and transportation network resources. Three stages exist at which transit services can affect passengers [85]. These stages are:

Origin-point stage: At this level, passengers wait for the next bus. Passenger waiting time can be longer than expected due to irregular headway or technical issues that might affect the bus. Moreover, all passengers may not board the arriving bus due to capacity limitation. Upon being queried, the proposed smart bus system will provide passengers with information regarding the position of the bus, travel time estimate and information about the number of vacant seats in the bus. This information will help passengers to plan their waiting time accordingly.

Boarding stage: At this stage, the passenger services time (PST) is a function of passenger demand. A boarding passenger's wait-time must allow passengers in the bus to alight. Here, we assume that near each health facility there is a boarding stage.

Arrival stage: The arrival stage is the stage where passengers reach their final destination. Their arrival can be on time, ahead or delayed based on the deviation from the timetable. For the patients, the arrival stage will be the stage near the destination facility where the patient has been referred.

Various solutions exist that are used to track data related to the departure and arrival times of buses. Radio frequency (RF) transceivers [86] are installed on buses and bus stops to enable buses to communicate their location to bus stops. The estimated arrival time of the bus is calculated by microprocessors at the bus stops. The calculated time is displayed on screens that are installed at the stages. Some solutions employ SMS services on global system for mobile communication (GSM) modules to transmit bus positions to databases. GPS tracking devices are mounted on buses to provide location data. Sending of data to the database is via the hyper text transfer protocol [87,88]. Alternatively, location data can be streamed from android devices in the bus. The bus location and travel time approximation can be accessed through android devices or web portals. Unfortunately, some of these systems cannot handle an increase in requests. The bandwidth demand for sending the data from buses to servers is high. Hence, there is need to explore scalable and efficient solutions [89].

6.1. System Architecture

This section discusses the overall system architecture of our proposed smart transport system. Figure 6 shows the proposed system architecture. Each public bus is attached with an IoT smart transport kit which comprises a variety of modules including the GPS module and two sets of laser lights with light dependent resistors (LDRs) positioned at the back and front entryway of the bus, respectively, and a smart phone that acts as a mobile hotspot. Both the GPS and the laser light with an LDR read and feed real-time data to an Arduino Uno microcontroller. The Arduino Uno connects to the Internet via NodeMCU, which sends the collected data to an MQTT broker on the cloud using the Wi-Fi provided by the mobile hotspot that provides the Internet connection. Since the smart buses will also be servicing patients, the system will need to be reliable. The patients need to accurately know the arrival time of the bus at a bus stop and the time it will reach the destination and also the status of the capacity of the bus (whether it is full or there are vacant seats in the bus). Using the real time location of buses, it is possible to calculate the approximate time for the bus to reach a particular position.

This is possible since a bus's location data can be saved on the server together with corresponding timestamps. To count the number of passengers alighting and boarding, we make use of the laser beam. The choice of the laser is motivated by the fact that it does not scatter and is invisible. Since passengers queue and enter or exit the bus one by one, with the laser, the system will not miss any passenger. Hence the counter will be reliable.

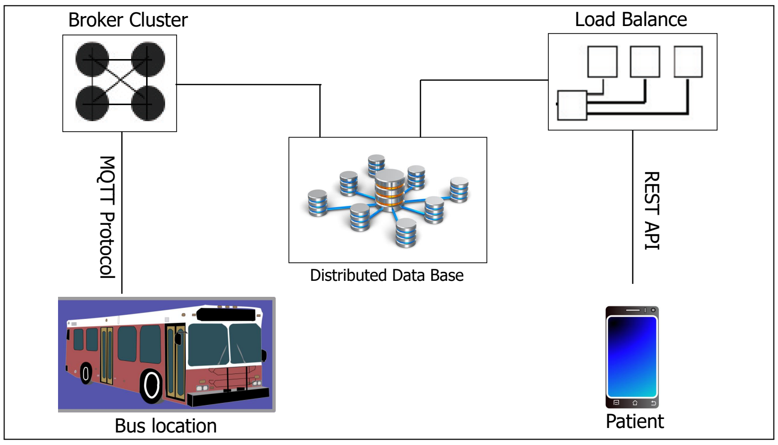


Figure 6. System architecture.

As pointed out above, each of the public buses will be tagged with an IoT-based smart bus system. Figure 7 shows the framework of the developed IoT kit. We briefly describe the function of each of the components of the system.

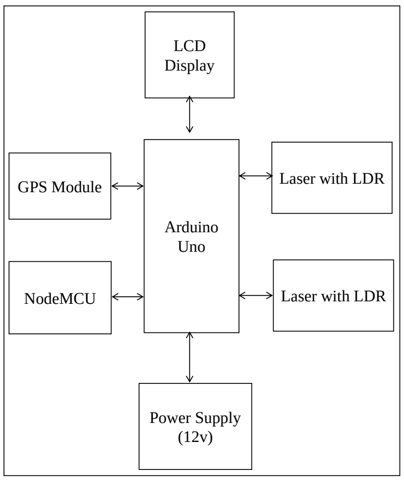


Figure 7. Framework of developed smart bus system

Arduino Uno: This is the heart of our framework. It works as a CPU unit. Arduino Uno will generate sensor data that will have to be sent to the server. This data will include the location data of the bus, time and distance approximation of departure and arrival and also information about the capacity of the bus. Arduino will take these data and send them to NodeMCU over a serial connection.

Laser: A laser is a device emitting light through optical amplification based on the stimulated emission of electromagnetic radiation. The term "laser" originated as "light amplification by stimulated emission of radiation" [90]. The primary wavelengths of laser radiation for most current applications include the ultraviolet, visible and infrared regions of the spectrum. Ultraviolet radiation for lasers consists of 180 and 400 nm wavelengths. The visible region lies between 400 and 700 nm wavelengths. The infrared region of the spectrum consists of radiation with 700 nm and 1 mm wavelengths. When the intensity of the radiation is sufficiently high, damage to the absorbing tissue may happen [91]. In the proposed system, the primary functionality of the laser technology is to detect motion (i.e., entry and exist of passengers) and serve as a counter to determine the number of vacant seats in the bus. The counting starts when there is a discontinuity in the laser beam falling on the sensor. The discontinuity is detected at the entrance (or exit) of the door of the public buses at the instant of time where the laser beam is not being felt on the sensor. Here we utilize the laser technology whose path is invisible.

NodeMCU: NodeMCU is a development board that incorporates the ESP8266 Wi-Fi chip. The ESP8266 is programmable like any other microcontroller. Since Arduino Uno does not have network capabilities, the NodeMCU interfaces the Arduino Uno microcontroller, connects the system to the Internet and drives the output for the GPS and the lasers. The Wi-Fi connection is provided by the mobile hotspot that is fixed in each bus. An application server implemented using Node.js collects the data from the MQTT broker through the publish/subscribe mechanism and saves the data using the NSQL Couch DB database.

GPS receiver: GPS is a space-based radio navigation system that broadcasts highly accurate navigation pulses to users on or near the Earth. This system is used to collect the real-time co-ordinates for the system. The GPS receives the satellite signals and then the position coordinates with latitude and longitude are determined by it. The location is determined with the help of GPS and the transmission mechanism. After receiving the data, the tracking data can be transmitted using any wireless communication systems. These units are connected at the output of the system. The GPS port is serially connected with the system.

6.2. MQTT Protocol

The message queuing telemetry transport (MQTT) is a client server publish/subscribe messaging transport protocol. It is lightweight, open, simple and easily implementable. MQTT is ideal for use in IoT applications that require a small code footprint and/or network bandwidth as a premium. The protocol runs over network protocols that provide ordered, lossless, bi-directional connections, e.g., TCP/IP. In the MQTT protocol there is a broker and a receiver. The broker has 'topics' through which a recipient of data generated by a sender is determined. To receive the data, a receiver subscribes to a particular topic. A receiver receives the data published by any sender with that particular topic [92]. In the proposed system, the topic will be the route number/name, such that the published location data from buses will contain this information. The location data will comprise the current latitude and longitude of a bus and will be published at a frequency of 5 s. To ensure the scalability of the system, the MQTT brokers can be clustered as in Figure 8. Each of the brokers maintains the same topic list . In case of one broker crushing, another broker automatically handles the requests, balancing within themselves. The advantage of clustering is the assured system availability. For each route number, the location data is received via MQTT and stored in the database. The broker can be directly modified to write to the database upon reception of data or an MQTT client can be created to perform this function.

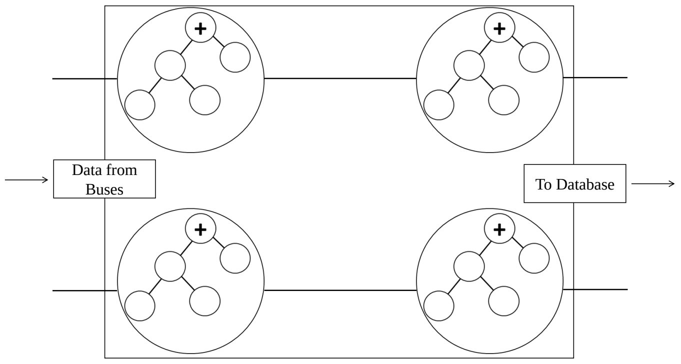


Figure 8. Message queuing telemetry transport (MQTT) clusters: Every node maintains a tree of topics.

6.3. Data Access by Users

Various platforms can be used to enable user access to the system including web, smartphones or *SMS*. The proposed system allows users to access the data via Representational state transfer (REST) APIs. HTTP requests are sent to the server and the server returns fetched data based on user context. The location information can easily be shown to users via Google maps or an equivalent map service. Through a simple application, patients and general passengers can easily search a bus by bus number directly. In case the user is not aware of the bus number, he/she will be asked to first select the nearest bus stage (i.e., the boarding point). The process can easily be automated by keeping a database of bus stops along with their corresponding location data. Distance to the the system is calculated by comparing a user's location to that of the bus stops, applying the Haversine formula:

- Lat_1 : User latitude
- $Long_1$: User longitude
- Lat_2 : Bus stop latitude
- $Long_2$: Bus stop longitude
- d_{Long} : $Long_2 - Long_1$
- d_{Lat} : $Lat_2 - Lat_1$
- r : Earth's Radius.

Then

$$\begin{aligned} H &= \left(\sin\left(\frac{d_{\text{Lat}}}{2}\right)\right)^2 + \cos\left(Lat_1\right) \times \cos\left(Lat_2\right) \times \left(\sin\left(\frac{d_{\text{long}}}{2}\right)\right)^2 \\ B &= 2 \times \tan -1 \left(\sqrt{H}, \sqrt{1-H}\right) \\ \text{Distance} &= Br \end{aligned} \tag{18}$$

From Equation (18) above, the nearest bus stop will be the one with the shortest distance. The live location of a bus is queried based either on target destination or bus number. If the user enters a bus number, the user API returns the last known bus location from the central buses collection. To allow mapping of the boarding stop–destination point to the route number, a database of all buses on all routes can be maintained. Hence, if a user selects a boarding point and a destination point, he/she gets a live location of all buses on a route that takes him/her to the destination.

In Table 2 above, the first row shows the route numbers, whilst the corresponding columns list bus stops on that route. For example, if a passenger queries data by boarding and destination bus stages, the query is then checked in the table. Take stops B and A where B is the boarding stage and A is the destination bus stage. The system simply checks the table to verify if bus stop B occurs in the table and is followed by A. However, bus stop A need not immediately follow B. In Table 2, we see that routes number 101, 201 and 301 have such conditions. The query will then fetch the live locations of the buses on these three routes and return them to the user. The queries from the user module will be the same across all platforms. The output for the HTTP GET request will take the following formats.

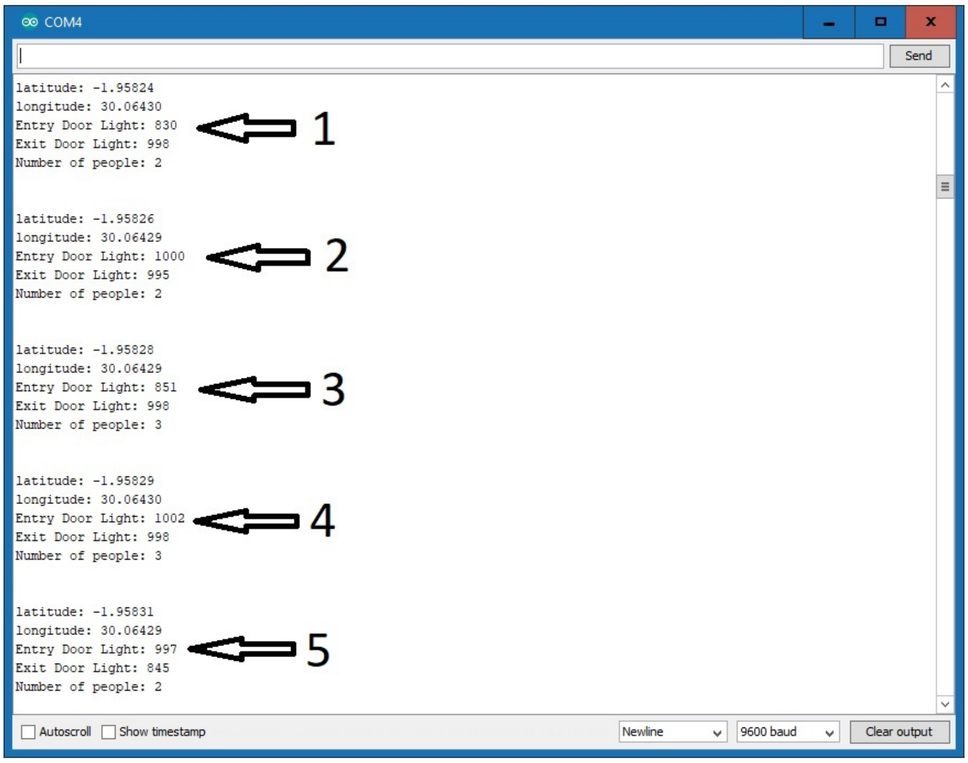
- Query by registration number: The query by registration number will return a JavaScript Object Notation (JSON) response fetching data from 'buses' collection with the parameters latitude, longitude, bus registration number, bus route number, direction and time stamp.
- Query by route number: The query by route number will return a JSON array containing objects, with each of the objects giving details of active buses running on that route. The data will be fetched from a collection with the name 'route number'.
- Query by boarding point and destination: This query will return a JSON array containing objects for active buses on every route, from the boarding point to the destination. This data will be fetched from various collections based on which routes are applicable.

Table 2. Table of bus stops along routes.

101	**201**	**301**	**401**	**501**
B	B	A	S	A
A	A	Z	A	Q
C	H	B	D	C
E	R	H	G	D
D	L	G	F	S

6.4. Functionality Principle of the Counter

Figure 9 is the snapshot showing the system transmitting captured data. As pointed out above, the LDR (light dependent resistor) detects the increase in the intensity of the light if the laser is pointed directly at it. When there is an obstacle between the laser and the LDR, the LDR detects a decrease in the intensity of the light. Hence, by counting each decrease in the range of the optimum level, the system is able to count passengers entering and exiting the smart bus. From Figure 9, as per our simulations, the maximum light intensity of both the entry and exit doors' LDR is set at $\geq$900 such that any value of $<$900 signifies an obstacle and in our case this will signify a passenger leaving or entering the smart bus. From part 2 in Figure 9, we see that the number of passengers in the bus is 2 and the light intensities on the LDR for the entry and exit doors are 1000 and 995, respectively. Since both are above 900, this means there is no entry or exit of passengers from the smart bus. In part 3 of Figure 9, we observe that the light intensity for the entry door is $<$900 and this decrease corresponds to the increase in the number of passengers, which has increased from 2 to 3. For this simulation, the smart kit was stationary; hence, the fixed latitude and longitude values are -1.95824 and 30.06430, respectively.



```
latitude: -1.95824
longitude: 30.06430
Entry Door Light: 830
Exit Door Light: 998
Number of people: 2

latitude: -1.95826
longitude: 30.06429
Entry Door Light: 1000
Exit Door Light: 995
Number of people: 2

latitude: -1.95828
longitude: 30.06429
Entry Door Light: 851
Exit Door Light: 998
Number of people: 3

latitude: -1.95829
longitude: 30.06430
Entry Door Light: 1002
Exit Door Light: 998
Number of people: 3

latitude: -1.95831
longitude: 30.06429
Entry Door Light: 997
Exit Door Light: 845
Number of people: 2
```

Figure 9. Arduino Uno serial monitor displaying captured data.

7. Conclusions

The need to access quality health services is a basic human need. However, the health systems are experiencing many challenges that make meeting its goals of offering quality healthcare service a challenge. A mismatch between staffing ratios and service demand results in the overcrowding of patients in hospital waiting rooms leads to excessive patient waiting times, incomplete preventive service delivery and disgruntled medical staff. It is important that health facilities coordinate their efforts to ensure that all patients get the needed health care as and when they need it. In this paper, a coordinated approach between the health facilities has been proposed as a possible solution to deal with the problem of excess patient load and overcrowding. Building on the power of deep learning, a patient load prediction model that can be used to predict future patient loads is presented. Furthermore, to avoid putting a strain on the already fragmented ambulatory service in health facilities, we propose an approach that makes use of the existing public transport system to support the healthcare delivery system. A conceptual framework for an IoT-based smart bus transport system that can support timely delivery of health services has been presented. Using off-the-shelf sensors, an IoT smart transport kit is developed. This kit can be attached to ordinary public buses allowing patients and general passengers to query bus location and time related information to enable them move from one health facility to another. In this work, we have only focused on the transfer of first time patients from one health facility to another. Our future work will focus on scheduled patient arrivals and also the transfer of patient health files since a patient's past record may have a bearing on his/her current health status.

Author Contributions: The ideas presented in this work are a product of contributions by all the authors. The conceptualization of the idea and the development of the manuscript was done by author K.M. in consultation with the supervision team comprising authors S.K., C.M., K.J. and J.N. The supervision team played a key role in providing the needed advise and direction throughout the development process of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

IoT	Internet of Things
RF	Readio Frequency
GSM	Global System for Mobile communication
LDR	Light Dependent Resistor
GPS	Global Positioning System
MQTT	Message Queuing Telemetry Transport
CNN	Convolutional Neural Network
DBA	Deep Belief Architectures
DBM	Deep Belief Machines
OPD	Out Patient Department
ED	Emergency Department
API	Application Program Interface
ARIMA	Auto-Regressive Integrated Moving Average
NICU	Neonatal Intensive Care Unit
MRF	Markov Random Field
REST	Representational state transfer
JSON	JavaScript Object Notation
HIES	Health Information Exchange System
MICU	Mobile Intensive Care Unit
PST	Passenger Services Time
GLM	General Linear Model
VAR	Vector Autoregression
UCM	Unobserved Components Model
GARCH	Generalized Autoregressive Conditional Heteroscedasticity

References

1. Bahadori, M.; Teymourzadeh, E.; Ravangard, R.; Raadabadi, M. Factors affecting the overcrowding in outpatient healthcare. *J. Educ. Health Promot.* **2017**, *6*. [CrossRef]
2. Yarmohammadian, M.H.; Rezaei, F.; Haghshenas, A.; Tavakoli, N. Overcrowding in emergency departments: A review of strategies to decrease future challenges. *J. Res. Med. Sci.* **2017**, *22*. [CrossRef]
3. Tran, T.D.; Nguyen, U.V.; Minh, N.V.; Tran, B.X. Patient waiting time in the outpatient clinic at a central surgical hospital of Vietnam: Implications for resource allocation [version 3; referees: 2 approved]. *F1000Research* **2017**, *6*. [CrossRef]
4. Shukla, Y.; Tiwari, R.; Rohit, B.K.; Kasar, P.K. An assessment of OPD registration counter services and channelization of patients in NSCB Medical College Hospital, Jabalpur (MP). *Int. J. Med. Sci. Public Health* **2015**, *4*, 1468–1472. [CrossRef]
5. Obulor, R.; Eke, B.O. Outpatient queueing model development for hospital appointment system. *IJSEAS* **2016**, *2*, 15–22.
6. Alexopoulos, C.; Goldsman, D.; Fontanesi, J.; Kopald, D.; Wilson, J.R. Modeling patient arrivals in community clinics. *Int. J. Manag. Sci.* **2008**, *36*, 33–43. [CrossRef]
7. Fontanesi, J.M.; Guire, M.D.; Chiang, J.; Holcomb, K.; Sawyer, M. Application of workflow analysis tools in outpatient primary care settings. *Jt. Comm. J. Qual. Improv.* **2000**, 26, 654–660. [PubMed]
8. Zarei, E.; Daneshkohan, A.; Khabiri, R.; Arab, M. The Effect of Hospital Service Quality on Patient's Trust. *Iran. Red Crescent Med. J.* **2014**, *17*. [CrossRef] [PubMed]
9. Sun, J.; Lin, Q.; Zhao, P.; Zhang, Q.; Xu, K.; Chen, H.; Hu, C.J.; Stuntz, M.; Li, H.; Li, Y. Reducing waiting time and raising outpatient satisfaction in a Chinese public tertiary general hospital—An interrupted time series study. *BMC Public Health* **2017**, *17*, 668. [CrossRef] [PubMed]
10. Dinesh T.A.; Singh, S.; Nair, P.; Remya, T.R. Reducing Waiting Time in Outpatient Services of Large University Teaching Hospital—A Six Sigma Approach. *Manag. Health* **2013**, *17*, 31–37.

11. Kulshrestha, A.; Singh, J. Inter-hospital and intra-hospital patient transfer: Recent concepts. *Indian J. Anesth.* **2016**, *60*, 451–457. [CrossRef] [PubMed]
12. Sokol-Hessner, L.; White, A.A.; Davis, K.F.; Herzig, S.J.; Hohmann, S.F. Inter-hospital transfer patients discharged by academic hospitalists and general internists: Characteristics and outcomes. *J. Hosp. Med.* **2015**, *11*, 245–250. [CrossRef] [PubMed]
13. Owad, A.A.; Samaranayake, P.; Karim, A.; Ahsan, K.B. An integrated lean methodology for improving patient flow in an emergency department—Case study of a Saudi Arabian hospital. *Prod. Plan. Control* **2018**, *29*, 1058–1081. [CrossRef]
14. Derlet, W.R. Overcrowding in Emergency Departments: Increased Demand and Decreased Capacity. *Ann. Emer. Med.* **2002**, *39*, 430–432. [CrossRef] [PubMed]
15. Gajanan, A. Reducing Wait Time Prediction in Hospital Emergency Room: Lean Analysis Using a Random Forest Model. Master's Thesis, University of Tennessee, Knoxville, TN, USA, May 2017.
16. Sarah, P.; Shaun, G.; Nigam, H.S. Predicting Emergency Department Visits. *AMIA Jt. Summits Transl. Sci. Proc.* **2016**, *2016*, 438–445.
17. Lv, Y.; Duan, Y.; Kang, W.; Li, Z.; Wang, F. Traffic flow prediction with big data: A Deep Learning Approach. *IEEE Trans. Intell. Transp. Syst.* **2015**, *16*, 2015. [CrossRef]
18. Gal, A.; Mandelbaum, A.; Schnitzler, F.; Senderovich, A.; Weidlich, M. Traveling time prediction in scheduled transportation with journey segments. *Inf. Syst.* **2015**, *64*, 266–280. [CrossRef]
19. McCarthy, B.D.; Propp, K.; Cohen, A. Learning from Health Information Exchange Technical Architecture and Implementation in Seven Beacon Communities. *EGEMS* **2014**, *2*. [CrossRef] [PubMed]
20. Google Maps. Available online: https://cloud.google.com/maps-platform/ (accessed on 3 December 2018).
21. Srinivasa, K.G.; Sowmya, B.J.; Abhinav, S.; Aahan, U.R.S. Data Analytics Assisted Internet of Things Towards Building Intelligent Healthcare Monitoring Systems: IoT for Healthcare. *J. Organ. End User Comput.* **2018**, *30*. [CrossRef]
22. Chen, M.; Liang, L.L.; Chang, Y.T.; Juang, W.C. Emergency department overcrowding: Quality improvement in a Taiwan Medical Center. *J. Formos. Med. Assoc.* **2018**, *17*, 30790–30798. [CrossRef]
23. Leveraging Mobile Apps to Reduce Emergency Room Overcrowding. Available online: https://www.mobilesmith.com/wp-content/uploads/2017/06/Leveraging-Mobile-Apps-to-Reduce-Emergency-Room-Overcrowding.pdf (accessed on 22 November 2018).
24. Crowding, Boarding, and Patient Throughput. Available online: https://www.ena.org/docs/default-source/resource-library/practice-resources/position-statements/crowdingboardingandpatientthroughput.pdf?sfvrsn=5fb4e79f_4 (accessed on 22 November 2018).
25. Sethi, D.; Subramanian, S. When place and time matter: How to conduct safe inter-hospital transfer of patients. *Saudi J. Anaesth.* **2014**, *8*, 104–113. [CrossRef] [PubMed]
26. Hill, B. Optimization of Interhospital Transfer of Patients to Reduce Emergency Department Overcrowding. Available online: https://scinapse.io/papers/1734556231 (accessed on 4 September 2018).
27. Santillan, D. Uses of satisfaction data: Report on improving patient care. *Soc. Sci. Med.* **2000**, *12*, 24–26.
28. Mensah, J. Optimizing Patient Flow and Resource Utilization in Out Patient Clinic: A Comparative Study of Nkawie Government Hospital and Aniwaa Health Center. *J. Appl. Bus. Econ.* **2014**, *16*, 181–188.
29. Dan, T.; Qualls, C. Time series forecasts of emergency department patient volume, length of stay, and acuity. *Ann. Emerg. Med.* **1994**, *23*, 299–306.
30. Rotstein, Z.; Wilf-Miron, R.; Lavi, B.; Shahar, A.; Gabbay, U.; Noy, S. The dynamics of patient visits to a public hospital ED: A statistical model. *Am. J. Emerg. Med.* **1997**, *15*, 596–599. [CrossRef]
31. Batal, H.; Tench, J.; Mcmillan, S.; Adams, J.; Mehler, P.S. Predicting patient visits to an urgent care clinic using calendar variables. *Acad. Emerg. Med. Off. J. Soc. Acad. Emerg. Med.* **2001**, *8*, 48–53. [CrossRef] [PubMed]
32. Reis, B.Y.; Mandl, K.D. Time series modeling for syndromic surveillance. *BMC Med. Inform. Decis. Mak.* **2003**, *3*, 1–11. [CrossRef] [PubMed]
33. Brillman, J.C.; Burr, T.; Forslund, D.; Joyce, E.; Picard, R.; Umland, E. Modeling emergency department visit patterns for infectious disease complaints: Results and application to disease surveillance. *BMC Med. Inform. Decis. Mak.* **2005**, *4*, 1–14. [CrossRef] [PubMed]
34. Flottemesch, T.J.; Gordon, B.D.; Jones, S.S. Advanced Statistics: Developing a formal model of emergency department census and defining operational efficiency. *Acad. Emerg. Med. Off. J. Soc. Acad. Emerg. Med.* **2007**, *14*, 799–809. [CrossRef] [PubMed]

35. Boyle, J.; Wallis, M.; Jessup, M.; Crilly, J.; Lind, J.; Miller, P.; Fitzgerald, G. Regression forecasting of patient admission data. In Proceedings of the 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Vancouver, BC, Canada, 20–25 August 2008; IEEE: Piscataway, NJ, USA, 2008; pp. 3819–3822.
36. Au-Yeung, S.W.; Harder, U.; Mccoy, E.J.; Knottenbelt, W.J. Predicting patient arrivals to an accident and emergency department. *Emerg. Med. J.* **2009**, *26*, 241–244. [CrossRef] [PubMed]
37. Kam, H.J.; Sung, J.O.; Park, R.W. Prediction of Daily Patient Numbers for a Regional Emergency Medical Center using Time Series Ananlysis. *Healthc. Inform. Res.* **2010**, *16*, 158–165. [CrossRef] [PubMed]
38. Capan, M.; Hoover, S.; Jackson, E.V.; Paul, D.; Locke, R. Time Series Analysis for Forecasting Hospital Census: Application to the Neonatal Intensive Care Unit. *Appl. Clin. Inform.* **2016**, *7*, 275–289. [PubMed]
39. Morzuch, B.J.; Allen, P.G. Forecasting hospital emergency department arrivals. In Proceedings of the 26th Annual Symposium on Forecasting, Santander, Spain, 11–14 June 2006.
40. Schweigler, L.M.; Desmond, J.S.; McCarthy, M.L.; Bukowski, K.J.; Ionides, E.L.; Younger, J.G. Forecasting models of emergency department crowding. *Acad. Emerg. Med.* **2009**, *16*, 301–308. [CrossRef] [PubMed]
41. Marcilio, I.; Hajat, S.; Gouveia, N. Forecasting daily emergency department visits using calendar variables and ambient temperature readings. *Acad. Emerg. Med.* **2013**, *20*, 769–777. [CrossRef] [PubMed]
42. Kim, K.; Lee, C.; O'Leary, K.; Rosenauer, S.; Mehrotra, S. *Predicting Patient Volumes in Hospital Medicine: A Comparative Study of Different Time Series Forecasting Methods*; Tech. Rep.; Northwestern University: Evanston, IL, USA, 2014.
43. Pan, W.T. A Newer Equal Part Linear Regression Model: A Case Study of the Influence of Educational Input on Gross National Income. *Eurasia J. Math. Sci. Technol. Educ.* **2017**, *13*, 5765–5773. [CrossRef]
44. Jones, S.S.; Thomas, A.; Evans, R.S.; Welch, S.J.; Haug, P.J.; Snow, G.L. Forecasting daily patient volumes in the emergency department. *Acad. Emerg. Med.* **2008**, *15*, 159–170. [CrossRef] [PubMed]
45. Aladag, C.H.; Aladag, S. Forecasting the number of outpatient visits with different activation functions. *Adv. Time Ser. Forecast.* **2012**, 26–33.
46. Xu, M.; Wong, T.C.; Chin, K.S. Modeling daily patient arrivals at Emergency Department and quantifying the relative importance of contributing variables using artificial neural network. *Decis. Support Syst.* **2013**, *54*, 1488–1498. [CrossRef]
47. Srikanth, K.; Arivazhagan, D. An Efficient Patient Inflow Prediction Model for Hospital Resource Management. *IJEECS* **2017**, *7*, 809–817. [CrossRef]
48. Malhotra, P.; Vig, L.; Shroff, G.; Agarwal, P. Long short term memory networks for anomaly detection in time series. In Proceedings of the 23rd European Symposium on Artificial Neural Networks, Bruges, Belgium, 22–24 April 2015; Presses Universitaires de Louvain: Louvain-la-Neuve, Belgium, 2015; Volume 23.
49. Hamilton, D.J. *Time Series Analysis*; Princeton University Press: Princeton, NJ, USA, 1994; Volume 2.
50. Azoff, E.M. *Neural Network Time Series Forecasting of Financial Markets*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1994.
51. Lee, H.; Pham, P.; Largman, Y.; Ng, A.Y. Unsupervised feature learning for audio classification using convolutional deep belief networks. In *Advances in Neural Information Processing Systems, Proceedings of the Neural Information Processing Systems 2009, Vancouver, BC, Canada, 7–10 December 2009*; Neural Information Processing Systems Foundation, Inc.: San Diego, CA, USA, 2009; pp. 1096–1104.
52. Suzuki, K.; Kiryu, T.; Nakada, T. Fast and precise independent component analysis for high field fmri time series tailored using prior information on spatiotemporal structure. *Hum. Brain Mapp.* **2002**, *15*, 54–66. [CrossRef] [PubMed]
53. Kuremoto, T.; Kimura, S.; Kobayashi, K.; Obayashi, M. Time series forecasting using a deep belief network with restricted boltzmann machines. *Neurocomputing* **2014**, *137*, 47–56. [CrossRef]
54. Turner, J.T. Time Series Analysis Using Deep Feed Forward Neural Networks. Ph.D. Thesis, University of Maryland, Catonsville, MD, USA, 2014.
55. Längkvist, M. Modeling Time-Series with Deep Networks. Ph.D. Thesis, Örebro University, Örebro, Sweden, 2014.
56. LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time series. In *The Handbook of Brain Theory and Neural Networks*; AT&T Bell Laboratories: Holmdel, NJ, USA, 1995.
57. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inform. Process. Syst.* **2012**, *1*, 1097–1105. [CrossRef]

58. Torres, J.F.; Troncoso, A.; Koprinska, I.; Wang, Z.; Martinez-Alvarez, F. Deep Learning for big data time series forecasting applied to solar power. In *Advances in Intelligent Systems and Computing*; Springer: Cham, Switzerland, 2018.
59. Bao, W.; Yue, J.; Rao, Y. A deep learning framework for financial time series using stacked autoencoders and long-short term memory. *PLoS ONE* **2017**, *12*, e0180944. [CrossRef] [PubMed]
60. Video Analysis to Detect Suspicious Activity Based on Deep Learning. Available online: https://medium.com/@everisUS/video-analysis-to-detect-suspicious-activity-based-on-deep-learning-fee2032ea14a (accessed on 5 May 2019).
61. Abdel-Hamid, O.; Mohamed, A.; Jiang, H.; Penn, G. Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition. In Proceedings of the 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Kyoto, Japan, 25–30 March 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 4277–4280.
62. Kim, Y. Convolutional neural networks for sentence classification. *arXiv* **2014**, arXiv:1408.5882.
63. Yang, J.; Nguyen, M.N.; San, P.P.; Li, X.L.; Krishnaswamy, S. Deep convolutional neural networks on multichannel time series for human activity recognition. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
64. Mozo, A.; Ordozgoiti, B.; Gomez-Canaval, S. Forecasting short-term data center network traffic load with convolutional neural networks. *PLoS ONE* **2018**, *13*, e0191939. [CrossRef] [PubMed]
65. Negnevitsky, M.; Mandal, P.; Srivastava, A.K. Machine learning applications for load, price and wind power prediction in power systems. In Proceedings of the 15th International Conference on Intelligent System Applications to Power Systems, Curitiba, Brazil, 8–12 November 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 1–6. [CrossRef]
66. Fragkos, G.; Apostolopoulos, P.A.; Tsiropoulou, E.E. ESCAPE: Evacuation strategy through clustering and autonomous operation in public safety systems. *Future Internet* **2019**, *11*, 20. [CrossRef]
67. Li, Y. Deep Reinforcement Learning: An Overview. Available online: https://arxiv.org/pdf/1701.07274.pdf (accessed on 7 August 2019).
68. Ivanov, S.; D'yakonov, A. Modern Deep Reinforcement Learning Algorithms. *arXiv* **2019**, arXiv:1906.10025.
69. Irpan, A. Deep Reinforcement Learning Doesn't Work Yet. Available online: https://www.alexirpan.com/2018/02/14/rl-hard.Html (accessed on 23 November 2018).
70. Henderson, P.; Islam, R.; Bachman, P.; Pineau, J.; Precup, D.; Meger, D. Deep reinforcement learning that matters. In Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
71. Hippert, H.S.; Pedreira, C.E.; Souza, R.C. Neural networks for short-term load forecasting: A review and evaluation. *IEEE Trans. Power Syst.* **2001**, *16*, 44–55. [CrossRef]
72. Senjyu, T.; Mandal, P.; Uezato, K.; Funabashi, T. Next day load curve forecasting using hybrid correction method. *IEEE Trans. Power Syst.* **2005**, *20*, 102–109. [CrossRef]
73. Jose, T.; Alicia, T.; Irene, K.; Wang, Z.; Francisco, M.A. Deep learning for big data time series forecasting applied to solar power. In *International Joint Conference SOCO'18-CISIS'18-ICEUTE'18, San Sebastián, Spain, 6–8 June 2018*; Advances in Intelligent Systems and Computing; Springer: Cham, Switzerland, 2018; pp. 123–133._12. [CrossRef]
74. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
75. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Boston, MA, USA, 2016.
76. Sze, V.; Chen, Y.; Yang, T.; Emer, J.S. Efficient processing of deep neural networks: A tutorial and survey. *Proc. IEEE* **2017**, *105*, 2295–2329. [CrossRef]
77. Richard, H.R.H.; Rahul, S.; Misha, A.M.; Rodney, J.D.; Seung, S.S. Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature* **2000**, *405*, 947–951.
78. Agarap, A.F. Deep learning using rectified linear units (ReLU). *Comput. Sci.* **2018**, arXiv:1803.08375.
79. Ji, H.; Yoon, S.; Heo, E.Y.; Hwang, H.; Kim, J.W. Technology and policy challenges in the adoption and operation of health information exchange systems. *Healthc. Inf. Res.* **2017**, *23*, 314–321. [CrossRef] [PubMed]
80. Li, M. Scaling Distributed Machine Learning with System and Algorithm Co-Design. Available online: https://www.cs.cmu.edu/~muli/file/mu-thesis.pdf (accessed on 5 August 2019).

81. Huynh, L.N.; Lee, Y.; Balan, R.K. Deepmon: Mobile GPU-Based Deep Learning Framework for Continuous Vision Applications. In Proceedings of the 15th Annual International Conference on Mobile Systems, Applications, and Services, Niagara Falls, NY, USA, 19–23 June 2017.
82. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
83. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
84. Yang, K.; Gong, X.; Liu, Y.; Li, Z.; Xing, T.; Chen, X.; Fang, D. cDeepArch: A compact deep neural network architecture for mobile sensing. In Proceedings of the 2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Kowloon, Hong Kong, 11–13 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–9. [CrossRef]
85. Everything Flows: Transportation and Material Flows in Hospital Logistics. Available online: https://www.medica-tradefair.com/cgi-bin/md_medica/lib/pub/tt.cgi/Everything_flows_transportation_and_material_flows_in_hospital_logistics.html?oid=88697&lang=2&ticket=g_u_e_s_t, (accessed on 24 May 2019).
86. Lu, K.; Han, B.; Zhou, X. Smart Urban Transit Systems: From Integrated Framework to Interdisciplinary Perspective. *Urban Rail Transit* **2018**, *4*, 49–67. [CrossRef]
87. Gaurav, C.; Niket, G.; Manal, C.; Jitesh, D.; Saylee, G. Real Time Bus monitoring and Passenger Information System. *Int. J. Soft Comput. Eng.* **2012**, *1*, 34–38.
88. Maruthi, R.; Jayakumari, C. SMS based bus tracking system using open source technologies. *Int. J. Comput. Appl.* **2014**, *86*, 44–46. [CrossRef]
89. Salim, A.; Ibrahim, I. Design and Implementation of Web-Based GPS-GPRS Vehicle Tracking System. *Int. J. Comput. Sci. Inform. Technol.* **2013**, *3*, 443–448.
90. Jay, L.; Reshul, D.; Sontakke, S.; Rahul, A. Scalable tracking system for public buses using IoT technologies. In Proceedings of the 2017 International Conference on Emerging Trends & Innovation in ICT (ICEI), Pune, India, 3–5 February 2017.
91. Lasers. Available online: https://www.explainthatstuff.com/lasers.html (accessed on 7 February 2019).
92. Laser Exposure. Available online: https://www.mcgill.ca/ehs/laboratory/radiation/non-ionizing-radiation/laser-exposure (accessed on 7 February 2019).

© 2019 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).