computation

Article

DeepFog: Fog Computing-Based Deep Neural Architecture for Prediction of Stress Types, Diabetes and Hypertension Attacks

Rojalina Priyadarshini [1,*], Rabindra Kumar Barik [2] and Harishchandra Dubey [3,*]

1 School of Computer Science and Engineering, KIIT Deemed to be University, Bhubaneswar 751024, India
2 School of Computer Application, KIIT Deemed to be University, Bhubaneswar 751024, India; rabindra.mnnit@gmail.com
3 Center for Robust Speech Systems, The University of Texas at Dallas, Richardson, TX 75080, USA
* Correspondence: priyadarshini.rojalina@gmail.com (R.P.); harishchandra.dubey@utdallas.edu (H.D.); Tel.: +1-469-618-2851 (H.D.)

Received: 13 October 2018; Accepted: 26 November 2018; Published: 4 December 2018

Abstract: The use of wearable and Internet-of-Things (IoT) for smart and affordable healthcare is trending. In traditional setups, the cloud backend receives the healthcare data and performs monitoring and prediction for diseases, diagnosis, and wellness prediction. Fog computing (FC) is a distributed computing paradigm that leverages low-power embedded processors in an intermediary node between the client layer and cloud layer. The diagnosis for wellness and fitness monitoring could be transferred to the fog layer from the cloud layer. Such a paradigm leads to a reduction in latency at an increased throughput. This paper processes a fog-based deep learning model, *DeepFog* that collects the data from individuals and predicts the wellness stats using a deep neural network model that can handle heterogeneous and multidimensional data. The three important abnormalities in wellness namely, (i) diabetes; (ii) hypertension attacks and (iii) stress type classification were chosen for experimental studies. We performed a detailed analysis of proposed models' accuracy on standard datasets. The results validated the efficacy of the proposed system and architecture for accurate monitoring of these critical wellness and fitness criteria. We used standard datasets and open source software tools for our experiments.

Keywords: fog computing; deep learning; deep neural network; stress prediction; diabetes mellitus; hypertension attack; smart health; connected health

1. Introduction

The world is growing industrially and the mortality rate has increased. However, at the same time, the number of lifestyle diseases has also increased. These diseases include type-2 diabetes, cardiac attack, hypertension attack and obesity. The major influencing factors behind these diseases are the type of diet, level of stress, lack of physical activities, and environmental conditions. In certain situations, the side effect of these diseases may lead to fatal conditions such as paralysis attack, cardiac arrest, irregular heartbeat, shortness of breath, chest pain, which require quick attention. The prevalence of diabetes is predicted to increase twice from 171 million in 2000 to 366 million in 2030 only in India [1–4]. The number of people with diabetes has risen from 108 million in 1980 to 422 million in 2014 [5]. According to a World Health Organization (WHO) report, diabetes mellitus is the foremost reason behind early blindness, kidney failure, heart attacks, stroke and lower limb amputation [6]. In 2014, in America, around 410,000 people died due to hypertension [7]. Therefore, it is a requirement to have a quick monitoring and early diagnosis system, which will have low latency, minimum response time and high accuracy. The Government is spending a huge amount of funding on Information Technology infrastructure-based health solutions and support services. Due to high computing facilities, health data collection, processing and analysis could be done faster and the extracted information could be transmitted to individuals as and when required. With these types of solutions, there exist some underlying issues related to data transmission latency, handling and processing huge amounts of data, the accuracy of the working solution. Many health care monitoring systems exist, which use cloud-based systems that store and analyze the bulk amount of data collected from different sources. It has some substantial benefits like scalability, the ability to store a large amount of data, low maintenance, deployment and service cost [8]. Bulk amounts of data transmission over the network may cause a delay in transmission, which in turn affects the overall performance of the system. Particularly, in applications like health monitoring, where real-time data transfer is an essential requirement, the system fails to perform correctly by avoiding network latency and a communication delay occurred through the generation of large amounts of data from IoT devices. Therefore, there is a requirement to reduce communication and network latency caused due to huge data transmission over cloud servers. Now, as a solution, researchers and industry personnel are going for a new computing paradigm known as Fog computing (FC). Fog computing was first named by Salvatore Stolfo [9] and was defined as an extended computing version of the cloud. It includes prominent features like facilitating networking, computing, infrastructure and storage support as the backbone for end-user computing. Along with this, it maintains latency constraints. According to CISCO, it is mentioned that "Fog computing is a distributed paradigm that provides cloud-like services to the edge of network" [10].

In all fog-based applications [2,3,11–18] an intermediary fog layer is built between the physical layer, where edge devices like the sensors deployed in smart health care devices [3,12,19], smart phones, and wearable devices like smart watches, bands from where the row data are collected and the top layers where cloud servers are present. The generalized architecture of fog computing is comprised of three levels [20,21] is depicted in Figure 1.

(1) **Physical Level:** It may be formed of physical devices like network switches, routers, smart phones, setup boxes, smart refrigerators, and vehicles. These devices are domain specific, depending on the application field. These might be wearable sensors, smart watches or a smart phone in the case of personalized health care monitoring systems. Otherwise, the devices may be smart refrigerators, an intelligent air conditioner in a smart home application. These may be resource constrained or resource intensive based on the application types. The interconnected devices, which are geospatially closer to each other form a virtual cluster (VC). When a new fog instance is admitted to this level, it may join any existing cluster according to the geographical proximity. Similarly, a fog instance may be disconnected from any cluster. The duties of these devices are to sense all of the happening events and transmissions of the same to its immediate upper layer. Herein, the FC paradigm where not every data packet from the physical device is submitted to the cloud, rather some amount of the required real time and latency specific analysis is carried out in this layer, which is sent to the fog layer for further processing.

(2) **Fog Level:** The middle level consists of the fog computing elements called fog nodes, the supporting hardware and software elements including all of the core elements of the network responsible for providing all of the networking support. It is a collection of edge devices such as routers, gateways, switches and access points. These devices are capable of handling all sorts of incoming data generated from the lower level and process them as such. Data may accumulate temporarily and the essential data may be sent out to the above layer for persistence storage.

(3) **Cloud Level:** The topmost level is the cloud layer, which is constituted of a cloud server and data centers. These are meant for bulk data processing and storage of computationally intensive data and applications.

In a traditional cloud computing (CC) architecture, the cloud server is always overloaded with user queries and applications, whereas in an FC architecture, access to the cloud server becomes more controlled and well utilized and has a better performance. The fog layer residing between the physical and cloud layers has some added advantages as it performs some extra tasks such as local storage, processing and some sort of mining in the edge devices. Therefore, it acts as a smart and intelligent gateway [22,23]. In most of the literature, researchers have applied an FC layer between the cloud and core network layers to overcome bandwidth latency, delays in network traffic thus, reducing the response time [24,25]. For all of these reasons, a fog layer can be used in health care applications to handle massive real time data generated from low level devices, which can continuously monitor the data and process the data to predict the stress types, emotional attacks and diagnose diseases like diabetes. Therefore, it is possible to design a system by which the affecting parameters are continuously monitored and guarded. The different stress levels can be predicted and if any risk is there, an early diagnosis and notification could be provided to the patients, close relatives and doctors as well in the case of an emergency. This system could be used by doctors and hospitals to facilitate services to patients who are located remotely. The end users of this service could be elderly persons or individuals who are physically disabled, live alone, need continuous monitoring or reside far away from healthcare service providers to manage their health conditions in a better way even in remote places. Some of the measured parameters, which affect hypertension are systolic blood pressure (SBP), diastolic blood pressure (DBP), total cholesterol (TC), blood glucose (BG) levels, heart rate (HR), and the level of stress and anxiety. These parameters are difficult to monitor regularly without the help of a hospital or clinic visits. The main contributions of this paper are:

- To propose *DeepFog* i.e., a fog based healthcare tele-monitoring system, which could provide services such as remote diagnosis of hypertension levels, as well as a prediction of hypertension attack based on features taken from user's health related data and the symptoms seen.
- To predict the stress level of a person, that helps the doctor to measure the risk of a hypertension attack related to the stress density.
- To build a deep neural network-based classifier to handle large amounts of user data collected from various sources, which can predict the risk of a hypertension attack.
- To build a deep neural network classifier, that can classify normal people and diabetic people in a fog environment by which doctors can be provided with quick information regarding the presence of diabetes mellitus in a patient to handle the patient properly.

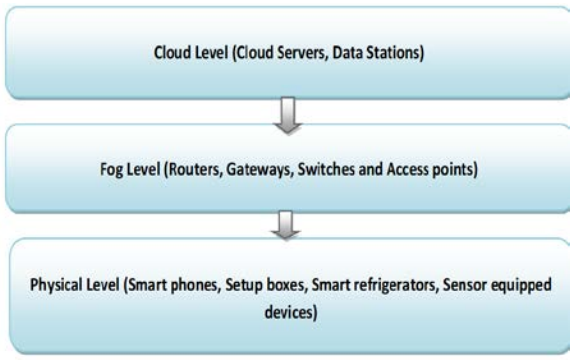
Cloud Level (Cloud Servers, Data Stations)
Fog Level (Routers, Gateways, Switches and Access points)
Physical Level (Smart phones, Setup boxes, Smart refrigerators, Sensor equipped devices)

Figure 1. General three level architecture of fog computing.

For the attainment of the stated objectives, an intelligent deep learning-based fog model is proposed, which can perform real-time monitoring and synthesis of user's health parameters to detect whether a person is diabetic. If the user is passing through any stress, then it can identify the type of the resulting stress. This result can help the doctors to handle the patient in a proper way. For this, a deep neural network is used to classify the stress types. The data used to predict the stress types are (1) electro dermal activity (EDA); (2) heart rate (HR); (3) arterial oxygen saturation (SpO2) and (4) temperature [26,27]. The health parameters chosen to predict the risk of hypertension are (1) systolic blood pressure (SBP); (2) diastolic blood pressure (DBP); (3) total cholesterol (TC); (4)

high-density lipoprotein (HDL); (5) low-density lipoprotein (LDL); (6) plasma glucose concentration (PGC) and (7) HR [28]. To detect diabetes mellitus, the data considered are (1) PGC; (2) DBP; (3) 2-h serum insulin (mu U/mL); (4) body mass index (BMI); (5) the diabetes pedigree function; (6) age in years [29–31]. These data are collected from the users and stored in the fog nodes for synthesis and analysis. The extracted information from the analyzed result is sent to cloud servers and in emergency situations, alerts are generated and sent to users, doctors, relatives, and caretakers of the concerned to make a decision regarding later actions. The details of the information are saved persistently in cloud storage for record keeping, which can be shared with Government institutes, domain experts, and doctors to further analyze the data.

This paper is organized as follows. Section 2 contains the related work about the use of fog computing in the healthcare domain to do an analysis of data, the existing tele-monitoring system used in healthcare. Section 3 presents a detailed description of the proposed system architecture. Section 4 provides experimental details, the carried out case studies along with the results, experimental setup, and performance evaluation. Conclusion and future work are depicted in Section 5.

2. Related Works

The related work has been categorized into three sections. Section 1 discusses the work, which has been done in the domain of FC in healthcare. The second section includes the ongoing research in deep neural networks meant for the classification of problems. The third section emphasizes the advancement in FC-based tele healthcare monitoring systems. Cisco defined fog computing as a distributed paradigm that provides cloud-like services to the edge of the network. They have been applied in various areas like the analysis of geospatial data [32–34], designing augmented reality-based gaming applications [35], smart city applications like vehicular network management [36,37] for smart parking, and sensor data analysis [38–43].

A resurgence of interest has been seen in last few years towards artificial neural networks, specifically deep learning has been used extensively after its spectacular success in the area of image classification, regression problems in time series data, and natural language processing [44–46]. This has been used in stock market analysis and prediction [47–49], breast cancer classification [50], classification of electrocardiogram signals [51], image classification [52], object recognition [53], medical image analysis [54], and time-series data analysis [55]. Deep learning, otherwise known as feature learning, has been well suited for deriving basic features from the input data and the obtained extracted features can be used to train a model in a better way. The objective of choosing a deep neural network has two intentions. (1) To create a robust model that can extract features from the raw heterogeneous input data. (2) To create a model that can handle high dimensional data. The achievement of these goals has been attained by the given case studies.

The works which are only focusing on health care domain are discussed in this paragraph. Gia [23] has extracted some features from ECG data based on a wireless body area network (WBAN) and tried to detect the heart rate from the ECG signals. They have used FC as a gateway for mining the data, storing them in a distributed manner for providing a notification service at the edge of the network. The raw signals were captured, pre-processed and taken through a wavelet transformation. Their target was to get a better network bandwidth utilization and provide low-latency real-time notifications. Dubey proposed a fog-based architecture for designing a tele-healthcare monitoring system to monitor patients of Parkinson disease. They used time series data containing ECG signals or speech data. The raw data were collected from sensors deployed in the patient's body. Some filters were used on the speech data to extract the loudness and frequency. Additionally, to increase the bandwidth utilization for better network latency, the processed data were compressed by using the GNU zip program. They have emphasized optimizing battery power consumption [24]. Monteiro [25] also employed a fog-based solution for obtaining disorder in clinical speech data of Parkinson disease. The raw data were collected from a wearable watch. They tried to process the speech data to extract features such as loudness, short time energy, the zero-crossing rate of speech data. Unlike the previous author, they have not used any specific algorithm, rather they used customized python coding for data extraction. Rahmani [22] proposed a smart e-health monitoring system, which issues an early alert for a patient before any emergency arises. After capturing the noisy data, some filters were used to remove the noisy patterns; the data were compressed using both lossy and lossless compression methods to deal with communication latency. Each of these data analyses were done, which can make decisions about notifications. They have also used a public key encryption scheme to protect the locally stored data in the edge devices. Ahmad [56] exploited FC for designing a customer-centric healthcare application, which can act as a personal care taker giving notifications according to a customer's request. The health data acquisition was done by a smart phone with integrated sensors. The data were collected from different sources and they used an intermediate fog layer between the end user and the cloud to make the data more secure [57].

3. *DeepFog* Architecture

The proposed framework, i.e., *DeepFog* is comprised of three layers namely the user layer, the smart gateway fog layer and a cloud layer. The user layer is an integration of all smart data capturing devices to collect health-related feature data to diagnose diabetes and hypertension attacks, bio-signals to predict the stress types. The collected data are then transmitted to the middle fog layer to do real time processing, diagnosis and prediction. With no delay, the processed information, analyzed health related results, and diagnosis reports are sent to the cloud layer. Thereafter, these data are shared with domain experts, doctors, health-care related persons, and relatives who can take precautionary measures and immediate action in emergency situations.

The main responsibility of this layer is to accumulate all of the heath related parameters required to classify the stress types, to detect whether a user is diabetic and to predict the risk of hypertension attack. The health-related attributes are (1) SBP; (2) DBP; (3) HDL; (4) LDL; (5) PGC; (6) HR; (7) 2-h serum insulin; (8) BMI; (9) the diabetes pedigree function; (10) age in years; (11) EDA; (12) SpO2. Other data, which have an impact on a related attack include the surrounding, current physical activity, and Global Positioning System (GPS) data. All of these data are collected by the smart phones, wearable smart devices connected to the user's body. The sensors embedded in these devices are able to collect the data.Figure 2 represents the overall framework of the proposed model, where as Figure 3 presents the sequence diagram which gives an overview of the work flow of individual functions of the entire framework.

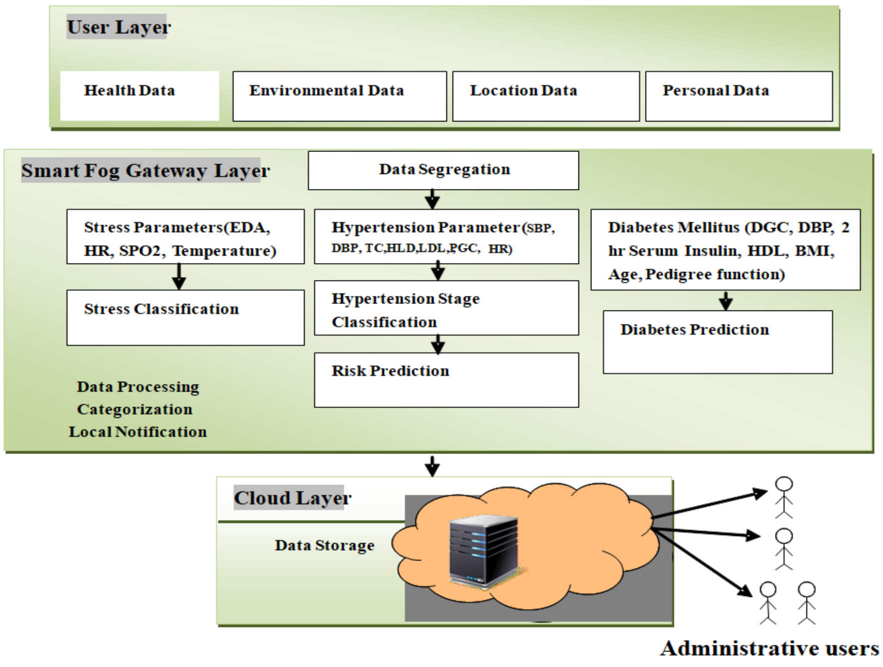

Figure 2. The proposed *DeepFog* framework has a middle layer that works as a smart gateway, residing between the user and the cloud layer. The user layer is an integration of all smart data capturing

devices to collect health-related feature data to diagnose diabetes and hypertension attacks, bio-signals to predict the stress types. Administrative users have access to maintain avariety of datasets stored in the cloud layer.

To validate the model, three case studies were undertaken. The first case study was carried out to find out the behavioral details of the patient. This provided information of whether a patient is in a relaxed state or is stressed. If she is stressed, then the type of stress she is going through. The second case study focused on detecting whether she is suffering from type-2 diabetes. The reason behind this is that the treatment of a normal person is different from that of a diabetic person. The third case study was taken to do a risk analysis of a hypertension attack.

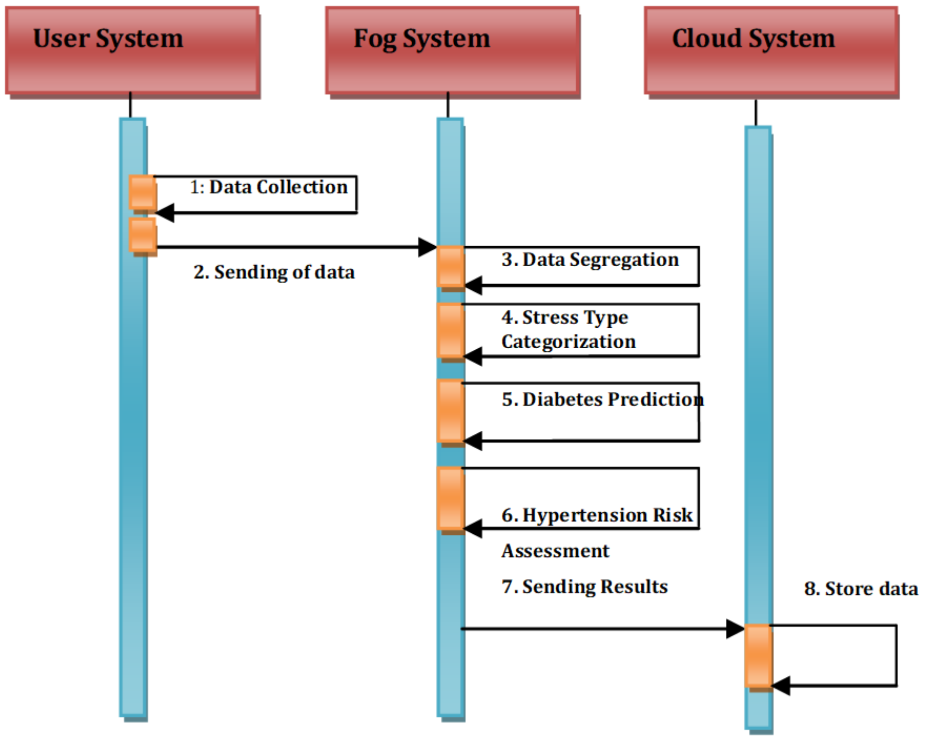

Figure 3. The proposed System model of the *DeepFog*.

3.1. Case Study 1—Categorization of Stress Types

While monitoring a patient remotely, especially if it is a case of hypertension, it is very much essential for anyone concerned to judge the stress type of the person. 'Stress' is a major parameter and plays a crucial role in knowing a patient's condition, as well as for making a decision about his medication. However, if the stress type is also known before, then it will be more helpful for the experts to make a decision. For example, if the stress type is physical, after resting for some time, it will be normalized. However, for the other two types of stress, there must be some medication plans to normalize the stress level. Therefore, a classification of the stress type is needed. For categorization of stress types, the dataset used was taken from data available at www.utdallas.edu. Birjandtalab [58] used the data for the visualization of neurological status. This data contains the bio-signals of 20 students, where readings were recorded of EDA, 3-dimensional accelerometer, temperature, HR, SpO2. The data were recorded while going through four neurological states. The states were(1) physically stressed (PS); (2) cognitively stressed (CS); (3) emotionally stressed (ES); (4) relaxed state. In this work, we have considered all four states. Pre-processing was carried out on the data to segregate the equal state data to merge them together. Then, all data samples of 20 subjects were merged to have a total sample space. The data were leveled. CS was leveled as '0', ES was leveled as '0.5' and PS was leveled as '1'. After assigning the levels to the corresponding data instances, the data sample was divided into two sets. Eighty percent of the total sample was taken as training data and the remaining 20% wastaken as test data. The data went through a normalization process according to the expression given in Equation (1), to bring them in the range of 0 to 1.

$$\text{IP norm} = \frac{\text{IP} - \text{IP min}}{\text{IP max} - \text{IP min}} \tag{1}$$

where IP norm represents the normalized value of the input parameter IP, IP max and IP min are the maximum and minimum value present for the original IP. Figures 4–7 represent the distribution of the parameters taken to detect the stress types. These graphs below show a clear variation of the features according to the stress types except for SpO2. For the SpO2 case, the area enclosed for three stress types is overlapping, so this feature can be discarded while doing the prediction. However, apart from this, all others are capable of obtaining the proper class of stress.

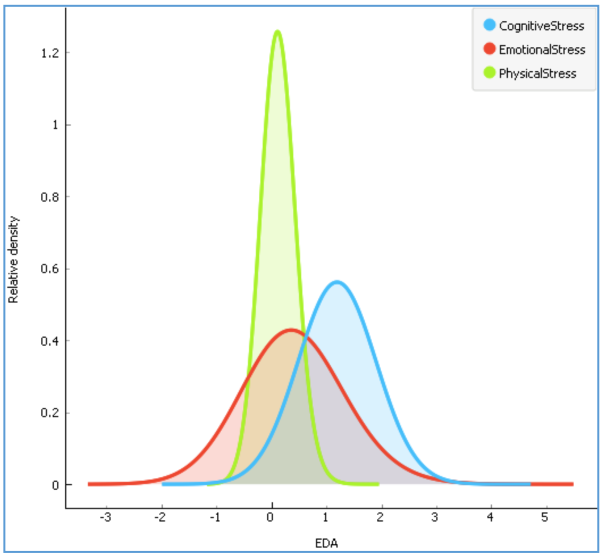

Figure 4. The distribution of according to three stress types.

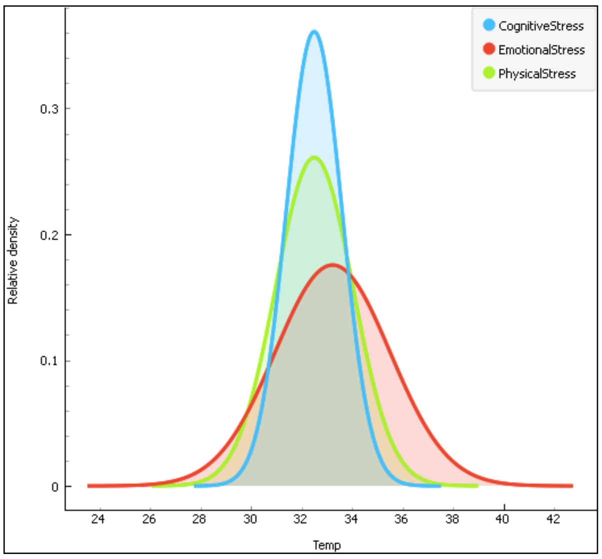

Figure 5. The distribution of the temperature data(Temp) according to the three stress types.

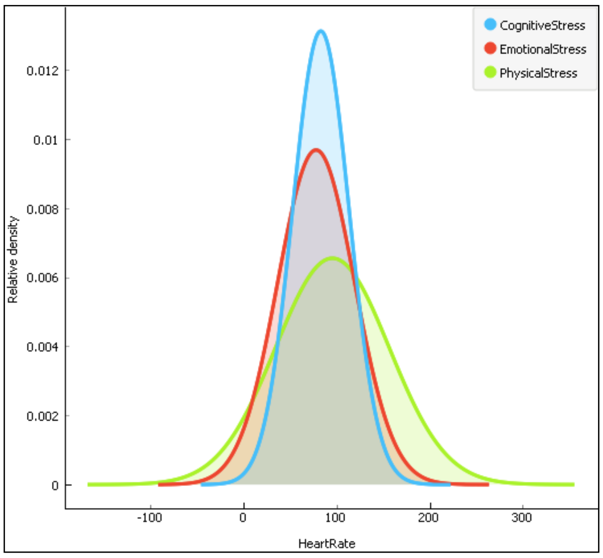

Figure 6. The distribution of the heartrate data according to the three stress types.

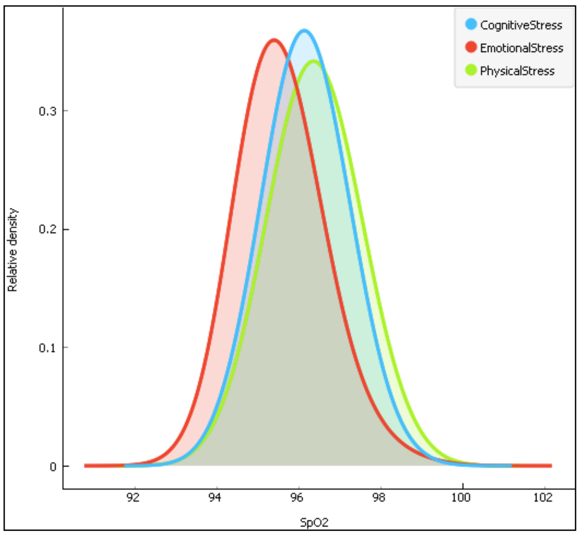

Figure 7. The distribution of the arterial oxygen saturation data (SpO2) according to the three stress types.

3.2. Case Study 2—Prediction of Diabetes

The treatment of a diabetic person is different than that of a non-diabetic person. Due to the side effects generated, any drug must be handled for a diabetic person. There exist some recent traditional methods that are used to do an early prediction of diabetes. Liu and Gao have employed a method based on differential entropy on gene expression data for the early detection of diabetes [59]. Wu has exploited an improvised *K*-Means algorithm and logistic regression to design a prediction model for type-2 diabetes mellitus [60].For a health expert, knowledge about being or not being diabetic will be a great help. Therefore, there is a requirement for detecting it before any decision. For predicting diabetes, the data used are (1) the plasma glucose concentration;(2) diastolic blood pressure;(3) 2-h serum insulin (mu U/mL);(4) body mass index;(5) the diabetes pedigree function;(6) age in years [29–31]. These data can be collected using health sensors embedded in smart phones. For experimental purposes, a standard dataset from University of California learning repository (C.L. Blake, C.J. Merz) was taken and used. All of these data samples were normalized by following Equation (1). Figures 8–11 show the distribution of four major features (2 h serum insulin, DBF, PGC and DBP) to predict diabetes, which shows a clear isolation of features according to their classes.

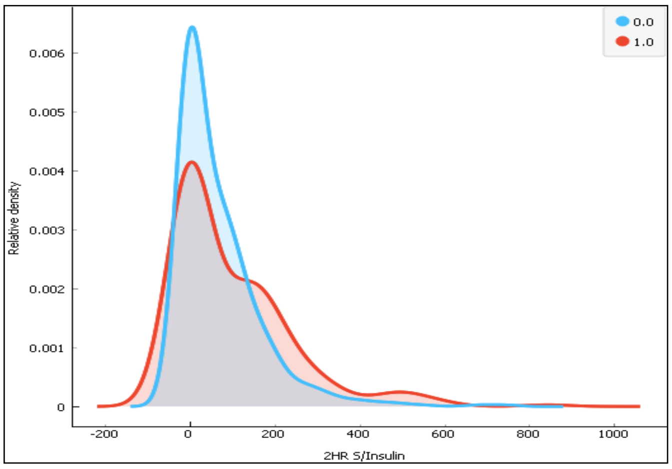

Figure 8. The distribution of 2-h Serum insulin for the two classes (0-diabetes and 1-non-diabetes).

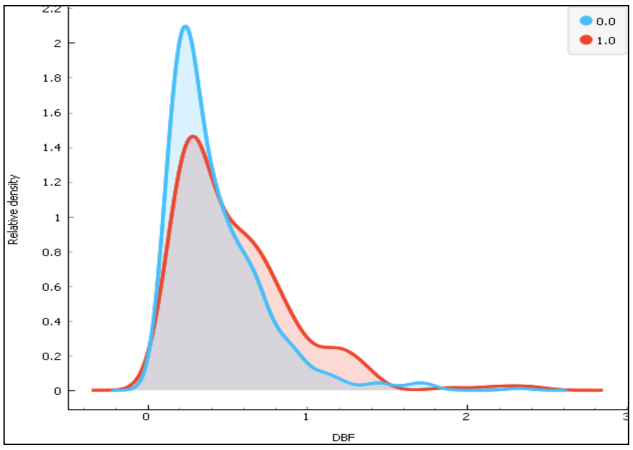

Figure 9. The distribution of the diabetes pedigree function(DBF) for the two classes.

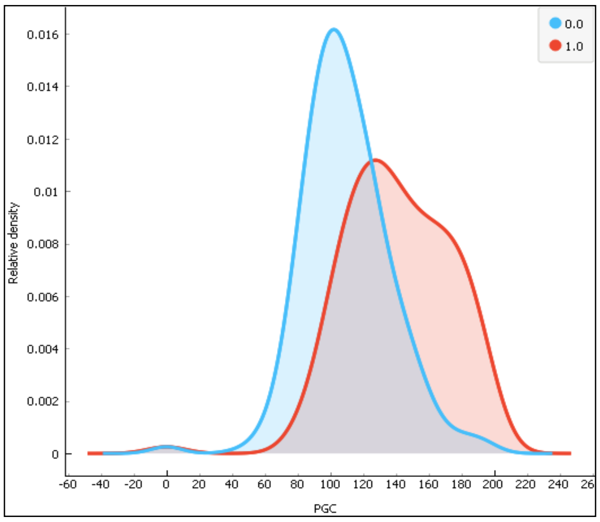

Figure 10. The distribution of the plasma glucose concentration (PGC) for the two classes (0-diabetes and 1-non-diabetes).

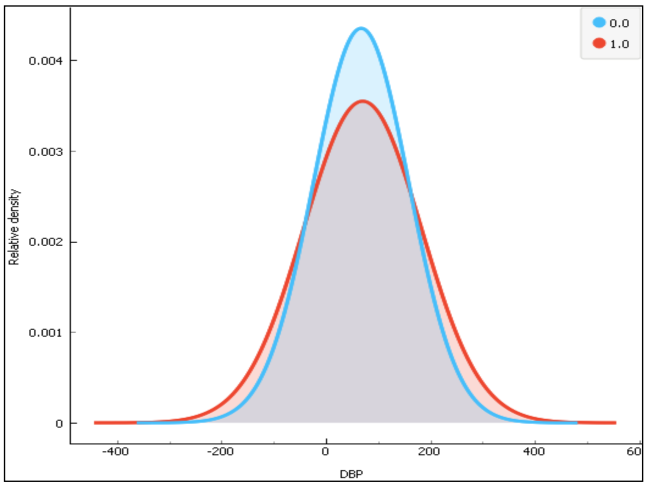

Figure 11. The distribution of diastolic blood pressure (DBP).

3.3. Case Study 3—Risk Assessment of Hypertension Attack

Hypertension is a type of infirmity, which may lead to some other critical and life-threatening diseases like cardiac failure, thickening of the heart muscle, and renal failure. It is diagnosed as a diastolic or systolic blood pressure of more than 90 mm Hg or 120 Hg respectively for at least two measuring instances. However, a sudden rise or fall of these values could cause the risk of a hypertension stroke by which one person may move to paralysis or comma like dangerous situations. Risk assessments of this disease are dependent on a multitude of factors including a few transient ecological circumstances, which may artificially elevate blood pressure readings [61]. The identifying factors, which influence these situations are grouped into five types of data, which are depicted in Table 1. The effect of these identifying factors is also shown in the table and clinical health information such as SBP, DBP, HDL, LDL have a larger influence on hypertension than location and time [62].

Table 1. Types of data used, their attributes, and their effect.

S.No	Type of Data	Attribute	Description and Effect	
1	Clinical information of health data	SBP, DBP, HDL, LDL, triglycerides, cholesterol, micro-albumin, urine albumin-creatinine ratio, heart rate	These data can be captured by body sensors and have a high influence on hypertension attack.	↑
2	Personal information	Height, weight, age, gender, the presence of disease in family history, smoking habits	Body mass index can be calculated, the presence of obesity is detected, the attack is also dependent on positive pedigree family history	
3	Behavioral data	Stress level, type of stress, anxiety level, level of discomfort	Presence of stress may increase the risk of hypertension attack	**Effect of factors**
4	Surrounding Data	Temperature, humidity, air quality	If temperature and humidity component have more than normal, the chances of attack will be more	
5	GPS data	Location, time	The blood pressure reading will be more in high altitude regions and cold regions.	

The blood pressure measurements are classified into five stages according to the reading values of DBP and SBP parameters such as normal, the pre-hypertension stage, hypertension stage-1, hypertension stage-2 and the hypertensive crisis stage, which are depicted in Table 2 [63]. Patients belonging to hypertension stage-1, stage-2 and hypertensive crisis were the major portion of samples, which have a high probability of being the victim.

In this work, for assessing the risk of hypertension and to reduce the communication delay after training the deep neural network, during a real time prediction of hypertension attack, user requests were divided into two groups: (i) the high probable victim group and (ii) the low probable victim group. A priority was set for user requests that fall into the high probable group and these users were served first. This is because the high probable victim group is more susceptible to a hypertension stroke compared to the low probable victim group. Algorithm 1 was used to isolate the high probable victim group from the low probable victim group so that a higher priority would be given to serve the high probable victim group. For classifying the high probable and low probable victims, a rule has been set according to the data provided in Table 2. The intention behind filtering the high probable instances was to reduce communication latency.

Table 2. Categorization of stages of hypertension.

Stage of Hypertension	SDP Reading	DBP Reading
Normal	Less than 120	Less than 80
Pre-hypertension	$120 \leq \text{SDP} \leq 139$	$80 \leq \text{DBP} \leq 89$
Hypertension Stage-1	$140 \leq \text{SDP} \leq 159$	$90 \leq \text{DBP} \leq 99$
Hypertension Stage-2	$\text{SDP} \geq 160$	$\text{DBP} \geq 100$
Hypertensive Crisis	$\text{SDP} \geq 180$	$\text{DBP} \geq 110$

Algorithm 1: Separation of high probable victim group and low probable victim group

```
Input: S = {S1, S2, ......, Sn} where S is the set of total testing sample data
    1. Obtain the DBP and SDP value of Si; where Si represents the data of 'ith' instance out of
       Total sample set S
    2. If (DBP of Si >= 90 && SDP of Si >= 140)

       a.   If (Detect_Diabetes() is true for Si)
       b.   If (Predict_Stress() does not return Relax)

    3. Return Si;
    4. Assign Si to high probable victim group
       (End of if)
    5. Else
    6. Assign Si to low probable victim group.
Output: High probable victim group
```

Here, *Detect_Diabetes*() is the method to check diabetes mellitus stated in case study 2. It returns true or false. The *Predict_Stress*() method returns the mental state of a person. The values returned are the type of stress (cognitive, physical or emotional) or it may be a relaxed state stated in case study-1.

4. Experimental Setup, Results and Evaluation:

This section of the paper gives a representation of the experimental setup, implementation details, results and performance evaluation. This has been further divided into four subsections named as (1) setting up the fog-based system architecture;(2) data collection and pre-processing;(3) constructing a deep neural network model;(4) performance validation of the computing model.

4.1. Setting Up the Fog-Based System Architecture

In line with the suggested model discussed in Figure 3, the top most layer is a cloud layer configured with a set of open source software. For setting up the cloud storage, 'Owncloud' was used that was connected to the Apache web-server. To configure it, PHP and MySQL were needed in the background. The cloud server was deployed on Cent OS7. The database used in MySQL was MariaDB. This led to the persistent storage of the processed results forwarded from the fog network. MariaDB is a fork of MySQL, which is commonly used by a Linux distribution-like CentOS. The fog layer was comprised of three virtual machines (VM)that are considered as individual fog nodes. To maintain heterogeneity among the fog nodes, three VMs were made to run on different operating systems—two on Windows and one on Linux. For this, a Mininet emulator has been used. One virtual machine was responsible for storing data. The code for Algorithm 1 is written here, which is meant for separating the high and low probable victim groups. One was installed with the Deep learning tool Dl4j (deep learning for Java), responsible for the data analysis. This is further discussed in detail in Section 4.4. We have not captured data from sensors or any IoT devices in this work. The user interface was not designed fully through which the data could be captured from IoT devices or smart phones. For experimental purposes, we have used relevant and similar data, which are available. The used architecture is represented in Figure 12.

4.2. Data Collection

The details of the data used for carrying out the case studies are presented in Table 3. The data were collected from the sources provided in Table 1. For stress classification and diabetes prediction, the readily available datasets were used. However, for predicting the risk of a hypertension attack, a data set was prepared by extracted the parameters taken from the three datasets.

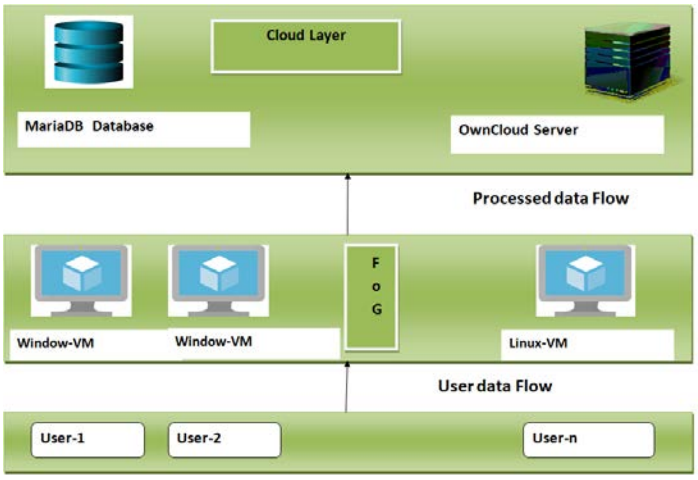

Figure 12. The fog-based system architecture used to deploy the *Deep-fog* system.

Table 3. Datasets used, their attributes, and the source of data.

Case Study Description	Data Attributes Used	Source of Data
Stress classification	EDA, HR, SpO2,temperature, 3-dimensional accelerometer data	www.utdallas.edu/~{}nourani/Bioinformatics/Biosensor_Data
Type-2 diabetes detection	PGC, DBP, 2-h serum insulin (mu U/mL) 4), body mass index (BMI), diabetes pedigree function, age in years	https://archive.ics.uci.edu/ml/datasets/Diabetes
Hypertension risk assessment	SBP, DBP, total cholesterol (TC), HDL, LDL, PGC and HR	http://archive.ics.uci.edu/ml, https://archive.ics.uci.edu/ml/datasets/Diabetes, https://archive.ics.uci.edu/ml/datasets/ChronicKidneyDisease, www.utdallas.edu/~{}nourani/Bioinformatics/Biosensor_Data

For preparing the training data set for a hypertension attack, the data was created after combining some selected features taken from the three datasets, which were the diabetes dataset, the stress dataset and the kidney disease dataset. The kidney disease dataset contained25 attributes. Among them, one attribute was hypertension, which contained a yes or no value indicating the presence of hypertension. Among the 24 attributes, some of the selected feature attributes were chosen. For selecting the feature, a simple technique was used. Here, the data were clustered into two groups according to the presence of a yes or no value for the hypertension field using a *K*-means clustering algorithm. Then, the data clusters were analyzed to find the distribution of the contributing attributes in the formed cluster. If the data of a certain attribute was found to belong to the wrong cluster, then that attribute was not taken into the training of the hypertension risk assessment. The *K*-means algorithm was used for leveling data instances.

4.3. Data Visualization andExploratory Data Analysis

For doing the primary pre-analysis and visualization of the data, 'Orange 3.4.4' was employed [64]. 'Orange' is an open source, 'Python' based tool, which is a data visualization, machine learning and data mining tool kit. It was applied for carrying out explorative data analytics and interactive data visualization jobs. It is a Python library that is used for performing data analytics tasks. Figures 13 and 14 show the distribution of DBP and SBP data values belonging to two of the clusters. As they do not have any interfering values, these features have more impact on the formed clusters. Similarly, the data values plotted for RBC and potassium contents are presented in Figures 15 and 16 respectively. These sets of values are interfering with each other. Therefore, these are the weak features for segregating the positive and negative hypertension data.

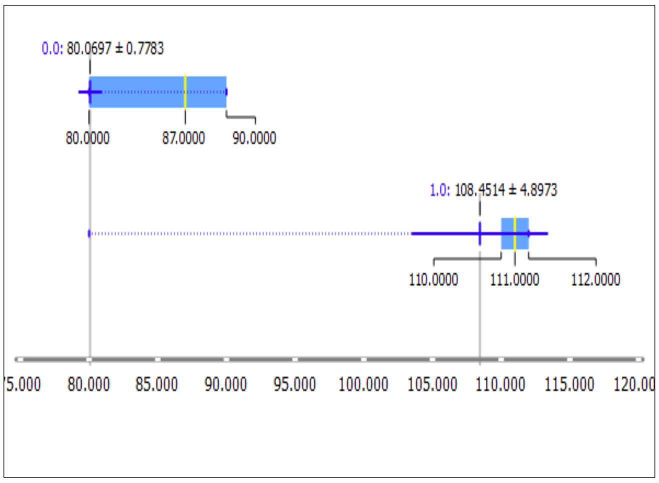

Figure 13. Data distribution of diastolic blood pressure (DBP in a box-plot, showing the range data; the mean value after clustering the data using the *K*-means clustering technique. Here, 0.0 represents the cluster for the non-hypertension class and 1.0 represents the hypertension class. It can be seen that the two sets of data are non-overlapping in nature.

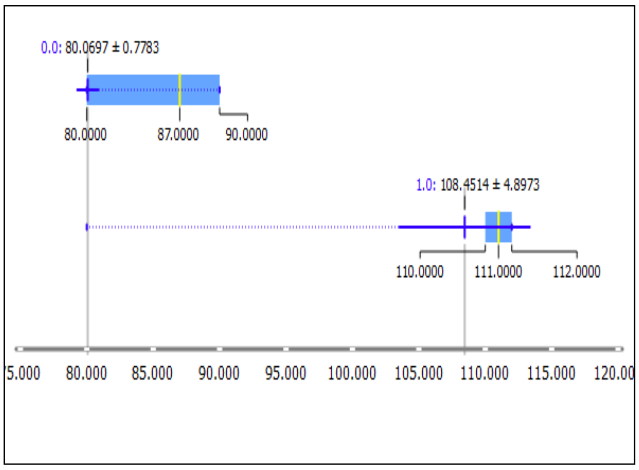

Figure 14. Data distribution of Systolic Blood Pressure SBP in a box-plot showing the range of the data, its mean value after clustering the data using the *K*-means clustering technique. Here, 0.0 represents the cluster for the non-hypertension class and 1.0 represents the hypertension class.

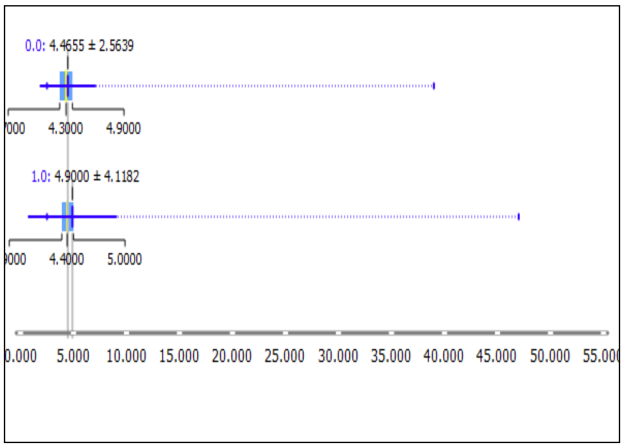

Figure 15. Data distribution of potassium box-plot, showing the range of the data, its mean value after clustering the data using the *K*-means clustering technique. Here, 0.0 reresents the cluster for the non-hypertension class and 1.0 represents the hypertension class. It can be seen that the two sets of data are overlapping in nature.

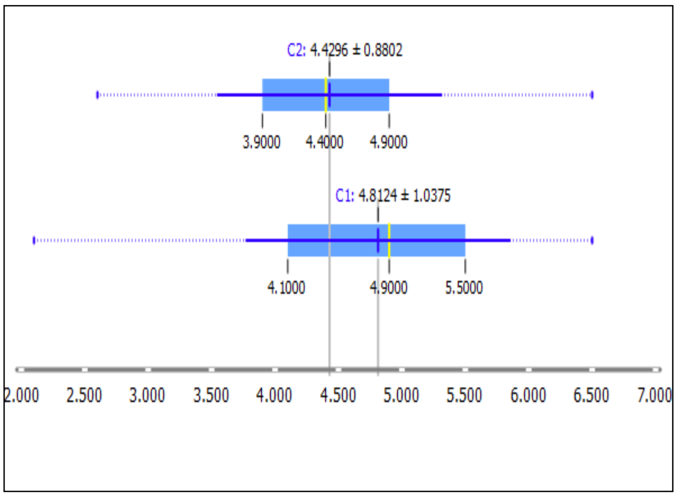

Figure 16. Data distribution of the Red Blood Cell RBC count in a box-plot showing the range of the data, its mean value after clustering the data using the *K*-means clustering technique. Here, 0.0 reresents the cluster for the non-hypertension class and 1.0 represents the hypertension class. It can be seen that the two sets of data are overlapping in nature.

4.4. Deep Neural Network Model

To build the prediction model, deep learning has been adapted. In our context, it can automatically explore the representations from the raw inputs. The proposed model can be described as follows. Given a training set composed of n features represented by T = {(I_i, O_i)}, i = 1, 2, ..., n, where $I_i \in R^T$, where I is a feature vector of dimension T, $O_i \in$ [1,C] is the corresponding class label of I_i. The neural network specifies a nonlinear relationship between the two components h_i and h_i + 1 through a network function, which has a form

$$h_i + 1 = \alpha(Wh_i + \delta) \tag{2}$$

where α is an activation function and the matrix W and δ are the model components or parameters that need to be adjusted. h_i + 1 and h_i can be represented as layers. Here, α is either aReLU or SoftMax function. In the case of a single layer neural network, there is only one layer. If they are augmented with multiple layers and adapted advanced learning strategies, they form a deep neural network (DNN). For a prediction model indulged with a prediction function, $O = f(I)$ can be built by stacking multiple network functions like h_1, h_2, ..., O given in Equations (3)–(5).

$$h_1 = \alpha_l(W_1 l + \delta_1) \tag{3}$$

$$h_2 = \alpha_l(W_2 h_1 + \delta_2) \tag{4}$$

$$O = \alpha_l(W_l h_l + \delta_{l-1}). \tag{5}$$

In the above equation, '*l*' is the number of layers. In a given training data set,$T = \{(\text{In}, \text{On}), i = 1, 2, \ldots, N\}$ contains inputs and targets and an error function $E = (Y_n, O_n)$, that computes the difference between output Y_n and target O_n. The model parameters $P = \{W_1, W_2,$ ...., $W_l, \delta_1, \delta_2,$ ...., $\delta_l\}$ have to be adjusted to minimize the net error. This error function is depicted in Equation (6).

$$\text{Min } (P)\left\{\sum_{i=1}^{n} E = (Y_n, O_n)\right\}. \tag{6}$$

DL4*j* is used for building a deep neural network for validating the proposed model. It uses a simple deep neural network having more than one hidden layer. DL4*j* is a distributed deep learning framework used for creating, training and deploying a deep neural network model. It is based on an object oriented language 'Java', thus it is platform independent. It is well suited for a big data environment as it can handle massive amounts of data in a very reasonable time frame. It could be integrated with other parallel and distributed programming environments like 'Hadoop' or'Spark' and it also supports GPU computing [65,66]. It has several built in classes and methods to create artificial deep models and to handle different types of data such as CSVRecordReader, ImageRecordReader, JacksonRecordReader, VideoRecordReader classes for handling input data like csv files, image data, JSON data and video data respectively. In this work, CSVRecordReader has been used to deal with csv data. A 'transform' class was used to normalize the data. 'MultiLayerConfiguration' was the most important class, which was used to configure the network. The numbers of the inputs were 7, 5 and 8 and the outputs were 2,4 and 2 respectively for diabetes, stress and hypertension prediction. The configuration details are provided in Table 4. The number of instances was increased by randomly combining the present instances and made 5000 instances in the case of diabetes and hypertension samples to effectively train the deep neural network. The stress dataset was large enough, so it is used as it is. From the 5000 instances, 3500 were used as the training set and rest was used for testing purposes. The 'pretrain ()' value was set to be '0', as this network used a supervised learning scheme. For unsupervised autoencoders, this 'pretrain ()' parameter must be set to 1. The dense layers have three inputs each. The learning rate parameters were fixed for the stated values depicted Table 4 using a hit and trial method. These parameters were fixed after executing several iterations and finalized against minimum error values. The input layer activation function was chosen as Tanh() for diabetes and hypertension whereas it was ReLU() for the stress data samples. The activation function in the output layer was SoftMax, through this the number of output nodes was set to the number of classes. The 'iteration' parameter was set as 1000. The 'iteration' parameter in DL4j was one update of the neural network model parameter. It was not the same as the epoch of conventional NN. An epoch means the number of iterations to complete a pass for all instances of the dataset. The experiment was conducted to set the 'iteration' parameter as 1000, then incremented by 1000 each time. However, there was no significant difference between the average accuracy value over 10 numbers of runs for 5000 and 6000 iterations.

4.5. Performance Evaluation

Table 5 represents the performance matrix of the proposed model for the diabetes data set, where as Tables 6 and 7 are presenting the performance matrix of stress classification and hypertension attack. In DL4j, the performance of a deep neural network was interpreted by a library class named as 'evaluation', contained a predefined method called get FeatureMatrix(). This method can be called after training the neural network on the test dataset. It produces a confusion matrix formed out of four evaluating parameters, which were accuracy, precision, recall and F1 score. Accuracy is the simple ratio of correctly classified instances to the total number of instances. Precision is the ratio of correctly predicted positive instances to the total predicted positive instances. Recall is the quantity calculated from the correctly predicted positive samples to total instances in the original class. The F1 score is a weighted average value of the precision and recall values. The expressions for precision and recall are given in Equations (2) and (3).

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{8}$$

where TP is true positive, FP is false positive, and FN is false negative values of the given sample. It can be observed from the result that, for the 5000 iteration value, the proposed model has achieved almost 92% accuracy for the stress dataset, 88% accuracy in hypertension cases, whereas for diabetes it is getting an accuracy of 84%. The resultant precision value, recall, and F1 score are also competent enough to validate the proposed model.

Table 4. Configuration of the deep neural network.

Dataset	No. of Inputs	No. of Outputs	No. of Hidden Layers	Learning Rate Parameter	Input and Hidden Layer Activation Function	Output Layer Activation Function
Diabetes	7	2	2	0.1	Tanh	SoftMax
Stress	5	4	3	0.11	ReLU	SoftMax
Hypertension	8	2	2	0.13	Tanh	SoftMax

Table 5. Performance matrix for diabetes mellitus prediction.

Performance Matrix for Diabetes Dataset							
Iteration = 1000, Learning Rate Parameter = 0.1				**Iteration = 5000, Learning Rate Parameter = 0.1**			
Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
0.7398	0.7428	0.7173	0.6465	0.7812	0.7678	0.8173	0.6321
0.7872	0.7320	0.6989	0.6000	0.7974	0.7458	0.6889	0.6122
0.7770	0.7523	0.7496	0.6703	0.8387	0.8523	0.7496	0.6898
0.7361	0.7189	0.6900	0.5896	0.8411	0.7189	0.6990	0.5987
0.7955	0.7761	0.7774	0.7120	0.8199	0.8773	0.7874	0.7234
0.7658	0.7416	0.7323	0.6480	0.8452	0.8428	0.7323	0.6346
0.8578	0.8861	0.7865	0.6632	0.8623	0.8873	0.8165	0.6789
0.7807	0.7825	0.7413	0.6667	0.8383	0.7897	0.7413	0.6782
0.8387	0.7243	0.6916	0.6900	0.8655	0.7255	0.6916	0.6989
0.7955	0.7761	0.7774	0.7120	0.7989	0.7773	0.7888	0.7121
0.7658	0.7420	0.7156	0.6182	0.8231	0.8432	0.7234	0.6261
0.7918	0.7643	0.7553	0.6710	0.8490	0.7655	0.7678	0.6711
0.8123	0.8302	0.6767	0.6547	0.8767	0.8314	0.6897	0.6245
0.8545	0.8489	0.6547	0.6400	0.8786	0.8501	0.6676	0.6534
0.7998	0.6756	0.6511	0.6100	0.7889	0.6768	0.6456	0.6298
0.7876	0.8156	0.6978	0.6001	0.8456	0.8168	0.7123	0.6122
0.8356	0.7889	0.6900	0.6032	0.8891	0.8100	0.7898	0.6977
0.8321	0.8213	0.6763	0.5432	0.8134	0.8225	0.6932	0.5771
0.8563	0.8600	0.6789	0.6523	0.8651	0.8612	0.6897	0.678
0.8490	0.8345	0.6751	0.6800	0.8989	0.7578	0.6451	0.6898
Average value over 20 iterations							
0.802945	**0.7807**	**0.71169**	**0.64355**	**0.840845**	**0.8010**	**0.726845**	**0.65593**

Table 6. Performance matrix for the stress classification model.

Performance Matrix for Stress Dataset							
Iteration = 1000, Learning Rate Parameter = 0.1				**Iteration = 5000, Learning Rate Parameter = 0.1**			
Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
0.871	0.8539	0.8173	0.7677	0.8822	0.8551	0.8182	0.7668
0.9184	0.8421	0.81	0.7112	0.9297	0.8434	0.8108	0.712
0.9082	0.8534	0.8707	0.7715	0.9194	0.8548	0.8714	0.7804
0.8673	0.83	0.78	0.6808	0.8785	0.8312	0.7889	0.6896
0.8267	0.8872	0.8685	0.7932	0.8379	0.8881	0.86938	0.8019
0.897	0.8527	0.8323	0.7492	0.9082	0.8538	0.8332	0.7559
0.989	0.8873	0.8865	0.7844	1.0002	0.8885	0.8866	0.7845
0.9119	0.8805	0.8413	0.7879	0.9233	0.8819	0.8419	0.7886
0.9587	0.8223	0.7916	0.8112	0.9699	0.8313	0.7925	0.8200
0.7967	0.8872	0.8774	0.8332	0.8079	0.8883	0.8782	0.8243
0.877	0.8531	0.8156	0.7394	0.8882	0.8547	0.8244	0.7472
0.923	0.8754	0.8553	0.7922	0.9342	0.8767	0.864	0.7923
0.9135	0.9413	0.7767	0.7759	0.9247	0.9493	0.7776	0.7761
0.8557	0.91	0.7547	0.7612	0.8669	0.9115	0.7625	0.7609
0.931	0.7867	0.7511	0.7312	0.9422	0.7965	0.7600	0.7316
0.9188	0.9267	0.7978	0.7213	0.932	0.9284	0.7983	0.7214
0.9356	0.9	0.79	0.7244	0.9368	0.9011	0.7988	0.7255
0.8933	0.9324	0.7763	0.6644	0.9045	0.9336	0.784	0.66441
0.9683	0.9101	0.7789	0.7735	0.9783	0.9113	0.7798	0.77437
0.9502	0.9456	0.7751	0.8012	0.9617	0.9468	0.7760	0.8013
Average value over 20 iterations							
0.905565	**0.8918**	**0.812355**	**0.75875**	**0.916335**	**0.881315**	**0.815824**	**0.760954**

Table 7. Performance matrix of the hypertension risk assessment model.

Performance Matrix for Hypertension Dataset							
Iteration = 1000, Learning Rate Parameter = 0.1				**Iteration = 5000, Learning Rate Parameter = 0.1**			
Accuracy	**Precision**	**Recall**	**F1 Score**	**Accuracy**	**Precision**	**Recall**	**F1 Score**
0.8398	0.8339	0.8173	0.7677	0.8510	0.9151	0.8182	0.7668
0.9184	0.8321	0.8100	0.7112	0.9297	0.9134	0.8108	0.7120
0.8782	0.8334	0.8707	0.7715	0.8894	0.9148	0.8714	0.7804
0.8673	0.7800	0.7801	0.6808	0.8785	0.8612	0.7889	0.6896
0.8267	0.7872	0.8685	0.7932	0.8379	0.8681	0.86938	0.8019
0.8970	0.7627	0.8323	0.7492	0.9082	0.8438	0.8332	0.7559
0.9890	0.8873	0.8865	0.7844	1.0002	0.9685	0.8866	0.7845
0.8819	0.8805	0.8413	0.7879	0.8933	0.9619	0.8419	0.7886
0.8507	0.8223	0.7916	0.8112	0.8619	0.9113	0.7925	0.8200
0.8267	0.8872	0.8774	0.8332	0.8379	0.9683	0.8782	0.8243
0.8870	0.8531	0.8156	0.7394	0.8982	0.9347	0.8244	0.7472
0.9030	0.8754	0.8553	0.7922	0.9142	0.9567	0.864	0.7923
0.8243	0.9413	0.7767	0.7759	0.8355	0.9700	0.7776	0.7761
0.8557	0.91	0.7547	0.7612	0.8669	0.9915	0.7625	0.7609
0.8310	0.7867	0.7511	0.7312	0.8422	0.8765	0.7600	0.7316
0.8888	0.8157	0.7978	0.7213	0.902	0.8974	0.7983	0.7214
0.8456	0.7891	0.7900	0.7244	0.8468	0.8702	0.7988	0.7255
0.8933	0.8217	0.7763	0.6644	0.9045	0.9029	0.7840	0.66441
0.8675	0.8651	0.7789	0.7735	0.8775	0.9463	0.7798	0.77437
0.8502	0.8556	0.7751	0.8012	0.8617	0.9287	0.7760	0.8013
Average value over 20 iterations							
0.871105	**0.841015**	**0.812355**	**0.75875**	**0.881875**	**0.920056**	**0.815824**	**0.760954**

The results were compared with some similar works done by various researchers in the last few years. The current work of stress classification was weighted against some existing models.

The parameters on which the works were evaluated are the types of tools used for the work, the objective of the work, the prediction accuracy percentage, the input data and the use of a fog environment. It has been presented in Table 8. The job of type-2 diabetes mellitus detection was compared on the basis of the methods used for the work, the accuracy percentage and the exploitation of cloud or fog in the reported work. This is shown in Table 9. The early detection of a hypertension attack is evaluated with other state of the art research on the basis of the domain application, the purpose of the research, the use of cloud storage (CS), fog environment (FE), and the prediction model (PM), as shown in Table 10.

Table 8. Comparative analysis of stress classification.

Author	Tools Used for Application Development	Purpose	Accuracy of Prediction Model	Data Used	Whether Used in Fog Environment
M.M. Sami et al. (2014) [67]	Support vector machine	Stress classification model	83.33%	EEG Signals	No
Q. Xu et al.(2015) [68]	*K*-Means clustering algorithm	To provide personalized recommended products for stress management	85.2%	EEG, ECG, EMG, GSR signals	No
S. H. Song et al. (2017) [69]	Deep belief network	To design stress monitoring system	66%	KNHANES VI	No
Deep-Fog	Deep neural network	To obtain the stress level of patients to assist doctors	91.63%	EDA, HR, SpO2, Temperature	Yes

Table 9. Comparative analysis of diabetes prediction.

Method	Accuracy (%)	Use Cloud for Storage	Used in Fog Environment	Author
Hybrid model	84.5%	No	No	HumarKehramanili (2008) [70]
Multilayer perceptron	81.90%	No	No	Aliza Ahmed et al. (2011) [71,72]
Decision tree	89.30%	No	No	Aliza Ahmed et al. (2011) [71]
Extreme learning machine	75.72%	No	No	R. Priyadarshini et al. (2014) [73]
Decision tree	84%	No	No	K.M.Orabi et al. (20216) [74]
Improved *K*-means andlogistic regression	95.42%	No	No	Han Wu et al. (2017) [60]
Proposed approach	84.11%	Yes	Yes	

Table 10. Comparative analysis of Hypertension risk detection.

Author	Application Domain	Purpose	Use of CS	Use in FE	Use of PM
Fernandez et al. (2017) [75]	A web-based hypertension monitoring system	To identify weakness in the clinical process of hypertension detection, to improvise the existing system	No	No	No
Zhou et al. (2018) [76]	Cloud and mobile internet-based hypertension management system	Provide services to the patient to let them know their cardiac status	Yes	No	No
S.Sood and Isha Mahajan (2018) [77]	IoT-fog based system	Real time monitoring and decision making	Yes	Yes	Back propagation neural network with precision 92.10%
Proposed Work-*Deep Fog*	Deep learning-based Fog system	Real time Assessment of hypertension in patients and sending the data to cloud	Yes	Yes	Deep neural network with precision 92.01%

This experiment was carried out on a 64-bit machine, using a windows 7 system, with an Intel quad core processor. The java version used was 1.8. DL4j had a user interface provision for providing the used software information and hardware information, presenting the attributes-like installed operating system, JVM version, JDK version, heap memory available, required memory for JVM, the current consumed memory, are provided in Tables 11 and 12 respectively. It can be noted that the experiment was carried out in a CPU mode, so if the ND4j backend showed a CPU, if it would execute in a parallel environment, it would have been showing the backend as GPU. The hardware information the model generated detailed information of the available heap memory and consumed heap memory. Similarly the allocated memory for JVM and the current consumption of JVM memory. As the used system was a quad core system, the available number of processors for JVM was four.

Table 11. Software information.

Host Name	OS	OS Architecture	JVM Name	JVM Version	ND4J Backend	ND4J Type
3LAB24-PC	Windows 7	Amd64	Java Hotspot (TM)64-bit Server VM	1.8.0_141	CPU	Float

Table 12. Hardware information.

JVM Current Memory	JVM Max Memory	Off-Heap Current Memory	Off-Heap Max Memory	JVM Available Processors	#Compute Devices
315.50 MB	871.50 MB	451.22 MB	871.50 MB	4	0

Figure 17 presents the loss function of the deep neural network. This is the function showing the resulting error value between the actual output and the calculated value. It can be seen that the error is gradually falling down and approaches towards zero. Figure 18 shows the ratio of the parameter updates during the execution of the subsequent layers. The network weight changes are shown against the number of iterations. Figure 19 shows the standard deviation of the activations, gradients and updates against time. It can be observed that these are approaching zero. Figure 20 gives the memory utilization report of the Java virtual machine (JVM) and the heap memory consumption was 64% at a particular instance. It means that out of the allocated 1 GB of off-heap memory, 64% was utilized for the current work. DL4j provided this interface, which lets the user know that the memory consumption of JVM and the off-heap memory. The off-heap memory is the portion of the main memory, which is not managed by JVM, thereby it reduces the complexity of garbage collection of JVM. The default allocation of the off-heap memory is one fourth of the available free main memory of the system. The memory limits of the off-heap memory can be controlled but for this experiment, the default values were taken. As a system with 4 GB RAM was used for the experiment, one fourth of this memory i.e., 1 GB was allocated to the off-heap memory.

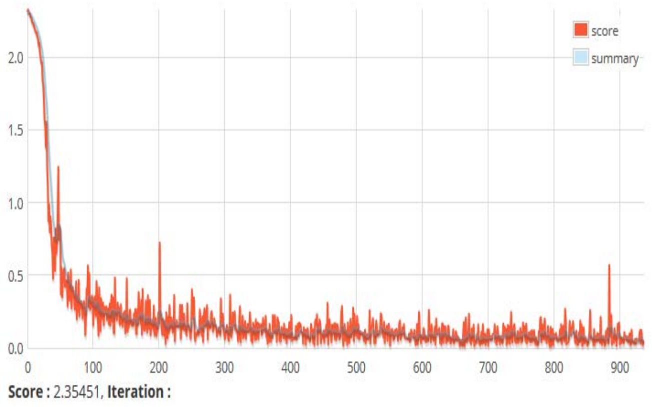

Figure 17. A snapshot of the score versus iteration chart. This shows the value of the loss function of a currently running set of instances. The loss function depicts the mean square error between the actual and target output.

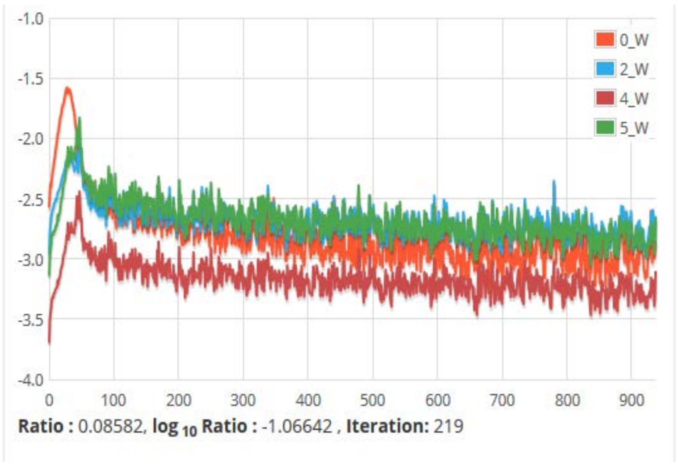

Figure 18. The ratio of parameters to updates (by layer) for all network weights vs. Iteration. It shows the update of the parameters against the number of iterations taken during training.

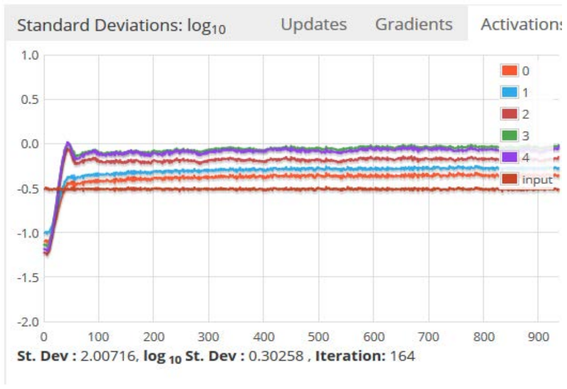

Figure 19. Standard deviations of the activations, gradients and updates versus time. Updates refer to the update of the training parameters such as momentum and learning rate parameter.

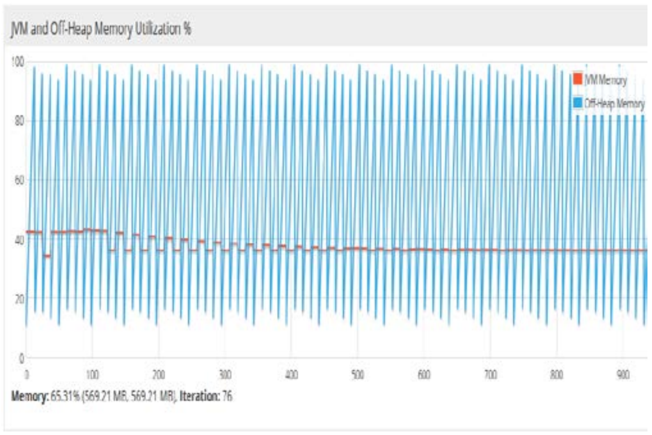

Figure 20. Percentage of the main memory utilization for JVM and heap memory. In DL4j, off-heap memory is the portion of the main memory, which is not used by JVM. Therefore, the work of the garbage collector by JVM is limited. The off-heap memory management was carried out using the underlying native operating system.

5. Conclusions

The whole world has scientifically and technologically come to a significant level. At the same pace, the lifestyle of people is changing. The amount of physical work and exercise has gone down and, at the same time, the stress level has gone up. This is leading to an alarming situation as it stimulates the growth rate of lifestyle disease like diabetes and hypertension. It now becomes a major concern of all including doctors, caregivers and the elderly population. With the significant developments in computation and advanced communication technology, effective solutions can be designed to solve this problem. This paper tries to propose a feasible fog computing-based deep learning model i.e., *DeepFog* to detect a person's mental state and does an early detection for type-2 diabetes from some of the captured data. It builds a model to assess the risk of hypertension attacks. The proposed solution further separates the high probable victim group of a hypertension attack, giving a priority to critical cases. The deep neural network was implemented using Python for constructing the hypertension dataset. The Java-based tool DL4j was leveraged to implement the proposed model. The fog environment was setup using a set of open source infrastructure. The results and discussions validate the efficacy of the proposed approach in wellness monitoring. We compared our results with other contemporary state-of-the-art research approaches where we found the proposed approach to be superior or competitive with current state-of-the-art technologies. In future work, we would like to further tune the neural models for various user-based studies. This would further create opportunities for research and development for robust systems development in the wellness monitoring area.

Author Contributions: Conceptualization, H.D.; Data curation, R.K.B.; Formal analysis, H.D.; Methodology, R.P.

Funding: This research received no external funding.

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

References

1. Kaveeshwar, S.A.; Cornwall, J. The current state of diabetes mellitus in India. *Aust. Med. J.* **2014**, *7*, 45. [CrossRef]
2. Barik, R.K.; Dubey, H.; Samaddar, A.B.; Gupta, R.D.; Ray, P.K. FogGIS: Fog Computing for geospatial big data analytics. In Proceedings of the IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics Engineering (UPCON), Varanasi, India, 9–11 December 2016; pp. 613–618.
3. Dubey, H.; Goldberg, J.C.; Abtahi, M.; Mahler, L.; Mankodiya, K. EchoWear: Smartwatch technology for voice and speech treatments of patients with Parkinson's disease. In Proceedings of the ACM Conference on Wireless Health, Bethesda, MD, USA, 14–16 October 2015; p. 15.
4. Dubey, H.; Monteiro, A.; Constant, N.; Abtahi, M.; Borthakur, D.; Mahler, L.; Sun, Y.; Yang, Q.; Akbar, U.; Mankodiya, K. Fog computing in medical internet-of-things: Architecture, implementation, and applications. In *Handbook of Large-Scale Distributed Computing in Smart Healthcare*; Springer: Cham, Switzerland, 2017; pp. 281–321.
5. Mathers, C.D.; Loncar, D. Projections of global mortality and burden of disease from 2002 to 2030. *PLoS Med.* **2006**, *3*, e442. [CrossRef] [PubMed]
6. WHO Fact Sheet Updated July 2017. Available online: https://www.who.int/immunization/newsroom/factsheets (accessed on 9 September 2017).
7. Perkins, G.D.; Jacobs, I.G.; Nadkarni, V.M.; Berg, R.A.; Bhanji, F.; Biarent, D.; Bossaert, L.L.; Brett, S.J.; Chamberlain, D.; de Caen, A.R.; et al. Cardiac arrest and cardiopulmonary resuscitation outcome reports: Update of the utstein resuscitation registry templates for out-of-hospital cardiac arrest: A statement for healthcare professionals from a task force of the International Liaison Committee on Resuscitation (American Heart Association, European Resuscitation Council, Australian and New Zealand Council on Resuscitation, Heart and Stroke Foundation of Canada, InterAmerican Heart Foundation, Resuscitation Council of Southern Africa. *Circulation* **2015**, *132*, 1286–1300. [PubMed]
8. Rosenthal, A.; Mork, P.; Li, M.H.; Stanford, J.; Koester, D.; Reynolds, P. Cloud computing: A new business paradigm for biomedical information sharing. *J. Biomed. Inform.* **2010**, *43*, 342–353. [CrossRef] [PubMed]
9. Shachtman, N. Feds Look to Fight Leaks with 'Fog of Disinformation'. *Wired Magazine*, 20 July 2012.
10. Cisco, I. *Cisco Visual Networking Index: Forecast and Methodology, 2011–2016*; CISCO White Paper; CISCO: San Jose, CA, USA, 2012; Volume 518.

11. Constant, N.; Borthakur, D.; Abtahi, M.; Dubey, H.; Mankodiya, K. Fog-assisted wiot: A smart fog gateway for end-to-end analytics in wearable internet of things. *arXiv*, 2017; arXiv preprint arXiv.
12. Dubey, H.; Mehl, M.R.; Mankodiya, K. Bigear: Inferring the ambient and emotional correlates from smartphone-based acoustic big data. In Proceedings of the 2016 IEEE First International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), Washington, DC, USA, 27–29 June 2016; pp. 78–83.
13. Borthakur, D.; Dubey, H.; Constant, N.; Mahler, L.; Mankodiya, K. Smart fog: Fog computing framework for unsupervised clustering analytics in wearable internet of things. In Proceedings of the 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Montreal, QC, Canada, 14–16 November 2017; pp. 472–476.
14. Barik, R.; Dubey, H.; Sasane, S.; Misra, C.; Constant, N.; Mankodiya, K. Fog2fog: Augmenting scalability in fog computing for health GIS systems. In Proceedings of the Second IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies, Philadelphia, PA, USA, 17–19 July 2017; pp. 241–242.
15. Barik, R.K.; Dubey, H.; Misra, C.; Borthakur, D.; Constant, N.; Sasane, S.A.; Lenka, R.K.; Mishra, B.S.; Das, H.; Mankodiya, K. Fog Assisted Cloud Computing in Era of Big Data and Internet-of-Things: Systems, Architectures, and Applications. In *Cloud Computing for Optimization: Foundations, Applications, and Challenges*; Springer: Cham, Switzerland, 2018; pp. 367–394.
16. Barik, R.K.; Priyadarshini, R.; Dubey, H.; Kumar, V.; Mankodiya, K. FogLearn: Leveraging fog-based machine learning for smart system big data analytics. *Int. J. Fog Comput. (IJFC)* **2018**, *1*, 15–34. [CrossRef]
17. Priyadarshini, R.; Barik, R.K.; Panigrahi, C.; Dubey, H.; Mishra, B.K. An Investigation Into the Efficacy of Deep Learning Tools for Big Data Analysis in Health Care. *Int. J. Grid High-Perform. Comput. (IJGHPC)* **2018**, *10*, 1–13. [CrossRef]
18. Barik, R.K.; Priyadarshini, R.; Dubey, H.; Kumar, V.; Yadav, S. Leveraging Machine Learning in Mist Computing Telemonitoring System for Diabetes Prediction. In *Advances in Data and Information Sciences*; Springer: Singapore, 2018; pp. 95–104.
19. Borthakur, D.; Peltier, A.; Dubey, H.; Gyllinsky, J.; Mankodiya, K. SmartEAR: Smartwatch-based Unsupervised Learning for Multi-modal Signal Analysis in Opportunistic Sensing Framework. In Proceedings of the IEEE/ACM 3rd International Conference on Connected Health: Applications, Systems and Engineering Technologies, Washington, DC, USA, 26–28 September 2018.
20. Computing, F. *The Internet of Things: Extend the Cloud to Where the Things Are*; Cisco White Paper; CISCO: San Jose, CA, USA, 2015.
21. Dastjerdi, A.V.; Gupta, H.; Calheiros, R.N.; Ghosh, S.K.; Buyya, R. Fog computing: Principles, architectures, and applications. In *Internet of Things*; Morgan Kaufmann: Burlington, MA, USA, 2016; pp. 61–75.
22. Rahmani, A.M.; Gia, T.N.; Negash, B.; Anzanpour, A.; Azimi, I.; Jiang, M.; Liljeberg, P. Exploiting smart e-Health gateways at the edge of healthcare Internet-of-Things: A fog computing approach. *Future Gener. Comput. Syst.* **2018**, *78*, 641–658. [CrossRef]
23. Gia, T.N.; Jiang, M.; Rahmani, A.M.; Westerlund, T.; Liljeberg, P.; Tenhunen, H. Fog computing in healthcare internet of things: A case study on ecg feature extraction. In Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing (CIT/IUCC/DASC/PICOM), Liverpool, UK, 26–28 October 2015; pp. 356–363. [CrossRef]
24. Dubey, H.; Yang, J.; Constant, N.; Amiri, A.M.; Yang, Q.; Makodiya, K. Fog data: Enhancing telehealth big data through fog computing. In Proceedings of the ASE BigData & Social Informatics 2015, Kaohsiung, Taiwan, 9 October 2015; p. 14. [CrossRef]
25. Monteiro, A.; Dubey, H.; Mahler, L.; Yang, Q.; Mankodiya, K. FIT A Fog Computing Device for Speech TeleTreatments. *arXiv*, arXiv:1605.06236.
26. Birjandtalab, J.; Cogan, D.; Pouyan, M.B.; Nourani, M. A non-EEG biosignals dataset for assessment and visualization of neurological status. In Proceedings of the 2016 IEEE International Workshop onSignal Processing Systems (SiPS), Dallas, TX, USA, 26–28 October 2016; pp. 110–114. [CrossRef]
27. Castilla-Guerra, L.; del Carmen Fernandez-Moreno, M. Chronic management of hypertension after stroke: The role of ambulatory blood pressure monitoring. *J. Stroke* **2016**, *18*, 31. [CrossRef]

28. Das, S.; Ghosh, P.K.; Kar, S. Hypertension diagnosis: A comparative study using fuzzy expert system and neuro fuzzy system. In Proceedings of the 2013 IEEE International Conference onFuzzy Systems (FUZZ), Hyderabad, India, 7–10 July 2013; pp. 1–7. [CrossRef]
29. Sandi, G.; Nugraha, I.G.B.B.; Supangkat, S.H. Mobile health monitoring and consultation to support hypertension treatment. In Proceedings of the 2013 International Conference onICT for Smart Society (ICISS), Jakarta, Indonesia, 13–14 June 2013; pp. 1–5. [CrossRef]
30. Priyadarshini, R.; Barik, R.K.; Dash, N.; Mishra, B.K.; Misra, R. A hybrid GSA-K-mean classifier algorithm to predict diabetes mellitus. *Int. J. Appl. Metaheuristic Comput. (IJAMC)* **2017**, *8*, 99–112. [CrossRef]
31. Barik, R.K.; Priyadarshini, R.; Dash, N. A Meta-Heuristic Model for Data Classification Using Target Optimization. *Int. J. Appl. Metaheuristic Comput. (IJAMC)* **2017**, *8*, 24–36. [CrossRef]
32. Garcia Lopez, P.; Montresor, A.; Epema, D.; Datta, A.; Higashino, T.; Iamnitchi, A.; Barcellos, M.; Felber, P.; Riviere, E. Edge-centric computing: Vision and challenges. *ACM SIGCOMM Comput. Commun. Rev.* **2015**, *45*, 37–42. [CrossRef]
33. Barik, R.K.; Dubey, H.; Mankodiya, K.; Sasane, S.A.; Misra, C. GeoFog4Health: A fog-based SDI framework for geospatial health big data analysis. *J. Ambient. Intell. Hum. Comput.* **2018**, 1–17. [CrossRef]
34. Barik, R.K.; Dubey, H.; Mankodiya, K. Soa-fog: Secure service-oriented edge computing architecture for smart health big data analytics. In Proceedings of the 2017 IEEE Global Conference on Signal and Information Processing (GlobalSIP), Montreal, QC, Canada, 14–16 November 2017; pp. 477–481.
35. Zao, J.K.; Gan, T.T.; You, C.K.; Méndez, S.J.R.; Chung, C.E.; Te Wang, Y.; Mullen, T.; Jung, T.P. Augmented brain computer interaction based on fog computing and linked data. In Proceedings of the 2014 International Conference onIntelligent Environments (IE), Shanghai, China, 30 June–4 July 2014; pp. 374–377. [CrossRef]
36. Campolo, C.; Molinaro, A.; Scopigno, R.; Ozturk, S.; Mišić, J.; Mišić, V.B. The MAC Layer of VANETs. In *Vehicular ad hoc Networks*; Springer: Cham, Switzerland, 2015; pp. 83–122.
37. Santos, J.; Wauters, T.; Volckaert, B.; De Turck, F. Fog Computing: Enabling the Management and Orchestration of Smart City Applications in 5G Networks. *Entropy* **2018**, *20*, 4. [CrossRef]
38. Perera, C.; Zaslavsky, A.; Christen, P.; Georgakopoulos, D. Sensing as a service model for smart cities supported by internet of things. *Trans. Emerg. Telecommun. Technol.* **2014**, *25*, 81–93. [CrossRef]
39. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the First Edition of the MCC Workshop on Mobile Cloud Computing, Helsinki, Finland, 13–17 August 2012; pp. 13–16. [CrossRef]
40. Bonomi, F.; Milito, R.; Natarajan, P.; Zhu, J. Fog computing: A platform for internet of things and analytics. In *Big Data and Internet of Things: A Roadmap for Smart Environments*; Springer: Cham, Switzerland, 2014; pp. 169–186.
41. Tang, B.; Chen, Z.; Hefferman, G.; Wei, T.; He, H.; Yang, Q. A hierarchical distributed fog computing architecture for big data analysis in smart cities. In Proceedings of the ASE BigData&SocialInformatics 2015, Kaohsiung, Taiwan, 7–9 October 2015; p. 28. [CrossRef]
42. Sarkar, S.; Misra, S. Theoretical modelling of fog computing: A green computing paradigm to support IoT applications. *IetNetw.* **2016**, *5*, 23–29. [CrossRef]
43. Suárez-Albela, M.; Fernández-Caramés, T.M.; Fraga-Lamas, P.; Castedo, L. A Practical Evaluation of a High-Security Energy-Efficient Gateway for IoT Fog Computing Applications. *Sensors* **2017**, *17*, 1978. [CrossRef] [PubMed]
44. Mudgal, S. *Deep Learning for Natural Language Processing*. Available online: http://pages.cs.wisc.edu/~{}shavlik/cs638/lectureNotes/dl_nlp_talk.pdf (accessed on 25 May 2017).
45. Deng, L.; Liu, Y. (Eds.) *Deep Learning in Natural Language Processing*; Springer: Berlin, Germany, 2018.
46. Collobert, R.; Weston, J. A unified architecture for natural language processing: Deep neural networks with multitask learning. In Proceedings of the 25th International Conference on Machine Learning, Helsinki, Finland, 5–9 July 2008; pp. 160–167. [CrossRef]
47. Abdel-Zaher, A.M.; Eldeib, A.M. Breast cancer classification using deep belief networks. *Expert Syst. Appl.* **2016**, *46*, 139–144. [CrossRef]
48. Chi, J.; Kim, H.-C. Prediction of Arctic Sea Ice Concentration Using a Fully Data Driven Deep Neural Network. *Remote Sens.* **2017**, *9*, 1305. [CrossRef]
49. Wang, S.; Weng, S.; Ma, J.; Tang, Q. DeepCNF-D: Predicting Protein Order/Disorder Regions by Weighted Deep Convolutional Neural Fields. *Int. J. Mol. Sci.* **2015**, *16*, 17315–17330. [CrossRef] [PubMed]

50. Al Rahhal, M.M.; Bazi, Y.; AlHichri, H.; Alajlan, N.; Melgani, F.; Yager, R.R. Deep learning approach for active classification of electrocardiogram signals. *Inf. Sci.* **2016**, *345*, 340–354. [CrossRef]
51. CireşAn, D.; Meier, U.; Masci, J.; Schmidhuber, J. Multi-column deep neural network for traffic sign classification. *Neural Netw.* **2012**, *32*, 333–338. [CrossRef] [PubMed]
52. Almisreb, A.A.; Jamil, N.; Din, N.M. Utilizing AlexNet Deep Transfer Learning for Ear Recognition. In Proceedings of the 2018 IEEE Fourth International Conference on Information Retrieval and Knowledge Management (CAMP), Kota Kinabalu, Malaysia, 26–28 March 2018; pp. 1–5. [CrossRef]
53. Qawaqneh, Z.; Mallouh, A.A.; Barkana, B.D. Age and gender classification from speech and face images by jointly fine-tuned deep neural networks. *Expert Syst. Appl.* **2017**, *85*, 76–86. [CrossRef]
54. Gao, X.W.; Hui, R.; Tian, Z. Classification of CT brain images based on deep learning networks. *Comput. Methods Programs Biomed.* **2017**, *138*, 49–56. [CrossRef] [PubMed]
55. 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]
56. Ahmad, M.; Amin, M.B.; Hussain, S.; Kang, B.H.; Cheong, T.; Lee, S. Health Fog: A novel framework for health and wellness applications. *J. Supercomput.* **2016**, *72*, 3677–3695. [CrossRef]
57. Yi, S.; Li, C.; Li, Q. A survey of fog computing: Concepts, applications and issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Hangzhou, China, 21 June 2015; pp. 37–42. [CrossRef]
58. Cogan, D.; Birjandtalab, J.; Nourani, M.; Harvey, J.; Nagaraddi, V. Multi-biosignal analysis for epileptic seizure monitoring. *Int. J. Neural Syst.* **2017**, *27*, 1650031. [CrossRef] [PubMed]
59. Liu, Z.P.; Gao, R. Detecting pathway biomarkers of diabetic progression with differential entropy. *J. Biomed. Inform.* **2018**, *82*, 143–153. [CrossRef]
60. Wu, H.; Yang, S.; Huang, Z.; He, J.; Wang, X. Type 2 diabetes mellitus prediction model based on data mining. *Inform. Med. Unlocked* **2018**, *10*, 100–107. [CrossRef]
61. Kaur, A.; Bhardwaj, A. Artificial Intelligence in hypertension diagnosis: A review. *Int. J. Comput. Sci. Inf. Technol.* **2014**, *5*, 2633–2635.
62. Huang, S.; Xu, Y.; Yue, L.; Wei, S.; Liu, L.; Gan, X.; Zhou, S.; Nie, S. Evaluating the risk of hypertension using an artificial neural network method in rural residents over the age of 35 years in a Chinese area. *Hypertension Res.* **2010**, *33*, 722–726. [CrossRef] [PubMed]
63. LaFreniere, D.; Zulkernine, F.; Barber, D.; Martin, K. Using machine learning to predict hypertension from a clinical dataset. In Proceedings of the 2016 IEEE Symposium Series on Computational Intelligence (SSCI), Athens, Greece, 6–9 December 2016; pp. 1–7. [CrossRef]
64. Demšar, J.; Zupan, B. Orange: Data mining fruitful and fun-a historical perspective. *Informatica* **2013**, *37*, 55–60.
65. Fox, J.; Zou, Y.; Qiu, J. *Software Frameworks for Deep Learning at Scale*. Internal Indiana University Technical Report. Available online: https://pdfs.semanticscholar.org/5d72/065c17cf5d0a7f916ffdb18cbf695fd846e8.pdf (accessed on 23 October 2018).
66. Sugomori, Y. *Java Deep Learning Essentials*; Packt Publishing Ltd.: Birmingham, UK, 2016.
67. Sani, M.M.; Norhazman, H.; Omar, H.A.; Zaini, N.; Ghani, S.A. Support vector machine for classification of stress subjects using EEG signals. In Proceedings of the 2014 IEEE Conference on Systems, Process and Control (ICSPC), Kuala Lumpur, Malaysia, 12–14 December 2014; pp. 127–131. [CrossRef]
68. Xu, Q.; Nwe, T.L.; Guan, C. Cluster-based analysis for personalized stress evaluation using physiological signals. *IEEE J. Biomed. Health Inform.* **2015**, *19*, 275–281. [CrossRef] [PubMed]
69. Song, S.H.; Kim, D.K. Development of a Stress Classification Model Using Deep Belief Networks for Stress Monitoring. *Heal. Inform. Res.* **2017**, *23*, 285–292. [CrossRef]
70. Kahramanli, H.; Allahverdi, N. Design of a hybrid system for the diabetes and heart diseases. *Expert Syst. Appl.* **2008**, *35*, 82–89. [CrossRef]
71. Ahmad, A.; Mustapha, A.; Zahadi, E.D.; Masah, N.; Yahaya, N.Y. Comparison between Neural Networks against Decision Tree in Improving Prediction Accuracy for Diabetes Mellitus. In *Digital Information Processing and Communications*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 537–545.
72. Michie, D.J.; Spiegelhalter, C.C. *Taylor Machine Learning, Neural and Statistical Classification*; Ellis Horward Series in Artifical Intelligence: New York, NY, USA, 1994.

73. Priyadarshini, R.; Dash, N.; Mishra, R. A Novel approach to predict diabetes mellitus using modified Extreme learning machine. In Proceedings of the 2014 International Conference on Electronics and Communication Systems (ICECS), Coimbatore, India, 13–14 February 2014; pp. 1–5. [CrossRef]
74. Orabi, K.M.; Kamal, Y.M.; Rabah, T.M. Early Predictive System for Diabetes Mellitus Disease. In *Industrial Conference on Data Mining*; Springer: Cham, Switzerland, July 2016; pp. 420–427. [CrossRef]
75. Ruiz-Fernández, D.; Marcos-Jorquera, D.; Gilart-Iglesias, V.; Vives-Boix, V.; Ramírez-Navarro, J. Empowerment of patients with hypertension through BPM, iot and remote sensing. *Sensors* **2017**, *17*, 2273. [CrossRef] [PubMed]
76. Zhou, R.; Cao, Y.; Zhao, R.; Zhou, Q.; Shen, J.; Zhou, Q.; Zhang, H. A novel cloud based auxiliary medical system for hypertension management. *Appl. Comput. Inform.* **2017**. [CrossRef]
77. Sood, S.K.; Mahajan, I. IoT-Fog based Healthcare Framework to Identify and Control Hypertension Attack. *IEEE Internet Things J.* **2018**. [CrossRef]