Indonesian Journal of Science & Technology

Journal homepage: http://ejournal.upi.edu/index.php/ijost/

A Neural Network Aided Real-Time Hospital Recommendation System

Om Adideva Paranjay, Dr Rajeshkumar V*

School of Electronics Engineering, Vellore Institute of Technology (VIT) Vellore, India

Correspondence: E-mail: omadidevparanjay@gmail.com

ABSTRACT

Over the past few years, there have been an overwhelming number of healthcare providers, hospitals and clinics. In such a situation, finding the right hospital for the right ailment can be a considerable challenge. Inspired by this challenge, this work attempts to build a model that can automatically recommend hospitals based on user requirements. In the past there have been important works in physician recommendation. Our proposed work aims to be more inclusive and provide an automated hospital recommendation system to patients based on neural networks driven classification. We suggest a model that considers several unique parameters, including geographical location. To optimize its usefulness, we design a system that recommends hospitals for general consultation, specialty hospitals, and in view of the pandemic, hospitals recommended for treatment of COVID-19. In this work, we adopt Neural Networks and undertake a comparative analysis between several different available supervised algorithms to identify one best suited neural architecture that can work best in the applied fields. Based on our results from the analysis, we train the selected neural network with context relevant data. In the image of the recommendation system, we develop a website that uses the trained neural network on its backend and displays the recommendation results in a manner interpretable by the end user. We highlight the process of choosing the right neural model for the backend of the website. To facilitate the working of the website in real-time, we use real time databases hosted on Google Firebase and edge devices on hospital ends. Additionally, we suggest two hospital side data updation tools. These tools would ensure that hospitals can update the parameters which change quickly in the real world to their latest values so as to maintain the precision of the system. We test the website with test data and find that the website recommends hospitals with sufficient precision in the specified format. The model has been designed with the limited amount of data available in this field, but its performance and utility can be easily improved with better quality and more abundant data.

ARTICLE INFO

Article History:
Submitted/Received 22 Apr 2020
First revised 22 May 2020
Accepted 23 Jul 2020
First available online 25 Jul 2020
Publication date 01 Sep 2020

Keywords:
Neural Networks,
Real-Time Database,
Recommendation System,
Raspberry Pi,
Google Firebase,
Deep Belief Networks,
Dense Neural Networks,
Website,
Python.

1. INTRODUCTION

Recommendation systems are widely used these days to understand user needs and then suggest products or services based on their preferences. Popular services like YouTube and Spotify make use of such systems to suggest videos and songs to their users based on the users' preferences. The recommendation system has also found its way into the healthcare sector in very important and significant ways, especially in the field of physician recommendation. Existing works in this field use cost of treatment and maximum improvement in the health of previous patients as parameters to recommend physicians (Wen *et al.*, 2020) or use clustering techniques, self-organizing maps and correlation (Tabrizi *et al.*, 2016) to find similarity between different patient groups and then recommend physicians to them. But these existing methods have certain limitations. For example, in the latter model although mathematical tools have been used in the said system, the work still depends on patient satisfaction grouping. Certainly, it is a novel methodology, but it does not provide an absolute basis to recommend physicians. More importantly, since these existing works are on physician recommendation, they do not consider geographical location as a parameter.

To bridge these existing gaps and address the allied issues, we are proposing a novel hospital recommendation system in this work. Here we choose unique parameters which are numerical, objective and are not subject to bias to carry our comparisons before recommendations. For example, hospital overall rankings and department specific rankings are used as the qualifying parameters. Rankings by reliable organizations provide a holistic view of the hospital very concisely. Using such common parameters helps in better comparison of those. Among other parameters, included are 'Doctors on Duty', 'Beds Available' and 'Distance'. These are parameters which can change quickly and have an influence over the quality of treatment a person may be provided. Thus, in our proposed work we use a combination of quickly changing and stationary parameters to be able to carry out comparisons from a macro perspective. Secondly, the changing situations due to the pandemic also motivated us to offer some solutions to problems that would arise in the near future. Now, most countries have demarcated hospitals for the treatment of COVID-19. However, these hospitals are not sustainable once the number of cases start to decline. As shown in the case of China, sooner or later, such hospitals must be shut down (Verma, 2020). The other possibility as most experts say the discovery of a vaccine is still far off (Gallagher, 2020). This creates a quizzical time where there are no specially demarcated hospitals to treat COVID-19 specifically, and there is no vaccine to treat it. This means, people will still get affected by the disease but will have to rely on general hospitals for treatment. In this work, we aim to model a system to recommend suitable hospitals in this time span so as to help people identify the right hospitals for Coronavirus treatment. Thus, among other specialty hospital recommendations, we have also proposed to embed a feature to enable the model to recommend hospitals for COVID 19 treatment. We choose to use Neural Networks to carry out the comparisons and provide recommendations.

In the real-world, Deep Learning has helped in several domains including real estate and finance. Especially in the field of classification, neural networks have become very popular (Zhang, 2000). Several supervised learning algorithms are available and choosing the best model for this application is also a major part of this work. The model best suited for this purpose is determined through a reliability, reproducibility and 'model accuracy' comparison. The use of real-time databases has been proposed which provide updated information about

the hospital's resources i.e. 'Beds Available' and 'Doctors on Duty' to the minute. This makes classification more accurate and makes the hospitals equally important stakeholders in this. The final product of this work is a website which uses the real-time databases to gather information and uses a trained neural model in the background for the classification purpose. It can be used by patients themselves, ambulance operators and potentially any user with a location service and internet enabled device.

This model would provide a shopping-cart experience of hospital selection where certain requirements are to be filled in. The website does the rest of the analysis and comparison and provides the most suitable results.

This article is organized into four sections. The next section deals with the data acquisition and methodology of the entire research. Section III deals with our findings from our research and the performance of our proposed website. The final section deals with our inferences, the conclusions we drew and our recommendations for improving the quality of this model and future extensions of this study.

2. MATERIALS AND METHODS

The work has been divided into three major sections: Hardware, Software and Neural Networks. The Hardware section consists of the Raspberry Pi/ Fingerprint Scanner hardware unit used at the hospital end to record the entry and exit of doctors. The concerned Raspberry Pi is connected to the real time databases hosted on Google Firebase and updates the values in real time. The Software part consists of the website, the real time database and the Hospital Bed Updation Tool. Finally, the Neural Networks section deals with the selection, development and analysis of Convolution Neural Network (CNN), Multi-Layer Perceptron Network (MLP), Dense Neural Network (DNN) and Deep Belief Network (DBN) for this application.

2.1 Data Acquisition

Relevant data is necessary for the training of the neural networks to be used in the process. We used data from two datasets; first, a dataset from Centers for Medicare and Medicaid Services (https://www.kaggle.com/cms/hospital-general-information), and second, a COVID -19. specific dataset from Global Pandemics Organization (https://globalepidemics.org/our-data/hospital-capacity/). The first one contains general information about 5000 hospitals in the USA and the second contains the hospital capacities in the different COVID-19 hotspots of the USA. Since the COVID-19 specific data is not sufficient for our work now, we created an artificial dataset by combining data from both the above-mentioned datasets.

2.2 Data Preprocessing

The first dataset acquired from CMS had several columns of information which was not required for this work, and had certain information missing (see **Figure 1**). To suit our research, certaincolumns of information were added to the dataset. Specifically, the columns titled 'Distance', 'Beds Available', 'Doctors on Duty' were added as random integers as they would appear in real world use. The real-time database that was hosted on Firebase was also formatted similarly so as to provide the very same parameters in the same arrangement as input. Thus, for compatibility with the actual data that would be acquired in real-time and to be able to consider all the parameters that this research aims to, dummy numbers were put into additional columns mimicking the real-world values.

	A	B	C	D	E	F	G	H	I
1	Facility Name	Hospital overall rating	Location	Beds Available	Doctors on Duty	Cardio Rating	Neuro Rating	Ortho Rating	Distance
2	NORTH TEXAS STA	3	(33.840213, -98.5	19	19	1	2	5	28
3	MAGEE GENERAL H	3	(31.870585, -89.7	5	10	3	3	2	2
4	CYPRESS CREEK HC	3	(30.024186, -95.4	7	7	2	4	1	12
5	MARSHALL COUNT	3	(45.789215, -97.	7	15	1	2	4	20
6	JOHN SEED HOSPIT	3	(26.51088, -81.9	16	10	2	3	2	12
7	ST. HOPE HOSPITA	3	(41.034395, -73.4	17	19	2	1	5	11
8	PARK ROYAL HOSP	2	(30.04461, -95.22	13	15	1	1	3	5
9	GREENWICH HOSP	3	(43.268092, -91.4	6	12	3	4	1	11
10	KINGWOOD PINES	3	(40.311262, -111.	6	19	1	4	3	5
11	VETERANS MEMOI	1	(42.840846, -106	3	6	1	1	1	19
12	PROVO CANYON B	3	(34.832836, -114	9	18	4	3	2	29
13	MANHATTAN PSYC	4	(37.205988, -93.2	20	20	5	1	5	10
14	SUMMIT MEDICAL	2	(32.371613, -99.	4	9	1	2	2	1
15	MCCURTAIN MEM	5	(43.828588, -96.7	4	13	5	5	5	22
16	SELLS HOSPITAL	5	(32.844744, -87.3	23	18	5	5	5	6
17	OCEANS BEHAVIOI	4	(32.630784, -97.	8	10	5	2	4	18
18	BRATTLEBORO RET	3	(34.479407, -82.5	29	6	1	4	3	26
19	GREENE COUNTY H	2	(41.890512, -87.	25	16	1	3	1	2
20	COMANCHE COUN	2	(29.692093, -95.4	30	6	1	2	1	18
21	SUNDANCE HOSPI	2	(42.504753, -96.4	27	19	2	1	3	27
22	PATRICK B HARRIS	4	(40.684765, -73.4	18	10	5	4	1	18
23	LBJ TROPICAL MEI	4	(32.472758, -93.	16	9	5	4	2	20
24	SUN BEHAVIORAL	3	(44.222213, -72.5	16	12	4	2	3	20
25	KANAKANAK HOSP	3	(41.4436, -86.14	3	12	4	2	2	22
26	DUNES SURGICAL I	3	(44.475809, -87.5	17	20	1	2	5	15
27	ST PETERS HOSPIT	4	(30.859492, -83.2	26	16	4	4	2	3
28	US PAIN & SPINE H	4	(44.955311, -90.5	8	9	5	4	1	15
29	PHYSICIANS BEHAV	3	(32.898527, -97.3	11	10	1	2	5	12

Figure 1. The obtained dataset was modified to remove certain existing fields and certain necessary fields were added in the same format as they would appear in the real-world applications. Finally, the above dataset was created.

A different dataset was created from combining the two acquired datasets for COVID 19 Hospitals. The dataset had to be developed because the requisite format of data for this kind of recommendation is not available yet and we had to use a set of different parameters. While 'Distance', 'Doctors on Duty' and 'Hospital Overall Rank' were repeated, 'Total Beds Available', "ICU Beds Available' and 'Experience in treating COVID-19' were the new parameters. Especially the latter one is important as prior experience in treating the disease does give certain hospitals an edge over others. This was denoted in binary; '1' for yes and '0' for no. Another real time database was hosted on Firebase within the same project specifically for this recommendation. The data within this database is in the same format as the training dataset. A separate database was created because the prevailing situation will not be the same for years to come. Thus, having a separate database will allow any changes to be made without having to edit the other database which would not require as many changes over time.

To convert all parameter values to a single score for each of the hospitals in the dataset, a procedure used in the United States Hospital Star Ranking Methodology (https://cpb-us-w2.wpmucdn.com/u.osu.edu/dist/c/28860/files/2016/08/Star_Rtngs_CompMthdlgy_052016-148w094.pdf) was employed. A Z-Score for each of the parametric values was determined using the following formulas:

$$(\bar{x}-x)/\sigma \quad (1)$$

$$(x-\bar{x})/\sigma \quad (2)$$

In the above equations x stands for the value, $\bar{x}$ stands for mean and σ stands for standard deviation. While the first equation is used for negative weightages, the second equation is used for positive weightage values.

Finally, the Z-Scores were converted into a single numerical value using a Weightage Formula. At this point three copies of the General dataset were made. The system is envisioned to work as a recommender for not only General Emergency Cases, but specific department cases also. For the time be-

ing we chose Cardiology, Neurology and Orthopedics as our specialized departments. These department specific recommendations use the same kind of data; thus, a copy of the original General dataset would suffice. To be able to train specialized classifiers for different requirements we chose to train the classifiers for each of the departments with different datasets. Therefore, we chose to make three different department specific datasets which only differed on the weightage formula. The different weightage formulae used in the General, Neuro, Orthopedic, Cardiac and COVID-19 models are given below in respective order:

$$0.39 \times R_{OV} + 0.2 \times B_A + 0.02 \times R_C + 0.02 \times R_O + 0.02 \times R_N + 0.05 \times D_D + 0.3 \times D \quad (3)$$

$$0.05 \times R_{OV} + 0.2 \times B_A + 0.02 \times R_C + 0.02 \times R_O + 0.4 \times R_N + 0.05 \times D_D + 0.3 \times D \quad (4)$$

$$0.05 \times R_{OV} + 0.2 \times B_A + 0.02 \times R_C + 0.4 \times R_O + 0.02 \times R_N + 0.05 \times D_D + 0.3 \times D \quad (5)$$

$$0.05 \times R_{OV} + 0.2 \times B_A + 0.4 \times R_C + 0.02 \times R_O + 0.02 \times R_N + 0.05 \times D_D + 0.3 \times D \quad (6)$$

$$0.35 \times P_E + 0.20 \times D + 0.1 \times B_{TA} + 0.15 \times B_{ICU} + 0.1 \times R_O + 0.1 \times D_D \quad (7)$$

In the above equations R_{OV} stands for Overall Hospital Rating, B_A stands for Beds Available, R_C stands for Cardiology Department Rating, R_O stands for Orthopedic Department Rating, R_N stands for Neurology Department Rating , D_D stands for Doctors on Duty ,D denotes the distance, P_E stands for Prior Experience in treating COVID 19, B_{TA} stands for Total Beds Available and B_{ICU} denotes ICU beds available.

The above weightage formulae produced a single score result. Within each dataset, 4996 scores were present, each corresponding to a single hospital. The 66th percentile of this data was calculated and used as the threshold value. An exception was made only for the COVID-19 dataset and the 50th percentile was calculated in this case. This was done because we wanted less rigor in the recommendation of such hospitals, because there are small number of hospitals that have dealt with COVID -19 to begin with.

The score of any hospital lying above the decided threshold value was labelled with a 1 for 'Yes'. Otherwise a 0 was given for 'No'. As of now, the work relies on hospital rankings that have already been done by ranking agencies. However, for greater uniformity, the ranking method devised in another study (Habibi *et al.*, 2019) can be used to treat the parameters and then label the dataset. We suggest, every hospital that is put into the database of this system be first ranked through a ranking methodology that hold true universally, such as the one stated before. After this all the intermediate scores were removed from the dataset. The final dataset to be fed into the neural networks only contained the name and parameter values of the hospital followed by a Yes/No label in binary notation. Thereafter, we moved to the development and analysis of our shortlisted neural models.

2.3 Neural Networks

The first neural network to be selected for analysis was a 2-Dimensional Convolutional Neural Network (CNN). As this was essentially a classification problem, using a CNN was logical. CNNs are inspired from the organization of neurons in the visual cortex of the brain. These networks work well with images as they can extract important features from images and assign proper weightage to them. CNNs use MaxPooling layers that fish out the important features from an image thereby making the image more and more concise. After a few layers of MaxPooling when the image has been reduced to a one-dimensional array of the most important features that define the image, it is passed through a Flatten layer

which makes the data ready to be put into a Dense Neural Layer. The Neurons in this layer undergo the learning process and are responsible for classifying the input images into predefined labels. In case of the dataset used in this work, the dimensionality of the dataset had to be increased in order to be able to input it into the network. The Neural architecture was built using the Keras library for TensorFlow which has several neural networks as predefined functions. Huber loss was used as the loss function and the Adam optimizer was used in this case (see **Figure 2**).

Next, a Perceptron Network (MLP) was chosen for analysis. MLPs are the simplest kind of neural networks. They are quite easy to build and train. Due to the Linear Separability property that Perceptrons are based on, they work well with classification. However, they excel only at linear classification and their accuracy is questionable in nonlinear classification situations. Therefore, they are also called Linear Binary Classifiers. In our dataset the binary labels have been reached through mathematical formulae based only upon addition and multiplication. So, the independent and target variables are neither too abstract nor related in a complicated manner. Therefore, we expected an MLP to perform well with this dataset. MLPs have an input layer, variable hidden layers and an output layer. The number of hidden layers in the architecture is variable. Finally, a Heaviside Activation Function classifies the inputs into each of the two labels. For our application, a multilayer Perceptron Network was used which used two hidden layers and was followed by the output layer. The output layer used the Heaviside Activation Function for classification. MLPs are not available as predefined functions in the Keras library, therefore, they had to be hardcoded on the basis of the algorithm.

Next, a Dense Neural Network (DNN) was used. These are also called Fully Connected Neural Networks (FCNN) as each neuron is connected to every other neuron in the previous layer. Because of this complex networking this architecture performs much better classification than MLPs. They can perform well even in non-linear classification cases. These are versatile networks and quite easy to build from the ground up. They are used at the end of CNNs for classification.

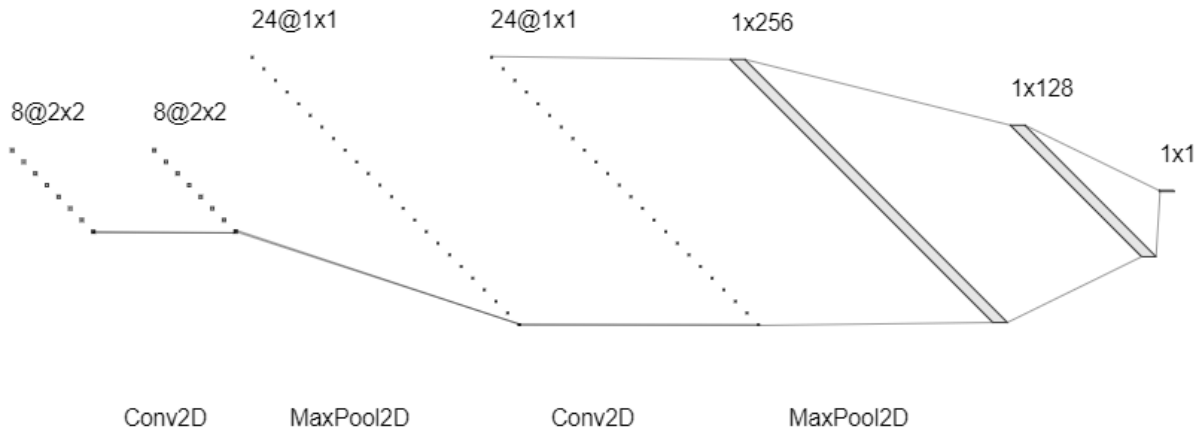

Figure 2. The architecture of the CNN that was developed for this application. Although the depth is shown as 8 and 24 it is only indictive. Every layer has a depth of 3005 indicating the number of datapoints in the training dataset.

Our dataset does not need feature extraction, therefore DNNs on their own could be trained on the training set and then be used for prediction and classification on the test set. DNNs are very powerful, even more than MLPs and CNNs simply because they are Fully Connected. Unlike CNNs which repetitively extract feature from a small field, DNNs can take into account every combination of every feature from the previous layer. The DNN used for this application had 5 layers. The input layer and the 3 hidden layers used 'Rectified Linear Unit' as the activation function. The final output layer used a 'Sigmoid' function as the activation function. Binary Crossentropy was used as the loss function for this network as it performs well with binary classification problems. As for the optimizer the Stochastic Gradient Descent or SGD optimizer was used (see **Figure 3**).

Lastly, a Deep Belief Network (DBN) was chosen for analysis. This is not available in the Keras Library as a predefined function However, this work used a GitHub repository (Albertbup, 2018) to build a DBN similar to a sequential Keras model. DBNs are generative networks and can be used in situations where the dataset is not very comprehensive. DBNs have hidden layers made of Restricted Boltzmann Machines and the final layer is a classifier. Thus, having a limited dataset and a classification problem at hand, the choice of DBN was logical.

Architecturewise, a DBN looks a lot like an MLP. A DBN, however, is a stack of Restricted Boltzmann machines (RBMs). The DBNs have a pre-training process wherein all the RBMs learn the features of the data globally unlike in CNNs. Thereafter, in the fine-tuning phase the labels are corresponded to each pattern stored in the RBMs. This makes the network more accurate. For our application, a DBN was designed with 2 hidden layers having 200 RBMs each and an output layer having one RBM which acts as the classifier. The architecture used the Rectified Linear Unit as the activation function for the RBMs. **Figure 4** is the architecture of the developed DBN with one input layer, two hidden layers and one output layer. It must be noted that in the figure below each circle is an RBM and not a neuron.

2.4 Website and Hospital Bed Updation Tool

In the final phase of the work, the website and the Hospital Bed Updation Tool were developed. The website was divided into two segments; the frontend and the backend. The backend was developed using Python as it had all the classifiers. As mentioned previously four different datasets had been created for different specialties of doctors. Upon training the datasets, 4 different neural models with unidentical characteristics were developed. All the five classifiers were loaded into the backend in the h5 file format. The backend was developed with the Flask framework, so that it could communicate with the frontend. The frontend was developed using HTML and CSS. For every selection or request made on the frontend, it communicated with the Flask app through a local host address. The backend was connected with the real-time database on Firebase. For every request from the frontend, the backend pulled the data from the database in a predefined csv format and input it in the classifier in the same format as the training dataset.

p- ISSN 2528-1410 | DOI: https://doi.org/10.17509/ijost.v5i2.24585 | e- ISSN 2527-8045

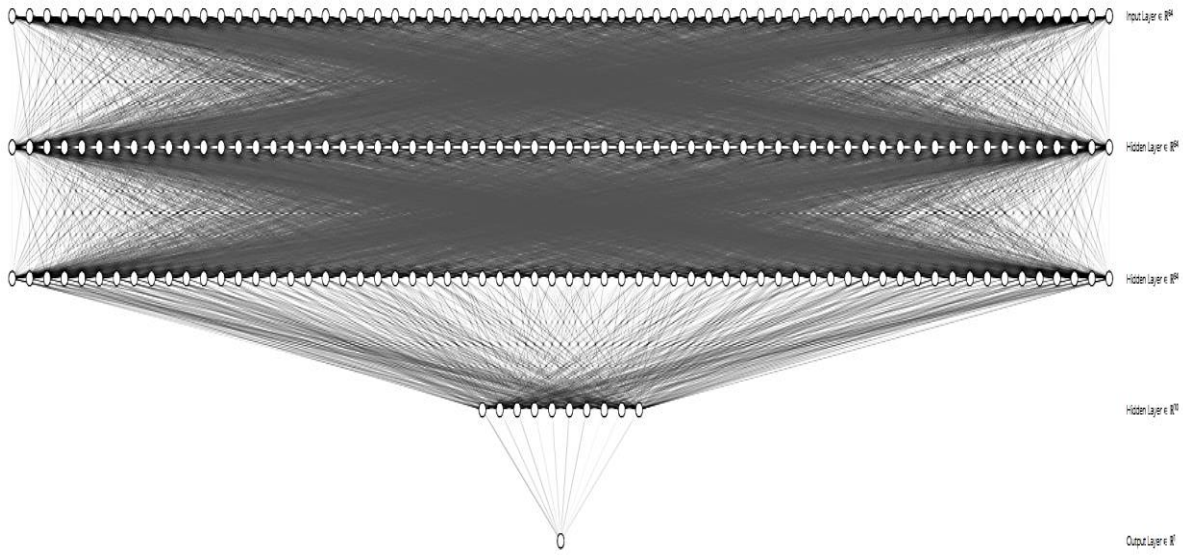

Figure 3. The architecture of the DNN that was developed for this application. As evident the input layer has 64 neurons followed by two hidden layers of 64 neurons each and one with 10 layers. The final output layer has one neuron.

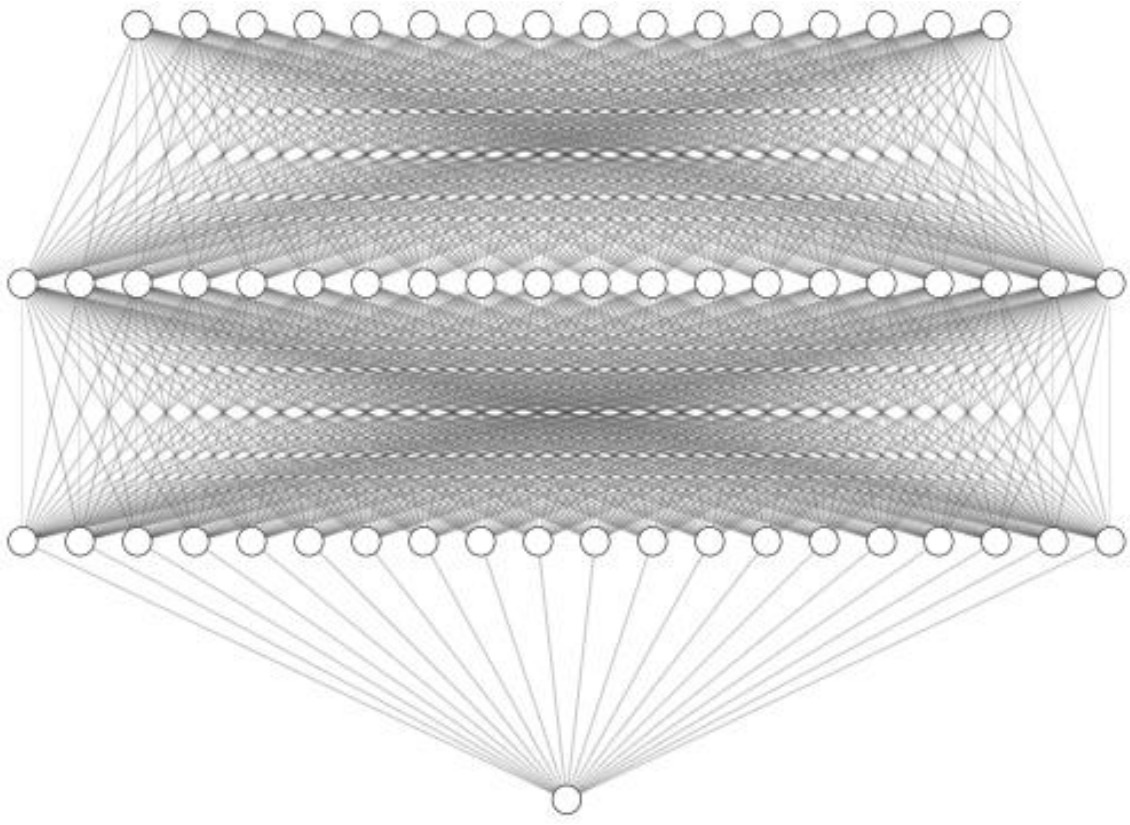

Figure 4. The architecture of the DBN that was developed for this application. It can be seen that the architecture is very similar to that of DNN.

The classifier read the parameters and according to its training, provided the recommendation result in binary. The binary data was mapped to their hospital names, then the name, location and phone number of the recommended hospital was displayed on the frontend. Some changes were made in case of the COVID-19 classifier. The data was pulled from a different database. The classifier read the parameters and provided the recommendations in binary. Additionally, the 'Prior Experience' column was checked. If the recommended hospitals had a 1 in that column, on the output end the same was informed as a statement, i.e. "Prior Experience with COVID-19: YES" Else, it was tagged 'No'.

The Hospital Bed Updation Tool is our implementation of a hospital side bed management system. The hospital administration is responsible for providing accurate number of beds available in the hospital on a regular interval. These changes will be committed to the Firebase database. The Bed Updation Tool is a simple website which takes the input from the user and makes the necessary modifications in the database. It was developed entirely using HTML, CSS and JavaScript. The website provides the option to choose the Hospital for which the change is supposed to be made, and an input form to fill in the latest data. Then it makes the changes in the Firebase database as necessary.

2.5 Raspberry Pi and Fingerprint Sensor

In this work we propose a method to keep count of number of doctors in a hospital on duty. In this method we have used a Raspberry Pi and a R307 fingerprint scanner module interfaced to it. The Raspberry Pi is connected to the real time database. The Pi stores the fingerprints of all the hospitals that work in the hospital. For each stored fingerprint, an odd number of punch ins adds 1 to the Doctors on Duty column in the real time database. And an even number of punch ins 1 is deducted from the column. This way, this system keeps count of the number of doctors on duty in a particular hospital and updates the changes in the database. It should be noted that each Raspberry Pi has been programmed to make changes in the 'Doctors on Duty' column under its specified hospital. Meaning a Pi programmed to make changes for Hospital 1, will not make changes in the 'Doctors on Duty' column under Hospital 2. Essentially, they are limited to the hospitals they are programmed for.

3. RESULTS AND DISCUSSION

Upon running our programs and training our models, the results we found have been discussed in this section. The Raspberry Pi fingerprint scanner worked as expected and updated the 'Doctors on Duty' column in real-time. The real-time firebase synchronized well with the Raspberry Pi (**Figure 5**).

The neural networks that were chosen for analysis performed as expected and gave results. First the CNN was analyzed and gave an accuracy of 71%. This percentage was because of the addition of extra dimensions to the data and the lack of a more diverse dataset. Moreover, CNNs are good for image data as they extract features from the images. In this case, the dataset was one dimensional, thus use of feature extraction could have taken away vital information from the dataset. Although the accuracy was not poor, in the case of healthcare an inaccuracy of 29% could be significant. After training, the Model Loss and model accuracy graphs were plotted (**Figure 6**).

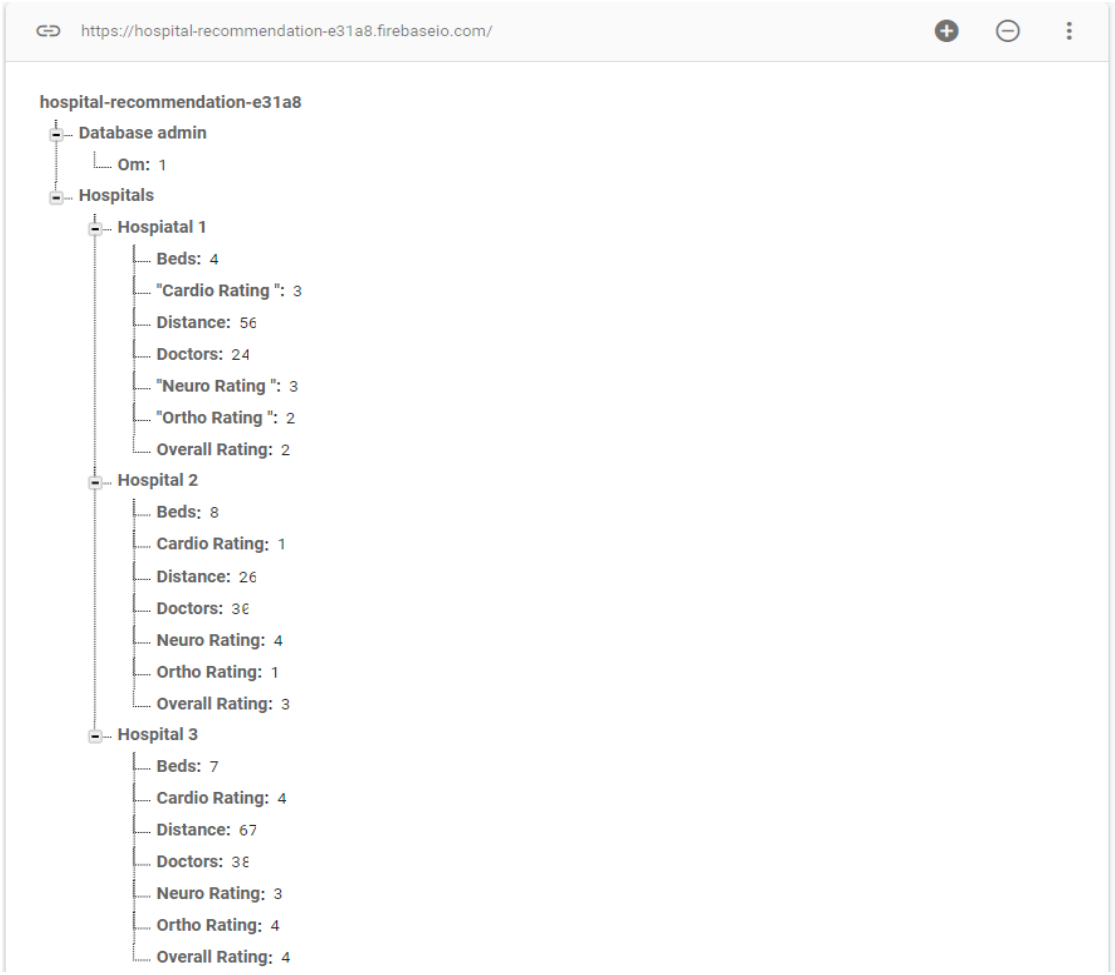

Figure 5. The Real-Time database that was developed on Google Firebase.

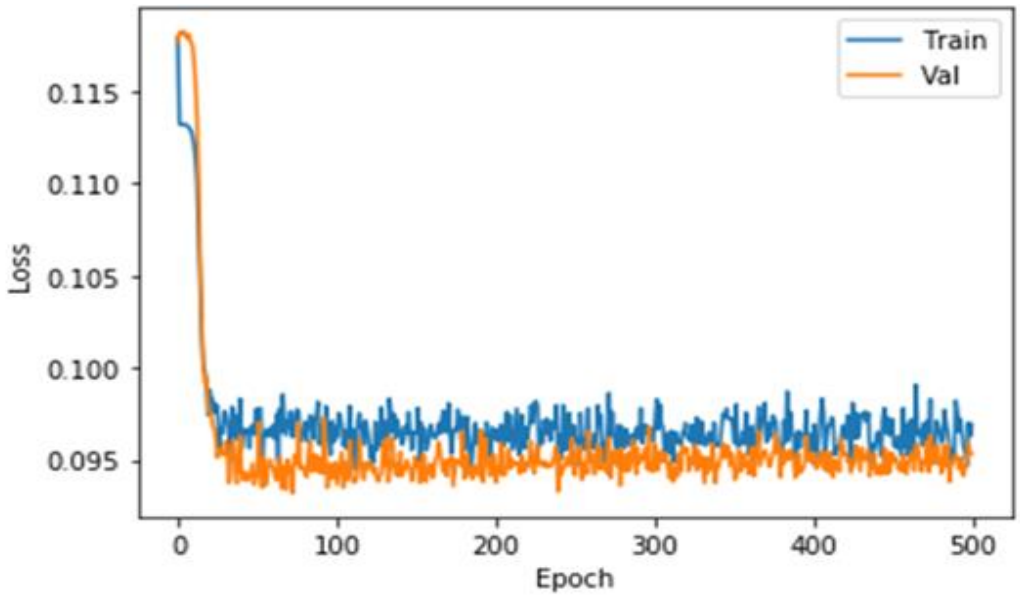

Figure 6. The Model Loss for the CNN.

As observable from the Model Loss graph, the loss falls down to a specific value and then oscillates around that value. This shows that the fit between the model and the data is not too good (**Figure 7**).

Next, in the Model Accuracy graph, it is visible that the accuracy rises to about 72% in the first few epochs. After that the accuracy value oscillates around the mean value of 72%.

Due to the low accuracy the CNN could not be used as a classifier for the website. Next, the Perceptron model was implemented and analyzed. The dataset was fed as is into the network without any addition of dimensions or other changes. The MLP gave an impressive accuracy of 95%. Since it is a linear classifier, it could not become more accurate than this. It also means that even in real-world use we might get a fraction of data which a linear classifier will not be able to classify accurately. The accuracy is impressive, but due to its shortcomings as a linear classifier, the MLP could not be used in the backend (**Figure 8**).

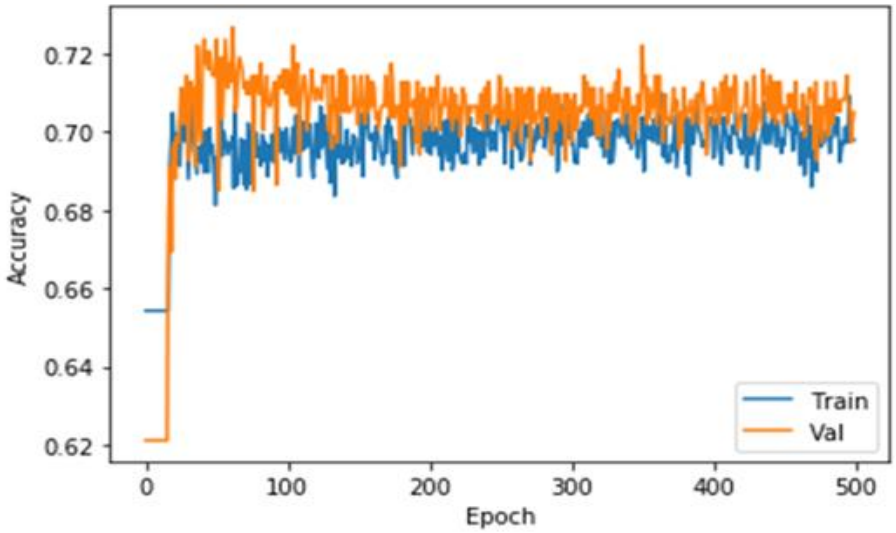

Figure 7. The Model Accuracy for the CNN.

```
[ ]  seed(1)
     # load and prepare data
     filename = 'RBF_dataset (1).csv'
     dataset = load_csv(filename)
     for i in range(len(dataset[0])-1):
       str_column_to_float(dataset, i)
     # convert string class to integers
     str_column_to_int(dataset, len(dataset[0])-1)
     n_folds = 3
     l_rate = 0.05
     n_epoch = 500
     scores = evaluate_algorithm(dataset, perceptron, n_folds, l_rate, n_epoch)
     print('Scores: %s' % scores)
     print('Mean Accuracy: %.3f%%' % (sum(scores)/float(len(scores))))

     Scores: [99.09154437456324, 97.48427672955975, 90.91544374563243]
     Mean Accuracy: 95.830%
```

Figure 8. The Model Accuracy for the ANN.

Since the MLP was not implemented using Keras, graphs like Model Loss and Model Accuracy could not be plotted. Instead, the accuracy was computed through a comparison between the predicted output of the network and the actual output in the test set.

The DNN was tested next and a higher accuracy was expected because of its fully connected feature. As expected, an accuracy of 98% was logged. The model accuracy and model loss graphs were plotted and the fit between the model and the dataset was found to be much better than that of the CNN. Moreover, the binary cross entropy loss function seemed to perform better overall as compared to Huber Loss. The DNN performed a better job at classification overall. Accuracy wise, the DNN could be considered to be used in the backend. Without the results of the DBN a conclusive decision could not be taken (**Figures 9 and 10**).

Finally, a DBN was developed and tested. The DBN gave an accuracy of 99%. This was expected as DBNs are generative networks which perform well even when the dataset is small. Because of the presence of RBMs, these networks learn the data more comprehensively. The only downside of this model was its long training time. While other models trained within 5 minutes, the DBN took nearly 10 minutes for 200 epochs. This raises questions on the tradeoff of accuracy and training time when bigger datasets are involved. DBNs had lost their popularity after the 2000s. This work however, shows that in specific cases of classification, DBNs can outperform other neural networks (**Figure 11**).

DBNs are not available as predefined functions in Keras, thus parameters like Model Loss and Model Accuracy could not be plotted against epochs. Even with an accuracy of nearly 100 % DBNs could not be chosen for the website backend (**Figure 12**).

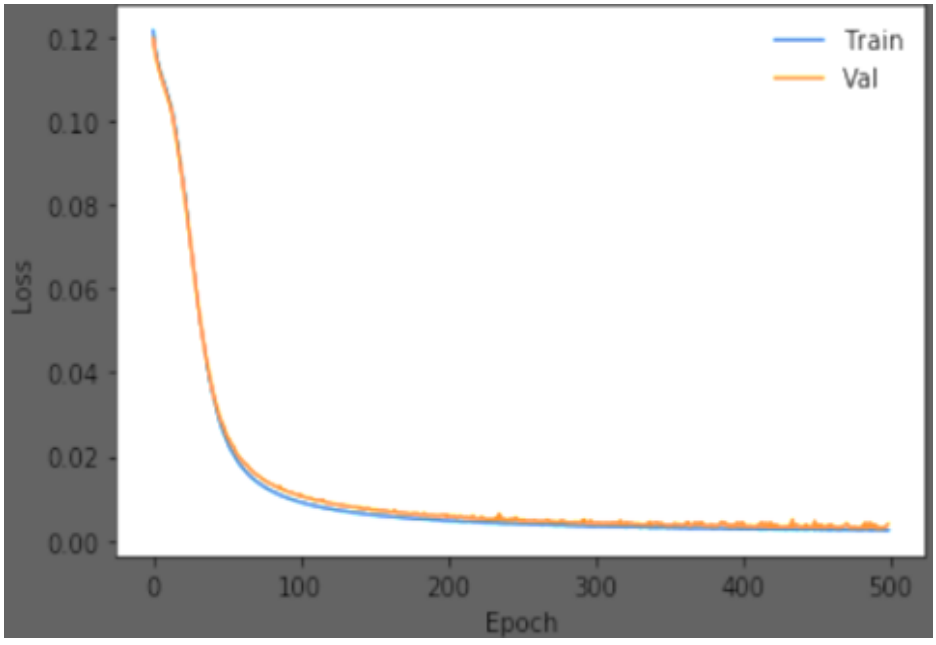

Figure 9. The Model Loss for the DNN.As observable the model loss decreases very gradually and keeps reducing until the end of the 500th epoch unlike the CNN.

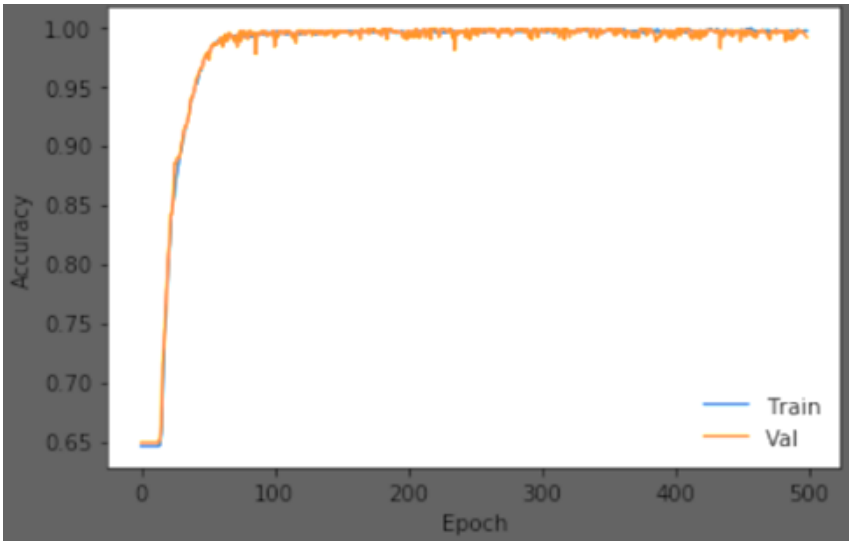

Figure 10. The Model Accuracy for the DNN.As observable the model accuracy increases to its maximum within the first 100 epoch and then remains more or less constant over the next 400 epochs.

```
[ ]  from sklearn.metrics.classification import accuracy_score
     print('Done.\nAccuracy: %f' % accuracy_score(Y_val,Y_pred))

     Done.
     Accuracy: 0.998447
```

Figure 11. The Model Accuracy for the DBN.As observable the model accuracy was calculated by a comparison between the predicted values and test set values.

```
[ ]  Y_pred = classifier.predict(X_val)
     Y_pred=np.asarray(Y_pred)
     Y_pred

     array([0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1,
            1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0,
            1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0,
            1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1,
            0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1,
            1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0,
            0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1,
            0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
            0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 0, 1, 0,
            1, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1,
            0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0,
            0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0,
            0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0,
            1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0,
            0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0,
            1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0,
            1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1,
            0, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 1,
            0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
            0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0,
            0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0,
            1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1,
            1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1,
            0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1,
            0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1,
            0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1,
            0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0,
            0, 1, 0, 1, 0, 1])
```

Figure 12. Output format of a prediction done by the DBN classifier on the test set. 1s refer to the hospitals to be recommended and the 0s denote the hospitals not to be recommended.

p- ISSN 2528-1410 | DOI: https://doi.org/10.17509/ijost.v5i2.24585 | e- ISSN 2527-8045

This is because of the problem of reproducibility. Since DBNs are not found natively on Keras API, they cannot be loaded using the Keras 'load_model' command. Moreover, the repository used does not provide the option to reload the model. The model could be trained on the website backend and then the freshly trained model could be used, every time the website is launched. This however, would increase the website loading time every time it is launched. Although a custom DBN network can be built by customizing every hyperparameter manually on TensorFlow, this is not the aim of the project and is beyond its scope. Since this research is not to build better implementable versions of Neural Networks, we did not go on to build a DBN over the TensorFlow framework. That said, with further work, it can be done. A possible explanation to the absence of DBNs on Keras is its lack of popularity. Only recently have DBNs been in popular use once again. Therefore, with consideration to the factors stated above the DNN was chosen to be implemented on the website backend. After choosing DNNs to be used at the backend of the website, it was also trained on other datasets. The accuracy results acquired until now were only from the training of the General Model dataset. Hereafter, the DNN was trained on the Neurology, Cardiology, Orthopedics and COVID-19 Model datasets. In all the cases the accuracy was found to be between 98% and 99%. Then, all the different classifiers were saved as h5 format files so they could be used on the website backend (**Table 1**). The table summarizes the accuracies of the different neural networks and provides an insight to how different neural networks can be used in this unique anchorless recommendation problem.

The website was developed using HTML, JavaScript and CSS, whereas the DNN classifier was deployed in the background as a Flask Framework built upon Python. The website worked dynamically and with the help of buttons switched between the different classifiers for recommendation of different kind of specialists (**Figure 13**).

Table 1. Summary of accuracies and activation functions of neural networks analyzed

Model	Activation Function	Accuracy
Convolution Neural Network	SoftMax	72
Perceptron Network	Heaviside	95
Dense Neural Network	Sigmoid	98
Deep Belief Network	Sigmoid	99

Figure 13. The website built for this work. The dropdown list on the right provides options for doctors from four different specialties.

Selections made on the website governed the use of the corresponding classifier in the backend. Therefore, the website used different classifiers, trained on different datasets, to predict results for the different departments. The datasets that were used to train the different classifiers varied on the basis of their weightage formula as shown previously. The presence of different classifiers made the website more accurate and it also meant that the same classification mechanism was not used for predicting very different kinds of results (**Figures 14 and 15**).

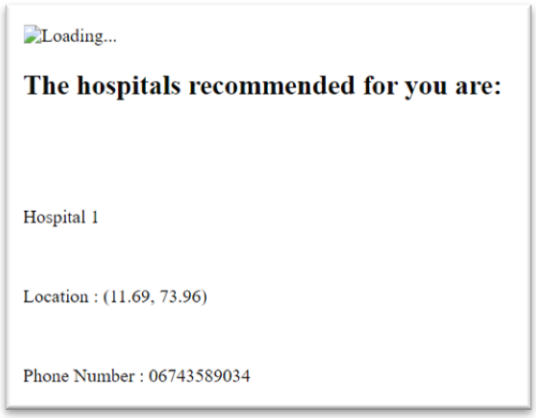

Figure 14. The format of the final result on the website.

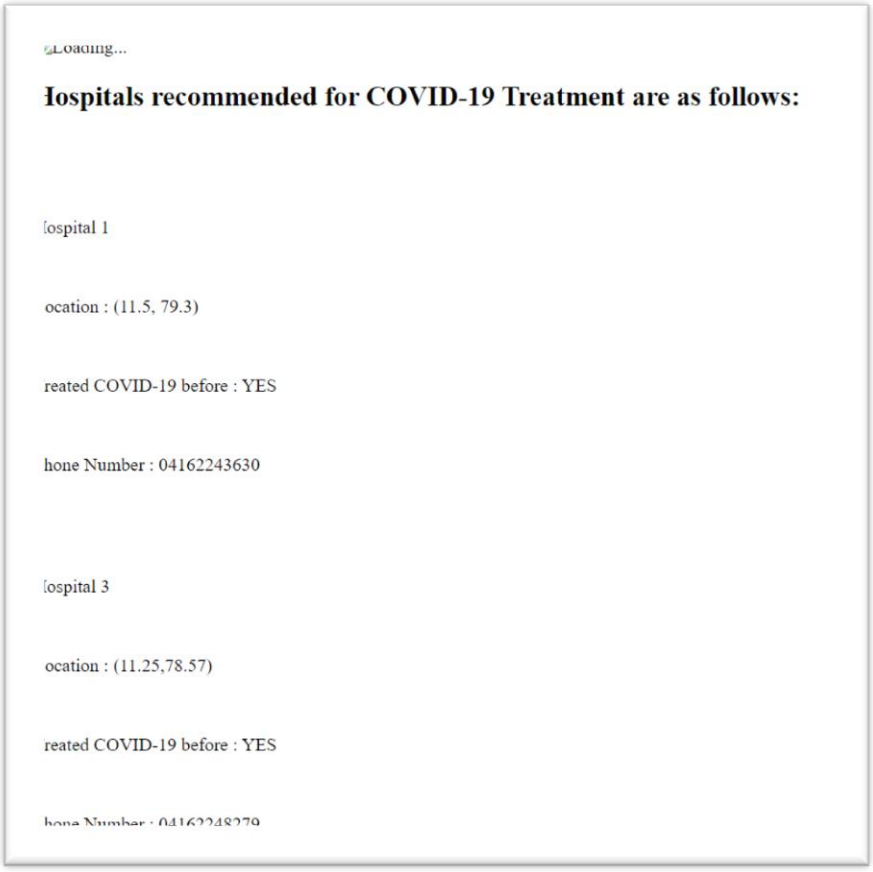

Figure 15. Output format for COVID-19 recommendation

The website recommends a hospital while providing its location and a contact number to make prior communication with the hospital. In case of COVID-19 recommendations, prior experience of hospitals is also reported, so as to help make more informed decisions. The website was tested on a database that had three hospitals listed within it. The orthopedics classifier found one hospital to be most suitable and thus recommended it. The COVID classifier has the same three hospitals on its database albeit with some additional information and modified formatting. The Classifier found two of the hospitals suitable and recommended those. The result attached is from the orthopedics and COVID-19 classifiers, although the other classifiers worked equally effectively and gave a recommendation for each of the different specialists. The Hospital Recommendation Tool worked on a database having three hospitals listed within it (**Figure 16**).

In the image above dummy numbers have filled in to demonstrate the working of the Updation tool. The next image shows the changes made to the Firebase database (**Figure 17**)

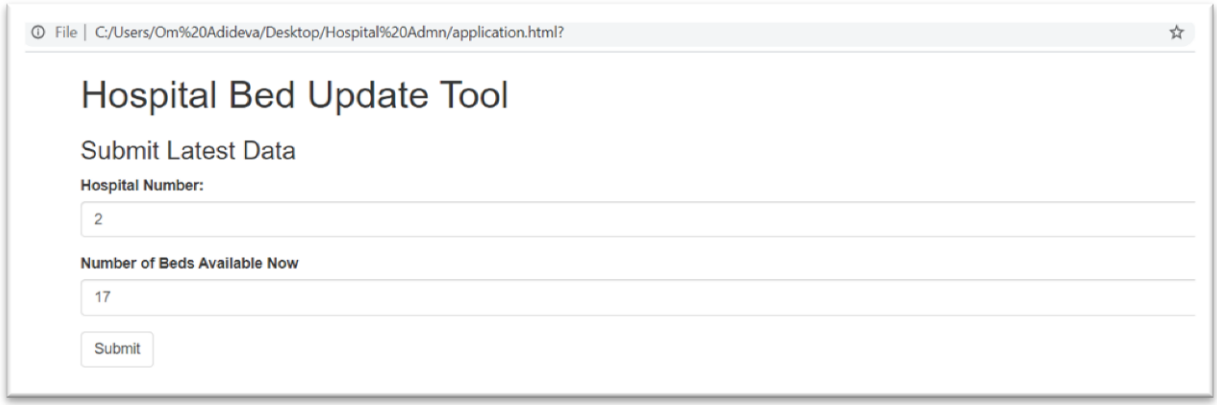

Figure 16. Interface of the Hospital Bed Updation Tool

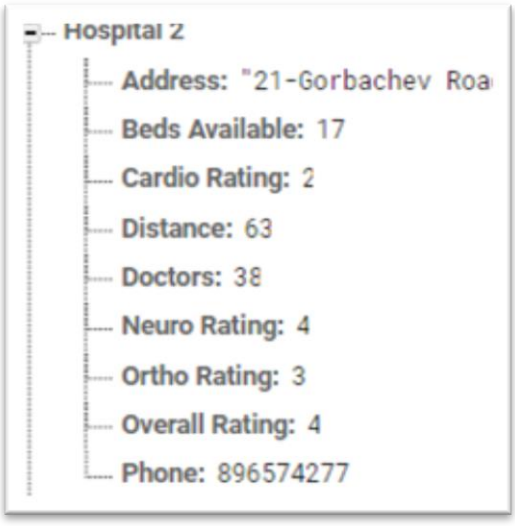

Figure 17. Changes in the real time database

This could be improved to automatically plot the route through Google Maps. More importantly the website could be transformed into an app for smartphones to improve the speed of this process. Voice assistants could also be integrated to provide a faster, contactless search service. Integration into Android Auto and similar services could be explored, as this would mean the navigation data could directly be sent to the car's dashboard. Secondly, some cases did arise during testing when the classifiers did not recommend any hospital. This can be improved by using fallback rules, such as distance comparison or other similar rules. We find that a hybrid model i.e. one that makes use of deep learning, fallback rules and common-sense computing (Cambria *et al.*, 2009) will make a better recommendation model and can be explored in further works. Next, problems caused due to data insufficiency can be combated. For example, the COVID 19 Hospitals data is not available yet. Therefore, an artificial dataset had to be created. In the future, as more data about hospitals is published, our work can be expanded to use more effective parameters and use more accurate data. Although a variety of data related to COVID 19 is available most are useful for predicting the trends of the pandemic or to predict the load that hospitals can take. None of the data is useful for a recommendation system explored in this work. The database created in this work can be held as an example and data can be published in accordance with that model. With accurate, verified and huge data the performance of the recommendation system can be increased manifold. Moreover, the work could be improved by integrating it into a healthcare ecosystem, such as one proposed by Schiza *et al.* (2018) in a study from 2018. This would provide more standardized data, more universally used rankings and better interoperability between different organizations. The work could be expanded to include parameters like cost of services of a hospital, insurance policy compatibility, readmission rate, quality of doctors etc. These parameters would require much more data, though, and thus it is logical to integrate this work with some existing ecosystem database. This system was built as a user application for any kind of user ranging from the patient itself, ambulance service providers or even the first responders during an emergency. In this vein, the application could be integrated with Smart Ambulance systems (Gupta *et al.*, 2016), thereby improving the Smart Ambulance's decision-making capabilities. The only hurdle that hinders this work is a lack of use of standardized ratings over the world. Thus, there is no one database that the system can latch on to and find the relevant data for every hospital. A possible solution for this is combining this work with another work done on hospital ranking algorithms (Albahri *et al.*, 2019). Since it uses body sensor data to rank hospitals it would be virtually universally compatible. Integrating both these works would ensure the availability of data. This work is still considered by us to be a facilitator and not a stand-alone tool. The better data that this system works with, the better results it will provide. An exemplary database was created for this work, which could be used as a standard to develop databases in real hospitals so as to facilitate the use of this work. Another work around that we have found is to create an intermediate database that is of the format needed by the neural nets. Data could be pulled from other existing databases in real time, collated and then filled in the right spots of the intermediate database. Although this solves the problem of having to create a new database, it poses another set of problems. A generic helper function cannot be made

that would operate across all databases. Thus, our final recommendation is to use this work in tandem with other efficient ranking tools and in situations that provide an opportunity to create detailed databases.

3. CONCLUSION

This work only serves as a proof of concept of the potential of neural networks in the field of healthcare. The website itself is in a rudimentary phase. Further work could be done to improve the functionality and aesthetics of the website. Although this solves the problem of having to create a new database, it poses another set of problems. A generic helper function cannot be made that would operate across all databases. Thus, our final recommendation is to use this work in tandem with other efficient ranking tools and in situations that provide an opportunity to create detailed databases.

4. AUTHORS' NOTE

The author(s) declare(s) that there is no conflict of interest regarding the publication of this article. Authors confirmed that the data and the paper are free of plagiarism.

5. REFERENCES

Albahri, A., Albahri, O., Zaidan, A., Zaidan, B., Hashim, M., Alsalem, M., Enaizan, O. (2019). Based multiple heterogeneous wearable sensors: A smart real-time health monitoring structured for hospitals distributor. *IEEE Access, 7*, 37269-37323.

Albertbup. (2018). albertbup/deep-belief-network. Retrieved from https://github.com/albertbup/deep-belief-network, retrieved on April 20, 2020.

Cambria, E., Hussain, A., Havasi, C., & Eckl, C. (2009). Common sense computing: From the society of mind to digital intuition and beyond. In *European Workshop on Biometrics and Identity Management* (pp. 252-259). Springer, Berlin, Heidelberg. https://sentic.net/common-sense-computing.pdf, retrieved on April 20, 2020

https://cpb-us-w2.wpmucdn.com/u.osu.edu/dist/c/28860/files/2016/08/Star_Rtngs_CompMthdlgy_052016-148w094.pdf, retrieved on April 20, 2020.

https://globalepidemics.org/our-data/hospital-capacity/, retrieved on April 20, 2020.

https://www.kaggle.com/cms/hospital-general-information, retrieved on April 20, 2020.

Gallagher, J. (2020, April 21). Coronavirus vaccine: When will we have one? *BBC News*. https://www.bbc.com/news/health-51665497, retrieved on April 21, 2020.

Gupta, P., Pol, S., Rahatekar, D., & Patil, A. (2016). Smart ambulance system. *International Journal of Computer Applications*, *6*, 23-26.

Habibi, A. N., Sungkono, K. R., & Sarno, R. (2019). Determination of Hospital Rank by Using Technique For Order Preference by Similiarity to Ideal Solution (TOPSIS) and Multi Objective Optimization on the Basis of Ratio Analysis (MOORA). *2019 International Seminar on Application for Technology of Information and Communication (iSemantic)* (pp. 574-578). IEEE. https://ieeexplore.ieee.org/abstract/document/8884278, retrieved on April 20, 2020.

Schiza, E. C., Kyprianou, T. C., Petkov, N., & Schizas, C. N. (2018). Proposal for an ehealth based ecosystem serving national healthcare. *IEEE journal of biomedical and health informatics, 23*(3), 1346-1357.

Tabrizi, T. S., Khoie, M. R., Sahebkar, E., Rahimi, S., & Marhamati, N. (2016, July). Towards a patient satisfaction based hospital recommendation system. In *2016 International Joint Conference on Neural Networks (IJCNN)* (pp. 131-138). IEEE. https://ieeexplore.ieee.org/abstract/document/7727190, retrieved on March 20, 2020.

Verma, K. J. M. (2020). COVID-19: China shuts down largest makeshift hospital in Wuhan. *Business Insider India*. https://www.businessinsider.in/international/news/covid-19-china-shuts-down-largest-makeshift-hospital-in-wuhan/articleshow/75157291.cms, retrieved on April 15, 2020.

Wen, H., Song, J., & Pan, X. (2020). Physician Recommendation on Healthcare Appointment Platforms Considering Patient Choice. *IEEE Transactions on Automation Science and Engineering, 17*(2), 886-899.

Zhang, G. P. (2000). Neural networks for classification: a survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *30*(4), 451-462.