IAES International Journal of Artificial Intelligence (IJ-AI)
Vol. 14, No. 1, February 2025, pp. 570~583
ISSN: 2252-8938, DOI: 10.11591/ijai.v14.i1.pp570-583

Accurate prediction of chronic diseases using deep learning algorithms

Ronald S. Cordova[1], Rolou Lyn R. Maata[1], Malik Jawarneh[2], Marwan I. Alshar'e[3], Oliver C. Agustin[4]
[1]Department of Computing Sciences, Gulf College, Muscat, Oman
[2]College of Computer Science and Informatics, Amman Arab University, Amman, Jordan
[3]Faculty of Computing and Information Technology, Sohar University, Sohar, Oman
[4]Vera Equinox Technologies Incorporated, Cabanatuan City, Philippines

Article Info

Article history:

Received Feb 11, 2024
Revised Jul 4, 2024
Accepted Jul 26, 2024

Keywords:

Chronic diseases
Data-driven healthcare
Deep learning
Healthcare analytics
Predictive analytics

ABSTRACT

In this paper, the researchers studied the effects of different activation functions in hidden layers and how they impact the overfitting or underfitting of the model in the multiclass prediction of chronic diseases. This paper also evaluated the effects of varying the number of layers, and hyperparameters and its impact on the accuracy of the model and its generalization capabilities. It was found that exponential linear unit (ELU) does not have a significant advantage over rectified linear unit (ReLU) when used as an activation function in the hidden layer. Additionally, the performance of softmax function, when used in the output layer, is the same as a classic sigmoid output activation function. In terms of the ability of the model to generalize, the researchers achieved a classification accuracy of 100% when the trained model was used to predict unseen data. Through this research, the researchers should be able to assist medical professionals and practitioners in Oman in the validation and diagnosis of chronic diseases in clinics and hospitals.

Corresponding Author:

Ronald S. Cordova
Department of Computing Sciences, Gulf College
Muscat-133, Oman
Email: ronald@gulfcollege.edu.om

1. INTRODUCTION

In Oman, there are 100 to 120 reports of medical errors–classified as harmful, life-threatening and fatal, which was the key discussion during the 8th Oman health exhibition and conference [1]. In 2015, the National Academy of Medicine in the United States (US) reported that 20% of most people will get an incorrect diagnosis at least once in their lives [2], these statistics account for over 12 million adults who seek outpatient medical care translating to about 5% of adults, or 1 out of 20 adult patients [3]. In Sulaimaniyah City, Iraq, it was found that non-physician healthcare workers are the primary cause of misdiagnosis and mistreatment of COVID-19 patients where patients were falsely diagnosed as having typhoid (63%), influenza (14%), pneumonia (9%), gastroenteritis (5%), common cold (4%), brucellosis (4%), and meningitis (1%) [4]. In 2021, it was found that missed vascular events, infections, and cancers account for 75% of serious harms caused by misdiagnosis [5].

Globally, world health organization (WHO) states that diagnostic errors occur in about 5% of adults in outpatient care settings, more than half of which have the potential to cause severe harm [6]. The result of this study will be a very relevant and indispensable tool in hospitals and healthcare providers in the early diagnosis of diseases which could have been prevented if accurately detected early. The disease prediction software is ideal for use in remote and rural areas where the availability of doctors is scarce.

In this paper, the researchers developed a deep learning model for the implementation of an artificial intelligence (AI)-assisted chronic disease prediction in the tracking system for chronic diseases (TSCD) project. Table 1 presents the deep learning frameworks toolkits that the researchers have considered in this project for evaluation. The objective of this proposed project is to develop a chronic disease multiclass prediction model to be used in TSCD. This prediction model is crucial to TSCD to help doctors in the diagnosis and verification of patient illnesses in hospitals, health centers and clinics. Specifically, this research paper aims to:

– Analyze the impact of different activation functions on the performance of hidden layers in neural network specifically, exponential linear unit (ELU) and rectified linear unit (ReLU).
– Investigate the relative performance of the sigmoid and softmax functions as output layer activation functions in neural networks.
– Compare the different performance and accuracy of the classification models with varying hyperparameter settings.

Table 1. Machine/deep learning framework

Framework	Programming languages	Jupyter lab	Platform	Open-source	Deep learning
PyTorch	Phyton, C++, Julia, C#	Yes	Linux, macOS, Windows, Android	Yes	Yes
TensorFlow	Phyton (keras), C/C++, Java, Go, Javascript, R, Julia, Swift, C#	Yes	Linux, macOS, Windows, Android	Yes	Yes
Apache MXNET	C++, Phython, Go, R, Scala, Perl, Clojure, C#	Yes	Linux, macOS, Windows, AWS, Android, iOS, JavaScript	Yes	Yes
LightGBM	Phython, R, C++, C, C#	Yes	Linux, macOS, Windows	Yes	No

Out of numerous deep learning frameworks evaluated [7], the researchers narrowed down the choice to TensorFlow because of its maturity, scalability, and easy-to-develop models, widely used at the production level in the industry, compared with the others. Pytorch [8] is a machine learning framework based on the Torch library, used for applications such as computer vision and natural language processing. Pytorch was released in September 2016. TensorFlow [9] is an end-to-end open-source machine learning platform. Released on November 9, 2015, it features a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in machine learning and developers easily build and deploy machine learning-powered applications.

Apache's open-source spin on a deep learning framework called MXNet which supports building and training deep learning networks in multiple languages [10], [11] was released on May 10, 2022. With efficiency and scalability in mind, MXNet is very fast, lightweight and portable [12], [13]. LightGBM [14] or light gradient-boosting machine, is a free and open-source distributed gradient-boosting framework for machine learning based on decision tree algorithms and used for ranking, classification, and other machine learning tasks. LightGBM was released in 2016.

According to the benchmark results published by viso.ai [15], TensorFlow and PyTorch show equal accuracy, the memory usage of TensorFlow is lower but the training time is substantially higher. According to the results, TensorFlow is often used for production applications due to its speed and scalability. It has an extensive library of pre-built models and which makes it ideal for the development of multiclass models. This research is relevant to the project due mainly to the fact that the optimized model as the result of this project, will be used in the implementation of the AI-assisted chronic disease evaluation system. This will be the core part of the TSCD to assist medical practitioners in Oman to be able to validate patients' diagnosis using AI.

2. METHOD

This section describes the method that was adopted in this research, comprised of the procedure in subsection 2.1. In subsection 2.2, the methods in building the prediction models were described, how to avoid overfitting and underfitting, and identifying the optimal learning rate. The full source code to replicate the simulations and experiments in this paper is available with the authors [16].

2.1. Research design

In the development of the deep learning model for the prediction of chronic disease, the framework in Figure 1 was established. In the modeling stage, we created 5 models with varying hyperparameters, randomized the dataset, optimized the learning curve, and the regularization parameters. Model training is completed when the stopping criteria have been achieved or the number of epochs has been reached without a considerable decrease in error rate. This process is performed with the other 4 models. In the performance evaluation stage, we select the two models with the highest prediction accuracy on unseen data. We used Jupyter Lab [17] in Anaconda Navigator and Google Colab [18] to write python script codes to perform data processing, modeling and performance evaluation of the resulting models in this paper.

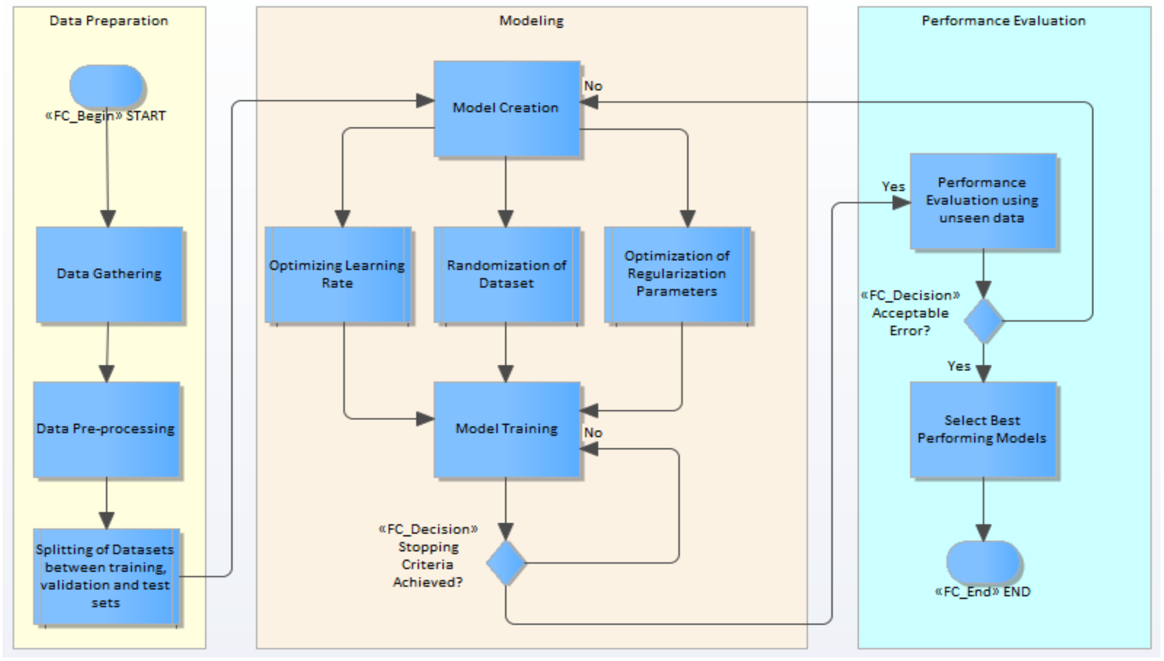

Data Preparation
Modeling
Performance Evaluation
«FC_Begin» START
Data Gathering
Data Pre-processing
Splitting of Datasets between training, validation and test sets
Model Creation
No
Optimizing Learning Rate
Randomization of Dataset
Optimization of Regularization Parameters
Model Training
No
«FC_Decision» Stopping Criteria Achieved?
Yes
Performance Evaluation using unseen data
«FC_Decision» Acceptable Error?
Yes
Select Best Performing Models
«FC_End» END

Figure 1. Framework for prediction models for chronic diseases

2.1.1. Data preparation

Data cleanup and preprocessing are fundamental steps in machine learning tasks. Prediction models rely on data to learn and make predictions. Raw data often contains inconsistencies, errors, missing values, extra columns, and redundant data that can drastically affect the prediction capability of the model.

In data preparation, a localized dataset focused on the Middle East region is not readily available. The dataset found in [19] was used in this paper, performed data cleanup and pre-processing on it, and split the dataset into training, validation, and test sets. The three sets of data will then be used in the modelling stage.

2.2.2. Modeling

The modelling stage utilizes the resulting data produced by the data preparation stage. In this stage, 5 multiclass models were created with varying number of hidden layers, identified the optimal learning rate, randomized the data sets, and set the regularization parameters and stopping conditions. Training of the models are then performed until a stopping criterion has been reached.

a) Overfitting and underfitting

Overfitting and underfitting is a fundamental issue [20] in machine learning. It results to a poor generalization capability of the model to accurately classify unseen data. This difficulty was also encountered in the previous research utilizing neural network models [21] and support vector machines [22], [23] in the multiclass classification of milled rice.

When training a model for several epochs, the accuracy of the model on the validation set would peak then after some time it will start to decrease or stagnate. This scenario exemplifies the overfitting of the model to the training data. The opposite of this scenario, underfitting, occurs when the model can be improved on the trained data but the model is not given enough time to learn the relevant pattern in the training data. To demonstrate this problem in TensorFlow [9], the Higgs dataset [24] was used to develop multi-classification models to see how overfitting and underfitting affect the accuracy of the model for this dataset. In Table 2, four models with varying numbers of hidden layers are shown. For these models, the number of output classes is fixed at 17.

Table 2. Deep learning model for higgs dataset with varying number of parameters

#	Model name	Layers					Output shape	Total params
		1	2	3	4	5		
1	TNY	464					16	481
2	SML	928	1,056				32	2017
3	MDM	1,856	4,160	4,160	65		64	10,241
4	LGE	14,848	262,656	262,656	262,656	513	512	803,329

In deep learning, neural network are trained using an optimization algorithm called stochastic gradient descent to minimize the loss function, thereby reducing the error between the network's prediction and the desired results. The learning rate acts as a crucial hyperparameter within this optimization process. We know that larger learning rate may results to missing the optimal solution but [25] has a solution for it. Models in Table 2 used the same configuration and learning rate as shown in Figure 2. This learning rate is expected to hyperbolically decrease by ½ of the base rate at 1,000 epochs, 1/3 at 2, 000 epochs, and so on.

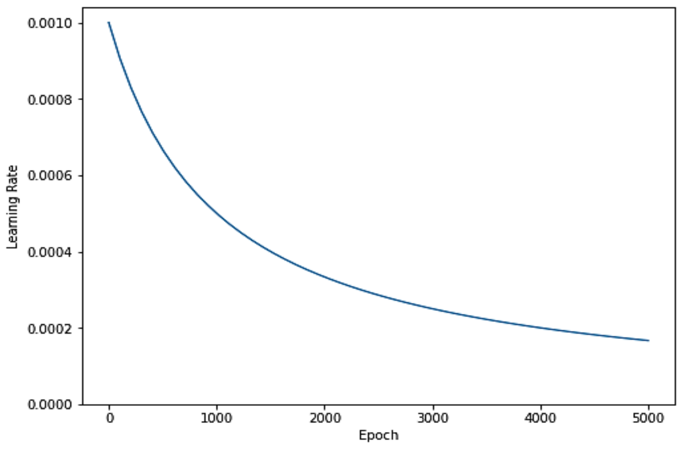


Figure 2. Learning rate

Figure 3 shows the plot between training and validation loss. Lower validation loss indicates better generalization to predict unseen data. The training ends after 200 epochs if there is no progress. In this simulation, the Tiny neural network (TNY) model has avoided overfitting the data while the more complex models (i.e., supervised machine learning (SML), multi-dimensional model (MDM), large gradient estimation (LGE)), tend to overfit the data more quickly. From the result in Figure 3, it was observed that:

- It is normal for both training and validation loss to have a small difference (TINY).
- It is normal to have training and validation loss to be moving in the same direction (TINY).
- When the training loss continues to improve but the validation loss starts to deteriorate, training is most likely close to overfitting (SML, MDM, LGE).
- If the validation loss is increasing (going upward), the model is overfitting (SML, MDM, LGE).

b) Strategies to avoid overfitting of data

Various strategies to avoid overfitting that were used in this paper are discussed in subsubsection 2.4.1 to 2.4.6. The strategies described in this subsubsection avoid overfitting of training data leading to improved generalization capability of the trained model for accurate prediction on unseen data.

– Training with more data

One of the most common strategies is to include the full range of inputs that the model is expected to handle for each classification output. Additional training data may only be relevant to cover new classes and interesting cases. A model trained this way tends to generalize better. Gütter *et al.* [26], it was shown that the training set size does play a role in affecting the robustness of the model against noise. However, this strategy may not work every time. If noisy data were added, this technique wouldn't help. This is the reason why data must be cleansed [27].

– Reducing the complexity of the network

Another strategy is to reduce the complexity of the network similar to what was done in this paper [28]. As shown in Table 2, the LGE model is comprised of a whopping 803,329 parameters, the red dash line in in Figure 3 shows the validation loss is shooting in an upward direction. It is enough to say that in this LGE model, overfitting could have been avoided if a less complicated network had been adopted as shown in the TNY model in Table 2.

– Early stopping

Early stopping is a regularization technique for a deep neural network that stops training when parameters do not result in any improvements on validation sets after a number of epochs [29]. The researchers are using this strategy in this paper by stopping the training after 100 epochs of no improvements. In fact, most if not all machine learning frameworks support this strategy.

– Regularization

Regularization is an excellent technique to address overfitting problems. Some examples of these techniques use linear regression [30], L1 regularization [31], and L2 regularization [32]. Figure 4 shows a training and validation loss for the LGE model that utilizes L2 regularization. This regularized version of the LGE model is competitive with the TNY model, with more resistance to overfitting than the LGE model it was derived from.

– Dropout

Dropout is one of the simplest ways [33] to avoid overfitting using regularization techniques that discard random nodes during training by ensuring that no variables are not co-dependent with one another. It means that the model will not learn redundant details of the input. An example is provided in Figure 5 where LGE utilizes a dropout regularization technique.

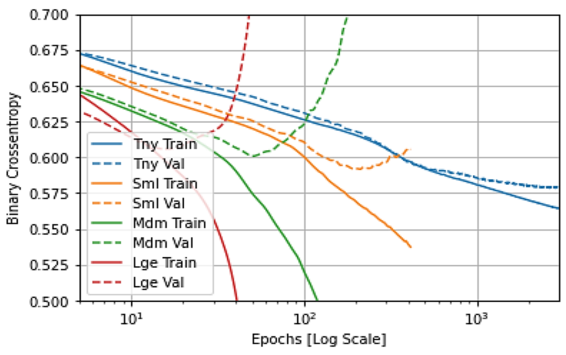


Figure 3. Training vs validation losses for Higgs dataset

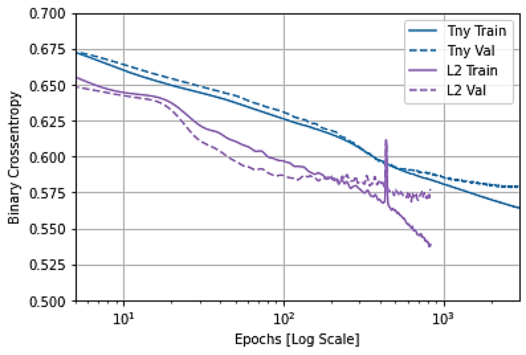


Figure 4. Training vs validation loss for LGE model with L2 regularization technique

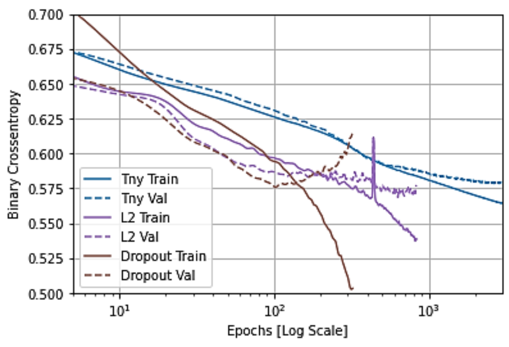


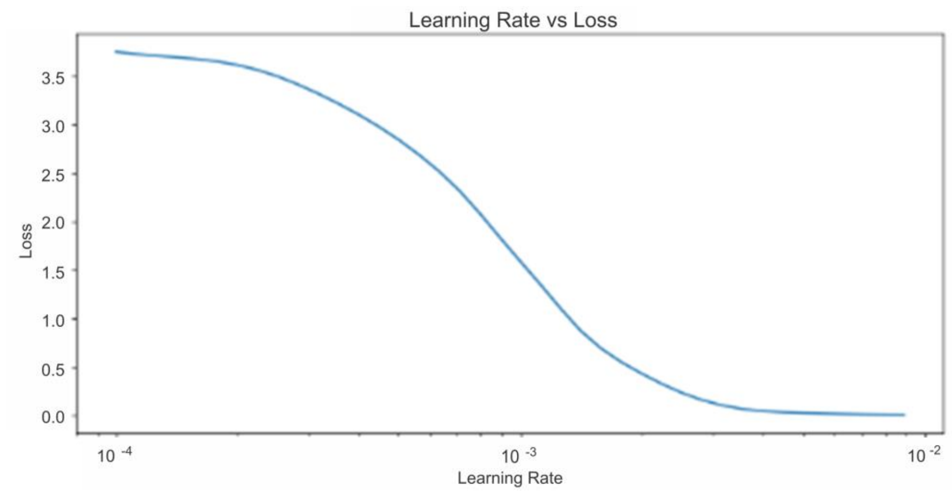
Figure 5. LGE Model with dropout regularization

c) Optimal learning rate estimation

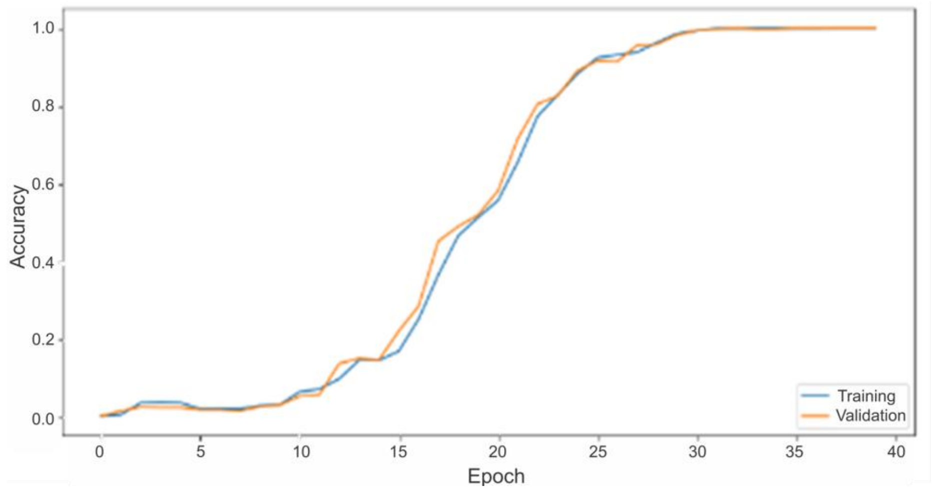
Hyperparameters decide the time and computational cost of training neural network models. They heavily influence the structure, prediction accuracy, and generalization capability of the model. The learning rate is one of those hyperparameters that requires tuning.

If the learning rate is incrementally small, then training is more reliable, but optimization will take more time because of the smaller step size to obtain the minimum of the loss function. On the other hand, if the learning rate was set a bit too high, then the training may not converge or diverge. In this case, the optimizer may not be able to achieve the minimum of the loss function which makes the model worst. Local quadratic approximation

[34], stochastic gradient descent [35] and the method used in super-convergence [25] are some of the learning rate estimation strategies. Figure 6(a) shows the learning rate used, Figures 6(b) and 6(c) shows an excellent convergence of the training/validation loss/accuracy.



(a)



(b)

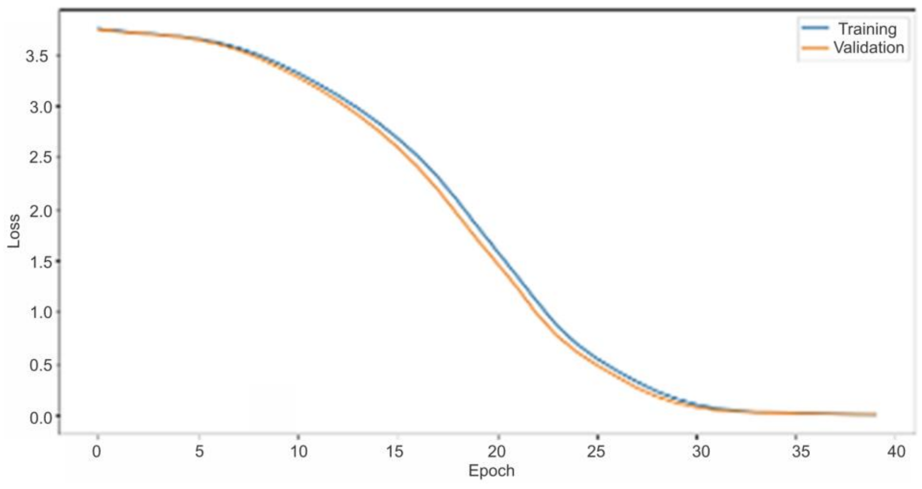


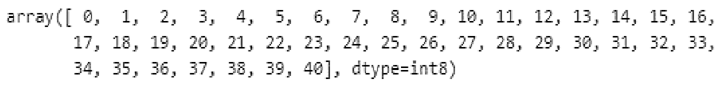
Figure 6. Learning rate, accuracy and losses (a) learning rate, (b) training vs validation accuracy, and (c) training vs validation loss

3. RESULTS AND DISCUSSION

In this section, the development of the classification models built using TensorFlow is explained in detail. Subsections 3.1 and 3.2 describe the considerations when designing and training a multiclass deep learning model. Subsections 3.3 and 3.4 describe in detail how the deep neural network model was developed. Finally, subsection 3.5 describes the performance evaluation and selection of the multiclass deep learning classification model in this paper.

3.1. Dataset cleanup and preparation

The dataset [19] is used in this paper to build the deep learning multiclass model. For this dataset, there are 134 columns in total–132 columns represent distinct symptoms, column #133 represents the different diagnoses of our output classes, and column 134 is an irrelevant column that must be removed. Furthermore, the output categories consisting of 41 diagnoses must be translated into numerical values represented as shown in Figure 7.

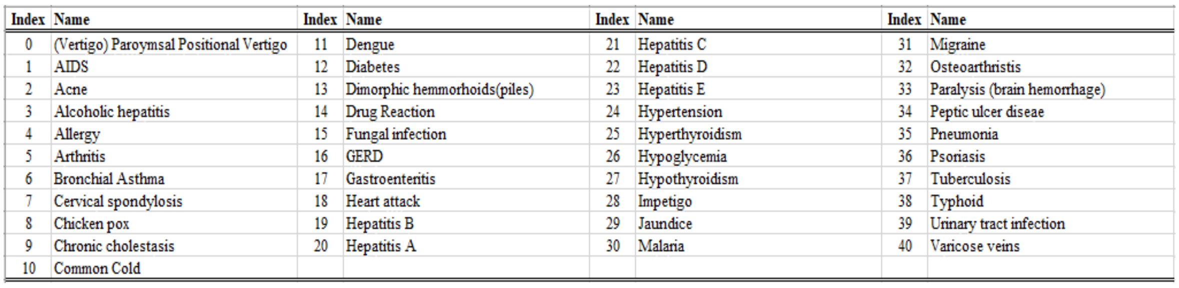


```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33,
       34, 35, 36, 37, 38, 39, 40], dtype=int8)
```

Figure 7. Indices of output categories

3.2. Splitting of the dataset into training, validation and test set

In the training dataset, there are 4,920 rows and 132 features. The training dataset was randomly split into 80/20 percent where the 20% validation set is comprised of 984 rows. The training and validation set will be used for training and fitting the model. The test dataset is comprised of 41 rows and 132 features which is unseen data to be used for testing the performance of the model. The number of features or columns represents various symptoms that can be encountered by a patient. The number of output classes or categories is 41 which represents the possible chronic diseases, as shown in Figure 8.



Index	Name	Index	Name	Index	Name	Index	Name
0	(Vertigo) Paroymsal Positional Vertigo	11	Dengue	21	Hepatitis C	31	Migraine
1	AIDS	12	Diabetes	22	Hepatitis D	32	Osteoarthristis
2	Acne	13	Dimorphic hemmorhoids(piles)	23	Hepatitis E	33	Paralysis (brain hemorrhage)
3	Alcoholic hepatitis	14	Drug Reaction	24	Hypertension	34	Peptic ulcer diseae
4	Allergy	15	Fungal infection	25	Hyperthyroidism	35	Pneumonia
5	Arthritis	16	GERD	26	Hypoglycemia	36	Psoriasis
6	Bronchial Asthma	17	Gastroenteritis	27	Hypothyroidism	37	Tuberculosis
7	Cervical spondylosis	18	Heart attack	28	Impetigo	38	Typhoid
8	Chicken pox	19	Hepatitis B	29	Jaundice	39	Urinary tract infection
9	Chronic cholestasis	20	Hepatitis A	30	Malaria	40	Varicose veins
10	Common Cold						

Figure 8. Chronic diseases (output categories or classes)

3.3. Building a multiclass classification model

This section outlines the process that was followed for building a multiclass prediction model for chronic diseases. Subsubsection 3.3.1, describes the training configuration used for 5 different classification models discussed in this paper. The number of features is 132, representing different symptoms encountered by a patient. The dataset is divided into training, validation, and test data with a number of instances equivalent to 3936, 984 and 41, respectively. This section also provides the settings used during the optimization of the model such as the maximum number of epochs 100, the number of training instances to process (batch size=50) at a time, and the number of steps per epoch 62. Subsubsection 3.3.2 provides the configuration used to decay the learning rate. Subsubsection 3.3.3 presents the models created whose hyperparameter settings range from a total of 481 up to 803,309 hyperparameters and a number of layers from 1 up to 5 hidden layers.

3.3.1. Training configuration

Table 3 presents the configuration for all the models to be built in this paper. The total number of possible symptoms is represented by the total number of features. Using the 80:20 Pareto rule, the researchers split the training set into training instances equal to 3,936 and validation instances equal to 984. The final test data which is the unseen data instances is 41 will be used to evaluate the performance of the models.

Table 3. Deep learning model training configuration

Settings	Output shape
Total number of features	132
Number of validation instances	984
Number of training instances	3,936
Number of unseen data instances	41
Buffer size	100
Batch size (epochs)	50
Steps per epoch	62

3.3.2. Learning rate

Inverse decay function is a strategy for scheduling the learning rate during training. It adjusts the learning rate based on the current iteration step during training. The learning rate is increased as the inverse of the iteration, often written as (1):

$$\text{learning\_rate} = \frac{r}{\left(1+\frac{dr\times s}{ds}\right)} \tag{1}$$

where r-initial learning rate, dr–decay rate, ds–decay step and s–step. For our models, parameters are set in accordance with Table 4. As the training proceeds, learning rate will decay based on (1), and parameters in Table 4.

Table 4. Learning rate configuration

Settings	Value
Initial learning rate (r)	0.001
Decay steps (ds)	Steps per epoch x 100
Decay rate (dr)	0.6

3.3.3. Hyperparameter model settings

There are 5 models used in this paper in which complexities vary significantly. The first model-MODEL1 is the simplest with a total parameter of 11,177 while the most complex is MODEL5 with 93,609 parameters. The researchers experimented with varying the number of hidden layers and several parameters by using the settings in Table 5 in different trial runs.

Table 5. Hyperparameters settings

#	Model name	Layers					Output shape	Total params
		1	2	3	4	5		
1	MODEL1	64					41	11,177
2	MODEL2	128	256				41	60,585
3	MODEL3	128	128	256			41	77,097
4	MODEL4	128	128	128	256		41	93,609
5	MODEL5	64	64	64	64	256	41	48,169

In this paper, the input layer is always set to use dense. The default output layer used is the softmax [36] function but the most commonly used activation function-sigmoid function [37], was also tested to identify whether softmax will perform best using the hyperparameter settings in Table 5. The activation function used in layers 2–5 was ELU [38]. ReLU [39], [40] was also used as an activation function in hidden layers to evaluate whether considerable performance improvements will be compared to ELU.

3.4. Training the models

The researchers performed various trial runs to find the best model in this paper. The learning rate in Figure 9 was used in all models. During the course of training the models described in this paper, an issue has been encountered where the results obtained are different each time. This prevented us from moving forward. The researchers were able to fix the problem by setting a fixed random seed globally. Figure 9 shows the summary of the accuracy of the model when applied to the training set and validation set. Prediction accuracy and losses on unseen data are described in more detail in the next section.

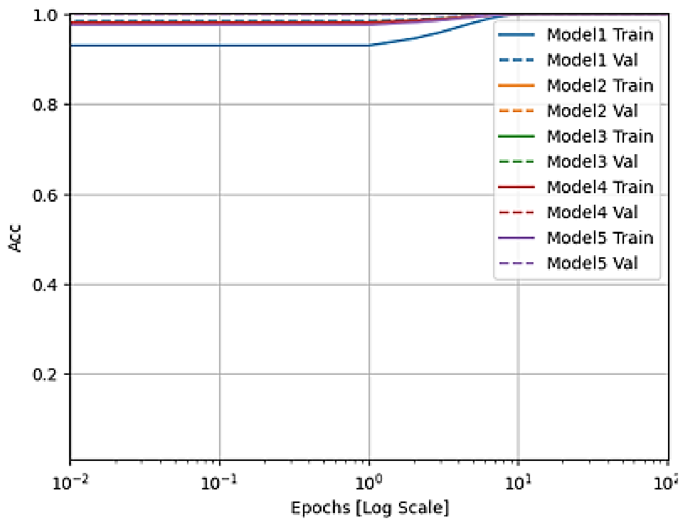


Figure 9. Validation vs training accuracy

As observed in Figure 6, the graph of all the models in terms of accuracy has converged to 1.0 which is normal. The graph tells us that the weighing values have already been optimized at epoch <100 because it has reached the maximum possible accuracy of 1.0. Figure 10 presents the final model that was trained and built in this project. For simplistic implementation in a production environment, MODEL1 is recommended. If a more robust model is to be desired, then MODEL5 is highly recommended. Its numerous numbers of hyperparameters are not to be desired, but the loss is taken into account, MODEL5 outperforms MODEL1 by a factor of 213 times.

Figure 10 shows the number of layers, and type of activation functions for both MODEL1 and MODEL2. Hidden layers use ELU [38] and output layers utilizes Softmax [36]. Note that the number of inputs corresponds to the number of possible symptoms and the output corresponds to the number of output categories which is 41. The correct output classification is determined by taking the index of the maximum output value.

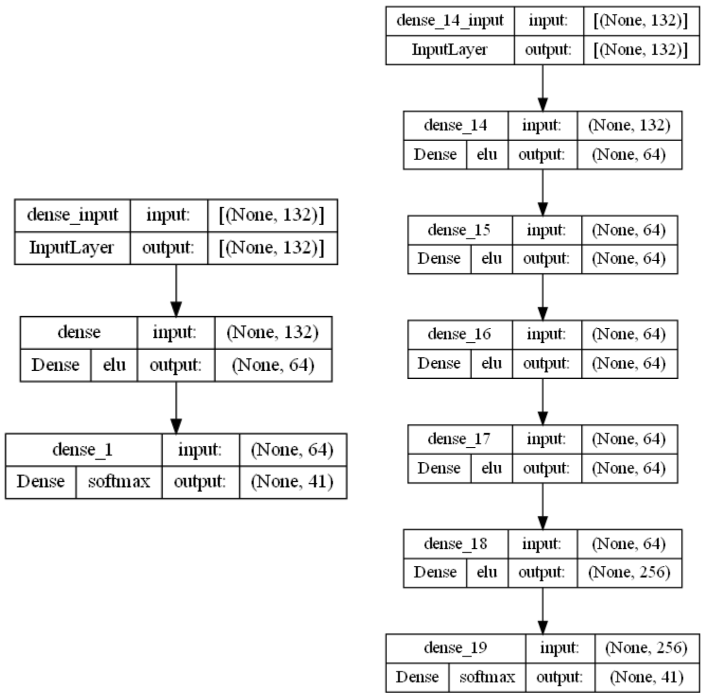

dense_input | InputLayer | input: [(None, 132)] | output: [(None, 132)]
dense | Dense | elu | input: (None, 132) | output: (None, 64)
dense_1 | Dense | softmax | input: (None, 64) | output: (None, 41)
dense_14_input | InputLayer | input: [(None, 132)] | output: [(None, 132)]
dense_14 | Dense | elu | input: (None, 132) | output: (None, 64)
dense_15 | Dense | elu | input: (None, 64) | output: (None, 64)
dense_16 | Dense | elu | input: (None, 64) | output: (None, 64)
dense_17 | Dense | elu | input: (None, 64) | output: (None, 64)
dense_18 | Dense | elu | input: (None, 64) | output: (None, 256)
dense_19 | Dense | softmax | input: (None, 256) | output: (None, 41)

Figure 10. MODEL1 vs MODEL5

3.5. Performance evaluation

Based on the evaluation of the model that was conducted, it was evident that all of the models performed well on test data with perfect accuracy. However, When the model was used to predict data that were not encountered before, MODEL1 and MODEL5 were the clear victors as shown in Table 6. Some ML practitioners normally neglect accuracy versus loss metrics. But in this paper, the researchers equally treat this metric because it provides information on the loss function used and how well the model has performed in terms of minimizing this calculated loss. Table 7 shows the performance of the 5 models in the classification of unseen data.

Table 6. The prediction accuracy of unseen data model

Data sample	Model				
	1	2	3	4	5
Test data	1.0	1.000000	1.000000	1.000000	1.0
Unseen data	1.0	0.97619	0.97619	0.97619	1.0

Table 7. Loss vs accuracy on unseen data model

	Model				
	1	2	3	4	5
Loss	0.02007	0.0335	0.08491	0.10055	0.00009
Accuracy	1.00000	0.9761	0.97619	0.97619	1.00000

3.5.1. Precision, recall and F1-score

Precision, recall, and F1-score are essential metrics for evaluating the performance of the model. Precision in (2) refers to the ability of the classifier not to label a negative sample as positive and recall in (3) is the ability of the classifier to find all the positive samples. F1-score in (4), on the other hand, is an evaluation metric that measures a model's accuracy. It combines the precision and recall score of a classification model. The accuracy metric computes how many times a model made a correct prediction across the entire dataset.

$$Precision = \frac{TP}{TP+FP} \tag{2}$$

$$Recall = \frac{TP}{TP+FN} \tag{3}$$

$$F1 - score = 2\, x \frac{Precision\ x\ Recall}{Precision+Recall} \tag{4}$$

where TP–is true positive, FP–is false positive, and FN–is false negative. Figure 11 shows the performance of the model showing precision=1.0, recall=1.0, and f-score=1.0.

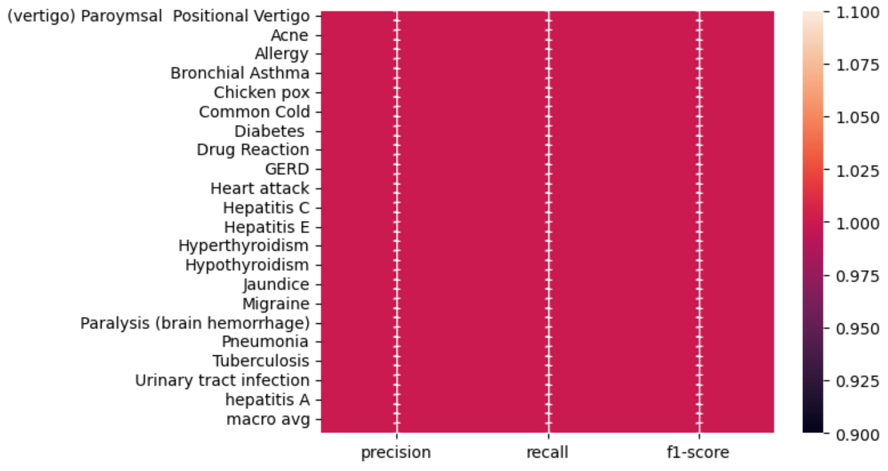


Figure 11. Precision, recall, and F1-score

3.5.2. Confusion matrix

Another method for measuring the performance of a classification model is the confusion matrix. In confusion matrix, the predicted values are mapped in matrix form against the expected values as shown in Figure 12. A confusion matrix is formed as a table that compares the actual classifications of ground truth data with the predictions made by a classification model. It provides a clear breakdown of how many instances were correctly and incorrectly classified for each output classes.

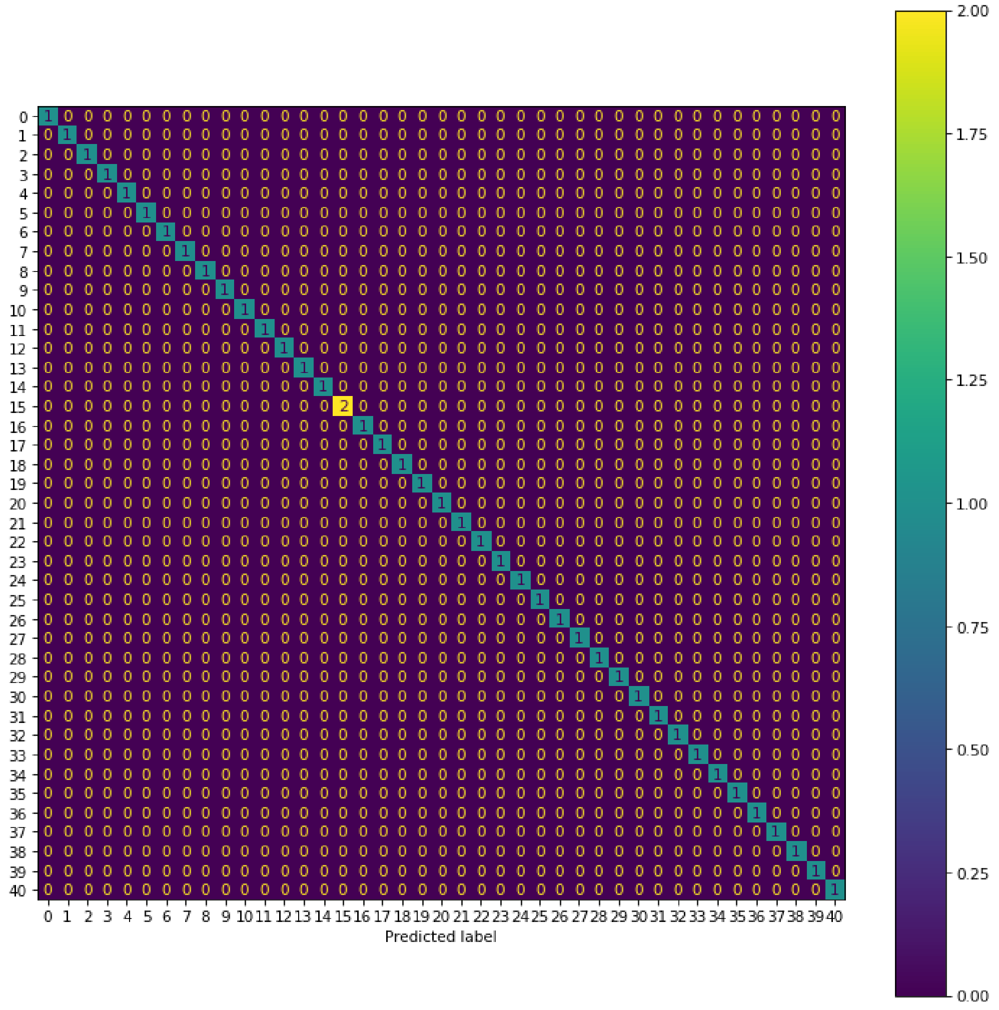


Figure 12. Confusion matrix of unseen data from MODEL5

4. CONCLUSION

Changing the activation function of the models from ELU to ReLU does not yield much difference in terms of accuracy and losses. Softmax output activation function is a very common deep learning multiclass classification, but it does not yield noticeable advantage over the classic sigmoid activation output function as well. It was also observed that the repeatability of the prediction models is impossible if the global random seed is not set to a fixed value. Comparing the different performance and accuracy of the five trained models that were developed in this paper, we came up with the conclusion that MODEL1 (simplest) and MODEL5 (more complex) provide the most accurate classification output, scoring 1.0 on precision, recall, and f-score on the unseen data. The best performing trained models that were developed in this paper will be used to build an AI-assisted patient diagnosis system in the TSCD project. It will be interesting to know how the models will perform in a more realistic scenario. In the future, we hope to build a web-based application or desktop application software that can be used by healthcare and hospitals across the Sultanate of Oman. One area that is worth pursuing to further this research is to increase the capability of the prediction model to diagnose a wider range of symptoms by adding more datasets. One of the biggest hurdles to make this possible, however, is the availability of datasets. Finally, to further improve the accuracy and generalization capability of the prediction models in this paper, a new feature must be added to indicate the degree (e.g., from 1–10) are symptoms as experienced by the patient.

ACKNOWLEDGEMENTS

The research leading to these results has received funding from the Ministry of Higher Education, Research and Innovation (MoHERI) of the Sultanate of Oman under the Block Funding Program (Block Funding Agreement No.: MoHERI/BFP/GULF/01/2021, Project Code: BFP/RGP/ICT/21/451).

REFERENCES

[1] Times News Service, "Mission to cut down on medical errors in Oman," *Times of Oman*, 2018. Accessed: May 15, 2023. [Online]. Available: https://timesofoman.com/article/149311/oman/mission-to-cut-down-on-medical-errors

[2] L. Bernstein, "20 percent of patients with serious conditions are first misdiagnosed, study says," *The Washington Post*, 2017. Accessed: Oct. 14, 2022. [Online]. Available: https://www.washingtonpost.com/national/health-science/20-percent-of-patients-with-serious-conditions-are-first-misdiagnosed-study-says/2017/04/03/e386982a-189f-11e7-9887-1a5314b56a08_story.html

[3] Docpanel, "How common is misdiagnosis-infographic," *DocPanel Technologies*, 2019. Accessed: Mar. 15, 2023. [Online]. Available: https://www.docpanel.com/blog/post/how-common-misdiagnosis-infographic

[4] H. M. Mustafa, D. S. Abdulateef, and H. S. Rahman, "Misdiagnosis of COVID-19 infection before molecular confirmation in Sulaimaniyah City, Iraq," *European Journal of Medical Research*, vol. 27, no. 1, 2022, doi: 10.1186/s40001-022-00704-0.

[5] D. E. Newman-Toker *et al.*, "Rate of diagnostic errors and serious misdiagnosis-related harms for major vascular events, infections, and cancers: Toward a national incidence estimate using the 'big Three,'" *Diagnosis*, vol. 8, no. 1, pp. 67–84, 2021, doi: 10.1515/dx-2019-0104.

[6] WHO, "Patient safety," *World Health Organization*, 2023. Accessed: Mar. 15, 2023. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/patient-safety

[7] G. Nguyen *et al.*, "Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey," *Artificial Intelligence Review*, vol. 52, no. 1, pp. 77–124, 2019, doi: 10.1007/s10462-018-09679-z.

[8] A. Paszke *et al.*, "Automatic differentiation in pytorch," *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, USA, pp. 1-4, 2017.

[9] M. Abadi *et al.*, "Tensorflow: large-scale machine learning on heterogeneous distributed systems," *arXiv-Computer Science*, pp. 1-19, 2015.

[10] J. Mellon, "MXNet: a growing deep learning framework," *Carnegie Mellon University*, 2022, [Online]. Available: http://insights.sei.cmu.edu/blog/mxnet-a-growing-deep-learning-framework/

[11] T. Chen *et al.*, "MXNet: a flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv-Computer Science*, pp. 1-6, 2015.

[12] S. J. Lawrence, "Deep learning frameworks comparison," *Scaler*, 2023. Accessed: May. 18, 2023. [Online]. Available: https://www.scaler.com/topics/tensorflow/mxnet-vs-tensorflow/

[13] M. Opala, "Deep learning frameworks comparison – tensorflow, keras, MXNet and more," *Netguru*, 2022. Accessed: Mar. 14, 2023. [Online]. Available: https://www.netguru.com/blog/deep-learning-frameworks-comparison

[14] G. Ke *et al.*, "LightGBM: a highly efficient gradient boosting decision tree," *Advances in Neural Information Processing Systems*, pp. 1-9, 2017.

[15] G. Boesch, "Pytorch vs tensorflow: a head-to-head comparisot," *Viso AI*, 2023. Accessed: Feb. 10, 2023. [Online]. Available: https://viso.ai/deep-learning/pytorch-vs-tensorflow/

[16] O. C. Agustin, "GitHub - whaldsz/deep-learning: Source codes for disease prediction deep learning," *GitHub*, 2022. Accessed: Jul. 18, 2022. [Online]. Available: https://github.com/whaldsz/deep-learning/tree/main

[17] T. Kluyver *et al.*, "Jupyter Notebooks - a publishing format for reproducible computational workflows," *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, IOS Press, pp. 87-90, 2016, doi: 10.3233/978-1-61499-649-1-87.

[18] T. Carneiro, R. V. M. D. Nobrega, T. Nepomuceno, G. B. Bian, V. H. C. D. Albuquerque, and P. P. R. Filho, "Performance analysis of google colaboratory as a tool for accelerating deep learning applications," *IEEE Access*, vol. 6, pp. 61677–61685, 2018, doi: 10.1109/ACCESS.2018.2874767.

[19] P. Patil, "Disease symptom prediction, version 2." *Kaggle*, 2020. Accessed: Nov. 29, 2023. [Online]. Available: https://www.kaggle.com/datasets/itachi9604/disease-symptom-description-dataset

[20] H. Zhang, L. Zhang, and Y. Jiang, "Overfitting and underfitting analysis for deep learning based end-to-end communication systems," *2019 11th International Conference on Wireless Communications and Signal Processing, WCSP 2019*, 2019, doi: 10.1109/WCSP.2019.8927876.

[21] O. C. Agustin and B. J. Oh, "Automatic milled rice quality analysis," *Proceedings of the 2008 2nd International Conference on Future Generation Communication and Networking, FGCN 2008*, vol. 2, 2008, doi: 10.1109/FGCN.2008.170.

[22] O. C. Agustin and B. J. Oh, "Weight estimation and classification of milled rice using support vector machines," *VISAPP 2009 - Proceedings of the 4th International Conference on Computer Vision Theory and Applications*, vol. 1, pp. 377–380, 2009, doi: 10.5220/0001755803770380.

[23] C. C. Chang and C. J. Lin, "LIBSVM: a Library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, 2011, doi: 10.1145/1961189.1961199.

[24] P. Baldi, P. Sadowski, and D. Whiteson, "Searching for exotic particles in high-energy physics with deep learning," *Nature Communications*, vol. 5, 2014, doi: 10.1038/ncomms5308.

[25] L. N. Smith and N. Topin, "Super-convergence: very fast training of neural networks using large learning rates," *Proceedings Volume 11006, Artificial Intelligence and Machine Learning for Multi-Domain Operations Applications*, 2019, doi: 10.1117/12.2520589.

[26] J. Gütter, A. Kruspe, X. X. Zhu, and J. Niebling, "Impact of training set size on the ability of deep neural networks to deal with omission noise," *Frontiers in Remote Sensing*, vol. 3, 2022, doi: 10.3389/frsen.2022.932431.

[27] F. Ridzuan and W. M. N. Wan Zainon, "A review on data cleansing methods for big data," *Procedia Computer Science*, vol. 161, pp. 731–738, 2019, doi: 10.1016/j.procs.2019.11.177.

[28] Y. Zhao, D. Wang, L. Wang, and P. Liu, "A faster algorithm for reducing the computational complexity of convolutional neural networks," *Algorithms*, vol. 11, no. 10, 2018, doi: 10.3390/a11100159.

[29] R. Caruana, S. Lawrence, and L. Giles, "Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping," *Advances in Neural Information Processing Systems*, 2001.

[30] M. Dialameh, A. Hamzeh, H. Rahmani, S. Dialameh, and H. J. Kwon, "DL-Reg: a deep learning regularization technique using linear regression," *Expert Systems with Applications*, 2020, doi: 10.1016/j.eswa.2024.123182.

[31] G. Li, Y. Gu, and J. Ding, "The efficacy of L_1 regularization in two-layer neural networks," *arXiv-Computer Science,* pp. 1-12, 2020.

[32] A. S. Saud and S. Shakya, "Analysis of L2 regularization hyper parameter for stock price prediction," *Journal of Institute of Science and Technology*, vol. 26, no. 1, pp. 83–88, 2021, doi: 10.3126/jist.v26i1.37830.

[33] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[34] Y. Zhu *et al.*, "Automatic, dynamic, and nearly optimal learning rate specification via local quadratic approximation," *Neural Networks*, vol. 141, pp. 11–29, 2021, doi: 10.1016/j.neunet.2021.03.025.

[35] A. Senior, G. Heigold, M. Ranzato, and K. Yang, "An empirical study of learning rates in deep neural networks for speech recognition," *ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6724–6728, 2013, doi: 10.1109/ICASSP.2013.6638963.

[36] S. Raghuram, A. S. Bharadwaj, S. K. Deepika, M. S. Khadabadi, and A. Jayaprakash, "Digital implementation of the softmax activation function and the inverse softmax function," *4th International Conference on Circuits, Control, Communication and Computing, I4C 2022*, 2022, doi: 10.1109/I4C57141.2022.10057747.

[37] S. Narayan, "The generalized sigmoid activation function: competitive supervised learning," *Information Sciences*, vol. 99, no. 1–2, pp. 69–82, 1997, doi: 10.1016/S0020-0255(96)00200-9.

[38] D. A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," *4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings*, 2016.

[39] A. F. Agarap, "Deep learning using rectified linear units (ReLU)," *arXiv-Computer Science*, pp. 1-7, 2019

[40] Y. Bai, "RELU-function and derived function review," *SHS Web of Conferences*, vol. 144, 2022, doi: 10.1051/shsconf/202214402006.

BIOGRAPHIES OF AUTHORS

Ronald S. Cordova received his Bachelor of Science in Computer Engineering in 1998 from Systems Plus College Foundation in Angeles City, Philippines. He earned a Master of Science in Information Technology from Hannam University in South Korea in 2003 and a Doctor of Philosophy in Information Technology from the same institution in 2009. Currently, he teaches advanced computing modules at Gulf College in academic affiliation with Cardiff Metropolitan University in the United Kingdom. Moreover, he serves as an adjunct professor at the Graduate School of the Angeles University Foundation in the Philippines. He has authored and co-authored numerous scientific publications in the area of artificial intelligence, big data analytics, cybersecurity, information systems, and the internet of things. He has also served as a guest speaker and presented numerous research papers at international conferences. In addition, he has completed and ongoing research projects from the Ministry of Higher Education, Research, and Innovation (MoHERI) of Oman, where he also functions as a reviewer for research projects. In addition, he serves as a reviewer for the International Association for Computer Information Systems (IACIS) and the Informing Science Institute (ISI) in the United States. He can be contacted at email: ronald@gulfcollege.edu.om.

Rolou Lyn R. Maata is a doctorate degree (Ph.D.) holder, book author, a Fellow of Advanced Higher Education UK, certified SAP lecturer, licensed professional teacher, research reviewer, and a computer science professor in various higher education institutions (HEIs) in the Philippines, Kingdom of Bahrain, and Sultanate of Oman. She is also a certified SAP lecturer recognized by SAP University in Munich, Germany and served as the main contact of SAP University Alliances EMEA in Bahrain. She attended SAP ERP trainings at Napier University, Scotland, UK and SAP University, Waldorf, Germany. She is an active research reviewer of The Research Council (TRC) of Oman and Institute of Informing Science in USA. She has presented and published numerous research papers in several reputable journals and peer-reviewed conference proceedings that are indexed in Scopus, IEEE, Google Scholar, and other research databases. Her research interests lie in the area of educational technologies, business and data analytics, computer security, software engineering, and information system. She was awarded twelve (12) research project grants by the Ministry of Higher Education, Research and Innovation (MOHERI) in the Sultanate of Oman, for which she serves as principal and co-principal investigators in the year 2018 to 2022. She also mentored various funded research of undergraduate students. She is the recipient of Best Researcher Award in 2021. She is currently working on two funded research projects title "Tracking system for chronic diseases using big data analytics" and "A proposed distributed and interoperable architecture model towards the implementation of blockchain technology in higher education". She can be contacted at email: rolou@gulfcollege.edu.om.

Malik Jawarneh is an accomplished Associate Professor of Computer Science at Oman College of Management and Technology. He earned his Ph.D. from The National University of Malaysia in 2016, specializing in Visual Informatics. His diverse research interests encompass Virtual Reality & Augmented Reality, IoT, artificial intelligence, big data, and computer security. He has contributed significantly to academia through co-authored research papers, patents, and active involvement as a reviewer for The Research Council of Oman. His notable projects include a trust-based model for smart city adoption and a big data analytics-driven tracking system for chronic diseases. Recognized for his exceptional contributions. He received the outstanding reviewer award from the Institute of Industry and Academic Research Incorporated journal and several awards for his publications. His recent best research award underscores his dedication to advancing knowledge and innovation, solidifying his stature as a distinguished researcher in computer science with a lasting impact on the academic community and beyond. He can be contacted at email: mjawarneh@omancollege.edu.om.

Marwan I. Alshar'e is a distinguished academic and researcher specializing in the domains of computer science and information technology. His extensive educational background, coupled with rich teaching and research experience, underscores his significant contributions to the field, exemplifying expertise, leadership, and dedication. Currently serving as an Assistant Professor at Sohar University in the Sultanate of Oman. He earned his Ph.D. from The National University of Malaysia (UKM), delving into vulnerabilities within trusted computing environments and devising a robust user authentication model to fortify such contexts. His pioneering research underscores his dedication to trailblazing solutions for emergent cybersecurity challenges. His instructional prowess spans diverse major disciplines, encompassing cybersecurity, computer science, software engineering, and IT. His adept design and delivery of modules across postgraduate and undergraduate programs cover indispensable subjects in these arenas. Alongside teaching, he has adeptly guided numerous Master's and Ph.D. students, providing unwavering support throughout their research endeavors. His expertise has also been sought for roles as both external and internal examiner for master's and Ph.D. theses. His engagement extends to impactful research collaborations, co-leading and overseeing projects encompassing information security, blockchain technology, big data analytics, usability evaluation of educational games, online human activity recognition, and data storage encryption. He can be contacted at email: mshare@su.edu.om.

Oliver C. Agustin was born in Alfonso Castañeda, Nueva Vizcaya, Philippines in 1974. He received the B.S. degree in electronics and communications engineering from Wesleyan University–Philippines in 1995, M.S. and Ph.D. degrees in information technology from Hannam University, Daejeon, South Korea in 2005 and 2009, respectively. From 1997 to 2000, he was a faculty member at Wesleyan University–Philippines. He has been an assistant professor at Angeles University Foundation, Angeles City. He has been a reviewer at Journal of Information Technology Organization Editorial Review Board. His research interests include electronics, computer vision, image processing, neural network, enterprise software integration, and cloud computing. He holds one patent. He was a member of Institute of Electronics and Communication Engineers of the Philippines since 1997. He can be contacted at email: oliver.agustin@veraequinox.com.