International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Deep Learning and Machine Learning Approach to Breast Cancer Classification with Random Search Hyperparameter Tuning

Inutu Kawina*[1], Dr. Amarendra K. [2], Bhaskar Marapelli [3]

Submitted: 09/12/2023 **Revised**: 16/01/2024 **Accepted**: 30/01/2024

Abstract: Breast cancer is one of the main causes of death and a threat to women, artificial intelligence has grown in importance in the field of health over time. With the aid of random search hyperparameter tuning, this study proposed a machine learning and deep learning approach to breast cancer classification. Two datasets related to breast cancer were used in this study, and findings proved that optimizing random search hyperparameters enhances the models performance, in addition it is seen that machine learning classifiers are not as effective in classifying breast cancer as compared to deep learning. Among the deep learning approaches Convolutional neural networks showed the highest accuracy over deep neural networks on both datasets. It was further observed that random search hyperparameter tuning performed better on the Breakhis_400x dataset than on the breast histopathology dataset which could be attributed to the notion that hyperparameter tuning using random search performs better on small data than large data.

Keywords: *Breast cancer, Convolution neural network, Deep learning, Deep neural network, Machine learning, Random search hyperparameter tuning,*

1. Introduction

Breast cancer is considered a threat and one of the deadliest diseases to women when left untreated. Owing to the nature of this disease various organizations have united to come up with interventions on how to reduce cancer related cases and one of the interventions includes incorporating technology in the diagnosis and treatment. Reference [1] estimated 1 958 310 new instances of cancer and 609, 820 cancer losses were expected in the United States with 2.3 million women being diagnosed with breast cancer with 685 000 deaths worldwide [2]. With the high rates of cancer related cases, there is need for early diagnosis and treatments to reduce death rates.

Breast cancer is characterized by the unusual growth in the breast tissue with the potential to spread to other areas of the body when left untreated [3].

Artificial intelligence has revolutionized the way cancer is being handled. Artificial Intelligence entails teaching a machine to think, act, and learn, it is highly useful for analysing image data and excels at identifying patterns in massive volumes of data [4]. Therefore, in breast cancer treatment AI is being used to enhance breast cancer diagnosis, treatments, drug development etc. and with technology at its peak a lot of funds are going towards systems that could help identify cancer and address it in its early stages to reduce death rates. Machine learning and deep learning approaches are being used in the detection and classification of breast cancer. With the aid of random search hyperparameter this study aims to classify breast cancer using deep leaning and machine learning.

Objectives

The purpose of the study is:

Main Objective

To create a breast cancer classification system using deep learning and machine learning with random search hyper parameter tuning

Specific objectives

• To determine the ideal model parameters by using random search hyperparameter tuning.

• To determine which model is the best for classifying breast cancer.

2. Problem Statement

Due to the extreme severity of breast cancer, it must be treated and managed quickly. Artificial intelligence is one of the technologies that has been identified as a driving force in the diagnosis and treatment of this disease. It has been utilized in the identification and classification of breast cancer to treat it as early as possible, which has contributed to a decrease in the number of cancer-related deaths.

[1] *Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522502, Andhra Pradesh, India. Email ID: kawinainutu@gmail.com*
ORCID ID: https://orcid.org/0009-0001-5571-9026
[2]*Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522502, Andhra Pradesh, India. Email ID: amarendra@kluniversity.in*
ORCID ID: https://orcid.org/0000-0003-3597-0878
[3]*Associate Professor, Department of Computer Science and Information Technology, Koneru Lakshmaiah Education Foundation, Vaddeswaram 522502, Andhra Pradesh, India. Email ID: bhaskarmarapelli@gmail.com*
ORCID ID: https://orcid.org/0000-0002-0101-9083

As the use of AI is growing in cancer treatment so is the need to improve performance. Improving performance of machine learning and deep learning models can be a tedious process which entails using different parameters that that would need to be applied to them, and this could take time in finding the best parameters, thus, the need for introducing a random search parameter reduces the time taken in finding the parameters that could be applied to the model. Therefore, this study focuses on classifying breast cancer images using deep learning and machine learning classifiers with the aid of random search hyperparameter tuning to enhance the performance of the models.

3. Literature Review

This chapter presents several studies related to the study that have been conducted.

Cancer has always been a disease of concern around the globe with more interventions being put in place to fight cancer [5]. Numerous studies are being conducted to determine the most effective ways to combat it, and one approach is the development of technology that can reliably identify and classify it; artificial intelligence and machine learning are excellent resources for creating robust systems that yield accurate results [6].

As study by [7] was based on image classification that utilized a randomized search cv approach for hyper parameter tuning, this was due to the nature of the hyperparameter tuning where unlike Grid search it only samples out some random combinations instead of getting all the possible combinations, this can get the optimal settings that could enhance the model's performance while saving time and space. The study involved gathering a parameter distribution, doing a k-fold cross validation, and ultimately obtaining the optimal parameters for model training.

In this study, [8] optimized a grid search-based K-Nearest Neighbor (KNN) based breast cancer detection model to find the optimal hyper-parameter. The results demonstrated that adjusting the hyperparameters had a major effect on the KNN model's performance. The findings of comparing the KNN's performance with and without its tuned hyperparameter revealed that the model that was suggested performed 94.35% of the time with hyperparameter tuning and 90.10% of the time without it.

Reference [9] conducted a study with the purpose of optimizing the ResNet and Xception in diagnosing COVID-19, to obtain the best possible architecture for use in COVID-19 diagnosis, the parameters of both suggested models were tuned using randomized search. The outcomes demonstrated that ResNet performed better while employing Random search for diagnosis.

Another study [10] used a Densely Connected Convolutional Networks (DenseNet), which linked each layer and feature maps to all subsequent layers. The optimal value to be applied was determined by using the random search hyperparameter tuning method. Before performing an experiment on the selected candidates, it made use of the batch size and learning rate. The findings demonstrated that 64 was the lowest batch size that could be employed, and that a hyperparameter tuning strategy yielded an accuracy of 95% when the ideal learning rate was between 0.1 and 0.3.

Refence [11] suggested a deep neural network (deepCNN) to increase the precision of breast cancer classification. The learning rate, epoch count, and dropout rate of the DeepCNN model were enhanced using random search-based hyper-parameter optimization. A comparison was made between the outcomes and a few pre-trained models, including Xception, Resnet50, Resnet101V2, InceptionResNetV2, and VGG19. With the Random Search optimizer, the unique custom DeepCNN model achieved the highest accuracy of 99.18% out of all the models.

Another study utilized deep learning to effectively classify breast cancer as benign or malignant. The suggested method made use of picture pre-processing methods like standardization and normalization and employed convolutional neural networks to classify the images. Grid search and randomized search were used in the hyper-parameter tuning of convolutional neural networks. The INbreast dataset was used in the experiment and the results indicated that the most efficient method for using CNN to categorize mammography images for breast cancer classification was to use hyper-parameter tuning and optimization approaches in conjunction with normalization and standardization procedures [12].

Another study utilized a deep convolutional neural network (DCCN) with transfer learning, it used the mammogram image samples for breast cancer diagnosis. The contrast limited adaptive histogram equalization technique was applied to the region of interest to boost the contrast of the image tumor. The VGG-16 and VGG-19 network models were utilized with reduced convolution layers and max pooling layers providing more feature datasets. The results when compared indicated that the VGG-16 performed well with an accuracy of 82.5% when compared to VGG-19 [13].

4. Methodology

This section of the study presents the methods applied to analyze the dataset on breast cancer. It provides visual representation, methods and procedures followed in the analysis.

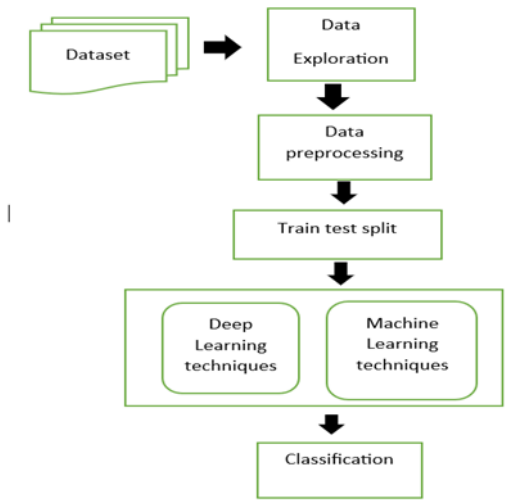

Fig. 1 Workflow of the methodology

4.1. Data collection

The study used two secondary datasets that are freely available on the Kaggle, the Breast Histopathology image dataset and the BreakHis_400X dataset.

4.2. Dataset description

The breast histopathology image dataset has 277,524 50 × 50 image patches that were extracted from 162 complete mount slide images of breast cancer samples (198,738 IDC negative and 78,786 IDC positive). The file name "u_xX_yY_classC.png" is assigned to the data, where 'u' is the patient ID (10253_idx5), X and Y are the coordinates from whence the patch was gathered, and C is the session, with 0 being IDC negative (-) and 1 representing IDC positive (+).

The BreakHis_400X image dataset is a breast cancer dataset with 1693 images of benign and malignant breast cancer. The dataset contains 547 images of benign and 1146 images of malignant. Benign breast cancer is considered non-cancerous as it is known not to invade other nearby tissues while the malignant are considered cancerous due to their nature of spreading to other parts.

4.3. Data exploration and preprocessing

Jupiter Notebook version 6.5.2 with python version 3.11 was used as an analysis tool to analyze the breast cancer datasets. Datasets preparation involved extracting the datasets from the various sources (Kaggle) and to begin working on the datasets involved importing various packages that were necessary for the preprocessing, analysis, training and visualizing of the image dataset.

4.3.1. Breast Histopathology dataset

The image dataset was imported and visualized to identify the cancer classes and it was discovered that the image dataset had patient's directory where each patient ID contained two classes which are 0 for IDC (-) and 1 for IDC (+). To have two classes instead of having multiple patient IDs and each with the classes all the images that had had class0.png were placed in class 0 and all the images that had class1.png were placed in class 1 directory to have a dataset that contained two classes.

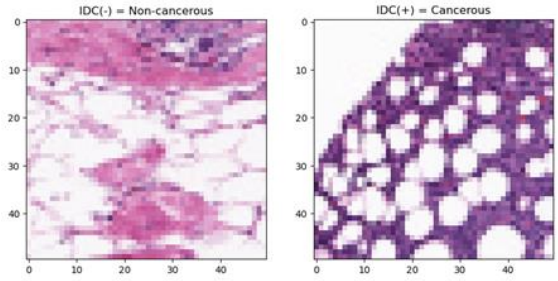

Fig. 2 Breast histopathology class images

Data exploration was done on the breast histopathology dataset, and it was discovered that the image dataset was highly unbalanced, with IDC (-) having 198 738 images and IDC (+) having 78 786 images.

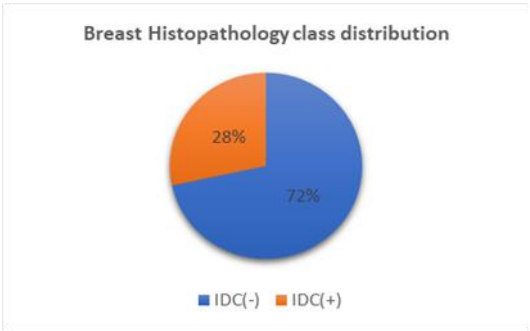

Fig. 3 Unbalanced breast histopathology dataset

In order to keep the model from being skewed towards one class, balancing was necessary. To balance the histopathology dataset, random under sampling was performed on both classes of which 50 000 images were randomly chosen from both IDC (-) and IDC (+).

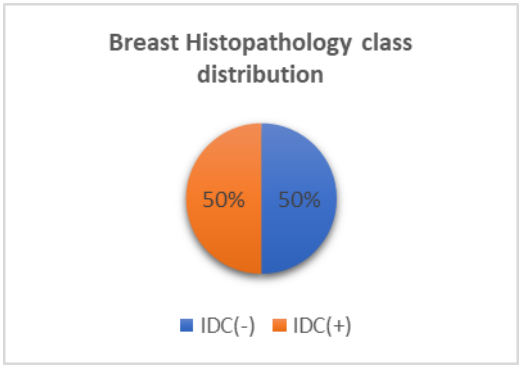

Fig. 4 Balanced breast histopathology dataset

4.3.2. BreakHis_400X dataset

Analysis of the breakHis_400X dataset began by importing packages that were needed, data exploration was performed on this dataset which contained two classes benign and

malignant.

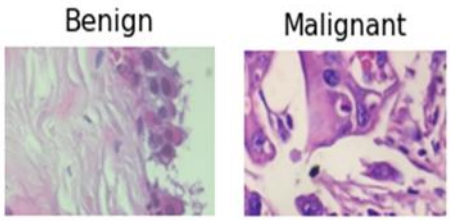

Fig. 5 BreakHis_400x dataset class images

Further data exploration of the breakHis_400x breast cancer dataset found the dataset to be unbalanced with Benign class having 547 images and malignant class having 1146 images. Therefore, the need to perform balancing on the dataset.

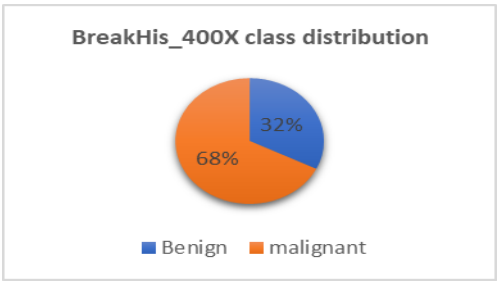

Fig. 6 Unbalanced BreakHis_400X dataset

To balance the BreakHis_400X breast cancer dataset synthetic minority over sampling technique (SMOTE) was used. Unlike the breast histopathology dataset, the breakhis_400X dataset is a small dataset comprising of 1693 images in total, performing random under sampling would reduce the number of images to be trained thereby affecting the performance of the model. Random oversampling and SMOTE are the two sampling strategies that could be applied to enhance the number of images in the minority class. The SMOTE was preferred in this study due to its ability to produce synthetic samples for the smaller class, thereby focusing the issue of class imbalance in datasets more effectively [14] than random oversampling which duplicates existing samples and increases the samples of the minority class, the problem with random oversampling is that it can cause overfitting when the model is being trained [15].

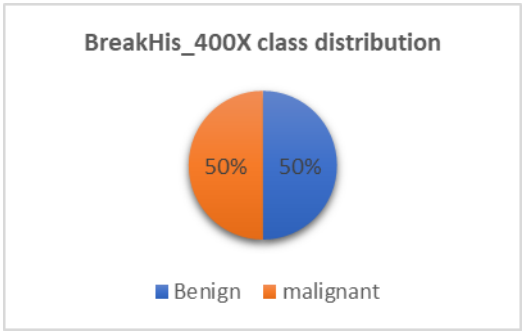

Fig. 7 Balanced BreakHis_400X dataset

Preprocessing of both the breast cancer datasets was the same and involved resizing of the images to 50 X 50 height and width and an interpolation of cv2 interlinear was used, Further preprocessing was conducted which involved the conversion of the images to NumPy array where the images where being converted into feature (X) and labels (y), then the features were normalized by dividing the feature (X) by 255.0, this was important as images pixels values are rescaled to the range 0 and 1 to make it compatible with the expected input for the models.

The next step included splitting the datasets. The datasets were divided into three categories: test, validation, and training. Of these, 70% were used for training, 15% were used for validation to assess the model's performance, and 15% were used for testing. Random search hyper parameter tuning used to search for the best possible combinations in model. It took a grid of parameters from the model and performed a random search by iterating over the parameters passed, then the best possible parameters that were identified were returned.

4.4. Machine learning classifiers

Machine learning is a field of artificial intelligence which makes predictions based on data of particular interest that has been fed into the machine learning model [16].

4.4.1 K-Nearest Neighbor (KNN)

The KNN is a machine learning classifier that is often used for classification tasks [17]. It is widely used to categorize unknown data points by locating the most common class among the k closest points in the training data and predicting the value of a new data point using their class [18]. The KNN model parameters that were passed were the n_neighbor, which specifies the number of neighbors required to classify and make predictions and the weights, which guarantees that the weights are spread equally among the neighbors. By default, the weights is set to uniform, which ensures that the weights are spread evenly among the neighbor's values. Another value for weights is distance, which guarantees that the weights are distributed according to the neighbors' distances from one another.

When training the KNN on the dataset without the random search hyperparameter tuning, the defaults were used where the n_neighbors took the value of 5 and the weights was set to uniform.

When it came to using random search hyper parameter tuning the n_neighbors explored included 3,5,7,9 to find the optimal n_neighbor that would be the best fit for the model and the weights parameter explored included uniform and distance. When defining the parameters for the search CV was done, the Parameters were set for the randomized Search CV which included the KNN model being passed with the random Search Parameters that were defined and the cv of 3. The cross validation of three denoted the number

of folds for cross-validation; a CV of three meant that the model would be assessed three times, with two portions utilized for training and one part for validation. The KNN model is then trained using the optimal parameters.

4.4.2. Gaussian Naive Bayes

Gaussian Naive Bayes is considered a probabilistic technique and Gaussian distribution-based machine learning classifier. It assumes that every parameter can independently predict the result [19]. The var smoothing is a parameter that was used by the gaussian naïve bayes in the breast cancer classification, this parameter is utilized to find the ideal value for stabilizing the calculation, it achieves this by artificially boasting the variance. By this it accounts for more samples that are far from the distribution mean and smoothens the curve [20].

When training the gaussian naïve bayes classifier on the data without random search hyperparameter tuning, the default var smoothing which is the 1e-09 was used. But when training using the random search hyperparameter tuning the var smoothing parameter was explored by creating an array of numbers that begin at 0 and end at -9.

4.4.3. Logistic regression

Logistic regression is a machine learning classifier that is used to assess the probability that an instance will belong to a particular class [21]. The parameters that were considered for Logistic regression included penalty, C, and the solver. The penalty is the parameter that is used to specify the type of regularization that is to be applied to the logistic regression model to prevent overfitting. The penalty takes up two values which are L1 which is used to add the sum of absolute values of the coefficients and L2 value which adds the sum of squared coefficients [22]. The solver was also used, it takes various values such as 'liblinear', 'saga', 'lbfgs' among others. and the other parameter considered is the C which was used to control the strength of regularization in the model. It is crucial to choose the C properly as the value can affect the strength of the regularization, A small value of C would indicate a strong regularization while the bigger value indicates a weak regularization [23].

Training the logistic regression without random search hyper parameter tuning the default values of the penalty, which is L2, for the solver is 'lbfg' and C is 1.0. When training using the random search the penalty that was explored was the L1 and L2, a solver of 'liblinear' and 'saga', these two values for solver were selected because the penalty of L1 and L2 can only support the 'liblinear' and 'saga'. The values for the C parameter that were explored in the random search included 0.1,1.0 and 10.0.

4.5 Deep learning

Deep learning is a branch of machine learning that is used to imitate the human brain by making use of neural networks to solve difficult tasks [24].

There are various types of deep learning such as Artificial neural network, Convolutional neural networks (CNN), Deep neural Network (DNN), Recurrent neural networks (RNN) among others. This study made use of only two deep learning techniques which include the Convolution neural network (CNN) and the Deep neural network (DNN).

4.5.1. Convolutional neural network (CNN)

Convolutional neural networks are becoming more and more prominent in fields like computer vision and facial recognition due to their ability to imitate the human brain [25].

The CNN model took the first convolutional layer with a filter which was applied to the kernel size to perform feature extraction on the input image and producing feature maps, a rectified function was then added to the first convolution layer which was used for removing any linearity that may have been create when the input image was being pre-processed. A max pooling was added the output from the first convolution layer to reduce the spatial dimension of the feature by getting the maximum value from the first convolution layer feature maps to have a reduced feature map, this ensure that the neural network learns relevant features in the image. A dropout was then added randomly dropping some units to prevent overfitting. The second convolutional layer took the output from the dropout as input and with a filter being applied to the kernel size to perform feature extraction in the input and produce feature maps and a rectified function was applied. A max pooling was then performed on the output from the second convolution layer and a dropout was then performed to the output from the max pooling to prevent overfitting. The output from the dropout was then transformed into a one-dimensional vector using the flatten () method. This one-dimensional vector was then sent to the fully connected layer, which received the neuron units and the rectified activation function. The output was then passed to the output layer which took one unit and sigmoid function for binary classification.

Following model definition, the model was compiled with the intention of ensuring that it learns. The compile method required three parameters: metrics for assessing the model's performance, the loss function for minimizing loss, and an optimizer for determining how the model's weights were updated during training.

When executing the fit method on the training data some parameters were added which included the Batch size and Epochs. The batch size specifies the number of samples to be spread throughout the network, it regulates how many training samples must be processed before the internal parameters of the model are changed [26]-[27]. Epochs on

the other hand, indicate how many times all the training data are iterated in a single cycle. The number of epochs can have an impact on the model's performance because too few epochs can result in underfitting and too many epochs can cause overfitting [28].

When training the CNN model without random search hyperparameter tuning the first convolutional layer took filters of 32 which was then applied to the kernel size of (3,3) to extract important features from an input image with input shape of height of 50 pixels, width of 50 pixels and a color channel of 3 (RGB), a rectified function was then added in the first convolution layer. A Maxpooling2D () with a pool size of 2 and a stride of 2 for spatial dimension and a dropout of 0.5 was then added to prevent overfitting. The second convolution layer (Conv2D) was added, and it took filters of 32 and then it was applied to the kernel size of (3,3) to create a feature map from the input it took from the dropout, then a Maxpooling2D () was added after the second Conv2D which took a stride of 2 and a pool size of 2. Following max pooling, a dropout of 0.5 was added, and the output was then passed to the flatten () method, which turned the output into a one-dimensional vector. The output was then passed to the fully connected layer, which used 128 units of neurons and a rectified activation function, and lastly to the output layer, which contained one neuron and a sigmoid activation function for binary classification. Then, the compilation method was raised, using binary cross entropy as the loss function along with accuracy metrics and the Adam optimizer. For training, a batch size of 32 and 20 epochs were used.

When training with random search hyperparameter tuning on the proposed model, the convolution neural network was wrapped in a cnn_model function, and within this function that's where the CNN architecture was being placed. The only difference from the CNN architecture which was used before random search hyperparameter was that the filters parameter in the first and second convolution layer was being added in the random search to search for the best filter that would be good fit for the model and neurons for the fully connected layer. When compiling the model, the optimizer was also added for the random search. To define the random search the Parameters with the Filters of 32,64 and 128, neurons of 64 and 128, Optimizer of 'Adam' and 'RMSProp', epochs values of 10,20,30 and 40 and a batch size of values 32 and 64. The defined parameters were then passed to the random search together with the estimator and a cross validation of 3.

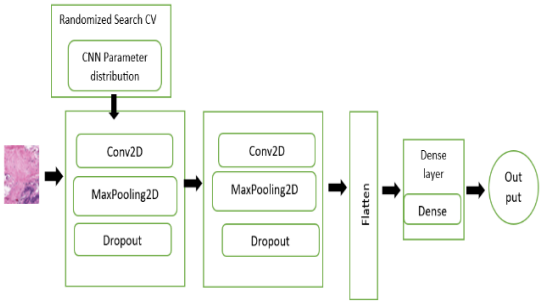

Fig. 8 Proposed Convolution Neural Network (CNN) Architecture

4.5.2. Deep neural network (DNN)

Another kind of deep learning approach is the deep neural network, which has multiple hidden layers and is founded on the artificial neural network (ANN) [29].

Deep neural network (DNN) model involved two dense layers and an output layer. The first dense layer took the neurons parameter that allowed the model to find important features in the input image, a rectified activation function was then added. And to prevent overfitting a dropout was then added. A second dense layer took a neuron as a parameter and a rectified activation function. A dropout was then performed on the output from the second layer, the output from the second dense layer was then passed to the flatten method which converted the output into a one-dimensional vector. The output from the flatten method was then passed to the output layer with a one neuron and a sigmoid activation function for binary classification.

After the model was defined, the compile method was called. It used the metrics parameter to assess the model's performance, the optimizer to decide how to update the model's weight, and the loss function to minimize the loss. When the fit method was applied to the training data, the batch size and epochs were passed.

Training the DNN model without using the random search hyperparameter tuning the first dense layer took the 64 neurons for learning and representing 64 different features, an input shape with height width and color channels of 3 (RGB) (50,50,3) and a rectified activation function for non-linearity. A Dropout of 0.5 was added after the first dense layer to prevent overfitting, a second dense layer took 64 neurons, and a rectified activation function, a dropout of 0.5 was added after the second dense layer as well and the output from the dropout was passed to the flatten method which converted the data to a one-dimensional vector and finally the output data from the flatten was passed to the output layer had had a one neuron and a sigmoid activation function for binary classification.

The random search hyperparameter tuning began by defining a function and wrapping the DNN model into the function. To define the random search the parameters such

as neurons with values 32,64 and 128 were passed, Optimizer of 'Adam' and 'RMSProp', epochs with values of 10,20,30 and 40 and a batch size of values 32 and 64. The defined parameters were then passed to the random search together with the estimator and a cross validation of 3.

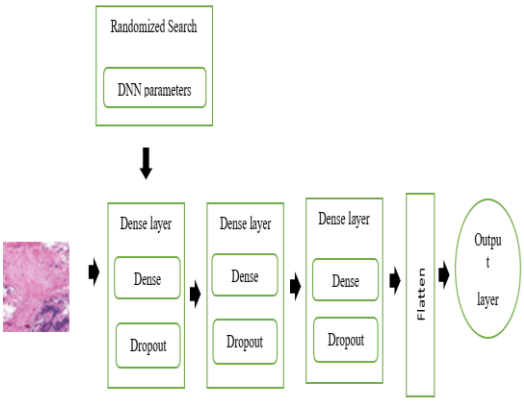

Fig. 9 Deep neural network architecture

4.6. Performance Metrics

The performance metrics are utilized to assess the model's performance. The accuracy, precision, recall, f1_score and Area under the curve (AUC) are among the metrics used to assess the model's performance.

Accuracy

Accuracy measures the proportion of correctly categorized images. It considers the number of accurate predictions to the overall number of predictions.

Accuracy = True Positive + True Negative/True positive + True Negative + False Positive + False Negative

Precision

Precision is a measure of performance that is utilized to determine the percentage of positive predictions. It entails how many of the classified items that were classified as positive predictions are actually significant.

Precision = True Positive/True Positive + False Positive

Recall

While recall calculates the percentage true positives that are actually correct, it calculates the proportion of the true positive to true positive and false negative. It aims to demonstrate the number of correct predictions were actually right.

Recall = True Positive/True Positive + False Negative

f1_Score

f1_score shows how well the model is performing, it takes into account precision and recall and combines them into one.

*f1_score = 2*precision*recall/precision+recall*

Area Under the curve (AUC)

The AUC is another crucial performance indicator that illustrates how well the model distinguishes between positive and negative cases.

5. Presentation of findings

This chapter presents the findings that were discovered in this study.

Table 1: Results after performing random search hyperparameter tuning**.**

Model Type	classifier	Parameter Identified	
		BreakHis_400X dataset	**Breast Histopathology dataset**
Machine learning	K-nearest neighbor	N_Neighbors: 3, weights: distance	N_Neighbors:9, Weights: distance
	Gaussian naïve bayes	Var_smoothing: 8.111308307896872e_05	Var_smoothing: 2.848035868435799e-08
	Logistic regression	Solver: liblinear, C: 10.0 Penalty: l1	Solver: saga, C: 1.0, Penalty: l1
Deep learning	Deep neural network (DNN)	Neuron: 128 Optimizer: 'RMSprop' Epochs: 40, Batch_size: 32,	Neurons: 128, Optimizer: 'Adam' Epochs: 20, Batch_size: 32
	Convolution neural network (CNN)	Filters: 128, neurons: 64 Optimizer: 'Adam', epochs: 40, Batch_size: 32	Filters: 64 neurons: 128 Optimizer: 'Adam', Epochs: 10, Batch_size: 32

Table 2: Breakhis_400X Dataset Machine learning performance before and after Random search

Classifier	Precision	Recall	f1_score	Accuracy	AUC
Before random Search Hyperparameter tuning					
K-Nearest Neighbour (KNN)	0.69	0.61	0.65	0.69	0.74
Gaussian Naïve Bayes	0.7	0.13	0.22	0.57	0.79
Logistic Regression	0.66	0.95	0.77	0.75	0.9
After Random search Hyperparameter tuning					
K-Nearest Neighbour (KNN)	0.68	0.68	0.68	0.7	0.73
Gaussian Naïve Bayes	0.75	0.68	0.71	0.74	0.79
Logistic Regression	0.64	0.96	0.77	0.73	0.9

Table 2 displays machine learning on the BreakHis_400X dataset both before and after the hyperparameter tuning of the random search. Prior to random search hyperparameter tuning, the KNN's results indicate 69% precision and 61% recall. With an f1_score of 65%, accuracy of 69%, and an AUC of 74%, the Gaussian naïve bayes model's results was poor before the random search hyperparameter but it had a good precision of 70% and an AUC of 79%, although the recall, f1_score, were below average. The logistic regression shows a good performance in terms of recall 95% and AUC 90% with a slightly lower precision, f1_score and accuracy. Following random search for KNN, the results display 84% f1_score, 85% accuracy, and 79% AUC. With an AUC of 86%, the CNN is more capable of differentiating between classes than the DNN, which has an AUC of 79%.

5.1. BreakHis_400X dataset Presentation of findings

68% precision, 68% recall, 68% f1_score, 70% accuracy, and 73% AUC. The Gaussian naïve bayes showed an improvement in performance with an accuracy of 74%, the precision of 75%, recall of 68%, an f1_score of 71%, and an AUC of 79%. The Logistic regression classifier shows a precision of 64%, recall of 96%, f1_score of 77%, an accuracy of 73% and AUC of 90%. Overall, Table 2.0 results demonstrate an improvement in the performance of the classifiers after random search hyperparameter tuning.

Table 3: Breakhis_400X Dataset Deep learning before and after Random Search hyperparameter tuning.

Classifier	Precision	Recall	f1_score	Accuracy	AUC
Before random Search Hyperparameter tuning					
Convolutional neural network (CNN)	0.82	0.84	0.83	0.84	0.82
Deep neural network (DNN)	0.9	0.65	0.75	0.8	0.79
After Random search Hyperparameter tuning					
Convolutional neural network (CNN)	0.84	0.91	0.87	0.88	0.86
Deep neural network (DNN)	0.8	0.88	0.84	0.85	0.79

Table 3 displays the deep learning models' performance both before and after random search hyperparameter tuning. The findings indicate that, overall, the deep learning model's performance showed that, prior to performing random search, the CNN model outperformed the DNN model with a precision of 82%, recall of 84%, f1_score of 83%, and accuracy of 8.84%, while the DNN model had a higher precision of 90%, recall of 65%, f1_score of 75%, and a lower accuracy of 80%. Upon conducting a random search, the findings show that the convolutional neural network achieved 84% precision, 91% recall, 87% f1_score, 88% accuracy, and 86% AUC. In contrast, the deep neural network (DNN) demonstrated 80% precision, 88% recall,

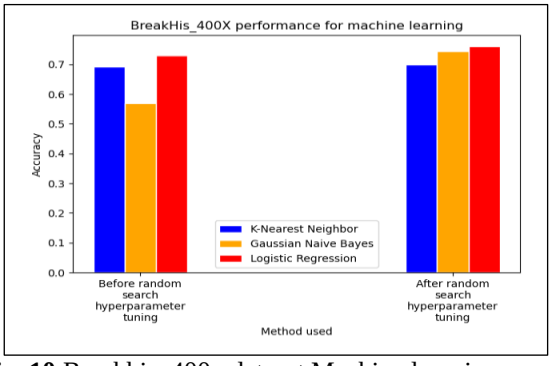

Fig. 10 Breakhis_400x dataset Machine learning accuracy performance

Fig 10 shows the accuracy performance of the breakhis_400X dataset, before and after hyper parameter tuning using random search. The figure shows that before random search was performed logistic regression of 75% accuracy while after hyperparameter tuning using random search it had 73% accuracy KNN followed Logistic regression with the 70% accuracy before and after hyperparameter tuning. Gaussian naïve bayes recorded the lowest before hyperparameter tuning with 57% accuracy and highest after hyperparameter tuning random search with 74%.

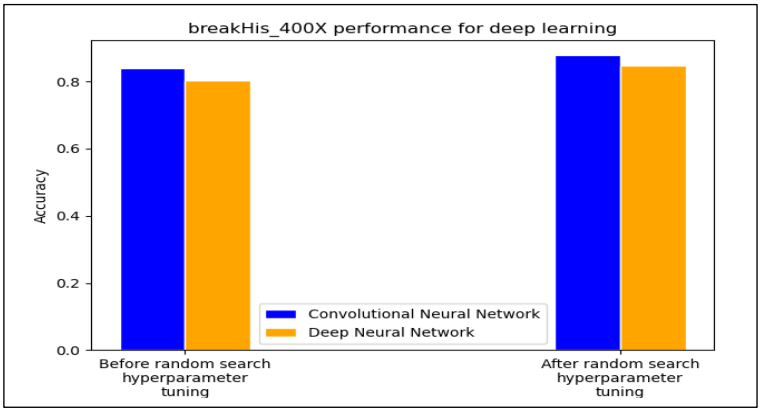

Fig. 11 BreakHis_400x Deep learning performance

Table 4: Breast Histopathology dataset Machine learning performance before and after Random search

Classifier	Precision	Recall	f1_score	Accuracy	AUC
Before random Search Hyperparameter tuning					
K-Nearest Neighbour (KNN)	0.74	0.82	0.78	0.77	0.83
Gaussian Naïve Bayes	0.74	0.74	0.74	0.74	0.77
Logistic Regression	0.63	0.78	0.77	0.73	0.83
After Random search Hyperparameter tuning					
K-Nearest Neighbour (KNN)	0.75	0.83	0.79	0.78	0.84
Gaussian Naïve Bayes	0.74	0.76	0.75	0.75	0.77
Logistic Regression	0.78	0.72	0.75	0.76	0.84

Table 5: Breast Histopathology dataset Deep learning performance before and after Random search

Classifier	Precision	Recall	f1_score	Accuracy	AUC
Before random Search Hyperparameter tuning					
Convolutional neural network (CNN)	0.68	0.66	0.67	0.89	0.82
Deep neural network (DNN)	0.58	0.88	0.71	0.64	0.74
After Random search Hyperparameter tuning					
Convolutional neural network (CNN)	0.86	0.92	0.84	0.84	0.88
Deep neural network (DNN)	0.73	0.72	0.73	0.8	0.79

5.2. Breast Histopathology dataset Presentation of findings

Table 4 shows the machine learning classifiers' performance on the breast Histopathology dataset both before and after random search hyper parameter tuning. It indicates that prior to random search hyperparameter tuning, the KNN had a high accuracy of 77% with a precision of 74%, recall of 0.82%, a f1_score of 78%, and an AUC of 83%. Prior to hyperparameter tuning, the least accurate logistic regression had a precision of 63%, a high recall of 78%, a f1_score of 77%, with an accuracy of 73%, and an AUC of 83%.

Figure 11 shows the accuracy performance of the breakhis_400x dataset in deep learning before and after hyperparameter tuning, this shows an improvement in the performance of the deep learning models when random search was performed with the CNN having the highest performance with 88% as compared to before hyperparameter tuning which had 84%. DNN also recorded an improvement in the performance with 85% as compared to before hyperparameter tuning 80%.

With an accuracy, precision, recall, and f1_score of 74%, the gaussian naïve bayes algorithm came in second from the k-nearest neighbor in terms of performance, but although it had an AUC of 77%. Furthermore, after the random search hyperparameter tuning, all of the machine learning classifiers performed better. Prior to the random search

hyperparameter tuning, the KNN outperformed the other machine learning classifiers overall.

Table 5 shows results of Breast Histopathology dataset Deep learning performance before and after Random search. The findings show that the CNN model performed well, with an accuracy of 89%, prior to hyperparameter tuning using random search, with low precision of 68%, recall of 66%, f1_score of 67%, and an AUC of 82%. In contrast, the DNN performed less well, with an accuracy of 64%, precision of 58%, f1_score of 71%, and an AUC of 74%, prior to hyperparameter tuning. Following random search hyper parameter tuning, both the CNN and DNN models showed improvement, with the CNN achieving high accuracy of 83% with precision of 86%, high recall of 92%, f1-score of 84%, and high AUC of 88%. The DNN achieved an accuracy of 80%, precision of 73%, recall of 72%, f1_score of 73%, and an AUC of 79%. It is evident that when it comes to breast cancer classification, the CNN model performs better than the DNN model.

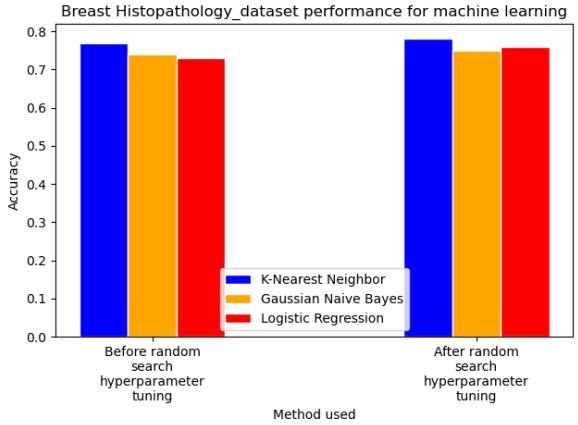

Fig. 12 Breast Histopathology accuracy performance for machine learning

Fig 12 demonstrates the breast histopathology's accuracy performance both before and after random search hyperparameter tuning on machine learning. The results indicate that the K-nearest neighbor had highest accuracy for before and after hyperparameter tuning with an improvement after hyperparameter it had 78% as compared to the 77% before hyperparameter tuning, the gaussian had an improvement from 74% to 75% and the logistic regression improved from 73% to 76% after performing random search hyperparameter tuning.

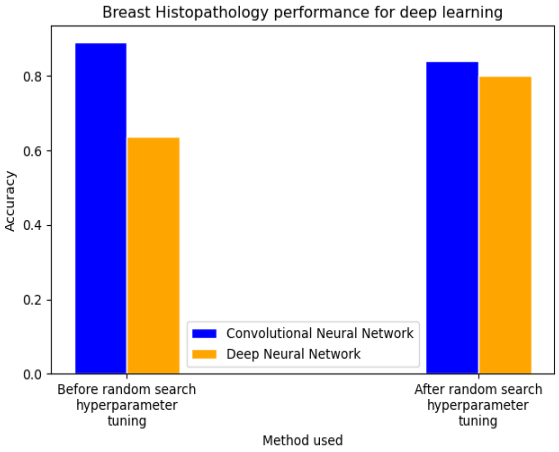

Fig. 13 Breast Histopathology dataset Deep learning performance

Fig 13 demonstrates the breast histopathology's accuracy performance both before and after random search hyperparameter tuning on deep learning This shows a reduction in the accuracy of the CNN model after using random search hyperparameter tuning as compared to before. The DNN however showed an improvement in its performance after performing random search hyperparameter tuning from 64% to 80%. Overall, the CNN model performed well as compared to the DNN model.

6. Discussion of findings

Performing random search hyperparameter tuning has shown an improvement in the performance of the model as compared to before performing tuning on the models, results show an improvement in performance evaluation in precision, recall, f1_score as compared to before, in both deep learning and machine learning approaches. It has been further observed that the deep learning and machine learning models performed better with random search hyperparameter tuning on the BreakHis_400X dataset than the breast histopathology dataset this is because smaller dataset tend to be effective in finding best combinations faster without costing any computational power with low dimensional data. However, because breast histopathology has a large degree of data, random searches would find it difficult to discover the hyperparameter effectively [30]. As a result, the hyperparameter technique would yield less than ideal findings, resulting in a less optimal model and performance.

The breakHis_400x dataset results show that the CNN Model is the highest in terms of performance when it comes to classifying breast cancer with an accuracy of 87.5% seconded by deep neural network (DNN) 84.6%. Amongst the machine learning classifiers, it has been observed that the gaussian naïve bayes performs quite good when random search hyperparameter tuning has been performed although the K-nearest neighbour is better than the other machine learning classifiers. The CNN model performed exceptionally well in terms of performance and accuracy than the DNN model in predicting between the two classes of breast cancer. This shows that CNN is better at classification than DNN. This is attributed to the fact that CNN has the ability to leverage features of the images through the use of convolutional layers than a DNN model [31].

7. Conclusion

In conclusion, this study suggested a deep learning and machine learning method for classifying breast cancer using random search as a hyperparameter tuning. The research was done on two publicly available datasets on Kaggle: The Breast Histopathology Cancer dataset and the BreakHis_400X dataset. The results show that Deep learning classifiers perform better at classifying images as compared to machine learning with CNN performing extremely good than DNN models. Additionally, it was found that the breakhis_400x dataset performed remarkably well with random search hyperparameter tuning as compared to the breast histopathology dataset. This is because the breakHis_400X dataset which is smaller in size allows for faster and more efficient search of good combinations without consuming computational power than the breast histopathology dataset. Future work will focus on finding better hyperparameter tuning techniques that would improve the performance of both deep learning and machine learning models.

References

[1] R.L. Siegel, K.D. Miller, N.S v. and A. Jemal, "Cancer Statistics," CA: A Cancer Journal for Clinicians, vol. 73, no. 1, pp. 17-48, 2023. https://doi.org/10.3322/caac.21763

[2] World health organization. “Breast cancer”. 2023. Available: https://www.who.int/news-room/fact-sheets/detail/breast-cancer

[3] World Health organization. “Cancer”. 2022. Available: https://www.who.int/news-room/fact-sheets/detail/cancer

[4] National Cancer Institute. “Common Cancer Types”. (n.d). Available at https://www.cancer.gov/types/common-cancers

[5] G.W. Prager, S Braga, B. Bystricky, C. Qvortrup, C. Criscitiello, E. Esin, G.S. Sonke, G.A. Martínez, J.S. Frenel, M. Karamouzis , M. Strijbos , O. Yazici , P. Bossi , S. Banerjee, T. Troiani, A. Eniu, F. Ciardiello, J. Tabernero, C.C. Zielinski, P.G. Casali, F. Cardoso, J.Y. Douillard, S. Jezdic , K. McGregor, G. Bricalli, M.Vyas, A. Ilbawi. “Global cancer control: responding to the growing burden, rising costs and inequalities in access”. ESMO Open. 2018 Feb 2;3(2):e000285. doi: 10.1136/esmoopen-2017-000285. PMID: 29464109; PMCID: PMC5812392.

[6]Y. Xu, X. Liu, X. Cao, C. Huang, E. Liu, S. Qian,X. Liu, Y. Wu, F. Dong, C. Qiu, K.Hua, W.Su, J. Wu, H. Xu , Y. Han , C. Fu, Z. Yin, M. Liu, R. Roepman., S. Dietmann, F. Kengara, Z. Zhang , L. Zhang, T. Zhao, J. Dai, J. Yang, L. Lan, M. Luo, Z. Liu, T. An, B. Zhang, X. He, S. Cong, X. Liu, W. Zhang, J.P. Lewis J. P. Tiedje, Q. Wang, Z. An, F. Wang, L. Zhang, T. Huang, C. Lu, Z. Cai, F. Wang and J. Zhang. “Artificial intelligence: A powerful paradigm for scientific research”. The Innovation. 2021, Vol 2, issue 4. https://doi.org/10.1016/j.xinn.2021.100179

[7] L. Yang. “Image classification of MNIST dataset by using machine learning techniques”. UC Merced. 2021. https://escholarship.org/uc/item/81b0z2hh

[8] Assegie A.T. (2021). An optimized K-Nearest Neighbor based breast cancer detection. Journal of Robotics and Control (JRC) Vol 2, Issue 3, pp 115-118. https://doi: 10.18196/jrc.2363

[9] H.H. Farag, L.A.A. Said, M.R.M. Rizk and M.B.E. Ahmed. “Hyperparameters Optimization for ResNet and Xception in the Purpose of Diagnosing COVID-19”.IOS Press. 2021, vol. 41, no. 2, pp. 3555-3571. https://doi:10.3233/JIFS-2109255

[10] A.K. Nugroho and H. Suhartanto. “Hyper-Parameter Tuning based on Random Search for DenseNet Optimization”. 2020 7th International Conference on

Information Technology, Computer, and Electrical Engineering (ICITACEE). 2020, pp. 96-99.

[11] M. Karthik, K. Thangavel and K. Sasirekha. “Novel Deep CNN Model based Breast Cancer Classification.” 2023 7th International Conference on Computing Methodologies and Communication (ICCMC). 2023, pp. 524-529.

[12] K. Sreekala and J. Sahoo. "Hyper Parameter Optimization of Convolutional Neural Networks for Breast Cancer Classification," 2021 International Conference on Advances in Computing and Communications (ICACC), Kochi, Kakkanad, India. 2021, pp. 1-6, https://doi: 10.1109/ICACC-202152719.2021.9708331.

[13] G. Jayandhi, L.J.S. Jasmine and M.S. Jones. X. “Mammogram image classification system using deep learning for breast cancer diagnosis”. AIP Conf. Proc. 2519, 030066 2022. https://doi.org/10.1063/5.0109640

[14] C. Maklin. “Synthetic Minority Over-sampling Technique (SMOTE)”. 2022. Available at https://medium.com/@corymaklin/synthetic-minority-over-sampling-technique-smote-7d419696b88c

[15] A. Özdemir, K. Polat and A. Alhudhaif. “Classification of imbalanced hyperspectral images using SMOTE-based deep learning methods”. Expert Systems with Applications. 2021, Vol 178, https://doi.org/10.1016/j.eswa.2021.114986\

[16] I.H. Sarker. “Machine Learning: Algorithms, Real-World Applications and Research Directions”. SN comput. Sci. 2021 Vol 2, article 160. https://doi.org/10.1007/s42979-021-00592-x

[17] V. Jain. “Introduction to KNN Algorithms". 2022. Available at https://www.analyticsvidhya.com/blog/2022/01/introduction-to-knn-algorithms/

[18] S.F. Khorshid and A.M.Abdulazeez. “Breast cancer diagnosis based on k-nearest neighbors: a Review”. Palarch’s Journal of Archaeology of Egypt/Egyptology 18(4), 2021 pp.1927-1951. ISSN 1567-214x

[19] C. Martins. “Gaussian Naive Bayes Explained with Scikit-Learn”. 2023. Available at https://builtin.com/artificial-intelligence/gaussian-naive-bayes

[20] K. Jain. “How to Improve Naive Bayes? Section 3: Tuning the Model in Python”. 2021. Available at https://medium.com/analytics-vidhya/how-to-improve-naive-bayes-9fa698e14cba

[21] V. Viswanatha, A.C. Ramachandra, B. Avinash and S. Shashank. “Breast cancer classification using logistic regression”. High Technology Letters. 2023, Vol 29, Issue 8. https://gjstx-e.cn/

[22] G.G. Pekhimenko (n.d). Penalized Logistic Regression for Classification.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel., P. Prettenhofer, r. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. “Scikit-learn: Machine Learning in Python”. Journal of Machine Learning Research. 2021, Vol 12 pp. 2825—2830. Available: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[24] I.H. Sarker. “Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions”. SN COMPUT. SCI. 2, article 420,2021. https://doi.org/10.1007/s42979-021-00815-1

[25] L. Alzubaidi, J. Zhang, A.J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamaría, A. Mohammed, F.M. Al-Amidie and L. Farhan. (2021). “Review of deep learning: concepts, CNN architectures, challenges, applications, future directions”. J Big Data 8, article 53, 2021. https://doi.org/10.1186/s40537-021-00444-8

[26] Simplilearn. “What is epoch in Machine Learning?”.2023 Available: https://www.simplilearn.com/tutorials/machine-learning-tutorial/what-is-epoch-in-machine-learning#:~:text=Batch%20size%20is%20a%20hyperparameter,more%20samples%20and%20making%20predictions.

[27] K. Shen. “Effect of batch size on training dynamics”.2018. Available: https://medium.com/mini-distill/effect-of-batch-size-on-training-dynamics-21c14f7a716e

[28] J Brownlee. “Difference Between a Batch and an Epoch in a Neural Network”.2022. Available: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/

[29] J. Lim, S. Jeong, S. Lim, H. Cho, J.Y. Shim, S. Hong S.C. Kwon, H. Lee, I. Moon, J. Kim, Y. Amashita. and M. Kano. “Development of Dye Exhaustion Behavior Prediction Model using Deep Neural Network”. Computer Aided Chemical Engineering. 2022, Vol 49, pp. 1825-1830. https://doi.org/10.1016/B978-0-323-85159-6.50304-3

[30] W. Jia, M. Sun, J. Lian and S. Hou. “Feature dimensionality reduction: a review”. Complex Intell. Syst. 8, 2022, pp 2663–2693. https://doi.org/10.1007/s40747-021-00637-x

[31] F. Nazari and W. Yan. “Convolutional versus Dense Neural Networks: Comparing the Two Neural Networks’ Performance in Predicting Building Operational Energy Use Based on the Building Shape”. n.d. Available: https://arxiv.org/ftp/arxiv/papers/2108/2108.12929.pdf