Journal of Carcinogenesis
Covering Genesis to Therapeutics

Journal of Carcinogenesis, ISSN: 1477-3163
2025; Vol 24: Issue 3s, https://carcinogenesis.com/

A Fine Tuned Deep Learning Model for Accurate Cancer Cell Classification

Achal Sharma[1], Dr. Rahul Mishra[2]

[1]Electronics & Communication Engineering, Dr. APJ Abdul KalamUniversity, Indore, India,
EmailID: achal007sharma@gmail.com

[2]Electronics & Communication Engineering, Dr.APJ Abdul KalamUniversity, Indore, India .
EmailID: rmishra_sati@yahoo.com

ABSTRACT

Lung cancer is one of the deadly disease around the world. In order to cure the disease appropriately, it is essential to identify the type of cancer lungs have. Therefore, the pathological analysis of cancer cells has been performed to identify the category of cancer. This categorization requires an expert and misclassification can impact the entire process of dealing with the disease. In this context, proposed work involve a deep learning model for identifying the cancer cell's type in to three categories namely Lung benign tissue, Lung adenocarcinoma, and Lung squamous cell carcinoma. In order to simulate the proposed model utilizes a dataset obtained from kaggle. This dataset is consist of a total of 15k images of three cancer classes, each class has a total of 5k images. There are two deep learning architectures sequential convolutional neural network (SCNN) and 2D-CNN has been trained. In this experiment, 2D-CNN model was found promising but the hyper parameter tuning has been performed. Based on the performed parameter tuning of the 2D-CNN model the following parameters are found optimal for classifying the cancer cell accurately.

Keywords: *Cancer cell classification, deep learning, hyper-parameter tuning, image classification, machine learning, performance improvement.*

How to Cite: Achal Sharma, Dr. Rahul Mishra, (2025) A Fine Tuned Deep Learning Model for Accurate Cancer Cell Classification, *Journal of Carcinogenesis*, *Vol.24*, *No.3s*, 110-119.

1. INTRODUCTION

Lung cancer is complicated disease and results in a significant burden on health care systems. In 2020, lung cancer resulted in the death of more then 1.5 million people worldwide. Lung cancer remains the most common cancer and a cause of death [1]. Histopathology is the diagnosis and study of diseases. This technique involve the analysis of the tissues and cells under a microscope. In lung cancer disease detection the Histopathology techniques are used to categorize the cancer type. However, this task has been done by the medical experts but small misunderstanding can impact the entire treatment process. Therefore, an automated and reliable technique is required for analysis and categorization of Histopathological images of cancer cells. The aim of the proposed model to making tissue diagnoses and helping clinicians to manage a patient's care.

In this context, as first step an appropriate dataset has been identified. This dataset contains Histopathological images of cancer cells. Next, two deep learning models has been configured to classify the image accurately. During this classification, it is found the classification performance is not consistent. Therefore, we need some optimal parameters for providing accurate classification of cells. In this context, first a different methods of hyper-parameter tuning has been studied. The hyper-parameter tuning is an approach to achieve optimal performance from the deep learning models. But, these techniques are high time consuming and also huge computational resource consuming. Therefore, we utilize an alternate method for tuning the developed CNN model. Further the experimental results has been concluded and future plan has been discussed.

2. PROPOSED WORK

The proposed work is aimed to provide a machine learning model for accurately detection of lung cancer type. This model can be used in Histopathological department for cancer cell classification. The proposed model for implementation is demonstrated in Figure 1.

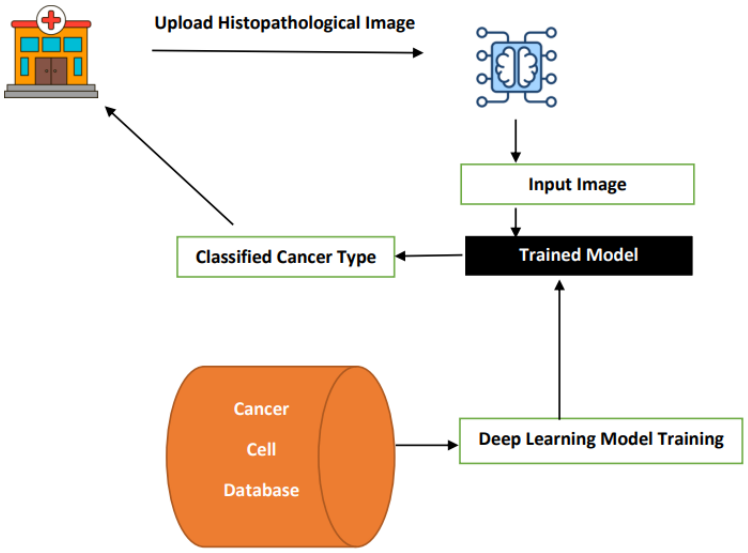


Figure 1 Proposed system architecture

The proposed model is mainly utilized by the clinic or cancer treatment department. Therefore, the user can upload a cancer cell image to the system. The system consist of a pre-trained machine learning model to accept the input image and perform the classification. In order to train the machine learning model, a sample database is used. This database consist of the lung cancer cell images and its types. The machine learning model take training and perform the classification task. The outcome of the classification is sent back to the medical service as the cancer cell type. Therefore, the model contains three main parts first part demonstrate the user interface, second part is defined as machine learning system and third part includes the database. The interface is defined by the communication between system user and machine learning model. Additionally, two other parts i.e. database and machine learning system has been explained in this section.

2.1 Database

In a machine learning model, the example database is playing a key role. Therefore, the database needed to include the data objects and the predetermined prediction. In this presented work, the dataset has been considered from the kaggle [1]. This dataset contains the data of two different category of cancer cells namely lung cancer and colon cancer. Therefore, dataset has five sub directories with 5000 images each. However, the presented work has been dedicated to develop solution for the lung cancer, thus only the data relevant to lung cancer has been considered. The sub directories are containing images of Lung benign tissue, Lung adenocarcinoma, and Lung squamous cell carcinoma. Thus, the dataset have three classes of images and a total of 15k images are available for analysis, which are belonging to three type of cancer. The image sample of dataset is demonstrated in Figure 2.

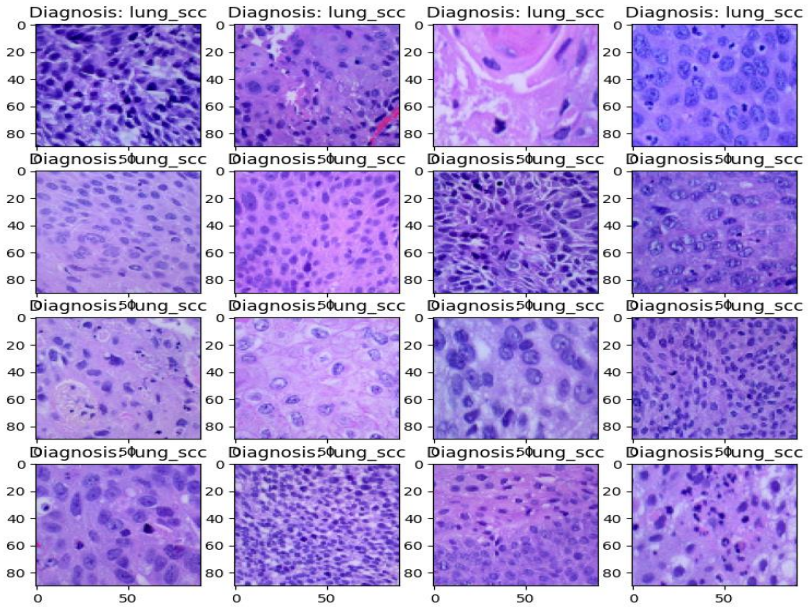


Figure 2 dataset image samples

Before utilizing these images, the image pixels of the dataset has been normalized and scaled between 0 and 1. That help to minimize the computational overhead of the machine learning algorithms. Therefore, the given images of the dataset are used after normalization of the dataset.

2.2 Machine Learning System

According to the available literature the Convolutional Neural Network (CNN) is the most used architecture for dealing with the image classification problems. Therefore, in this work, it is decided to utilize the deep learning models for performing training and recognition of the cancer cells. In order to train and test the model, two configurations of CNN have been considered namely sequential CNN and 2D CNN. The SCNN learn using a linear structure of data and the 2D-CNN can learn on two dimensional data. The simple configuration of SCNN is demonstrated in the table 1. The table 2 contains two columns, first column is containing the layer sequence number and second column contains the confirmation of the given layer. According to this architecture the image data is flattened first, than the flat data of image is passes through the neural network dense layers to learn on the basis of input parameters.

Table 1 Used SCNN configuration

Layers	Configuration
Layer 1	Type = Flatten
Layer 2	Type = Dense, Neurons = 360, activation = relu
Layer 3	Type = Dense, Neurons = 180, activation = relu
Layer 4	Type = Dense, Neurons = 128, activation = relu
Layer 5	Type = Dense, Neurons = 64, activation = relu
Layer 6	Type = Dense, Neurons = 32, activation = relu
Layer 7	Type = Dense, Neurons = 3, activation = softmax

The SCNN model is compiled with Adam optimizer and for calculation of loss categorical cross entropy has been used. Additionally the accuracy is used as performance matrix. Next, 2D-CNN model is configured. The 2D-CNN model is able to accept two dimensional inputs, thus the direct images are utilized to learn. The configuration of the 2DCNN model with their values are given in table 2. First column contains the layer number and second column contains the configuration and values. This model has a total of eight layers.

Table 2 configuration of 2D-CNN model used

Layers	Configuration
Layer 1	Type = conv2D, Filters = 32, kernel_size = (3,3), activation = 'relu', input_shape = (90,90,3)
Layer 2	Type = maxpool2D, size = (2, 2)
Layer 3	Type = Conv2D, Filters = 64, kernel_size = (3,3), activation = 'relu'
Layer 4	Type = maxpool2D, size = (2, 2)
Layer 5	Flatten
Layer 6	Type = Dense, neurons = 64, activation='relu'
Layer 7	Type = Dense, neurons = 32, activation='sigmoid'
Layer 8	Type = Dense, neurons = 3, activation='softmax'

The described 2DCNN model is compiled with Adam optimizer and categorical cross entropy loss function. Both the models have been configured according to the discussed configurations. Additionally, the training of the model has been performed with the 12016 color images and for validation a total of 3005 images was used. After training and validation of the model, the accuracy of the models have been measured. Figure 3 shows the training accuracy and validation accuracy of both the models. In both the diagrams, X axis shows the number of training epochs and the Y-axis shows the accuracy in terms of percentage (%). According to the measured accuracy of both the deep learning models, the SCNN is providing very fewer accuracy for both the context training and validation. The maximum accuracy of SCNN model has been found a total of 33%. On the other hand, the 2D-CNN model is providing higher accuracy for both training and validation initially. But, the sudden break down in performance of 2D-CNN has been observed after the epoch 75. Therefore, the initial performance of 2D-CNN is acceptable.

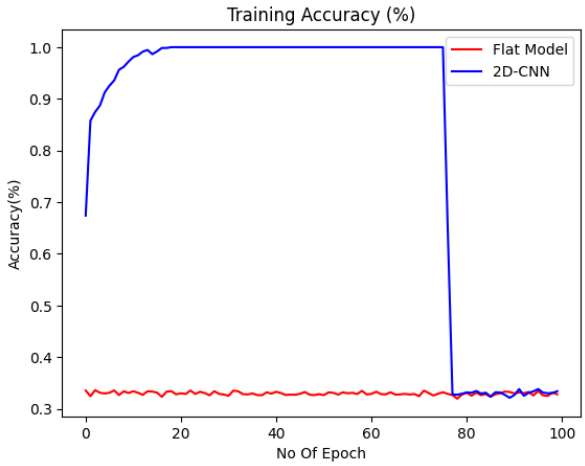


Figure 1 Training Accuracy of used deep learning models

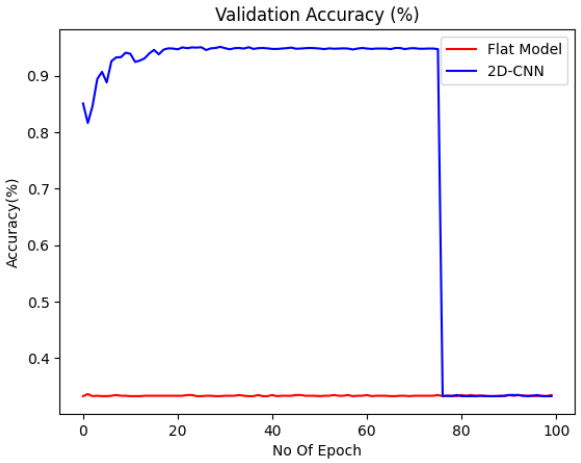


Figure 2 Validation Accuracy of used deep learning models

But, the 2DCNN model is needed to be tune for obtaining the better learning and classification accuracy. At the end of all the training epochs both the models are providing the similar accuracy 33%. In addition, for more detailed analysis precision, recall, and f-score have also been measured for validation sample. Table 3 contains the measured performance parameters.

Table 3 Classification results for both the models SCNN and 2D-CNN

Class	Precision	Recall	F-score	Accuracy	Precision	Recall	F-score	Accuracy
0	0.33	1.00	0.50	0.33	0.00	0.00	0.00	0.33
1	0.00	0.00	0.00		0.33	1.00	0.50	
2	0.00	0.00	0.00		0.00	0.00	0.00	

According to the results, both the classifiers are providing similar results. Therefore, the CNN tuning is essential for optimal results. Thus, the next section is discussing the hyper-parameter tuning for improving the classification results.

3. HYPER-PARAMETER TUNING

The hyper parameter tuning is a technique to identify the optimal neural network parameters, on which we can get highest performance. The hyper parameter tuning is required, when the model is not offering realistic or acceptable results. However, this process is a time consuming process but by using the tuning we can achieve optimal results. There are three type of approaches to perform parameter tuning:

1. **Hit and trial:** In this method, some random values of the neural network parameters are selected as input. Additionally, based on the behavior of the neural network, the model parameters are adjusted by varying them.
2. **Most popular first:** This method involve the use of those neural network parameters, which are used in most of the similar experiments. Therefore, designers are searching appropriate parameters form research articles.
3. **Using an optimizer:** It involves a heuristic search function and the possible set of parameters, which are needed to tune. These optimizers are applying these parameters to the neural network architecture and test the model for the optimal combination of parameters.

Among the above given methods, the parameter tuning of the model by using the first two approaches are not providing the assurance of best results. Additionally, third approach finds the optimal deep learning parameters by using optimization algorithms. There are some essential methods are:

1. **GridSearchCV:** Grid search is a “brute force” technique of hyper-parameter optimization. In this method, the model is fitted using all the possible combinations. In order to create the combination of parameters a grid of potential hyper-parameter values are used. Here, it is required to log each set’s model performance and then choose the combination that produces the best results. This approach is called GridSearchCV, because it searches for the best set of hyper-parameters from a grid of hyper-parameters values. That is an exhaustive approach to identify the ideal hyper-parameters. But it is a slow process and takes a lot of processing power and time.
2. **RandomizedSearchCV:** As the name, this search method selects values at random to the grid search. This method usage a predetermined set of numbers. Every iteration, random search attempts a different set of hyper-parameters and logs the model’s performance. It returns the combination that provided the best outcome after several iterations. This

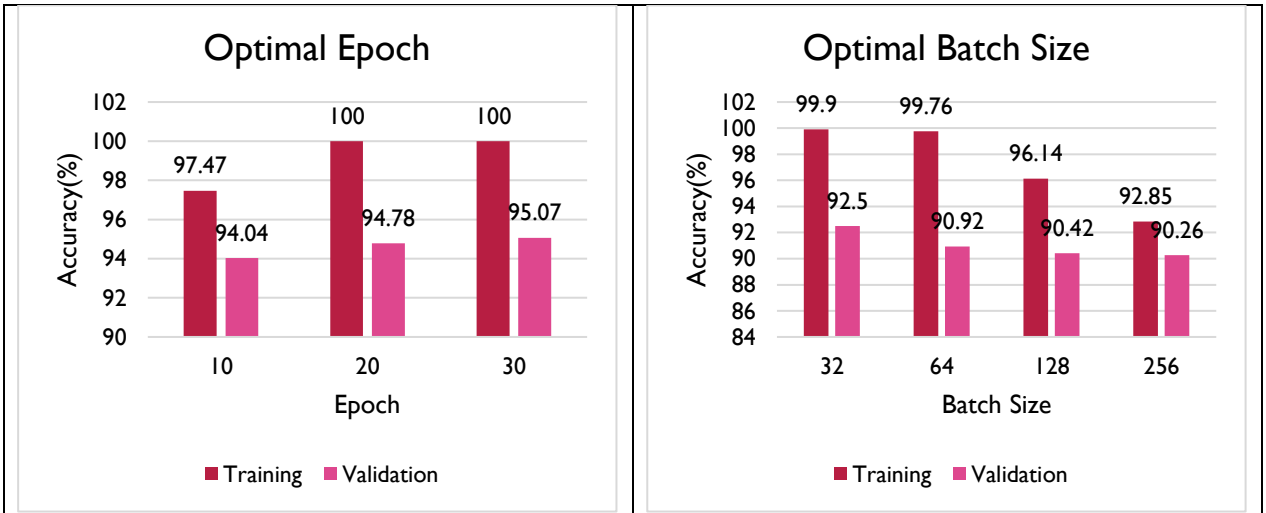
approach reduces unnecessary computation. Thus, it solves the drawbacks of GridSearchCV. It moves within the grid in a random fashion to find the best set of hyper-parameters. The advantage is that, in most cases, random search will produce a comparable result faster than a grid search.

3. **Bayesian Optimization:** Grid search and random search are often inefficient because they evaluate many unsuitable hyper-parameter combinations without considering the previous iterations' results. But, in Bayesian optimization, the search for optimal hyper-parameters are considered as an optimization problem. It considers the previous evaluation results for selecting the next hyper-parameter combination. For this purpose a probabilistic function is used to choose the combination. This method discovers good hyper-parameter combination in few iterations. This method is also used when the objective function is unknown. The probabilistic model estimates the probability of a hyper-parameter combination's objective function result based on past evaluation results.

Therefore, the use of Bayesian optimization is the beneficial for efficient and effective hyper-parameter tuning. In this context, first we have implemented the BayesianSearchCV for hyper-parameter optimization. But after a long running of experiment more than 48 hours, we don't find the solution. Additionally, we also found some computational resource limitations. Therefore, to find optimal hyper-parameters a manual validation technique has been used for tuning. In this context, some essential CNN parameters has been identified namely 'Epoch', 'Batch Size', 'Activation function', 'Weight initialization' and 'Optimizer'. The variation of these parameters are done and the performance has been logged one by one. The aim to select one best parameter once and then, after fixing it we evaluate the next parameter. Additionally, after a limited set of experiments, manually conclude the optimal combination of the hyper parameters. The details about the parameters and the recorded performance has been given as:

3.1 Epoch

In order to find the optimal epochs, the previously performed experimental experience has been used. Therefore, the number of epoch 10, 20 and 30 is considered as input. The last experiment has been performed on all the 15k images. The recorded accuracy of the 2D-CNN model is given in Figure 3(A) for the training and validation. The X-axis of the diagram contains epochs and Y axis shows the accuracy obtained by the model. According to the obtained results, the maximum training accuracy of the model is 100%, which is achieved in 20 and 30 epochs. Additionally, when considering the validation accuracy, the superior results has been found with the number of epoch 30. In, 30 epoch model provides a maximum accuracy of 95.07%. After this accuracy the accuracy of the 2D-CNN model is start declining. Therefore, the epoch 30 is the optimal learning epochs for next experiment.



(A) (B)

Figure 3 shows the performance during the hyper parameter tuning in terms of (A) Accuracy according to epochs (B) Accuracy according to batch size

3.2 Batch size:

The experiments with all the 15k images takes a significant amount of time. Therefore, to speed up the evaluation process, the number of samples from each classes are eliminated. Additionally, a total of 9k images are used for further experiments. In this section the batch size based performance influence is measured. The batch size influences the performance of learning algorithms, during the validation. Therefore, for experiment with the different batch size (i.e. 32, 64, 128 and 256) have been used. Additionally, by varying the batch size, the same 2D-CNN model has been trained and the performance has been recorded in terms of accuracy. Figure 3(B) contains the accuracy of the models after training of 30 epoch. Because, in previous experiment, we found the 30 epochs are providing highest accuracy. In this diagram, the X-axis shows the used

batch size and the Y-axis shows the corresponding accuracy of the model for training and validation. According to the results, with the increasing batch size the training and validation accuracy is reducing. Therefore, lower size of the batch size is providing the highest accurate results. Thus, after number of epochs 30 and batch size 32 is providing acceptable accuracy. Now, both the parameters are fixed for next experiment.

3.3 Activation functions

Activation function is a mathematical function, which is applied to the output of a neural network neuron. The purpose is to provide non-linearity into the model, to learn and represent complex patterns. In the case of linearity, neural network would behave like a linear regression model. The neural network can use different type of activation functions such as 'softmax', 'softplus', 'softsign', 'relu', 'tanh', 'sigmoid', 'hard_sigmoid', and 'linear'. Now in this experiment, the 2D-CNN is used with number of epochs 30 and batch size 32. The training has been performed and the accuracy has been recorded for training and validation samples. The accuracy of the prepared configurations based on different activation function has been given in figure 4. The X-axis of this diagram is demonstrating the different activation functions used. Additionally, the Y-axis shows the accuracy in percentage (%). According to the conducted experiments, there are four activation functions, which are promising for effective learning. These functions are namely softsign, relu, tanh and linear. Among them, the highest accuracy is achieved with the 'tanh' activation function. The 'tanh' function is providing an accuracy of 93.5%. Thus, in further experiments the tanh activation function is used for experiments.

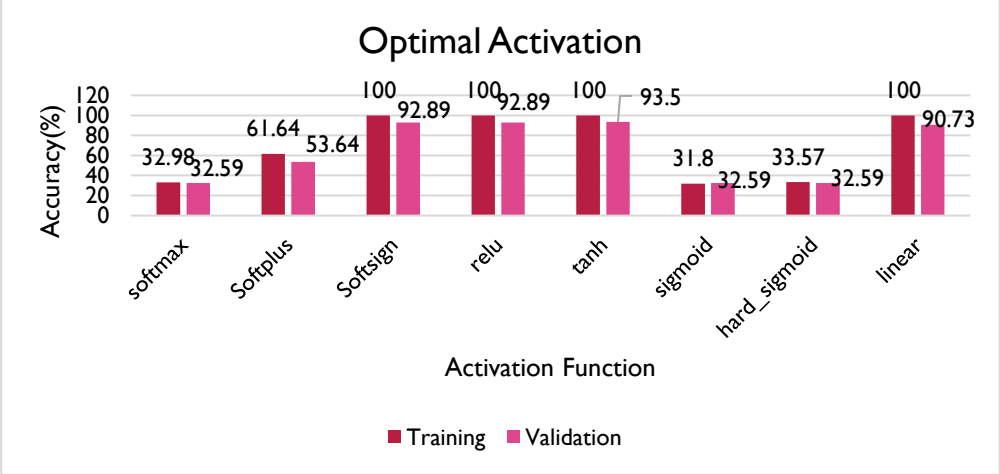


Figure 4 Accuracy according to activation functions

3.4 Initialization of network:

The initialization of neural network layer's weight also influences the learning process of the model. Therefore, to validate the network performance under different weight initialization function different weight initialization functions are used. These functions are 'uniform', 'lecun_uniform', 'normal', 'zero', 'glorot_normal', 'glorot_uniform', 'he_normal', and 'he_uniform'. The performance in terms of accuracy for the 2D-CNN model with different weight initialization functions are given in Figure 5. In this figure, the X-axis shows the name of weight initialization functions and the Y-axis shows the accuracy. According to the results, among the evaluated eight weight initialization functions, only five initialization functions are providing acceptable results i.e. above 90%. Additionally, 'glorot_uniform' based weight initialization technique is providing the highest accuracy 92.5%. Thus, in next experimental settings the 'glorot_uniform' based weight initialization technique is used.

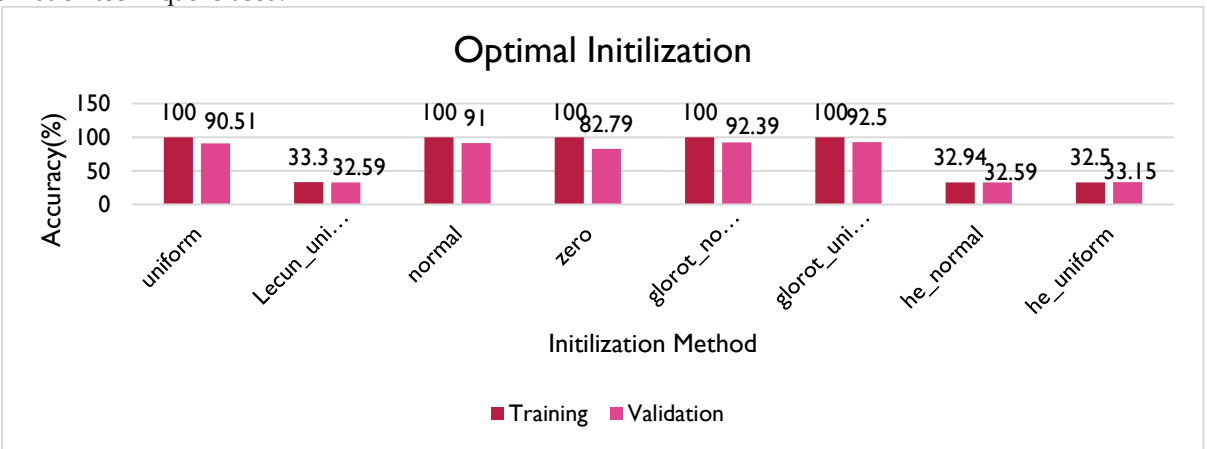


Figure 5 Accuracy of CNN model according to the different weight initialization techniques

3.5 Optimizer

The model support different optimization techniques to improve the learning and estimate new possible weights for the neural network. Therefore, the different optimizer functions are used to achieve better accuracy. Normally, in deep learning architectures various optimization techniques used among some common techniques are 'SGD', 'RMSprop', 'Adagrad', 'Adadelta', 'Adam', 'Adamax', and 'Nadam'.

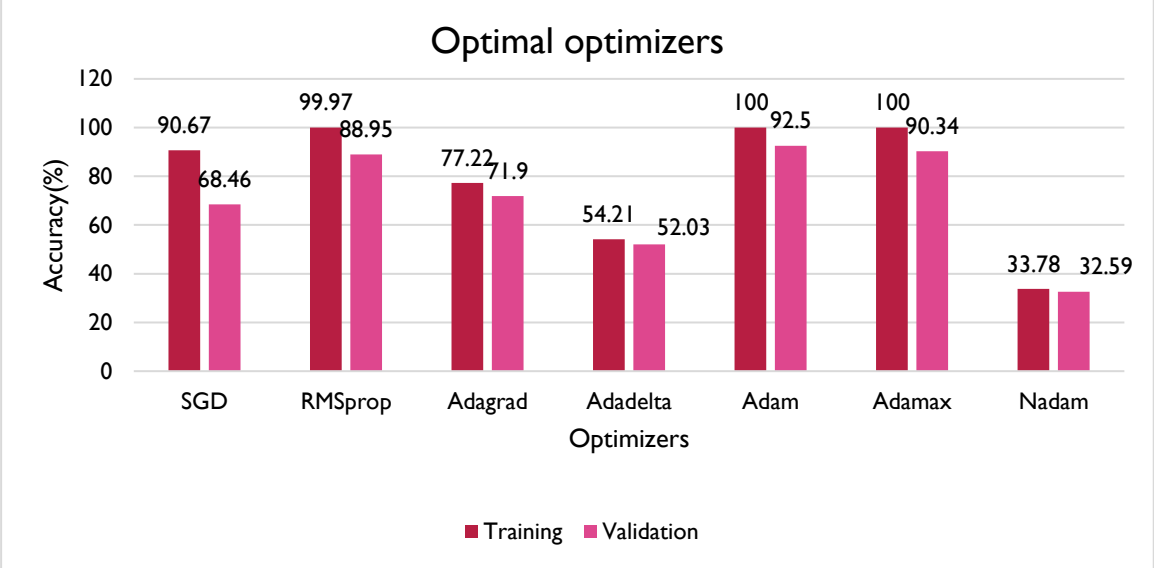


Figure 6 Accuracy according to different optimization algorithms

Therefore, in this experiment all these techniques have been considered one by one and the accuracy has been recorded for the 2D-CNN model. Figure 6 demonstrate the accuracy based on optimization functions. The X-axis of the diagram include the optimization functions and the Y-axis shows the accuracy in percentage (%). According to the variation in optimizer the classifier's accuracy is varying differently, but there are two promising optimizers have been concluded which are providing acceptable accuracy. The Adam and Adamax are the promising optimizers, which scores 92.5% and 90.34% accuracy. Thus, for the final model building Adam optimizer has been considered.

4. RESULT AND DISCUSSION

After performing the detailed investigation about the different parameters, which can improve the classification ability of neural network, the three configurations have been concluded. Table 4, demonstrates final and promising three architectures to get higher accuracy. After configuring the given models using python, the entire dataset images 15k images have been used. Therefore, the total training samples are 12016 images and the 3005 images are used for validation.

Table 4 Most promising architectures for cancer cell classification

Layers	Configuration 1	Configuration 2	Configuration 3
Layer 1	Type = conv2D, Filters = 32, kernel_size = (3,3), activation = 'relu', input_shape = (90,90,3)	Type = conv2D, Filters = 32, kernel_size = (3,3), activation = 'tanh', input_shape = (90,90,3)	Filters = 32, kernel_size = (3,3), kernel_initializer = 'glorot_uniform', activation = 'tanh', input_shape = (90,90,3)
Layer 2	Type = maxpool2D, size = (2, 2)	Type = maxpool2D, size = (2, 2)	Type = maxpool2D, size = (2, 2)
Layer 3	Type = Conv2D, Filters = 64, kernel_size = (3,3), activation = 'relu'	Type = Conv2D, Filters = 64, kernel_size = (3,3), activation = 'tanh'	Type = Conv2D, Filters = 64, kernel_size = (3,3), activation = 'tanh'
Layer 4	Type = maxpool2D, size = (2, 2)	Type = maxpool2D, size = (2, 2)	Type = maxpool2D, size = (2, 2)
Layer 5	Flatten	Flatten	Flatten
Layer 6	Type = Dense, neurons = 64, activation='relu'	Type = Dense, neurons = 64, activation='tanh'	Type = Dense, neurons = 64, activation='tanh'
Layer 7	Type = Dense, neurons = 32, activation='sigmoid'	Type = Dense, neurons = 32, activation='tanh'	Type = Dense, neurons = 32, activation='tanh'

Layer 8	Type = Dense, neurons = 3, activation='softmax'	Type = Dense, neurons = 3, activation='softmax'	Type = Dense, neurons = 3, activation='softmax'
Epoch	10	30	30
Batch size	32	32	32
Optimizer	Adam	Adam	Adam
Loss	categorical cross entropy	categorical cross entropy	categorical cross entropy

Based on the conducted experiments, the performance of the fine tune models are demonstrated in figure 6 and figure 7. Figure 6 contains the classification accuracy of the models for training and Figure 7 shows the validation accuracy of the models. In both the diagrams, the X-axis contains the epoch and the Y-axis shows the relevant accuracy in percentage (%). According to the training accuracy of the models, all the three models are reaches to the 100% accuracy but the model 3 shows the smooth and fast learning as compared to other two implemented model.

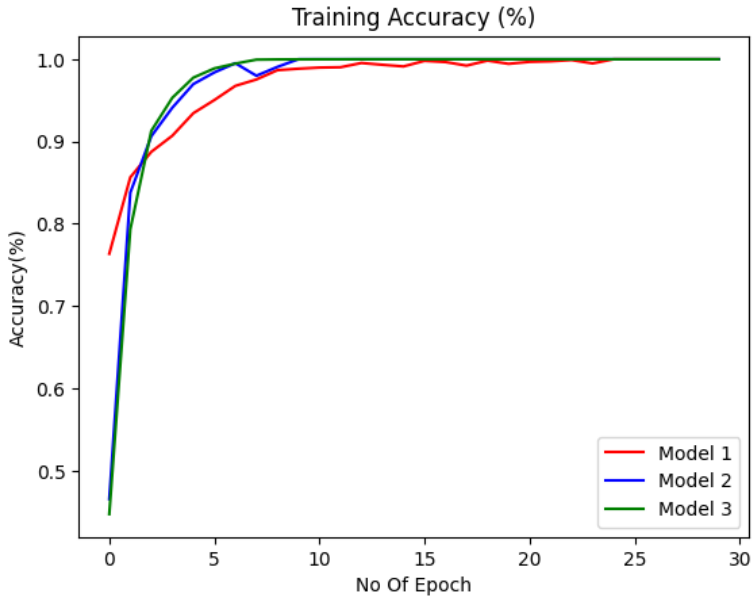


Figure 6 Fine-tuned 2D-CNN model's training accuracy

In addition, when focusing on the validation accuracy then it is found the model 1 is providing the highly fluctuating performance. On the other hand, the model 2 and model 3 have less variation in the accuracy. Additionally, the results are more consistent as compared to model 1. Thus the model 3 is most effective and acceptable 2D-CNN model for cancer cell classification.

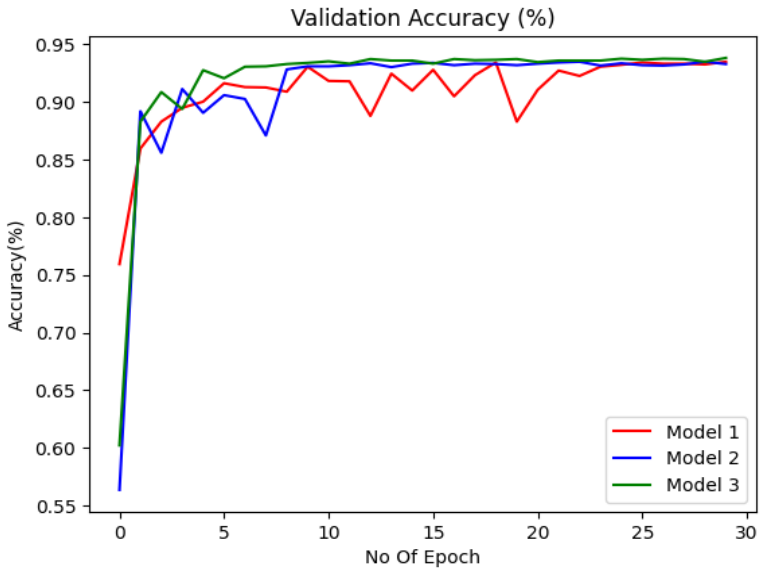


Figure 7 Validation performance of the models

Table 5 shows the performance of the CNN model in terms of precision, recall and f-score. Based on the classification results of the implemented models, the model three is providing the superior performance. The model 3 is able to classify 94% accurate cancer cells. Basically, we consider the performance according to the class-wise classification of cancer cell images. Additionally, using the classification the trained model 3 is leading both the classifier's performance for all the three classes, which are needed to identify.

Table 5 precision, recall and f-score of the CNN model

Models	Precision			Recall			F-score			Accuracy
	0	1	2	0	1	2	0	1	2	
Model 1	0.91	0.98	0.91	0.90	0.99	0.92	0.90	0.99	0.92	0.93
Model 2	0.91	0.99	0.90	0.89	1.00	0.91	0.90	0.99	0.90	0.93
Model 3	0.91	0.99	0.92	0.91	1.00	0.91	0.91	0.99	0.91	0.94

Next, the classification outcomes in terms of confusion matrix has been also discussed. The confusion matrix helps to understand, how accurately the model is classifying the images. In this experiment, a total of 996 images of class 0, 1006 images of class 1 and a total 1003 images of class 2 has been used for validation. According to the performed validation of model 1 the confusion matrix of the model 1 is given in table 6. According to the results, among 996 images of class 0, model 1 are recognizing 903 images as class 0, and 9 images are misclassified as class 1 and 84 images as class 2. Similarly, from 1006 images of class 1, the model 1 recognizes 989 images accurately as class 1. Additionally, misclassified 17 images as class 0. Finally, when considering class 2 images then, it is found the model is recognizing 917 images accurately as class 2 and miss classified 85 images as class 0 and 1 image as class 1.

Table 6 confusion matrix for model 1

Classes	0	1	2
0	903	17	85
1	9	989	1
2	84	0	917

The confusion matrix of the model 2's validation is given in table 7. During this validation, it is found the 896 images are classified as the class 0. On the other hand, 1 image is miss classified as class 1 and 89 images as class 2. Similarly, among the total images of 1006 images of class 1, a total of 998 images are accurately classified as class 1. Additionally, 9 images are miss classified as class 0 and 3 images class 2. Finally, among the total 1003 images of class 2, a total of 909 images are correctly recognized and a total of 100 images are miss classified as class 0.

Table 7 confusion matrix for model 2

Classes	0	1	2
0	896	9	100
1	1	998	0
2	89	3	909

The confusion matrix of the model 3 is demonstrated using table 8. According to the validation based confusion matrix of model 3, among a total of 996 images of class 0, the model 3 is able to recognize the images as class 0 is 911 and 3 images are misclassified as class 1 and 88 images are misclassified as 88. Secondly, among a total of 1006 images of class 1, the model 3 is able to recognize 996 images accurately and only 13 images are misclassified as class 0. Additionally, 1 image is misclassified as class 2. Finally, when considering the classification of 1003 images of class 2, a total of 912 images are correctly classified. On the other hand, a total of 81 images are misclassified as class 0.

Table 8 confusion matrix for model 3

Classes	0	1	2
0	911	13	81
1	3	996	0
2	88	1	912

According to all the evaluation and experiments, the model 3 is found most appropriate model for performing cancer cell classification. Therefore, the model 3 is an acceptable model for preparing the proposed application.

5. CONCLUSION AND FUTURE WORK

In this paper, a machine learning model has been proposed for accurately cancer cell classification. The aim is to support the research and analysis purpose for cancer treatment practices. In this context, the deep learning model has been proposed for system development. According to the initial analysis, the CNN model is not providing a reliable result. Therefore, the hyper-parameter tuning is performed for finding an appropriate configuration of the CNN. In this parameter tuning practices the five hyper-parameters namely epoch, batch size, activation function, optimizer and weight initialization function have been considered. According to the evaluation of different combination of CNN configurations, the optimal number of epoch is found 30. After that, up 75 epoch it shows slightly increasing accuracy but sudden break down in performance occurred. Secondly, when considering the batch size, than it is observed, the smaller batch size helps to improve accuracy of model but increases the cost of training in terms of training time. However, 32 batch size is providing optimal results.

Next, the activation function have been evaluated. According to the experiments, for our case the 'Tanh' function is providing the highest accuracy. Further, when the weight initialization function is considered. According to the observation, it is found, the 'glorot_uniform' based weight initialization function is providing the accurate results. Finally, when considering the optimizer function, then the 'Adam' optimizer is found the most appropriate then other optimizer function for the cancer cell classification. Based on the hyper parameter analysis, three most promising configurations has been concluded and their performance analysis has been performed. The comparison and results demonstrated in terms of precision, recall, f-score, and accuracy. In addition, the effectiveness of classification is also demonstrated in terms of confusion matrix. The final model achieves up to 94% classification accuracy.

REFERENCES

[1] Z. Wu, F. Wang, W. Cao, C. Qin, X. Dong, Z. Yang, Y. Zheng, Z. Luo, L. Zhao, Y. Yu, Y. Xu, J. Li, W. Tang, S. Shen, N. Wu, F. Tan, N. Li, J. He, "Lung cancer risk prediction models based on pulmonary nodules: A systematic review", Thorac Cancer. 2022; 13 : 664–677.

[2] https://www.kaggle.com/datasets/andrewmvd/lung-and-colon-cancer-histopathological-images?resource=download