Lung Cancer Radar: Augmentation and categories in diagnosis of lung cancer

Ahmed Mohamed Ahmed Hassan[1]
Mohamed Mahmoud Bakr El-Tohfa[2]
Mohamed Ragab Mohamed El-Morsy[3]
[1, 2, 3] Kafr El sheikh STEM High School
[1] Nasr City, Cairo, Egypt, AhmedHassan200282@gmail.com
[2] Kafr Mostnad, El Beheira, Egypt, mohamed.1818042@stemksheikh.moe.edu.eg
[3] Alalamia, Biala, Kafr El-Sheikh, mohamed.1818036@stemksheikh.moe.edu.eg

DOI: 10.31364/SCIRJ/v9.i09.2021.P0921877
http://dx.doi.org/10.31364/SCIRJ/v9.i09.2021.P0921877

***Abstract*- The crucial issue in dealing with lung cancer globally is the diagnosis process. for a poor citizen, it could take up to days to determine certainly that he has the tumor, and at this point it would be in the late stage. Plus, from every five people, two are misdiagnosed, and to complete the normal diagnosis steps it takes time, effort, and money; however, in the last decade, computer-based applications proved to be a valuable technique to solve this problem. So, it is decided to construct an application by utilizing artificial intelligence (AI), specifically Machine Learning, to solve the issue of diagnosis by focusing on accuracy and time as design requirements, by predicting if the patient has the tumor or not; besides, the application will determine what is its category: adenocarcinoma and large cell carcinoma; moreover, after testing and training the application for 30 trails (epoch) in the end-user version and using the augmentation technique, the accuracy reached to 94.21%, which is about 1.57 times that of human power. Plus, the time to predict the tumor type did not exceed 3 seconds. It is concluded that the application could clearly reduce the time, effort, and money, which are compensated in the diagnosis procedures, and still produces efficient results.**

***Index Terms*—Artificial Intelligence, Machine learning, augmentation, adenocarcinoma, large cell carcinoma.** *(Key words)*

I. Introduction

For Egypt, eleven grand challenges slow the development and the road to prosperity; however, Egypt is a developing nation and that means it must eliminate all these challenges which range from health issues to industrial deterioration and pollution. In health Issues, the expression of lung cancer is an extreme fear to the citizens, as it is the gravest disease in the whole world (in Egypt, it is about 4.7% (fig. 1)). The eminent issue in the lung cancer treatment is to discover the tumor early and that for sure comes from the diagnosis, which was a real problem to find a way to cut out the diagnosis time, effort, and money consumed in the procedures. So, it is decided to use artificial intelligence for solving this obstacle according to the current capstone challenge.

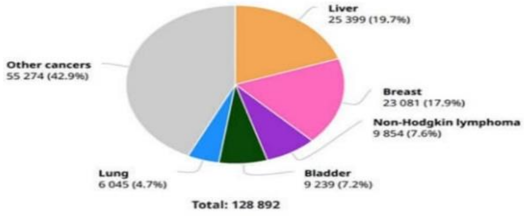


Figure 2 shows the Image processing using segmentation technique

The main problem which is focused on is the identification of the cancer tumor in the lung if exists and determining its kind accordingly: Adenocarcinoma and large cell Carcinoma, and this process will be done by scanning the computed tomography (CT) images of real patients [John Hopkins, 2012]. Plus, this operation will be fulfilled by utilizing Artificial Intelligence and machine learning by constructing an application using the python language and convolution neural network (CNN). This application is desired to overcome two essential obstacles as a test to measure its qualifications, the main design requirements that appears in every diagnosis: Time consumption and accuracy (efficiency).

One of the previous attempts to solve these issues of the diagnosis was by employing BUOY application [BUOY, 2014] which depended on the patient entering his own symptoms as the inputs of the application to provide it with sufficient information to predict if the patient has the tumor or it is just a normal inflammation. This process took neglectable time; nevertheless, the notable problem that raised from this technique is the uncertainty of the results, as the patient is the one who is responsible for accounting the inputs and it may be irrelevant. So, the accuracy of this solution is damaged.

Another solution was offered that combines a CNN model with an enormous dataset before using segmentation (fig. 2) technique [G.A & P.K, 2018] which resulted 20021 images in total that produced a very high accuracy; however, the main problem was in the time consumed in training and revealing the results as they made numerous numbers of models and trained each. So, their application cannot readily used for each circumstance as it took long for training each time in every new place.

The team's proposed solution and the idea were chosen by focusing on three fundamental points: avoiding prior solutions' weaknesses, advancing their strengths, and adding new views. The first step was to find a way to reduce the consumed time in the training portion, and that was fulfilled by not construction

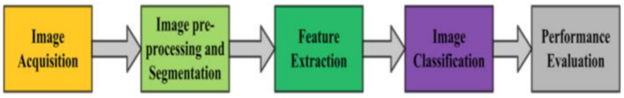


Figure 2 shows the Image processing using segmentation

© 2021, Scientific Research Journal

separated models, but by forming different models in the same model. Also, the segmentation process will be replaced by another technique called Augmentation. The design requirements will be tested by comparing the results with doctors and previous solution ones.

II. Materials (SOFTWARE)

1. Spyder Source IDE
2. Python Language
3. NumPy Library
4. Pathlib Library
5. Matplotlib.pylab Library
6. CV2 Library
7. Train_test_split Library
8. TensorFlow/keras layers/models/ Sequential Library

III. Methods and Test plan

Due to the fact the idea is an application which relays on the AI codding integrated with Python, the first step was to sketch the process on papers (fig. 4) [Swetha Subramanian, 2018] to ensure that there is no syntax error or systematic error. The application was constructed as following:

- **Stage 1**: Importing all libraries. Importing a library called "pathlib" to enter each image in its own path. Forming the dataset for each of the training section and the testing. Entering each image with extension "jpg". Determining the objects that will be compared (normal, Adenocarcinoma, and large cell carcinoma). Then, selecting two variables (X, Y) which are the arrays. Resizing each image into (128 pixels*128 pixels). Selecting of (try and except) function.
- **Stage 2:** Selecting the CNN path by selecting six hidden layers. Selecting the Max pooling function. Selecting an activation function called (Soft Flattening of the 2D matrix into a 1D matrix as computer understand only 1D matrix. Densing 1D matrix into 128 neurons only.
- **Stage 3:** Dropping out some neurons in response to generalize the dataset. Densing the final matrix into 3 neurons. Selecting of accuracy as the metrics of the model. Determining the number of epochs (number of filtrations for each image).

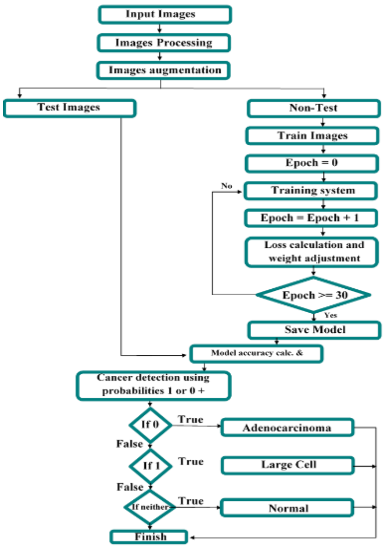


Figure 4 shows the Flow

- **Stage 4:** Saving the final model in a name called (model.h5) to summon it in the prediction code. Selecting the prediction code by function called (model. Predict ('name of the image').

Test plan:

Testing the application was conducted into three phases: Alpha, Beta, and End-user; in addition, to test the design requirements, in each phase, a code was specifically used which utilizes the efficiency formula mentioned in the Data Analysis section, and the number of epochs was increased gradually increased till reaching the maximum accuracy. The test plan was as following:

the **Alpha stage**, two models will be designed with different factors to construct the basic application: resizing images is different (model (2) is 256, and model (1) is 128), adding different number of hidden layers in each one (model (2) is 5, and model (1) is 6), and either adding or removing SoftMax function. This act was in response to reach the maximum accuracy in the least time.

the **Beta section**, the two models are tested according to the following parameters: using 10 epochs, 10 times of training. The highest one was then transferred to the End-user stage.

The **End-user stage**, an additional code was added "the Augmentation" that is used to manipulate the images of the training set to increase the dataset for more credibility and higher efficiency. Next, the two models are trained with 30 epochs, and the final products will be compared using the parameters of accuracy and time consumed; moreover, both models will be tested with the testing dataset of 10 unknown images to the application to test the same parameters (design requirements) in both to be the End-user product.

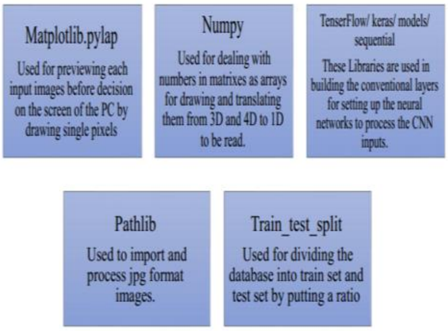
IV. Analysis

The idea depends on some specific procedures:

At first, importing all libraries (fig. 5). Then collecting the CT- scans as a database for (Normal state, Adenocarcinoma and Large Cell Carcinoma) [CancerImagingArchive, 2001] next, uniting all images to JPG extension by using image convertor application to not make differences during classification. The procedures of coding were as the following:

First, importing all libraries:

Each library has a specific function (fig. 5). Constructing the code that reads the dataset and forming the three objects of the idea (normal, Adenocarcinoma, and large cell carcinoma) to link each type to its specific folder of datasets.



Matplotlib.pylap
Used for previewing each input images before decision on the screen of the PC by drawing single pixels

Numpy
Used for dealing with numbers in matrixes as arrays for drawing and translating them from 3D and 4D to 1D to be read.

TenserFlow/ keras/ models/ sequential
These Libraries are used in building the conventional layers for setting up the neural networks to process the CNN inputs.

Pathlib
Used to import and process jpg format images.

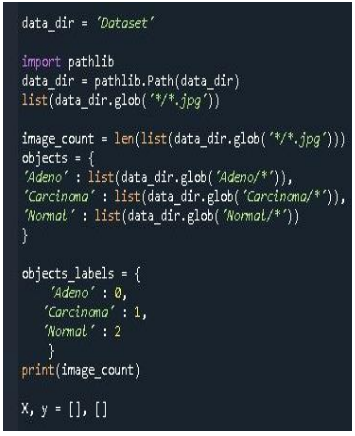
Train_test_split
Used for dividing the database into train set and test set by putting a ratio

Figure 5 shows each library function

© 2021, Scientific Research Journal

Second, selection of the path of each image:

Importing the Pathlib library to select a specific path for each image to organize the determination of data's features. Next, the identification of each image in (X, Y) variables to represent images by the matrix to be understandable by the computer. Resizing each image -using (CV2) library- into (128 pixels *128 pixels) to be suitable for training the model on the graphic processing unit (GPU) (fig 6). The next step is to select (try and except) function as if there is a different extension or different size of the image as selected then, this image will be skipped.



```
data_dir = 'Dataset'

import pathlib
data_dir = pathlib.Path(data_dir)
list(data_dir.glob('*/*.jpg'))

image_count = len(list(data_dir.glob('*/*.jpg')))
objects = {
'Adeno' : list(data_dir.glob('Adeno/*')),
'Carcinoma' : list(data_dir.glob('Carcinoma/*')),
'Normal' : list(data_dir.glob('Normal/*'))
}

objects_labels = {
    'Adeno' : 0,
   'Carcinoma' : 1,
   'Normal' : 2
    }
print(image_count)

X, y = [], []
```

Figure 6 shows the path code

Third, train _test split function:
Dividing the data into 90% train and 10% test by using the(train_test_split) function. It had been decided to use these specific percentage to prevent under fitting error

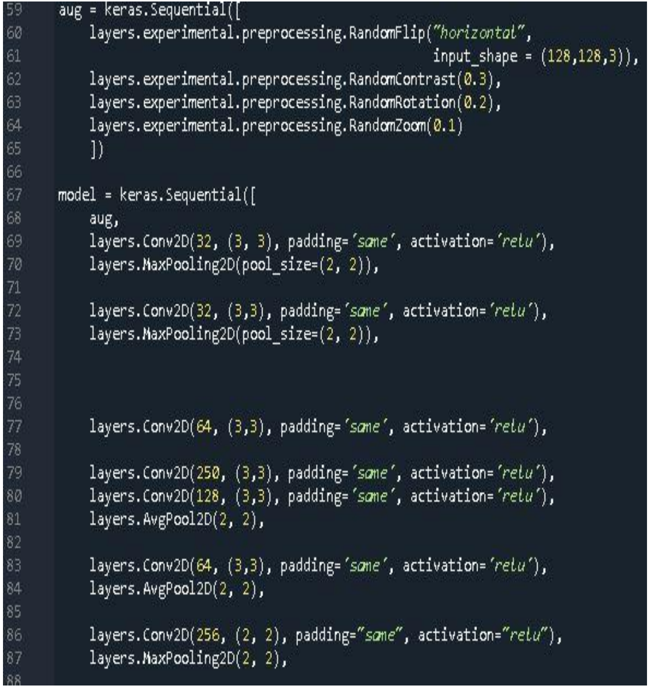
Fourth, Augmentation function (Aug):
Constructing (Aug) function by the following codes: first, select (random flip) function to rotate image randomly then interring the shape of the images. Second, select (Randomcontrast) function to change the concentration of color by a ratio (0, 3). Third, select (Rundomrotation) by the ratio (0, 2). Fourth, select (Rundomzome) function by ratio (0, 1). These specific ratios had been determined not to make a huge difference in the datasets.

The AUG Code:

```
Aug = keras.Sequential([
layers.experimental.preprocessing.Randonf1ip("horizontol",
                                   input_shape = (128,128,3)),
layers.experimental.preprocessing.RandomContrast(0.3),
layers.experimental.preprocessing.RandamRotation(0.2),
layers.experimental.preprocessing.RandamZoom( 0.1)
])
```

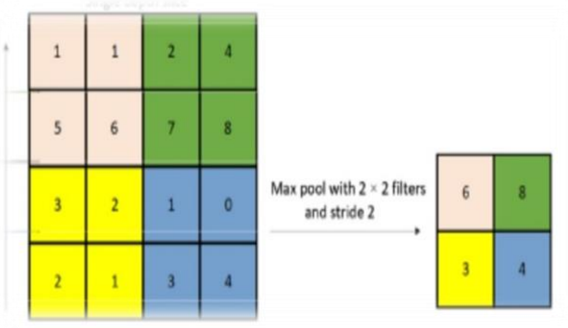
Fifth, convolution neural networks (CNN) processes:

Structuring (CNN) processes (fig. 7) (fig. 8) [Thomas wood, 2019] to determine the features of the images. at first, using Sequential library from Keras. Adding convolution two dimensions (CONV2D) hidden layer to determine the pixels of each image as a matrix to be understood by the computer. Selecting Max pooling function to extract the most special matrix for each image (fig 9). It is decided to use seven hidden layers with their Max pooling function to produce a more accurate matrix. Flattening the input matrix to convert the 2D matrix to a 1D matrix as the computer understand only 1D matrix. Densing these layers to be compared with 128 different neurons. Adding the (Drop out) function to generalize the other parts of the images. Densing the input matrix into three objects: (Normal (neither), Adenocarcinoma (0), and large cell Carcinoma (1)). Making these CNN process easier by using activation function called (SoftMax).



```
aug = keras.Sequential([
    layers.experimental.preprocessing.RandomFlip("horizontal",
                                                 input_shape = (128,128,3)),
    layers.experimental.preprocessing.RandomContrast(0.3),
    layers.experimental.preprocessing.RandomRotation(0.2),
    layers.experimental.preprocessing.RandomZoom(0.1)
    ])

model = keras.Sequential([
    aug,
    layers.Conv2D(32, (3, 3), padding='same', activation='relu'),
    layers.MaxPooling2D(pool_size=(2, 2)),

    layers.Conv2D(32, (3,3), padding='same', activation='relu'),
    layers.MaxPooling2D(pool_size=(2, 2)),



    layers.Conv2D(64, (3,3), padding='same', activation='relu'),

    layers.Conv2D(250, (3,3), padding='same', activation='relu'),
    layers.Conv2D(128, (3,3), padding='same', activation='relu'),
    layers.AvgPool2D(2, 2),

    layers.Conv2D(64, (3,3), padding='same', activation='relu'),
    layers.AvgPool2D(2, 2),

    layers.Conv2D(256, (2, 2), padding="same", activation="relu"),
    layers.MaxPooling2D(2, 2),
```

Figure 8 shows the path code



1	1	2	4
5	6	7	8
3	2	1	0
2	1	3	4

Max pool with 2 × 2 filters and stride 2

6	8
3	4

Figure 9 shows Max Pooling

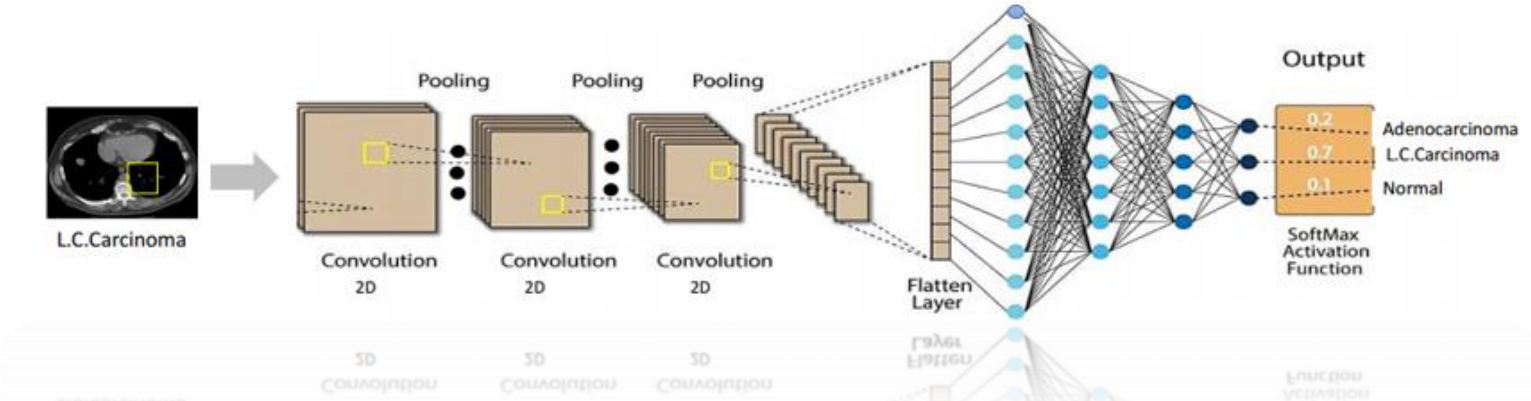


Figure 7 shows the CNN process

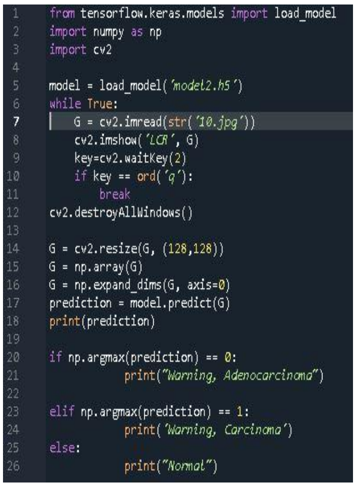
© 2021, Scientific Research Journal

Sixth, compiling the final model, then selecting the accuracy as the last function of the model:

Determining 30 epochs for the final model, saving the final model in only one code by name (Model.5h), to protect the primary code from accidental mistakes.

Seventh, the prediction phase:

Entering the prediction code (fig 10) to predict the type of the input image by selecting the name of the image.



```
from tensorflow.keras.models import load_model
import numpy as np
import cv2

model = load_model('model2.h5')
while True:
    G = cv2.imread(str('10.jpg'))
    cv2.imshow('LCR', G)
    key=cv2.waitKey(2)
    if key == ord('q'):
        break
cv2.destroyAllWindows()

G = cv2.resize(G, (128,128))
G = np.array(G)
G = np.expand_dims(G, axis=0)
prediction = model.predict(G)
print(prediction)

if np.argmax(prediction) == 0:
            print("Warning, Adenocarcinoma")

elif np.argmax(prediction) == 1:
            print('Warning, Carcinoma')
else:
            print("Normal")
```

Figure 10 shows the prediction code

Eighth, the Testing part:

After constructing the model, it was observed that the collected data have enormous differences; thus, it had been decided to collect enough data from the same clinical as the differences will be only in the position of cancer cells. During the final test, model (2) was promoted from the beta phase. One version of it with Augmentations (MAUG) and the other without (M) were tested. The final average accuracy of (M) was higher than the accuracy of (MAUG), although this was expected because with larger dataset more identification errors arise, but it has higher applicability in wider areas, if added enough dataset for each type of Augmentation.

Calculation of this final accuracy was by (the computer in logarithm):

$$Accuracy = \frac{(\mathrm{TP} + \mathrm{TN})}{(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})}$$

Notice: True (T) means accepted to be compared. False (F) means not accepted. Positive (P) means the results is correct. Negative (N) means the result is false.

A view of design requirements:

It is chosen to utilize Augmentation to increase the amount of database to produce more accurate results. Also, employing 30 epochs to get the average accuracy for each model, linking the final model by one code, and adding it to the prediction code were fulfilled, to avoid training the code for every input image as its result of this model may be wrong and that will destroy the accuracy.

Specific laws, theories, and basic of Python coding:

1. Augmentation is a programming way to increase the number of datasets by changing the zoom on images, random rotation, and changing the concentration of the colors. CNN processes are programming steps.
2. From statistics, the mean equation was used to compute both the final mean time and accuracy: Mean Accuracy equals the sum of all epochs' accuracies over number of epochs; moreover, the idea in total depends on the concept of sampling distribution.
3. In physics, [Halliday, 2015], after clear understanding the differences between X-ray images and CT-images by using Compton Effect and X-ray scattering, CT scans were chosen, as they are 3D images and used by all clinics over the globe.

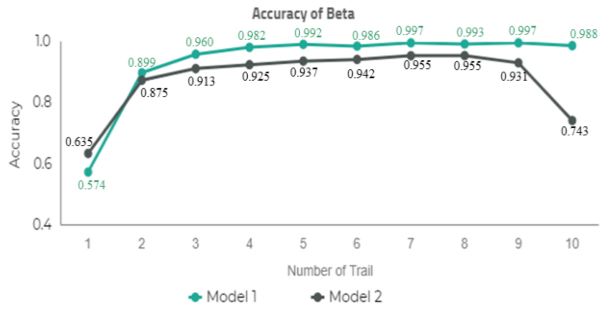


Figure11 shows the Accuracy results for beta phase

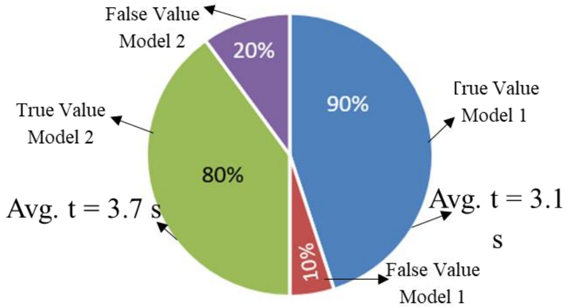


Figure12 shows the results and average time (Avg. t) for the test of the beta phase models

V. Results

A. Beta Phase

After Training the two models on 10 epochs, the accuracy curve for each model resulted (fig. 11). Then the average accuracy for each model calculated to be 0.937 (93.70%) for model (1) and 0.881 (88.10%) for model 2. The time and the results for the beta phase's test was estimated (fig. 12) with an average time score of 3.1 and 3.7 for model (1) and model (2), respectively. So, model (1) was chosen to be the final model as it has higher accuracy and lower time consumption than that of model (2).

B. End-user Phase

From Beta phase, Model (1) was selected to be used in the End-user due to its high accuracy. In addition to, it is decided to train the model on 30 epochs with and without AUG, and fig. (13) shows the accuracy for this training. Then, the average accuracy for the end-user models (with and without AUG) are collected: 0.9421 (94.21%) and 0.983 (98.30%), respectively. After training the models in the End-user phase, the final test

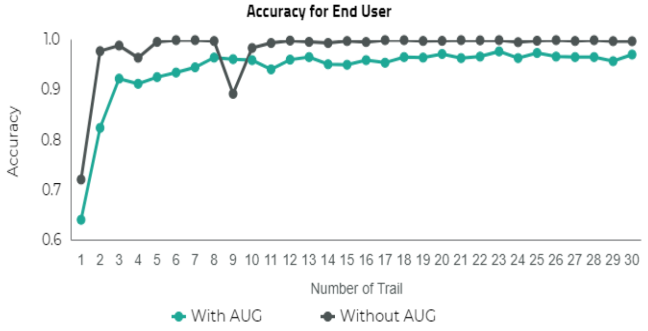


Figure13 shows the Accuracy results for end-user phase models

© 2021, Scientific Research Journal

Table 1 shows the time consumed for tests of each model in the end-user phase

No. of tests	Aug Model time	Without Aug Model time
1	3	2.92
2	2.66	2.33
3	2.01	2.00
4	1.32	1.11
5	1.56	1.37
Average time	2.09	1.94

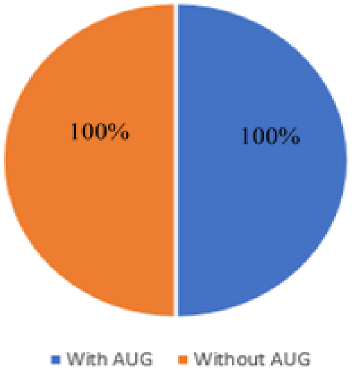


Figure14 shows the test results for end-user phase models

was conducting (fig. 14). Both models (with Aug and without Aug) scored the same: identified all the cases without no issue. The time for End-user stage's test was computed (table 1) with an average score of 2.09 and 1.94 for the model with Aug. and the model without Aug, consecutively.

VI. Conclusion

After testing the final model, it is deduced that the selection of more epochs increases the efficiency of the results. In the beta phase, it was about 93.708% for the model (1) and it was about 98.301% without Augmentations in the End-user phase; Plus, rising the number of hidden layers does not affect the time of training nor predicting subsequentially. In contrast, it contributes notably to increasing the accuracy. Also, it is concluded that using the SoftMax function decrease the processing time as in the beta phase it was around 3.1 seconds for the model with SoftMax and it was approximately 3.7 seconds without SoftMax function; Plus, rising the number of hidden layers does not affect the time of training nor predicting subsequentially. In contrast, it contributes notably to increasing the accuracy.

From the End-user phase, it is deduced that the reason for 94.21% accuracy for the model with Augmentation and 98.301% for the model without Augmentation is that the data was not enough for each difference as (AUG) works randomly. So, Augmentation model with 94.21% accuracy was selected as the results as Augmentation is more applicable for future development and on large scales.

However, this accuracy (94.21%) was higher than that of Performance analysis of various machine learning-based approaches for detection and classification of lung cancer in humans (prior solution) that was maintained in the introduction (88.55%). As well as, this model takes only 2.09 seconds for producing decisions, while the previous prior solution takes about 4.35 seconds.

VII. Acknowledgment

Great thankfulness and appreciation to anyone helped us to complete our mission especially;

- Eng. Shady Mohamed (Student in Kafr El-Sheikh University)
- Mr. Ahmed Emara (school principal)
- Ms. Noha Basiony Abd E-Samad (English supervisor and mentor)

VIII. References

[1] SwethaSubramanian.(2018.).Swethasubramanian/ LungCancerDetection Retrieved December 15, 2020, from https://github.com/swethasubramanian/LungCancerDetection

[2] Lung cancer types. (2012, March). Retrieved February 10, 2021, from

https://www.hopkinsmedicine.org/health/conditions-and-diseases/lungcancer/lung-cancer-types

[3] "@welcomeai". (2014). Buoy Health - Symptom Checker - Check your symptoms and clarify your options for care. The Buoy A.I. health assistant guides you on your way to well, the moment you feel sick. Retrieved January 18, 2021, from https://welcome.ai/tech/data-resourcesmanagement/buoy-health-symptom-checker

[4] Convolutional Neural Network. (2019, May 17). Retrieved January 18, 2021, from https://deepai.org/machine-learning-glossary-andterms/convolutional-neural-network

[5]] Collections Quick Download. (2001, April). Retrieved January 18, 2021, from

https://www.cancerimagingarchive.net/collections-quick-download/

[6] Halliday. (2015, December). Fundamental of physics. US: Wiley.

[7] Singh, G. A., & Gupta, P. K. (2018). Performance analysis of various machine learning-based approaches for detection and classification of lung cancer in humans. Neural Computing and Applications, 31(10), 6863-6877. doi:10.1007/s00521-018-3518-x