Faculty of Computers and Artificial Intelligence, Helwan university

Selected Topics in Computer Science 2

Team members: 6 members

|  |  |
| --- | --- |
| Name | ID |
| Gerges Hanna | 20180173 |
| Abanoub Kamal | 20180003 |
| Abanoub Magdy | 20180004 |
| Abanoub Rafaat | 20180002 |
| Andrew Emad | 20180140 |
| Kirollos Hany | 20180442 |

Under the supervision of Dr Wesam El-behaidy

# Paper Details:

* Paper name: Multi-class Classification for the Identification of COVID-19 in X-Ray Images Using Customized Efficient Neural Network
* Citation: Hussain, A., Imad, M., Khan, A., Ullah, B. (2022). Multi-class Classification for the Identification of COVID-19 in X-Ray Images Using Customized Efficient Neural Network. In: Boulouard, Z., Ouaissa, M., Ouaissa, M., El Himer, S. (eds) AI and IoT for Sustainable Development in Emerging Countries. Lecture Notes on Data Engineering and Communications Technologies, vol 105. Springer, Cham. <https://doi.org/10.1007/978-3-030-90618-4_23>
* Dataset: This paper used Chest X-ray images obtained from the open-source Kaggle repositories, and the dataset contains a mix of chest X-rays images (COVID19, normal and pneumonia). A total of 3798 images for the training and testing.
* This paper proposed CNN network architecture consists of multiple conventional layers (CONV),

subsampling layer (polling layer), and fully connected layers. The proposed CNN

model takes image is an input with (128, 128, 3) size. The proposed network consists

of four Blocks; each block contains convolutional and pooling layers. In the first

block, the convolutional layer having 32 filters with (3, 3) size and the same padding,

while the Maxpooling size is (2, 2). The second block contains one convolution layer

having 64 filters of size (3, 3) and the same padding, while the Maxpooling size is the

same. The third and Fourth blocks contain one convolution layer having 128 filters

of size (3, 3) with the same padding and max-pooling size (2, 2). The two fully connected layers (FC) using ReLU activation function. In contrast, the output layer has (1,7) classed and SoftMax activation functions

The next figure shows the architecture of the proposed model from the paper

Table

Description automatically generated

* The paper results are demostriated in the next table:

|  |  |
| --- | --- |
| **Model** | **Accuracy** |
| AlexNet | 90% |
| VGG16 | 94% |
| Proposed Model | 95% |

# Project Description

## Dataset:

Only fruits were used from [Fruits and Vegetables Image Recognition Dataset](https://www.kaggle.com/datasets/kritikseth/fruit-and-vegetable-image-recognition). Some augmentation algorithms were applied to each image. The dataset contains 7 classes, each class has a different number of sample.

|  |  |
| --- | --- |
| Class label | Number of samples |
| Apple | 26 |
| Banana | 29 |
| Grapes | 46 |
| Kiwi | 18 |
| Pear | 42 |
| Pineapple | 32 |
| Watermelon | 17 |
|  | Total: 210 |

Each sample has different width and height from the rest. This issue is solved in the preprocessing phase to be 128 pixels for width, 128 pixels for height and three channels for color (RGB).

The augmentation process is taking care of the different number of samples per class by applying various functions, such as horizontal and vertical flip, blurring, Gaussian noise, and some random rotation and crops. These functions enabled us to produce 300 samples per class led to the creation of 2100 images.

|  |  |
| --- | --- |
| Class label | Number of samples |
| Apple | 300 |
| Banana | 300 |
| Grapes | 300 |
| Kiwi | 300 |
| Pear | 300 |
| Pineapple | 300 |
| Watermelon | 300 |
|  | Total: 2100 |

## Implementation details:

The dataset has been divided into 60% training (1260 samples), 20% testing (420 samples) and 20% validation (420 samples). Stratify by the target labels with a random state equal to 42

The next block diagram shows all phases of the project from reading the dataset to testing:

Diagram

Description automatically generated

We used all hyperparameters that have been used in the paper, and applied some fine tuning to them to find the best parameters that reach the highest accureices. The model has been trained on 10 epochs to find these parameters. This does not mean we reached the best accuracy we could for this model. Training, validation and testing data have the same percentage each time the model has been trained. The input shape is (128, 128, 3, and the output length is equal to 7 as well. Accuracies are measured using cross-validation accuracy which is equal to three iterations.

The next table is a description of some of the hyperparameters.

|  |  |
| --- | --- |
| Hyperparameter | Description |
| n\_hidden | Number of flattened hidden layers |
| n\_neurons | Number of neurons in the single flatted hidden layer |
| Add\_dropout | Adding a dropout layer after each hidden layer |
| Add\_batch\_normalization | Adding batch normalization after each hidden layer |

We adjusted some hyperparameters to reach the best accuracies

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| N\_hidden | N\_neurons | Activation function | Batch normalization | Dropout | Optimizer | Learning rate | Accuracy train | Accuracy validation |
| 1 | 64 | Tanh | False | True | SGD | 0.0003 | 0.936111111 | 0.858730159 |
| 1 | 64 | Elu | True | True | SGD | 0.0001 | 0.987301587 | 0.861904762 |
| 2 | 32 | Relu | False | False | SGD | 0.0001 | 0.996031746 | 0.861904762 |
| 2 | 32 | Tanh | True | False | SGD | 0.0003 | 1 | 0.900793651 |
| 2 | 64 | Relu | False | True | Adam | 0.0003 | 0.93452381 | 0.906349206 |
| 2 | 32 | Elu | False | True | SGD | 0.0003 | 0.999206349 | 0.922222222 |
| 2 | 64 | Elu | False | False | Adam | 0.0001 | 1 | 0.926190476 |
| 2 | 32 | Elu | True | False | Adam | 0.0001 | 0.997222222 | 0.935714286 |
| 2 | 64 | Elu | True | True | Adamax | 0.0003 | 1 | 0.949206349 |
| 2 | 64 | Elu | True | False | Adam | 0.0001 | 1 | 0.962698413 |
| 1 | 32 | Relu | True | False | Adam | 0.001 | 1 | 0.9714 |

After finding the best parameters, We trained the model with these hyperparameters on 48 epochs to reach the accuracies that is demonstrated in the next table.

|  |  |
| --- | --- |
| N\_hidden | 1 |
| N\_neurons | 32 |
| Activation function | Relu |
| Batch normalization | True |
| Dropout | False |
| Optimizer | Adam |
| Learning rate | 0.001 |
| Epochs | 48 |
| Train accuracy | 100% |
| Validation accuracy | 98.095% |
| Test accuracy | 99.2857% |

The next two figures show the learning curve and explain that the model doesn’t suffer from overfitting due to the equality of the validation accuracy and loss with the train accuracy, and it doesn’t suffer from underfitting due to high values of the validation accuracy and train accuracy.

Chart

Description automatically generatedChart, histogram

Description automatically generated

The next figure shows the architecture for the best model:

Table

Description automatically generated