Character Recognition Using Pre-Trained Models and Performance Variants Based on Datasets Size: A Survey

Ali Benaissa[1,*], Abdelkhalak Bahri[2], Ahmed El Allaoui[3] and Youssef Bourass [4]
[1]Department of Mathematics and Computer Science, Data Sciences and Competitive Intelligence team (DSCI), Laboratory of applied Science, National School of Applied Science, Al Hoceima, Morocco.
[2]Department of Mathematics and Computer Science, Data Sciences and Competitive Intelligence team (DSCI), Laboratory of applied Science, National School of Applied Science, Al Hoceima, Morocco.
[3] Computer Science department faculty of science and technology Errachidia, Morocco.
[4]Computer Science, Signals, Automation and Cognitivism Laboratory Faculty of science, Dhar El Mahraz University Fez, Maroc.
[*]ali.benaissa.gry@gmail.com

Abstract— **The most efficient and beneficial mechanism to the feature of extracting data from an image, has been the Convolutional Neural Network (CNN) and it is used in many fields (Optical character recognition, image classification, object recognition and Facial recognition etc.). In this papier, we studied the character classification problems, using pre-trained models based on Convolutional Neural Network (CNN), and how the performance can change the outcome of dataset that is given. For that, we have used five pre-trained models' such as VGG16/19, ResNet, Xception et MobileNet. The experiment shows that Xception had the best performance rate compared to other models for all datasets, VGG16/19 performance rate are variants depend on dataset. However, Experiments shows that ResNet achieve the worst accuracy rate compared to other methods.**

Keywords—Character detection, Character recognition, Image classification, pre-trained models, Keras Library, Transfer Learning.

I. INTRODUCTION

Classification is an important process in an automatic image classification system [1], image classification in Artificial Intelligence became a challenging problem, as well as the field of image processing [2] which is one of the key research objectives, this is mainly due to the diversity of techniques and methods that are used in this context, either in classification or in object recognition [3].

Character classification [4] is the corner stone to extract a piece of information which we want to get from an image by processing it to a typed understandable form, and it has made a big impact in areas that want to step-ahead and go from written document to digitalized ones.

Nowadays, the Deep Learning algorithms [5] used in image classification have shown good results and high performance in a predictive accuracy, many novels Deep Learning based image classification algorithms has been highly developed as image classification steps into a brand-new era thanks to the sustained development of machine learning, such as k-Nearest Neighbor, Support Vector Machine, Convolutional Neural Network (CNN) [6] etc.

Convolutional Neural Network (CNN) has been recognized as the most powerful and effective mechanism for feature extraction, but traditional classifiers connected to CNN do not fully grasp the extracted features. Therefore, we will present a comparative direction between proposed solutions about the image classification problem using CNN.

As we present in this research paper we will discuss the use of the Deep Learning techniques, especially convolution neural network (CNN), will focus on pre-trained [7] models' application from Keras Library [8], it's a deep learning API programming in Python, that use TensorFlow [9] as a platform, it was developed for being fast and smooth in experimentation.

In the purpose to apply the Transfer Learning [10] on our three different datasets, the first has 26 classes (alphabetic from a to z in lowercase), the second and third has 62 classes (alphabetic from a to z in lowercase and uppercase and numbers from 0 to 9), they have different parameters of training, testing and evaluation. For this purpose, we have selected five modules from Keras Library (VGG16/19 [14], Xception [15], MobileNet [16] and ResNet [17]) and trained them on three datasets, we got mixed results, between 80% and 100% on VGGs, Xception, ResNet models, except for MobileNet model, the results were not as good as we expected, in fact, they were poor.

II. RELARED WORK

Convolutional Neural Network (CNN) automatically extract features from the image by constructing many different layers of CNN [20], that generate a feature hierarchy, The shallower frontal convolutional layer uses a smaller perceptual domain that allows learning of some local features of the image, and the deeper posterior convolutional layer uses a larger perceptual domain and can learn more abstract features (such as object size, position, directional directions and information).

A new CapsNet-based algorithm [21], [22] has recently been proposed, providing viable ideas for further refinement of the results. We believe CapsNet has the potential to achieve better performance by making some changes to the hyperparameters.

In their work [14], the Visual Geometry Group studied the effect of convolutional network depth on its accuracy in large-

scale image recognition setting. Their main contribution is a thorough evaluation of networks of increasing depth using an architecture with very small convolutional filters (3x3), showing that a significant improvement over prior art configurations can be obtained by increasing the depth at 16 - 19 layers of weight.

In their paper [15], Microsoft Research has shown that convolution and highly indivisible convolution lie at either end of a discrete spectrum, with the Inception module representing an intermediate point in between. These observations led us to propose replacing the Inception module with deep separable convolutions in the neural machine vision architecture. They presented a new architecture based on this idea, called Xception, which has a similar number of parameters to Inception V3[11], Xception shows small increases in sort performance for the ImageNet [12] dataset and large increases for the JFT dataset [13].

Google Inc. in [16] launched an efficient model called MobileNets for mobile applications and embedded vision, it's based on a simplified architecture that uses deep separable convolutions to build lightweight deep neural networks. They introduce two hyperparameters that efficiently balance latency and accuracy.

III. METHODS:

A. Networks:

1) VGG16/19:

The number 16 in the term VGG alludes to the deep neural network's 16 layers (VGGnet), This indicates that VGG16 is a large network with over 138 million parameters. Even by today's standards, it's a massive and impressive network. The simplicity of the VGGNet16 surface, on the other hand, is what makes the network more appealing. It may be argued that all its architecture comes together just by glancing at it.

A few convolution layers are followed by a pooling layer that decreases the height and breadth of the image. When it comes to the filter's numbers, we can employ to do our research, we have roughly 64 options, which we can expand them to around 128 and subsequently to 256. We can utilize 512 filters in the final levels, Figure shows the VGG16 architecture.

The VGG19 model (also known as VGGNet-19) is similar to the VGG16 except that it has 19 layers. The numbers "16" and "19" refer to the model's weight layers (convolutional layers). VGG19 contains three more convolutional layers than VGG16.

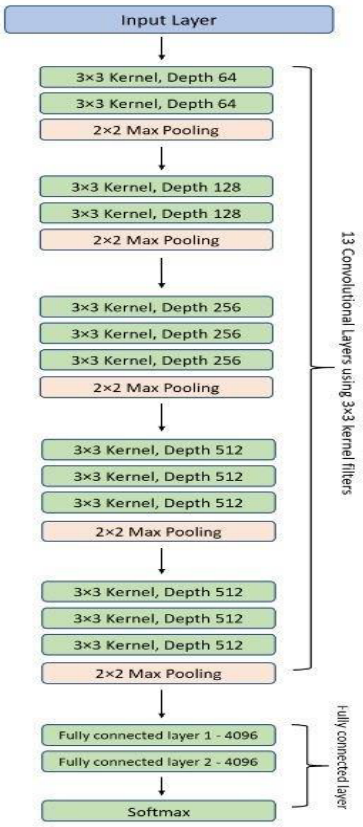

FIGURE I. VGG16 ARCHITECTURE.

2) MobileNet:

Convolution layers that are depth wise separable are used to construct MobileNets, a depth wise convolution and a pointwise convolution make up each depth wise separable convolution layer. MobileNet contains 28 layers if depth wise and pointwise convolutions are counted separately, the width multiplier hyperparameter can be adjusted to reduce the number of parameters in a conventional MobileNet to 4.2 million, the figure II detailes the MobileNet architecture.

Type / Stride	Filter Shape	Input Size
Conv / s2	3 × 3 × 3 × 32	224 × 224 × 3
Conv dw / s1	3 × 3 × 32 dw	112 × 112 × 32
Conv / s1	1 × 1 × 32 × 64	112 × 112 × 32
Conv dw / s2	3 × 3 × 64 dw	112 × 112 × 64
Conv / s1	1 × 1 × 64 × 128	56 × 56 × 64
Conv dw / s1	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 128	56 × 56 × 128
Conv dw / s2	3 × 3 × 128 dw	56 × 56 × 128
Conv / s1	1 × 1 × 128 × 256	28 × 28 × 128
Conv dw / s1	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 256	28 × 28 × 256
Conv dw / s2	3 × 3 × 256 dw	28 × 28 × 256
Conv / s1	1 × 1 × 256 × 512	14 × 14 × 256
5× Conv dw / s1	3 × 3 × 512 dw	14 × 14 × 512
5× Conv / s1	1 × 1 × 512 × 512	14 × 14 × 512
Conv dw / s2	3 × 3 × 512 dw	14 × 14 × 512
Conv / s1	1 × 1 × 512 × 1024	7 × 7 × 512
Conv dw / s2	3 × 3 × 1024 dw	7 × 7 × 1024
Conv / s1	1 × 1 × 1024 × 1024	7 × 7 × 1024
Avg Pool / s1	Pool 7 × 7	7 × 7 × 1024
FC / s1	1024 × 1000	1 × 1 × 1024
Softmax / s1	Classifier	1 × 1 × 1000

FIGURE II. MOBILENET ARCHITECTURE.

3) ResNet:

The essential innovation with ResNet, short for Residual Networks, was that it allowed us to train extraordinarily deep neural networks with 150+ layers effectively, The ResNet-50 model is divided into five stages, each with its own convolution and identity block. There are three convolution layers in each convolution block, and three convolution layers in each identity block. There are around 23 million trainable parameters in the ResNet-50, the Figure III detailed the ResNet-50 architecture.

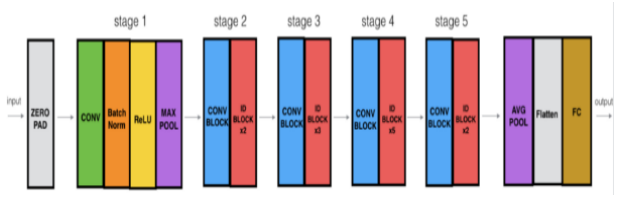

FIGURE III. RESNET-50 ARCHITECTURE.

4) Xception:

The term Xception refers to the extreme form of Inception. It's even better than Inception-v3 using a modified depth wise separable convolution, the modified depth wise separable convolution is known as SeparableConv. SeparableConvs are considered as Inception Modules and used throughout the deep learning architecture, all flows have residual (or shortcut/skip) connections, which were first proposed by ResNet. as can be shown in figure IV.

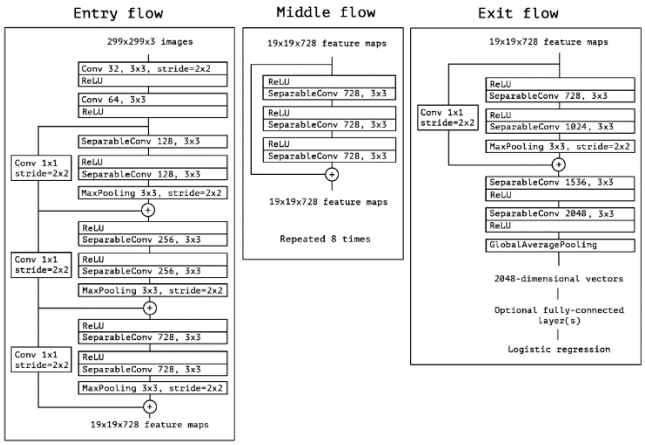

FIGURE IV. XCEPTION ARCHITECTURE.

B. Data augmentation:

We have used OpenCV library [23] to resize the images in our datasets, and pre-processing library from Keras, to prepare and processing the images in our system. In addition to this, we have used a normalization (of 255 as a float32 type), to get a total information from an image, and remove the distortions.

C. Hyperparameters:

In the used datasets, we split them to 80% of train and 20% of test, for the validation we had 80% from the train and 20% from the test, the following table shows the datasets details:

TABLE I. DATASETS PARAMETRS.

	Classes	Train	Test	Validation
Dataset 1	26	484	212	156
Dataset 2	62	1190	298	372
Dataset 3	62	41029	10258	12864

IV. EXPERIMENTS:

D. Datasets:

The first dataset has only alphabetics character in lowercase, with 26 classes, the second dataset has alphabetics character lowercase, uppercase and numbers from 0 to 9 with 64 classes and the third dataset has alphabetic characters and numbers with 64 classes, all datasets contains images with a white background and black character in the middle of the shape, for all models, we resize the images to 32x32 as an input shape parameter, except the Xception model we resize the images to 71x71, because its only accepts 71x71 input shape or higher, The figure V show a sample of dataset used.

FIGURE V. SAMPLE OF DATASETS

E. Dataset 1:

In this section, we will look for all models result on dataset 1, therefore, we regroup the results in a graph for being clear and simplify to read, the all models trained on 12 epochs.

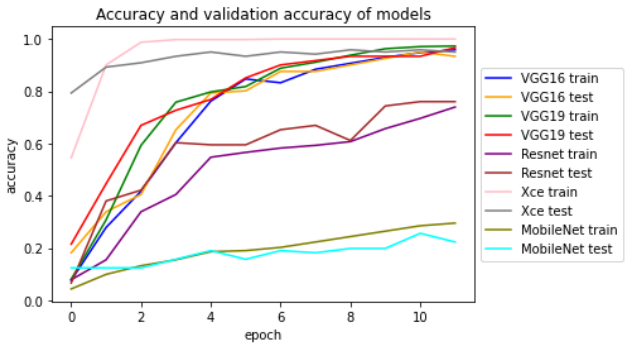

FIGURE VI. ACCURACY AND VALIDATION ACCURACY OF MODELS DATASET 1

In figure VI, Xception, VGG16 and VGG19 shows a high accuracy for train and test, and the longer the go stable they get, with a slight of overfitting [21], while, MobileNet and ResNet shows a bad result, where we see the validation accuracy (test) is wobbling.

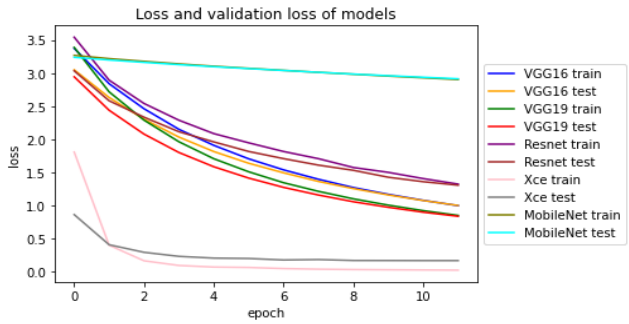

FIGURE VII. LOSS AND VALIDATION LOSS OF MODELS DATASET 2

Figure VII represent how loss (train) and validation loss (test) has changed during the 12 epochs, as we see, Xception achieve the best result in loss and validation loss, while MobileNet show the bad result, as for VGG16, VGG19 and ResNet the result was not bad as compared to the dataset size.

TABLE II MODELS SCORE DATASET 1

Score / Model	Accuracy for test images	Accuracy for validation images
VGG16	98.347 %	92.949 %
VGG19	99.174 %	93.59 %
ResNet	81.818 %	70.513 %
MobileNet	28.099 %	25.641 %
Xception	95.041 %	94.231 %

Table II present the score evaluate of the models, that shows VGG19 and Xception achieve the best results in comparison with the other models.

F. Dataset 2:

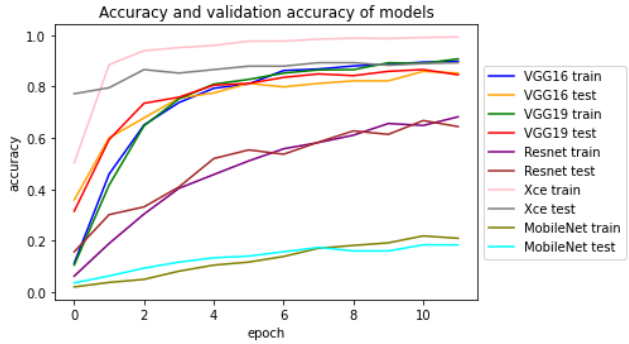

FIGURE VIII. ACCURACY AND VALIDATION ACCURACY OF MODELS DATASET 2

Figure VIII represent how accuracy (train) and validation accuracy (test) has changed during the 12 epochs, Xception, VGG16 and VGG19 shows a high accuracy for train and test, however, Xception shows less or low validation accuracy, compared to the one in dataset 1, and the longer the go stable they get, with a slight of overfitting, while, MobileNet and ResNet shows a unsatisfying result, where we see the validation accuracy (test) is low.

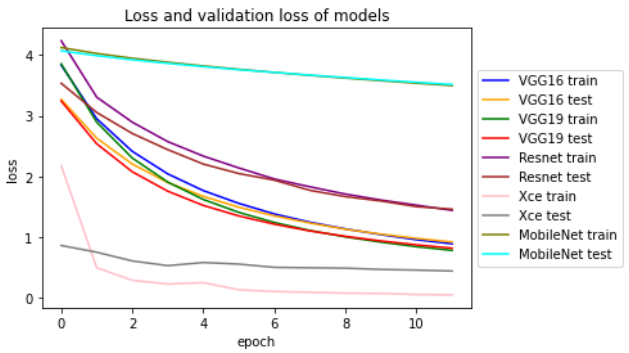

FIGURE IX. LOSS AND VALIDATION LOSS OF MODELS DATASET 2

Figure IX represent how loss (train) and validation loss (test) has changed during the 12 epochs, as we see, Xception achieve the best result in loss and validation loss, while MobileNet show the bad result, as for VGG16, VGG19 and ResNet the result was not bad compared to the dataset in a medium size.

TABLE III MODELS SCORE DATASET 2

Score / Model	Accuracy for test images	Accuracy for validation images
VGG16	88.591 %	92.949 %
VGG19	99.174 %	93.59 %
ResNet	81.818 %	70.513 %
MobileNet	28.099 %	25.641 %
Xception	89.262 %	93.28 %

Table III present the score evaluate of the models, that shows VGG 19 achieve the best results in comparison with the other models.

G. Dataset 3:

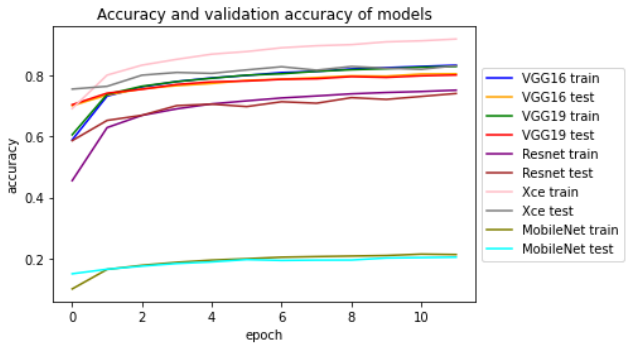

FIGURE X. ACCURACY AND VALIDATION ACCURACY OF MODELS DATASET 1.

Figure X represent how accuracy (train) and validation accuracy (test) has changed during the 12 epochs, Xception, VGG16 and VGG19 shows a high accuracy for train and test. However, Xception shows a little low validation accuracy, compared to the one in dataset 1, as we note that the overfitting is shrink compared to the results, we got in the sections before, and the longer the go stable they get, also we note that ResNet

achieve a medium performance, while, MobileNet shows a bad result, where we see the validation (train) and validation accuracy (test) is low.

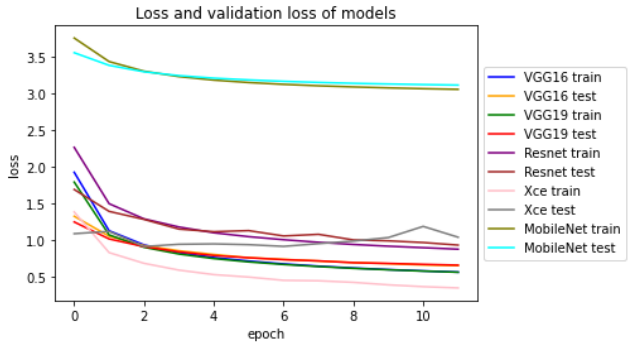

FIGURE XI. LOSS AND VALIDATION LOSS OF MODELS DATASET 3.

Figure XI represent how loss (train) and validation loss (test) has changed during the 12 epochs, Xception, VGG16 and VGG19 shows a low loss for train and test. However, Xception shows a little high validation loss, compared to the one in dataset 1, as we note that the overfitting is shrink compared to the results we got in the sections before, and the longer the go stable they get, also we note that ResNet achieve a medium loss, while, MobileNet shows a bad result, where we see the loss (train) and validation loss (test) is low.

TABLE IV. MODELS SCORE DATASET 3

Score / Model	Accuracy for test images	Accuracy for validation images
VGG16	98.347 %	92.949 %
VGG19	99.174 %	93.59 %
ResNet	81.818 %	70.513 %
MobileNet	28.099 %	25.641 %
Xception	95.041 %	94.231 %

H. Results:

To conclude the experiment section, we got a variants results expend on dataset size and the used model, we note that some models may perform better in a small dataset, but in the large ones also the performance might be low or medium, as example we have VGG19, some models can have a high performance only on the large dataset, as example we have VGG16, and the others can have a good performance on any size of dataset, as example we have Xception. This variant may due to the architecture of the models or the shape of images in the dataset. The modify of the layers that contain the models, using Fine-Tuning technique, can improve the model's performance.

V. CONCULSION:

As we presented in this paper, we compared between different pre-trained models' application from Keras Library, the result of Transfer Learning diverse from dataset to the other, depending on the datasets size. The efficiency of models in certain datasets didn't mean that they were efficient in all datasets.

The architect of model most focus on upcoming research, so it can adapt to any kind of dataset, using Fine-Tuning [18].

In future works, we plan to improve the character classification results, by combining several methods, and semantic information.

REFERENCES

[1] Pralhad Gavali ME, J. Saira Banu PhD, in Deep Learning and Parallel Computing Environment for Bioengineering Systems, 2019.

[2] Anna Syberfeldt, Fredrik Vuoluterä, Image Processing based on Deep Neural Networks for Detecting Quality Problems in Paper Bag Production, Procedia CIRP, Volume 93 ,2020, Pages 1224-1229, ISSN 2212-8271.

[3] Ajeet Ram Pathak, Manjusha Pandey, Siddharth Rautaray, Application of Deep Learning for Object Detection, Procedia Computer Science, Volume 132,2018, Pages 1706-1717, ISSN 1877-0509.

[4] Mayur Bhargab Bora, Dinthisrang Daimary, Khwairakpam Amitab, Debdatta Kandar, Handwritten Character Recognition from Images using CNN-ECOC, Procedia Computer Science, Volume 167, 2020, Pages 2403-2409, ISSN 1877-0509.

[5] Pin Wang, En Fan, Peng Wang, Comparative analysis of image classification algorithms based on traditional machine learning and deep learning, Pattern Recognition Letters, Volume 141, 2021, Pages 61-67, ISSN 0167-8655.

[6] Sakshi Indolia, Anil Kumar Goswami, S.P. Mishra, Pooja Asopa, Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach, Procedia Computer Science, Volume 132, 2018, Pages 679-688, ISSN 1877-0509.

[7] U.K. Lopes, J.F. Valiati, Pre-trained convolutional neural networks as feature extractors for tuberculosis detection, Computers in Biology and Medicine, Volume 89,2017, Pages 135-143, ISSN 0010-4825.

[8] Chollet, F. (2015) keras, GitHub. https://github.com/fchollet/keras

[9] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow,Andrew Harp, Geoffrey Irving, Michael Isard, Rafal Jozefowicz, Yangqing Jia, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Mike Schuster, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[10] J Praveen Gujjar, R Prasanna Kumar H, Niranjan N. Chiplunkar, Image Classification and Prediction using Transfer Learning in Colab Notebook, Global Transitions Proceedings, 2021, ISSN 2666-285X.

[11] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens. 2015. Rethinking the Inception Architecture for Computer Vision.

[12] Deng J, Dong W, Socher R, Li L-J, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. 2009. p. 248–55.

[13] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. In NIPS, 2014.

[14] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for large-scale Image Recognition.

[15] Franc¸ois Chollet. 2017. Xception: Deep Learning with Depthwise Separable Convolutions.

[16] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen. Google

[17] Inc. 2019. MobileNetV2: Inverted Residuals and Linear Bottlenecks.

[18] T. Alshalali and D. Josyula, "Fine-Tuning of Pre-Trained Deep Learning Models with Extreme Learning Machine," 2018 International Conference on Computational Science and Computational Intelligence (CSCI), 2018, pp. 469-473.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. Microsoft Research. 2015. Deep Residual Learning for Image Recognition.

[20] T. E. de Campos, B. R. Babu and M. Varma. Character recognition in natural images. In Proceedings of the International Conference on Computer Vision Theory and Applications (VISAPP), Lisbon, Portugal, February 2009.

[21] Qi Xu, Ming Zhang, Zonghua Gu, Gang Pan, Overfitting remedy by sparsifying regularization on fully-connected layers of CNNs, Neurocomputing, Volume 328, 2019, Pages 69-74, ISSN 0925-2312.

[22] B. Biswal, Geetha Pavani P, Prasanna T, Prakash Kumar karn, Robust segmentation of exudates from retinal surface using M-CapsNet via EM routing, Biomedical Signal Processing and Control, Volume 68, 2021, 102770, ISSN 1746-8094.

[23] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.