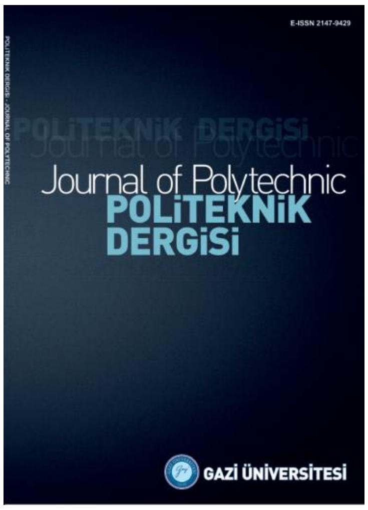


POLİTEKNİK DERGİSİ

JOURNAL of POLYTECHNIC

An experimental study for evaluating the performance of CNN pre-trained models in noisy environments

Önceden eğitilmiş cnn modellerin gürültülü ortamlarda görüntü sınıflandırması açısından değerlendirilmesi

***Yazar(lar) (Author(s))**: Halit BAKIR[1], Sefa Burhan EKER[2]*

ORCID[1]: 0000-0003-3327-2822

ORCID[2]: 0000-0003-0682-2016

To cite to this article: Bakır H. and Eker S. B., "An experimental study for evaluating the performance of CNN pre-trained models in noisy environments", ***Journal of Polytechnic***, 27(1): 355-369, (2024).

Bu makaleye şu şekilde atıfta bulunabilirsiniz: Bakır H. ve Eker S. B., "Önceden eğitilmiş cnn modellerin gürültülü ortamlarda görüntü sınıflandırması açısından değerlendirilmesi", ***Politeknik Dergisi***, 27(1): 355-369, (2024).

Erişim linki (To link to this article): http://dergipark.org.tr/politeknik/archive

DOI: 10.2339/politeknik.1162469

An Experimental Study for Evaluating the Performance of CNN Pre-Trained Models in Noisy Environments

Highlights

- ❖ *Testing the effects of noise on the performance of well-known CNN models.*
- ❖ *Multiple CNN models tested using images containing different Gaussian noise levels.*
- ❖ *The output of CNN models decreased dramatically based on the proportion of the noise.*
- ❖ *The well-known CNN models need some aid models to work properly in noisy environments.*

Graphical Abstract

Testing the efficiency of the well-known CNN pre-trained models in terms of classifying images in noisy environments.

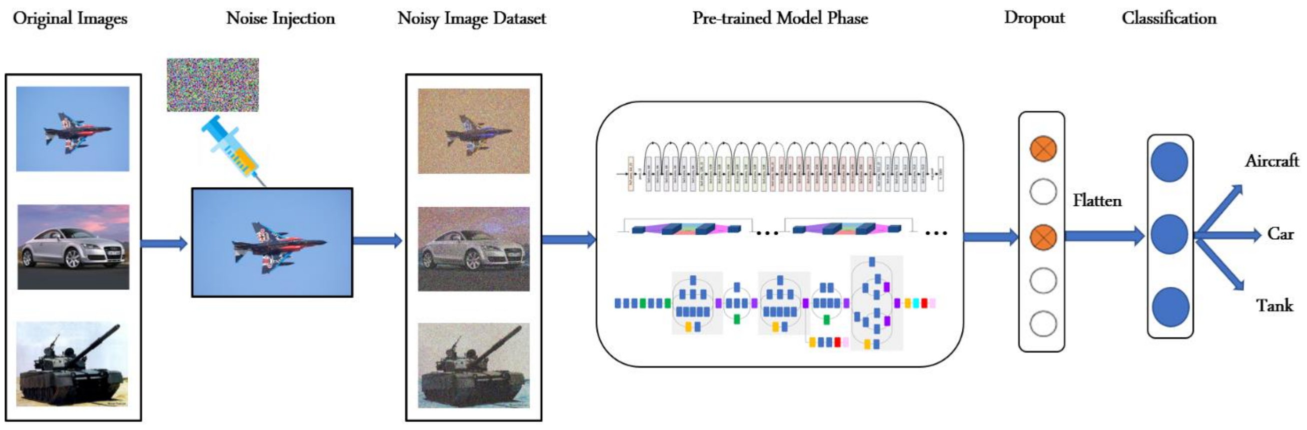


Figure. The scenario that was applied for testing the well-known CNN models

Aim

This work aims at testing the efficiency of the pre-trained models in terms of classifying images in noisy environments.

Design & Methodology

We proposed injecting Gaussian noise into the images in the used datasets gradually to see how the performance of that models can be affected by the proportion of the noise in the image. Afterward, three different case studies have been conducted for evaluating the performance of six different well-known pre-trained models namely MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception.

Originality

To the best of our knowledge, this is the first time that the effects of noise on well-known pre-trained CNN architectures have been comprehensively investigated with this number of considered models.

Findings

We noted that the classification accuracy and F1-score of the tested models dropped down dramatically by increasing the proportion of the injected noise. For example, while the classification accuracy of the tested MobileNet well-known model was 98.66% before adding the noise it dropped down to 35.33% after injecting 75% of Gaussian noise to the images of the datasets.

Conclusion

The obtained results showed that while these types of models can work very well in ideal environments their performances can drop down due to the conditions of the working environment, which reflects the need for some auxiliary models that should be used as a pre-processing phase to improve the performance of these models.

Declaration of Ethical Standards

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

An Experimental Study for Evaluating the Performance of CNN Pre-Trained Models in Noisy Environments

Halit BAKIR*[1], Sefa Burhan EKER[2]

[1] Department of Computer Engineering, Sivas University of Science and Technology, Sivas, Turkey

[2] Department of Aircraft Engineering, Sivas University of Science and Technology, Sivas, Turkey

(Geliş/Received : 15.08.2022 ; Kabul/Accepted : 03.04.2023 ; Erken Görünüm/Early View : 11.12.2023)

ABSTRACT

This work aims at testing the efficiency of the pre-trained models in terms of classifying images in noisy environments. To this end, we proposed injecting Gaussian noise into the images in the used datasets gradually to see how the performance of that models can be affected by the proportion of the noise in the image. Afterward, three different case studies have been conducted for evaluating the performance of six different well-known pre-trained models namely MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception. In the first case study, it has been proposed to train these models using a high-quality image dataset and test them using the same datasets after injecting their images with different levels of Gaussian noise. In the second case study, we proposed training the models using the created noisy image datasets in order to investigate how the training process can be affected by the noises in the environment. In the third case study, we proposed using the non-local means algorithm to denoise the images in the noisy datasets and testing the models trained using the original datasets using these de-noised image datasets. To the best of our knowledge, this is the first time that the effects of noise on well-known pre-trained CNN architectures have been comprehensively investigated with this number of considered models. The obtained results showed that while these types of models can work very well in ideal environments their performances can drop down due to the conditions of the working environment, which reflects the need for some auxiliary models that should be used as a pre-processing phase to improve the performance of these models.

Keywords: Pre-Trained model, gaussian noise, CNN, denoising techniques.

Önceden Eğitilmiş CNN Modellerin Gürültülü Ortamlarda Görüntü Sınıflandırması Açısından Değerlendirilmesi

ÖZ

Bu çalışma, önceden eğitilmiş CNN mimarilerinin gürültülü ortamlarda görüntüleri sınıflandırmadaki etkinliğini test etmeyi amaçlamaktadır. Bu amaçla, kullanılan veri kümelerindeki görüntülere kademeli olarak Gauss gürültüsü ekleyerek, bu modellerin performanslarının, görüntülerdeki gürültü oranından nasıl etkilenebileceğini göstermeyi hedefledik. Ardından, önceden eğitilmiş altı farklı CNN mimarisinin (MobileNet, ResNet, GoogleNet, EfficientNet, VGG19 ve Xception) performanslarını değerlendirmek için üç farklı vaka çalışması yapılmıştır. İlk vaka çalışmasında, bu mimarilerin yüksek kaliteli görüntü kümesi kullanılarak eğitilmesi, ardından aynı görüntülere farklı düzeylerde Gauss gürültüsünün enjekte edilmesi ve daha sonra gürültü içeren veri kümeleri kullanılarak bu mimarilerin test edilmesi önerilmiştir. İkinci vaka çalışmasında, CNN mimarilerindeki eğitim sürecinin ortamdaki gürültülerden nasıl etkilenebileceğini araştırmak için, oluşturulan gürültülü görüntü veri setleri kullanılarak modellerin eğitilmesi önerilmiştir. Üçüncü vaka çalışmasında ise, gürültülü veri kümelerindeki görüntülerin gürültüsünü gidermek için Non-local Means algoritmasının kullanması ve orijinal veri kümesi ile eğitilmiş modelleri, gürültüden arındırılmış veri kümeleri ile test edilmesi önerilmiştir. Bildiğimiz kadarıyla bu, önceden eğitilmiş CNN modelleri üzerinde gürültünün etkilerinin bu kadar fazla model ile deneysel olarak gösterildiği ilk çalışmadır. Elde edilen sonuçlar, bu tür modellerin ideal ortamlarda çok iyi çalışabilmelerine rağmen, gerçek hayattaki uygulamalarda çalışma ortamının koşulları nedeniyle model performanslarının düşebileceğini göstermiştir ki bu da gerçek hayattaki uygulamalarda bu modellerin performanslarını artırmak için bir ön işleme aşaması olarak kullanılması gereken bazı yardımcı modellere olan ihtiyacı göstermektedir.

Anahtar Kelimeler: Önceden eğitilmiş model, gauss gürültüsü, CNN, gürültü giderme teknikleri.

1. INTRODUCTION

In recent years, deep learning algorithms have come to the fore with their unique ability to conduct image classification and object detection tasks. Especially, convolutional neural network models play a very big role in image classification and detection domain. So, many successful CNN models such as VGGNet, AlexNet, ResNet, GoogleNet, MobileNet, have been proposed and made available as pre-trained models to be used in achieving other tasks. Although the success of these models is proven in the classification of high-quality

* *Sorumlu yazar(Corresponding Author)*
e-mail: halit.bakir@sivas.edu.tr

images, they may not give the same good results in low-quality or noise-containing images. And since the images collected in the desired application areas are not of that quality and they maybe contains different types of noises these types of pre-trained models should be tested under these conditions i.e. using poor-quality images. Noise in images may be caused by sensor limitations or may occur during the collection and compression of image data. In addition, environmental factors such as low light conditions, short exposures, multi-spectral shots, and using low-quality cameras can also cause image noising [1]. There are multiple noise types that can occur during collecting images in real-life applications due to the mentioned reasons, such as impulsive noise [2], additive noise, amplifier noise, speckle noise and salt-and-pepper noise etc [3].

In this study, the effect of Normal Gaussian noise on the performance of these well-known models has been investigated. To this end, we constructed an image dataset containing three classes by collecting the image from some image datasets available for free in Kaggle datasets repository. Then, we proposed to train and test the motioned models using the constructed high-quality original dataset images in order to test the classification performance of those models in a noise-free environment. After that, three different noise-injected image datasets have been constructed by injecting the normal Gauss noise with varied proportions, i.e. 25%, 50%, and 75%. We tested the performance of the previously trained models using the same images used during the training process but after adding some noise to that images. Results were surprising, while these well-known models are doing very well on the high-quality images, their performance dropped down madly after injecting a simple proportion of noise. So, it is concluded that these types of models may be insufficient for classifying images in real-life applications, such as radar systems, drone and aircraft goal detection systems, or medical fields.

2. RELATED WORK

In previous studies, different state-of-the-art techniques such as machine learning [4], artificial neural network [5], deep learning techniques [6], and convolutional neural networks [7], [8] have been used for image classification and detection. Also, some pre-trained models and transfer learning techniques have been adopted for this purpose in some other studies.

In [9], artificial neural network has been hybridized with classification tree for detecting fish in complex underwater environments. Particularly, the classification tree has been used as a feature extraction phase in order to obtain a reduced instance representing the collected dataset's images. The extracted features have been fed into the proposed artificial neural network model adopted as a classifier in the study. However, since each image in the dataset cannot be noise-free, it has become important to de-noise images containing noise before conducting the classification task. Therefore, the de-noising process has been conducted in many studies using classical filtering methods or CNN architectures. For example, in [10], linear and non-linear filters have been combined with an optimization algorithm called the cuckoo algorithm in order to determine the optimal filtering order for various types of noise. In [11], a deep neural network-based noise reduction method has been used to preserve image edges. To this end, an edge data set obtained using the Canny algorithm has been used. Then Shearlet Transform is used to eliminate the noise from the edges. In [12], the convolutional neural network has been used for nucleus detection and classification. Particularly, a new Neighbourhood Ensemble Estimator (NEP) algorithm has been used in conjunction with CNN to more accurately classify the detected cell nuclei. In [13], classification of brain tumour types such as meningioma, glioma, and pituitary has been conducted using AlexNet, GoogLeNet, and VGGNet. With this study, it has been proven that the transfer learning method is more effective than other methods for brain tumour classification. In [14], a transfer learning-based deep learning model has been used to classify patients with COVID-19 effects. It has been revealed from the obtained experimental results that the deep transfer learning-based Covid-19 classification is more efficient than other supervised models. Moreover, deep learning techniques have started to be used in the classification of agricultural plants and the detection of plant diseases. For example, in [15], some pre-trained models have been adopted and used as a feature extraction phase in order to extract efficient features used for training the Support Vector Machine classifier. Also, AlexNet, ResNet, GoogleNet, and VGGNet well-known pre-trained models have been used for classifying pests and insects that prevent the growth of plants such as sugarcane, rice, and maize [16]. It has been shown that such practices can be beneficial in order to be protected from these pests in the agricultural sector. In [17], the effect of Speckle noise, which is frequently seen in SAR images, on CNN models has been investigated. A method called DCC-CNNs (De-speckling and classification coupled CNNs) is used to distinguish multiple ground targets in SAR images containing strong and varying speckle noises. The proposed method achieved 82% of accuracy when it has been used for classifying 10 different ground targets. In [18], some CNN pre-trained model has been adopted as a feature extraction phase for a deep learning model used for diagnosis of cataract disease in the eye.

In [19], leaf classification has been conducted using some well-known CNN architectures such as AlexNet, Vgg16, Vgg19, ResNet50, GoogleNet and comparison study has been conducted. It has been observed that the success rate increased by increasing the number of iterations adopted for training the used pre-trained models. In [20],the researchers adopted extreme learning machine local receptive fields (ELM-LRF) method for conducting classification of brain tumor types.

It has been stated that the proposed approach outperformed ELM and AlexNet models. In [21], 3D ESA based ResNet50 architecture is proposed to classify hyperspectral images which includes high dimensions. This method was compared with 7 other architectures and it was seen that the proposed method gives good results as much as the other methods.

In some previous works, the effects of noise on performance of deep learning models (especially scratch ones) have been investigated. For example, in [22], the effect of the different types of image noises on a proposed CNN model has been investigated. The training data has been injected with Gaussian and S&P noises. Also, de-noising techniques have been used to see the effects of these techniques on the accuracy of the model. The results showed that training CNN models using noisy data could give good results to some extent. In [23], the effects of image degradations on CNN-based face recognition approaches has been investigated. In the results of the study, it has been stated that the performance of the models decreased significantly against image noise, blur, and occlusion. However, it has been stated that the CNN models were robust to color distortions and changes in color balance. In [24], the performance of a proposed CNN on noisy fish images has been analyzed. It has been stated that the CNN models trained using well-annotated images gave correct classification results on noisy images. In [25], Inception-v3 has been used to see the effect of image noise on the classification of skin lesions. The model has been evaluated using impulse noise, Gaussian noise, and a combination of these two noises. This study showed that the accuracy of the Inception v3 trained using the noisy skin lesion image dataset decreased compared with the original dataset-based Inception v3 model. In [26], it has been observed how the performance of CNN models is affected when trained medical disease images contain various levels of noise. To this end, the AlexNet CNN model was trained with noisy images and noise-free images, and the obtained results have been compared. It has been stated that the accuracy was high when the noise harmoniously covered the area of the disease and low when the noise did not cover the area of the disease.

In this paper a comprehensive study has been conducted for investigating the effects of noise on the well-known pretrained CNN models' performance. To this end, we proposed injecting different proportions of noise into a benchmark dataset and one more dataset constructed for this aim. We constructed multiple image noisy datasets and used them for testing the performance of multiple well-known CNN architectures including VGG 19, ResNet, InceptionNet (GoogleNet), MobileNet, EfficientNet and Xception. We conducted multiple case studies over these CNN architectures such as training them using the original datasets (Noise-free one) and testing them using noisy image datasets, training them using different noise proportions containing image dataset and test them using noise containing datasets, or denoising the images and testing the models using de-noised image datasets. When we investigated the literature, we observed that the effects of noise on well-known CNN architectures have been considered limitedly, and a few numbers of models have been evaluated such as Inception v3 and AlexNet. To the best of our knowledge this is the first study that comprehensively investigate the effects of noise on pre-trained CNN architectures with this number of tested models.

3. METHOD

Generally, the images collected in real-life applications are not always a high-quality and contain some type of noise due to the conditions of the environment and devices. The noise can be either additive or multiplicative format in the noisy images. In the additive noise model, an additional noise signal is added to the original signal to produce a noisy signal. In the multiplicative noise model, the original signal is multiplied by the noise signal for constructing the noisy one.

3.1. Noise Types

Noise in images can take various types such as **Impulse Noise, Speckle Noise, salt and pepper noise, Gaussian Noise, etc**. However, because that the Gaussian noise is the most common noise type, and it is similar to the noises that come across in nature life applications, this noise, i.e. Gaussian Noise, has been adopted in all conducted case studies. Gaussian noise, also called electronic noise (because it occurs in amplifiers or detectors), is a common noise type can be detected widely because that this type of noise is distributed based on the normal distribution probability function. An image containing Gaussian noise consists of the pixels of the original image in addition to the Gaussian noise pixels which is a probability distribution function spread out in the form of a bell.

3.2. Artificial intelligence (AI)

Artificial intelligence has revolutionized numerous domains across industries, significantly enhancing efficiency and productivity. In healthcare, AI-driven diagnostic tools can analyze medical images and patient data to provide quicker and more accurate diagnoses. In finance, AI algorithms are employed for fraud detection, risk assessment, and trading optimization. Moreover, AI's versatility extends to autonomous vehicles, natural language processing, Cybersecurity, data communication, customer service chatbots, and so many other applications [27]-[33]. Neural networks, whether classical or convolutional, represent a cornerstone of AI, driving advancements in image recognition, speech processing, and complex data analysis, making them one of the most pivotal techniques in the field.

3.2.1. Classical Neural Network

Artificial neural networks are a type of mathematical models mimics the structure of biological neural networks in the brain and its ability to learn and generalization. The learning process of the artificial neural network is based on feeding a big number of data samples that we know its real output into the network, and request the model to make a prediction on the fed data samples' output. After that, the predicted output is compared with the real output using some function and the weights of the network are adjusted using some optimization algorithm. This process is repeated until the predicted output will be very close to the real output.

An artificial neural network consists of 3 main layers i.e. input layer, one or more hidden layers, and the output layer. The data samples are fed into the input layer which passes them to the hidden layers after applying some processes. Each layer of the hidden layers extracts the most important features from the input and passes them to the next layer, the extracted features in each layer are called feature map. Finally, the output layer is responsible for making the last decision on the data passed to the model. There is something called weights located between the previous and next layers' neurons. The weights of the connections are very important for the final decision taken by the model. These weights are adjusted during the training phase of the network using a process called backpropagation.

3.3. Convolutional Neural Network

The convolutional neural network is a deep neural network technique mostly used and gives the best results in the image processing domain. The convolutional neural network reduces computational cost thanks to its working logic which is based on extracting feature maps describing the input image and reduce its dimensional. So, a descriptive feature map is extracted in each layer and passed to the next layer. The feature map extracted by the last convolution layer is converted to a vector and passed to a classical neural network architecture which responsible for taking the final decision about the input image. Any CNN model consists of 3 main layers, i.e. convolution layer, pooling layer, and fully connected layer, each of which will be described briefly.

3.3.1. Convolution Layer

The convolution layer is the basis block of the CNN model. Most of the mathematical calculations and the parameters to be trained in any CNN model exist in this layer. This layer contains filters to create feature map that summarizes the input images as low in dimensional but rich in information feature matrix. The weights that are wanted to be adjusted in this layer are some filters that are multiplied by the feature map contents come from the previous layers. The filter is shifted over the feature map and multiplied with its pixels in each location to create new feature map that will be passed to the next layer.

3.3.2. Pooling Layer

The pooling layer is also known as down-sampling layer since this layer applies a down-sampling operation to the feature map created in the previous convolutional layer. The pooling layer is implemented to reduce the dimensions of the feature maps, which reduces the number of parameters to learn and the total computational cost of the model.

3.3.3. Fully Connected Layer

Simply, this layer is a classical artificial neural network layer used for achieving the classification process and generating the output of the model. This layer is called fully connected since each neuron on it is connected to all neurons in the previous layer.

3.3.4. Dropout

Dropout is a regularization technique used for eliminate some nodes or links to some nodes in a specific layer of the neural network in order to mitigate overfitting problem.

3.3.5. Batch Normalization

This type of layers used to make the neural network training process more efficient, faster, and stable. Particularly, this layer used for normalizing the output of hidden layers in the neural network for reducing the number of epochs required and reducing generalization error of the CNN model.

3.4. Transfer Learning

The transfer learning is a method used for saving training time and effort. Particularly, this method adopts models trained to achieve a specific task to be used in conducting similar tasks. With another words, the adjusted weights of the pretrained models are used without conducting any training for these models which need a very advance hardware to be achieved.

3.5. Used Pre-Trained DL architectures

There are a lot of pre-trained models available for use, those have been used in this article will be briefly described in the next section.

3.5.1. GoogleNet (InceptionNet)

Basically, GoogleNet [34] or inceptionNet is a CNN model composed of multiple inception blocks. The idea of Inception block is using multiple filters or a bank of filters in a parallel manner rather than serial way. This means that the deep learning model will be a little bit wider rather than a lot of depth. Generally, each Inception block contains 1x1, 3x3 and 5x5 filter types in the same depth level. The outputs of these filters are stacked together to form an information rich features map used as input for the next block.

3.5.2. MobileNet

MobileNet is a model initially proposed to be used in platforms that have limited resources such as smartphone devices and embedded systems. This model is based on conducting the convolutional operation as two sub-operations such that the total overhead time of this operation is decreased. MobileNet, also gives very good results on normal computer platforms [35].

3.5.3. ResNet

Resnet stand for Residual Network [36], is one of today's successful and famous deep learning structures. Generally, it is difficult to train the neural networks when the number of layers is too much due to exploding/vanishing gradient and overfitting problems. So, to mitigate these problems ResNet introduces "Skip Connection" technique to skip over some unimportant or not used connections during the training process.

3.5.4. Xception (Extreme version of Inception) [37]

This model also has been proposed by Google, where a modified version of depthwise separable convolutional layer used in MobileNet model has been introduced. Particularly, while the depthwise convolution is followed by the pointwise convolution in the original depthwise convolutional block the pointwise convolution is used first followed by the depthwise convolution operation in the modified version of depthwise convolutional layer. There is another difference introduced by Xception model alongside with using modified version of depthwise separable convolutional layer, where, there is no intermediate ReLU non-linearity (activation function in the intermediate layers) in the Xception model.

3.5.5. VGG19

VGG is a commonly used neural network because it performs well, it was produced by a trusted team from a prestigious university, it was trained for weeks on a massive set of training data, it generalizes well to different use cases, and it was released to the public for free. Generally, there are two versions of VGG architectures namely VGG 16 and VGG 19. In this paper we will use the VGG 19 version of this architecture.

3.5.6. EfficientNet

EfficientNet is a convolutional neural network architecture and scaling method that uniformly scales all dimensions of depth/width/resolution using a compound coefficient [38]. EfficientNet, first introduced in 2019 is among the most efficient models that reaches State-of-the-Art accuracy on both imagenet and common image classification transfer learning tasks. The smallest base model is similar to MnasNet, which reached near-SOTA with a significantly smaller model. By introducing a heuristic way to scale the model, EfficientNet provides a family of models (B0 to B7) that represents a good combination of efficiency and accuracy on a variety of scales. Such a scaling heuristic allows the efficiency-oriented base model (B0) to surpass models at every scale, while avoiding extensive grid-search of hyperparameters.

3.6. Used Dataset

In this work, we used two different datasets for investigating the effects of noise on the performance of multiple well-known CNN architectures. The first dataset composed of three classes constructed for evaluating the effects of noise on the performance of deep learning models. The first dataset images have been collected from two different datasets available for free on Kaggle web site. We can note from the table 1 that the dataset is unbalanced where the first class contains 2015 images, the second class contains 1248 images, and the third class contains 1080 images. In order to balance the number of samples in the dataset's classes, 1000 images have been selected randomly from each class. So, the used first dataset contains 3000 images related to 3 classes. The first dataset will be called as Tank-Car-Aircraft dataset till the end of this paper. The second dataset is Mnist benchmark dataset. Mnist dataset contains a large number of handwritten digits. Particularly, Mnist dataset composed of 60,000 image samples as a training dataset and 10,000 image samples as a testing dataset.

Moreover, in order to conduct the desired evaluation study, six different noisy image datasets have been constructed from the two original image datasets. Particularly, different proportions of Gaussian noise i.e. 25%, 50%, and 75% have been injected into the images of the two original datasets. As a result, three datasets each of which contain 3000 noisy image samples, and three datasets each of which contains 70,000 noisy image samples have been constructed. The average of *Peak signal-to-noise ratio* (*PSNR*) has been calculated between the original and noisy images to show the impacts of the noise injection on the quality of the original images as illustrated in table 2.

Table 1. Used first image dataset

Class	# of samples
Fighter	1248
Car	2015
Tank	1080

Table 2. The average of Peak signal-to-noise ratio (PSNR) between the original and noisy images

Dataset	**PSNR for 25% Gaussian noise**	**PSNR for 50% Gaussian noise**	**PSNR for 75% Gaussian noise**
Tank-Car-Aircraft dataset	19.17	15.45	12.92
Mnist dataset	15.05	11.28	8.74

3.7. Proposed Test Scenario

The main goal of this study is to test the efficiency of the previously mentioned well-known pre-trained deep learning models in terms of classifying images in a noisy-environments. To this end, Gaussian noise has been injected at different proportions into the original dataset's images and three different datasets have been constructed. The purpose of using Gaussian is that this noise is equivalent to the noises that can be encountered in real life. To examine how the models, behave at different noise levels, 25%, 50% and 75% of Gaussian noise's proportion have been injected into the images of the original dataset. Python script has been prepared in order to achieve the noise injection process as illustrated in algorithm 1.

Some examples of gaussian noises injected at different proportions into the original images are given in the Figure 1.

We have selected six different well-known pre-trained convolutional neural network models to investigate the effects of Gaussian noise on deep learning models. Particularly, MobileNet, ResNet, VGG19, Xception, EfficientNet and GoogleNet have been selected for

Table 3. The average of Peak signal-to-noise ratio (PSNR) between the noisy and de-noised images

Dataset	PSNR for 25% Gaussian noise	PSNR for 50% Gaussian noise	PSNR for 75% Gaussian
Tank-Car-Aircraft dataset (With noise)	19.17	15.45	12.92
Tank-Car-Aircraft dataset (Without noise)	24.05	19.34	15.40

investigating the effects of noise on the pre-trained deep learning models. We conducted three different experimental study as follow:

3.7.1. First Case Study

In this case study the above-mentioned models have been trained using the original two datasets. With another words, in order to monitor the effects of the noise on deep learning models' performance, first of all we tried to evaluate the performance of the mentioned models by training them using the original image datasets. Then the trained models have been saved, desired noise proportion injected, and the models' classification performance have been evaluated using the noise-containing datasets.

3.7.2. Second Case Study

In this case study, we tried to train and test the above-mentioned models using noise-containing datasets in order to investigate the effects of the noise on the efficiency of the training process of the pre-trained deep learning models. So, the mentioned pre-trained models have been trained and tested six times using 30%, 50%, and 75% Gaussian noise-containing datasets.

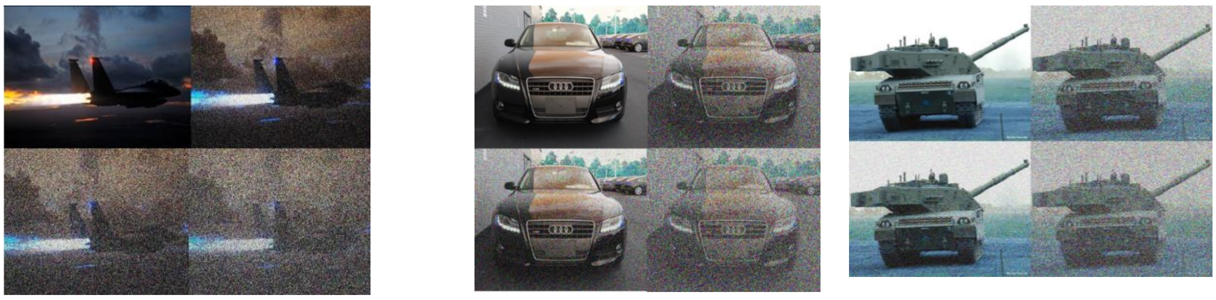
A dropout of 20% has been adopted after the pre-trained models to mitigate overfitting problem. The dataset has been splitted to be 10% as a validation dataset, 10% as a testing dataset, and the rest 80% as a training dataset. The figure 2 illustrates the proposed evaluation model.

4. EXPERIMENTAL RESULTS:

All experiments in this work have been conducted based on Google Colab GPU run-time. Multiple well-known Python libraries such as Open-CV, Numpy, Tensorflow, Keras, Tensorflow-hub, etc. have been used for applying the proposed model. Classification accuracy and F1-score have been used as metrics for evaluating the proposed model's performance.

4.1. First Case Study Results

First of all, each of MobileNet, ResNet, Xception, VGG19, EfficientNet and GoogleNet models have been trained and tested separately using the original images in the Tank-Car-Aircraft dataset. The results obtained using all the models were so promise. Particularly, the classification accuracy was 98.66%, 99.00%, 98.00%, 99.66%, 98.33%, and 97.33% for MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception



a. *Fighter class* b. *Car class* c. *Tank class*

Figure 1. Examples of dataset images after injecting 25%, 50% and 75% Gaussian noise.

3.7.3. Third Case Study

In this case study, we used the non-local means algorithm for denoising the noisy image datasets i.e. 25%, 50%, and 75% noise proportion containing datasets. Non-local means algorithm is based on replacing the value of a pixel with an average of selected other pixels' values: small patches centered on the other pixels are compared to the patch centered on the pixel of interest, and the average is performed only for pixels that have patches close to the current patch. After that, the pre-trained models trained using the original three-classes dataset have been tested using the three de-noised datasets. Table 3 illustrates the PSNR values for the three-class dataset before and after applying the de-noising process.

respectively. On the other hand, The F1-score of the first class was, 100%, 100%, 99%, 100%, 100%, and 99% for MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception respectively. The F1-score of the second class was 99%, 99%, 98%, 100%, 98%, and 97% for MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception respectively. And finally, The F1-score of the third class was 98%, 98%, 97%, 99%, 97%, and 97% for MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception respectively. So, all the pre-trained models achieved the classification task very efficiently. The results obtained using the original copy of Tank-Car-Aircraft dataset is illustrated in the Table 4. Furthermore, Figure 3 illustrates the confusion matrixes obtained by testing the proposed models using the original copy of Tank-Car-Aircraft image dataset (three classes). We can see from the confusion matrix of EfficientNet model that this model accurately predicted the class of all the images in the dataset except one image, which counted as a big success for any deep learning model.

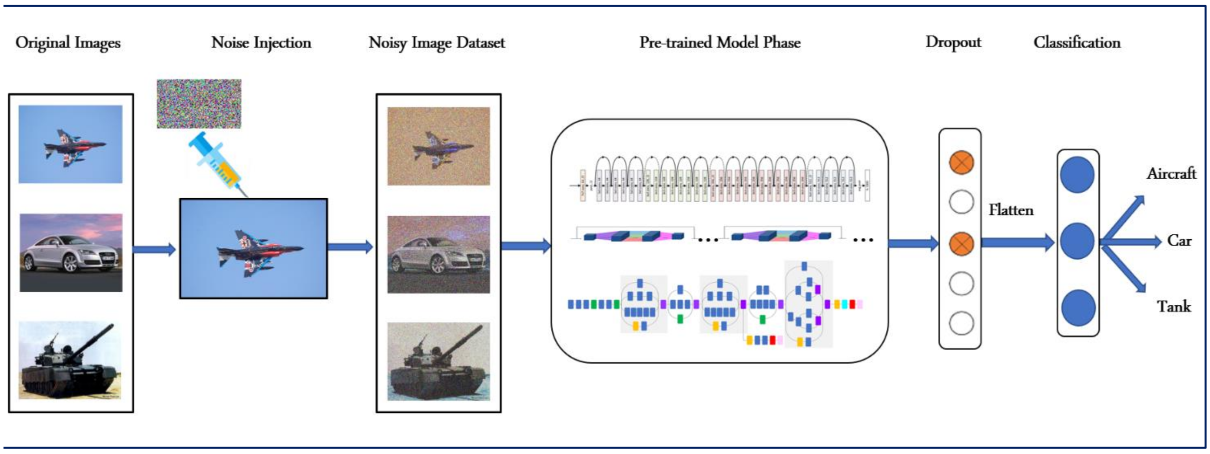


Figure 2. Applied testing scenario

Also, we can see that the MobileNet model could accurately predict the class of all the images in the dataset except 4 images. The same goes for ResNet and VGG 19 which predicted the class of the most images of the dataset except 3 images for ResNet and 5 for VGG19. The performance of GoogleNet and Xception was low to some extent compared with the other models, where, GoogleNet model could not predict 6 images correctly, and the Xception could not predict 8 images correctly. So, we can conclude from the first experiment conducted using the Tank-Car-Aircraft image dataset that the used pre-trained models can achieve classification tasks accurately, where the classification accuracy of all of the used models exceeded 98% and reached 99% using the ResNet and EfficientNet pre-trained models. On the other hand, when we trained the CNN architectures using the original copy of the Mnist dataset a high accuracy and F1-score values have been obtained from all the models where the values of accuracy and F1-score were 99% or 100% almost in all cases as illustrated in table 5. So, we can conclude that this type of well-known CNN architectures can give very high detection rates when trained and tested using high-quality images.

After that, the trained models have been saved to test their accuracy after injecting some noise into the images in the Tank-Car-Aircraft and MNIST datasets. Three different noise proportions have been injected into the images testing datasets i.e. the testing section of Tank-Car-Aircraft and MNIST. In other words, some noise similar to that exists in nature has been added to the images used for testing the models that have been trained using the two-original dataset. Firstly, the trained models have been evaluated using 25% Gaussian noise injected dataset. Although of the low proportion of the injected noise, the results have dropped out somewhat, especially the performance of MobileNet and Xception models. We can see from the Table 6 that the accuracy of MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception trained using the original copy of the Tank-Car-Aircraft dataset has dropped down into 86.33%, 95.99%, 97.33%, 98.31%, 97.11%, and 89.77% respectively. The performance of the tested models continued drop downing dramatically by increasing the proportion of the noise injected into the images. Particularly, by increasing the proportion of the noise into 50% the classification accuracy of MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception trained using the original copy of Tank-Car-Aircraft dataset decreased into 42.66%, 92.33%, 88.33%, 80.59%, 81.68%, and 25.49% respectively. Also, by increasing the proportion of the noise proportion into 75% the classification accuracy of MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception trained using the original copy of Tank-Car-Aircraft dataset decreased into 35.33%, 73.33%, 68.99%, 45.76%, 26.41%, and 14.13% respectively. So, we can note the big decrease in the performance by comparing the results of the used models before and after adding the desired noise proportions. The same things can be concluded from the confusion matrixes of the used pre-trained models illustrated in Figure 4.

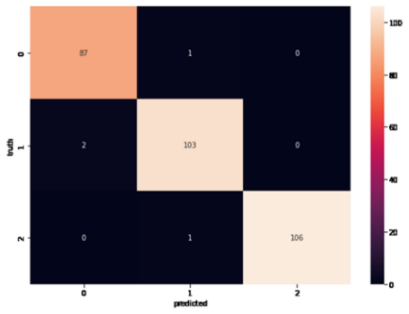
We can see that when the noise proportion reached 75% the MobileNet, EfficientNet, VGG19, and Xception pre-trained models almost saw all the images as if it is related to the same class. We can note that from the confusion matrixes of the tested models where the images of tank and aircraft classes have been classified as car class. This can be related to that the aircraft objects are so small, and they almost disappeared after injecting 75% of Gaussian noise into the images. The same thing goes for the tank images, where these types of images are similar to car objects and they can be detected as car after adding noise into images. This can be observed clearly in figure 1. Also, we can note the big decreasing in the performance of other used CNN architectures, for example, we can note the decrease in the performance of ResNet and GoogleNet pre-trained models where ResNet failed to correctly classify 80 data samples, and GoogleNet failed to correctly classify 93 data samples out of 300 samples included in the testing dataset. Furthermore, table 7 illustrates a comparison between the F1-scores of the models trained using the original copy of the Tank-Car-Aircraft dataset before and after adding the noise to the dataset. We can see from the table clearly the dramatically decrease in the F1-score of the models for all classes by increasing the proportion of the injected noise.

Table 4. Classification results obtained by training the models using the original copy of Tank-Car-Aircraft dataset.

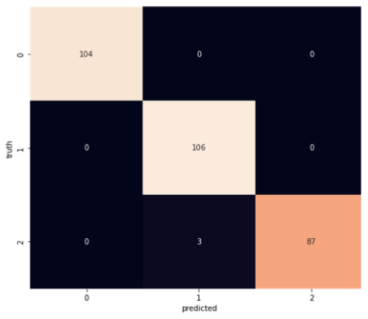
	Accuracy	**F1 score**		
		Class1	**Class2**	**Class3**
MobileNet	98.66%	100%	99%	98%
ResNet	99.00%	100%	99%	98%
GoogleNet	98.00%	99%	98%	97%
EfficientNet	99.66%	100%	100%	99%
VGG19	98.33%	100%	98%	97%
Xception	97.33%	99%	97%	97%

The same thing can be observed from table 8 where the CNN architectures trained using the Mnist original dataset have been used for classifying the Mnist test dataset after adding a different level of Gaussian noise into its images.

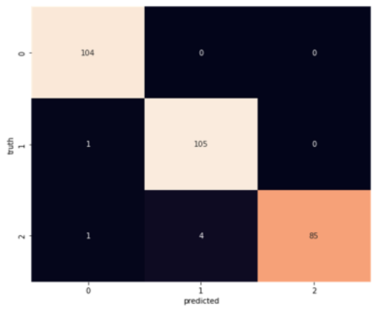
We can observe from this case study that all the CNN architectures can be affected by the noises caused by different noise resources in real-life applications. The Xception model was the weakest model against noise where its performance dropped down even after adding only a small proportion of noise into the images in the dataset.



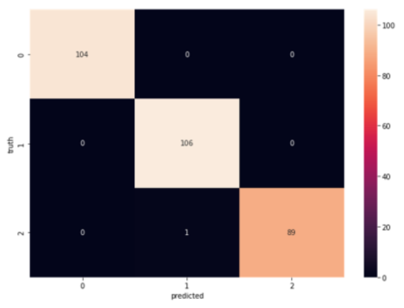
a. MobileNet Confusion Matrix



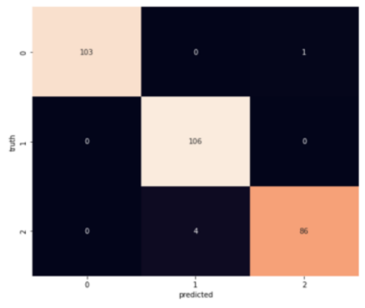
b. ResNet Confusion Matrix



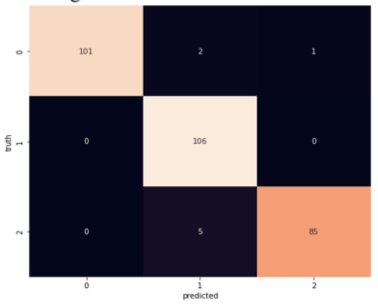
c. GoogleNet Confusion Matrix



d. EfficientNet Confusion Matrix



e. VGG19 Confusion Matrix



f. Xception Confusion Matrix

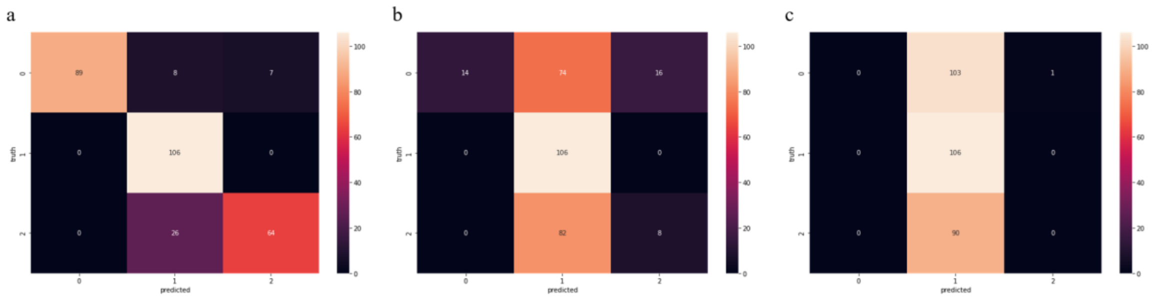
Figure 3. Confusion matrixes of the proposed model obtained using the first original dataset (three classes).

Second Case Study Results

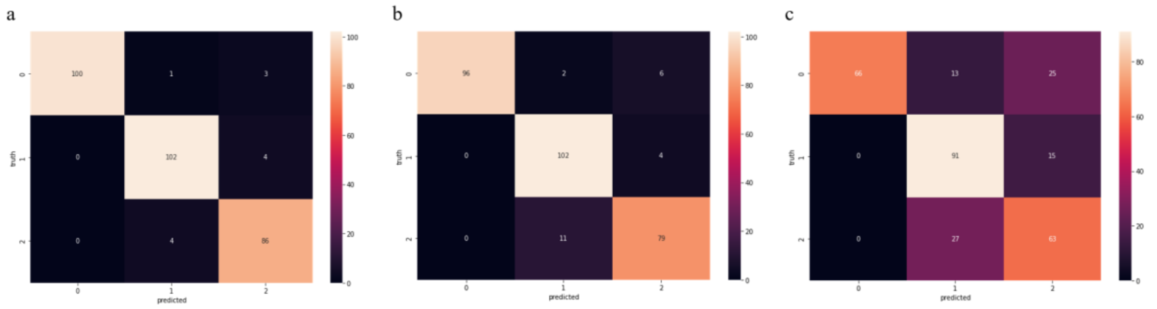
In the second case study, we proposed training the CNN architectures using the noise-containing image datasets in order to test if the model's training process can be affected by decreasing the quality of the image dataset. We will present the results obtained using this case study only for the constructed Tank-Car-Aircraft image dataset due to the limitation in space. The results obtained from this case study is illustrated in table 9. We can see from the obtained results that the performance of the models is decreased dramatically when the training dataset contains some noise. Also, it can be observed that the amount of decrease in the performance of models is proportional to the amount of noise injected into the images. Furthermore, the obtained results showed that MobileNet and Efficient models where the weakest architectures against Gaussian noise. Particularly, while the MobileNet model obtained 98.66% of classification accuracy when it has been trained using noise-free image dataset. The classification accuracy of MobileNet dropped down to 93.33%, 76.33%, and 68.00% after injecting 25%, 50%, and 75% of Gaussian noise into the training dataset's images. On the other hand, the obtained results showed that the classification accuracy of EfficientNet was 99.66% when it has been trained using high quality images. The classification accuracy of EfficientNet dropped down to 95.99%, 80.00%, and 65.66% after injecting 25%, 50%, and 75% of Gaussian noise into the training dataset's images. So, EfficientNet model and MobileNet model were the weakest models against this type of noise that come across in our life applications i.e. Gaussian noise.

4.2. Third Case Study Results

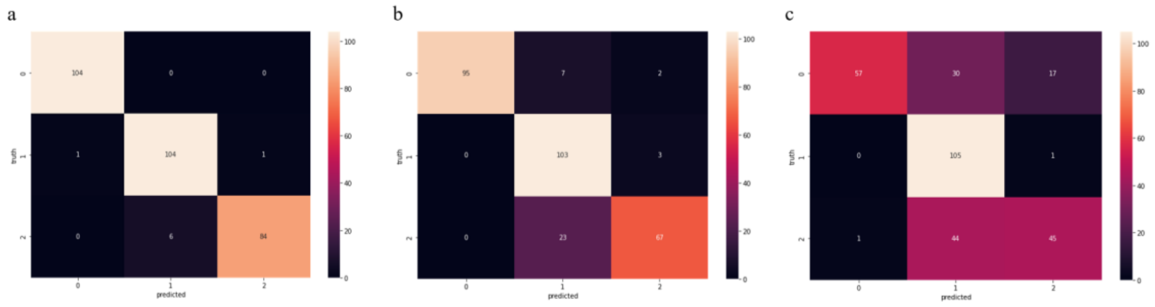
In this case study, the non-local means algorithm (NLM) has been adopted for denoising the noisy images of the datasets that contain 25%, 50%, and 75 % Gaussion noise. The NLM filter has shown remarkable performance in cancelling Gaussian noise [39]. In real-life applications, deep learning models can be trained using some quality images but after training the model and adopting it in the application domain the images collected from the environment maybe will contain some noise. Therefore, in this case study, we tried to simulate this scenario by training the model using the original image dataset, injecting some noise into the images, applying a denoising operation to the images, and then trying to detect them using the trained models. In other



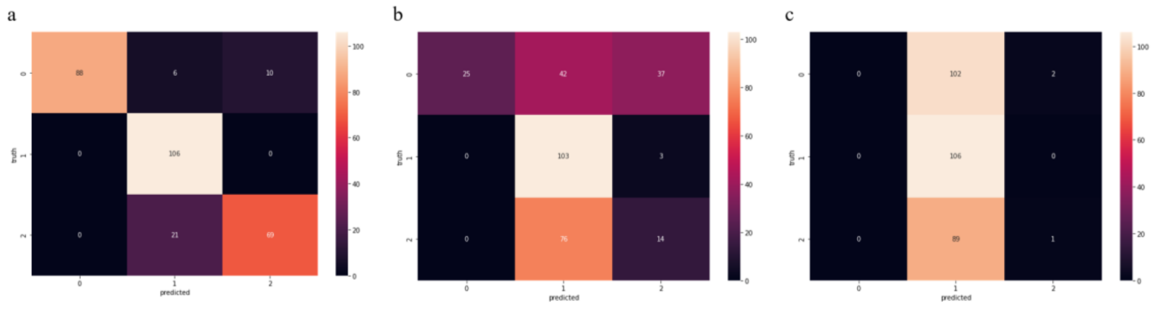
MobileNet model results for different noise proportions.



ResNet model results for different noise proportions.



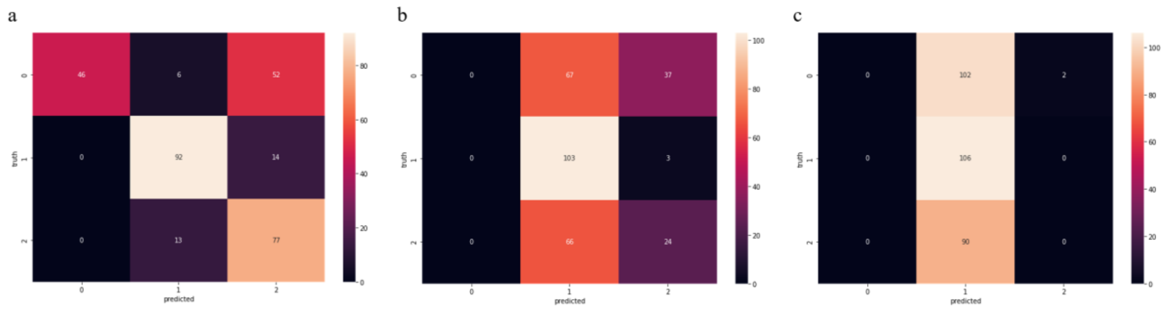
GoogleNet model results for different noise proportions.



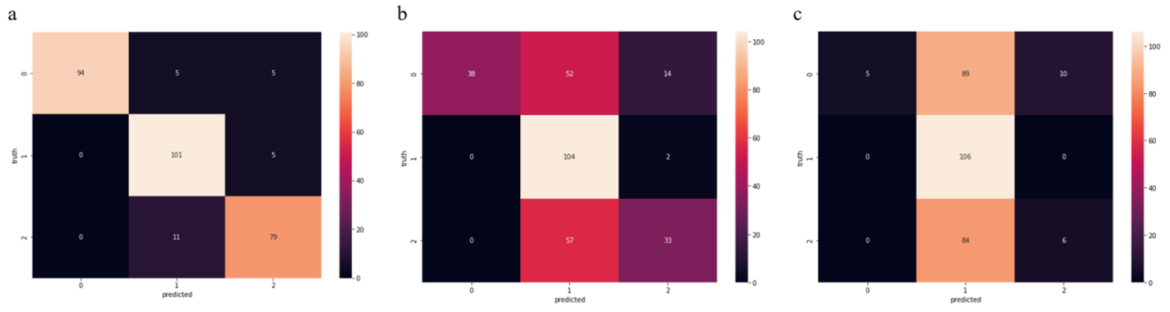
EfficientNet model results for different noise proportions.

Figure 4. The confusion matrix of the pretrained models in noisy version of Tank-Car-Aircraft dataset.

The noise proportion is 25%, b. The noise proportion is 50%, and c. The noise proportion is 75%.



VGG19 model results for different noise proportions.



Xception model results for different noise proportions.

Figure 4 (continue). The confusion matrix of the pretrained models in noisy version of Tank-Car-Aircraft dataset. The noise proportion is 25%, b. The noise proportion is 50%, and c. The noise proportion is 75%.

Table 5. Classification results obtained by training the models using the original copy of the MNIST benchmark dataset.

Model	**ACC**	**F1 score**									
		Classes									
		0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
MobileNet	98.48%	99%	100%	99%	99%	99%	99%	99%	96%	99%	97%
ResNet	98.63%	99%	99%	98%	99%	99%	98%	99%	98%	98%	98%
GoogleNet	98.87%	99%	100%	99%	99%	99%	97%	97%	99%	99%	99%
Efficient Net	99.28%	100%	99%	100%	100%	99%	99%	99%	99%	100%	99%
VGG19	99.27%	100%	99%	99%	99%	99%	99%	99%	99%	99%	99%
Xception	99.44%	100%	99%	100%	100%	99%	99%	99%	99%	100%	99%

Table 6. Comparing the classification accuracy of the models trained using Tank-Car-Aircraft original dataset and its noisy version.

		Original dataset	**25% Gaussian**	**50% Gaussian**	**75% Gaussian**
ACCURACY	**MobileNet**	98.66%	86.33%	42.66%	35.33%
	ResNet	99.00%	95.99 %	92.33%	73.33%
	GoogleNet	98.00%	97.33%	88.33%	68.99%
	EfficientNet	99.66%	98.31%	80.59%	45.76%
	VGG19	98.33%	97.11%	81.68%	26.41%
	Xception	97.33%	89.77%	25.49%	14.13%

words, we trained the model using the original Tank-Car-Aircraft dataset (without noise) and tested it using the noisy image datasets after applying some denoising operations on their images.

Aircraft dataset (without noise) and tested it using the noisy image datasets after applying some denoising operations on their images

some details from the images which causes this decrease in the performance. For example, the performance of MobileNet improved from 86.33%, 42.66%, and 35.33% when the model tested using 25%, 50%, and 75% noisy datasets respectively to 89.99%, 77.66%, and 56.99%

Table 7. Comparation between the F1-score of the tested models before and after injecting the noise to the images in the Tank-Car-Aircraft dataset.

Class	Model	Original	%25 Gaussian	%50 Gaussian	%75 Gaussian
Class1	MobileNet	100%	92%	24%	0%
	ResNet	100%	98%	96%	78%
	GoogleNet	99%	100%	95%	70%
	EfficientNet	100%	98%	39%	0%
	VGG19	100%	61%	0%	0%
	Xception	99%	91%	54%	9%
Class2	MobileNet	99%	86%	58%	52%
	ResNet	99%	96%	92%	77%
	GoogleNet	98%	96%	86%	74%
	EfficientNet	100%	90%	63%	53%
	VGG19	98%	85%	60%	52%
	Xception	97%	91%	65%	55%
Class3	MobileNet	98%	80%	14%	0%
	ResNet	98%	94%	88%	65%
	GoogleNet	97%	96%	83%	59%
	EfficientNet	99%	86%	19%	2%
	VGG19	97%	66%	31%	0%
	Xception	97%	88%	47%	11%

Table 8. Comparing the classification accuracy of the models trained using original MNIST dataset and tested using its noisy test dataset.

		Original dataset	25% Gaussian	50% Gaussian	75% Gaussian
ACCURACY	MobileNet	98.48%	94.96%	71.66%	47.63%
	ResNet	98.63%	74.71%	18.70%	10.19%
	GoogleNet	98.87%	98.48%	83.87%	53.43%
	EfficientNet	99.28%	98.31%	80.59%	45.76%
	VGG19	99.27%	98.90%	81.67%	26.41%
	Xception	99.44%	89.77%	25.49%	14.13%

It is can be observed from table 10 that a good improvement in the performance of all models has been recorded in most of the cases, especially when the amount of noise was 50% or 75%. Also, it can be noted that when the trained models tested using 25% noisy dataset the classification accuracy was better than when the denoising process has been conducted. This is related to that the injected amount of noise is very shallow so it cannot affect the performance of the adopted models, also, the adopted de-noising algorithm maybe delete

when the models tested using the same noisy datasets after applying de-noising algorithm to them. We can see also that the performance of GoogleNet and ResNet models decreased after applying the de-noise process, this maybe related to the fact that the denoising algorithm can delete some important features used by these models from the images. These results compatible with [33], where it has been stated that that NLM can lead to sub-optimal denoising performance when the destructive nature of noise generates some outliers inside patches.

Also, this indicates that the classical denoising algorithms can negatively affect the performance of these type of models. So, we will expand this case study in our future works and test different denoising algorithms for this purpose. Also, we will try to propose a new denoising algorithm for improving the performance of these types of models in different types of noisy environments.

Table 9. Classification results obtained by training the models using the Tank-Car-Aircraft dataset injected with different noise levels (second case study).

Noise proportion	Model	**Accuracy**	**F1 score**		
			Class1	**Class2**	**Class3**
0%	**MobileNet**	98.66%	100%	99%	98%
	ResNet	99.00%	100%	99%	98%
	GoogleNet	98.00%	99%	98%	97%
	EfficientNet	99.66%	100%	100%	99%
	VGG19	98.33%	100%	98%	97%
	Xception	97.33%	99%	97%	97%
25%	**MobileNet**	93.33%	96%	94%	89%
	ResNet	96.33%	98%	96%	95%
	GoogleNet	94.33%	97%	93%	92%
	EfficientNet	95.99%	100%	94%	93%
	VGG19	94.99%	98%	94%	93%
	Xception	95.33%	98%	94%	94%
50%	**MobileNet**	76.33%	81%	78%	68%
	ResNet	92.67%	96%	92%	90%
	GoogleNet	87.99%	89%	88%	87%
	EfficientNet	80.00%	88%	79%	71%
	VGG19	84.66%	87%	84%	82%
	Xception	90.33%	92%	92%	87%
75%	**MobileNet**	68.00%	73%	71%	56%
	ResNet	85.33%	90%	84%	81%
	GoogleNet	75.67%	83%	78%	59%
	EfficientNet	65.66%	74%	72%	39%
	VGG19	80.33%	87%	81%	72%
	Xception	80.66%	82%	85%	74%

Table 10. Classification results obtained by testing original Tank-Car-Aircraft trained models using denoised image datasets (Third case study).

Model	**Accuracy**						
	Original	**25% Gaussian Noise**		**50% Gaussian Noise**		**75% Gaussian Noise**	
		Noise	**De-Noise**	**Noise**	**De-Noise**	**Noise**	**De-Noise**
MobileNet	98.66%	86.33%	89.99%	42.66%	77.66%	35.33%	56.99%
ResNet	99.00%	95.99%	94.66%	92.33%	79.00%	73.33%	56.99%
GoogleNet	98.00%	97.33 %	94.33%	88.33%	80.00%	68.99%	62.33%
EfficientNet	99.66%	98.31%	93.99%	80.59%	80.00%	45.76%	78.66%
VGG19	98.33%	97.11%	90.33%	81.68%	84.66%	26.41%	80.00%
Xception	97.33%	89.77%	94.99%	25.49%	85.66%	14.13%	69.99%

Figure 5 illustrates some examples of images injected with different proportions of noises before and after applying the denoising process.

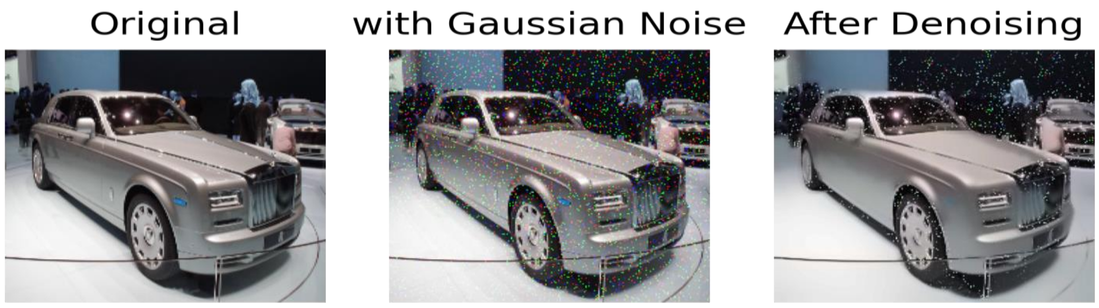
5. CONCLUSION

This study has proven that the well-known CNN architectures namely MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception maybe cannot working very well in real life applications where the processed images can contain a different type of noises related to the working conditions. To this end, we trained and tested these well-known CNN architectures using noise-free image dataset and they proof their efficiency when the image quality is high where the classification accuracy was 98.66%, 99.00%, 98.00%, 99.66%, 98.33%, and 97.33% for MobileNet, ResNet, GoogleNet, EfficientNet, VGG19, and Xception respectively. After that, we proposed injecting Gaussian noise, which a type of noise similar to natural noises, with different proportion i.e. 25%, 50%, and 75% into the images in the used datasets and monitoring the performance of these architectures to see how can affected.

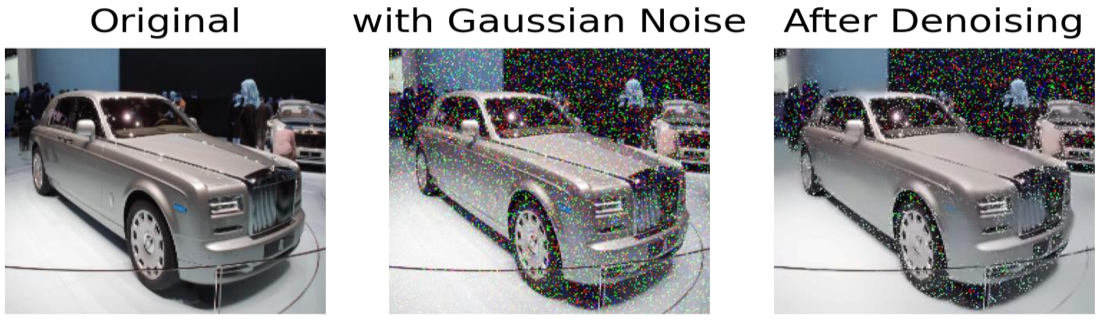
Particularly, we conducted three different case studies. In the first case study, we suggested training the above-mentioned CNN architectures using high quality images of the used two image datasets, and testing them using the image datasets injected with different proportions of Gaussian noise. We noted that the classification accuracy and F1-score of the tested models dropped down dramatically by increasing the proportion of the injected noise. For example, while the classification accuracy of the tested MobileNet well-known model was 98.66% before adding the noise it dropped down to 35.33% after injecting 75% of Gaussian noise to the images of the datasets. Also, MobileNet model's F1-score decreased dramatically by increasing the injected noise proportion and reached 0%, 52%, 0% for the first class, second class, and third class respectively, which means that the model almost classifies all the data samples of the dataset as if it is related to one class.

In the second case study, we proposed training the adopted well-known CNN models using the constructed noisy image datasets. It has been observed that the performance of models decreased in this case compared to that obtained by trained the CNN models using high quality images. For example, while the MobileNet model obtained 98.66% of classification accuracy when it has been trained using a noise-free image dataset, its accuracy dropped down to 93.33%, 76.33%, 68.00% when 25%, 50%, and 75 % of Gaussion noise have been injected into the images in the training dataset respectively.

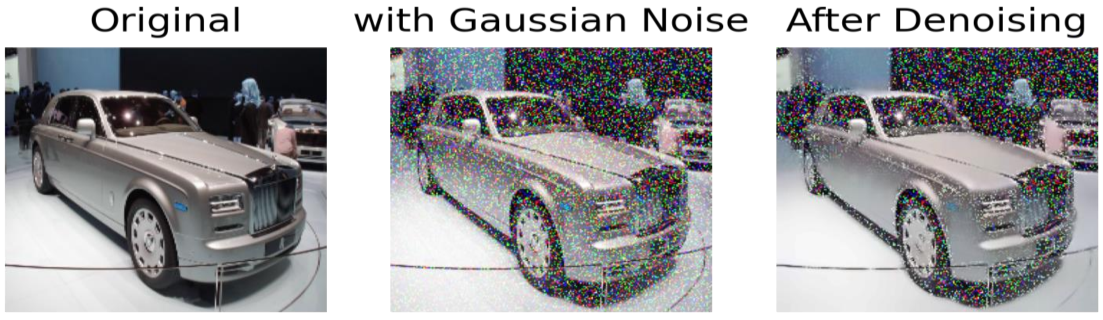
In the third case study, we proposed adopting a denoising algorithm and trying to eliminate the injected noise from the image, and training the CNN architectures using the original datasets and testing them using the denoised image datasets. The results showed that performance of models improved in the most of case compared with the



a. The ratio of Gaussian Noise injected to the original image is 25%



b. The ratio of Gaussian Noise injected to the original image is 50%



c. The ratio of Gaussian Noise injected to the original image is 75%

Figure 5. Examples of some images from the dataset in their original version, noise-contains version, and denoised version.

results of the first case study. For example, the performance of MobileNet improved from 86.33%, 42.66%, and 35.33% when the models have been tested using 25%, 50%, and 75% noisy datasets to 89.99%, 77.66%, and 56.99% respectively when the models have been tested using the same noisy datasets after applying de-noising techniques. On the other hand, we noted that the performance of some models such as GoogleNet and ResNet have been negatively affected by the denoising process. This may be related to that the denoising algorithm deleted some important features used by these models from the images, or maybe the denoising algorithm changed the values of image pixels to be different from the original values used for training the CNN models. Also, this may be related to that the applied classic denoising algorithm is not suitable for deleting the noise from the images in such a way that can be detected by the CNN architectures. So, in order to tackle this problem in our future works we will propose a deep learning-based denoising algorithm that is more suitable for deleting the noise so that the images will be detected more accurately by the CNN architectures.

Therefore, the results of this study showed that the performance of the pre-trained models, which can achieve a high classification performance in high quality images, degrade considerably when the images are exposed to Gaussian noise. Also, the results of this study show that the classical noise-denoising algorithms can cause the deletion of some important features from the images which can affect the performance of CNN architectures negatively. So, this reflecting the urgent need to auxiliary models or co-models that can be used as a pre-processing phase for these models in order to alleviate the effects of the noise in the images collected in real-life applications and increase the efficiency of this type of models. To this end, in our future works, the conducted study will be expanded by testing some other noise types such as Salt & Pepper noise, Speckle noise. etc. In addition, we will suggest auxiliary or pre-processing models for de-noise the images before entering the classification phase. Particularly, we will try some deep learning architectures such as Auto-encoder which proof their efficiency in this domain. So, we will tune this type of architectures such that it can eliminate the most famous types of noises.

ACKNOWLEDGMENT

This work has been supported by the Scientific Research Projects Coordination Unit of the Sivas University of Science and Technology. Project Number: 2023-GENL-Müh-0007.

AUTHORS’ CONTRIBUTIONS

Halit BAKIR: Shared in conducting experiments, Visualization, Review the manuscript and Supervision.

Sefa Burhan EKER: conducting the experiments, write manuscript draft, and wrote the manuscript.

DECLARATION OF ETHICAL STANDARDS

The author(s) of this article declare that the materials and methods used in this study do not require ethical committee permission and/or legal-special permission.

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

FINAL NOTE

This work has been conducted in the context of “Deep learning techniques (Derin öğrenme teknikleri)” lecture.

REFERENCES

[1] X. Lin, D. Bhattacharjee, M. el Helou, and S. Susstrunk, “Fidelity Estimation Improves Noisy-Image Classification with Pretrained Networks,” ***IEEE Signal Process Lett***, 28: 1719–1723, (2021).

[2] A. Awad, “Denoising images corrupted with impulse, Gaussian, or a mixture of impulse and Gaussian noise,” ***Engineering Science and Technology, an International Journal***, 22: 746–753, (2019).

[3] R. Rajni and A. Anutam, “Image Denoising Techniques - An Overview,” ***Int J Comput Appl***, 86: 13–17, (2014).

[4] M. Sheykhmousa, M. Mahdianpari, H. Ghanbari, F. Mohammadimanesh, P. Ghamisi, and S. Homayouni, “Support Vector Machine Versus Random Forest for Remote Sensing Image Classification: A Meta-Analysis and Systematic Review,” ***IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing***, 13: 6308–6325, (2020).

[5] M. Goyal, T. Knackstedt, S. Yan, and S. Hassanpour, “Artificial intelligence-based image classification methods for diagnosis of skin cancer: Challenges and opportunities,” ***Computers in Biology and Medicine***, 127: 104065, (2020).

[6] X. Zhou *et al.*, “A Comprehensive Review for Breast Histopathology Image Analysis Using Classical and Deep Neural Networks,” ***IEEE Access***, 8: 90931–90956, (2020).

[7] J. Naranjo-Torres, M. Mora, R. Hernández-García, R. J. Barrientos, C. Fredes, and A. Valenzuela, “A review of convolutional neural network applied to fruit image processing,” ***Applied Sciences (Switzerland)***, 10: 3443, (2020).

[8] L. Guo, “SAR image classification based on multi-feature fusion decision convolutional neural network,” ***IET Image Processing***, 16: 1–10, (2022).

[9] V. D. Jan Almero, E. Sybingco, and E. P. Dadios, “An Image Classifier for Underwater Fish Detection using Classification Tree-Artificial Neural Network Hybrid; An Image Classifier for Underwater Fish Detection using Classification Tree-Artificial Neural Network Hybrid,” ***In2020 RIVF international conference on computing and communication technologies (RIVF),*** 14: 1-6, (2020).

[10] M. Malik, F. Ahsan, and S. Mohsin, “Adaptive image denoising using cuckoo algorithm,” ***Soft comput***, 20: 925–938, (2016).

[11] H. R. Shahdoosti and Z. Rahemi, “Edge-preserving image denoising using a deep convolutional neural network,” ***Signal Processing***, 159: 20–32, (2019).

[12] K. Sirinukunwattana, S. E. A. Raza, Y. W. Tsang, D. R. J. Snead, I. A. Cree, and N. M. Rajpoot, “Locality Sensitive Deep Learning for Detection and Classification of Nuclei in Routine Colon Cancer Histology Images,” ***IEEE Trans Med Imaging***, 35: 1196–1206, (2016).

[13] A. Rehman, S. Naz, M. I. Razzak, F. Akram, and M. Imran, “A Deep Learning-Based Framework for Automatic Brain Tumors Classification Using Transfer Learning,” ***Circuits Syst Signal Process***, 39: 757–775, (2020).

[14] Y. Pathak, P. K. Shukla, A. Tiwari, S. Stalin, and S. Singh, “Deep Transfer Learning Based Classification Model for COVID-19 Disease,” ***IRBM***, 43: 87–92, (2022).

[15] V. K. Shrivastava and M. K. Pradhan, “Rice plant disease classification using color features: a machine learning paradigm,” ***Journal of Plant Pathology***, 103: 17–26, (2021).

[16] K. Thenmozhi and U. Srinivasulu Reddy, “Crop pest classification based on deep convolutional neural network and transfer learning,” ***Comput Electron Agric***, 164:104906, (2019).

[17] J. Wang, T. Zheng, P. Lei, and X. Bai, “Ground Target Classification in Noisy SAR Images Using Convolutional Neural Networks,” ***IEEE J Sel Top Appl Earth Obs Remote Sens***, 11: 4180–4192, (2018).

[18] Bakir H, Yilmaz Ş. “Using Transfer Learning Technique as a Feature Extraction Phase for Diagnosis of Cataract Disease in the Eye” ***International Journal of Sivas University of Science and Technology***, 1: 17–33, (2022).

[19] Doğan F, Türkoğlu İ. “Derin öğrenme algoritmalarının yaprak sınıflandırma başarımlarının karşılaştırılması” ***Sakarya University Journal of Computer and Information Sciences***, 1: 10–21, (2018).

[20] A. Ari and D. Hanbay, “Deep learning based brain tumor classification and detection system,” ***Turkish Journal of Electrical Engineering and Computer Sciences***, 26: 2275–2286, (2018).

[21] H. Firat and D. Hanbay, “3B ESA Tabanlı ResNet50 Kullanılarak Hiperspektral Görüntülerin Sınıflandırılması Classification of Hyperspectral Images Using 3D CNN Based ResNet50,” ***in 2021 29th Signal Processing and Communications Applications Conference (SIU)***, *Istanbul*, 6–9, (2021).

[22] T. S. Nazaré, G. B. Costa, W. A. Contato, and M. Ponti, “Deep convolutional neural networks and noisy images,” ***Iberoamerican Congress on Pattern Recognition***, Valparaíso, 416–424, (2017).

[23] S. Karahan, M. K. Yildirum, K. Kirtac, F. S. Rende, G. Butun, and H. K. Ekenel, “How image degradations affect deep CNN-based face recognition?,” ***in 2016 international conference of the biometrics special interest group (BIOSIG)***, IEEE, 1–5, (2016).

[24] A. Ali-Gombe, E. Elyan, and C. Jayne, “Fish classification in context of noisy images,” ***in International conference on engineering applications of neural networks***, Athens, 216–226, (2017).

[25] X. Fan *et al.*, “Effect of image noise on the classification of skin lesions using deep convolutional neural networks,” ***Tsinghua Sci Technol***, 25: 425–434, (2019).

[26] K. Sriwong, K. Kerdprasop, and N. Kerdprasop, “The Study of Noise Effect on CNN-Based Deep Learning from Medical Images,” ***Int J Mach Learn Comput***, 11: 202-207, (2021).

[27] Bakır, Halit, and Rezan Bakır. "DroidEncoder: Malware detection using auto-encoder based feature extractor and machine learning algorithms." ***Computers and Electrical Engineering*** 110: 108804, (2023).

[28] Bakır, Halit, Ayşe Nur Çayır, and Tuğba Selcen Navruz. "A comprehensive experimental study for analyzing the effects of data augmentation techniques on voice classification." ***Multimedia Tools and Applications***: 1-28, (2023).

[29] DURAN, Abdulmuttalip, and Halit BAKIR. "Hiperparametreleri Ayarlanmış Makine Öğrenimi Algoritmalarını Kullanarak Android Sistemlerde Kötü Amaçlı Yazılım Tespiti." ***International Journal of Sivas University of Science and Technology*** 2: 1-19, (2023).

[30] Özcan, Büşra, and Halit Bakır. "YAPAY ZEKA DESTEKLİ BEYİN GÖRÜNTÜLERİ ÜZERİNDE TÜMÖR TESPİTİ." ***In International Conference on Pioneer and Innovative Studies***, 1: 297-306, (2023).

[31] Doğan, E. R. O. L., and Halit BAKIR. "Hiperparemetreleri Ayarlanmış Makine Öğrenmesi Yöntemleri Kullanılarak Ağdaki Saldırıların Tespiti." ***In International Conference on Pioneer and Innovative Studies***, 1: 274-286, (2023).

[32] Demircioğlu, Ufuk, Asaf Sayil, and Halit Bakır. "Detecting Cutout Shape and Predicting Its Location in Sandwich Structures Using Free Vibration Analysis and Tuned Machine-Learning Algorithms." ***Arabian Journal for Science and Engineering***: 1-14, (2023).

[33] Bakır, Halit, and Kholoud Elmabruk. "Deep learning-based approach for detection of turbulence-induced distortions in free-space optical communication links." ***Physica Scripta,*** 98: 065521, (2023).

[34] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. “Going deeper with convolutions” ***In Proceedings of the IEEE conference on computer vision and pattern recognition***, IEEE, 1-9, (2015).

[35] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M, Adam H. “Mobilenets: Efficient convolutional neural networks for mobile vision applications”, ***arXiv preprint arXiv:1704.04861***, (2017).

[36] He K, Zhang X, Ren S, Sun J. “Deep residual learning for image recognition” ***In Proceedings of the IEEE conference on computer vision and pattern recognition***, IEEE, 770-778, (2016).

[37] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” **in *Proceedings of the IEEE conference on computer vision and pattern recognition,*** IEEE, 1251–1258, (2017).

[38] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” **in *International conference on machine learning***, PMLR, 6105–6114, (2019).

[39] Kazemi M, Menhaj MB. “A non-local means approach for Gaussian noise removal from images using a modified weighting kernel” ***arXiv preprint arXiv***:1612.01006, (2016)