A study of the effect of feature noise on different neural network algorithms

Alexandros Chantzaras
Athens, Greece
dit2122dsc@go.uop.gr

Alexandros-Konstantinos Vitsas
Athens, Greece
dit2103dsc@go.uop.gr

Georgios Spiliakos
Athens, Greece
dit2119dsc@go.uop.gr

Nikolaos Kyriakakis
Athens, Greece
dit2112dsc@go.uop.gr

***Abstract*—In recent years, various researchers have performed experiments and developed techniques to filter out or reduce feature noise in deep learning models. Moreover, different types of noise have been used as a strategy to generalize and create robust models. In this work, we have used statistical hypothesis-testing to evaluate how separate input noise types can affect different non pre-trained neural network algorithms concerning an image classification experiment.**

***Index Terms*—hypothesis-testing, input noise, neural networks, adversarial, ResNet18, SqueezeNet 1.0, AlexNet**

I. Introduction

Deep learning and deep neural networks have seen a fast and outstanding development growth, especially in the field of computer vision and object analysis [1]. Image data usually contains feature noise, largely due to the widespread transmission of image data over telecommunication networks [2].

The above is considered a widespread problem for researchers and a variety of algorithms and techniques are being examined and proposed to tackle this issue [3]. Quantifying the level of noise (i.e., determining the number of corroded pixels in an image) enables the recognition of a particular noise type and whether it could be additive or multiplicative [4].

Feature noise may affect a model's behaviour, yet it is sometimes appropriate to train models with noisy data. As such, those models are affected in different ways than those trained on clean input, alas this is not always feasible [5]. Nevertheless, the addition of noise during training is sometimes considered a good practice due to its regularization effect and the improvement of the robustness of the model [5].

Although we are more familiar with the effect of well-known types of noise such as Speckle [6], Salt & Pepper [7] or Gaussian [8], there are kinds of noise which could be extremely harmful for a neural network's predictive capabilities. Adversarial noise and adversarial attacks are such an example, and currently pose as a major obstacle in the widespread use of neural networks in many aspects of daily life, as well as in more sophisticated scenarios such as the security management of an organization [9].

II. Related Work

The addition of random noise in data during the training process of a neural network has been shown to improve it's generalization ability and to prevent the learning process from common pitfalls, such as the entrapment in local minima [4]. Contrary to the previous statement, other studies which have been conducted, identified that poor image quality due to noise, can hinder classification performance in systems that employ neural networks [5].

In general, noisy data could render classification difficult, with this observation stemming from the comparison of performance between networks that have been tested on clean input, and those which have been trained on noisy or restored data [5]. As [7] suggests, the use of denoising methods improved the performance of the tested neural networks when such a technique was applied.

In real world applications though, networks trained with noisy images have an advantage when dealing with noise in future data, even when it occurs at a different level [5]. Moreover networks trained on varying noise levels seem to improve their resilience to other types of noise as well.

Despite neural networks' outstanding performance on a variety of image processing tasks, deep neural networks have been shown to be vulnerable to adversarial samples. In [6], four different architectures of deep neural networks, Caffe Reference Model, VGG16, VGG-CNN-S and GoogleNet, were examined in terms of their susceptibility to blur noise. The concluding results were that even for moderate blur levels, the accuracy of all networks decreased significantly and was attributed to the removal of important textures.

Adversarial examples have been of high interest, however it seems unlikely to be encountered in most real world applications [6].

III. Noise Types and Neural Networks

A. Gaussian

Gaussian noise is statistical noise that has a probability density function equal to that of the normal distribution, also known as Gaussian Distribution [10].

In terms of image processing, what this means is that if we were to acquire the image of the scene repeatedly, we would find that the intensity values at each pixel fluctuate, so that we get a distribution of pixel values centered on the actual intensity value for that pixel [11].

The magnitude of Gaussian noise depends, and in fact is directly proportional to the standard deviation (sigma). Larger values of sigma indicate a larger amount of noise in the image.

B. Speckle

Speckle noise is a kind of noise that arises due to the effect of environmental conditions on the imaging sensor during image acquisition [12]. It is usually found in coherent imaging systems such as Synthetic aperture radar images (SAR), tomography, ultrasound medical images and in any laser-based imaging system. In general, the final image is formed by processing backscatter returns from radar pulses. This can result in a significant variation regarding pixel intensity, due to the illumination perceived by surfaces with roughness, thus creating this granular interference known as speckle.

There are many applications developed to reduce speckle noise, mostly focused on two basic categories: optical and numerical. Optical solutions concentrate on light coherence reduction during image capturing and numerical solutions focus on filtering methods [13] [14].

C. Salt and Pepper

Salt and Pepper is a sub-category of *Impulse* noise and is added to an image by addition of both random bright (with 255 pixel value) called *Salt* noise and random dark (with 0 pixel value) called *Pepper* noise all over the image [15].

Impulse noise occurs due to the quick transitions such as faulty switching, and can be caused by dead pixels, or due to analog-to-digital conversion errors, or bit errors in the transmission, etc [2]

This type of noise taints the quality of the image significantly, leading to difficulties in succeeding image processing tasks such as image recognition and edge detection [16].

The are several types of filters which can tackle the salt pepper type of noise. The **Max** filter is useful for finding the brightest points in an image and tackle the pepper noise. It uses the maximum intensity value in a sub image area. This filter reduces the pepper noise because it has very low values of intensities [10].

On the other hand, the **Min** filter is used to determine the darkest point in an image by using the minimum intensity value in a sub image area and finally eliminates the salt noise of an image [10].

Furthermore, two more types of filters can also be encountered which can tackle the salt pepper noise as a whole, with them being the **Mean** filter and the **Median** filter [10].

D. Random Erasing

Another interesting approach which was included in our experiment is a data augmentation technique called Random Erasing [17].

Random Erasing states that by randomly selecting a rectangle within an image and alternating the pixel values bound by this rectangle, we can achieve improvements in classification, object detection and person re-identification. This technique provides robustness to occlusion and enhances the generalization ability of CNN based models.

The rationale of using the random erasing technique in our experiment was to produce images with obstructions (similarly to the rest of noise types) and observe the effect over the chosen neural networks.

E. Adversarial Noise

Adversarial noise is a special type of feature noise, created with the end goal of confusing a neural network [18]. This results in a high number of miss-classifications of the given input, yet the latter is usually identical to the human eye.

Feeding adversarial images during testing to a neural network is called an adversarial attack [18]. There exists a plethora of such attacks, but in most cases the plan is to find small perturbations to add to the original input, resulting in its labeling as a different class.

Adversarial attacks are categorized in *white box* and *black box* attacks. White box attacks assume that the attacker has total access to the targeted model's weights [19].

Black box attacks make no assumptions regarding the target model and are the ones considered more dangerous, since they use more sophisticated methods to overcome the lack of information about the targeted system [19].

Regardless of the above, attacks can also be classified as *non targeted* or *targeted* ones. Untargeted attacks aim to distort the image in a way such that it is simply miss-classified, whereas targeted attacks drive the input to a specific class and tend to pose much more serious threats [19].

In this work, we will cover two of the most famous adversarial attacks, namely fast gradient sign method (FGSM) and projected gradient descent (PGD) [18] [20]. Both of these methods are considered white box attacks and our implementation of them focuses on the untargeted versions.

Fast gradient sign method uses the gradients of the neural network in order to craft an adversarial example. Instead of minimizing the loss w.r.t the parameters of the network, we aim to maximize the loss by adjusting the gradients w.r.t the input image.

$$x^{'} = x + \epsilon * \nabla_x sign(J(\theta, x, y))$$

Projected gradient descent is a more sophisticated and successful adversarial attack than the simpler but faster FGSM counterpart. The main idea behind this attack is to utilize the gradients w.r.t the input image as in the previous method, but this time when a new point x' is produced, the idea is to project it on the boundary of the classification space.

This results in finding smaller perturbations which will still affect the network performance without distorting the original image as much. Essentially, this algorithm tries to find as small perturbations as possible, so the adversarial image resembles the original as much as possible. This procedure is repeated multiple times as PGD is an iterative algorithm.

$$x^{'} = x + \epsilon * \nabla_x sign(J(\theta, x, y)) \rightarrow x^{''} = project(x^{'}, x)$$

F. ResNet18

ResNet18 is a variation of residual networks containing 18 deep layers. The purpose of residual networks is aimed towards nullifying the problem of vanishing gradients. The latter originates from the procedure of the gradient descent, with the end goal of finding optimal weights. Since the network has many layers, multiplication can result in gradient's values being diminished until vanished, resulting in decreased or even negative performance. The approach of ResNet is to introduce shortcut connections between layers, setting up a different shortcut for the gradient to pass through and also enable identity function which ensures that higher layers on the model hierarchy do not perform worse than lower layers. [21]

G. AlexNet

AlexNet is a convolutional neural network which was used to solve the image classification problem of ImageNet and was introduced in 2012 [22]. It consists of 5 convolutional layers, 3 max-pooling layers and 3 fully connected layers and its final output is fed to 1000-way softmax corresponding to the 1000 classes of ImageNet. The convolution layers consist of convolutional filters and the ReLU non-linear activation function.The max-pooling layers are used to reduce the dimensionality by allowing assumptions to be made about features contained in the sub-regions binned. The overall architecture of the network has 60 million parameters and data augmentation and dropout techniques were used in order to avoid over-fitting.

H. SqueezeNet 1.0

SqueezeNet's architecture fundamental motivation was to ensure high level of accuracy but with a smaller number of parameters, making the model easier to fit into computer memory as well as transmitted over a network. The main idea behind this type of network is based on the Fire Modules. These are comprised of a squeeze convolution layer of 1x1 filter which is fed to an expand layer, a mix of 1x1 and 3x3 convolution filters, providing a reduced number of connections, therefore resulting in a reduced number of total parameters. The number of filters is gradually increased towards the end of the network, while the learning rate is gradually decreased throughout the training process. [23]

Fig. 1. Starting from left to right, the original image is displayed first, then the images with Gaussian, Speckle, Salt & Pepper and Random Erasing respectively.

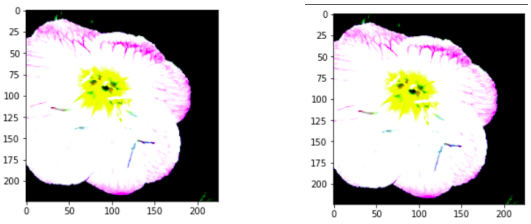

Fig. 2. Original image (left), adversarial image with PGD (right).

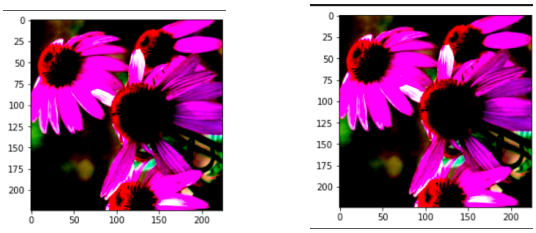

Fig. 3. Original image (left), adversarial image with FGSM (right).

IV. Methodology

A. Research Questions

- What is the effect of various noise types, regarding the accuracy score for an image classification task, when used as input in three different neural network topologies, namely AlexNet, ResNet-18 and SqueezeNet-1.0.
- Is a network trained with a specific type of adversarial noise also effective in mitigating the negative effects of an adversarial attack of a different type?
- Does adversarial noise always decrease the accuracy of a neural network?
- Does adversarial noise lead to an effective way of defending against adversarial attacks when used in the training of neural networks?
- Does random erasing help in the generalization and robustness of neural networks, regardless of the their underlying topology?
- How does training with Gaussian input noise affect the performance of AlexNet and SqueezeNet 1.0 under FGSM adversarial attacks?

B. Hypothesis Testing

In order to give answers to the above questions, statistical hypothesis-testing methods were used [24]. The distributions of samples that were used to perform the tests were produced experimentally. The experiments were designed in order to produce samples of the accuracy scores in a classification task of three different convolutional neural network architectures (AlexNet [25], ResNet-18 [26], SqueezeNet-1.0 [27]).

The utilized networks were trained with images from the *Oxford 102 Flower* available at https://www.robots.ox.ac.uk/~vgg/data/flowers/102/. After splitting the dataset into a training and testing set, six types of noise (Gaussian, Speckle, Salt and Pepper, Random Erasing, FGSM and PGD) were inserted into the training set and only two of them (FGSM, PGD) were inserted into the test set. All three networks were trained with the noisy, as well as the original images and subsequently tested on the same test sets. The experimental process is described thoroughly in the following sections. Having produced the distributions of the trained networks' accuracy performance on the test sets, the following statistical hypothesis-tests were conducted:

1) *Shapiro-Wilk test* [28]: The assumption of samples coming from a normal [29] distribution is required in a number of statistical tests that we wished to conduct. In order to verify if this assumption holds for our generated experimental samples, this non-parametric statistical test was used for all our sample distributions, to test the null hypothesis that a given sample comes from a normal distribution.
2) *Levene's test* [30]: The assumption of homoscedasticity, namely the equality of sample distributions' variance, is required in a number of multi-sample statistical tests that we intended to conduct. The validity of this assumption was assessed for a number of subgroups of our samples using this parametric test. The null hypothesis of Levene's test states that the variances of the populations from which a fixed group of two or more samples are drawn, are equal.
3) *ANOVA* [31]:The one-way analysis of variance (ANOVA) is used to determine whether there are any statistically significant differences between the means of three or more independent groups. The null hypothesis assumes the equality of the means of the groups and is tested against the alternative hypothesis, stating that there are at least two group means that are statistically significantly different from each other. The reliability of this test strongly depends on the following assumptions [32]:
   - Each sample was drawn from a normally distributed population.
   - The variances of the populations that the samples come from are equal.
   - The observations in each group are independent of each other and the observations within groups were obtained by a random sample.

In our case, the first two assumptions were assessed by conducting Shapiro-Wilk and Levene's tests for the samples participating in the groups for which ANOVA test was conducted. The third assumption was ensured to be satisfied by appropriate experimental design.

4) *Student's T-test* [33]: A two sample T-test is used to determine whether or not two population means are equal. The null hypothesis, that the difference in group means is zero, is tested - in the case of the two-tailed version of this test - against the alternative hypothesis that the difference in group means is different from zero. In the case of a left-tailed version of this test, the alternative hypothesis states that the population mean corresponding to the first sample is less than the one corresponding to the second sample. The assumptions [34] that should be assessed before conducting a T-test are the following:
   - The observations in one sample are independent of the observations in the other sample.
   - Both samples are approximately normally distributed.
   - Both samples have approximately the same variance.

   In our case, the first assumption was ensured to be satisfied by appropriately designing the experiments, while the rest of the assumptions were assessed with the use of Shapiro-Wilk and Levene's tests.

The null and alternative hypotheses of the above statistical tests are illustrated in the table IV-B below, according to the following notation: Let $X = \{X_1, ..., X_k\}$, $k \in \mathbb{N}$, where X_i is a random sample, i.e. $X_i = (x_1, ...x_n)$, $n \in \mathbb{N}$, $i \in \{1, ..., k\}$. Let μ_i and σ_i denote the mean and variance of each X_i, respectively.

Test	H_0	H_1
Shapiro-Wilk	$X_i \sim \mathcal{N}(\mu_i, \sigma_i^2), i \in \{1, ..., k\}$	X_i does not follow a normal distribution
Levene's	$\sigma_1^2 = \sigma_2^2 = ... = \sigma_i^2, i \in \{1, ..., k\}$	$\sigma_m^2 \neq \sigma_l^2, m, l \in \{1, ..., i\}, m \neq l$
ANOVA	$\mu_1 = \mu_2 = ... = \mu_i, i \in \{1, ..., k\}$	$\mu_m^2 \neq \mu_l^2, m, l \in \{1, ..., i\}, m \neq l$
Two-tailed T-student	$\mu_i = \mu_j, i, j \in \{1, ..., k\}, i \neq j$	$\mu_i \neq \mu_j, i, j \in \{1, ..., k\}, i \neq j$
Left-sided T-student	$\mu_i = \mu_j, i, j \in \{1, ..., k\}, i \neq j$	$\mu_i < \mu_j, i, j \in \{1, ..., k\}, i \neq j$
Right-sided T-student	$\mu_i = \mu_j, i, j \in \{1, ..., k\}, i \neq j$	$\mu_i > \mu_j, i, j \in \{1, ..., k\}, i \neq j$

TABLE I
NULL AND ALTERNATIVE HYPOTHESES

One of the outcomes of the statistical tests mentioned above, when conducted on a desired group of samples is the *p-value*. P-value is the probability of obtaining results at least as extreme as the observed results of a statistical hypothesis test, assuming that the null hypothesis is correct [35]. If p-value $< \alpha$, the null hypothesis H_0 is rejected against the alternative hypothesis H_1, where α stands for the level of statistical significance [36] of the statistical test. For the purpose of this work, α was set to 0.05 for all the statistical tests conducted. If p-value $\geq \alpha$, we can claim that there is not sufficient evidence to reject the null hypothesis H_0 [24]. In addition to the p-value of each test conducted, *effect sizes* were reported for T-student and ANOVA tests, as well [37]. Effect size is a quantitative measure of the magnitude of the experimental effect. The larger the effect size the stronger the relationship between two variables [38]. *Cohen's d* [39], [40] and η^2 [40] are the measures of the effect sizes which were chosen to be reported, for Student's T-tests and ANOVA tests, respectively. The tables II and III illustrate a detailed view of the statistical tests that we were planning to conduct and the sample distributions that participated in each one of them.

Test	Accuracy samples for which each test was used	Number of tests
Shapiro-Wilk	All the accuracy samples produced from the experiments	42
Levene's	Accuracy samples from a fixed network trained with six different noise types and tested on the test set with no noise	3
	Accuracy samples from all the networks trained with a fixed noise type and tested on the test set with no noise	6
ANOVA	Accuracy samples from a fixed network trained with six different noise types and tested on the test set with no noise	3
	Accuracy samples from all the networks trained with a fixed noise type and tested on the test set with no noise	6

TABLE II
HYPOTHESES TESTS USED

Adversarial attack	Network's accuracy samples No 1	Network's accuracy samples No 2	Number of tests
Two-Tailed			
FGSM attack	Alexnet trained with Gaussian noise	Squeezenet trained with Gaussian noise	1
	Alexnet trained with Gaussian noise	Resnet trained with Gaussian noise	1
	Resnet trained with Gaussian noise	Squeezenet trained with Gaussian noise	1
Left-Tailed			
FGSM	Alexnet FGSM	Alexnet PGD	1
	Squeezenet FGSM	Squeezenet PGD	1
	Alexnet no noise	Alexnet random erasing	1
	Alexnet no noise	Alexnet FGSM	1
	Alexnet no noise	Alexnet PGD	1
	Squeezenet no noise	Squeezenet random erasing	1
	Squeezenet no noise	Squeezenet FGSM	1
	Squeezenet no noise	Squeezenet PGD	1
PGD	Alexnet no noise	Alexnet random erasing	1
	Alexnet no noise	Alexnet FGSM	1
	Alexnet no noise	Alexnet PGD	1
	Squeezenet no noise	Squeezenet random erasing	1
	Squeezenet no noise	Squeezenet FGSM	1
	Squeezenet no noise	Squeezenet PGD	1

TABLE III
STUDENT T-TEST USED FOR FGSM ATTACKED TEST SET

The implementation for each statistical test is available in the source code folder.

V. EXPERIMENT

In this section, we describe the experimental workflow that was used to obtain numerical observations, in order to evaluate the previously stated hypotheses. For the purposes of our experiments, we used three different convolutional neural network topologies, namely AlexNet, ResNet-18 and SqueezeNet-1.0. The three mentioned neural networks were not trained beforehand, as we wanted to have an overview of unbiased results from a clean experimentation structure. The choice of these models was not random as they pose

important differences in their architecture. The selected dataset is the Oxford 102 Flower, which is an image classification dataset containing 102 flower categories (.jpg format).

A. Configuration

In view of the fact that a multitude of variables would have to be used interchangeably during multiple experimental executions, we established a single entry point to our algorithm. This was a configuration file that would be parsed, as a first step, by our algorithm in order to dynamically choose and initialize any of the required variables for each experiment instance. Such variables are noise type, mean and standard deviation for specific noise types, epsilon, alpha and iterations for adversarial attacks, epochs, neural network model etc.

B. Preprocessing

The images originally have different size, pose and light variations. Therefore, we applied preprocessing and data augmentation techniques to prepare the data before they were used as input to the models. Furthermore, the dataset was split into training, test and validation sets (80%-10%-10%). The transformations that were applied consisted of resizing and cropping the input so that all images have the same size. Furthermore, random rotations and flipping were applied to reduce the risk of over-fitting. Finally, standardization across each chromatic channel (RGB) was performed (mean=[0.485, 0.456, 0.406], standard deviaton=[0.229, 0.224, 0.225]) to achieve reduced training times. For the validation and test sets, resizing, cropping and standardization similar to the training set was performed.

C. Execution

Following the configuration of the experimental variables and preprocessing of our data we have a generic implementation of a neural network pipeline. Namely we used a a training function, an evaluation function as our low level processes and an abstract function to handle the aforementioned low level processes.

- **Fit** function: This function consists of an iterative procedure for the chosen epochs value. For the purposes of this experiment, in order to achieve consistency and compare the hypotheses results in an unbiased manner, we retained static values for the number of epochs (epochs=50), the neural network optimizer (optimizer=Stochastic Gradient Descent with learning rate=0.001 and momentum=0.9) and cross-entropy as our loss function. Another task of this function was to keep track of the accuracy and loss that was achieved after each epoch to ensure the smooth process operation.
- **Train** function: In the train function, the images used for training are iterated over by first initializing the gradients and then using the forward method for the chosen model. Then, the loss function is used to calculate the loss according to each image label. Following is the back propagation step and finally the updating of weights for each model.
- **Evaluation** function: The evaluation function is used in two cases. Firstly, it is used after the training phase in each epoch to log the accuracy and loss from images on the validation set (10% of the total dataset). It's second task is similar to the first but its used after the whole training and validation process has finished. We use the test set (10% of the total dataset) to gather observations for loss and accuracy that later are to be used to create the distributions for our statistical tests.

The final evaluation results of each combination of parameters (model/noise type) were logged in a .txt file.

VI. Results

Tables IV, V, VI, VII, VIII below, illustrate the outcomes of all the statistical hypothesis-tests conducted.

Network and training input noise	p value	H_0
No noise in the test set		
Alexnet trained with images with no noise	0.123	Not sufficient evidence to reject
Alexnet trained with images with Gaussian noise	0.208	Not sufficient evidence to reject
Alexnet trained with images with pgd adversarial noise	0.922	Not sufficient evidence to reject
Alexnet trained with images with salt and pepper noise	0.506	Not sufficient evidence to reject
Alexnet trained with images with eraser noise	0.321	Not sufficient evidence to reject
Alexnet trained with images with fgsm adversarial noise	0.831	Not sufficient evidence to reject
Alexnet trained with images with speckle noise	0.831	Not sufficient evidence to reject
Resnet trained with images with no noise	0.975	Not sufficient evidence to reject
Resnet trained with images with Gaussian noise	0.0220	Rejected
Resnet trained with images with speckle noise	0.705	Not sufficient evidence to reject
Resnet trained with images with salt and pepper noise	0.0695	Not sufficient evidence to reject
Resnet trained with images with eraser noise	0.847	Not sufficient evidence to reject
Resnet trained with images with pgd adversarial noise	0.924	Not sufficient evidence to reject
Resnet trained with images with fgsm adversarial noise	0.306	Not sufficient evidence to reject
Squeezenet trained with images with no noise	0.168	Not sufficient evidence to reject
Squeezenet trained with images with Gaussian noise	0.0862	Not sufficient evidence to reject
Squeezenet trained with images with speckle noise	0.172	Not sufficient evidence to reject
Squeezenet trained with images with salt and pepper noise	0.011	Rejected
Squeezenet trained with images with eraser noise	0.261	Not sufficient evidence to reject
Squeezenet trained with images with pgd adversarial noise	0.558	Not sufficient evidence to reject
Squeezenet trained with images with fgsm adversarial noise	0.754	Not sufficient evidence to reject
FGSM adversarial noise attack on the test set		
Alexnet trained with images with fgsm adversarial noise	0.864	Not sufficient evidence to reject
Alexnet trained with images with pgd adversarial noise	0.910	Not sufficient evidence to reject
Alexnet trained with images with Gaussian noise	0.172	Not sufficient evidence to reject
Squeezenet trained with images with fgsm adversarial	0.863	Not sufficient evidence to reject
Squeezenet trained with images with pgd adversarial noise	0.457	Not sufficient evidence to reject
Squeezenet trained with images with Gaussian noise	0.464	Not sufficient evidence to reject

TABLE IV
SHAPIRO-WILK TEST RESULTS

As shown in table IV, the accuracy samples coming from Resnet and Squeezenet having been trained on the set with Gaussian and salt and pepper noise respectively, and subsequently tested on the set with images with no noise, do not come from a normal

Networks' accuracy samples tested on the test set with no noise	p value	H_0	Result interpretation
Alexnet trained with 6 noise types	0.993	Not sufficient evidence to reject	Probably homoscedastic samples
Resnet trained with 6 noise types	0.998	Not sufficient evidence to reject	Probably homoscedastic samples
Squeezenet trained with 6 noise types	0.517	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with Gaussian noise	0.564	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with speckle noise	0.386	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with salt and pepper noise	0.969	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with eraser noise	0.712	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with fgsm noise	0.424	Not sufficient evidence to reject	Probably homoscedastic samples
The three networks trained with pgd noise	0.530	Not sufficient evidence to reject	Probably homoscedastic samples

TABLE V
LEVENE'S TEST RESULTS

Networks' accuracy samples tested on the test set with no noise	p value	H_0	η^2	Effect size interpretation [40]
Alexnet trained with 6 noise types	0.274×10^{-2}	Rejected	0.273	Large Effect
Resnet trained with 6 noise types	0.292×10^{-14}	Rejected	0.385	Large Effect
Squeezenet trained with 6 noise types	0.145×10^{-12}	Rejected	0.385	Large Effect
The three networks trained with Gaussian noise	0.237×10^{-3}	Rejected	0.200	Large Effect
The three networks trained with speckle noise	0.178×10^{-1}	Rejected	0.102	Large Effect
The three networks trained with salt and pepper noise	0.364×10^{-2}	Rejected	0.140	Large Effect
The three networks trained with eraser noise	0.140×10^{-1}	Rejected	0.108	Large Effect
The three networks trained with FGSM adversarial noise	0.449×10^{-2}	Rejected	0.134	Large Effect
The three networks trained with pgd adversarial noise	0.999×10^{-1}	Not sufficient evidence to reject	0.596×10^{-1}	Intermediate Effect

TABLE VI
ANOVA TEST RESULTS

Networks trained on images with no noise	Training input noise	p value	H_0	Cohen d	Effect size interpretation
Alexnet	Gaussian	0.492×10^{-4}	Rejected	0.232	Small Effect
	Speckle	0.692×10^{-5}	Rejected	1.34	Large Effect
	Salt and pepper	0.447×10^{-4}	Rejected	1.18	Large Effect
	Eraser	0.797	Not sufficient evidence to reject	0.232	Small Effect
	FGSM	0.112	Not sufficient evidence to reject	0.342	Small Effect
	PGD	0.411	Not sufficient evidence to reject	0.631×10^{-1}	No Effect
Resnet	Speckle	0.451×10^{-2}	Rejected	0.754	Large Effect
	Salt and pepper	0.836×10^{-6}	Rejected	1.51	Large Effect
	Eraser	0.710	Not sufficient evidence to reject	0.154	Small Effect
	FGSM	0.715	Not sufficient evidence to reject	0.159	Small Effect
	PGD	0.520	Not sufficient evidence to reject	0.137×10^{-1}	No Effect
Squeezenet	Gaussian	0.530×10^{-9}	Rejected	2.08	Large Effect
	Speckle	0.126×10^{-2}	Rejected	0.882	Large Effect
	Eraser	0.228	Not sufficient evidence to reject	0.209	Small Effect
	FGSM	0.158	Not sufficient evidence to reject	0.281	Small Effect
	PGD	0.292	Not sufficient evidence to reject	0.153	Small Effect

TABLE VII
RIGHT-TAILED T-STUDENT TEST RESULTS: NO NOISE VS SIX NOISE TYPES

Adversarial attack	Network's accuracy samples No 1	Network's accuracy samples No 2	p value	H_0	Cohen d	Effect size interpretation [39]
Two-Tailed						
FGSM attack	Alexnet trained with Gaussian noise	Squeezenet trained with Gaussian noise	0.241×10^{-63}	Rejected	23.6	Large Effect
	Alexnet trained with Gaussian noise	Resnet trained with Gaussian noise	0.794×10^{-21}	Rejected	3.72	Large Effect
	Resnet trained with Gaussian noise	Squeezenet trained with Gaussian noise	0.178×10^{-71}	Rejected	32.6	Large Effect
Left-Tailed						
FGSM attack	Alexnet trained with FGSM	Alexnet trained with PGD	0.495×10^{-24}	Rejected	4.50	Large Effect
	Squeezenet trained with FGSM	Squeezenet trained with PGD	0.292×10^{-51}	Rejected	14.3	Large Effect
	Alexnet trained with no noise	Alexnet trained with random erasing	0.135×10^{-19}	Rejected	3.62	Large Effect
	Alexnet trained with no noise	Alexnet trained withFGSM	0.290×10^{-57}	Rejected	18.2	Large Effect
	Alexnet trained with no noise	Alexnet trained with PGD	0.819×10^{-62}	Rejected	21.9	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with random erasing	0.831×10^{-44}	Rejected	10.6	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with FGSM	0.194×10^{-66}	Rejected	26.3	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with PGD	0.147×10^{-77}	Rejected	41.0	Large Effect
PGD attack	Alexnet trained with no noise	Alexnet trained with random erasing	0.376×10^{-13}	Rejected	2.52	Large Effect
	Alexnet trained with no noise	Alexnet trained with FGSM	0.677×10^{-56}	Rejected	17.3	Large Effect
	Alexnet trained with no noise	Alexnet trained with PGD	0.506×10^{-62}	Rejected	22.1	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with random erasing	0.261×10^{-50}	Rejected	13.8	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with FGSM	0.280×10^{-79}	Rejected	43.9	Large Effect
	Squeezenet trained with no noise	Squeezenet trained with PGD	0.153×10^{-84}	Rejected	54.2	Large Effect

TABLE VIII
STUDENT T-TEST RESULTS FOR FGSM ATTACKED TEST SET

distribution with a 0.05 level of statistical significance. Due to the violation of the normality assumption, these two samples were excluded from the Student T-tests.

Figures 4, 5, 6 below illustrate the theoretical populations - normal distributions- from which each sample of the accuracy of the networks trained on the noisy, as well as the original training set were drawn. These samples were obtained from testing the networks on a set with no noise inserted.

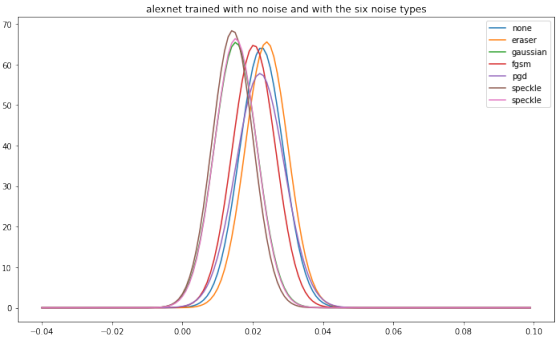

Fig. 4. Alexnet

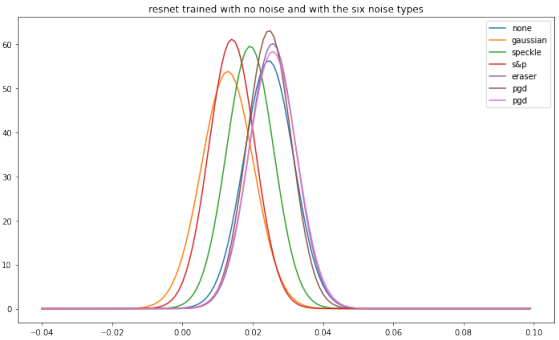

Fig. 5. Resnet

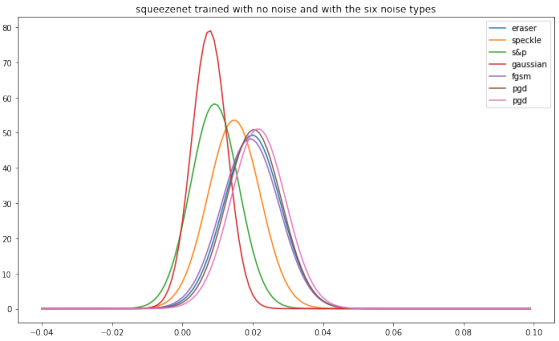

Fig. 6. Squeezenet

An analogous illustration is provided in figures 7-12. Each figure illustrates the theoretical populations - normal distributions - from which the accuracy samples of all three networks trained with a fixed noise type were drawn.

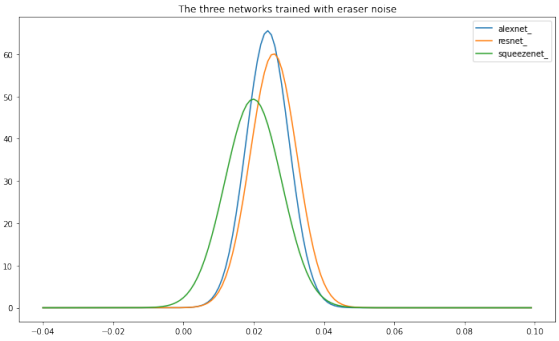

Fig. 7. Speckle noise

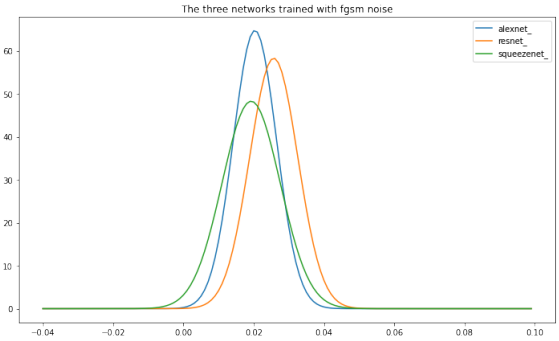

Fig. 8. FGSM noise

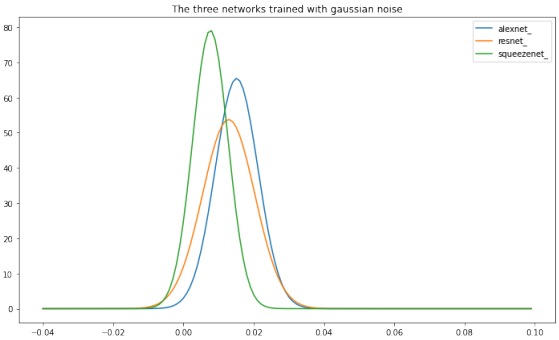

Fig. 9. Gaussian noise

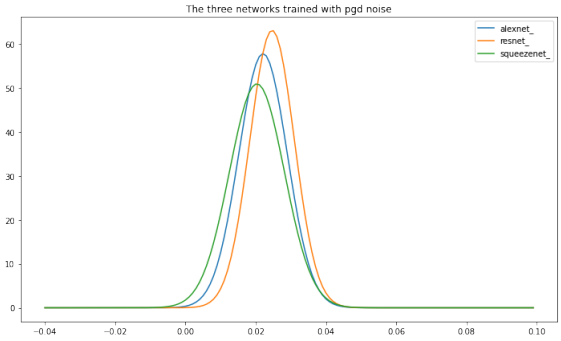

Fig. 10. PGD noise

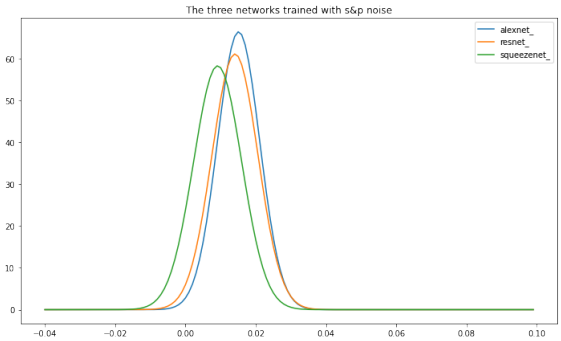

Fig. 11. Salt and pepper noise

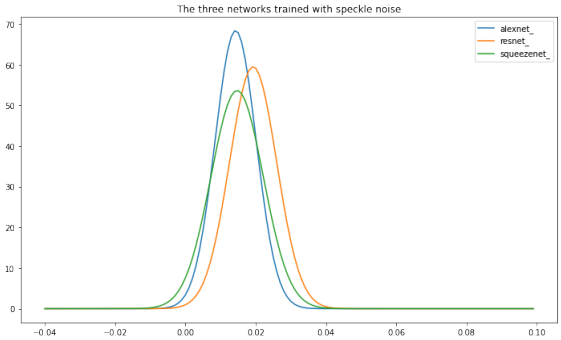

Fig. 12. Speckle noise

VII. Conclusion

Feature noise has various effects in different neural network architectures, whether those are negative or in several other cases positive.

We have observed how different types of noise affect the performance of the three selected models used in the context of this work. The loss and accuracy scores indicate that all of the models got affected in some different way, when trained with the selected types of noise. The above statement is proved true on a level of statistical significance 95%, as it is confirmed by the rejection of the ANOVA test hypothesis. This can be visually verified through fig 2-4. It is worth noting that in Alexnet's case we observe a smaller effect size which implies that the different noise types have a reduced influence. Finally, in all three cases the effect size implies a large effect of the phenomenon.

Adversarial training could in some cases hurt the generalization performance of a neural network, when that network is presented with clean images during the testing phase. In our experiment though, there is not enough evidence to conclusively state that PGD and FGSM adversarial training affect the accuracy scores of the three networks in a statistical significant level when the networks are provided clean test images.

In general, a good practice is considered to be the training of a neural network with adversarial examples in order to be robust against a possible adversarial attack in the future. As it is illustrated in Table VIII, training neural networks with adversarial examples truly leads to an improvement in their predictive capabilities, when attacked with adversarial noise examples with a degree of statistical significant level of 95%.

As both Table VII, Table VIII point, random erasing boosts the performance of each of the networks which are evaluated, and achieves better generalization ability and robustness. Based on our tests, the latter is illustrated on a level of 95% of statistical significance.

Finally, according to Table VIII we can observe that SqueezeNet is more robust in a case of an adversarial attack, in contrast to AlexNet. We suggest that this occurs due to the fact that SqueezeNet comprises of far fewer parameters than it's counterpart AlexNet, therefore the attack space is smaller, and that may be the reason for the difference in performance under adversarial noise.

REFERENCES

[1] G. De Barros Paranhos da Costa, W. Contato, T. Nazare, J. Neto, and M. Ponti, "An empirical study on the effects of different types of noise in image classification tasks," 09 2016.

[2] J. Al Azzeh, B. Zahran, and Z. Alqadi, "Salt and pepper noise: Effects and removal," *JOIV : International Journal on Informatics Visualization*, vol. 2, 07 2018.

[3] F. Russo, "A method for estimation and filtering of gaussian noise in images," *IEEE Transactions on Instrumentation and Measurement*, vol. 52, no. 4, pp. 1148–1154, 2003.

[4] C. Tian, L. Fei, W. Zheng, Y. xu, W. Zuo, and C.-W. Lin, "Deep learning on image denoising: An overview," *Neural Networks*, vol. 131, 08 2020.

[5] T. Nazaré, G. De Barros Paranhos da Costa, W. Contato, and M. Ponti, *Deep Convolutional Neural Networks and Noisy Images*, 01 2018, pp. 416–424.

[6] R. Ren, Z. Guo, Z. Jia, J. Yang, N. Kasabov, and C. Li, "Speckle noise removal in image-based detection of refractive index changes in porous silicon microarrays," *Scientific Reports*, vol. 9, 10 2019.

[7] S. Sharma and E. K. Kumar, "A comparison of salt and pepper noise removal filters," pp. 2627–2630, 08 2016.

[8] L. Badri, "Development of neural networks for noise reduction," *Int. Arab J. Inf. Technol.*, vol. 7, pp. 289–294, 07 2010.

[9] N. Narodytska and S. Kasiviswanathan, "Simple black-box adversarial attacks on deep neural networks," 07 2017, pp. 1310–1318.

[10] C. Song, S. Sudirman, and M. Merabti, "A spatial and frequency domain analysis of the effect of removal attacks on digital image watermarks‖," 01 2010.

[11] A. Boyat and B. Joshi, "A review paper: Noise models in digital image processing," *Signal Image Processing : An International Journal*, vol. 6, 05 2015.

[12] A. Maity, A. Pattanaik, S. Sagnika, and S. Pani, "A comparative study on approaches to speckle noise reduction in images," in *2015 International Conference on Computational Intelligence and Networks*, 2015, pp. 148–155.

[13] V. Bianco, P. Memmolo, M. Leo, S. Montrésor, C. Distante, M. Paturzo, P. Picart, B. Javidi, and P. Ferraro, "Strategies for reducing speckle noise in digital holography," *Light: Science Applications*, vol. 7, 08 2018.

[14] R. Castaneda, J. Garcia-Sucerquia, and A. Doblas, "Speckle noise reduction in coherent imaging systems via hybrid median–mean filter," *Optical Engineering*, vol. 60, no. 12, pp. 1 – 12, 2021. [Online]. Available: https://doi.org/10.1117/1.OE.60.12.123107

[15] R. Chan, C.-W. Ho, and M. Nikolova, "Salt-and-pepper noise removal by median-type noise detectors and detail-preserving regularization," *Image Processing, IEEE Transactions on*, vol. 14, pp. 1479 – 1485, 11 2005.

[16] Q. Xu, Q. Zhang, D. Hu, and J. Liu, "Removal of salt and pepper noise in corrupted image based on multilevel weighted graphs and igowa operator," *Mathematical Problems in Engineering*, vol. 2018, pp. 1–11, 05 2018.

[17] Z. Zhong, L. Zheng, G. Kang, S. Li, and Y. Yang, "Random erasing data augmentation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, pp. 13 001–13 008, Apr. 2020. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/7000

[18] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 2014. [Online]. Available: https://arxiv.org/abs/1412.6572

[19] J. Chen, M. Su, S. Shen, H. Xiong, and H. Zheng, "Poba-ga: Perturbation optimized black-box adversarial attacks via genetic algorithm," *Computers Security*, vol. 85, pp. 89–106, 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404818314378

[20] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," 2017. [Online]. Available: https://arxiv.org/abs/1706.06083

[21] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[22] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[23] F. N. Iandola, M. W. Moskewicz, K. Ashraf, S. Han, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and <1mb model size," *CoRR*, vol. abs/1602.07360, 2016. [Online]. Available: http://arxiv.org/abs/1602.07360

[24] D. J. Biau, B. M. Jolles, and R. Porcher, "P value and the theory of hypothesis testing: an explanation for new researchers," *Clinical Orthopaedics and Related Research®*, vol. 468, no. 3, pp. 885–892, 2010.

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[26] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.

[28] S. S. Shapiro and M. B. Wilk, "An analysis of variance test for normality (complete samples)," *Biometrika*, vol. 52, no. 3/4, pp. 591–611, 1965.

[29] X. Zhang, *Gaussian Distribution*. Boston, MA: Springer US, 2010, pp. 425–428. [Online]. Available: https://doi.org/10.1007/978-0-387-30164-8_323

[30] H. Levene *et al.*, "Contributions to probability and statistics," *Essays in honor of Harold Hotelling*, pp. 278–292, 1960.

[31] L. Sthle and S. Wold, "Analysis of variance (anova)," *Chemometrics and Intelligent Laboratory Systems*, vol. 6, no. 4, pp. 259–272, 1989. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0169743989800954

[32] Zach, "How to check anova assumptions," Aug 2021. [Online]. Available: https://www.statology.org/anova-assumptions/

[33] Student, "The probable error of a mean," *Biometrika*, pp. 1–25, 1908.

[34] Zach, "The four assumptions made in a t-test," Mar 2021. [Online]. Available: https://www.statology.org/t-test-assumptions/

[35] R. L. Wasserstein and N. A. Lazar, "The asa statement on p-values: context, process, and purpose," pp. 129–133, 2016.

[36] W. Kenton, “Statistical significance definition,” Mar 2022. [Online]. Available: https://www.investopedia.com/terms/s/statistically_significant.asp

[37] G. M. Sullivan and R. Feinn, “Using effect size—or why the p value is not enough,” *Journal of graduate medical education*, vol. 4, no. 3, pp. 279–282, 2012.

[38] A. Carpenter, “Effect size,” Dec 2020. [Online]. Available: https://towardsdatascience.com/effect-size-d132b0cc8669

[39] J. Cohen, *Statistical power analysis for the behavioral sciences*. Routledge, 2013.

[40] D. Lakens, “Calculating and reporting effect sizes to facilitate cumulative science: A practical primer for t-tests and anovas,” Nov 2013. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3840331/