A noise robust convolutional neural network for image classification

Momeny, Mohammad

Elsevier
2021-06

Momeny, M, Latif, A M, Sarram, M A, Sheikhpour, R & Zhang, Y D 2021, 'A noise robust convolutional neural network for image classification', Results in Engineering, vol. 10, 100225. https://doi.org/10.1016/j.rineng.2021.100225

http://hdl.handle.net/10138/571465
10.1016/j.rineng.2021.100225

Results in Engineering 10 (2021) 100225

Contents lists available at ScienceDirect

Results in Engineering

journal homepage: www.editorialmanager.com/rineng/Default.aspx

Full Length Article

A noise robust convolutional neural network for image classification

Mohammad Momeny [a,*], Ali Mohammad Latif [a], Mehdi Agha Sarram [a], Razieh Sheikhpour [b], Yu Dong Zhang [c]

[a] Department of Computer Engineering, Faculty of Engineering, Yazd University, Yazd, Iran
[b] Department of Computer Engineering, Faculty of Engineering, Ardakan University, P.O. Box 184, Ardakan, Iran
[c] Department of Informatics, University of Leicester, Leicester, UK

ARTICLE INFO

Keywords:
Convolutional neural network
Noise
Image classification
Adaptive pooling
Adaptive convolution
Adaptive data augmentation

ABSTRACT

Convolutional Neural Networks (CNNs) are extensively used for image classification. Noisy images reduce the classification performance of convolutional neural networks and increase the training time of the networks. In this paper, a Noise-Robust Convolutional Neural Network (NR-CNN) is proposed to classify the noisy images without any preprocessing for noise removal and improve the classification performance of noisy images in convolutional neural networks. In the proposed NR-CNN, a noise map layer and an adaptive resize layer are added to the architecture of convolutional neural network. Moreover, the noise problem is considered in different components of NR-CNN such that convolutional layer, pooling layer and loss function of the convolutional neural network are improved for robustness of CNN to noise. The adaptive data augmentation based on noise map are introduced to improve the classification performance of the proposed NR-CNN. Experimental results demonstrate that the proposed NR-CNN improves the noisy image classification and the network training speed.

1. Introduction

Image classification is used in various applications, such as security, educational and promotional systems. In recent years, many researches have been done to design automated systems for extracting fundamental features from images [1–9]. Convolutional neural network (CNN) is an effective method for image classification which uses convolutional, pooling and fully-connected layers for learning process. This network is a kind of multi-layer neural networks which consists of neurons with trainable weights and biases [10–16].

Noisy images are the destructive factors in the training of convolutional neural networks and decrease the classification performance of the networks [17]. Noise removal from the images is an important issue in the image processing which is often done as a preprocessing step [17–21].

Various types of noise such as impulse noise, missing image samples, packet loss in image transmission, damaged image and tampered images influence the image quality and make images unsuitable for image processing. Impulse noise is one of the most common types of noise that occurs during image acquisition, recording, and transmission [22,23]. The intensity of a pixel corrupted by the impulse noise is much higher or lower than those of its uncorrupted neighbors [23]. Missing image samples occur when parts of an image are missing, damaged or partially occluded by undesired objects. Image inpainting has been widely used to solve the problem and repair damaged/missing pixels of the images [24–26]. Packet loss is a problem that occurs during image transmission in Wireless Multimedia Sensor Networks (WMSNs) and causes the degradation of image quality [27,28]. Damaged images refer to the images degraded by dropouts and outliers. Outliers (anomalous pixel values) are generated by calibration errors, and dropouts by display monitor faults and abrasion on photographic material [29–31]. Digital images may be tampered maliciously which leads to reduce the quality of them. Restoration of the tampered images is a costly process [32].

Images corrupted by noise affect the performance of the convolutional neural networks. The noisy images are often restored in the preprocessing step, which causes the improvement of the classification performance of CNN. However, the noisy images may not be completely restored in the preprocessing step due to the high density of noise, which leads to the negative impact on the learning and validation of the CNN. Moreover, preprocessing for noise removal is a costly and time-consuming process. The aim of this paper is to propose a Noise-Robust Convolutional Neural Network (NR-CNN) for classification of noisy images, which does not require preprocessing for noise removal. In the proposed method, the robustness of CNN is carried out to various types of

* Corresponding author.
E-mail address: mohamad.momeny@gmail.com (M. Momeny).

https://doi.org/10.1016/j.rineng.2021.100225
Received 22 February 2021; Received in revised form 25 March 2021; Accepted 22 April 2021

noise such as impulse noise, missing image samples, packet loss in image transmission, damaged images and tampered images. Proposed NR-CNN modifies the architecture of the basic convolutional neural network for robustness to noise by adding a noise map layer and an adaptive resize layer, and considers the classification of noisy images in different components of NR-CNN. Extensive experiments show the effectiveness of the proposed NR-CNN in classification of noisy images.

We can summarize the advantages of the proposed NR-CNN as follows:

- The proposed method can be simultaneously used for robustness of CNN to several types of noise. An image with different noise types is processed only once by the proposed NR-CNN.
- The proposed NR-CNN requires no preprocessing for restoration of noisy images and speeds up the classification of these images in the training stage.
- The performance of the proposed NR-CNN is better than other methods in classification of noisy images.

The contributions of this paper are as follows, each of which is described in the relevant subsection:

1. A convolution neural network is presented which is robust to noise and improves the classification performance of noisy images. The proposed NR-CNN classifies the noisy images without any preprocessing for noise removal.
2. An adaptive resize layer is placed at the NR-CNN to increase the robustness of the network to noise.
3. Adaptive filtering is applied in the architecture of the convolutional layer of the proposed NR-CNN, and an algorithm is proposed for adaptive stride to robust the network to noise.
4. An adaptive pooling operator to noise is considered in the proposed NR-CNN to decrease noise effects in the pooling layer.
5. Adaptive data augmentation based on noise map are presented for increasing the classification performance of the proposed NR-CNN.

The reminder of this paper is organized as follows. The proposed noise-robust convolutional neural network is described in Section 2. In this section, different components of the proposed CNN are expressed in details. Extensive experiments are conducted for evaluating the proposed NR-CNN and provided in Section 3. The conclusion of the paper is presented in Section 4.

2. Proposed noise-robust convolutional neural network

The convolutional, pooling, and fully-connected layers are main components of convolutional neural networks [13]. The convolution layer collected of several convolution kernels obtains feature maps. Mathematically, the feature value in the feature map of a layer can be computed by Ref. [33]:

$$z_{i,j,k} = \mathbf{w}_k^T \mathbf{x}_{ij} + b_k \tag{1}$$

where i, j are the location in the kth feature map, $\mathbf{x}_{ij}$ is the input patch, $\mathbf{w}_k$ is the weight vector of the kth filter, and b is the bias of the layer. The pooling layer, located among convolutional layers, plans to get shift-invariance by down sampling of the feature maps. Denoting the activation function of CNN as act(·), pooling function as pooling(·), and $\mathscr{R}_{ij}$ as a local neighborhood, down sampling operation in the feature map of a layer is calculated by Ref. [13]:

$$y_{i,j,k} = pooling\left(act\left(z_{i,j,k}\right)\right), \forall (m,n) \in \mathscr{R}_{ij} \tag{2}$$

The fully-connected layers may be placed after convolutional and pooling layers. For classification issues, the softmax function is usually applied in the output layer (the last layer of CNNs). The optimum values of parameters of CNN (e.g., the weight vectors and bias terms) for a classification issue can be achieved by minimizing a loss function [13]. Denoting $\boldsymbol{x}^{(n)}$ as the nth input data, $\boldsymbol{y}^{(n)}$ as real target label of nth input data, $\boldsymbol{o}^{(n)}$ as the nth output of CNN classification, and $\boldsymbol{\theta}$ as all the parameters, the loss of CNN can be obtained as (3) [33]:

$$\mathscr{L} = \frac{1}{N} \sum_{n=1}^{N} \ell(\boldsymbol{\theta}; \boldsymbol{y}^{(n)}, \boldsymbol{o}^{(n)}), n \in [1, \ldots, N] \tag{3}$$

In this section, a Noise-Robust Convolutional Neural Network (NR-CNN) is presented which classifies noisy images without any preprocessing for noise removal. General process of the proposed NR-CNN is presented in Fig. 1. The components of the NR-CNN are described in the following.

2.1. Noise map layer

The proposed NR-CNN has a noise map layer placed at the beginning of the NR-CNN and used to detect impulse noise, missing image samples, packet loss in image transmission, damaged images and tampered images. The noise map layer detects various types of noise and generates a noise map for each image to indicate the noisy and uncorrupted images. The architecture of the noise map layer is shown in Fig. 2.

As can be seen from Fig. 2, impulse noise is detected via a local consensus index schemea method [34]. Packet loss in image transmision is detected based on the image content [28]. Moreover, missing image samples are detected using the target regions [24], damaged imges using the statistical approach [29], and tampered images using the image authentication method [32].

In the proposed method, for robustness of CNN to noise, a noise map is generated for each image based on the noise type detected in the previous step. As shown in Fig. 3, each image contains four channels: noise map channel, red channel, blue channel and green channel. Therefore, the images are given with four channels to CNN. In the training process based on the noise map, NR-CNN learns to deal with noisy pixels.

2.2. Adaptive resize layer to noise

The convolutional neural networks usually start with the convolutional layer. The input images of the CNN should generally have a fixed size (i.e. 244 × 244) [35]. Therefore, the dimensions of the images larger than the size of the CNN input should be reduced. In this paper, a new layer called the adaptive resize layer is placed in the CNN to increase the robustness of the network to noise. The task of the adaptive resize layer is to improve the dimension reduction method for larger images than the CNN input using the noise map. In the selection of a pixel from a few

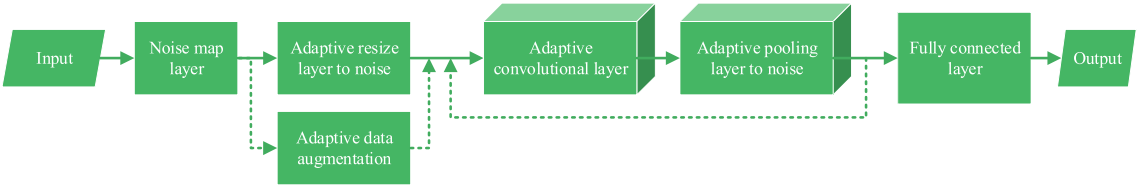

Fig. 1. General process of the proposed NR-CNN.

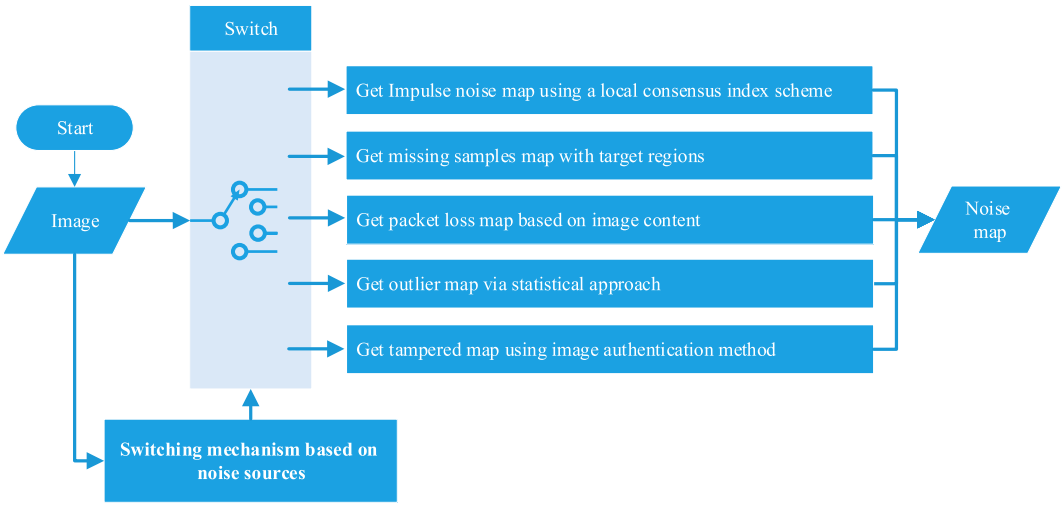

Fig. 2. The architecture of the noise map layer.

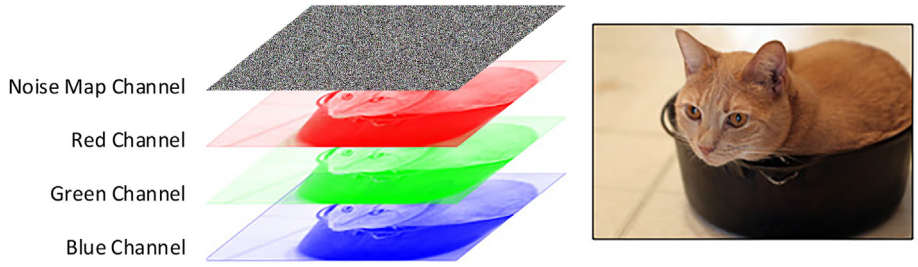

Fig. 3. Four channels for each image.

pixels for reduceing the image size using the noise map, the noisy pixel is removed, and the dimension reduction is performed using the rest of the pixels. The adaptive resize layer makes the noisy pixels are not participated in the dimension reduction process of the CNN input image. The beginning of the CNN with the adaptive resize layer is shown in Fig. 4.

2.3. Adaptive convolutional layer to noise

Convolutional layer parameters include a set of learnable filters. In this paper, the convolutional layer architecture of CNN is modified by adaptive filtering to present an effective method for robustness of CNN to noise.

2.3.1. Adaptive filtering

The proposed method by dropping the noisy connections between source and kerner of convolution, reduces the effect of noise on the CNN. Adaptive filtering with droppted noisy connections is performed based on the pixel values in CNN which increases the classification accuracy. Fig. 5 shows the architecture of the proposed convolutional layer for robustness of the network to noisy images. Dropping noisy connections prevents entering noisy pixels to the next layers. The proposed method for removing noise connections can be used for different convolution kernel size. It is worth mentioning that the noise map is updated in each layer.

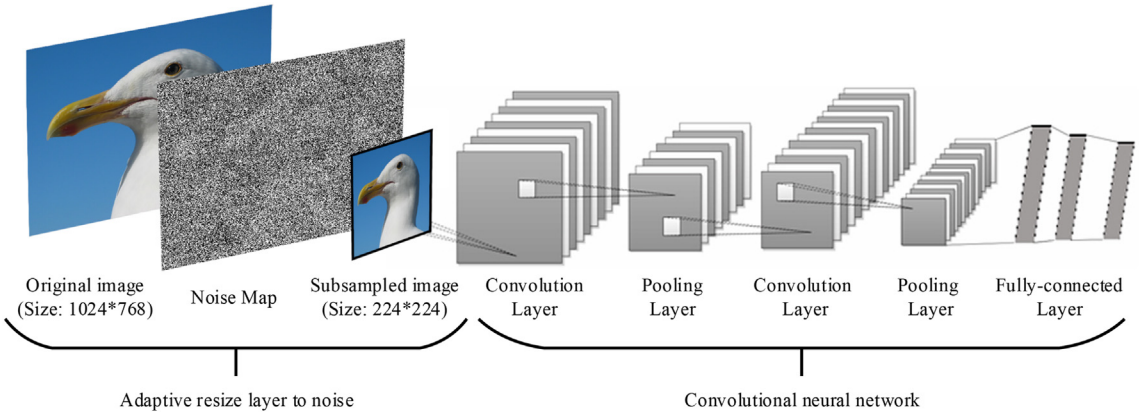

Fig. 4. The beginning of CNN with the adaptive resize layer to noise.

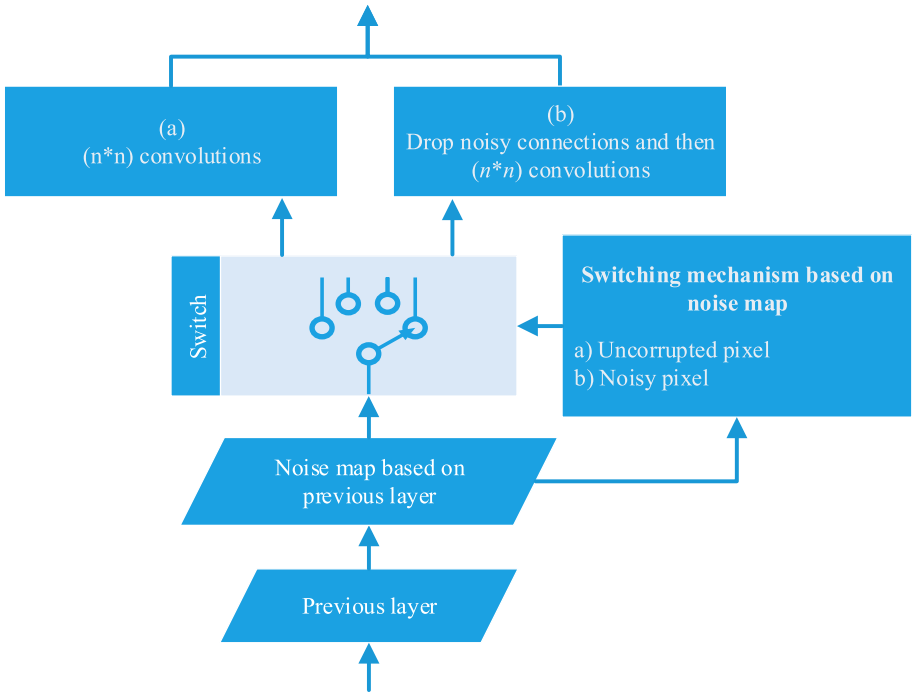

Fig. 5. The architecture of the proposed convolutional layer for robustness of the network to noisy images.

2.3.2. *Adaptive stride to noise*

The proposed algorithm for adaptive stride to noise is described in Algorithm 1, which improves the classification accuracy of noisy images.

Algorithm 1
The proposed algorithm for adjusting the adaptive stride to noise

1 Make a bit matrix as a stride map
2 Mark the selected pixels in each stride
3 Match the noise map and stride map to detect noisy pixels
4 Change the filter position from the noisy area to the closest uncorrupted one in the neighborhood of w × w

Correcting the adaptive stride map makes the noisy pixels are not considered in the classification process. As can be seen in Fig. 6, the noise map and stride map are initially matched together, then the stride operation is done. Fig. 7 illustrates the workflow of the proposed method for adaptive stride to noise in CNN to improve the classification of noisy images. w × w is the area which is searched to find the closest uncorrupted position for changing the position of the filter.

2.4. *Adaptive pooling layer to noise*

In convolutional architecture, several pooling layers can be placed between convolutional layers. Reducing the width and height of the input image is done in the pooling layer to decreases the number of parameters and computations [36]. A pooling function replaces the network output in a specific location with the statistical summary of neighboring outputs. Noise has a direct impact on the pooling layer. In this paper, a new method for modifying the pooling operator is proposed to improve the accuracy of the convolutional neural network in the classification of noisy images. A numerical example of the max-pooling operator is shown in Fig. 8. To eliminate the noise in the pooling layer, the pooling operator should prevent the processing of the noisy pixels. Fig. 9 demonstrates the proposed method for improving the max-pooling operator to remove low density noise. Noisy pixels are indicated by black squares. As shown in Fig. 9, avoiding the processing of noisy pixel in the pooling operator prevents the selection of noisy values for using in subsequent CNN layers, which increases the accuracy of image classification. Given that the pooling operator is repeated several times in the layers of convolutional neural network, the proposed method for the pooling of noisy pixels can be used repeatedly. Fig. 10 shows the proposed method for improving the max-pooling operator to deal with high density noisy pixels. Fig. 11 illustrates the workflow of the proposed method for improving the max-pooling operator to remove noise. In the workflow, a switch controls the selection of pooling method based on the pixel values in CNN.

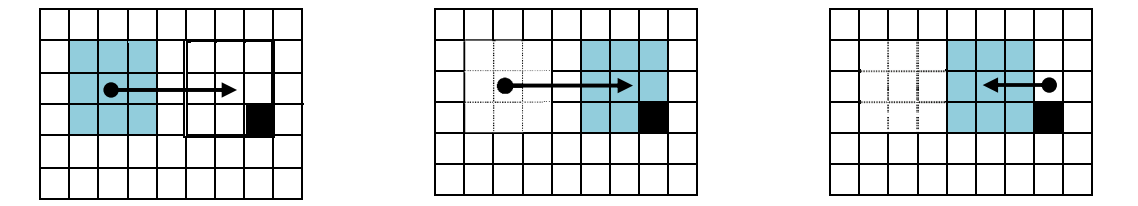

Fig. 6. The adaptive stride in CNN based on noise map. Noisy pixels are indicated by black color. (For interpretation of the references to color in this figure legend, the reader is referred to the Web version of this article.)

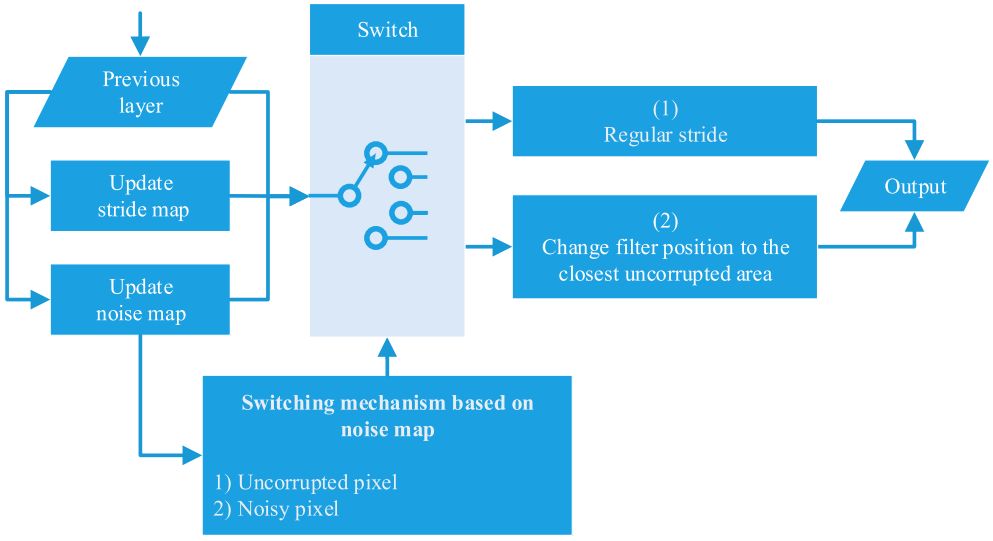

Fig. 7. The workflow of the proposed adaptive stride to noise.

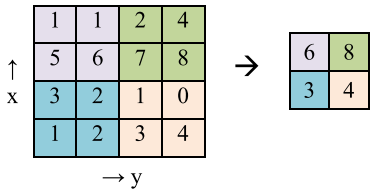

Fig. 8. 2 × 2 max-pooling operator with a stride of 2 pixels for uncorrupted pixels.

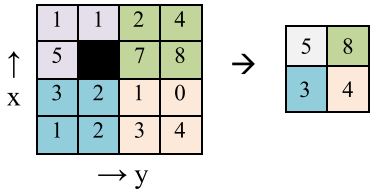

Fig. 9. The proposed method for improving 2 × 2 max-pooling operator with a stride of 2 pixels to remove low density noise (Black squares indicate the noisy pixels).

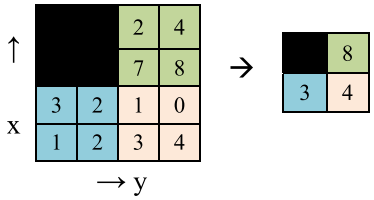

Fig. 10. The proposed method for improving the max-pooling operator to remove high density noise (Black squares indicate the noisy pixels).

2.5. Data augmentation based on noise map

Data augmentation can be used to generate additional data for CNN without imposing the cost of labeling [37]. In this paper, Eq. (1) is proposed for data augmentation by changing the light intensity of the images based on the noise map.

$$O(i,j) = I(i,j) + (1 - P(i,j)) \times X(i,j) \tag{4}$$

where I is the input image, P is the noise map, X is the light intensity change and O is the output image. Given the value of noise map (P), the following conditions are considered for each pixel of the image:

$$P_{ij} = \begin{cases} 0 & \textit{The pixel is uncorrupted} \\ 1 & \textit{The pixel is noisy} \end{cases} \tag{5}$$

It is worth noting that X involves the integer numbers in the interval [-a +a] which determines the intensity change for each pixel. In the proposed method, the noisy pixels are not considered in the light intensity change process, and only the values of uncorrupted pixels are changed. This decreases the computational overhrad. Fig. 12 shows the data augmentation architecture based on the noise map.

2.6. The computational overhead of the proposed NR-CNN

The computational overhead of the algorithms used for improving the quality of noisy images is important [38]. Typically, images corrupted by impulse noise, missing image samples, packet loss in image transmission, damaged images and tampered images are restored in two steps. Uncorrupted or noisy pixels are detected in the first step. Restoring the noisy pixels and improving the image quality are done in the second step [22,23,39].

In the first step of the proposed method, noise detection is performed by generating the noise map, and in the second one, the robustness of CNN to noise is done without any preprocessing for image restoration which is a contribution of this paper. The proposed CNN uses the switching method to ignore noisy pixels. With given noise map in the second step, the computational overhead is low because our method only prevents processing of noisy pixels.

3. Experiments

In this section, extensive experiments are performed to evaluate the performance of the proposed NR-CNN for classification of noisy images and compare to the deep VGG-Net models [40] with medium and slow architectures, GoogleNet [41] and ResNet [42].

Accuracy and error rate have taken as criteria for the comparison among methods, in accordance to the following [43,44]:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}, \tag{8}$$

TP : *True positive,*

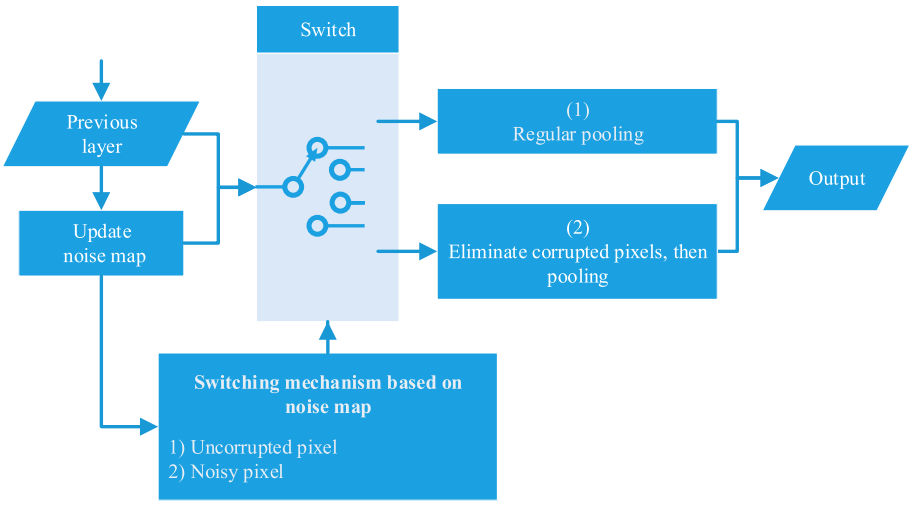

Fig. 11. The workflow of the proposed method for improving the max-pooling operator to remove noise.

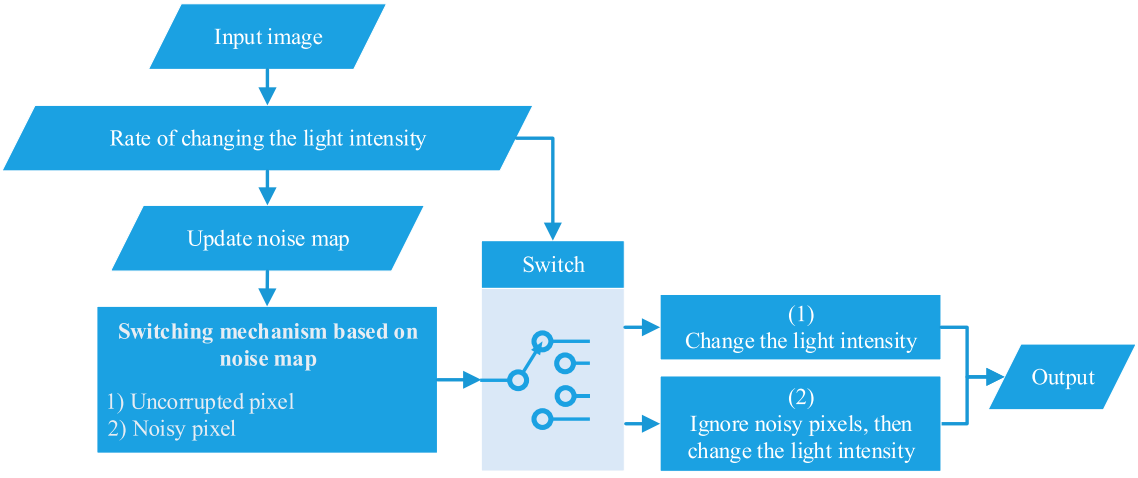

Fig. 12. Data augmentation architecture based on noise map.

TN : *True negative*,
FP : *False positive*,
FN : *False negative*

$$Error\ rate = 1 - Accuracy \tag{7}$$

3.1. Simulation of noises

The uniform distribution is used to simulate all types of noise. To create a tampered image, noise is added to the image as lines with a width of 1 pixel and a length of 5 pixels. To create a damaged image, solid circles with 2 pixels radius are added to the image. To simulate packet loss in the image, solid rectangles with the width of 2 pixels and the length equal to the length of the input image are randomly added to the image. To create missing samples, 2 × 2 squares are added to the image as noise. Fig. 13 shows a variety of noises with 20% density.

3.2. Comparison of the proposed NR-CNN with VGG-Net models

The structure of proposed NR-CNN can be added and used for any pre-trained CNN architecture in the test stage. For example, the proposed NR-CNN are configured based on the medium and slow architectures to compare with VGG-Net models. Table 1 demostrates the configuration of the proposed NR-CNN based on the VGG-Net-Medium model. In Table 2, the configuration of the proposed NR-CNN is shown based on the VGG-Net-Slow model. Proposed NR-CNN contains 5 convolutional layers and 3 fully connected layers. The input of NR-CNN is a fixed-size 224 × 224 RGB image.

In the first convolutional layer, the filter is adaptive to noise as shown in Fig. 5. Adaptive stride to noise described in Algorithm 1 is used in the proposed NR-CNN. Local Response Normalization (LRN) [35] is applied for normalization and adaptive max-pooling for downsampling. The noise map is updated in each layer. The activation function for all weight layers (except for the last layer) is the REctification Linear Unit (RELU) [35].

For training the networks, the ILSVRC-2012 dataset containing images of 1000 classes [35] is used. The dataset is divided into two sets: training (1.3 M uncorrupted images) and validation (50K uncorrupted and noisy images) setes. The top-5 error is used to evaluate the classification performance. The top-5 error is the main evaluation criterion used in ILSVRC and computed as the proportion of images such that the ground-truth category is outside the top-5 predicted categories. The dimensions of the original images in the Image-Net database are multiples of the input of NR-CNN. Therefore, using the noise map in the dimension reduction process at the adaptive resize layer, a portion of the image noises is removed. Adaptive convolution layer based on the uncorrupted and noisy pixels improve the classification performance. Determining the

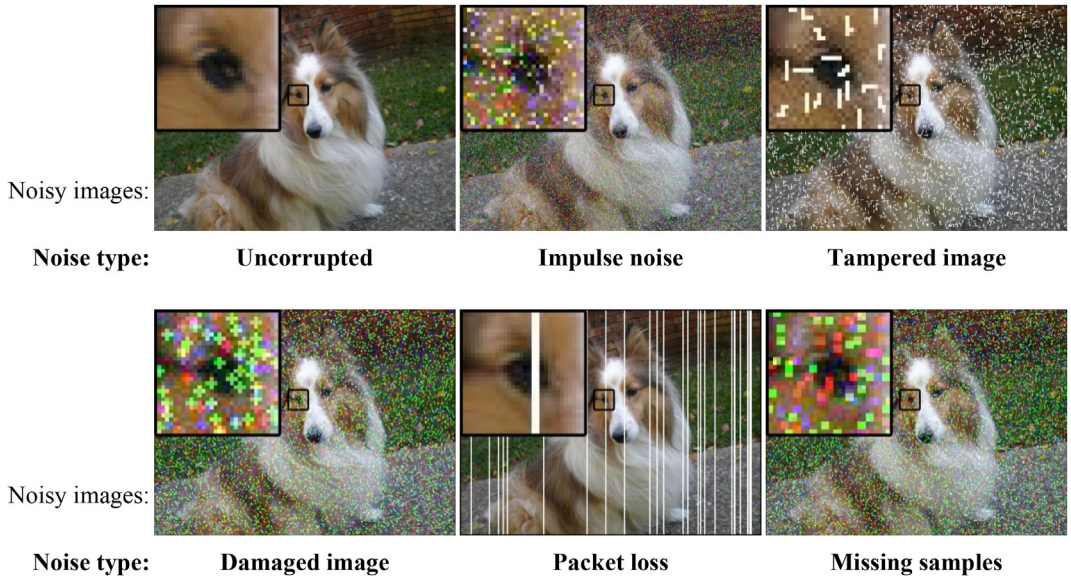

Fig. 13. A variety of noises with 20% density. Images are magnified.

stride map in each convolutional layer based on the noise map causes that noisy pixels are not considered in the classification process. The adaptive pooling layer in the proposed NR-CNN prevents the processing of noisy pixels using the pooling operator.

Table 3 shows the classification results of validation set with 10%–50% noise using VGG-Net-Medium and proposed NR-CNN. The results demonstrate that the proposed NR-CNN has better perfomance than VGG-Net-Medium for classification of noisy images. The classification results of the validation set with 10%–50% noise using VGG-Net-Slow and NR-CNN are shown in Table 4. The results indicate the superiority of the proposed NR-CNN in the classification of noisy images.

3.3. Comparison of the proposed NR-CNN with GoogleNet and ResNet

To further investigate the performance of the proposed NR-CNN, several experiments are performed to compare it with GoogleNet [41] and ResNet [42] in classification of the noisy images. Table 5 shows the

Table 1
Configuration of the proposed NR-CNN based on the VGG-Net-Medium model.

Layer	VGG-Net-Medium	NR-CNN	Size	Number	Stride	Pading
1	-	**Noise detection**	–	–	–	–
2	Input	**Adaptive resize,** Input	–	–	–	–
3	Convolution	**Adaptive convolution**	7 × 7	96	2	0
4,5	ReLU, LRN	ReLU, LRN	–	–	–	–
6	Max-pooling	**Adaptive max-pooling**	3 × 3	–	2	0
7	Convolution	Convolution	5 × 5	256	2	1
8,9	ReLU, LRN	ReLU, LRN	–	–	–	–
10	Max-pooling	Max-pooling	3 × 3	–	2	0
11	Convolution	Convolution	3 × 3	512	1	1
12	ReLU	ReLU	–	–	–	–
13	Convolution	Convolution	3 × 3	512	1	1
14	ReLU	ReLU	–	–	–	–
15	Convolution	Convolution	3 × 3	512	1	1
16	ReLU	ReLU	–	–	–	–
17	Max-pooling	Max-pooling	3 × 3	–	2	0
18	Fully connected	Fully connected	–	4096	1	0
19	ReLU	ReLU	–	–	–	–
20	Fully connected	Fully connected	–	4096	1	0
21	ReLU	ReLU	–	–	–	–
22	Fully connected	Fully connected	–	1000	1	0
23	Loss function	Loss function	–	–	–	–

Table 2
Configuration of the proposed NR-CNN based on the VGG-Net-Slow model.

Layer	VGG-Net-Slow	NR-CNN	Size	Number	Stride	Pading
1	–	**Noise detection**	–	–	–	–
2	Input	**Adaptive resize,** Input	–	–	–	–
3	Convolution	**Adaptive convolution**	7 × 7	96	2	0
4,5	ReLU, LRN	ReLU, LRN	–	–	–	–
6	Max-pooling	**Adaptive max-pooling**	3 × 3	–	3	0
7	Convolution	Convolution	5 × 5	256	1	0
8	ReLU	ReLU	–	–	–	–
9	Max-pooling	Max-pooling	2 × 2	–	2	0
10	Convolution	Convolution	3 × 3	512	1	1
11	ReLU	ReLU	–	–	–	–
12	Convolution	Convolution	3 × 3	512	1	1
13	ReLU	ReLU	–	–	–	–
14	Convolution	Convolution	3 × 3	512	1	1
15	ReLU	ReLU	–	–	–	–
16	Max-pooling	Max-pooling	3 × 3	–	3	0
17	Fully connected	Fully connected	–	4096	1	0
18	ReLU	ReLU	–	–	–	–
19	Fully connected	Fully connected	–	4096	1	0
20	ReLU	ReLU	–	–	–	–
21	Fully connected	Fully connected	–	1000	1	0
22	Loss function	Loss function	–	–	1	0

Table 3
The classification result (Top-5 error) of noisy images using VGG-Net-Medium and NR-CNN.

Noise type	VGG-Net-M					NR-CNN				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Impulse noise	0.45	0.69	0.85	0.93	0.96	0.19	0.2	0.21	0.22	0.24
Tampered image	0.38	0.51	0.62	0.71	0.80	0.19	0.22	0.24	0.27	0.30
Damaged image	0.36	0.50	0.62	0.70	0.76	0.19	0.21	0.23	0.24	0.25
Packet loss	0.47	0.65	0.76	0.84	0.90	0.18	0.19	0.19	0.20	0.22
Missing samples	0.34	0.49	0.60	0.70	0.77	0.19	0.21	0.23	0.24	0.26

Table 4
The classification performance (Top-5 error) of noisy images using VGG-Net-Slow and NR-CNN.

Noise type	VGG-Net-S					NR-CNN				
	10%	20%	30%	40%	50%	10%	20%	30%	40%	50%
Impulse noise	0.47	0.72	0.87	0.94	0.98	0.18	0.20	0.21	0.22	0.23
Tampered image	0.38	0.51	0.62	0.72	0.81	0.19	0.21	0.24	0.26	0.29
Damaged image	0.36	0.51	0.62	0.70	0.77	0.19	0.21	0.22	0.24	0.25
Packet loss	0.49	0.65	0.77	0.85	0.91	0.18	0.18	0.19	0.20	0.22
Missing samples	0.34	0.49	0.61	0.70	0.77	0.19	0.21	0.22	0.24	0.26

Table 5
The configuration of the proposed NR-CNN to compare with GoogleNet.

#	GoogleNet	NR-CNN	Pad	Stride	#	GoogleNet	NR-CNN	Pad	Stride	#	GoogleNet	NR-CNN	Pad	Stride
0	–	**Noise detection**	-	–	52	Conv	Conv	0	1	104	Conv	Conv	0	1
1	Input	**Adaptive Resize, Input**	-	–	53	ReLU	ReLU	–	–	105	ReLU	ReLU	–	–
2	Conv	**Adaptive Conv**	3	2	54	Concat	Concat	–	–	106	Conv	Conv	0	1
3	ReLU	ReLU	–	–	55	Pooling	Pooling	0x2x0x2	3	107	ReLU	ReLU	–	–
4	Pooling	**Adaptive pool**	0x1x0x1	2	56	Conv	Conv	0	1	108	Conv	Conv	0	1
5	LRN	LRN	–	–	57	ReLU	ReLU	–	–	109	Conv	Conv	0	1
6	Conv	Conv	0	1	58	Conv	Conv	0	1	110	ReLU	ReLU	–	–
7	ReLU	ReLU	–	–	59	ReLU	ReLU	–	–	111	Conv	Conv	0	1
8	Conv	Conv	1	1	60	Conv	Conv	0	1	112	ReLU	ReLU	–	–
9	ReLU	ReLU	–	–	61	Conv	Conv	0	1	113	Pooling	Pooling	1	1
10	LRN	LRN	–	–	62	ReLU	ReLU	–	–	114	Conv	Conv	0	1
11	Pooling	Pooling	0x1x0x1	2	63	Conv	Conv	0	1	115	ReLU	ReLU	–	–
12	Conv	Conv	0	1	64	ReLU	ReLU	–	–	116	Conv	Conv	1	1
13	ReLU	ReLU	–	–	65	Pooling	Pooling	1	1	117	ReLU	ReLU	–	–
14	Conv	Conv	0	1	66	Conv	Conv	0	1	118	Conv	Conv	2	1
15	ReLU	ReLU	–	–	67	ReLU	ReLU	–	–	119	ReLU	ReLU	–	–
16	Pooling	Pooling	1	1	68	Conv	Conv	1	1	120	Conv	Conv	0	1
17	Conv	Conv	0	1	69	ReLU	ReLU	–	–	121	ReLU	ReLU	–	–
18	ReLU	ReLU	–	–	70	Conv	Conv	2	1	122	Concat	Concat	–	–
19	Conv	Conv	1	1	71	ReLU	ReLU	–	–	123	Pooling	Pooling	0x1x0x1	2
20	ReLU	ReLU	–	–	72	Conv	Conv	0	1	124	Conv	Conv	0	1
21	Conv	Conv	2	1	73	ReLU	ReLU	–	–	125	ReLU	ReLU	–	–
22	ReLU	ReLU	–	–	74	Concat	Concat	–	–	126	Conv	Conv	0	1
23	Conv	Conv	0	1	75	Conv	Conv	0	1	127	ReLU	ReLU	–	–
24	ReLU	ReLU	–	–	76	ReLU	ReLU	–	–	128	Pooling	Pooling	1	1
25	Concat	Concat	–	–	77	Conv	Conv	0	1	129	Conv	Conv	0	1
26	Conv	Conv	0	1	78	ReLU	ReLU	–	–	130	ReLU	ReLU	–	–
27	ReLU	ReLU	–	–	79	Pooling	Pooling	1	1	131	Conv	Conv	1	1
28	Conv	Conv	0	1	80	Conv	Conv	0	1	132	ReLU	ReLU	–	–
29	ReLU	ReLU	–	–	81	ReLU	ReLU	–	–	133	Conv	Conv	2	1
30	Pooling	Pooling	1	1	82	Conv	Conv	1	1	134	ReLU	ReLU	–	–
31	Conv	Conv	0	1	83	ReLU	ReLU	–	–	135	Conv	Conv	0	1
32	ReLU	ReLU	–	–	84	Conv	Conv	2	1	136	ReLU	ReLU	–	–
33	Conv	Conv	1	1	85	ReLU	ReLU	–	–	137	Concat	Concat	–	–
34	ReLU	ReLU	–	–	86	Conv	Conv	0	1	138	Conv	Conv	0	1
35	Conv	Conv	2	1	87	ReLU	ReLU	–	–	139	ReLU	ReLU	–	–
36	ReLU	ReLU	–	–	88	Concat	Concat	–	–	140	Conv	Conv	0	1
37	Conv	Conv	0	1	89	Conv	Conv	0	1	141	ReLU	ReLU	–	–
38	ReLU	ReLU	–	–	90	ReLU	ReLU	–	–	142	Pooling	Pooling	1	1
39	Concat	Concat	–	–	91	Conv	Conv	0	1	143	Conv	Conv	0	1
40	Pooling	Pooling	0x1x0x1	2	92	ReLU	ReLU	–	–	144	ReLU	ReLU	–	–
41	Conv	Conv	0	1	93	Pooling	Pooling	1	1	145	Conv	Conv	1	1
42	ReLU	ReLU	–	–	94	Conv	Conv	0	1	146	ReLU	ReLU	–	–
43	Conv	Conv	0	1	95	ReLU	ReLU	–	–	147	Conv	Conv	2	1
44	ReLU	ReLU	–	–	96	Conv	Conv	1	1	148	ReLU	ReLU	–	–

(*continued on next page*)

Table 5 (*continued*)

#	GoogleNet	NR-CNN	Pad	Stride	#	GoogleNet	NR-CNN	Pad	Stride	#	GoogleNet	NR-CNN	Pad	Stride
45	Pooling	Pooling	1	1	97	ReLU	ReLU	–	–	149	Conv	Conv	0	1
46	Conv	Conv	0	1	98	Conv	Conv	2	1	150	ReLU	ReLU	–	–
47	ReLU	ReLU	–	–	99	ReLU	ReLU	–	–	151	Concat	Concat	–	–
48	Conv	Conv	1	1	100	Conv	Conv	0	1	152	Pooling	Pooling	0	1
49	ReLU	ReLU	–	–	101	ReLU	ReLU	–	–	153	Conv	Conv	0	1
50	Conv	Conv	2	1	102	Concat	Concat	–	–	154	SoftMax	Loss Function	–	–
51	ReLU	ReLU	–	–	103	Pooling	Pooling	0x2x0x2	3					

Table 6
The configuration of the proposed NR-CNN to compare with ResNet.

#	ResNet	NR-CNN	Pad	Stride	#	ResNet	NR-CNN	Pad	Stride	#	ResNet	NR-CNN	Pad	Stride
0	–	**Noise Detection**	–	–	59	ReLU	ReLU	–	–	118	Conv	Conv	0	1
1	Input	**Adaptive Resize, Input**	–	–	60	Conv	Conv	0	1	119	BatchNorm	BatchNorm	–	–
2	Conv	**Adaptive Conv**	3	2	61	BatchNorm	BatchNorm	–	–	120	Sum	Sum	–	–
3	BatchNorm	BatchNorm	–	–	62	ReLU	ReLU	–	–	121	ReLU	ReLU	–	–
4	ReLU	ReLU	–	–	63	Conv	Conv	1	1	122	Conv	Conv	0	1
5	Pooling	**Adaptive Pooling**	0x1x0x1	2	64	BatchNorm	BatchNorm	–	–	123	BatchNorm	BatchNorm	–	–
6	Conv	Conv	0	1	65	ReLU	ReLU	–	–	124	ReLU	ReLU	–	–
7	BatchNorm	BatchNorm	–	–	66	Conv	Conv	0	1	125	Conv	Conv	1	1
8	Conv	Conv	0	1	67	BatchNorm	BatchNorm	–	–	126	BatchNorm	BatchNorm	–	–
9	BatchNorm	BatchNorm	–	–	68	Sum	Sum	–	–	127	ReLU	ReLU	–	–
10	ReLU	ReLU	–	–	69	ReLU	ReLU	–	–	128	Conv	Conv	0	1
11	Conv	Conv	1	1	70	Conv	Conv	0	1	129	BatchNorm	BatchNorm	–	–
12	BatchNorm	BatchNorm	–	–	71	BatchNorm	BatchNorm	–	–	130	Sum	Sum	–	–
13	ReLU	ReLU	–	–	72	ReLU	ReLU	–	–	131	ReLU	ReLU	–	–
14	Conv	Conv	0	1	73	Conv	Conv	1	1	132	Conv	Conv	0	1
15	BatchNorm	BatchNorm	–	–	74	BatchNorm	BatchNorm	–	–	133	BatchNorm	BatchNorm	–	–
16	Sum	Sum	–	–	75	ReLU	ReLU	–	–	134	ReLU	ReLU	–	–
17	ReLU	ReLU	–	–	76	Conv	Conv	0	1	135	Conv	Conv	1	1
18	Conv	Conv	0	1	77	BatchNorm	BatchNorm	–	–	136	BatchNorm	BatchNorm	–	–
19	BatchNorm	BatchNorm	–	–	78	Sum	Sum	–	–	137	ReLU	ReLU	–	–
20	ReLU	ReLU	–	–	79	ReLU	ReLU	–	–	138	Conv	Conv	0	1
21	Conv	Conv	1	1	80	Conv	Conv	0	2	139	BatchNorm	BatchNorm	–	–
22	BatchNorm	BatchNorm	–	–	81	BatchNorm	BatchNorm	–	–	140	Sum	Sum	–	–
23	ReLU	ReLU	–	–	82	Conv	Conv	0	2	141	ReLU	ReLU	–	–
24	Conv	Conv	0	1	83	BatchNorm	BatchNorm	–	–	142	Conv	Conv	0	2
25	BatchNorm	BatchNorm	–	–	84	ReLU	ReLU	–	–	143	BatchNorm	BatchNorm	–	–
26	Sum	Sum	–	–	85	Conv	Conv	1	1	144	Conv	Conv	0	2
27	ReLU	ReLU	–	–	86	BatchNorm	BatchNorm	–	–	145	BatchNorm	BatchNorm	–	–
28	Conv	Conv	0	1	87	ReLU	ReLU	–	–	146	ReLU	ReLU	–	–
29	BatchNorm	BatchNorm	–	–	88	Conv	Conv	0	1	147	Conv	Conv	1	1
30	ReLU	ReLU	–	–	89	BatchNorm	BatchNorm	–	–	148	BatchNorm	BatchNorm	–	–
31	Conv	Conv	1	1	90	Sum	Sum	–	–	149	ReLU	ReLU	–	–
32	BatchNorm	BatchNorm	–	–	91	ReLU	ReLU	–	–	150	Conv	Conv	0	1
33	ReLU	ReLU	–	–	92	Conv	Conv	0	1	151	BatchNorm	BatchNorm	–	–
34	Conv	Conv	0	1	93	BatchNorm	BatchNorm	–	–	152	Sum	Sum	–	–
35	BatchNorm	BatchNorm	–	–	94	ReLU	ReLU	–	–	153	ReLU	ReLU	–	–
36	Sum	Sum	–	–	95	Conv	Conv	1	1	154	Conv	Conv	0	1
37	ReLU	ReLU	–	–	96	BatchNorm	BatchNorm	–	–	155	BatchNorm	BatchNorm	–	–
38	Conv	Conv	0	2	97	ReLU	ReLU	–	–	156	ReLU	ReLU	–	–
39	BatchNorm	BatchNorm	–	–	98	Conv	Conv	0	1	157	Conv	Conv	1	1
40	Conv	Conv	0	2	99	BatchNorm	BatchNorm	–	–	158	BatchNorm	BatchNorm	–	–
41	BatchNorm	BatchNorm	–	–	100	Sum	Sum	–	–	159	ReLU	ReLU	–	–
42	ReLU	ReLU	–	–	101	ReLU	ReLU	–	–	160	Conv	Conv	0	1
43	Conv	Conv	1	1	102	Conv	Conv	0	1	161	BatchNorm	BatchNorm	–	–
44	BatchNorm	BatchNorm	–	–	103	BatchNorm	BatchNorm	–	–	162	Sum	Sum	–	–
45	ReLU	ReLU	–	–	104	ReLU	ReLU	–	–	163	ReLU	ReLU	–	–
46	Conv	Conv	0	1	105	Conv	Conv	1	1	164	Conv	Conv	0	1
47	BatchNorm	BatchNorm	–	–	106	BatchNorm	BatchNorm	–	–	165	BatchNorm	BatchNorm	–	–
48	Sum	Sum	–	–	107	ReLU	ReLU	–	–	166	ReLU	ReLU	–	–
49	ReLU	ReLU	–	–	108	Conv	Conv	0	1	167	Conv	Conv	1	1
50	Conv	Conv	0	1	109	BatchNorm	BatchNorm	–	–	168	BatchNorm	BatchNorm	–	–
51	BatchNorm	BatchNorm	–	–	110	Sum	Sum	–	–	169	ReLU	ReLU	–	–
52	ReLU	ReLU	–	–	111	ReLU	ReLU	–	–	170	Conv	Conv	0	1
53	Conv	Conv	1	1	112	Conv	Conv	0	1	171	BatchNorm	BatchNorm	–	–
54	BatchNorm	BatchNorm	–	–	113	BatchNorm	BatchNorm	–	–	172	Sum	Sum	–	–
55	ReLU	ReLU	–	–	114	ReLU	ReLU	–	–	173	ReLU	ReLU	–	–
56	Conv	Conv	0	1	115	Conv	Conv	1	1	174	Pooling	Pooling	0	1
57	BatchNorm	BatchNorm	–	–	116	BatchNorm	BatchNorm	–	–	175	Conv	Conv	0	1
58	Sum	Sum	–	–	117	ReLU	ReLU	–	–	176	SoftMax	Loss function	–	–

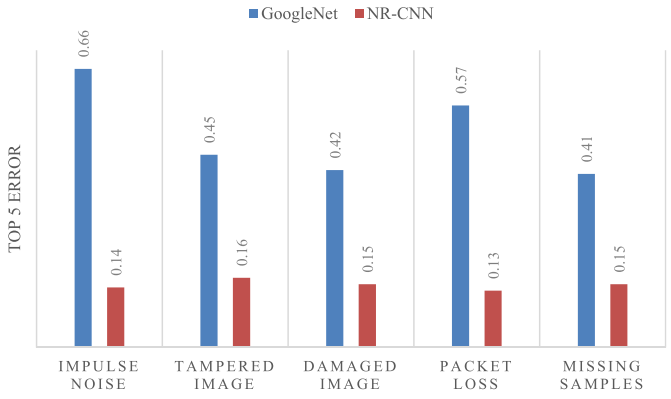

Fig. 14. The classification results of the images corrupted with 25% noise density using GoogleNet and proposed NR-CNN.

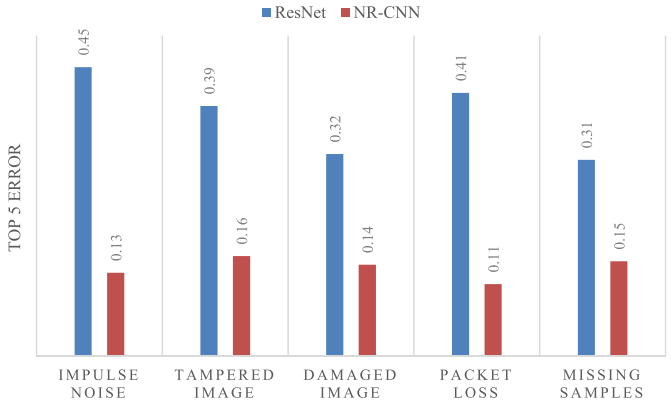

Fig. 15. The classification results of the images corrupted with 25% noise density using ResNet and proposed NR-CNN.

Table 7
The first custom configuration of the proposed NR-CNN to compare with CNN.

Layer	CNN	NR-CNN	Size	Number	Stride
0	–	**Noise detection**	–	–	–
1	Input	**Adaptive resize, Input**	–	–	–
2	Convolution	**Adaptive Convolution**	5*5	20	1
3	Max pooling	**Adaptive Max pooling**	2*2	–	2
4	Convolution	Convolution	5*5	50	1
5	Max pooling	Max pooling	2*2	–	2
6	Convolution	Convolution	4*4	500	1
7	ReLU	ReLU	–	–	1
8	Fully connected	Fully connected	1*1	10	1
9	Loss function	Adaptive loss function	–	–	–

configuration of the proposed NR-CNN to compare with GoogleNet. In Table 6, the configuration of the proposed NR-CNN for comparison with ResNet is demonstrated. The input images are resized to 224*224 by the proposed adaptive resize method. Adaptive convolution and adaptive pooling is used in the related first layer. For training the proposed NR-CNN, the ILSVRC-2015 dataset containing uncorrupted images of 1000 classes [37] is used. The error rate (Top-5 error) in the classification is used as the evaluation criterion for model testing. The results of the proposed NR-CNN, GoogleNet and ResNet for classification of the noisy images are shown in Figs. 14 and 15, respectively. As can be seen from Fig. 15, the proposed NR-CNN using the ResNet configuration outperforms other models. The results of Figs. 14 and 15 indicate that the proposed NR-CNN using both configurations has better performance compared to GoogleNet and ResNet for classification of the noisy validation set.

Table 8
The second custom configuration of the proposed NR-CNN to compare with CNN.

Layer	CNN	NR-CNN	Size	Number	Stride
0	–	**Noise detection**	–	–	–
1	Input	**Adaptive resize, Input**	–	–	–
2	Convolution	**Adaptive Convolution**	5*5	20	1
3	Max pooling	**Adaptive Max pooling**	2*2	–	2
4	Convolution	Convolution	5*5	50	1
5	Max pooling	Max pooling	2*2	–	1
6	ReLU	ReLU	–	–	1
7	Convolution	Convolution	4*4	60	1
8	Max pooling	Max pooling	2*2	–	1
9	Convolution	Convolution	3*3	500	1
10	Max pooling	Max pooling	2*2	–	1
11	ReLU	ReLU	–	–	1
12	Fully connected	Fully connected	1*1	10	1
13	Loss function	Adaptive loss function	–	–	–

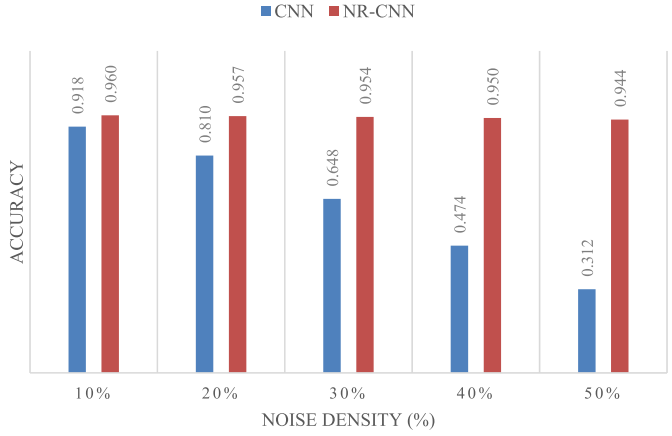

Fig. 16. The classification results of the images corrupted by impulse noise using first custom configuration (Table 7).

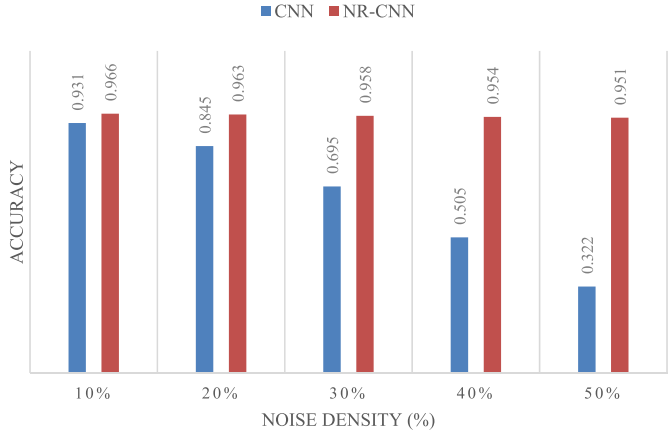

Fig. 17. The classification results of the images corrupted by impulse noise using second custom configuration (Table 8).

3.4. Comparison of NR-CNN with custom configuration of CNN

To further investigate the performance of the proposed NR-CNN, several experiments are performed to compare it with other models using two different configurations in classification of the images corrupted by impulse noise.

Table 7 shows the first configuration of the proposed NR-CNN to compare with CNN. In this configuration, the proposed NR-CNN consists of 3 convolutional layers, 2 pooling layers, 1 ReLU layer and 1 fully

connected layer. In Table 8, the second configuration of the proposed NR-CNN for comparison with CNN is demonstrated. In the second configuration, the proposed NR-CNN contains 4 convolutional layers, 4 pooling layers, 2 ReLU layer and 1 fully connected layer. In both configurations, the input images are resized from 128*128 to 28*28 by the proposed adaptive resize method. The noise map is updated in each layer. To train the proposed model, the MNIST database [45] is used, which includes 70, 000 images with 10 classes. In the experiments, 60,000 uncorrupted images are used for training set and 10,000 noisy images for test set. The accuracy in the classification is used as the evaluation criterion for model testing. As can be seen from Fig. 16 and Fig. 17, the proposed NR-CNN using the second configuration outperforms other models. Also, the proposed NR-CNN has better performance compared to CNN for classification of the images corrupted by impulse noise.

4. Conclusion and future researches

In this paper, we proposed a convolution neural network (NR-CNN) which is robust to a variety of noises. The Proposed NR-CNN classifies the noisy images without any preprocessing for noise removal. The images corrupted by impulse noise, missing image samples, packet loss in image transmission, damaged images and tampered images were used to consider the robustness of the proposed CNN to noise. To robust the proposed CNN to noise, a noise map layer and an adaptive resize layer were added to the convolutional neural network. Moreover, the adaptive convolution layer, the adaptive pooling layer were introduced to consider the noise problem in different components of the proposed CNN. To improve the classification performance of the proposed CNN, the adaptive data augmentation based on noise map was also provided. Experimental results indicated that the proposed CNN has better performance in classification of noisy images compared with VGG-Net-Medium,VGG-Net-Slow, GoogleNet and ResNet. Moreover, the proposed CNN requires no preprocessing for noise removal, which speeds up the classification of noisy images based on CNN.

In the future, the aim is to use the proposed noise-robust convolutional neural network for object detection in noisy images. Moreover, other fundamental noises such as Gaussian noise, Poisson noise and Speckle noise will be used in the images to be classified by the proposed method.

CRediT author statement

Mohammad Momeny: Conceptualization, Methodology, Investigation, Data curation, Software, Validation, Writing - original draft. Ali Mohammad Latif: Investigation, Validation, Writing - review & editing. Mehdi Agha Sarram: Supervision, Writing - review & editing. Razieh Sheikhpour: Writing - review & editing. Yu Dong Zhang: Validation, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] M. Shariati, M.S. Mafipour, P. Mehrabi, A. Shariati, A. Toghroli, N.T. Trung, M.N.A. Salih, A novel approach to predict shear strength of tilted angle connectors using artificial intelligence techniques, Eng. Comput. 1 (2020) 1–21, https://doi.org/10.1007/s00366-019-00930-x.

[2] M. Shariati, M.S. Mafipour, P. Mehrabi, M. Ahmadi, K. Wakil, N.T. Trung, A. Toghroli, Prediction of concrete strength in presence of furnace slag and fly ash using Hybrid ANN-GA (Artificial Neural Network-Genetic Algorithm), Smart Struct. Syst. 25 (2020) 183–195, https://doi.org/10.12989/sss.2020.25.2.183.

[3] M. Shariati, M.S. Mafipour, J.H. Haido, S.T. Yousif, A. Toghroli, N.T. Trung, A. Shariati, Identification of the most influencing parameters on the properties of corroded concrete beams using an Adaptive Neuro-Fuzzy Inference System (ANFIS), Comput. Concr. 25 (2020) 155–170, https://doi.org/10.12989/cac.2020.25.1.083.

[4] M. Shariati, M.S. Mafipour, B. Ghahremani, F. Azarhomayun, M. Ahmadi, N.T. Trung, A. Shariati, A novel hybrid extreme learning machine–grey wolf optimizer (ELM-GWO) model to predict compressive strength of concrete with partial replacements for cement, Eng. Comput. 1 (2020) 1–21, https://doi.org/10.1007/s00366-020-01081-0.

[5] M. Safa, P.A. Sari, M. Shariati, M. Suhatril, N.T. Trung, K. Wakil, M. Khorami, Development of neuro-fuzzy and neuro-bee predictive models for prediction of the safety factor of eco-protection slopes, Phys. A Stat. Mech. Its Appl. 550 (2020), https://doi.org/10.1016/j.physa.2019.124046.

[6] M. Shariati, M.S. Mafipour, P. Mehrabi, Y. Zandi, D. Dehghani, A. Bahadori, A. Shariati, N.T. Trung, M.N.A. Salih, S. Poi-Ngian, Application of Extreme Learning Machine (ELM) and Genetic Programming (GP) to design steel-concrete composite floor systems at elevated temperatures, Steel Compos. Struct. 33 (2019) 319–332, https://doi.org/10.12989/scs.2019.33.3.319.

[7] M. Shariati, M.S. Mafipour, P. Mehrabi, A. Bahadori, Y. Zandi, M.N.A. Salih, H. Nguyen, J. Dou, X. Song, S. Poi-Ngian, Application of a hybrid artificial neural network-particle swarm optimization (ANN-PSO) model in behavior prediction of channel shear connectors embedded in normal and high-strength concrete, Appl. Sci. 9 (2019) 5534, https://doi.org/10.3390/app9245534.

[8] A. Jahanbakhshi, K. Kheiralipour, Evaluation of image processing technique and discriminant analysis methods in postharvest processing of carrot fruit, Food Sci. Nutr. 8 (2020) 3346–3352, https://doi.org/10.1002/fsn3.1614.

[9] H. Azarmdel, A. Jahanbakhshi, S.S. Mohtasebi, A.R. Muñoz, Evaluation of image processing technique as an expert system in mulberry fruit grading based on ripeness level using artificial neural networks (ANNs) and support vector machine (SVM), Postharvest Biol. Technol. 166 (2020), https://doi.org/10.1016/j.postharvbio.2020.111201.

[10] T. Liang, X. Xu, P. Xiao, A new image classification method based on modified condensed nearest neighbor and convolutional neural networks, Pattern Recogn. Lett. 94 (2017) 105–111, https://doi.org/10.1016/j.patrec.2017.05.019.

[11] C. Barat, C. Ducottet, String representations and distances in deep Convolutional Neural Networks for image classification, Pattern Recogn. 54 (2016) 104–115, https://doi.org/10.1016/j.patcog.2016.01.007.

[12] K. Nogueira, O.A.B. Penatti, J.A. dos Santos, Towards better exploiting convolutional neural networks for remote sensing scene classification, Pattern Recogn. 61 (2017) 539–556, https://doi.org/10.1016/j.patcog.2016.07.001.

[13] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, T. Chen, Recent advances in convolutional neural networks, Pattern Recogn. 77 (2018) 354–377, https://doi.org/10.1016/j.patcog.2017.10.013.

[14] S. Ji, W. Xu, M. Yang, K. Yu, 3D Convolutional neural networks for human action recognition, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 221–231, https://doi.org/10.1109/TPAMI.2012.59.

[15] G. Lin, Q. Wu, L. Qiu, X. Huang, Image super-resolution using a dilated convolutional neural network, Neurocomputing 275 (2018) 1219–1230, https://doi.org/10.1016/j.neucom.2017.09.062.

[16] S. Yu, S. Jia, C. Xu, Convolutional neural networks for hyperspectral image classification, Neurocomputing 219 (2017) 88–98, https://doi.org/10.1016/j.neucom.2016.09.010.

[17] J. Yim, K.A. Sohn, Enhancing the performance of convolutional neural networks on quality degraded datasets, DICTA 2017 - 2017 Int. Conf. Digit. Image Comput. Tech. Appl. (2017) (2017-Decem) 1–8, https://doi.org/10.1109/DICTA.2017.8227427.

[18] M.T. Islam, S.M. Mahbubur Rahman, M. Omair Ahmad, M.N.S. Swamy, Mixed Gaussian-impulse noise reduction from images using convolutional neural network, Signal Process. Image Commun. 68 (2018) 26–41, https://doi.org/10.1016/j.image.2018.06.016.

[19] S. Gai, Z. Bao, New image denoising algorithm via improved deep convolutional neural network with perceptive loss, Expert Syst. Appl. 138 (2019) 112815, https://doi.org/10.1016/j.eswa.2019.07.032.

[20] L. Jin, W. Zhang, G. Ma, E. Song, Learning deep CNNs for impulse noise removal in images, J. Vis. Commun. Image Represent. 62 (2019) 193–205, https://doi.org/10.1016/j.jvcir.2019.05.005.

[21] W. Zhang, L. Jin, E. Song, X. Xu, Removal of impulse noise in color images based on convolutional neural network, Appl. Soft Comput. J. 82 (2019), https://doi.org/10.1016/j.asoc.2019.105558.

[22] I. Turkmen, The ANN based detector to remove random-valued impulse noise in images, J. Vis. Commun. Image Represent. 34 (2016) 28–36, https://doi.org/10.1016/j.jvcir.2015.10.011.

[23] Z. Zhang, D. Han, J. Dezert, Y. Yang, A new adaptive switching median filter for impulse noise reduction with pre-detection based on evidential reasoning, Signal Process. 147 (2018) 173–189, https://doi.org/10.1016/j.sigpro.2018.01.027.

[24] A. Javaheri, H. Zayyani, F. Marvasti, Sparse recovery of missing image samples using a convex similarity index, Signal Process. 152 (2018) 90–103, https://doi.org/10.1016/j.sigpro.2018.05.022.

[25] B. Dong, H. Ji, J. Li, Z. Shen, Y. Xu, Wavelet frame based blind image inpainting, Appl. Comput. Harmon. Anal. 32 (2012) 268–279, https://doi.org/10.1016/j.acha.2011.06.001.

[26] V.K. Alilou, F. Yaghmaee, Non-texture image inpainting using histogram of oriented gradients, J. Vis. Commun. Image Represent. 48 (2017) 43–53, https://doi.org/10.1016/j.jvcir.2017.06.003.

[27] D. Shabtay, N. Raviv, Y. Moshe, Video packet loss concealment detection based on image content, in: Eur. Signal Process. Conf., 2008. https://ieeexplore.ieee.org/abstract/document/7080325/. (Accessed 21 September 2018).

[28] G. Nikolakopoulos, P. Stavrou, D. Tsitsipis, D. Kandris, A. Tzes, T. Theocharis, A dual scheme for compression and restoration of sequentially transmitted images

over Wireless Sensor Networks, Ad Hoc Netw. 11 (2013) 410–426, https://doi.org/10.1016/j.adhoc.2012.07.003.
[29] R.G. Everitt, R.H. Glendinning, A statistical approach to the problem of restoring damaged and contaminated images, Pattern Recogn. 42 (2009) 115–125, https://doi.org/10.1016/j.patcog.2008.06.009.
[30] H. Li, W. Luo, J. Huang, Localization of diffusion-based inpainting in digital images, IEEE Trans. Inf. Forensics Secur. 12 (2017) 3050–3064, https://doi.org/10.1109/TIFS.2017.2730822.
[31] A.C. Kokaram, On missing data treatment for degraded video and film archives: a survey and a new Bayesian approach, IEEE Trans. Image Process. 13 (2004) 397–415, https://doi.org/10.1109/TIP.2004.823815.
[32] C. Qin, C.C. Chang, K.N. Chen, Adaptive self-recovery for tampered images based on VQ indexing and inpainting, Signal Process. 93 (2013) 933–946, https://doi.org/10.1016/j.sigpro.2012.11.013.
[33] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (2017) 84–90, https://doi.org/10.1145/3065386.
[34] X. Xiao, N.N. Xiong, J. Lai, C.-D. Wang, Z. Sun, J. Yan, A local consensus index scheme for random-valued impulse noise detection systems, IEEE Trans. Syst. Man, Cybern. Syst. (2019) 1–17, https://doi.org/10.1109/tsmc.2019.2925886.
[35] A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, Commun. ACM 60 (2017) 84–90, https://doi.org/10.1145/3065386.
[36] R.C. O'Reilly, D. Wyatte, S. Herd, B. Mingus, D.J. Jilk, Recurrent processing during object recognition, Front. Psychol. 4 (2013) 1–14, https://doi.org/10.3389/fpsyg.2013.00124.
[37] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A.C. Berg, L. Fei-Fei, ImageNet large scale visual recognition challenge, Int. J. Comput. Vis. 115 (2015) 211–252, https://doi.org/10.1007/s11263-015-0816-y.
[38] K.H. Jin, J.C. Ye, Sparse and low-rank decomposition of a hankel structured matrix for impulse noise removal, IEEE Trans. Image Process. 27 (2018) 1448–1461, https://doi.org/10.1109/TIP.2017.2771471.
[39] I.F. Jafar, R.A. Alna'Mneh, K.A. Darabkh, Efficient improvements on the BDND filtering algorithm for the removal of high-density impulse noise, IEEE Trans. Image Process. 22 (2013) 1223–1232, https://doi.org/10.1109/TIP.2012.2228496.
[40] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc. (2015) 1–14. http://arxiv.org/abs/1409.1556. (Accessed 4 November 2018).
[41] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit, 2015, pp. 1–9, https://doi.org/10.1109/CVPR.2015.7298594.
[42] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2016-Decem (2016) 770–778, https://doi.org/10.1109/CVPR.2016.90.
[43] A. Jahanbakhshi, M. Momeny, M. Mahmoudi, Y.D. Zhang, Classification of sour lemons based on apparent defects using stochastic pooling mechanism in deep convolutional neural networks, Sci. Hortic. (Amsterdam) (2020) 263, https://doi.org/10.1016/j.scienta.2019.109133.
[44] M. Momeny, A. Jahanbakhshi, K. Jafarnezhad, Y.D. Zhang, Accurate classification of cherry fruit using deep CNN based on hybrid pooling approach, Postharvest Biol. Technol. 166 (2020), https://doi.org/10.1016/j.postharvbio.2020.111204.
[45] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition, Proc. IEEE 86 (1998) 2278–2323, https://doi.org/10.1109/5.726791.