August 5, 2019

A Deep Learning Framework for Joint Image Restoration and Recognition

Ruilong Chen, *University of Sheffield*
Lyudmila Mihaylova, *University of Sheffield*
Hao Zhu, *Chongqing University of Posts and Telecommunications*
Nidhal Carla Bouaynaya, *Rowan University*

Circuits, Systems, and Signal Processing (2020) 39:1561–1580
https://doi.org/10.1007/s00034-019-01222-x

A Deep Learning Framework for Joint Image Restoration and Recognition

Ruilong Chen[1] · Lyudmila Mihaylova[1] · Hao Zhu[2] · Nidhal Carla Bouaynaya[3]

Received: 1 September 2018 / Revised: 5 July 2019 / Accepted: 6 July 2019 / Published online: 5 August 2019

Abstract

Image restoration and recognition are important computer vision tasks representing an inherent part of autonomous systems. These two tasks are often implemented in a sequential manner, in which the restoration process is followed by a recognition. In contrast, this paper proposes a joint framework that simultaneously performs both tasks within a shared deep neural network architecture. This joint framework integrates the restoration and recognition tasks by incorporating: (i) common layers, (ii) restoration layers and (iii) classification layers. The total loss function combines the restoration and classification losses. The proposed joint framework, based on capsules, provides an efficient solution that can cope with challenges due to noise, image rotations and occlusions. The developed framework has been validated and evaluated on a public vehicle logo dataset under various degradation conditions, including Gaussian noise, rotation and occlusion. The results show that the joint framework improves the accuracy compared with the single task networks.

Keywords Capsule networks · Image restoration · Image recognition · Deep neural networks

✉ Lyudmila Mihaylova
l.s.mihaylova@sheffield.ac.uk

Ruilong Chen
chen8131928@gmail.com

Hao Zhu
haozhu1982@gmail.com

Nidhal Carla Bouaynaya
bouaynaya@rowan.edu

1 Department of Automatic Control and Systems Engineering, The University of Sheffield, Sheffield S1 3JD, UK

2 Chongqing University of Posts and Telecommunications, Chongqing 400065, People's Republic of China

3 Department of Electrical and Computer Engineering, Rowan University, Glassboro, NJ 08028, USA

1 Introduction

Image recognition is an important field, especially for autonomous systems. The field witnessed new opportunities and became very popular with the development of convolutional network networks (CNNs) in 2012 [15]. Before the creation of CNNs, the majority of image recognition approaches included a hand-crafted feature detection and feature description process [15,17]. Optimization algorithms for nonconvex data and image restoration are proposed in [4,5]. Compressed sensing algorithms incoprorating second-order derivatives in efficient ways are developed in [6], approaches based on conjugate priors in [7] and on supporting machines in [19]. In such approaches, image features are extracted at pixel level or regional level and then embedded in the recognition process [32]. Unlike traditional hand-crafted feature methods, such as the scale-invariant feature transform (SIFT) [17], CNNs automatically detect the image features by multiple convolutional and nonlinear operations. However, a CNN has a large number of parameters to learn, which requires large datasets.

Image restoration, on the other hand, aims to recover a clear image from its noisy, rotated and occluded version that the sensor provides. It is known as an ill-posed inverse problem [18] and has been a focus of significant research. The majority of works consider a super-resolution single image and its denoising. For example, the total variation [20] and BM3D algorithm [3] achieve good performance over single image denoising. The algorithms reported in [10,14,27,30] achieve state-of-the-art performance on a single image with super-resolution [9]. However, these methods are only applicable to particular types of image degradation. For example, the BM3D approach is designed only for image denoising.

Deep learning methods extract image features from groups of images and hence can be used for both image denoising and image restoration. In fact, deep neural networks take advantage of the availability of big data and the automatic learning process, which outperforms traditional image restoration methods [9,18]. Recently, deep neural networks have been developed for the purpose of image restoration. For example, Mao et al. [18] proposed a CNN architecture that could perform image denosing and construct super-resolution images. Being purely data driven, the approaches for learning restoration are promising as they do not need models nor assumptions about the nature of the degradation [18].

1.1 Related Convolutional Neural Network Approaches

Lecun et al. proposed the first CNN architecture, called LeNet [16], which lays down the beginning of the development of CNN architectures able to deal with big volumes of data and more and more complex inference tasks. Next, AlexNet achieved the best performance on ImageNet in 2012. Subsequently, different CNN architectures were developed rapidly and achieved higher than human performance on datasets, such as ImageNet [15].

Unlike neural networks, where neurons in each layer are fully connected to neurons in the next layer, each layer in a CNN shares the weights by using convolutional kernels. This process decreases tremendously the number of weights when compared

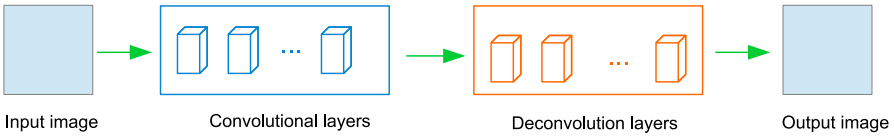

Fig. 1 A typical restoration architecture based on CNNs

with neural networks. Therefore, it can prevent the over-fitting problem, which is one of the main challenges in neural networks [25]. The CNN architectures are mainly composed of convolution operations, followed by nonlinearity (activation functions) and pooling operations.

The ZF-Net [31] architecture applies a smaller kernel size than AlexNet in the first layer. The reason behind this modification is that a smaller size in the first convolutional layer helps retain more original pixel information in the input volume. A large filtering size, e.g., $[11 \times 11]$, proved to be skipping relevant information from the input. ZF-Net achieved a smaller error rate than AlexNet on ImageNet [8].

The VGG-NET architechture [24] further enhances the depth of the CNNs up to 19 layers and uses a unique kernel size of $[3 \times 3]$. Google-Net [26] model increases the number of layers to 22 and applies an inception module, in which different convolutional feature maps, generated by convolutional kernels of different sizes, are combined at every layer. The Res-Net [11], built up with a 152-layer architecture, introduces the idea of residual learning, which builds shortcut connections between layers to mitigate the vanishing gradient problem and improve the optimization process. In 2015, Res-Net achieved the best accuracy on ImageNet.

Figure 1 shows the *generic coding–decoding architecture of a CNN for image restoration* [18]. The restoration framework is based on the typical convolutional operations of CNNs. The main difference is the lack of pooling and fully connected layers in the restoration process. The developed CNN approaches for restoration and recognition differ significantly from each other. The recognition process discards information, layer by layer, and finishes the process by providing a representative feature vector that is fed to neurons in the last fully connected layer. The *CNNs for recognition* discard information to extract the most representative feature in an image, while CNNs for restoration need to keep detailed information of the ground truth. In particular, the pooling process is not appropriate for restoration.

However, these networks and other state-of-the-art networks, such as RED-Net [18], VGG [24] and ResNet [11], face difficulties in dealing with image occlusion, rotation, denoising and super-resolution. This motivated us to develop efficient image restoration methods that particularly deal with rotation and occlusion. In addition, it would be beneficial if the restoration and recognition share a framework, which could jointly perform both tasks, rather than in a sequential manner (restoration followed by recognition).

From the wide variety of deep learning methods, we focus on capsule CNNs due to their robustness to rotation over convolutional CNNs. The idea of capsules has been proposed by Sabour et al. [21] to address known limitations of conventional CNNs with respect to rotation invariance. A *capsule* is a group of neurons, whose length

represents the probability of the object's (or part of the object) existence, and the orientation represents the instantiation parameters [21]. Previous works [2,21] show that the capsule network approach achieves better results on image recognition than conventional CNNs.

The main contributions of this work can be summarized as follows: 1) A deep learning architecture for joint image restoration and recognition is proposed based on capsules and conventional CNNs; 2) a new multi-loss function that combines the restoration and classification losses is proposed. A hyper-parameter is used to control the trade-off between the two loss functions. The linear combination of the restoration and classification losses can be viewed as a form of regularization as it constrains the search space for possible candidate solutions for each individual task; 3) a capsule CNN is proposed to handle image restoration from rotation and occlusion; 4) the developed framework is validated and its performance is evaluated using a public dataset.

The rest of this paper is organized as follows. Section 2 introduces the principles of the generic CNN and capsule network. Section 3 presents the developed joint framework for image restoration and recognition based on CNNs and capsule networks. Section 4 gives detailed evaluation of the performance of the proposed deep learning framework compared with state-of-the-art approaches. Finally, Sect. 5 summarizes the results.

2 Theoretical Background for Convolutional and Capsule Neural Networks

2.1 The Convolutional Neural Network Approach

Convolutional layers extract image feature maps by using different convolutional kernels. Suppose that n convolutional kernels are used in the k^{th} layer. Then the i^{th}, $i = 1, 2, \ldots, n$, convolutional feature map in the $(k+1)$ layer can be denoted as:

$$\mathcal{I}_i^{k+1} = f\left(\sum_j \mathbf{V}_i * \mathcal{I}_j^k\right), \tag{1}$$

where $\mathcal{I}_j$ is the j^{th} feature map and $\mathbf{V}_i$ is the i^{th} kernel. Here, $\mathcal{I}_j$ can be a channel of the original image, a pooling map or a convolutional map, $f(\cdot)$ denotes a nonlinear activation function and $*$ represents the convolution operation. The Rectified Linear Unit (ReLU) with the following nonlinear function $g(x) = \max(0, x)$ is often used [15] in CNNs.

A pooling process often decreases the size of the input feature maps. Hence, pooling can be regarded as a down-sampling operation. Each pooling map in layer $k+1$ is obtained by a pooling operation over the corresponding feature maps in the previous layer k and is given as follows:

$$\mathcal{I}_i^{k+1} = \text{pool}(\mathcal{I}_i^k), \tag{2}$$

where the index i goes though all maps in layer k and pool$(\cdot)$ represents the pooling method. A window shifts on the previous maps, and the mean value (or the maximum value) in each window is extracted to form a pooling map. The convolution and pooling operations are the two main operations in CNNs. The last layer is reshaped to a vector form and then fully connected with neurons in the same way as in generic neural networks. The loss function $\mathbb{L}_{\text{cla}}$, typically used in image recognition, is the cross-entropy function and is defined as follows:

$$\mathbb{L}_{\text{cla}} = \sum_i \left[-y_i \ln(\hat{y}_i) - (1 - y_i) \ln(1 - \hat{y}_i)\right], \tag{3}$$

where $\mathbf{y}$ and $\hat{\mathbf{y}}$ are, respectively, the ground truth label and the predicted label vectors for an image in the one-hot coded manner (a one-hot vector contains only one value equal to 1 and all other elements are zero valued), with the i^{th} entry denoted as y_i and $\hat{y}_i$.

2.2 The Capsule Network Approach

Connections between layers are of scalar-scalar type in generic CNNs. In a *capsule network* [21], neurons are combined in a group to represent an entity or part of an entity. In particular, a neuron is replaced with a group of neurons and the connections between capsule layers become of vector–vector type. For each capsule (represented as a vector), a *nonlinear squash function* $f(\cdot)$ is defined:

$$f(\mathbf{x}) = \frac{||\mathbf{x}||_2^2}{1 + ||\mathbf{x}||_2^2} \frac{\mathbf{x}}{||\mathbf{x}||_2}, \tag{4}$$

with $\mathbf{x}$ being the input vector of the squash function and $||\cdot||_2$ denotes the l_2-norm. This function makes the length of short vectors shrink close to 0 and long vectors expand close to 1. Hence, the output length can be used to represent the probability that an entity exists. The output $\mathbf{v}_j$ of the j-th capsule is given by:

$$\mathbf{v}_j = f(\mathbf{h}_j), \tag{5}$$

where $\mathbf{h}_j$ is the input of the j-th capsule. The parameters of each capsule represent various properties such as position, scale and orientation of a particular entity [21].

Except for the capsules in the first capsule layer, the total input $\mathbf{h}_j$ of the j-th capsule is a weighted sum:

$$\mathbf{h}_j = \sum_i c_{ij} \mathbf{o}_{j|i}, \tag{6}$$

where $\mathbf{o}_{j|i}$ is the predicted output of capsule j in the current layer given the input capsule i from the previous layer and c_{ij} are coefficients determined by a routing process.

Let q_{ij} denote the log prior probabilities that capsule i (in the previous layer) is coupled with capsule j (in the current layer). The coefficients c_{ij} can then be expressed as:

$$c_{ij} = \frac{\exp(q_{ij})}{\sum_d \exp(q_{id})}, \tag{7}$$

where the index d refers to all capsules in the current layer. q_{ij}s are initialized with zeros and updated by a routing algorithm. In the routing algorithm, q_{ij} is updated by the following process:

$$q_{ij}^{(r+1)} = q_{ij}^{(r)} + \left\langle \mathbf{v}_j, \mathbf{o}_{j|i} \right\rangle, \tag{8}$$

where r is an iteration index. The term $\left\langle \mathbf{v}_j, \mathbf{o}_{j|i} \right\rangle$ is the inner product between the predicted output and its actual output (of capsule j in the current layer). The assumption is intuitive since all capsules from the previous layer will predict the value of capsule j in the current layer. If the prediction made by capsule i from the previous layer is similar to the actual output $\mathbf{v}_j$, capsule i should have a high probability of the contribution; Hence, the coupling coefficient c_{ij} increases.

In Eqs. (6) and (8), the predictions $\mathbf{o}_{j|i}$ can be calculated by the output capsules $\mathbf{u}_i$ from the previous layer:

$$\mathbf{o}_{j|i} = \mathbf{W}_{ij}\mathbf{u}_i, \tag{9}$$

where $\mathbf{W}_{ij}$ are transformation matrices connecting capsules between two adjacent layers. Suppose there are C classes, then the final capsule layer has C capsules, with the length of each capsule representing the existence probability of the corresponding object. To allow multiple classes in the same image to exist, a margin loss function is used, with the loss function $\mathbb{L}_i$ for class i ($i = 1, 2, \ldots, C$) given by:

$$\mathbb{L}_i = y_i \max(0, m^+ - ||\mathbf{v}_i||_2)^2 + \lambda(1 - y_i)\max(0, ||\mathbf{v}_i||_2 - m^-)^2, \tag{10}$$

where $||\mathbf{v}_i||_2$ is the length of the vector $\mathbf{v}_i$ in the final capsule layer and $y_i = 1$ if and only if the object of class i exists. This leads the length of capsule $\mathbf{v}_i$ to be above m^+ if an object of class i is present and induces the length of capsule $\mathbf{v}_i$ to be below m^- when an object of class i is absent. Here, λ is a controlling parameter and the total classification loss function is calculated by $\mathbb{L}_{\text{cla}} = \sum_i \mathbb{L}_i$, which simply sums the losses from all the final layer capsules.

In capsule networks [21], the back-propagation algorithm is applied to update the convolutional kernels and the transformation matrices. A routing process updates the weights for the coupling coefficients c and the log prior probabilities q. In capsule networks, the vector–vector transformation could potentially extract more relevant features than the scalar–scalar transformation in CNNs.

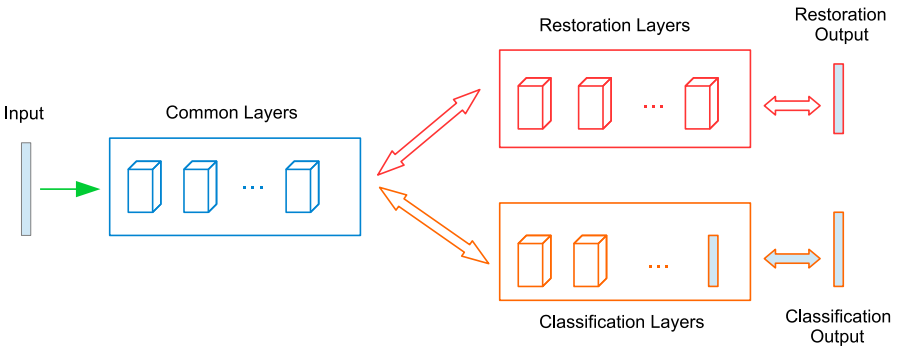

Fig. 2 A general joint framework for image restoration and recognition

3 A Joint Framework for Image Restoration and Recognition

Conventional methods [28] use the restoration and recognition pipeline, in which a restoration stage is followed by a recognition process [1]. However, a joint framework that simultaneously performs restoration and recognition would be more efficient. Figure 2 illustrates the general architecture of a joint framework. The proposed joint framework can remove noise, correct a rotated image, recover an occluded image and perform recognition. This joint framework integrates the restoration and recognition tasks by incorporating (i) common layers, (ii) restoration layers and (iii) classification layers. The total loss function combines the classification and restoration losses. For each input image, the restoration error function is given by:

$$\mathbb{L}_{\text{res}} = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (\mathbf{R}(i, j) - \mathbf{I}(i, j))^2 , \tag{11}$$

where $\mathbf{R}(i, j)$ is the predicted value at location index (i, j) in the last restoration layer and $\mathbf{I}(i, j)$ is the ground truth training image intensity at location index (i, j).

The gradient descent method is then applied to minimize $\mathbb{L}_{\text{res}}$ and $\mathbb{L}_{\text{cla}}$ for the three-pathway framework. Hence, the total loss function is given as:

$$\mathbb{L}_{\text{total}} = \beta \mathbb{L}_{\text{cla}} + (1 - \beta)\mathbb{L}_{\text{rec}}, \tag{12}$$

where β is a hyper-parameter that controls the restoration and classification tasks. Note that if $\beta = 1$, only classification is performed and if $\beta = 0$, only restoration is performed. A value of $0 < \beta < 1$ performs a weighted operation of restoration and classification. We will study numerically the effect of the parameter β on both tasks. Algorithm 1 summarizes this joint framework.

Algorithm 1 Training process of a joint framework

Require:
The original training images and labels.
The designed joint framework. The image degradation parameters.
Ensure:
Randomly initialize the convolutional kernels.
Separate the training data into small batches.
1: **for** *iteration index*=1, *iteration index* ++ **do**
2: **for** *batch index*=1, *batch index* ++ **do**
3: Contaminate the source image with image degradations, such as rotation and occlusion.
4: Run the network forward using the degraded images.
5: Calculate the total loss.
6: Calculate the gradient of all the weights and update the weights.
7: **end for**
8: Decrease the learning rate.
9: **end for**
10: **return** The weights in the common layers, restoration layers and classification layers.

3.1 A Joint Restoration–Recognition Convolutional Neural Network Framework

The proposed joint restoration–recognition CNN framework is illustrated in Fig. 3. The framework comprises three common layers, three restoration layers and five classification layers (three pooling layers and two convolutional layers). In order to keep the size of the input image after convolution, kernels of size [11 × 11] are applied to all the common and restoration layers with a padding size of [5 × 5]. Symmetric connections are applied by setting gate factors to 0.1 and 0.2 on conv1 and conv2, respectively [18].

For recognition, two convolutional layers and three max-pooling layers are applied. Convolutional kernels of size [6 × 6] without padding are applied. A fully connected layer is then connected with the output label. The softmax function is applied in the last stage of the classification. Hence, the output represents the probability of the input data belonging to the corresponding class.

Notice that the loss function for image restoration is calculated as the average pixelwise difference between the ground truth images and predicted images. Such a definition is sensitive to image rotation, as rotation can involve a huge variation in the loss function without changing the image content. In fact, rotation is known to seriously influence the classification accuracy in CNNs [13,22].

3.2 A Joint Framework Based on Capsule Networks

Compared with a convolutional process, which transfers scalar inputs to scalar outputs, a capsule [21] transfers data from a group of neurons to a group of neurons between adjacent capsule layers. Instead of using the max-pooling process, which only finds the local response from an individual layer, a routing process is applied to capsule networks to detect active capsules across layers. Using a routing process, each capsule predicts the output of higher-level capsules. A lower-level capsule becomes active if its prediction agrees with the true output of higher level capsules using a dot product

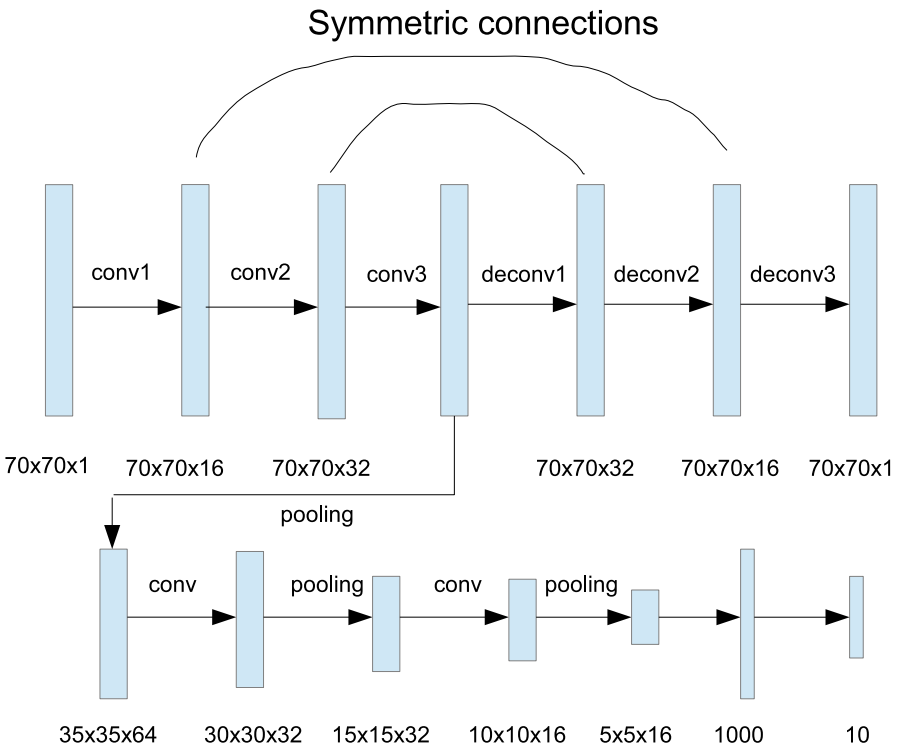

Fig. 3 The joint CNN framework for image restoration and recognition

measurement. In the last fully connected capsule layer, weights are optimized by the margin loss function given in Eq. (10).

The capsule network has a decoder process that allows the reconstruction of the input image. Hence, the weights are not only updated by the classification but also depend on the reconstruction loss function. The capsule network can be transformed for image restoration in the following way: Feed the corrupted images as the input and calculate the loss between the ground truth images and the restored images. In such a way, the joint framework automatically learns the weights based on the ground truth images. Figure 4 illustrates the architecture of the joint image restoration and recognition framework based on capsule networks.

4 Performance Evaluation

The proposed approaches are evaluated on an open dataset for vehicle logo recognition provided by Huang et al. [12]. This dataset is currently the biggest available vehicle logo dataset. It has ten categories and each category contains 1000 training images and 150 testing images. All images have a size of [70 × 70] pixels. Figure 5 shows 20 test images, which will be used for validation and evaluation purposes.

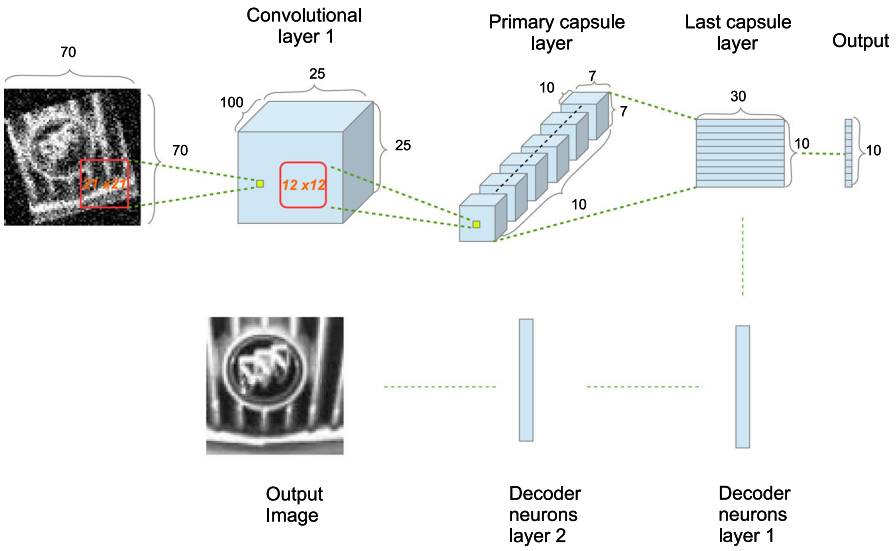

Fig. 4 The joint capsule framework for image restoration and recognition

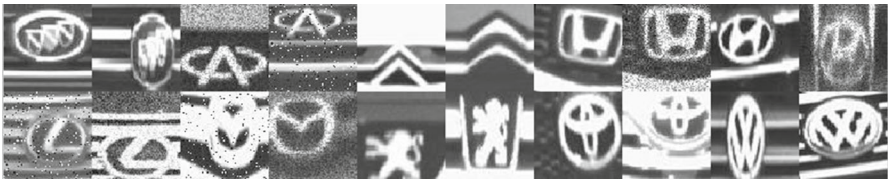

Fig. 5 Twenty test images for illustration purpose

The architecture given in Fig. 3 is the designed joint CNN architecture (Joint-CNN-Net). In order to present a comparison with state-of-the-art networks, we extend the RED-Net [18] by adding two fully connected layers for recognition (the same as last two recognition layers in Joint-CNN-Net) after the conv layers, while keeping the restoration network the same as in RED-Net10. RED-Net has similarities with the well-know recognition networks VGG [24] for the network structure and ResNet [11] for the symmetric skip connections.

In the proposed capsule Joint-Cap-Net framework, shown in Fig. 4, three iterations are applied in the routing process. The performance evaluation of all networks is conducted in Python with the PyTorch toolbox on a laptop with the following specifications: Intel CPU I5 and Nvidia GTX 1070 (extended GPU). The performance of each model is measured in terms of accuracy (percentage of correctly classified images) on the entire test dataset (1500 images).

The training time for the Joint-CNN-Net is about 3 h, and the training time for Joint-RED-Net and Joint-Cap-Net is an hour and a half. At test time, all models are appropriate for real-time implementation because only a one forward pass, involving matrix-vector multiplication and function evaluation, is needed. For our vehicle logo application, each training epoch (10,000 images) took around 2 min for Joint-CNN-

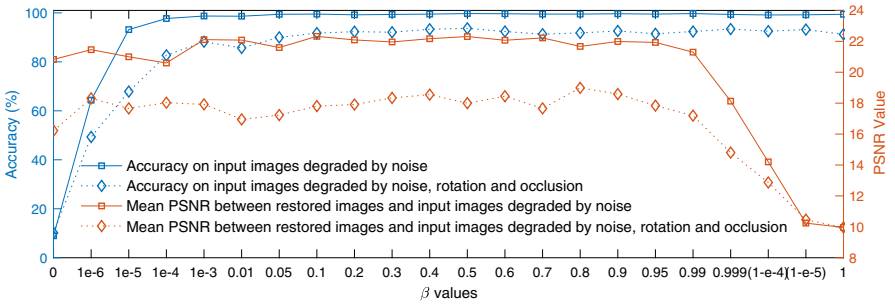

Fig. 6 Validation with different values of β using the Joint-Cap-Net on images degraded only by noise, and images degraded by combing effects from noise, rotation and occlusion

Net, around 1 min for Joint-Red-Net and Joint-Cap-Net. During the testing phase, the evaluation over 1,500 images took 3.29s for the Joint-CNN-Net, 2.54s for Joint-RED-Net and 1.46s for the Joint-Cap-Net.

In this section, the Joint-CNN-Net and the Joint-Cap-Net are evaluated under various degradation conditions including noise, rotation and occlusion. Their accuracy is evaluated using the model generated at the 100-th epoch in the training stage. The peak signal-to-noise ratio (PSNR), in [dB], and the structural similarity (SSIM) index [29] are used to compare each original test image with its degraded version (O-D) and its recovered version (O-R). The PSNR (the higher the better) and SSIM (ranges from -1 to 1, the higher the better) are measures for comparing the differences between two images. The PSNR focuses on the difference between the pixel-pixel intensity values and the SSIM considers the structures within an image [23].

4.1 Trade-Off Between Restoration and Recognition Performance

A natural question to ask is how the joint framework would change the performance of restoration and recognition when compared with the individual implementation of each task. It is a generic framework that achieves a trade-off between restoration and classification by setting different values of the hyper-parameter β. For example, if $\beta = 1$, the joint framework is purely a recognition framework. On the contrary, a purely restoration framework could be built by setting $\beta = 0$, in which recognition is no longer considered in the weight updating process. In order to test the trade-off between the results from these two tasks, values of β are evaluated in the range of $[0, 1]$.

Two scenarios are considered here: (1) the simultaneous task of denoising and recognition, where all training and test images are degraded by a zero-mean Gaussian noise; (2) the simultaneous task of restoration and recognition, in which, in addition to Gaussian noise, random rotations in the range of $[-50°, 50°]$ and occlusions by a rectangular box with random length (up to 30 pixels) constitute additional challenges.

Figure 6 shows validation results for the Joint-Cap-Net using different values of the trade-off parameter β. For the recognition task, the accuracy is relatively high when β is larger than 0.05. The accuracy tends to drop significantly only when β is smaller

Table 1 Evaluations of single task networks and joint frameworks

	PSNR ($\beta = 0$)	Accuracy ($\beta = 0.5$) (%)	PSNR ($\beta = 0.5$)	Accuracy ($\beta = 1$) (%)
Joint-CNN-Net	13.02	**80.25**	**13.07**	67.85
Joint-RED-Net	**12.96**	**92.39**	12.46	84.67
Joint-Cap-Net	20.83	**99.71**	**22.31**	99.42

The bold indicates the best performance in terms of accuracy, PSNR and SSIM

than 0.001. For restoration, the PSNR tends to drop quickly when β is larger than 0.95. Variations in accuracy and PNSR when β is in the range of [0.05, 0.95] can be explained by the uncertainties due to noise, the random rotations and occlusions. In addition, there is no evidence that the restoration and recognition architectures could achieve better results by setting $\beta = 1$ and $\beta = 0$, respectively. On the contrary, involving a classification loss function gives better results than without it for most of the β values. Figure 6 shows high accuracy and significant restoration results when β is in the range of [0.05, 0.95]. Hence, in the following experiments, β is set equal to 0.5.

This paper has also built two other joint frameworks based on CNNs and compared them with the single task network by setting $\beta = 0$ as a restoration network and $\beta = 1$ for the recognition network. Table 1 presents results from the comparison of the pure recognition network ($\beta = 1$), pure restoration network ($\beta = 0$) and joint framework ($\beta = 0.5$) for Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net. Table 1 shows that joint frameworks improve the accuracy compared with the single task networks.

The proposed joint architecture is flexible and can embed a state-of-the-art single task network either for recognition or for restoration. By sharing common layers, the joint framework demonstrates better performance when compared with single task networks. For example, Table 1 shows that the joint frameworks increase the recognition accuracy by 12.4%, 7.72% and 0.29% for Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net, respectively. For image restoration, the PSNR increased significantly for Joint-Cap-Net and slightly dropped from 12.96 to 12.46 for Joint-RED-Net.

4.2 Robustness Evaluations

4.2.1 Noise Robustness Evaluation

The first two rows of Fig. 7 show 20 test images degraded by an additive zero-mean Gaussian noise with variance equal to 0.1. During the training phase, the noisy images are the input to the network models. The original training images and their corresponding labels are the ground truth for updating the weights of the joint restoration–recognition frameworks. In the test phase, noisy images are used to evaluate the trained models. The next two rows illustrate the recovered images by Joint-CNN-Net and Joint-RED-Net, respectively. The last two rows illustrate the restoration effects by Joint-Cap-Net. Clearly the noise degradation has been rectified by Joint-CNN-Net and Joint-Cap-Net, while Joint-RED-Net retained some dark noisy pixels leading to low visual quality for some images.

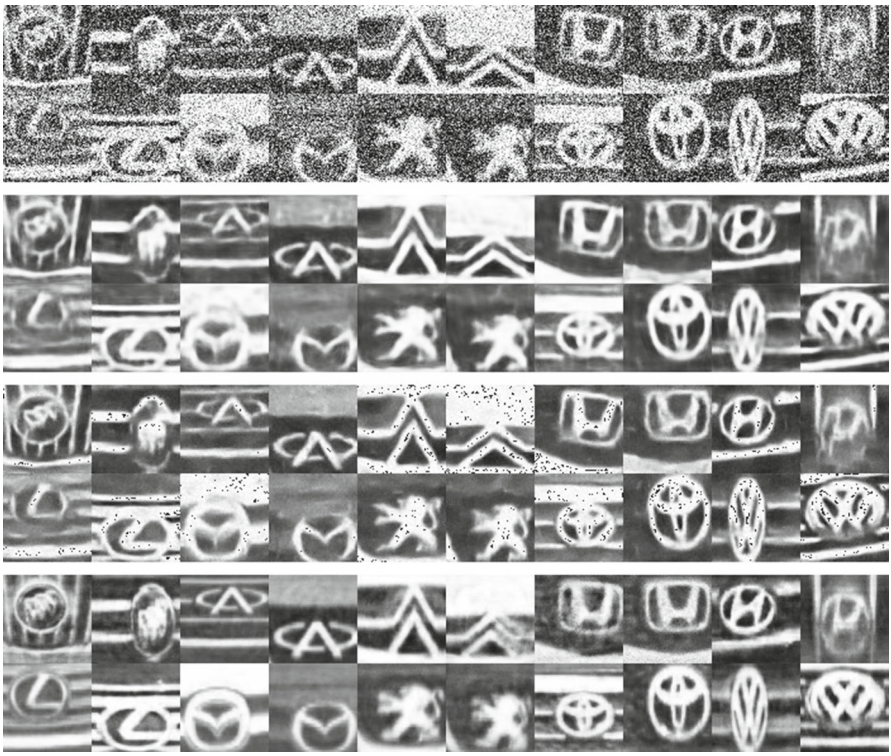

Fig. 7 First 2 rows: Noisy images. Restored images by Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net, respectively

By comparing these restored images with their corresponding ground truth, as shown in Fig. 5, it is evident that some of the images recovered by Joint-Cap-Net have even better visual quality than the ground truth. For example, the two ground truth "Lexus" images have a slight noise inside, while the Joint-Cap-Net removes this noise in the recovered images. For the image denoising task, both the Joint-Cap-Net and Joint-CNN-Net frameworks achieve good results. The denoising results from these two frameworks are similar. Notice that the Joint-Cap-Net automatically rotates images, for example, the last "VW" image has been rotated. These are due to the 2D convolutional kernels preserving the spatial information in CNNs, while the capsules are not restricted to pixel-to-pixel recovery.

Table 2 summarizes the performance of Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net based on all test images. Joint-CNN-Net and Joint-RED-Net achieve accuracy of 98.20% and 97.87% with noisy images, respectively. Meanwhile, Joint-Cap-Net achieves an accuracy of 99.33% under the same noise conditions. The PSNR and SSIM are both improved by the restoration process due to the noisy effects having been removed. The improvement in PSNR and SSIM indicates the high level of similarity of the recovered images with the original images. Notice that the Joint-CNN-Net automatically corrects the rotation of images, which results in a negative effect on

Table 2 Performance of the Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net on noisy images

	Accuracy (%)	PSNR (O-D)	PSNR (O-R)	SSIM (O-D)	SSIM (O-R)
Joint-CNN-Net	98.20	11.99	21.49	0.32	0.62
Joint-RED-Net	97.87		**22.29**		0.64
Joint-Cap-Net	**99.33**		22.06		**0.65**

The bold indicates the best performance in terms of accuracy, PSNR and SSIM

PSNR and SSIM. High SSIM values indicate high similarity with the ground truth based on pixel-to-pixel comparisons. However, this does not mean better visual quality because the ground truth itself could be noisy. Denoising the image would result in a better visual quality while leading to a reduced similarity index value. For instance, the Joint-RED-Net has the highest PSNR while having the lowest visual restoration. Hence, one needs to be careful in interpreting the PSNR and SSIM values.

4.2.2 Rotation Robustness Evaluation

Since the Joint-Cap-Net and Joint-Cap-Net frameworks have the ability of automatically rotate an image, the robustness of the framework has been tested on different rotation angles. In both training and testing stages, rotated images are the input of the joint frameworks. Each image is randomly rotated within the maximum bounds of 20°, 40°, 60° and 80°.

Figure 8 shows the rotation restoration results of the three joint networks by setting random rotation angles up to 40°. Again the first two rows are the test images and the following sets of two rows represent the corresponding restoration outputs by Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net, respectively. The restoration results of Joint-CNN-Net and Joint-RED-Net show that the recovered images are blurred. This can be explained by the fact that the loss function of the restoration process is based on a pixel–pixel correspondence, where every pixel in the restored image is forced to be close to the ground truth image. However, the correct mapping between the input pixels and the restoration pixels has been changed when the input image is rotated. This mapping distortion requires an input pixel to be close to both the corresponding ground input pixels and its neighborhood pixels. Hence, the restored image becomes blurred. On the contrary, the Joint-Cap-Net is able to recover the rotated images automatically, with the noise being removed.

The corresponding accuracy performance measures, PSNR and SSIM, are given in Table 3. These values indicate that the Joint-Cap-Net performs better than the Joint-CNN-Net and Joint-CNN-Net in the presence of image rotations. An interesting property of Joint-RED-Net is that it learns the noise pixels from the input and tends to recover noise pixels as there is a skip connection process in the network [18]. This results in a high PSNR and SSIM when the rotation angle equals 0. For the Joint-CNN-Cap, the SSIM (O-R) is even lower than SSID (O-D) in the Joint-CNN-Net when the image rotation bound is over 40°. According to the obtained results, the Joint-CNN-Net and Joint-RED-Net are not suitable for dealing with image rotation.

Fig. 8 The rotation restoration result of the Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net when the images are randomly rotated up to 40°

Table 3 Performance of the Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net on rotated images

	Angles	0°	20°	40°	60°	80°
	PSNR (O-D)	100	16	11.66	10.29	9.73
	SSIM (O-D)	1	0.36	0.20	0.13	0.10
Joint-CNN-Net	Accuracy (%)	99.87 %	99.73%	97.93 %	89.06%	86.39%
	PSNR (O-R)	27.47	16.27	13.97	13.13	12.60
	SSIM (O-R)	0.80	0.38	0.17	0.10	0.06
Joint-RED-Net	Accuracy (%)	99.30 %	99.20%	99 %	99.13%	98.33%
	PSNR (O-R)	**85.53**	16.25	13.60	12.66	12.13
	SSIM (O-R)	**0.99**	0.38	0.16	0.09	0.06
Joint-Cap-Net	Accuracy (%)	**100** %	**100**%	**99.93** %	**99.93**%	**99.73**%
	PSNR (O-R)	24.04	**22.21**	**21.61**	**20.87**	**21.03**
	SSIM (O-R)	0.69	**0.66**	**0.64**	**0.61**	**0.62**

The bold indicates the best performance in terms of accuracy, PSNR and SSIM

Fig. 9 Occlusion restoration results by the Joint-CNN-Net , Joint-RED-Net and Joint-Cap-Net with the maximum occlusion box of size [30 × 30]

In contrast, the Joint-Cap-Net has much better performance in terms of accuracy and robustness to image rotations. For instance, when all training and testing images are randomly rotated within an angle range from $-60°$ to $60°$, the Joint-Cap-Net achieves an accuracy of 99.93%, while Joint-CNN-Net could only achieve an accuracy of 89.06%. This is due to the capsules extracting more robust features than the max-pooling in CNNs. The Joint-RED-Net was adapted from the state-of-the-art recognition network VGG and ResNet. This results in a similar recognition when compared with Joint-Cap-Net. In addition, the PSNR and SSIM measures have been greatly improved for the Joint-Cap-Net, when comparing the O-R with O-D.

4.2.3 Occlusion Robustness Evaluation

The developed Joint-Cap-Net could also recover images that are partly occluded. In both training and testing stages, occluded images are used as the input of the joint framework. A white square box of a random size is applied to cover image contents in order to simulate the occlusion effects. The occlusion boxes are randomly located in an image and the length of the box is a random integer varying from 0 to 30 pixels.

As presented earlier, the first two rows in Figure 9 show the effects when the occlusion boxes are added to 20 sample images. The intermediate two rows show

Table 4 Performance of Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net on occluded images

	Accuracy (%)	PSNR (O-D)	PSNR (O-R)	SSIM (O-D)	SSIM (O-R)
Joint-CNN-Net	99.47	25.86	21.48	0.91	0.67
Joint-RED-Net	99.07		**31.85**		**0.95**
Joint-Cap-Net	**100**		22.07		0.66

The bold indicates the best performance in terms of accuracy, PSNR and SSIM

the recovered version of the corresponding test images by Joint-CNN-Net, followed by images restored by Joint-RED-Net and Joint-Cap-Net. As shown in Table 4, the recognition results of Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net are similar, with an accuracy of 99.47%, 99.07% and 100%, respectively.

With respect to image restoration, Joint-CNN-Net and Joint-RED-Net have certain recovery abilities, at least the white boxes become slightly transparent. Joint-RED-Net recovers the noise, and the recovered images have undesirable effects. In contrast, the Joint-Cap-Net has removed completely the blocking effects and it is difficult to detect that anything has been occluded. The PSNR and SSIM of the recovered images have decreased in the Joint-CNN-Net and Joint-Cap-Net with respect to degradation. This can be explained with the fact that the occlusion changes only a limited small area of the image, while the images recovered by the Joint-CNN-Net and Joint-Cap-Net change the value on every pixel. The occlusion effects have been removed and the visual qualities have been improved by both frameworks, especially by the Joint-Cap-Net. The Joint-RED-Net has high PSNR and SSIM values while conveying a low visual quality.

4.2.4 Mixed-Degradation Robustness

In the previous evaluations, we notice that the Joint-Cap-Net always tends to denoise and rotate the angles automatically even when the networks were trained on other tasks. In order to validate the performance of the proposed framework under very challenging conditions, different image degradations are combined together in the training and testing stages. The results are presented in Fig. 10. The first two rows show the combined degradation due to a zero-mean Gaussian noise (variance equal to 0.1), rotation (with a random angle from $-60°$ to $60°$) and occlusion (with a square white box with a random size from 0 to 30 pixels). The following rows of Fig. 10 represent the recovered images from Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net, respectively.

Clearly, Joint-CNN-Net and Joint-RED-Net cannot deal with rotation and occlusion. This makes the recovered images difficult to distinguish by a human observer. On the contrary, the Joint-Cap-Net successfully recovers the image after the combined degradation of rotation, occlusion and noise.

Table 5 shows the corresponding improvement in terms of PSNR and SSIM. For the recognition, the Joint-CNN-Net achieves an accuracy of 65.42%. Despite the poor recovery, the Joint-RED-Net achieves a descent recognition accuracy of 90.73%, and the Joint-Cap-Net achieves an accuracy of 91.39%.

Fig. 10 The restoration result of the Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net with combined Gaussian noise, rotation and occlusion

Table 5 Performance of the Joint-CNN-Net, Joint-RED-Net and Joint-Cap-Net on combined degradations

	Accuracy (%)	PSNR (O-D)	PSNR (O-R)	SSIM (O-D)	SSIM (O-R)
Joint-CNN-Net	65.42	7.55	12.82	0.05	0.09
Joint-RED-Net	90.73		12.33		0.06
Joint-Cap-Net	**91.39**		**17.26**		**0.43**

The bold indicates the best performance in terms of accuracy, PSNR and SSIM

5 Summary

Image restoration and recognition are important tasks that are usually implemented separately. This work develops joint frameworks for image restoration and recognition that simultaneously perform both tasks. A joint framework comprises common layers, classification layers and restoration layers. We have developed three implementations (namely the Joint-CNN-Net, the Joint-RED-Net and the Joint-Capt-Net) and have tested them over different image degradation including noise, rotation and occlusion. The experiments show that the joint frameworks improve the performance when compared with single task networks (either recognition or restoration). The

joint frameworks based on CNNs (Joint-CNN-Net and Joint-RED-Net) achieve good results on noisy images but have limitations on image rotation and occlusion. The joint capsule network, called Joint-Caps-Net, achieves better results than the Joint-CNN-Net and Joint-RED-Net in terms of recognition accuracy and restoration measures. The key to the success of learning capsules is due to a more efficient routing process compared to the pooling process in CNNs. Finally, the proposed joint frameworks are not restricted to the considered application but could be applied to other image recognition/restoration tasks and could use different inner network architectures.

Acknowledgements This work has been partially supported by the "SETA project: An open, sustainable, ubiquitous data and service for efficient, effective, safe, resilient mobility in metropolitan areas" funded from the European Union's Horizon 2020 research and innovation program under Grant Agreement No. 688082. Nidhal C. Bouaynaya was funded by a US National Science Foundation (NSF) grant under Award Number DUE-1610911. This work was jointly supported by the Research Funds of Chongqing Science and Technology Commission (Grant No. cstc2017jcyjAX0293).

References

1. G. Chen, Y. Li, S.N. Srihari, Joint visual denoising and classification using deep learning, in *Proceedings of IEEE International Conference on Image Processing, Phoenix, AZ, USA* (2016), pp. 3673–3677
2. R. Chen, M.A. Jalal, L. Mihaylova, R. Moore, Learning capsules for vehicle logo recognition, in *Proceedings of the International Conference on Information Fusion, Cambridge, UK* (2018)
3. K. Dabov, A. Foi, V. Katkovnik, K. Egiazarian, Image denoising by sparse 3-D transform-domain collaborative filtering. IEEE Trans. Image Process. **16**(8), 2080–2095 (2007)
4. A. Daneshmand, F. Facchinei, V. Kungurtsev, G. Scutari, Flexible selective parallel algorithms for big data optimization, in *Proceedings of the 48th Asilomar Conference on Signals, Systems and Computers* (2014), pp. 3–7
5. A. Daneshmand, F. Facchinei, V. Kungurtsev, G. Scutari, Hybrid random/deterministic parallel algorithms for convex and nonconvex big data optimization. IEEE Trans. Signal Process. **63**(15), 3914–3929 (2015)
6. I. Dassios, K. Fountoulakis, J. Gondzio, A second-order method for compressed sensing problems with coherent and redundant dictionaries. arXiv preprint: arXiv:1405.4146 (2014)
7. I.K. Dassios, K. Fountoulakis, J. Gondzio, A preconditioner for a primal-dual newton conjugate gradient method for compressed sensing problems. SIAM J. Sci. Comput. **37**(6), A2783–A2812 (2015)
8. J. Deng, W. Dong, R. Socher, L.J. Li, K. Li, L. Fei-Fei, Imagenet: a large-scale hierarchical image database, in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA* (2009), pp. 248–255
9. C. Dong, C.C. Loy, K. He, X. Tang, Image super-resolution using deep convolutional networks. IEEE Trans. Pattern Anal. Mach. Intell. **38**(2), 295–307 (2016)
10. D. Glasner, S. Bagon, M. Irani, Super-resolution from a single image, in *Proceedings of the IEEE International Conference on Computer Vision, Kyoto, Japan* (2009), pp. 349–356
11. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA* (2016), pp. 770–778
12. Y. Huang, R. Wu, Y. Sun, W. Wang, X. Ding, Vehicle logo recognition system based on convolutional neural networks with a pretraining strategy. IEEE Trans. Intell. Transp. Syst. **16**(4), 1951–1960 (2015)

13. M. Jaderberg, K. Simonyan, A. Zisserman, et al., Spatial transformer networks, in *Proceedings of the Advances in Neural Information Processing Systems, Montréal, Quebec, Canada* (2015), pp. 2017–2025
14. K.I. Kim, Y. Kwon, Single-image super-resolution using sparse regression and natural image prior. IEEE Trans. Pattern Anal. Mach. Intell. **32**(6), 1127–1133 (2010)
15. A. Krizhevsky, I. Sutskever, G.E. Hinton, ImageNet classification with deep convolutional neural networks, in *Proceedings of the 25th International Conference on Neural Information Processing Systems, Lake Tahoe, Nevada, USA* (2012), pp. 1097–1105
16. Y. Lecun, L. Bottou, Y. Bengio, P. Haffner, Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
17. D.G. Lowe, Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
18. X. Mao, C. Shen, Y.B. Yang, Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections, in Proceedings of the Advances in Neural Information Processing Systems, Barcelona, Spain (2016), pp. 1802–2810
19. T. Ni, J. Zhai, A matrix-free smoothing algorithm for large-scale support vector machines. Inf. Sci. **358–359**, 29–43 (2016)
20. L.I. Rudin, S. Osher, E. Fatemi, Nonlinear total variation based noise removal algorithms. Phys. D: Nonlinear Phenom. **60**(1–4), 259–268 (1992)
21. S. Sabour, N. Frosst, G.E. Hinton, Dynamic routing between capsules, in *Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA* (2017), pp. 3859–3869
22. D. Scherer, A. Müller, S. Behnke, Evaluation of pooling operations in convolutional architectures for object recognition, in *Proceedings of International Conference on Artificial Neural Networks, Thessaloniki, Greece* (2010), pp. 92–101
23. H.R. Sheikh, M.F. Sabir, A.C. Bovik, A statistical evaluation of recent full reference image quality assessment algorithms. IEEE Trans. Image Process. **15**(11), 3440–3451 (2006)
24. K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, in *Proceedings of the International Conference on Learning Representations, San Diego, CA, USA* (2015), pp. 1–14
25. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting. J. Mach. Learn. Res. **15**(1), 1929–1958 (2014)
26. C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, et al., Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA* (2015), pp. 1–9
27. R. Timofte, V. De, L. Van Gool, Anchored neighborhood regression for fast example-based super-resolution, in *Proceedings of the IEEE International Conference on Computer Vision, Sydney, NSW, Australia* (2013), pp. 1920–1927
28. R. Wang, D. Tao, Recent progress in image deblurring. CoRR **abs/1409.6838** (2014). arxiv:1409.6838
29. Z. Wang, A.C. Bovik, H.R. Sheikh, E.P. Simoncelli, Image quality assessment: from error visibility to structural similarity. IEEE Trans. Image Process. **13**(4), 600–612 (2004)
30. J. Yang, Z. Lin, S. Cohen, Fast image super-resolution based on in-place example regression, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2013), pp. 1059–1066
31. M.D. Zeiler, R. Fergus, Visualizing and understanding convolutional networks, in *Proceedings of the European Conference on Computer Vision, Zurich, Switzerland* (2014), pp. 818–833
32. H. Zhu, K.V. Yuen, L. Mihaylova, H. Leung, Overview of environment perception for intelligent vehicles. IEEE Trans. Intell. Transp. Syst. **18**(10), 2584–2601 (2017)