Journal of Computer Science

Original Research Paper

Classification of Transformed and Geometrically Distorted Images using Convolutional Neural Network

Sanad Aburass, Ammar Huneiti and Mohammad Belal Al-Zoubi

Department of Computer Science, University of Jordan Amman, Jordan

Article history
Received: 14-06-2022
Revised: 14-07-2022
Accepted: 26-07-2022

Corresponding Author:
Sanad Aburass
Department of Computer Science, University of Jordan Amman, Jordan
Email: saburass@miu.edu

Abstract: Convolutional Neural Network is a deep learning method that is used in many image-related applications, such as image recognition and classification, it has achieved great performance in these fields, but it still suffers from some shortcomings. One of these shortcomings is not being able to be invariant to the input data due to some image transformations like translation, rotation, scaling, and geometric distortions such as skewness, perspective distortion and pincushion distortion. This study presents an optimized CNN which uses the Geometric Heat Flow (GHF) to improve the performance of the CNN regarding the invariant limitation and classification accuracy. GHF is a partial differential equation that expresses how the heat would diffuse on a surface concerning time in a specific location. GHF is invariant to image transformations and geometric distortions if it was taken concerning the object's arc length which will lead to an invariant CNN. The experiments show that GHF improves the performance of the CNN, and the proposed work achieves an accuracy of 98.09% on the MNIST handwritten dataset, 92.58% on the MNIST-Fashion dataset, and 86.09% on the CIFAR-10 dataset.

Keywords: Convolutional Neural Network, Image Geometric Distortion, Image Transformations, Invariant

Introduction

CNN is one of the most used methodologies in image classification, it achieves good performance in classifying images, but it still has some drawbacks (Liu *et al*., 2017; McNeely-White *et al*., 2019; Alom *et al*., 2019). One of these drawbacks is not being able to be invariant because of some transformations and geometric distortions of the input images (Jaderberg *et al*., 2015). Some techniques typically use dataset augmentation to overcome this matter (Mallat, 2013; McNeely-White *et al*., 2019; Anselmi, *et al*., 2016), but this needs more training data and a larger number of model parameters, and might significantly increase the training time. (Jaderberg *et al*., 2015) and (McNeely-White *et al*., 2019). The result of such a problem is obvious when dealing with domain-specific problems. For example, in medical applications, the rotation of the image can be inessential due to the symmetrical nature of some biological assemblies. Nevertheless, the scale is constant throughout the imaging procedure and should not be thought of as a nuisance factor. Furthermore, scale invariance can reduce the performance if the size of the object is informative. For example, when distinguishing cancer cells from healthy cells (Su, *et al*., 2015) and (Laptev *et al*., 2016).

Invariance and Equivariance are different from each other but sometimes they are used interchangeably. Equivariance means varying in a similar proportion while invariant means no variance at all (Lenc and Vedaldi, 2015) and (Chidester *et al*., 2018). Formally, a function f is considered to be equivariant with respect to a transformation T if $f(T(x)) = T(f(x))$. Therefore, applying a transformation to x is equivalent to applying the transformation to the result of the function $f(x)$. The Invariant is a special case of equivariant. A function f is considered to be invariant concerning a transformation T if $f(T(x)) = f(x)$, therefore, the result throughout the function f does not change when the transformation is applied to the original image (Serfling, 2010).

Convolutional Neural Network is translation equivariance by nature because of the convolution operations (Worrall *et al*., 2017) since it convolves on the input image to extract the features of that image. Therefore, if an object is translated, it will still be perceived with disregard for its position in the input image. Also, the pooling operations might make the CNN rotation equivariance but only if the image was slightly rotated, but as the rotation degree increases the CNN might fail to classify the image accurately. While CNN is a translation and slight rotation

equivariance, it is not a translation, scaling or rotation invariant (Cohen *et al*., 2017; Hinton *et al*., 2012; Worrall *et al*., 2017; Cheng *et al*., 2018; Aburass *et al*., 2020).

The main objective of this study is to enhance the CNN performance concerning the invariant limitation to accomplish higher performance in image classification, this is done by applying GHF to the CNN. GHF is a mathematical model that uses partial differential equations to describe how the heat would distribute within the boundaries of an object. It is proven that the geometric heat flow is invariant to affine transformations and geometric distortions (Cao, 2003; Sapiro, 2006). After computing the geometric heat flow of the image, it can be concatenated with the flattened vector of the CNN to make it more informative and descriptive since the geometric heat flow is invariant to image transformations and geometric distortions which should lead to better performance and better classification accuracy.

Literature Review

Jaderberg *et al*. (2015), have presented a self-contained module for neural networks and implemented spatial transformations of features by using a localization network, parametrized sampling grid, and spatial transformer networks. While they have accomplished decent results this study is considered data augmentation, and it is not a solid solution to the invariant problem of the CNN.

Laptev *et al*. (2016), presented a framework to combine previous knowledge on nuisance variations with data when training the network. They have formulated a set of transformations and produced several images based on these transformations. Afterward, these transformed images are delivered through the initial layers of the network and using the TI-POOLING operator to form transformation-invariant features. While they have accomplished transformation invariance by pooling transformed feature maps, it added enormous computational complexity to the network because of the forward and backward passes for each element.

Shen *et al*. (2016), proposed an approach to solve the invariance problem of the CNN, the authors transformed the feature maps that are produced after each convolutional layer with random transformations, such as translation, rotation, and scaling, then fed the transformed feature map to the next layer. They have achieved good accuracy, but this approach is considered to be data augmentation which is not a solid approach to solving the invariant problem in the CNN.

Worrall *et al*. (2017), have proposed a network that is equivariant to patch-wise shifting and continuous 360° rotation. They reconstructed the convolutional filters using derivations from complex harmonics, returning a maximal response and orientation for every receptive field patch. Using these new filters CNN could become invariant to rotation and translation but not scaling. Moreover, their work has a disadvantage of the higher per-filter computational cost as they need to originate and reconstruct all the filters in the CNN.

Raj (2017), presented an approach that uses Zernike moments in CNN to evaluate the discrimination between the face and non-face patterns, and gender classification using facial expression recognition. They used Zernike moments as initial filters, to show some unique structures of the image that might be helpful to distinguish faces from non-faces images and gender classifications. In facial expression recognition, they have accomplished an accuracy of 87.22%. The main drawback of their work is feature loss. The use of a filter based on Zernike moments might lead to feature loss in some cases.

Hinton *et al*. (2012; 2011), presented a new architecture that contains capsules. These capsules comprise a group of neurons that are responsible for the instantiation parameters of an entity such as pose velocity and albedo; these capsules will then represent information in a hierarchal form.

Although this study is impressive it has some defects. The authors have not specified how the weights "W" are learned. Also, the algorithm yields an additional hyperparameter "r" which means more computational complexity. While the algorithm has achieved state-of-the-art accuracy on the MNIST dataset it fails to perform well in the CIFAR10 dataset.

Cheng *et al*. (2018), presented a technique that makes CNN invariant to rotation. They added a rotation invariant layer and Fisher discriminative layer to the network to make it invariant to rotation. These layers will try to learn the rotations of the object based on the class, so they could predict the rotation of an object when it distinguishes it. They have implemented their approach to some famous CNN like VGG and AlexNet and accomplished high accuracy, but the work is only focused on the rotation invariant, and they did not solve translation or scaling invariant problems in CNN.

McNeely-White *et al*. (2019), considered the CNN representations invariance and equivariance to input image transformations. They have estimated the linear relationships among representations of the original and transformed images. Jaderberg *et al*. (2015) used the data augmentation technique which, as it has previously mentioned, is not a robust solution for the invariant problem.

Guo *et al*. (2020) proposed an adaptive filters approach using a recurrent gated network to select the most efficient filters to be used based on previous knowledge from the previous layers. Although the use of adaptive filters boosted the accuracy of the CNN, it is not fully adaptive since they have used the same predefined filters of the CNN.

Table 1: Summary of previous works based on the problem's aspects

Previous works	Translation invariance	Rotation invariance	Scaling invariance	Used Data augmentation	Solved Geometric distortion problem
McNeely-White *et al*. (2019)	✓	✓	✓	✓	✗
Jaderberg *et al*. (2015)	✓	✓	✗	✓	✗
Hinton *et al*. (2012)	✓	✓	✓	✗	✗
Cheng *et al*. (2018)	✗	✓	✗	✗	✗
Laptev *et al*. (2016)	✓	✓	✓	✓	✗
Worrall *et al*. (2017)	✓	✓	✗	✓	✗
Henriques and Vedaldi (2017)	✓	✗	✗	✗	✗
Dai *et al*. (2017)	✓	✗	✗	✗	✗
Dieleman *et al*. (2016)	✓	✗	✗	✗	✗
Marcos *et al*. (2017)	✓	✗	✗	✗	✗

There are many methods to encode rotation equivariance for general image classification. One straightforward technique is to transform the domain of the image to a substitute domain, like the log-polar domain (Henriques and Vedaldi, 2017; Schmidt and Roth, 2012) where rotation can become some other transformation that is simpler to manage, but this can be volatile to translations and this warping might present distortion, as pixels near the image center are sampled more heavily than pixels near the boundary. The spatial transform layer (Jaderberg *et al*., 2015) and deformable convolutional layer (Dai *et al*., 2017) allow the network to learn some sampling patterns that are non-regular and can help to learn rotation invariance, however, invariance is not enforced, which would be a challenge for tasks with small training sets.

Dieleman *et al*. (2016) proposed an approach in which feature maps of the standard network are made equivariant or invariant to rotation by combinations of cyclic slicing, stacking, rolling, and pooling. RotEqNet (Marcos *et al*., 2017), has enhanced this idea by storing, for every feature map for a corresponding filter, only the maximal response across rotations and the value of the equivalent rotation, to keep pose information, which led to improved results and significant storage savings.

We have summarized the studied research based on the invariant problem aspects, Table 1 shows that in detail. The problem aspects are:

1) Solving the translation invariance problem
2) Solving the rotation invariance problem
3) Solving the scaling invariance problem
4) The use of data augmentation
5) Solving the image geometric distortion problem

In this study, we have proposed an approach to make the CNN invariant to image transformations such as Translation, Rotation, and Scaling, which also makes the CNN invariant to geometric distortions that might affect the image.

Materials and Methods

This section presents the proposed approach to enhance the CNN using the geometric heat flow, by adding a new layer to the CNN called The Heat Layer. Figure 1 illustrates the used mechanism. The Heat Layer contains the following phases as shown in Fig. 2.

The Heat Equation

The heat equation is a partial differential equation that describes how heat or temperature would distribute on a surface or in a body concerning time in a specific location. According to the second law of Thermodynamics, if two homogeneous bodies were in direct contact and one is hotter than the other, then the temperature will flow from the hotter object to the colder one at a rate proportional to the heat difference (Bennett and Myers, 1982; Mikula, 2002; Aubert *et al*., 2006).

The heat equation can be expressed as:

$$\frac{\partial u}{\partial t} = \alpha \nabla^2 u \tag{1}$$

Where, u is the heat that we want to know, t is for time and α is the diffusivity constant.

If we have one dentitional space the heat equation can be rewritten as:

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \tag{2}$$

Where, x is a location in one-dimensional space.

But in two-dimensional space as an image, the heat equation would be rewritten as:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) \tag{3}$$

Where, x is a location on the x-axis and y is a location on the y-axis.

In other words, the heat equation is used to predict the temperature at a specific point using the second derivative of that point concerning time, and after some time, we will have a heat flow.

One-Dimensional Heat Flow

In the one-dimensional case, the heat equation will predict the temperature of a point based on the temperatures of the points exactly near that point. As shown in Fig. 3.

To predict the temperature of point T2 over time, we must compute its second derivative based on the temperatures of T1 and T3. Based on equation 2 above the temperature of T2 over time would be as follows:

$$\frac{\partial u}{\partial t} = \alpha\left(\frac{T1+T3}{2} - T2\right) \tag{4}$$

So, based on the diffusivity constant and the average temperature between T1 and T3, T2 will heat up or cool down producing a heat flow.

Two-Dimensional Heat Flow

In two-dimensional space, the heat equation will work in the same way as in one-dimensional space, but it will consider the second derivative on the x-axis and the second derivative on the y-axis at the same time (Aubert *et al*., 2006).

In this study, we assume that the values of the pixels in the image are values of temperature to produce the heat flow over time, as shown in Fig. 4.

To travel with a 2D image through time, we need to calculate the steps for each difference for *x*, *y*, and *t* as follows:

$$x_i = i\Delta x$$
$$y_j = j\Delta y$$
$$t_k = k\Delta t$$

Where, *i*, *j*, and k are steps for each difference for *x*, *y*, and *t* respectively.

Therefore, the temperature of a point (*x*, *y*) at a specific time can be written as:

$$u(x,y,t) = u_{i,j}^{k} \tag{5}$$

And the heat equation can be rewritten using finite differencing as:

$$\frac{u_{i,j}^{k+1} - u_{i,j}^{k}}{\Delta t} = \alpha\left(\frac{u_{i+1,j}^{k} - 2u_{i,j}^{k} + u_{i-1,j}^{k}}{\Delta x^2} + \frac{u_{i,j+1}^{k} - 2u_{i,j}^{k} + u_{i,j-1}^{k}}{\Delta y^2}\right) \tag{6}$$

We can rearrange the equation to be as follows:

$$u_{i,j}^{k+1} = \gamma\left(u_{i+1,j}^{k} + u_{i,j+1}^{k} + u_{i,j-1}^{k} - 4u_{i,j}^{k}\right) + u_{i,j}^{k} \tag{7}$$

Where:

$$\gamma = \alpha\frac{\Delta t}{\Delta x^2}$$

In other words, the heat equation can predict the temperature in a specified point on a 2D space in the next time step based on the second derivative of that point in the x-axis and the second derivative of the y-axis, as shown in Fig. 5.

To solve the heat equation, two constraints must be resolved first, which are the boundary condition and the initial condition. The boundary condition is the initial boundaries of the heat at time =0 and the initial condition is the initial temperature of the boundaries at time = 0.

As mentioned above we assume that the pixels' values are temperature values and at time =0 are the initial conditions, and we consider the surrounding contours of the objects in the image as the boundary conditions.

Geometric Heat Flow

The problem that we are trying to solve in this study is the invariant problem in CNN. CNN still suffers from the issue of being not invariant to large image transformations such that translation, rotation, and scaling, also it is not invariant to image geometrical distortions such as skewness, projection distortion, and barrel distortion. The heat equation explained above is not invariant to image transformations and geometric distortions, which means if the object was transformed or was geometrically distorted the heat flow will be changed. This problem arises because the heat equation takes the second derivative spatially depending on the x and y location of the boundary condition. In other words, if the boundaries of the object were transformed into another location, rotated, scaled, or were geometrically distorted the heat equation will be changed. So, we need a heat flow that is geometrically invariant to affine transformations and geomatic distortions, and that is exactly what (Sapiro, 2006) has proven mathematically in his book "Geometric partial differential equations and image analysis".

Instead of being dependent on the Euclidean space, we need to find a property of the boundary condition to derive concerning it to achieve invariant heat flow, and this property would be the arc length of the curvature.

As proven in the book (Sapiro, 2006), we can achieve an invariant geometric heat flow using the following equation:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial s^2} \tag{8}$$

The derivative is now being taken concerning the affine arc length of the object's contours. The arc length is

the distance between two points along a curved line. This curved line is not necessarily a regular line it could be irregular, as shown in Fig. 6.

To find the arc length, the curve must be divided into small straight segments and then add up the length of these lines based on the Pythagorean Theorem (Stewart *et al*., 2020), as shown in the equation below:

$$s = \int_a^b \sqrt{(dx^2) + (dy^2)} \tag{9}$$

Now we need to derive the curves we have concerning the arc length, the first derivative of s would be as follows:

$$s' = \sqrt{\left(\frac{dx}{ds}\right)^2 + \left(\frac{dy}{ds}\right)^2} \tag{10}$$

According to the equation above, we need to calculate the second derivative of the arc length to achieve an invariant heat flow. The second derivative of s is shown below (Stewart *et al*., 2020):

$$s'' = \frac{\left(2\frac{dx}{ds} * \frac{d^2x}{ds^2}\right) + \left(2\frac{dy}{ds} * \frac{d^2y}{ds^2}\right)}{2\sqrt{\left(\frac{dx}{ds}\right)^2 + \left(\frac{dy}{ds}\right)^2}} \tag{11}$$

$$\frac{\partial u}{\partial t} = \frac{\left(\frac{dx}{ds} * \frac{d^2x}{ds^2}\right) + \left(\frac{dy}{ds} * \frac{d^2y}{ds^2}\right)}{\sqrt{\left(\frac{dx}{ds}\right)^2 + \left(\frac{dy}{ds}\right)^2}} \tag{12}$$

Alvarez *et al*. (1993) and Cao (2003), have proven that this is the unique affine geometric flow that holds all the key characteristics of a scale space for planar curves. Also, (Sapiro, 2006), stated that the heat flow over time will lead to a deformation of the curve, but all these deformations are smooth without creating self-intersections or any singularities. In this way, we can have an invariant heat flow that describes the surrounding boundaries of the object regardless of any transformations or geometric distortions.

The Heat Layer

The main idea behind this study is to enhance CNN so it can classify transformed and geometrically distorted images more accurately, by adding a new layer after each convolution layer, we are going to call it The Heat Layer. The heating layer will take the feature map, that was produced from the convolution layer, as input and then calculate GHF for each feature in that feature map. After that, it will accumulate all the GHF that were produced from all the convolutions so they can be concatenated with the flattened vector. Figure 1 shows an illustration of CNN after adding the heating layer, then it is followed by the phases of the proposed approach in detail.

In the conventional CNN, the convolution layers give their feature maps either to a pooling layer or to another convolution layer. But in this study, we have added a new layer that will take the feature map from the convolution layers, to compute the GHF of each feature in that feature map. The following are the phases of the inside of the Heat Layer after getting a feature map.

Phase 1: *Contours Drawing*

In this phase, the heating layer will look up the features of the feature map by drawing contour boundaries of the adjacent regions that have the same color or intensity. We have used the algorithm of (Suzuki, 1985) to draw the contours of the features, and used median thresholding to have individual features with not overlapping. Figure 7 below shows an example of an image after drawing the contours.

Phase 2: *Features Separation*

In this phase, the features will be separated to compute the GHF for each feature aside. Separating the features will help to produce accurate GHF which will lead to a more generalized view of the image. For example, if we took the face image above after drawing the contours, the separated features would be as shown in Fig. 8.

Phase 3: *Geometric Heat Flow Generation*

After separating the features, the GHF will be generated for each feature, as explained in the previous section. For the face image that we used above, the GHF for each feature would be as shown in Fig. 9.

Phase 4: *Heat Flow Flattening*

The heat flows are 2-Dimensional matrices, so they must be converted to a 1-dimensional vector so they can be concatenated with the flattened vector of the CNN before giving it to the fully connected layer for classification. Fig. 10 below shows an example of the flattening operation.

Phase 5: *Concatenation*

Finally, after accumulating all the flattened GHF from all the convolution layers in one vector, this vector will be concatenated with the flattened vector of the CNN so it can be given to the fully connected layer. This should make the classification more invariant to image transformations and geometric distortion. Figure 11 below shows an example of the concatenation operation.

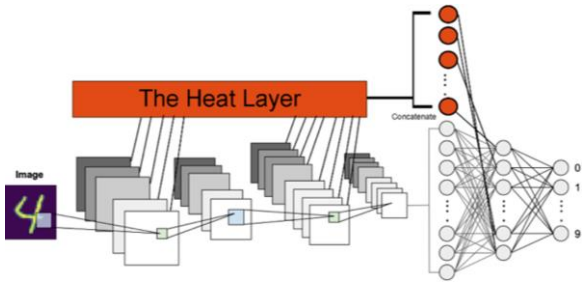

Fig. 1: CNN with the heat layer

Fig. 2: Phases of the heat layer

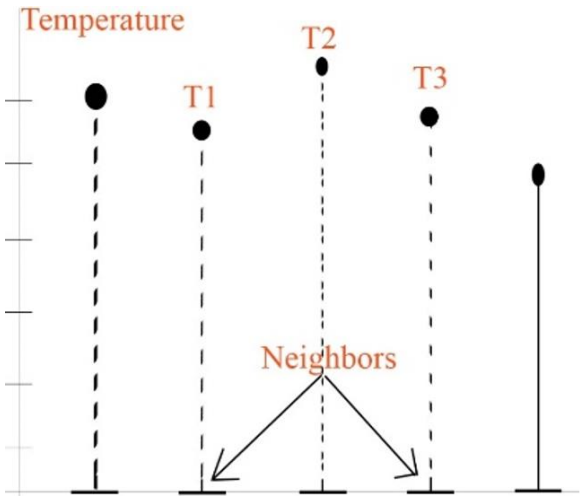

Fig. 3: Neighbor points in one-dimensional space

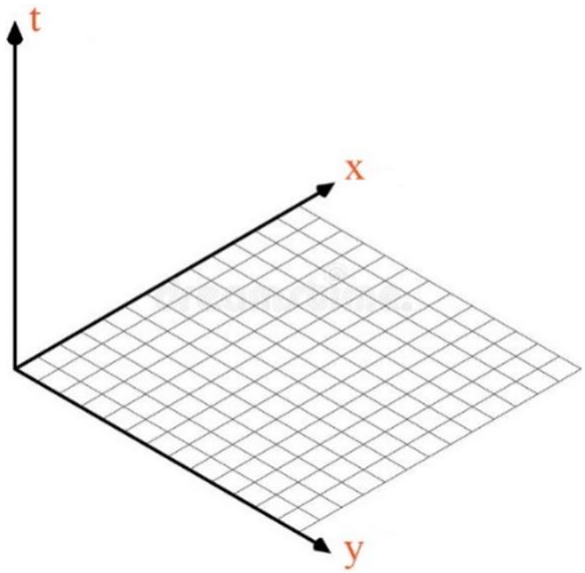

Fig. 4: 2D image over time

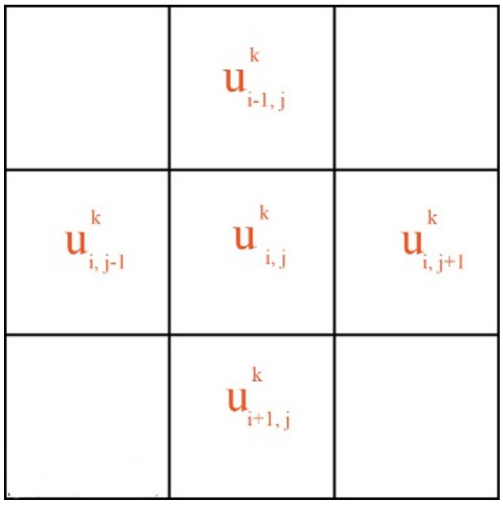

Fig. 5: Point Neighbors in a 2D image

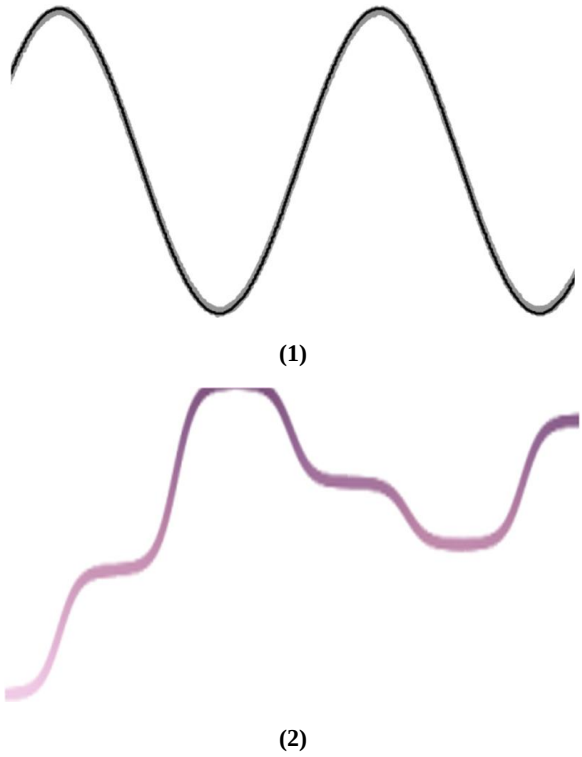

Fig. 6: Regular curve line and irregular curve line

Fig. 7: (a) the original image of a face, (b) the gray image, and (c) the contours of the image

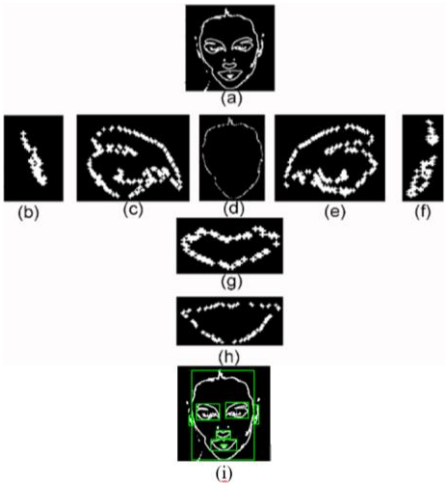

Fig. 8: The separated feature of a face image. (a) the original image, (b) the left ear, (c) the left eye, (d) the boundaries of the face, (e) the right eye, (f) the right ear, (g) the nose, (h) the mouth, and (i) the selected regions

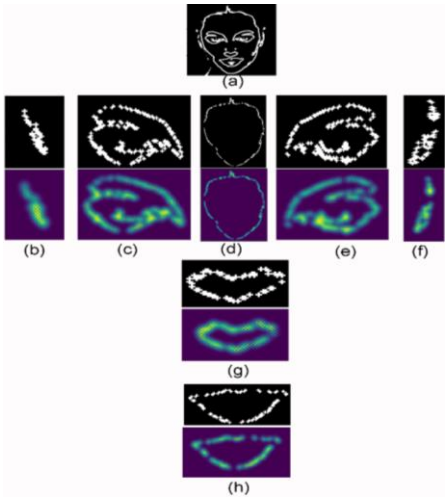

Fig. 9: The separated feature of a face image. (a) the original image, (b) the left ear, (c) the left eye, (d) the boundaries of the face, (e) the right eye, (f) the right ear, (g) the nose, (h) the mouth, and (i) the selected regions

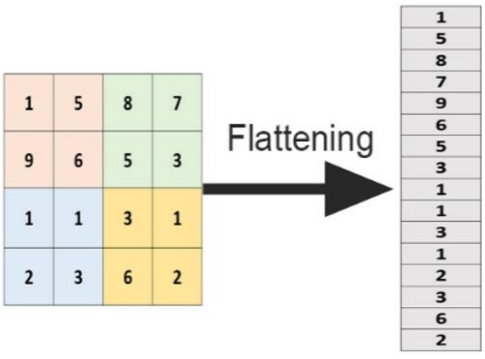

Fig. 10: Flattening

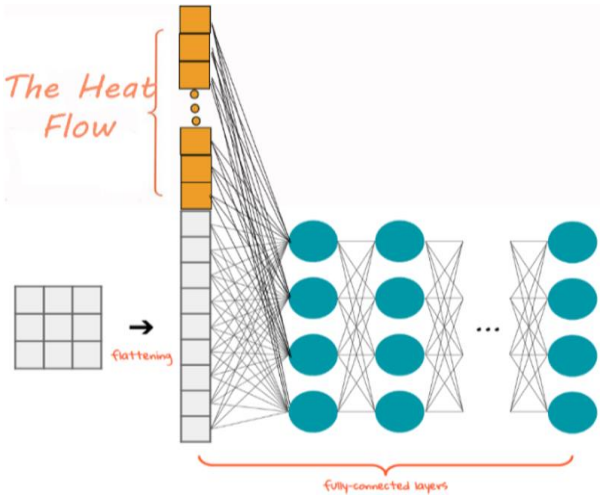

Fig. 11: Concatenation

Experimental Results

This approach was executed using the Python TensorFlow platform by Google collab. The approach was tested on three benchmark datasets which are the MNIST handwritten digits dataset, MNIST fashion dataset, and CIFAR10 dataset. To make sure that the presented approach improves the CNN and makes it more invariant to image transformations and geometric distortions, we automatically make random image transformations and random geometric distortions on 90% of the testing image batches, which led to a degradation in the performance of the conventional CNN as shown in the results below, which confirms that conventional CNN is not invariant to transformation and geometric distortions.

We implement and test concatenating the geometric heat flow, then compare the results with the conventional CNN and with the results of (Shen *et al*., 2016). To solve the invariance problem of the CNN, (Shen *et al*., 2016) transformed the feature maps that are produced after each convolutional layer with random transformations, such as translation, rotation, and scaling, then fed the transformed feature map to the next layer. Table 2 shows the results of concatenating the geometric heat flow compared to the results of (Shen *et al*., 2016) approach on the MNIST hand written digits dataset.

Table 3 shows the results of the proposed approach implemented on the MNIST fashion dataset we compare our work with (Shen *et al*., 2016) approach and we achieve better results compared to their work.

We test the proposed approach and (Shen *et al*., 2016) approach on the CIFAR10 dataset and we achieve better performance than their approach as shown below in Table 4.

Also, we compare our approach with the work of (Raj, 2017) which uses Zernike Moments (ZM) as initial filters to extract invariant structures of the image, by implementing their method on the three datasets. ZM are projections of an image onto the complex Zernike polynomials that are orthogonal over the unit circle. So, a radius must be provided to calculate the ZM of the image. So, we use the degrees 45° and 90° to extract ZMs of the images.

Table 5 shows the results of our approach compared to the results of (Raj, 2017) approach to the MNIST handwritten digits dataset.

Table 6 shows the results of our approach implemented on the MNIST fashion dataset compared with the (Raj, 2017) approach, and we achieve better results compared to their work. Also, this stage tests (Raj, 2017) approach alongside our approach on the CIFAR10 dataset, and we achieve better performance than their approach as shown below in Table 7.

Figures 12, 13, and 14 show examples of real predictions of our approach on the MNIST handwritten digits dataset, MNIST fashion dataset, and CIFAR10 dataset, respectively, after transforming and geometrically distorting the testing data. Figure 15 shows the architecture of the CNN that was used to test the proposed approach. It comprises five Convolutional layers, two MaxPooling layers one Flatten layer, and two Dense Networks. Each one of the Convolutional is connected to the Heat Layer to provide it with the features that are needed to calculate the GHF of each feature. Then, the Heat Layer is connected with the Flatten vector before it is fed to the Dense networks.

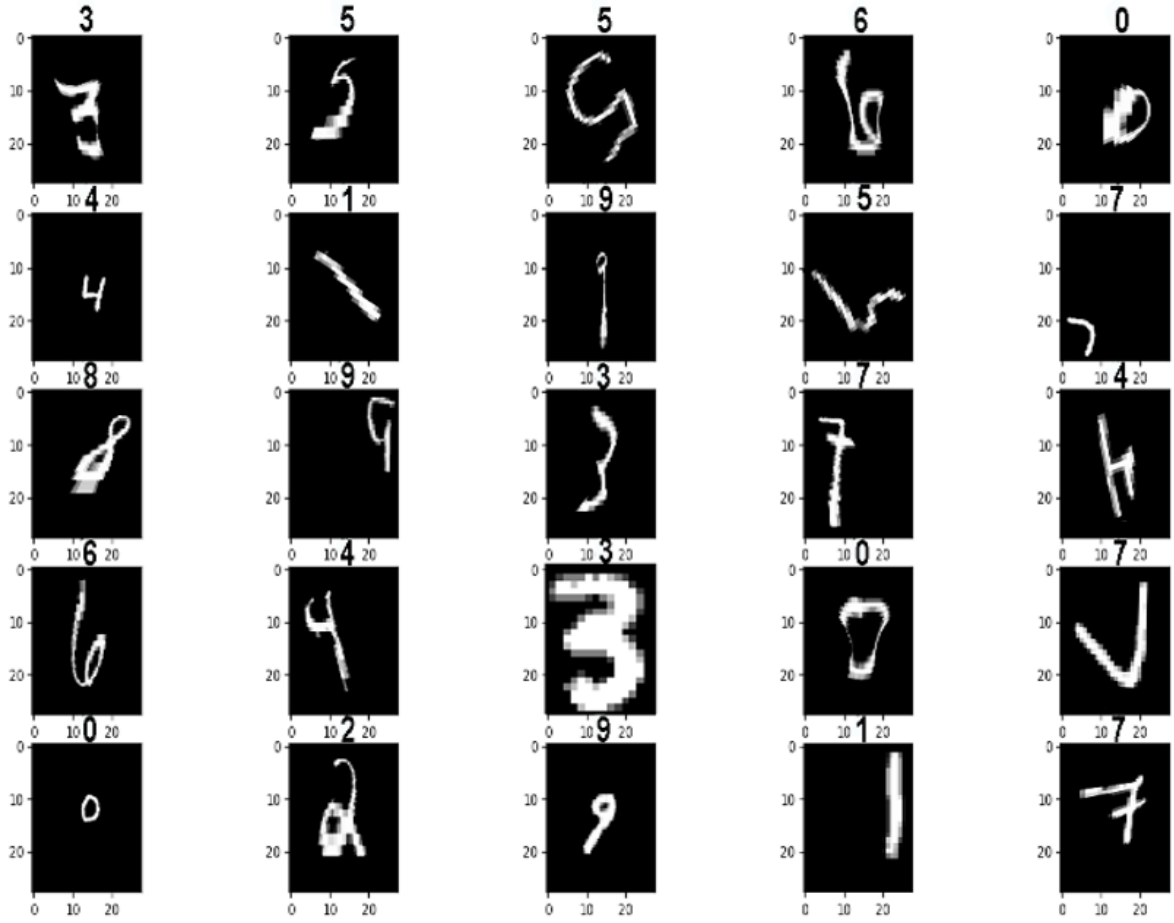

Fig. 12: Results of MNIST handwritten digits classification

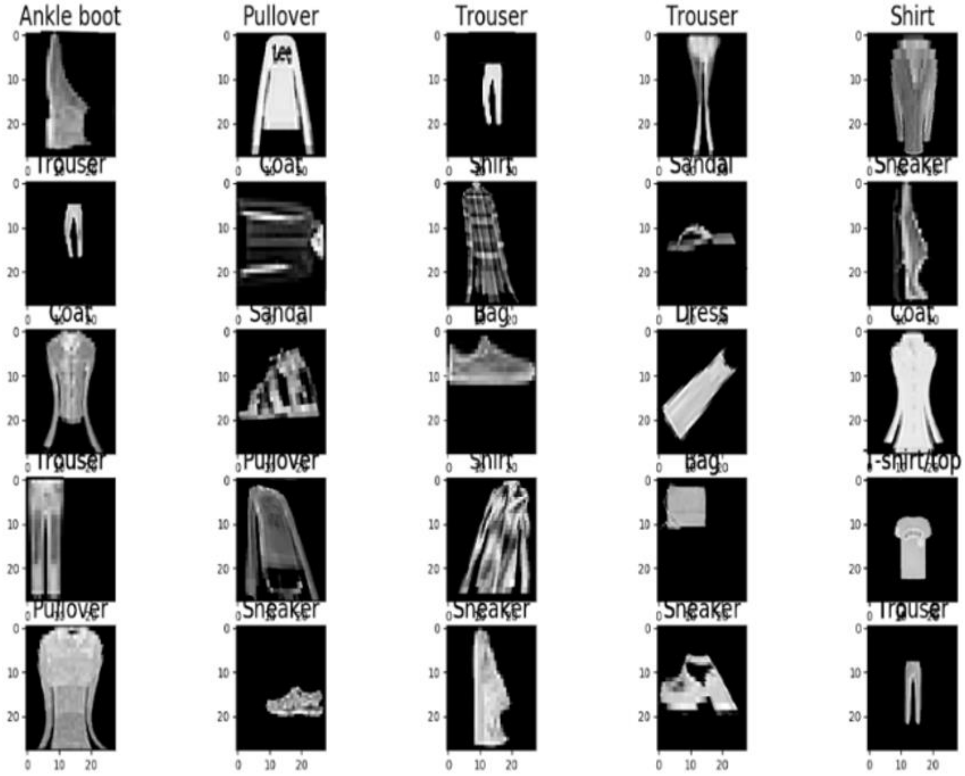

Fig. 13: Results of MNIST fashion dataset classification

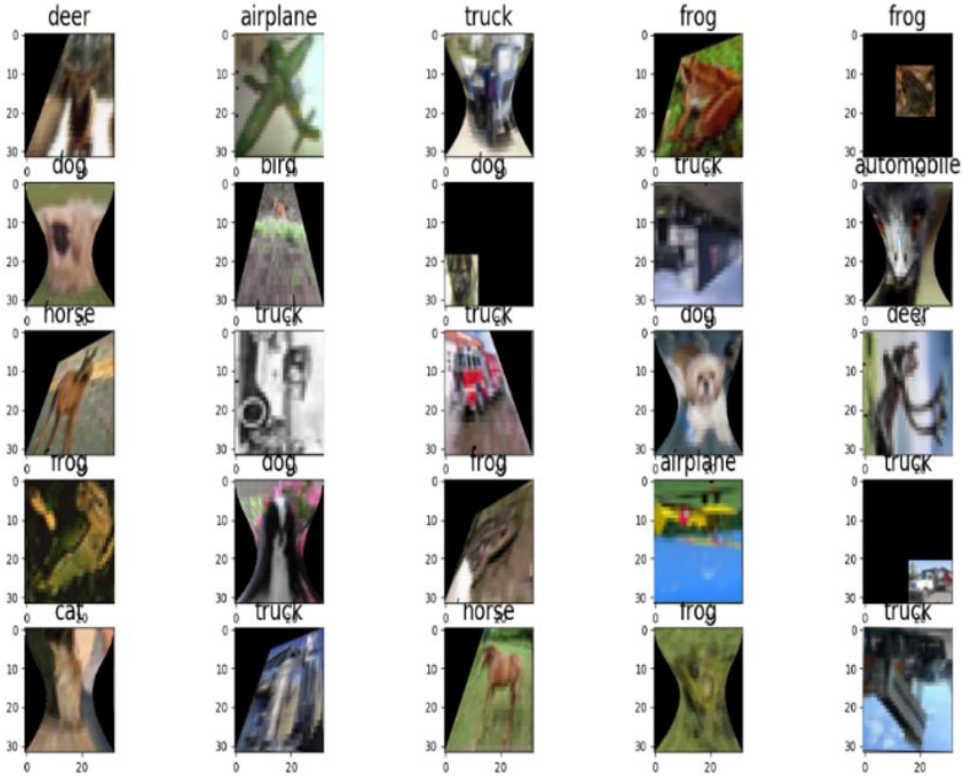

Fig. 14: Results of CIFAR10 dataset classification

Table 2: CNN-GHF MNIST handwritten digits results comparisons with (Shen *et al*., 2016)

Approach	# Of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	30	0.5850	94.78	96.90	97.10	96.99
	50	0.5450	95.89	97.40	97.24	97.31
	100	0.5410	95.98	97.49	97.46	97.47
(Shen *et al*., 2016)	30	0.0700	96.74	98.04	98.27	98.15
	50	0.0660	97.24	97.41	98.01	97.71
	100	0.0500	97.47	98.58	98.25	98.41
Geometric heat flow	30	0.0330	97.89	98.64	98.61	98.62
	50	0.0225	98.01	98.50	98.22	98.36
	100	0.0215	98.09	98.66	98.76	98.71

Table 3: CNN-GHF MNIST fashion results comparisons with (Shen *et al*., 2016)

Approach	# Of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	30	0.773	81.56	94.90	95.56	95.23
	50	0.658	82.46	95.35	95.70	95.52
	100	0.672	83.86	95.54	95.85	95.69
(Shen *et al.*, 2016)	30	0.315	87.13	96.87	96.81	96.84
	50	0.225	88.67	97.30	97.21	97.25
	100	0.220	88.83	97.67	97.50	97.58
Geometric heat flow	30	0.245	89.58	97.88	97.38	97.63
	50	0.185	92.31	98.06	97.78	97.92
	100	0.145	92.58	98.13	97.91	98.02

Table 4: CNN-GHF CIFAR10 results comparisons with (Shen *et al*., 2016)

Approach	# of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	300	1.257	54.79	93.85	91.60	92.71
	500	1.050	58.38	94.46	92.54	93.49
	1000	0.915	62.52	94.71	93.50	94.10
(Shen *et al*., 2016)	300	0.635	64.24	95.01	92.97	93.98
	500	0.444	74.46	95.34	93.41	94.37
	1000	0.416	83.26	95.70	94.26	94.97
Geometric heat flow	300	0.398	73.67	95.49	94.17	94.83
	500	0.235	80.80	95.82	95.34	95.58
	1000	0.206	86.09	96.18	96.10	96.14

Table 5: CNN-GHF MNIST handwritten digits results comparisons with (Raj, 2017)

Approach	# Of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	30	0.5850	94.78	96.90	97.10	96.99
	50	0.5450	95.89	97.40	97.24	97.31
	100	0.5410	95.98	97.49	97.46	97.47
(Raj, 2017) 45 Degrees	30	0.5650	97.51	97.40	97.45	95.38
	50	0.5450	96.19	97.56	97.50	97.45
	100	0.5370	96.28	97.61	97.55	97.50
(Raj, 2017) 90 Degrees	30	0.5630	95.48	97.53	97.43	97.47
	50	0.5540	95.78	97.58	97.49	97.53
	100	0.5470	96.08	97.60	97.53	97.56
Geometric heat flow	30	0.0330	97.89	98.64	98.61	98.62
	50	0.0225	98.01	98.50	98.22	98.36
	100	0.0215	98.09	98.66	98.76	98.71

Table 6: CNN- GHF MNIST fashion results comparisons with (Raj, 2017)

Approach	# Of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	30	0.773	81.56	94.90	95.56	95.23
	50	0.658	82.46	95.35	95.70	95.52
	100	0.672	83.86	95.54	95.85	95.69
(Raj, 2017) 45 Degrees	30	0.673	82.26	95.40	95.69	95.54
	50	0.639	83.56	95.56	95.80	95.68
	100	0.651	84.76	95.65	95.96	95.80

Table 6: Continuous

(Raj, 2017) 90 Degrees	30	0.691	82.06	95.48	95.51	95.49
	50	0.647	83.36	95.60	95.67	95.63
	100	0.599	84.46	95.67	95.87	95.77
Geometric heat flow	30	0.245	89.58	97.88	97.38	97.63
	50	0.185	92.31	98.06	97.78	97.92
	100	0.145	92.58	98.13	97.91	98.02

Table 7: CNN- GHF CIFAR10 results in comparisons with (Raj. 2017)

Approach	# Of Epochs	Loss	Acc.	Precision	Recall	F1 Score
Conventional CNN	300	1.2570	54.790	93.85	91.600	92.71
	500	1.0500	58.380	94.46	92.540	93.49
	1000	0.9150	62.520	94.71	93.500	94.10
(Raj, 2017) 45 Degrees	300	1.1170	55.390	94.65	91.890	93.25
	500	0.9670	59.590	94.66	93.130	93.89
	1000	0.7620	62.990	94.89	93.740	94.31
(Raj, 2017) 90 Degrees	300	1.1420	54.790	94.90	92.590	93.73
	500	1.0210	58.810	95.11	93.560	94.33
	1000	0.8530	62.490	95.55	93.790	94.66
Geometric heat flow	300	0.3980	73.670	95.49	94.170	94.83
	500	0.2350	80.800	95.82	95.340	95.58
	1000	0.2060	86.090	96.18	96.100	96.14

```
Layer (type)
=================================
conv2d_20 (Conv2D)

The Heat Layer

conv2d_21 (Conv2D)

The Heat Layer

conv2d_22 (Conv2D)

The Heat Layer

max_pooling2d_11 (MaxPooling 2D)

conv2d_23 (Conv2D)

The Heat Layer

conv2d_24 (Conv2D)

The Heat Layer

flatten_1 (Flatten)

dense_2 (Dense)

dense_3 (Dense)

=================================
Total params: 11,158,474
Trainable params: 11,158,474
Non-trainable params: 0
```

Fig. 15: The used CNN architecture

Discussion

The use of the Geometric Heat Flow of each feature in the image has led to an improvement in the performance of the conventional CNN, and since the GHF is invariant to geometric transformations and geometric distortions, it has made the CNN invariant as well.

Shen *et al*. (2016) in their approach, used random feature maps transformation to solve the invariant problem in the CNN, but this approach is more like data augmentation which is an inefficient approach to solve this problem. The use of Zenick moments as initial filters led to feature loss which led to an increase in loss and a decrease in accuracy. On the other hand, concatenating the Geometric Heat Flow achieves better loss, accuracy, precision, recall, and F1 score on the three datasets MNIST handwritten digits, MNIST fashion dataset, and CIFAR 10 dataset.

Conclusion

This study presents an approach to enhance CNN regarding the invariant problem by using geometric heat flow. The mechanism behind this approach is to add the Heat Layer to the CNN, which draws the contours of the objects in the image then computes the values of the geometric heat flow of the contours separately, then, concatenates the computed values of the geometric heat flow with the flattening vector before feeding it to the fully connected layer to make the vector more discriminative and more informative. In this study, we randomly transform and geometrically distort 90% of the testing data to test that our approach makes the CNN invariant to image transformations and geometrical distortions. Then we compare our work with the conventional CNN and the work of (Shen *et al*., 2016) and (Raj, 2017) on the three datasets, the MNIST handwritten digits, MNIST fashion dataset, and CIFAR 10 dataset, and we show that the proposed approaches classify transformed and geometrically distorted images

successfully. The results show that our method gives the best results in all cases namely loss, accuracy, precision, recall, and F1 score. The proposed approach achieves an accuracy of 98.09% on the MNIST dataset, 92.58% on the MNIST Fashion dataset, and 86.09% on CIFAR10.

Although this approach makes the CNN invariant it adds a huge burden on the CNN, as the GHF should be calculated for each feature in the Convolutional Layers, therefore the training time and the testing time are increased significantly. In the future, we will investigate the use of the GHF in other fields of image processing such as Shadow Detection.

Acknowledgment

We want to express our thanks and gratitude to all those who helped us throughout this study.

Author's Contributions

Sanad Aburass: Designed the approach, provided the results of the experiments and laid out the first draft of the paper.

Ammar Huneiti and Mohammad Belal Al-Zoubi: Supervised the design of the experiments and re-written some parts of the paper.

Ethics

This article is original and contains unpublished material. The corresponding author confirms that all of the other authors have read and approved the manuscript and no ethical issues involved.

References

Aburass, S., Huneiti, A., & Al-Zoubi, M. B. (2020). Enhancing Convolutional Neural Network using Hu's Moments. *International Journal of Advanced Computer Science and Applications*, *11*(12).

Alom, M. Z., Taha, T. M., Yakopcic, C., Westberg, S., Sidike, P., Nasrin, M. S., ... &Asari, V. K. (2019). A state-of-the-art survey on deep learning theory and architectures. *Electronics*, *8*(3), 292. DOI.org/10.3390/electronics8030292

Alvarez, L., Guichard, F., Lions, P. L., & Morel, J. M. (1993). Axioms and fundamental equations of image processing. *Archive for rational mechanics and analysis*, *123*(3), 199-257.

Anselmi, F., Leibo, J. Z., Rosasco, L., Mutch, J., Tacchetti, A., &Poggio, T. (2016). Unsupervised learning of invariant representations. *Theoretical Computer Science*, *633*, 112-121. DOI.org/10.1016/j.tcs.2015.06.048

Aubert, G., Kornprobst, P., & Aubert, G. (2006). *Mathematical problems in image processing:Partial differential equations and the calculus of variations* (Vol. 147, p. 26). New York: Springer. DOI.org/10.1109/ACCESS.2020.2982487

Bennett, C. O., & Myers, J. E. (1982). Momentum, heat and mass transfer (Vol. 370, p. 569). New York: McGraw-Hill.

Cao, F. (2003). *Geometric curve evolution and image processing*. Springer Science & Business Media.

Cheng, G., Han, J., Zhou, P., & Xu, D. (2018). Learning rotation-invariant and fisher discriminative convolutional neural networks for object detection. *IEEE Transactions on Image Processing*, *28*(1), 265-278. DOI.org/10.1109/TIP.2018.2867198

Chidester, B., Do, M. N., & Ma, J. (2018). Rotation equivariance and invariance in convolutional neural networks. *arXiv preprint arXiv:1805.12301*. DOI.org/10.48550/arXiv.1805.12301

Cohen, G., Afshar, S., Tapson, J., & Van Schaik, A. (2017, May). EMNIST: Extending MNIST to handwritten letters. In *the 2017 international joint conference on neural networks (IJCNN)* (pp. 2921-2926). IEEE. DOI.org/10.1109/IJCNN.2017.7966217

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H., & Wei, Y. (2017). Deformable convolutional networks. In *Proceedings of the IEEE international conference on computer vision* (pp. 764-773).

Dieleman, S., De Fauw, J., &Kavukcuoglu, K. (2016, June). Exploiting cyclic symmetry in convolutional neural networks. In *International Conference on Machine Learning* (pp. 1889-1898). PMLR.

Guo, Y., Li, Y., Wang, L., & Rosing, T. (2020, April). Adafilter: Adaptive filter fine-tuning for deep transfer learning. In *Proceedings of the AAAI Conference on Artificial Intelligence* (Vol. 34, No. 04, pp. 4060-4066). DOI.org/10.1609/aaai.v34i04.5824

Henriques, J. F., &Vedaldi, A. (2017, July). Warped convolutions: Efficient invariance to spatial transformations. In *International Conference on Machine Learning* (pp. 1461-1469). PMLR.

Hinton, G. E., Krizhevsky, A., & Wang, S. D. (2011, June). Transforming auto-encoders. In *International Conference on Artificial Neural Networks* (pp. 44-51). Springer, Berlin, Heidelberg. DOI.org/10.1007/978-3-642-39593-2_1

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A. R., Jaitly, N., ... & Kingsbury, B. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, *29*(6), 82-97. DOI.org/10.1109/MSP.2012.2205597

Jaderberg, M., Simonyan, K., & Zisserman, A. (2015). Spatial transformer networks. *Advances in neural information processing systems*, *28*.

Laptev, D., Savinov, N., Buhmann, J. M., &Pollefeys, M. (2016). Ti-pooling: Transformation-invariant pooling for feature learning in convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 289-297).

Lenc, K., &Vedaldi, A. (2015). Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 991-999).

Liu, W., Wang, Z., Liu, X., Zeng, N., Liu, Y., &Alsaadi, F. E. (2017). A survey of deep neural network architectures and their applications. *Neurocomputing*, *234*, 11-26. DOI.org/10.1016/j.neucom.2016.12.038

Mallat, S. (2013). Group invariant scattering. *Communications on Pure and Applied Mathematics*, *65*(10), 1331-1398. DOI.org/10.1002/cpa.21413

Marcos, D., Volpi, M., Komodakis, N., &Tuia, D. (2017). Rotation equivariant vector field networks. In *Proceedings of the IEEE International Conference on Computer Vision* (pp. 5048-5057).

McNeely-White, D. G., Beveridge, J. R., & Draper, B. A. (2019, August). Inception and ResNet: Same Training, same features. In *Biologically Inspired Cognitive Architectures Meeting* (pp. 352-357). Springer, Cham. DOI.org/10.1002/alr.22854

Mikula, K. (2002). Image processing with partial differential equations. In *Modern methods in scientific computing and Applications* (pp. 283-321). Springer, Dordrecht. DOI.org/10.1007/978-3-030-32040-9_12

Raj, A. N. J. (2017). Invariant moments-based convolutional neural networks for image analysis. *International Journal of Computational Intelligence Systems*, *10*, 936-950. DOI.org/10.2991/ijcis.2017.10.1.62

Sapiro, G. (2006). *Geometric partial differential equations and image analysis*. Cambridge university press.

Schmidt, U., & Roth, S. (2012, June). Learning rotation-aware features: From invariant priors to equivariant descriptors. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2050-2057). IEEE. DOI.org/10.1109/CVPR.2012.6247909

Serfling, R. (2010). Equivariance and invariance properties of multivariate quantile and related functions, and the role of standardization. *Journal of Nonparametric Statistics*, *22*(7), 915-936. DOI.org/10.1080/10485250903431710

Shen, X., Tian, X., He, A., Sun, S., & Tao, D. (2016, October). Transform-invariant convolutional neural networks for image classification and search. In *Proceedings of the 24th ACM international conference on Multimedia* (pp. 1345-1354). DOI.org/10.1145/2964284.2964316

Stewart, J., Clegg, D. K., & Watson, S. (2020). *Calculus*: *early transcendentals*. Cengage Learning.

Su, H., Maji, S., Kalogerakis, E., & Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision* (pp. 945-953).

Suzuki, S. (1985). Topological structural analysis of digitized binary images by border following. *Computer vision, graphics, and image processing*, *30*(1), 32-46. DOI.org/10.1016/0734-189X(85)90016-7

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., & Brostow, G. J. (2017). Harmonic networks: Deep translation and rotation equivariance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 5028-5037)