Distortion-aware CNNs for Spherical Images

Qiang Zhao[1], **Chen Zhu**[2,1], **Feng Dai**[1*], **Yike Ma**[1], **Guoqing Jin**[1], **Yongdong Zhang**[3]
[1] Key Laboratory of Intelligent Information Processing of Chinese Academy of Sciences, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
[3] University of Science and Technology of China, Hefei, China

Abstract

Convolutional neural networks are widely used in computer vision applications. Although they have achieved great success, these networks can not be applied to 360° spherical images directly due to varying distortion effect. In this paper, we present distortion-aware convolutional network for spherical images. For each pixel, our network samples a non-regular grid based on its distortion level, and convolves the sampled grid using square kernels shared by all pixels. The network successively approximates large image patches from different tangent planes of viewing sphere with small local sampling grids, thus improves the computational efficiency. Our method also deals with the boundary problem, which is an inherent issue for spherical images. To evaluate our method, we apply our network in spherical image classification problems based on transformed MNIST and CIFAR-10 datasets. Compared with the baseline method, our method can get much better performance. We also analyze the variants of our network.

1 Introduction

In the last decade, the society has witnessed great progress of deep neural networks [LeCun *et al.*, 2015], which have been widely used in computer vision, speech recognition, natural language processing, social network filtering and bioinformatics. As a specialized kind of deep neural networks, convolutional neural networks (CNNs) have made tremendous successes in computer vision and achieved state-of-the-art performance in many computer vision applications, including image classification [Krizhevsky *et al.*, 2012; Simonyan and Zisserman, 2015], object detection [Girshick *et al.*, 2014], semantic segmentation [Long *et al.*, 2015], and image super-resolution [Dong *et al.*, 2016].

By providing fields-of-view far beyond the conventional planar images, spherical images are becoming more and more popular and have been successfully applied in a number of recent new applications, including virtual navigation [Zhao *et al.*, 2013], 3D scene reconstruction [Micusik and Kosecka, 2009], scene recognition [Xiao *et al.*, 2012] and scene understanding [Yang and Zhang, 2016]. As a feature extractor, convolutional neural networks also have great application potentials in problems involving these particular type of data [Hu *et al.*, 2017; Lai *et al.*, 2017]. However, existing CNN architecture is awkward to process these images. This is because the underlying structure of spherical images is different with that of planar images processed by conventional CNNs. Specifically, the most existing convolutional networks are designed to deal with problems where the underlying image data have a regular planar grid structure, whereas the spherical images are the signals defined on the sphere as shown in Figure 1. Due to the non-linearity of this manifold, there is no regular planar grid available.

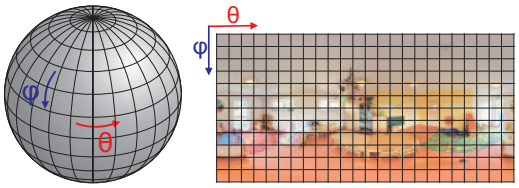

Figure 1: The structure of spherical image: the viewing sphere and equirectangular projected image.

To deal with this problem, two simple but naive approaches are adopted by communities. The first approach ignores the difference between the underlying structure of planar images and spherical images, and directly applies conventional CNNs to spherical images [Boomsma and Frellsen, 2017]. However this approach suffers from distortion problems. As the sphere is a non-developable surface, there does not exist any mapping method that will not introduce perceptual distortions [Zorin and Barr, 1995]. Equirectangular projection, which is the mapping method of spherical images, is not an exception. In the second strategy, each spherical image is converted to multiple perspective planar images. These planar images are then fed to CNNs for further processing. This approach is immune to image distortions, but needs to re-sample the spherical image. To achieve high accuracy, a large number of perspective images should be generated, which will increase the computational cost. Furthermore, the intermediate representation in CNN can not be shared across these perspective images. The reason is that the spherical im-

*Corresponding author: fdai@ict.ac.cn

age should be projected to different tangent planes. This prevents amortization of convolution operations as noted in [Su and Grauman, 2017].

In this paper, we propose a distortion-aware CNN for 360° spherical images. Our network is composed of distortion-aware convolutional layers and pooling layers, which *explicitly* take the distortions of spherical image into account. For each pixel, our method samples a non-regular grid based on its distortion level through perspective projection. As the sampling process already accounts for distortions, we can then use regular square kernels for convolution as in conventional CNN. Unlike the method proposed by Su and Grauman [2017], which breaks the parameter sharing and learns one kernel for each row of spherical image, the kernels in our network are shared by all the pixels. Thus our network has much less parameters and enjoys the parameter sharing property. For efficiency, our network uses the same set of small local sampling grids to successively approximates larger image patches. This avoids projecting viewing sphere to different tangent planes, which is computationally intensive for real problems. We also deal with the inherent boundary problem of spherical images to respect the fact that the sphere is a closed surface. To evaluate our method, we transform the well known MNIST and CIFAR-10 dataset to spherical ones and compare our method with baseline method, i.e. conventional CNNs trained and tested on the transformed dataset. Experimental results show that our method can get significantly better performance than the baseline.

The rest of this paper is organized as follows: Section 2 reviews the most related work including spherical images and generalized CNNs. We briefly describe the geometry for spherical images in Section 3, which is followed by the details of our algorithm in Section 4. Section 5 gives the evaluations of the proposed method. Section 6 concludes the paper and gives the possible directions of future work.

2 Related Work

In this section, we discuss some work related to this paper, covering spherical images and generalized CNNs.

2.1 Spherical Images

Unlike traditional planar images, which only record a small portion of the 3D scene, spherical images capture all the visual information of the world surrounding the view point. Thus they can provide field of view far beyond planar images and can give more immersive experience to the users.

Due to the development of image stitching techniques [Brown and Lowe, 2007] and the maturity of panoramic imaging systems [Uyttendaele *et al.*, 2004], the past decade has witnessed the increasing trend of spherical images being more and more easily obtained. These large field of view images are widely used to provide a vivid visual impression by the head mounted displays such as Oculus and HTC Vive, media sharing web sites such as Facebook and YouTube and online street view services such as Google Street View and Microsoft Bing Maps Streetside.

These data are also capturing the attention of research communities. Micusik and Kosecka [2009] reconstructed the 3D city models from street view spherical image sequences. SUN360 [Xiao *et al.*, 2012] project performs scene recognition and view detection by leveraging spherical image dataset. Zhao et al. [2013] proposed a real time virtual navigation method between two spherical images. Yang and Zhang [2016] recovered the shape of a 3D room from an indoor spherical image. To extract features, these methods first convert the spherical image to a set of perspective images, then extract features from these planar images. Contrasted, in this paper, we propose convolutional networks, which can directly extract features from spherical images.

2.2 CNNs on General Domains

Conventional CNNs are designed to solve problems where the coordinates of the underlying data representation have a regular grid structure. Recently, there appear works that try to apply convolutional networks on more general domains, such as graph and spherical signals.

As a natural generalization of grid structure, graphs offer the possibility for extending the notion of convolution. Bruna et al. [2014] discussed how to extend the convolutional architectures to graph structures. They proposed two constructions of deep neural networks on graphs. The first one is spatial construction, which is based on the local filtering and hierarchical clustering of the nodes of graph. The other one is spectral construction, which is based on the spectrum of graph Laplacian. Duvenaud et al. [2015] used convolutional networks on graphs to learn molecular fingerprints. Their construction is similar to the spatial construction of Bruna, but can take the graphs of arbitrary size and shape as input.

To process the signals on sphere, Khasanova and Frossard [2017] treated each spherical image as a graph and used graph-based CNN to classify these images. Cohen et al. [2017] proposed spherical convolutional networks. Their method is based on the convolutions on the sphere and rotation group. However this method may suffer from bandwidth problem, since the convolutions on the sphere are solved by generalized Fourier transform. Su and Grauman [2017] transformed the spherical convolutional network learning problem to a model distillation problem. Their method transfers a pre-trained conventional CNN model to a new network, so that the transferred network can be used to process spherical images. To account for the varying distortion effects, they learned one kernel for each row of the spherical image. This makes the transferred network has huge number of parameters and slows the rate of convergence when training. Our proposed CNN also processes equirectangular projected spherical images directly. However our method inherits the parameter sharing property of conventional CNNs and has much less parameters than that of Su and Grauman [2017].

3 Preliminary

Our goal is to design CNNs that can be applied directly to spherical images. Before introducing our method, we first describe the structure of spherical images, i.e. equirectangular projection.

As shown in Figure 1, equirectangular projection maps meridians and circles of latitude of viewing sphere to vertical

Figure 2: The local patches with same polar angle (the red and green regions) in spherical image have same distortion, while that with different polar angles (the red and blue regions) have different distortion effects. The boundary of spherical image can split one object into two parts (the magenta regions).

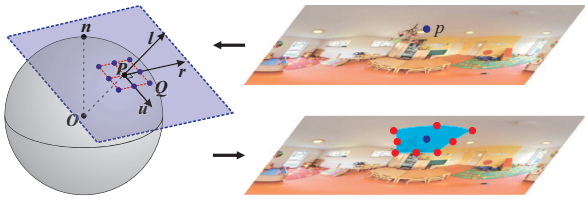

Figure 3: To sample the non-regular grid for $\mathbf{p}$, we project the viewing sphere to the tangent plane at $\mathbf{P} = \mathcal{T}(\mathbf{p})$ and sample a regular grid on the tangent plane (the blue points on the left subfigure). Then the sampled grid is transformed on the original spherical image for the target grid (the points on the right below image).

and horizontal coordinates of spherical image respectively. Given a pixel (x, y) of the spherical image, we can obtain its spherical coordinates (θ, ϕ) on the viewing sphere as

$$\begin{cases} \theta = \frac{2x\pi}{w} \\ \phi = \frac{y\pi}{h} \end{cases}, \tag{1}$$

where w and h are the width and height of spherical image respectively, θ is the azimuthal angle and ϕ is the polar angle. Then the 3D position of the pixel on the viewing sphere is

$$\begin{cases} X = \cos(\theta)\sin(\phi) \\ Y = \sin(\theta)\sin(\phi) \\ Z = \cos(\phi) \end{cases}. \tag{2}$$

Correspondingly, given a point (X, Y, Z) on the viewing sphere, its 2D pixel coordinates (x, y) can be determined by the inverse transformation of the above procedure. For convenience, we denote the transformation as $\mathcal{T}$ and the inverse transformation as $\mathcal{T}^{-1}$ in the following.

4 Our Approach

In this section, we first analyze the problem of spherical image convolution, then we describe the distortion-aware convolution and pooling. Finally, we give the discussions about our approach and the implementation details.

4.1 Problem Analysis

The basic operation in CNN is convolution. This operation first samples a grid over the input feature map $\mathbf{f}$ around a location $\mathbf{p}$, then takes the sum of sampled values weighted by convolution kernel $\mathbf{w}$ and assigns the result to the corresponding element of output feature map $\mathbf{g}$. This is expressed as

$$\mathbf{g}(\mathbf{p}) = \sum_{\Delta\mathbf{p}\in\mathcal{R}} \mathbf{w}(\Delta\mathbf{p})\mathbf{f}(\mathrm{s}(\mathbf{p}, \Delta\mathbf{p})), \tag{3}$$

where $\mathrm{s}(\mathbf{p}, \Delta\mathbf{p})$ denotes the sampling process and $\mathcal{R}$ defines the convolution region $\{\mathrm{s}(\mathbf{p}, \Delta\mathbf{p})|\Delta\mathbf{p} \in \mathcal{R}\}$. The set $\mathcal{R}$ is $\{(-1,-1), (-1,0), \ldots, (1,0), (1,1)\}$, for example, when the convolution kernel size is 3×3.

Conventional CNNs always operate on the data that has a regular grid structure, i.e.

$$\mathrm{s}(\mathbf{p}, \Delta\mathbf{p}) = \mathbf{p} + \Delta\mathbf{p}, \ \forall\Delta\mathbf{p} \in \mathcal{R}. \tag{4}$$

However this regular grid sampling strategy can not be applied to spherical images directly. The main reason is that equirectangular projection will introduce varying distortions. That is to say, the distortions of local patches in spherical image only depend on their polar angles ϕ and are independent of azimuthal angles θ. This is validated by Figure 2, in which we back project planar images of the same size to different locations of spherical image. We can see that if two locations have equal ϕ, the distortion effects of projected regions would be same, e.g. the two regions bounded by red and green curves. Another reason is that the boundary pixels of spherical image and planar image should be processed differently. For planar images, if $\mathbf{p}$ is a boundary pixel, zero padding should be applied when sampling the grid through Equation 4, as $\mathbf{p} + \Delta\mathbf{p}$ may be an invalid pixel position. For spherical images, the padding should be prohibited. Instead, wrap address mode should be used as the left and right boundaries of spherical image correspond to the same meridian of viewing sphere. The region bounded by magenta curve in Figure 2 shows this problem.

To solve the varying distortion problem, Su and Grauman [2017] adopted the usual regular grid sampling in Equation 4, but used different convolution kernels $\mathbf{w}$ for the pixels in different rows. These kernels have different shapes and sizes. For example, kernels $\mathbf{w}$ for pixels close to top and bottom boundaries of image are long and narrow rectangles, while those for central pixels are squares. In contrast, our main idea is sampling a *non*-regular grid around location $\mathbf{p}$ through sampling operator $\mathrm{s}(\mathbf{p}, \Delta\mathbf{p})$, and performing convolution using Equation 3 without changing the shape and/or size of convolution kernel $\mathbf{w}$, which is shared by all the pixels of spherical image. Thus our network has much less parameters than the work [Su and Grauman, 2017] and is easy and fast to train.

4.2 Distortion-aware Convolution

To sample the distortion-aware non-regular grid for $\mathbf{p}$, we project the viewing sphere to the tangent plane at point $\mathbf{P} = \mathcal{T}(\mathbf{p})$. Then we sample a regular grid on the tangent plane and transform the grid to the original spherical image for the target non-regular grid.

We first define a 3D Cartesian coordinate system, whose origin is given by the point $\mathbf{P}$ and axes are determined by the tangent plane. As shown in Figure 3, we set the look-at axis $\mathbf{l}$ of the coordinate system as $\mathbf{l} = \mathcal{T}(\mathbf{p})$, and compute the right axis as

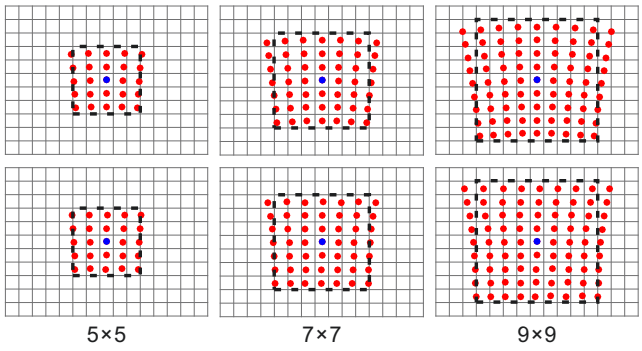

Figure 4: Effective receptive field of convolution layer with large kernels (top) and that determined by stacked convolution layers with small 3×3 kernels (bottom). In each subfigure, the black square indicates the receptive field of conventional CNNs.

$$\mathbf{r} = \mathbf{l} \times \mathbf{n}, \tag{5}$$

where $\mathbf{n} = (0, 0, 1)$ is the north pole of the viewing sphere. Then the up axis $\mathbf{u}$ is the cross product of look-at axis and right axis, i.e.

$$\mathbf{u} = \mathbf{l} \times \mathbf{r}. \tag{6}$$

Because the patches projected to tangent plane have no distortions, we can safely sample a regular gird on the tangent plane. For example, the 3D coordinates of point $\mathbf{Q}$ on the tangent plane with offset $\Delta\mathbf{p}$ are

$$\mathbf{Q} = \mathbf{P} + \Delta\mathbf{p}\frac{2\pi}{w}[\mathbf{r}\ \mathbf{u}]^{\mathrm{T}}, \tag{7}$$

where $\frac{2\pi}{w}$ computes the scale of one pixel under the assumption that the viewing sphere is an unit sphere. After a regular grid is sampled on the tangent plane, we transform it to original spherical image through $\mathcal{T}^{-1}(\mathcal{N}(\mathbf{Q}))$, where $\mathcal{N}(\mathbf{Q}) = \mathbf{Q}/\|\mathbf{Q}\|$ is the normalization operation. Finally, the sampling operator taking distortions into account is

$$\mathrm{s}(\mathbf{p}, \Delta\mathbf{p}) = \mathcal{T}^{-1}(\mathcal{N}(\mathcal{T}(\mathbf{p}) + \Delta\mathbf{p}\frac{2\pi}{w}[\mathbf{r}\ \mathbf{u}]^{\mathrm{T}})). \tag{8}$$

Based on Equation 8, we can sample a non-regular grid and perform distortion-aware convolution using Equation 3.

4.3 Distortion-aware Pooling

In convolutional networks, pooling operation is often used to provide translation invariance and reduce the size of representation. Our networks also include distortion-aware pooling layers, which use the operator in Equation 8 to sample the pooling region.

4.4 Discussion

In this section, we give some discussions about the boundary problem and receptive field.

Boundary Problem

The projection given in Section 4.2 is primarily introduced to deal with the distortion problem. As we treat the spherical image as a signal on the sphere during projection, it also naturally solves the boundary problem of spherical images, which is not touched in [Su and Grauman, 2017].

Receptive Field

Existing convolutional networks often utilise very small kernels, i.e. 3×3, in convolutional layers. Besides of increasing the depth of networks [Simonyan and Zisserman, 2015], this strategy also implicitly increases the size of effective receptive field, which is necessary for performance improvement. In this paper, we also use small 3×3 convolution kernels throughout the whole net. However, the effective receptive field of multiple stacked convolutional layers with small kernels is not the same as that provided by one layer with larger kernels. This is because the 9 pixels in each 3×3 kernels correspond to different tangent planes, i.e. different points of tangency $\mathcal{T}(\mathbf{p})$, we can not simply take the union of their receptive fields to form a larger one. Therefore stacking multiple layers with small receptive field is just a way to approximate one layer with larger receptive field, which is more reasonable. To access whether the differences would greatly affect the performance of our network, we illustrate the effective receptive field of one convolution layer with large kernels and that determined by stacked convolution layers with small 3×3 kernels. From Figure 4 we can see that the sampling locations of receptive field are similar, which encourages us to use small kernels in our networks. We also note that the receptive field of conventional CNNs indicated by black square has much deviation from reasonable receptive field. This is the main reason why conventional CNNs have low performance on spherical images.

4.5 Implementation

We implement our distortion-aware CNNs based on Caffe framework [Jia *et al.*, 2014]. In Caffe, convolution is reduced to matrix-matrix multiplication, which is highly optimized in BLAS libraries and can be efficiently computed on GPU devices. This involves rearranging image patches into matrix columns during forward pass by `im2col` and remapping the matrix back to the image during backpropagation by `col2im`. Therefore we only need to rewrite these functions. Distortion-aware pooling is implemented by rewriting corresponding functions in file `pooling_layer`.

In our CPU implementation, we precompute the sampling offsets $\mathrm{s}(\mathbf{p}, \Delta\mathbf{p}) - \mathbf{p}$ for each polar angle ϕ and then use them during forward and backward pass. This strategy can greatly reduce the computation cost as the distortions only depend on ϕ. For GPU version, we implement the sampling within CUDA kernels without precomputation, which can be executed parallelly. Because the sampling operator $\mathrm{s}(\mathbf{p}, \Delta\mathbf{p})$ can give fractional locations, bilinear interpolation is used when accessing the features $\mathbf{f}(\mathrm{s}(\mathbf{p}, \Delta\mathbf{p}))$ in Equation 3.

5 Experimental Result

In this section, we evaluate our approach. We first introduce the experimental setup, then give the main result of our method and a baseline method on two well known dataset. Finally, we give some analysis about our networks.

5.1 Experimental Setup

Experimental setup is introduced from three aspects: the dataset used in our experiment, the network architecture and the baseline methods.

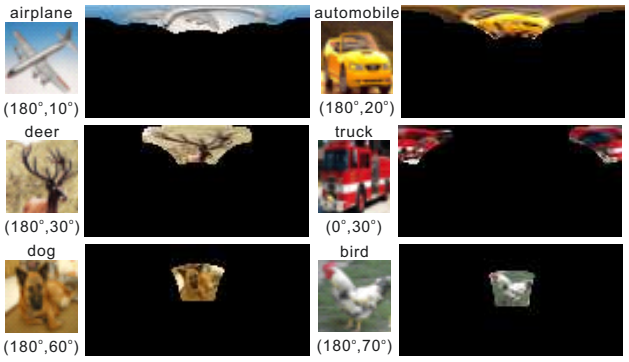

Figure 5: Converted CIFAR-10 dataset: for each image, we show its class, backprojection location (θ, ϕ) and converted image. Note that compared with the first two rows, the converted images in the third row have less distortions.

Dataset

A well-known spherical image dataset is SUN360 dataset [Xiao *et al.*, 2012], which contains true spherical images belonging to different place categories. However the discriminative contents of the images in this dataset all locate at the less distorted central regions, which can not be used to thoroughly evaluate the performances of our method. To make the discriminative contents of the images having different levels of distortions, we leverage two existing planar image datasets, i.e. CIFAR-10 [Krizhevsky, 2009] and MNIST [Lecun *et al.*, 1998], and transform the images in each dataset to spherical ones as if they are captured by panoramic cameras. The CIFAR-10 dataset consists of $50,000$ training images and $10,000$ testing images in 10 classes. Each 32×32 image in this dataset is randomly back projected to locations $(\theta, \phi) \in \{0°, 180°\} \times \{10°, 20°, 30°\}$ on spherical image. This gives us a dataset of spherical image of resolution 128×64. A number of exemplar converted spherical images, their classes and back projection locations are shown in the first two rows of Figure 5. MNIST is a dataset of handwritten digits, which has a training set of $60,000$ images and a test set of $10,000$ images. For each 28×28 images of this dataset, we convert it to a 112×56 spherical image using the same method for CIFAR-10 dataset transformation.

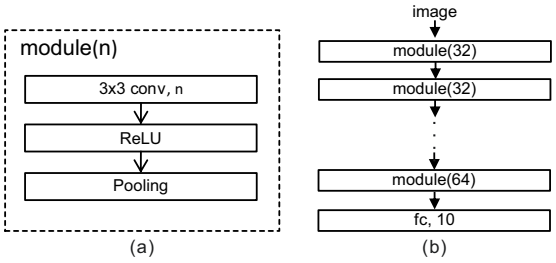

Figure 6: The network architecture in our experiment: (a) Each module `module(n)` contains a 3×3 convolutional layer with n feature maps and a pooling layer. Between them, a ReLU activation function is used for non-linearity. (b) The network is composed of different numbers of modules and ends with a 10-way fully-connected layer and softmax.

Baseline Methods

Three methods can be used as baseline. The first method is conventional CNN that trains the model on the original planar CIFAR-10 dataset or MNIST dataset, and tests it on the corresponding transformed dataset. However, according our experiments, this method is not much better than random guess, e.g. 10.53% classification accuracy for CIFAR-10 dataset. Therefore we do not give the performance of this baseline in the following sections. The second method is converting each spherical image in dataset into a cube map and applying conventional CNNs on each face of the cube map. Although this method does not suffer from spherical distortion, it can not avoid perspective projection distortion [Jaderberg *et al.*, 2015]. Thus this method only has a little better performance than random guess, e.g. 24.87% classification accuracy for CIFAR-10 dataset, which is also not listed in the following. The third method is training and testing conventional CNN both on the transformed datasets, which is used as baseline in the rest of this paper.

Network Architecture

We stress that our goal is not to achieve competitive performance on spherical datasets compared with the state-of-the-art methods for planar datasets, but to *demonstrate that our method can get better performance than baseline*. Therefore we use simple network for both our method and baseline method. For fair comparison, the architecture and hyperparameters of network in our method and baseline are same. As shown in Figure 6, the networks are composed of different numbers of modules, each of which contains a 3×3 convolutional layer and a pooling layer. ReLU activation function is used for non-linearity. For networks containing 5 modules, the size of feature map in each module is $32, 32, 64, 64$ and 64 respectively. The network is ended with a 10-way fully-connected layer and softmax. When training the networks, we use ADAM algorithm [Kingma and Ba, 2014] with mini-batch size of 128. The learning rate is set as 10^{-3}.

5.2 Main Result

The classification accuracies of our method and baseline with 5 modules using average pooling on transformed MNIST dataset and CIFAR-10 dataset are listed in Table 1 [1]. From the table we can see that our method achieve better performance than baseline method. On the simple MNIST dataset, both methods have accuracy higher than 90.00%. Our method achieves 2.10% higher performance than the baseline method. Compared with MNIST dataset, the CIFAR-10 dataset consists of natural images and is more complex. On this dataset, our method gains 4.51% improvement, which validates the importance of distortion-aware CNN in more difficult classification tasks. This encourages us to test its performance on larger dataset in future.

5.3 Analysis

Many factors may influence the performance of convolutional networks. In this section, we analyse the change of classification accuracy of networks on CIFAR-10 dataset when we

[1] The comparison between average pooling and max pooling is discussed in Section 5.3.

Table 1: The classification accuracy (%) of different methods on MNIST and CIFAR-10 datasets: in the third and fifth columns, the number in the bracket indicates the performance improvement of our method compared with baseline. The following tables can also be read with the same way as this.

	max pooling		average pooling	
dataset	baseline	ours	baseline	ours
MNIST	97.48	98.22 (0.74)	92.98	95.08 (2.10)
CIFAR	61.84	65.11 (3.27)	60.11	64.62 (4.51)

use different pooling functions, different numbers of module layers and different levels of distortion for spherical dataset. The result on MNIST is given in Table 4 without discussion for the sake of space.

Pooling Type

In convolutional networks, the two most popular pooling functions are max pooling and average pooling. Besides of average pooling, we have also tested the classification accuracies of networks with 5 modules when max pooling is used. The performance of networks with different pooling functions are given in Table 1. We can see that our method is always better than baseline method, especially when average pooling is used. Compared with average pooling, the networks with max pooling achieve better performance both in our method and baseline method. For example, the max pooling in baseline makes 1.73% improvement, while that in our method only makes 0.49% improvement. This shows that our network is less sensitive to the type of pooling functions.

Module Layers

We test four variants for the networks of our method and baseline method. These variants contain $2, 3, 4$ and 5 layers respectively. The performance of different variants is shown in Table 2. Stacking multiple convolutional layers would increase the overall network receptive field. Theoretically, this would also improve the network performance. From the table we can see that this holds true for our method and baseline method, where the networks having 5 layers achieve the best performance. Another fact we can get from the table is that our method is consistently better than baseline method regardless of what pooling function is used and how many module layers are contained in the networks. The performance improvement is highest when there are 5 layers. This motivates us to use deeper networks in the task. However when there are 6 module layers, it appears to be overfitting the training dataset. Therefore the networks in this paper have depth of up to 5 layers (except for the last fully-connected layer).

Table 2: The classification accuracy (%) of different methods with different number of module layers on CIFAR-10 dataset

	max pooling		average pooling	
♯ layers	baseline	ours	baseline	ours
2	52.80	54.32 (1.52)	52.03	53.74 (1.71)
3	59.59	61.72 (2.13)	59.56	60.85 (1.29)
4	62.32	63.78 (1.46)	60.30	63.74 (3.44)
5	61.84	65.11 (3.27)	60.11	64.62 (4.51)

Table 3: The classification accuracy (%) of different methods on CIFAR-10 dataset with different level of distortions

distortion	max pooling		average pooling	
level	baseline	ours	baseline	ours
more	61.84	65.11 (3.27)	60.11	64.62 (4.51)
less	66.99	67.65 (0.66)	64.81	67.46 (2.65)

Levels of Distortion

Strictly speaking, the converted images shown in Figure 5 are not spherical images, as they contain undefined black regions. Therefore these images only simulate a limited number of different distortion effects compared with true spherical images. In this section, we additionally collect a new spherical image dataset by randomly back projecting each image of CIFAR-10 dataset to locations $(\theta, \phi) \in \{0°, 180°\} \times \{60°, 70°, 80°\}$. As these projection locations are near to equator, the new dataset has less distortion than the dataset introduced in Section 5.1 (the third row vs. the first two rows in Figure 5). Table 3 gives the performance of our method and baseline method on the old more-distorted dataset and the new less-distorted dataset. Our method is better than the baseline method on both dataset with either max pooling or average pooling. For each method, it can get much better performance when there are less distortions. For example, baseline method achieve about 5.15% and 4.70% higher accuracy on the less distorted dataset when max pooling and average pooling are used perspectively. The difference of our method on these two dataset is small, i.e. 2.54% and 2.84%, due to our distortion-aware convolution and pooling. The improvement of our method over baseline method is more obvious when there are more distortions. Thus our distortion-aware CNNs are indispensable tools for spherical image processing, especially when meaningful parts of these images are more distorted.

6 Conclusion

In this paper, we introduce distortion-aware CNNs for spherical images. To account for varying distortion effects of spherical images, our method samples different non-regular regions and use the same convolution kernel for different locations when performing convolution. Our method also deals with the boundary problem, which is an inherent issue for spherical images. Experimental results on two datasets show that our method does not suffer from distortion problem and can get better performance than baseline methods.

In future, we would like to test the performance of distortion-aware CNNs on more difficult classification tasks. We also would like to investigate the possibility of apply-

Table 4: The classification accuracy (%) of different methods with different number of module layers on MNIST dataset

	max pooling		average pooling	
♯ layers	baseline	ours	baseline	ours
2	92.49	93.02 (0.53)	88.90	90.42 (1.52)
3	96.89	97.76 (0.87)	93.69	95.34 (1.65)
4	97.49	98.20 (0.71)	93.97	95.17 (1.20)
5	97.48	98.22 (0.74)	92.98	95.08 (2.10)

ing our distortion-aware CNNs on other tasks, such as object detection and semantic segmentation. Evaluating the performance of distortion-aware CNNs on 360° videos is another direction of future work.

Acknowledgments

This work is supported by National Key R&D Program of China (2016YFB0801203), National Natural Science Foundation of China (61525206,61702479,61771458), and the Science and Technology Service Network Initiative of the Chinese Academy of Sciences (KFJ-STS-ZDTP-018).

References

[Boomsma and Frellsen, 2017] Wouter Boomsma and Jes Frellsen. Spherical convolutions and their application in molecular modelling. In *NIPS*, pages 3436–3446. 2017.

[Brown and Lowe, 2007] Matthew Brown and David G. Lowe. Automatic panoramic image stitching using invariant features. *IJCV*, 74(1):59–73, 2007.

[Bruna *et al.*, 2014] Joan Bruna, Wojciech Zaremba, Arthur Szlam, and Yann LeCun. Spectral networks and deep locally connected networks on graphs. *arXiv: 1312.6203v3*, 2014.

[Cohen *et al.*, 2017] Taco S. Cohen, Mario Geiger, Jonas Köehler, and Max Welling. Convolutional networks for spherical signals. *arXiv: 1709.04893*, 2017.

[Dong *et al.*, 2016] Chao Dong, Chen Change Loy, and Xiaoou Tang. Accelerating the super-resolution convolutional neural network. In *ECCV*, pages 391–407, 2016.

[Duvenaud *et al.*, 2015] David K Duvenaud, Dougal Maclaurin, Jorge Iparraguirre, Rafael Bombarell, Timothy Hirzel, Alan Aspuru-Guzik, and Ryan P Adams. Convolutional networks on graphs for learning molecular fingerprints. In *NIPS*, pages 2224–2232. 2015.

[Girshick *et al.*, 2014] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[Hu *et al.*, 2017] Hou-Ning Hu, Yen-Chen Lin, Ming-Yu Liu, Hsien-Tzu Cheng, Yung-Ju Chang, and Min Sun. Deep 360 pilot: Learning a deep agent for piloting through 360° sports videos. In *CVPR*, pages 1396–1405, 2017.

[Jaderberg *et al.*, 2015] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and koray kavukcuoglu. Spatial transformer networks. In *NIPS*, pages 2017–2025. 2015.

[Jia *et al.*, 2014] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM MM*, pages 675–678, 2014.

[Khasanova and Frossard, 2017] Renata Khasanova and Pascal Frossard. Graph-based classification of omnidirectional images. In *ICCV workshop*, pages 869–878, 2017.

[Kingma and Ba, 2014] Diederik P. Kingma and Jimmy Lei Ba. Adam: a method for stochastic optimization. *arXiv: 1412.6980*, 2014.

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105. 2012.

[Krizhevsky, 2009] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

[Lai *et al.*, 2017] Wei-Sheng Lai, Yujia Huang, Neel Joshi, Christopher Buehler, Ming-Hsuan Yang, and Sing Bing Kang. Semantic-driven generation of hyperlapse from 360° video. *IEEE TVCG*, 2017.

[Lecun *et al.*, 1998] Yann Lecun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[LeCun *et al.*, 2015] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–444, 2015.

[Long *et al.*, 2015] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, pages 3431–3440, 2015.

[Micusik and Kosecka, 2009] Branislav Micusik and Jana Kosecka. Piecewise planar city 3D modeling from street view panoramic sequences. In *CVPR*, pages 2906–2912, 2009.

[Simonyan and Zisserman, 2015] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[Su and Grauman, 2017] Yu-Chuan Su and Kristen Grauman. Learning spherical convolution for fast features from 360° imagery. In *NIPS*, pages 529–539. 2017.

[Uyttendaele *et al.*, 2004] Matthew Uyttendaele, Antonio Criminisi, Sing Bing Kang, Simon Winder, Richard Szeliski, and Richard Hartley. Image-based interactive exploration of real-world environments. *IEEE CGA*, 24(3):52–63, 2004.

[Xiao *et al.*, 2012] Jianxiong Xiao, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *CVPR*, pages 2695–2702, 2012.

[Yang and Zhang, 2016] Hao Yang and Hui Zhang. Efficient 3d room shape recovery from a single panorama. In *CVPR*, pages 5422–5430, 2016.

[Zhao *et al.*, 2013] Qiang Zhao, Liang Wan, Wei Feng, Jiawan Zhang, and Tien-Tsin Wong. Cube2Video: Navigate between cubic panoramas in real-time. *IEEE TMM*, 15(8):1745–1754, 2013.

[Zorin and Barr, 1995] Denis Zorin and Alan H. Barr. Correction of geometric perceptual distortions in pictures. In *Annual Conference on Computer Graphics and Interactive Techniques*, pages 257–264, 1995.