2020 19th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)

Upgrade your network in-place with deformable convolution

Wei Xi
School of Internet of Things Engineering
Jiangnan University
Wuxi, China
6171910034@stu.jiangnan.edu.cn

Li Sun
School of Internet of Things Engineering
Jiangnan University
Wuxi, China
lisun@jiangnan.edu.cn

Jun Sun
School of Internet of Things Engineering
Jiangnan University
Wuxi, China
junsun@jiangnan.edu.cn

***Abstract*—Improving the performance of the network on is a topic that all deep learning researchers are working together. More new algorithms are proposed for different tasks. But most of these can't avoid spending a lot of time retraining the network model. Deformable convolution is a convolution structure that can extract better features of objects. This paper proposes a new method that can upgrade the standard convolution part of the network to the deformable convolution in-place, inherit the original model parameters, and reduce the time and computational resource cost for retraining. We analyzed the effects of introducing deformable convolution at different depths of the network on speed and performance. And on the detection and semantic segmentation tasks of the PASCAL VOC and COCO, a lot of experiments were carried out on our methods, and have an effective improvement.**

Keywords-component; upgrade; deformable convolution; in-place (key words)

I. INTRODUCTION *(HEADING 1)*

Due to the ability of deep neural networks to fit and generalize data sets through a large number of parameters, many problems that cannot be solved in non-deep learning way can be solved. Especially in the field of computer vision, the convolution operation has the advantages of local connection and weight sharing, which makes the convolutional neural network based on convolution operation in multiple computer vision tasks, such as detection, tracking, and semantic segmentation, instance segmentation, pose estimation and so on, achieve the state of art results.

When we already have a baseline algorithm for a task, we always want to use some methods to get further improvement. Fig .1 shows a common training model. The researchers often choose the following methods:

1) *Increase the quality and quantity of train data.[1]*
2) *Better parameter optimization.[2]*
3) *Change better network structure.[3]*

Buts all the previous methods require full retraining, which consumes a lot of time and computational costs.

We need a simple, in-place upgrade, do not full retraining, does not significantly increase network parameters, and an easy-to-implement method to enhance network performance. Deformable convolution [4] is a notable convolution module.

Standard convolution operations have better advantages in Euclidean data than fully connection operations due to local connections and weight sharing, but there are still some problems. First, the standard convolution can only rely on the stacking convolution module or expanding the size of the convolution kernel to get a larger receptive field [5]. Second, the standard convolution can only sample the data using grid sampling, but the distribution of the data is often not at the grid points, such as the characteristics of a circle or a triangle. For features with partial deformations, standard convolution does not extract features well, but deformable convolution operation solves these two problems.

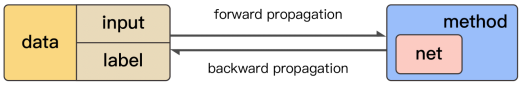

Figure 1. Different part of a training model.

The deformable convolution operation is an extension of the standard convolution operation. It uses an additional standard convolution operation to introduce a spatial offset to the value on the grid point compared to the standard convolution, used to change the position of the uniform grid sampling, so that the convolution operation can learn free deformation information. This does not only refer to changes in shape, but also includes changes in the range of receptive fields. The learning of this bias comes from inputting the feature map of the current deformable convolution module. After sufficient learning, different positions on different feature maps will generate different offset information for the current convolution kernel.

Given the advantages of deformable convolution, networks using deformable convolution have achieved good results on different tasks. But this still does not get rid of the shortcomings of the need to completely retrain the network. In this paper, we propose an algorithm that uses deformable convolution to enhance the effects of existing networks. This algorithm can upgrade the standard convolution in the network to a deformable convolution, and the network upgrade is in-place. It will not degrade the results of the network, only a small amount of fine-tuning with data can achieve better results than before. A series of experiments have been designed to prove that this algorithm is effective.

II. UPGRADE STANDARD CONVOLUTION USING DEFORMABLE CONVOLUTION

In a computer vision task that inputs three-dimensional (channel, wide, high) images, the resulting feature map is also the same 3D tensor data as the input,

2473-3636/20/$31.00 ©2020 IEEE
DOI 10.1109/DCABES50732.2020.00069

and the deformable convolution allows the sample points to be offset in the two-dimensional spatial domain of the non-channel dimension, making the network to learn more free features. In an existing standard convolutional neural network that has been trained, the sampling point is not offset, so that the standard convolution can be conveniently upgraded to a zero offset deformable convolution in-place without destroying any original parameters.

A. *Deformable Convolution*

The standard 2D convolution is first sampled from the regular sample grid $\mathcal{R}$ on the input feature map. For a standard convolution operation with 3x3 kernel and dilation 1, the sample grid are as follows.

$$\mathcal{R} = \{(-1,-1),(-1,0),(-1,1),(0,-1),(0,0),(0,1),(1,-1),(1,0),(1,1)\}$$

And then calculate the sum of the value of the sample point p_n multiplied by the product of the weight w. $\mathrm{y}(p_c)$ represents the value on the output feature map y when the p_c is the sampling center.

$$\mathrm{y}(p_c) = \sum_{p_n \in \mathcal{R}} w(p_n) \cdot x(p_c + p_n + \triangle p_n)$$

Where $\triangle p_n$ represents the offset of the deformable convolution, which is the standard convolution when $\triangle p_n = 0$.

Fig. 2 shows the illustrion of 3×3 deformable convolution and 3×3 standard convolution.

B. *How to upgrade*

For a standard 2D convolution operation

$$f_{output} = \mathrm{w} \times f_{input} + \mathrm{bias}$$

For a deformable 2D convolution operation

$$\mathrm{offset} = w_{offset} \times f_{input} + b_{offset}$$

$$f_{output} = \mathrm{D}\left(f_{input}, w_{offset}, W, bias\right)$$

Where D is the convolution operation according to the offset in the deformable convolution.

In the deformable convolution, When $w_{offset} = 0, b_{offset} = 0$, since the product of any number and 0 is 0, $\mathrm{offset} = 0$ can be obtained, in other words, the offset value $\triangle p_n$ in the deformable convolution relative to the regular grid sampling point is 0.so we have

$$\mathrm{D}\left(f_{input}, offset, W, bias\right) = \mathrm{w} \times f_{input} + \mathrm{bias},$$

Under this condition

$$f_{output} = \mathrm{w} \times f_{input} + \mathrm{bias}.$$

Deformable convolution downgrade to standard convolution.

Although a trained deformable convolution network may not have $\mathrm{w}_{\mathrm{offset}} = 0, b_{offset} = 0$, and instead we can use it to upgrade the standard convolution to a deformable convolution. Fig. 3 shows the process of upgrade network with deformable convolution.

For a standard convolution whose convolution kernel size is $D_k \cdot D_k$, the number of input channels is M, and the number of output channels is N, the number of parameters is

$$D_K \cdot D_K \cdot M \cdot N + N .$$

Similarly, for a deformable convolution of the same parameter configuration, the number of parameters is

$$D_K \cdot D_K \cdot M \cdot N \cdot + M + 2 \cdot D_K \cdot D_K \cdot M \cdot D_K \cdot D_K .$$

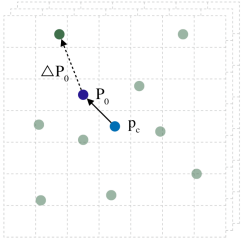

Figure 2. The blue dots represent the center of the sample p_c, the purple dots represent the sample offset p_n at standard convolution, and the green dots represent the additional offset $\triangle p_n$ at deformable convolution.

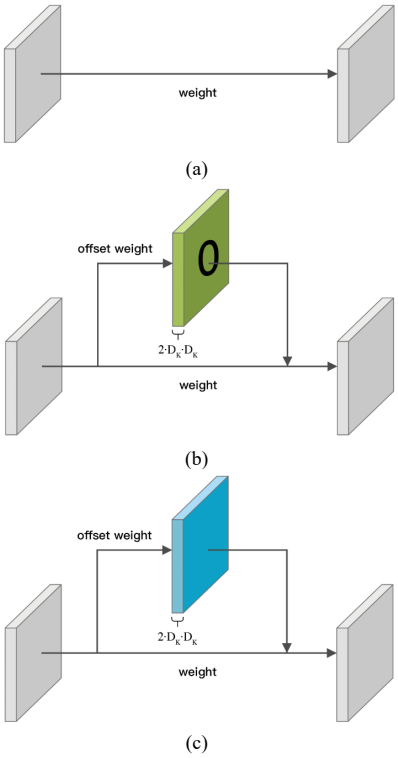

Figure 3. (a) is standard convolution module,(b) is the deformable convolution module after upgrade without training and the offset featuremap drawn green is zero tensor,(c) is the deformable convolution module after upgrade with training.

In the common backbone network Resnet-101, the convolution kernel size of stage4_conv8_weight is 3x3, the number of input channels is 512, the number of output channels is 512, and there is no offset. The number of parameters of standard 2D convolution is 2359296,the number of parameters for the deformable 2d convolution is 2442240 . which is only 3.5% higher than the standard 2D convolution parameter but due to the extraction with better learning ability for spatial transformation, the performance of the network is greatly improved.

III. EXPERIMENT

A. Dai's experiment

Dai tried to add the deformable convolution module to the convolutional network in different tasks to evaluate the deformation convolution effect. He used the pre-trained resnet-101 network on the ImageNet dataset as the backbone network, trying to replace the 3x3 standard convolutional layers of the last 1, 2, 3, and 6 with the deformable convolutional layer. The parameters of the new deformable convolution layer are reinitialized. The deformable convolution network is then evaluated on the PASCAL VOC [6] dataset for target detection and semantic segmentation tasks. Table. 1 below shows some of the experimental results.

TABLE I. THE RESULT OF DAI'S EXPERIMENT

usage of deformable convolution(#layers)	**DeepLab**	**Faster R-CNN**
	mIOU(%)	***mAP@0.5(%)***
none (0, baseline)	69.7	78.1
res5c (1)	73.9	78.6
res5b,c (2)	74.8	78.5
res5a,b,c (3, default)	**75.2**	78.6
res5 & res4b22,b21,b20 (6)	74.8	**78.7**

The experimental results from Dai show that the deformable convolution is significant for the improvement of the task effect. Focus on the metric, more deformable convolutional layers don't necessarily bring a better Result here, although it will improve in most cases. At the same time, more stacks of deformable convolution can also slow down the network, especially during training time. After comparing with the experimental results, replacing the last 3 convolutional layers is a good trade-off for all tasks.

But when Dai has a baseline model and wants to add the deformable convolutional layer to the baseline model, since the parameters of the baseline model cannot be used on the new model, it is still chosen to train a network completely from scratch on the basis of the pre-training model. It is obviously a waste to abandon the original baseline model that has been successfully trained.

B. Efficient upgrade

Initializing the deformable convolution using the standard convolution parameter is an in-place operation. The replaced new model inherits the parameters of the original model and has the same calculation results as the original model. After being replaced by a deformable convolution, the network's ability to extract geometric deformation features of the object is greatly increased. We can get a better result than the original network through a small number of iterative training. This approach is obviously less time and computational resources than retraining a same deformable convolutional network.

On the PASCAL VOC dataset, we use the VOC2012 trainval dataset and the VOC2007 trainval dataset as the trainset, use the VOC2007 test dataset as the valid set and use the SSD [8] model with Resnet50 as the feature extraction network for training. We used three experimental methods to train, and compared the mAP@0.5 and training loss for different training methods.

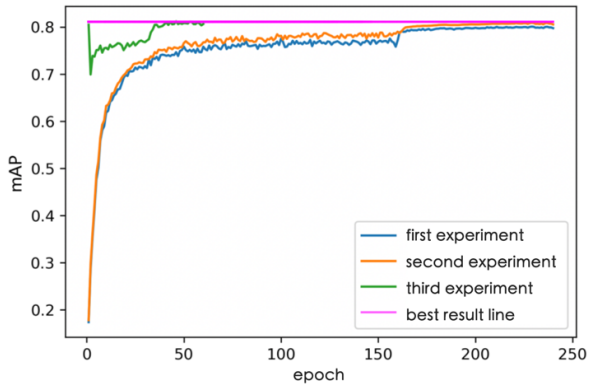

Figure 4. mAP@0.5 of three experimental results,the blue line is first experiment's result, the orange line is second experiment's result, the green line is third experiment's result,the magenta line is the best result.

In the first experiment, the first half of the network is the Resnet-50 feature extraction network pre-trained by ImageNet dataset, and the second half is the reinitialized SSD detection module. All convolutional layers are standard convolution.

In the second experiment, the first half of the network was ImageNet pre-trained Resnet-50 feature extraction network, but replaced the last three convolution layers of the feature extraction network with reinitialized deformable convolution layers, and the latter half remained Initialized SSD detection module.

In the third experiment, the network trained for the first time was used as the initial network, and the last three convolution layers of the feature derivation network were upgraded to deformable convolution layers, that is, the network structure of the network was the same as in the second experiment. Before the training, the same input yields the same results as the first test.

The training methods of the three experiments were all SGD algorithms with a moment of 0.8, and the initial learning rate was 0.001.Table. 2 shows different hyperparameter. Fig. 4 shows the result of three experiments.

TABLE II. THE DETIAL HYPERPARAMETER OF EXPERIMENT

Experiment number	1	2	3
epochs	240	240	60
Weight attenuation epoch	160,200	160,200	40,50
mAP@0.5	80.1	80.9	**81.0**

From the experimental results, we prove that using the upgraded network of the baseline model for training takes

much less time than the retrained network, and the effect of the model reaches the fully retrained network. For a network that we have successfully trained, upgrading a standard convolution module to a deformable convolution module is far more convenient and faster than rebuilding a deformable convolution network of the same structure.

C. Detection and Semantic segmentation

The final results were explored using deformable convolution on the task of detection on PASCAL VOC and COCO [7].

1) Detection on PASCAL VOC2012

Following the evaluation protocol of PASCAL VOC detection task, we use VOC 2012 trainval set, and VOC 2007 trainval to train or finetune, and test on VOC 2007 test set, the mean Average Precision (mAP) with IOU (Intersection over Union) threshold of 0.5 as the metric. For the SSD model, we used a base learning rate of 0.01 for a total of 240 epochs and decay the learning rate at 160 and 200 epochs with 0.1, batchsize is 32. For the post-processing stage of the network model, NMS(Non-Maximum Suppression) with a threshold of 0.45 is used. On the YOLO3 model, total of 200 epochs to train and decay the learning rate at 160 and 180 epochs with 0.1.

After get the base network model through training or download Mxnet reimplement, we upgraded the last three standard convolution modules of the feature extract network in the network to the deformable convolution. Then use the same dataset and metric as the base model to train and test. When finetune the new network, we only train a quarter of the previous epoch. And the learning rate is decayed by two-thirds and five-fifths of the previous total training epoch with 0.1. When the epoch is not an integer, round off it.

All upgraded networks show better performance than the base network, Table. 3 shows the results of the SSD and yolov3[9] models for the different base network trainers.

TABLE III. THE RESULT ON VOC

model	input size	backbone	upgrade	base model trainer	mAP @0.5
SSD	512	Resnet50	F	Gluon	80.1
			T		**80.9**
SSD	512	Resnet50	F	Ours	79.9
			T		**80.8**
yolo3	320	Darknet53	F	Gluon	79.3
			T		**80.0**
yolo3	320	Darknet53	F	Ours	79.2
			T		**80.1**
yolo3	416	Darknet53	F	Gluon	81.5
			T		**82.0**
yolo3	416	Darknet53	F	Ours	81.6
			T		**82.0**

2) Detection on COCO

Following the evaluation protocol of COCO detection task, we use COCO 2017 train set to train or finetune, and test on COCO 2017 valid set, the mAP with IOU threshold of 0.5:0.95, 0.5, 0.75 as the metric. For the SSD model. The training process runs on four V100compute nodes, the others is same as did on PASCAL VOC. On the YOLO3 model, total of 280 epochs to train and decay the learning rate at 220 and 250 epochs with 0.1.

After get the base network model through training or download Mxnet reimplement, we deal with it as before.

All upgraded networks show better performance than the base network, Table. 4 shows the results of the SSD and yolov3 models for the different base network trainers.

TABLE IV. THE RESULT ON COCO

model	input size	back bone	upg-rade	base model trainer	mAP @0.5 :0.95	mAP @0.5	mAP @0.75
SSD	512	Resnet50	F	Gluon	30.6	50.0	32.2
			T		30.8	50.4	**32.5**
SSD	512	Resnet50	F	Ours	30.5	50.0	32.1
			T		30.9	50.5	**32.6**
yolo3	320	Darknet53	F	Gluon	33.6	54.1	35.8
			T		34.0	54.9	**36.2**
yolo3	320	Darknet53	F	Ours	33.5	54.1	35.9
			T		33.8	54.2	**36.2**
yolo3	416	Darknet53	F	Gluon	36.0	57.2	38.7
			T		36.9	58.2	**39.1**
yolo3	416	Darknet53	F	Ours	36.0	57.1	38.8
			T		36.9	58.3	**39.1**
yolo3	608	Darknet53	F	Gluon	37.0	58.2	40.1
			T		37.6	59.1	**41.0**

IV. CONCLUSION

This paper presents a new method that can upgrade the network in-place using deformable convolution module. And we designed a series of experiments to prove that this method is feasible and effective in detection tasks.

[1] Zhang, Hongyi et al. "mixup: Beyond Empirical Risk Minimization." *ArXiv*abs/1710.09412 (2017): n. pag.

[2] Kingma, Diederik P. and Jimmy Ba. "Adam: A Method for Stochastic Optimization." *CoRR* abs/1412.6980 (2014): n. pag.

[3] He, Kaiming et al. "Deep Residual Learning for Image Recognition." *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016): 770-778.

[4] Dai, Jifeng et al. "Deformable Convolutional Networks." *2017 IEEE International Conference on Computer Vision (ICCV)* (2017): 764-773.

[5] Yu, Fisher and Vladlen Koltun. "Multi-Scale Context Aggregation by Dilated Convolutions." *CoRR* abs/1511.07122 (2015): n. pag.

[6] Everingham, Mark et al. "The Pascal Visual Object Classes (VOC) Challenge." *International Journal of Computer Vision* 88 (2009): 303-338.

[7] Lin, Tsung-Yi et al. "Microsoft COCO: Common Objects in Context." *ECCV* (2014).

[8] Liu, Wei et al. "SSD: Single Shot MultiBox Detector." *ECCV* (2016).

[9] Redmon, Joseph and Ali Farhadi. "YOLOv3: An Incremental Improvement." *ArXiv*abs/1804.02767 (2018): n. pag.