InterActive: Inter-layer Activeness Propagation

Lingxi Xie[1*], Liang Zheng[2*], Jingdong Wang[3], Alan Yuille[1] and Qi Tian[2]

[1]Department of Statistics, University of California, Los Angeles, California, USA

[2]Department of Computer Science, University of Texas at San Antonio, Texas, USA

[3]Microsoft Research, Beijing, China

ABSTRACT

An increasing number of computer vision tasks can be tackled with deep features, which are the intermediate outputs of a pre-trained Convolutional Neural Network. Despite the astonishing performance, deep features extracted from low-level neurons are still below satisfaction, arguably because they cannot access the spatial context contained in the higher layers. In this paper, we present InterActive, a novel algorithm which computes the activeness of neurons and network connections. Activeness is propagated through a neuron in a top-down manner, carrying highlevel context and improving the descriptive power of lowlevel and mid-level neurons. Visualization indicates that neuron activeness can be interpreted as spatial-weighted neuron responses. We achieve state-of-the-art classification performance on a wide range of image datasets.

CONTRIBUTION

In this paper, we present a novel algorithm named **InterActive**, which is an efficient way of deep feature extraction from a single image.

First of all, we propose *a new baseline* for deep feature extraction. We do not simply resize the input image into the same fixed size (*e.g.*, 224×224 in VGGNet), but resize it into a larger scale (the area is approximately 512^2) while maximally preserving its aspect ratio. Such a simple improvement significantly boosts the classification accuracy with deep features.

In this method, the receptive field of a neuron is relatively small (*e.g.*, in the 19-layer VGGNet, a *pool-5* neuron can see 268^2 pixels) compared to the input image size (approximately 512^2). We argue that two problems occur under this setting, *i.e.*, the *big* problem and the *small* problem, illustrated in the figure to the right.

The main reason of these problems is that conventional deep feature extraction methods involve only forward-propagating visual signals through the network. The useful information contained in the remaining part of the network is discarded. We suggest that back-propagation is needed, which uses high-level visual clues to help mid-level feature extraction.

The proposed **InterActive** algorithm is based on such a motivation. A back-propagation process is performed in deep feature extraction, which uses a *score function* as the loss function. The gradient with respect to network weights can be considered as the *activeness* of each neuron connection, and collecting them obtains the *activeness* or the *weight* of each neuron.

THE PROPOSED ALGORITHM

Improved deep feature extraction

Input: an image with $W \times H$ pixels.
Reference model: a pre-trained VGGNet.
Original deep feature extraction: resizing the image into 224×224, passing it through the network, and extracting the intermediate response on the l-th layer.
Improved deep feature extraction: resizing the image so that:

- The aspect ratio is maximally preserved;
- The area (number of pixels) $\approx 512^2$;
- The width and height are multipliers of 32 (the down-sampling rate of VGGNet)

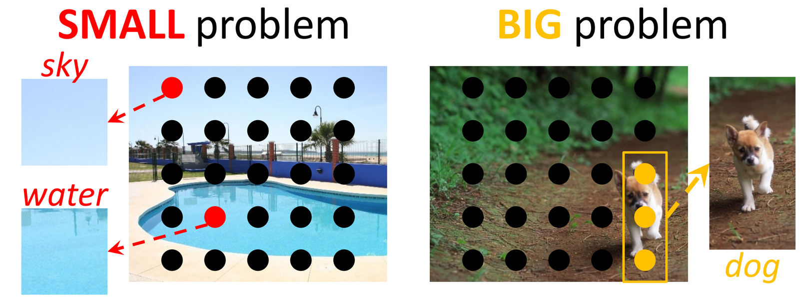
passing the resized image through the network, extracting the intermediate response on the l-th layer, and averaging over all the spatial positions.

Significant accuracy gain

	pool-5		*fc-6*	
	ORIG	IMPR	ORIG	IMPR
Caltech256	77.46	81.40	80.41	83.51
SUN-397	48.19	55.22	53.06	61.30
Flower-102	86.87	94.70	84.89	93.54

Two Problems

SMALL problem　　**BIG** problem



Main reason: conventional deep feature extraction only involves forward-propagation, useful information in high-level layers is discarded.
Solution: introducing a back-propagation process into deep feature extraction!

Inter-layer activeness propagation

Key idea: unsupervised back-propagation in the deep feature extraction process.

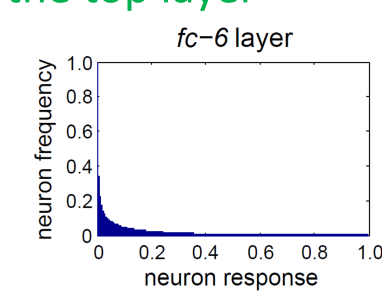
- Defining a *score function* at the top level;
- Back-propagating gradients to obtain the *activeness of neuron connections*;
- Collecting the activeness of neuron connections as the *activeness of neurons*.

Mathematical notations

A pre-trained CNN: $\mathbf{h}(\mathbf{X}^{(0)};\boldsymbol{\theta})$. $\mathbf{X}^{(0)}$ is the input, and $\boldsymbol{\theta}$ is the weights. $\mathbf{X}^{(t)}$ is the neuron responses on the t-th layer, a $W_t \times H_t \times D_t$ cube. $\mathbf{x}^{(t)}$ (D_t-dimensional) is the average over all spatial positions of $\mathbf{X}^{(t)}$.

Feature distribution on the top layer

We study the PDF of neuron responses on the T-th layer, i.e., the starting point of backprop. We assume the following PDF:



$f(\mathbf{x}^{(T)}) = C_p \times \exp\left\{-\left\|\mathbf{x}^{(T)}\right\|_p^p\right\}$, where p is the norm (1 or 2) and C_p is the normalization coefficient.

Score function and activeness of connections

The *activeness of neuron connection* is computed using the *score function*, *i.e.*, the gradient of log-distribution over $\boldsymbol{\theta}^{(t)}$: $\frac{\partial \ln f^{(T)}}{\partial \boldsymbol{\theta}^{(t)}} = \frac{\partial \ln f^{(T)}}{\partial \mathbf{X}^{(t+1)}} \times \frac{\partial \mathbf{X}^{(t+1)}}{\partial \boldsymbol{\theta}^{(t)}}$. Here, $\frac{\partial \ln f^{(T)}}{\partial \mathbf{X}^{(t+1)}}$ is the *layer score*, and $\frac{\partial \mathbf{X}^{(t+1)}}{\partial \boldsymbol{\theta}^{(t)}}$ is the *inter-layer activeness*.

Activeness (importance, weights) of neurons

Each neuron collects the activeness from the related connections as its activeness, $\tilde{x}_{w,h,d}^{(t)}$. $\tilde{x}_{w,h,d}^{(t)}$ is the improved deep feature, which can be **explicitly** written as $\tilde{x}_{w,h,d}^{(t)} = x_{w,h,d}^{(t)} \times \gamma_{w,h,d}^{(t)}$, where $\gamma_{w,h,d}^{(t)}$ is the weight of $x_{w,h,d}^{(t)}$ obtained with InterActive. We visualize these weights using a 2D heatmap: $\hat{\gamma}_{w,h}^{(t)} = \sum_d \gamma_{w,h,d}^{(t)}$.

Visualization results are shown in the figure to the right.

Visualization of neuron activeness

The ***last*** configuration: $T = L - 1$ (L is the total number of layers), the t-th layer receives the information from the highest level;
The ***next*** configuration: $T = t + 1$, the t-th layer receives the information from the next layer.

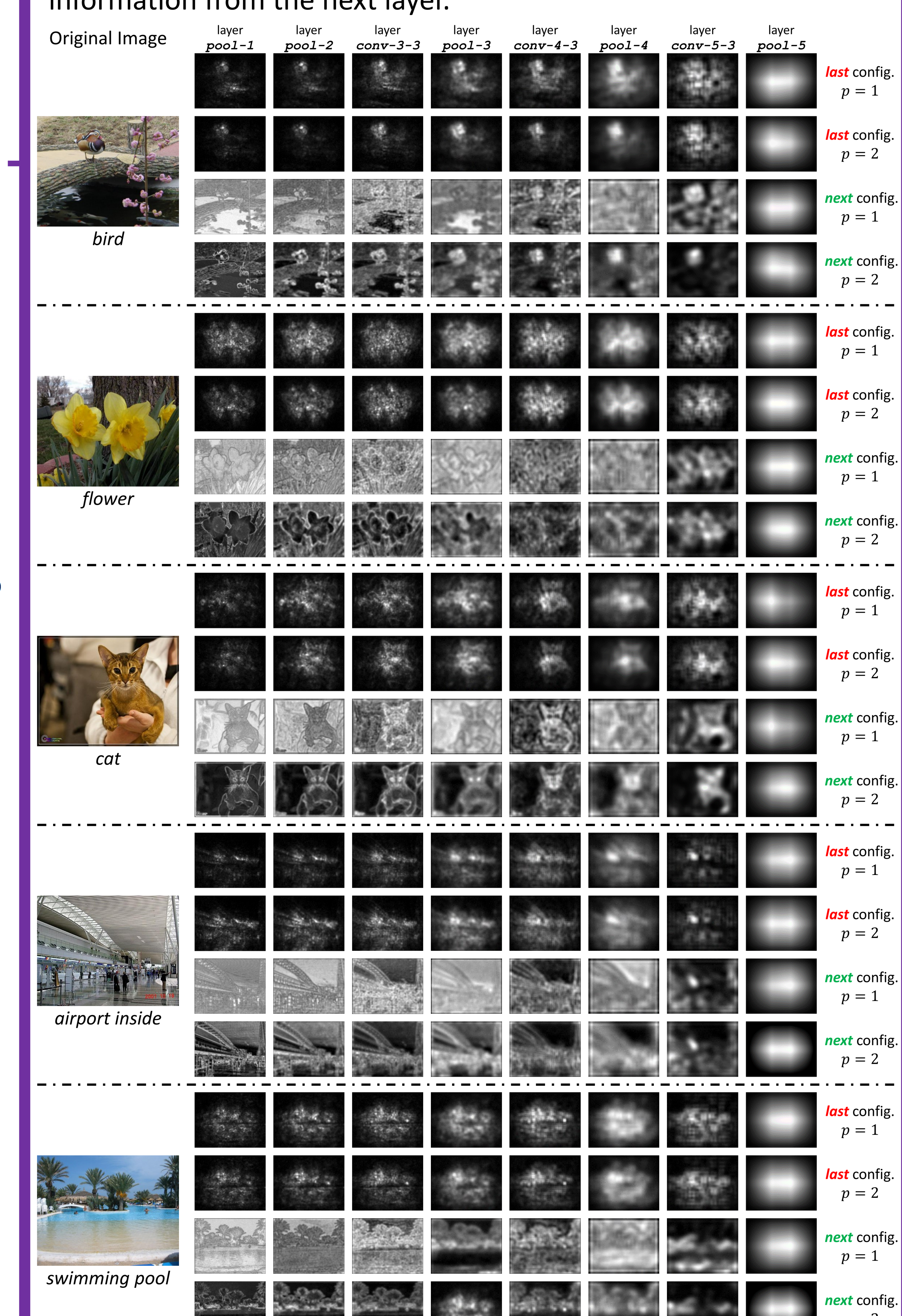


Image classification results with *low-level* (left), *mid-level* (middle) and *high-level* (right) deep features

Layer	Model	Dims	C-256	I-67	S-397	P-37	F-102	B-200
conv-3-3	ORIG, AVG-p	256	26.44	36.42	22.73	27.78	49.70	10.47
conv-3-3	ORIG, MAX-p	256	24.18	33.27	19.71	31.43	48.02	13.85
conv-3-3	***next***, $p=1$	256	27.29	36.97	22.84	28.89	50.62	10.93
conv-3-3	***next***, $p=2$	256	27.62	37.36	23.41	30.38	54.06	12.73
conv-3-3	***last***, $p=1$	256	34.50	39.40	25.84	49.41	60.53	24.21
conv-3-3	***last***, $p=2$	256	**35.29**	**39.68**	**26.02**	**50.57**	**61.06**	**25.27**
pool-3	ORIG, AVG-p	256	29.17	37.98	23.59	29.88	52.44	11.00
pool-3	ORIG, MAX-p	256	26.53	34.65	20.83	33.68	50.93	13.66
pool-3	***next***, $p=1$	256	29.09	38.12	24.05	30.08	52.26	10.89
pool-3	***next***, $p=2$	256	29.55	38.61	24.31	31.98	55.06	12.65
pool-3	***last***, $p=1$	256	36.96	41.02	26.73	50.91	62.41	24.58
pool-3	***last***, $p=2$	256	**37.40**	**41.45**	**27.22**	**51.96**	**63.06**	**25.47**

Layer	Model	Dims	C-256	I-67	S-397	P-37	F-102	B-200
conv-4-3	ORIG, AVG-p	512	49.62	59.66	42.03	55.57	76.98	21.45
conv-4-3	ORIG, MAX-p	512	47.73	55.83	40.10	59.40	75.72	23.39
conv-4-3	***next***, $p=1$	512	51.83	60.37	43.59	59.29	78.54	25.01
conv-4-3	***next***, $p=2$	512	53.52	60.65	44.17	63.40	80.48	31.07
conv-4-3	***last***, $p=1$	512	61.62	62.45	45.43	75.29	85.91	52.26
conv-4-3	***last***, $p=2$	512	**61.98**	**62.74**	**45.87**	**77.61**	**86.08**	**54.12**
pool-4	ORIG, AVG-p	512	60.39	66.49	49.73	66.76	85.56	28.56
pool-4	ORIG, MAX-p	512	57.92	62.96	47.29	69.23	84.39	30.01
pool-4	***next***, $p=1$	512	60.59	66.48	49.55	66.28	85.68	28.40
pool-4	***next***, $p=2$	512	62.06	66.94	50.10	72.40	87.36	37.49
pool-4	***last***, $p=1$	512	68.20	67.20	51.04	81.04	91.22	57.41
pool-4	***last***, $p=2$	512	**68.60**	**67.40**	**51.30**	**82.56**	**92.00**	**59.25**

Layer	Model	Dims	C-256	I-67	S-397	P-37	F-102	B-200
conv-5-3	ORIG, AVG-p	512	77.40	74.66	59.47	88.36	94.03	55.44
conv-5-3	ORIG, MAX-p	512	75.93	71.38	57.03	87.10	91.30	55.19
conv-5-3	***next***, $p=1$	512	80.31	**74.80**	59.63	90.29	94.84	67.64
conv-5-3	***next***, $p=2$	512	80.73	74.52	**59.74**	**91.56**	95.16	**73.14**
conv-5-3	***last***, $p=1$	512	80.77	73.68	59.10	90.73	95.40	69.32
conv-5-3	***last***, $p=2$	512	**80.84**	73.58	58.96	91.19	**95.70**	69.75
pool-5	ORIG, AVG-p	512	81.40	**74.93**	**55.22**	91.78	94.70	69.72
pool-5	ORIG, MAX-p	512	79.61	71.88	54.04	89.43	90.01	68.52
pool-5	***next***, $p=1$	512	81.50	72.70	53.83	92.01	95.41	71.96
pool-5	***next***, $p=2$	512	81.58	72.63	53.57	**92.30**	95.40	**73.21**
pool-5	***last***, $p=1$	512	81.60	72.58	53.93	92.20	**95.43**	72.47
pool-5	***last***, $p=2$	512	**81.68**	72.68	53.79	92.18	95.41	72.51

RESULTS

Results on some *small* datasets

We use the concatenated deep features from the *pool-1* layer through *fc-6* (9 layers in total)

Model	C-256	I-67	S-397	P-37	F-102	B-200
Murray *et.al.*, CVPR'14	–	–	–	56.8	84.6	33.3
Kobayashi *et.al.*, CVPR'15	58.3	64.8	–	–	–	30.0
Xie *et.al.*, ICCV'15	60.25	64.93	50.12	63.49	86.45	50.81
Ravazian *et.al.*, CVPR'14	–	69.0	–	–	86.8	61.8
Qian *et.al.*, CVPR'15	–	–	–	81.18	89.45	67.86
Xie *et.al.*, ICML'15	–	70.13	54.87	90.03	86.82	62.02
ORIG, AVG-pooling	84.02	78.02	62.30	93.02	95.70	73.35
ORIG, MAX-pooling	84.38	77.32	61.87	93.20	95.98	74.76
next, $p=1$	84.43	78.01	62.26	92.91	96.02	74.37
next, $p=2$	84.64	78.23	62.50	93.22	96.26	74.61
last, $p=1$	84.94	78.40	62.69	93.40	96.35	75.47
last, $p=2$	**85.06**	**78.65**	**62.97**	**93.45**	**96.40**	**75.62**

Please refer to our paper for more results

Results on ImageNet

We use **InterActive** to update the *fc-6* neurons

Model	top-1 error	top-5 error
VGGNet-16, *original*	24.2%	7.1%
VGGNet-16, *InterActive*	23.8%	6.9%
VGGNet-19, *original*	24.0%	7.0%
VGGNet-19, *InterActive*	23.5%	6.7%
VGGNet-combined, *original*	23.6%	6.7%
VGGNet-combined, *InterActive*	23.2%	6.5%

REFERENCES

Key references are numbered as they appear in the paper.

[8] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-scale Hierarchical Image Database. CVPR, 2009.

[23] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. NIPS, 2012.

[40] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. ICLR, 2015.

[42] A. Vedaldi and K. Lenc. MatConvNet-Convolutional Neural Networks for MATLAB. ACM-MM, 2015.

[48] L. Xie, R. Hong, B. Zhang, and Q. Tian. Image Classification and Retrieval are ONE. ICMR, 2015.

[54] M. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. ECCV, 2014.

ACKNOWLEDGEMENTS

This work was partly done when Lingxi Xie and Liang Zheng were interns at MSR. They contributed equally. This work was supported by NIH Grant 5R01EY022247-03, ONR N00014-12-1-0883 and ARO grant W911NF-15-1-0290, Faculty Research Gift Awards by NEC Labs of America and Blippar, and NSFC 61429201. We thank John Flynn, Xuan Dong, Jianyu Wang, Junhua Mao and Zhuotun Zhu for instructive discussions.