Geometric Neural Phrase Pooling: Modeling the Spatial Co-occurrence of Neurons

Lingxi Xie[1], Qi Tian[2], John Flynn[3], Jingdong Wang[4] and Alan Yuille[1]

[1]The Johns Hopkins University [2]University of Texas at San Antonio

[3]University of California, Los Angeles [4]Microsoft Research

ABSTRACT

Deep Convolutional Neural Networks (CNNs) are playing important roles in state-of-the-art visual recognition. This paper focuses on modeling the spatial co-occurrence of neuron responses, which is less studied in the previous work. For this, we consider the neurons in the hidden layer as neural words, and construct a set of geometric neural phrases on top of them. The idea that grouping neural words into neural phrases is borrowed from the Bag-of-Visual-Words (BoVW) model. Next, the Geometric Neural Phrase Pooling (GNPP) algorithm is proposed to efficiently encode these neural phrases. GNPP acts as a new type of hidden layer, which punishes the isolated neuron responses after convolution, and can be inserted into a CNN model with little extra computational overhead. Experimental results show that GNPP produces significant and consistent accuracy gain in image classification.

CONTRIBUTION

In this paper, we present **Geometric Neural Phrase Pooling (GNPP)**, an efficient yet effective algorithm to help CNN training.

GNPP is motivated by one of our previous work, namely Geometric Phrase Pooling (GPP) in the Bag-of-Visual-Words (BoVW) model. GPP works by considering local feature co-occurrence and performing non-linear smoothing. We find that these operations also help network training.

GNPP averages the response of a neuron with the maximal response among its neighboring neurons. This is to penalize the isolated neural responses. We argue that, especially in the high-level layers, isolated responses often correspond to unexpected noise in convolution, because neighboring neurons often share a large part of their receptive fields. Experiments reveal that GNPP often works better on high-level layers, which verifies our motivation.

GNPP can be considered as an intermediate layer in a CNN structure. Since GNPP does not change the geometric shape of data, it can be inserted, at least in theory, to any position in the neural network. In practice, we only insert it between a convolutional layer and a pooling layer because of our motivation.

GNPP produces **consistent accuracy gain** on several state-of-the-art deep networks for object recognition. Moreover, the extra time (+1.29%) and memory (+2.52%) costs of GNPP are almost negligible. Recently, we also observe that GNPP works well in other CNN-based models such as Faster RCNN or DeepLab.

THE PROPOSED ALGORITHM

Neural Word and Geometric Neural Phrase

- Defined on a hidden layer $\mathbf{X}^{(l)}$ of the CNN. For simplicity, denote $\mathbf{X}^{(l)}$ as $\mathbf{X}$.
  - ✓ $\mathbf{X}$ is a 3D cube with $W \times H \times D$ neurons
- We naturally consider the data as a set of D-dimensional **neural words**:

$$\mathcal{X} = \{\mathbf{x}_{w,h}\}_{w=1,h=1}^{W,H}$$

- A **geometric neural phrase** is a group of neighboring neurons:

$$\mathcal{G}_{w,h} = \left\{\mathbf{x}_{w,h}^{(k)}\right\}_{k=0}^{K}$$

  - ✓ $\mathbf{x}_{w,h}^{(0)} = \mathbf{x}_{w,h}$: the *central word*
  - ✓ $\mathbf{x}_{w,h}^{(k)}$ $(k > 0)$: the *side words*, located in a small neighborhood of $\mathbf{x}_{w,h}$
- **For each neural word, there defines a neural phrase.**

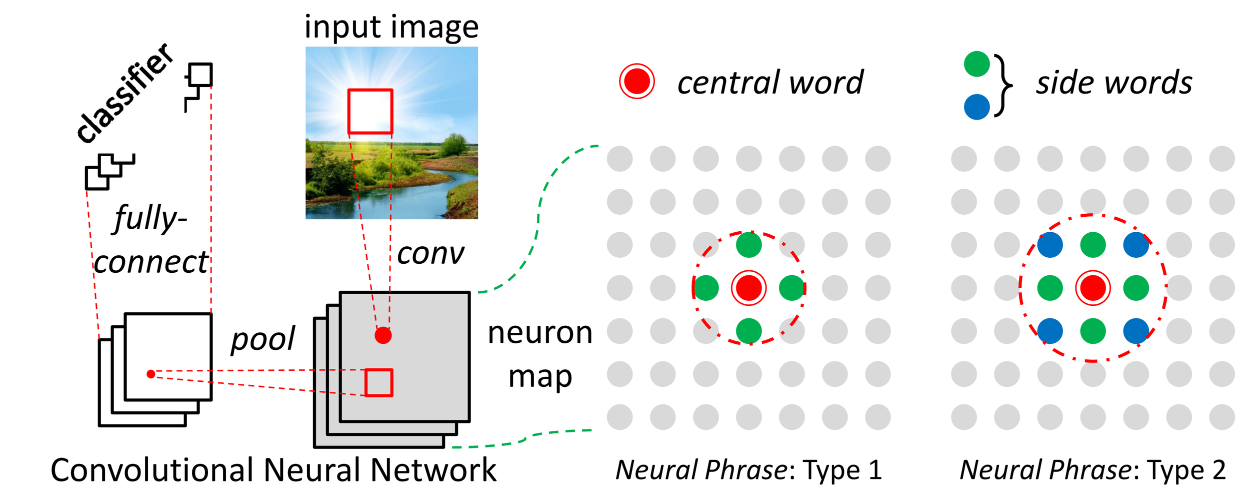
Geometric Neural Phrase Pooling

- A D-dimensional vector for each geometric neural phrase individually

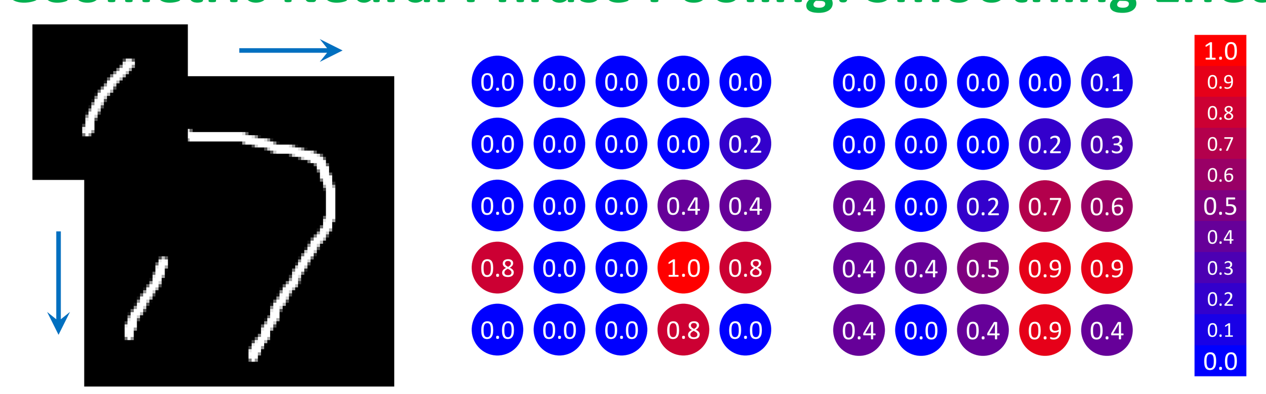
$$\mathbf{z}_{w,h} = \frac{1}{2}\left[\mathbf{x}_{w,h} + \max_{k>0}\left\{s_{w,h}^{(k)} \times \mathbf{x}_{w,h}^{(k)}\right\}\right]$$

  - ✓ $\max_{k>0}\{\ \}$: dimension-wise maximization
  - ✓ $s_{w,h}^{(k)} = \sigma$ or $s_{w,h}^{(k)} = \sigma^2$, according to the relative position
  - ✓ $\sigma \in]0,1]$: the smoothing parameter

Illustration of Geometric Neural Phrase

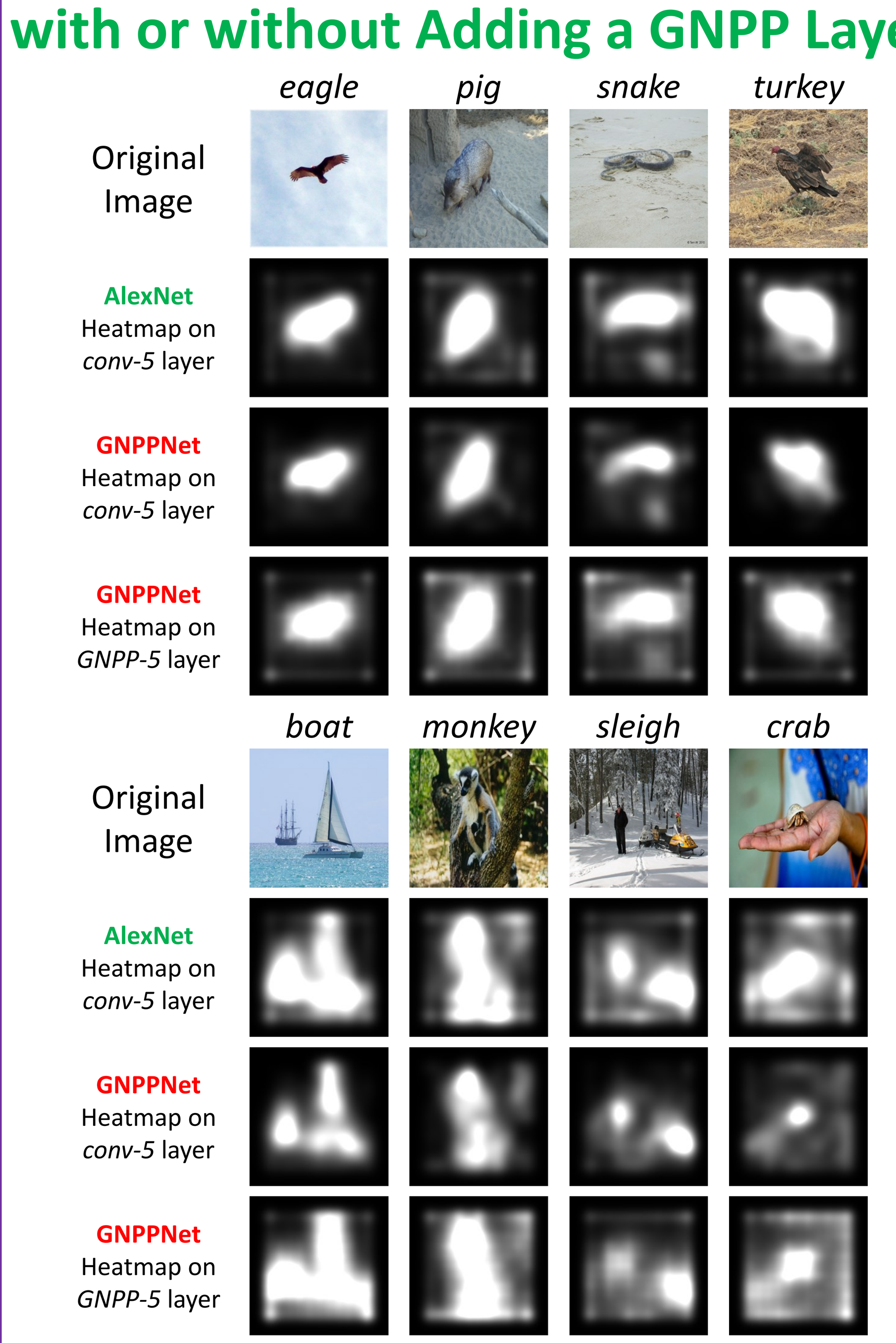


Geometric Neural Phrase Pooling: Smoothing Effect



GNPP penalizes isolated neural responses, and preserves the clustered responses. In neural networks, especially in high-level layers, the isolated responses often relate to random noise, therefore GNPP works well in these layers.

Visualization on Neural Responses with or without Adding a GNPP Layer



LeNet Results on MNIST (UL), SVHN (UR), CIFAR10 (LL) and CIFAR100 (LR)

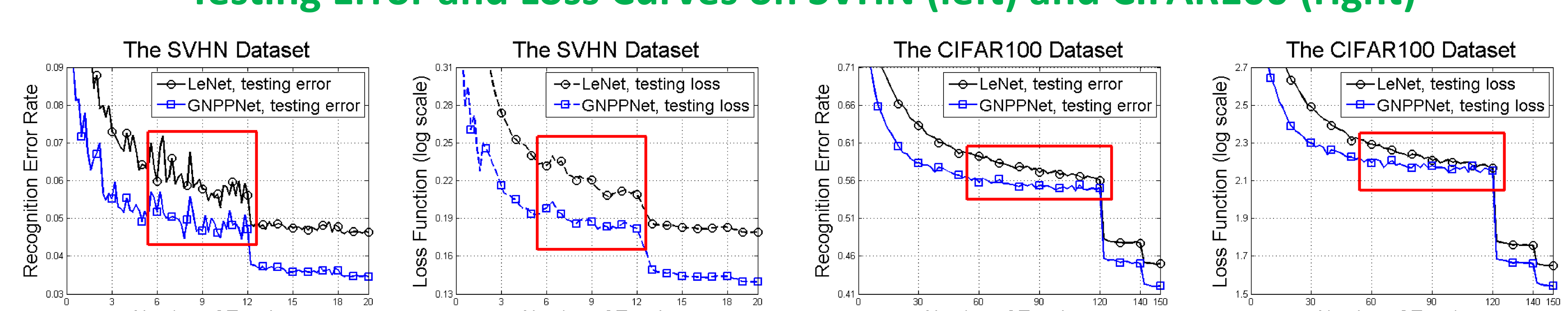
L1	L2	D	T1 (1.0)	T1 (0.9)	T1 (0.8)	T2 (1.0)	T2 (0.9)	T2 (0.8)
			$0.87 \pm .02$	$0.87 \pm .02$	$0.87 \pm .02$	$0.87 \pm .02$	$0.87 \pm .02$	$0.87 \pm .02$
√			$0.72 \pm .04$	$0.73 \pm .03$	$0.70 \pm .05$	$0.71 \pm .06$	$0.71 \pm .06$	$0.72 \pm .04$
	√		$0.75 \pm .03$	$0.79 \pm .02$	$0.77 \pm .05$	$0.73 \pm .04$	$0.75 \pm .04$	$0.73 \pm .05$
√	√		$\mathbf{0.72} \pm .03$	$\mathbf{0.67} \pm .04$	$\mathbf{0.69} \pm .04$	$\mathbf{0.63} \pm .03$	$\mathbf{0.64} \pm .03$	$\mathbf{0.67} \pm .03$
		√	$0.72 \pm .03$	$0.72 \pm .03$	$0.72 \pm .03$	$0.72 \pm .03$	$0.72 \pm .03$	$0.72 \pm .03$
√		√	$0.59 \pm .02$	$0.61 \pm .05$	$0.62 \pm .03$	$0.59 \pm .03$	$0.59 \pm .02$	$0.63 \pm .03$
	√	√	$0.63 \pm .03$	$0.62 \pm .07$	$0.64 \pm .03$	$0.62 \pm .05$	$0.60 \pm .03$	$0.65 \pm .03$
√	√	√	$\mathbf{0.58} \pm .05$	$\mathbf{0.55} \pm .05$	$\mathbf{0.57} \pm .02$	$\mathbf{0.54} \pm .05$	$\mathbf{0.56} \pm .04$	$\mathbf{0.61} \pm .05$

L1	L2	L3	T1 (1.0)	T1 (0.9)	T1 (0.8)	T2 (1.0)	T2 (0.9)	T2 (0.8)
			$4.63 \pm .06$	$4.63 \pm .06$	$4.63 \pm .06$	$4.63 \pm .06$	$4.63 \pm .06$	$4.63 \pm .06$
√			$4.46 \pm .06$	$4.47 \pm .05$	$4.42 \pm .09$	$4.42 \pm .08$	$4.42 \pm .07$	$4.43 \pm .09$
	√		$4.15 \pm .08$	$4.18 \pm .01$	$4.17 \pm .07$	$4.08 \pm .10$	$4.19 \pm .07$	$4.20 \pm .05$
√	√		$3.76 \pm .03$	$3.72 \pm .05$	$3.77 \pm .06$	$3.53 \pm .07$	$3.64 \pm .07$	$3.65 \pm .10$
		√	$4.10 \pm .05$	$4.07 \pm .03$	$4.10 \pm .05$	$4.10 \pm .07$	$4.10 \pm .03$	$4.14 \pm .07$
√		√	$3.55 \pm .10$	$3.60 \pm .03$	$3.67 \pm .06$	$3.47 \pm .05$	$3.47 \pm .02$	$3.55 \pm .09$
	√	√	$\mathbf{3.43} \pm .06$	$3.52 \pm .07$	$\mathbf{3.55} \pm .04$	$\mathbf{3.41} \pm .03$	$3.42 \pm .04$	$3.51 \pm .05$
√	√	√	$3.46 \pm .07$	$\mathbf{3.47} \pm .06$	$3.55 \pm .06$	$3.43 \pm .05$	$\mathbf{3.39} \pm .01$	$\mathbf{3.46} \pm .03$

L1	L2	L3	T1 (1.0)	T1 (0.9)	T1 (0.8)	T2 (1.0)	T2 (0.9)	T2 (0.8)
			$17.07 \pm .15$	$17.07 \pm .15$	$17.07 \pm .15$	$17.07 \pm .15$	$17.07 \pm .15$	$17.07 \pm .15$
√			$16.67 \pm .22$	$16.80 \pm .25$	$16.84 \pm .12$	$16.65 \pm .19$	$17.03 \pm .15$	$17.04 \pm .17$
	√		$15.79 \pm .22$	$16.09 \pm .17$	$15.95 \pm .31$	$15.69 \pm .11$	$16.07 \pm .27$	$15.90 \pm .09$
√	√		$15.49 \pm .15$	$15.31 \pm .20$	$15.51 \pm .25$	$15.27 \pm .10$	$15.29 \pm .14$	$15.28 \pm .16$
		√	$15.82 \pm .23$	$15.76 \pm .18$	$15.98 \pm .14$	$16.05 \pm .29$	$15.90 \pm .25$	$15.94 \pm .09$
√		√	$15.15 \pm .20$	$15.29 \pm .12$	$15.44 \pm .19$	$15.29 \pm .32$	$15.19 \pm .35$	$15.20 \pm .35$
	√	√	$\mathbf{14.92} \pm .18$	$15.00 \pm .18$	$15.15 \pm .15$	$\mathbf{14.83} \pm .25$	$14.93 \pm .20$	$14.92 \pm .16$
√	√	√	$14.97 \pm .17$	$\mathbf{14.83} \pm .23$	$\mathbf{14.78} \pm .17$	$15.22 \pm .16$	$\mathbf{14.79} \pm .26$	$\mathbf{14.85} \pm .26$

L1	L2	L3	T1 (1.0)	T1 (0.9)	T1 (0.8)	T2 (1.0)	T2 (0.9)	T2 (0.8)
			$44.99 \pm .19$	$44.99 \pm .19$	$44.99 \pm .19$	$44.99 \pm .19$	$44.99 \pm .19$	$44.99 \pm .19$
√			$44.62 \pm .17$	$44.53 \pm .45$	$44.78 \pm .06$	$44.43 \pm .29$	$44.58 \pm .36$	$44.58 \pm .52$
	√		$43.34 \pm .23$	$43.71 \pm .19$	$43.37 \pm .26$	$43.21 \pm .23$	$43.03 \pm .27$	$43.37 \pm .30$
√	√		$43.11 \pm .24$	$42.77 \pm .37$	$42.99 \pm .24$	$42.96 \pm .32$	$42.81 \pm .38$	$43.08 \pm .39$
		√	$43.99 \pm .07$	$43.63 \pm .11$	$43.50 \pm .26$	$43.38 \pm .37$	$43.34 \pm .27$	$43.46 \pm .25$
√		√	$42.85 \pm .38$	$42.81 \pm .27$	$42.82 \pm .29$	$43.08 \pm .27$	$42.79 \pm .34$	$42.93 \pm .22$
	√	√	$\mathbf{42.35} \pm .30$	$\mathbf{42.34} \pm .31$	$\mathbf{42.04} \pm .20$	$\mathbf{42.92} \pm .33$	$\mathbf{42.72} \pm .25$	$\mathbf{42.54} \pm .29$
√	√	√	$42.97 \pm .29$	$42.77 \pm .36$	$42.36 \pm .18$	$43.31 \pm .34$	$42.85 \pm .18$	$42.60 \pm .36$

Testing Error and Loss Curves on SVHN (left) and CIFAR100 (right)



RESULTS

Results on some *small* datasets

- We use BigNet [49] and Wide ResNet [50] as baselines. GNPP is inserted before the last pooling layer of each network.

	MNIST	SVHN	CIF10	CIF100
Zeiler [42]	0.45	–	2.80	–
Goodfellow [5]	0.47	–	2.47	–
Lin [21]	0.47	–	2.35	–
Wan [38]	0.52	0.21	–	1.94
Lee [21]	0.39	–	1.92	–
Liang [20]	0.31	–	1.77	–
BigNet, without GNPP	0.86	0.48	3.93	3.48
BigNet, with GNPP	0.68	0.43	3.65	3.25
WRN, without GNPP	0.66	0.45	3.69	3.27
WRN, with GNPP	0.63	0.41	3.61	3.21

Results on ImageNet

- We use AlexNet [2] as our baseline. GNPP is inserted before the last pooling layer.

ILSVRC2012	top-1	top-5
AlexNet, w/o GNPP	43.19	19.87
AlexNet, w/ GNPP	**42.16**	**19.24**

- GNPP builds latent neural connections. With GNPP, the equivalent # of connections between *conv-4* and *conv-5* increases from 149.5M to 348.9M. An alternative way is to increase the # of convolutional kernels, *e.g.*, using 512 kernels at *conv-5* increases the number to 299.0M.

ILSVRC2012	top-1	top-5	Time	Memory
AlexNet, w/ more kernels	42.45	19.47	+9.97%	+5.58%
AlexNet, w/ GNPP	**42.16**	**19.24**	**+1.29%**	**+2.52%**

GNPP is more effective and more efficient.

REFERENCES

Key references are numbered as they appear in the paper.

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

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

[26] L. Xie, Q. Tian, M. Wang and B. Zhang. Spatial Pooling of Heterogeneous Features for Image Classification. TIP, 2014.

[49] Nagadomi. The Kaggle CIFAR10 Network. https://github.com/nagadomi/kaggle-cifar10-torch7/. 2014.

[50] S. Zagoruyko, N. Komodakis. Wide Residual Networks. arXiv preprint, arXiv: 1605.07146, 2016.

ACKNOWLEDGEMENTS

This paper is supported by iARPA MICrONS contract D16PC00007, ONR N00014-12-1-0883, ARO grants W911NF-15-1-0290, Faculty Research Gift Awards by NEC Labs of America and Blippar, and NSFC 61429201. We thank Junhua Mao, Cihang Xie and Zhuotun Zhu for discussion.