From linear to bilinear, and beyond

Shu Kong

CS, ICS, UCI

2016-1-20

Outline

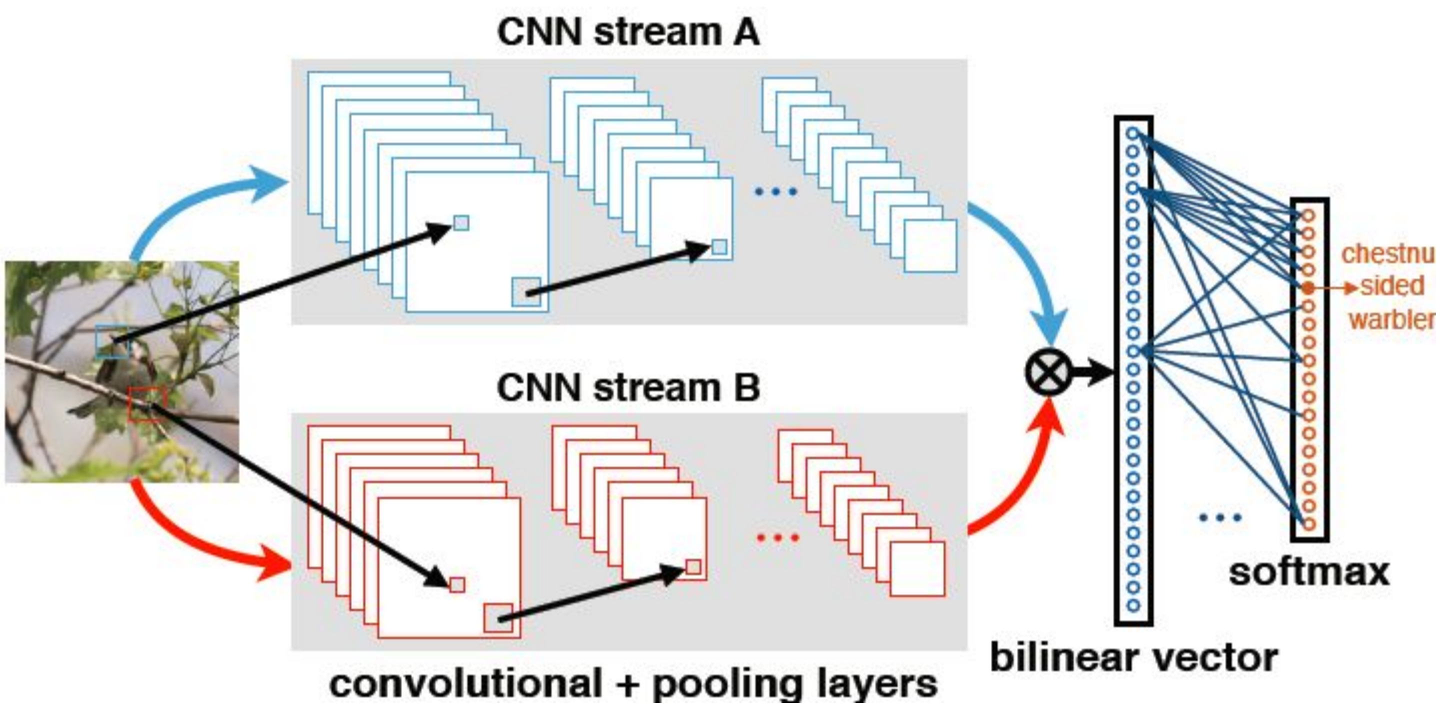
1. Bilinear CNN Models for Fine-grained Visual Recognition
2. At the very beginning of bi-linear idea
3. Bilinear SVM
4. Back to Bilinear CNN
5. Beyond bilinear
6. More?

Bilinear CNN Models for Fine-grained Visual Recognition

A bilinear model consists of two feature extractors whose outputs are multiplied using outer product at each location of the image and pooled to obtain an image descriptor.

Bilinear CNN Models for Fine-grained Visual Recognition

A bilinear model consists of two feature extractors whose outputs are multiplied using outer product at each location of the image and pooled to obtain an image descriptor.



Advantages of Bilinear CNN Models

1. Bilinear models can model local pairwise feature interactions in a translationally invariant manner, which is particularly useful for fine-grained categorization.

Advantages of Bilinear CNN Models

1. Bilinear models can model local pairwise feature interactions in a translationally invariant manner, which is particularly useful for fine-grained categorization.

2. It generalizes various orderless texture descriptors, such as Fisher Vector [1], VLAD [2], O2P [3].

[1] Improving the Fisher kernel for large-scale image classification
[2] Aggregating local descriptors into a compact image representation
[3] Semantic segmentation with second-order pooling.

Advantages of Bilinear CNN Models

1. Bilinear models can model local pairwise feature interactions in a translationally invariant manner, which is particularly useful for fine-grained categorization.
2. It generalizes various orderless texture descriptors, such as Fisher Vector [1], VLAD [2], O2P [3].
3. It allows end-to-end training using image labels only, and achieves state-of-the-art performance on fine-grained classification.

[1] Improving the Fisher kernel for large-scale image classification
[2] Aggregating local descriptors into a compact image representation
[3] Semantic segmentation with second-order pooling.

Cognitively…

Two streams

1. dorsal stream -- where pathway
2. ventral stream-- what pathway

Wikipediacally...

Bilinear as below

$$B(x, y) = x^{\mathrm{T}} A y = \sum_{i,j=1}^{n} a_{ij} x_i y_j.$$

function is linear w.r.t one variable when fixing the other

Mathematically...

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

Mathematically…

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

Mathematically…

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

Mathematically...

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

classification function $\mathcal{C}$

Mathematically...

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

classification function $\mathcal{C}$

feature function is a mapping $f : \mathcal{L} \times \mathcal{I} \rightarrow R^{c \times D}$, taking image $\mathcal{I}$ and a location $\mathcal{L}$ and outputs a feature of size $c \times D$

Mathematically…

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

classification function $\mathcal{C}$

feature function is a mapping $f : \mathcal{L} \times \mathcal{I} \rightarrow R^{c \times D}$, taking image $\mathcal{I}$ and a location $\mathcal{L}$ and outputs a feature of size $c \times D$

bilinear function $\text{bilinear}(l, \mathcal{I}, f_A, f_B) = f_A(l, \mathcal{I})^T f_B(l, \mathcal{I})$

Mathematically…

Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

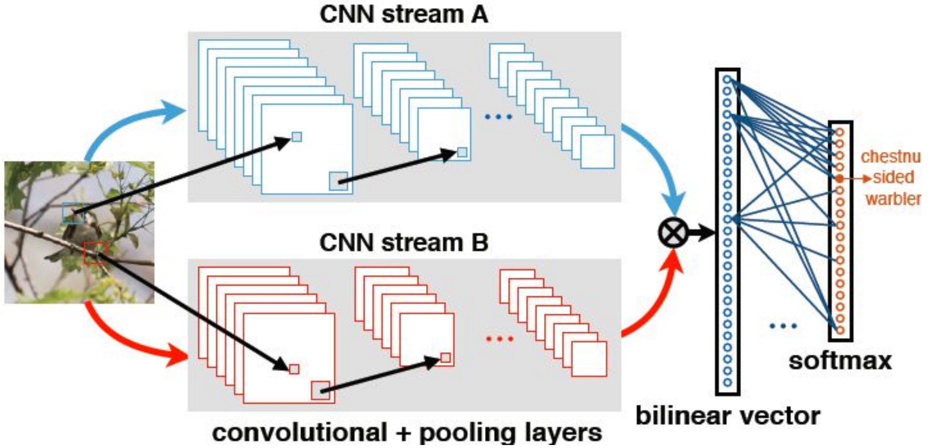
classification function $\mathcal{C}$

feature function is a mapping $f : \mathcal{L} \times \mathcal{I} \rightarrow R^{c \times D}$, taking image $\mathcal{I}$ and a location $\mathcal{L}$ and outputs a feature of size $c \times D$

bilinear function $\text{bilinear}(l, \mathcal{I}, f_A, f_B) = f_A(l, \mathcal{I})^T f_B(l, \mathcal{I})$

pooling $\phi(\mathcal{I}) = \sum_{l \in \mathcal{L}} \text{bilinear}(l, \mathcal{I}, f_A, f_B)$

Mathematically…



Bilinear model $\mathcal{B} = (f_A, f_B, \mathcal{P}, \mathcal{C})$

feature functions f_A and f_B

pooling function $\mathcal{P}$

classification function $\mathcal{C}$

feature function is a mapping $f : \mathcal{L} \times \mathcal{I} \to R^{c \times D}$, taking image $\mathcal{I}$ and a location $\mathcal{L}$ and outputs a feature of size $c \times D$

bilinear function $\text{bilinear}(l, \mathcal{I}, f_A, f_B) = f_A(l, \mathcal{I})^T f_B(l, \mathcal{I})$

pooling $\phi(\mathcal{I}) = \sum_{l \in \mathcal{L}} \text{bilinear}(l, \mathcal{I}, f_A, f_B)$

Operationally…

The pooled bilinear feature is $\mathbf{x} = A^T B$

Operationally...

The pooled bilinear feature is $\mathbf{x} = A^T B$

$$\mathbf{y} \leftarrow \operatorname{sign}(\mathbf{x})\sqrt{|\mathbf{x}|}$$

$$\mathbf{z} \leftarrow \mathbf{y}/||\mathbf{y}||_2$$

Operationally…

$$\mathbf{y} \leftarrow \mathrm{sign}(\mathbf{x})\sqrt{|\mathbf{x}|}$$

$$\mathbf{z} \leftarrow \mathbf{y}/||\mathbf{y}||_2$$

The pooled bilinear feature is $\mathbf{x} = A^T B$

Let $d\ell/d\mathbf{x}$ be the gradient of the loss function ℓ w.r.t $\mathbf{x}$

Operationally...

$$\mathbf{y} \leftarrow \operatorname{sign}(\mathbf{x})\sqrt{|\mathbf{x}|}$$

$$\mathbf{z} \leftarrow \mathbf{y}/||\mathbf{y}||_2$$

The pooled bilinear feature is $\mathbf{x} = A^T B$

Let $d\ell/d\mathbf{x}$ be the gradient of the loss function ℓ w.r.t $\mathbf{x}$

By chain rule, we have

$$\frac{d\ell}{dA} = B\left(\frac{d\ell}{d\mathbf{x}}\right)^T, \quad \frac{d\ell}{dB} = A\left(\frac{d\ell}{d\mathbf{x}}\right)$$

Operationally...

$$\mathbf{y} \leftarrow \operatorname{sign}(\mathbf{x})\sqrt{|\mathbf{x}|}$$

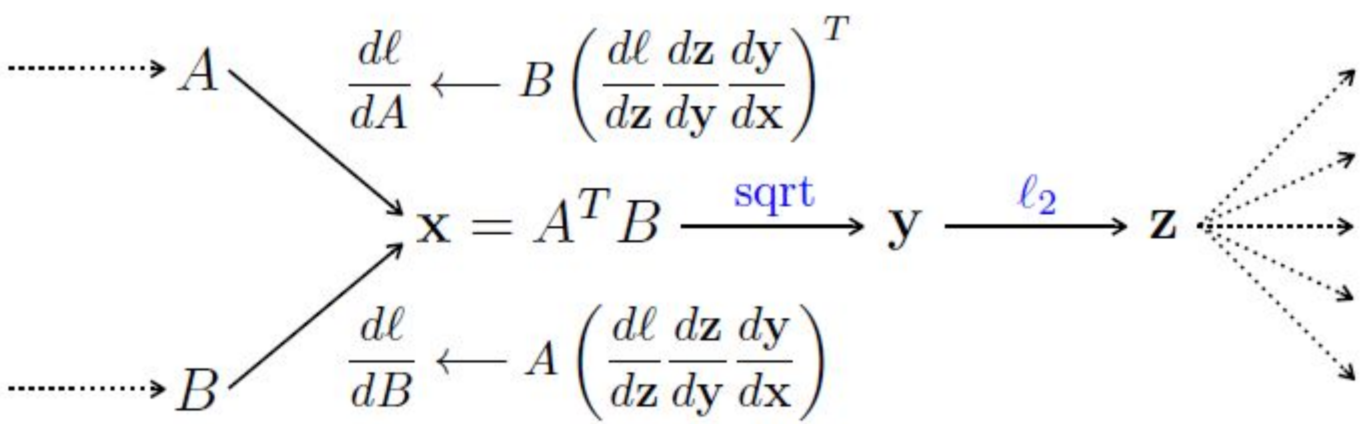
$$\mathbf{z} \leftarrow \mathbf{y}/||\mathbf{y}||_2$$

The pooled bilinear feature is $\mathbf{x} = A^T B$

Let $d\ell/d\mathbf{x}$ be the gradient of the loss function ℓ w.r.t $\mathbf{x}$

By chain rule, we have

$$\frac{d\ell}{dA} = B\left(\frac{d\ell}{d\mathbf{x}}\right)^T, \quad \frac{d\ell}{dB} = A\left(\frac{d\ell}{d\mathbf{x}}\right)$$



Experimentally...

FV-SIFT

Experimentally...

FV-SIFT

FC-CNN

Experimentally...

FV-SIFT

FC-CNN

FV-CNN

Experimentally...

FV-SIFT

FC-CNN

FV-CNN

B-CNN

Experimentally…

FV-SIFT

FC-CNN

FV-CNN

B-CNN

Two pretrained CNN models, M-Net (medium-size net) and D-Net (VGG19)

Experimentally...

FV-SIFT

FC-CNN

FV-CNN

B-CNN

Two pretrained CNN models, M-Net (medium-size net) and D-Net (VGG19)

Bilinear pooling over the output of last convolutional layer

Experimentally...

FV-SIFT

FC-CNN

FV-CNN

B-CNN

Two pretrained CNN models, M-Net (medium-size net) and D-Net (VGG19)

Bilinear pooling over the output of last convolutional layer

Finetune with softmax

Experimentally...

FV-SIFT

FC-CNN

FV-CNN

B-CNN

Two pretrained CNN models, M-Net (medium-size net) and D-Net (VGG19)

Bilinear pooling over the output of last convolutional layer

Finetune with softmax

Learn linear SVM as the classifier (Why?)

Experimentally...

Data augmentation -- left-right flipping

method	birds		birds + box		aircrafts		cars		FPS
	w/o ft	w/ ft	w/o ft	w/ ft	w/o ft	w/ ft	w/o ft	w/ ft	
FV-SIFT	18.8	-	22.4	-	61.0	-	59.2	-	10†
FC-CNN [M]	52.7	58.8	58.0	65.7	44.4	57.3	37.3	58.6	124
FC-CNN [D]	61.0	70.4	65.3	76.4	45.0	74.1	36.5	79.8	43
FV-CNN [M]	61.1	64.1	67.2	69.6	64.3	70.1	70.8	77.2	23
FV-CNN [D]	71.3	74.7	74.4	77.5	70.4	77.6	75.2	85.7	8
B-CNN [M,M]	72.0	78.1	74.2	80.4	72.7	77.9	77.8	86.5	87
B-CNN [D,M]	80.1	**84.1**	81.3	**85.1**	78.4	83.9	83.9	**91.3**	8
B-CNN [D,D]	80.1	84.0	80.1	84.8	76.8	**84.1**	82.9	90.6	10
Previous work	**84.1** [19], 82.0 [21] 73.9 [38], 75.7 [2]		82.8 [21], 73.5 [24] 73.0 [7], 76.4 [38]		72.5 [4], 80.7 [16]		**92.6** [21], 82.7 [16] 78.0 [4]		†on a cpu

method	birds mean per-class acc.		birds + box mean per-class acc.		aircrafts mean per-class acc.		cars fraction of correct	
	w/o ft	w/ ft	w/o ft	w/ ft	w/o ft	w/ ft	w/o ft	w/ ft
FV-SIFT	12.8	-	24.1	-	55.7	-	51.2	-
FC-CNN (M)	46.1	55.6	56.5	64.0	41.3	50.4	33.5	50.9
FC-CNN (D)	54.6	64.9	63.0	71.4	40.7	57.8	32.0	67.7
FV-CNN (M)	50.8	56.2	63.4	67.1	58.7	64.5	65.5	68.9
FV-CNN (D)	62.7	68.7	70.5	73.9	67.5	71.2	70.2	79.2
B-CNN (M,M)	66.6	72.5	70.7	77.2	67.9	73.5	73.9	82.3
B-CNN (D,M)	75.1	**80.9**	77.9	**81.9**	73.3	79.4	81.2	**88.2**
State-of-the-art	66.7 [7], 73.9 [35], 75.7 [2]		73.0 [7], 73.5 [21], 76.4 [35]		72.5 [4], **80.7** [16]		78.0 [4], 82.7 [16]	

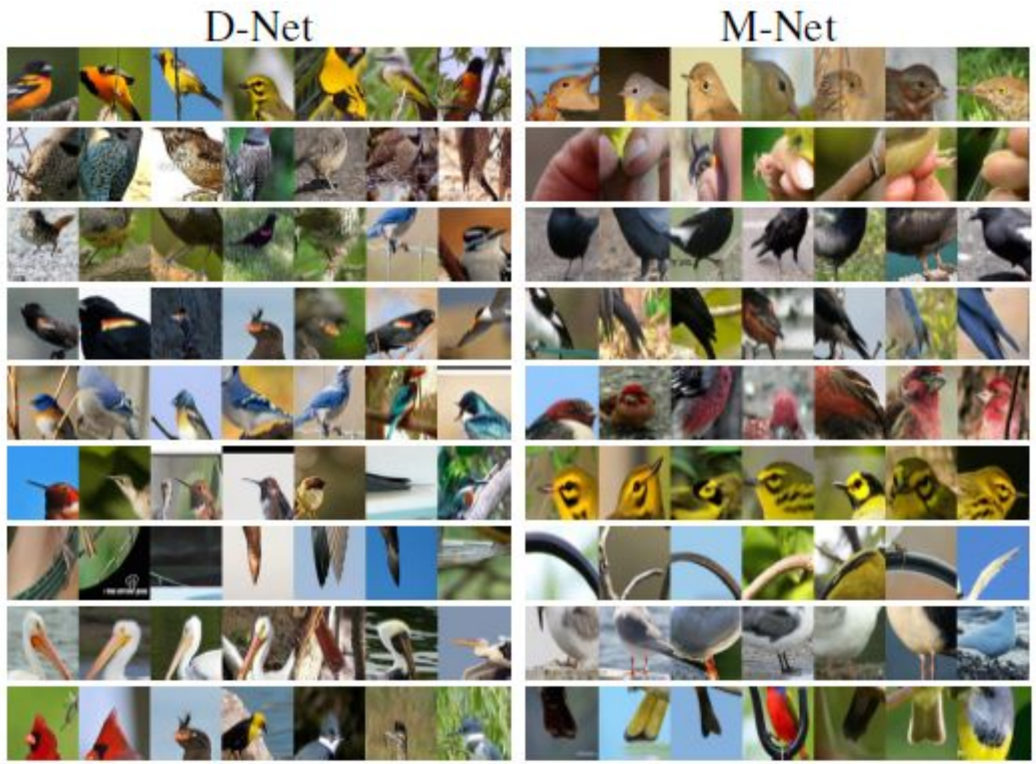
Diagnostically...

B-CNN(M,D)

normalization	accuracy	mAP
square-root + ℓ_2	80.1	81.3
square-root only	79.4	77.9
ℓ_2 only	77.3	79.6
none	74.7	70.9

Diagnostically...

B-CNN(M,D)



At the Very Beginning of Bilinear

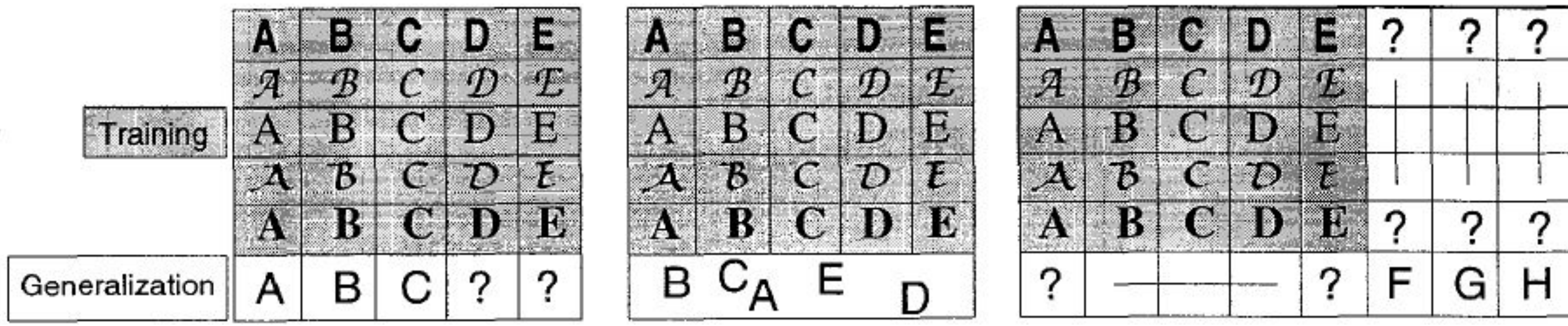
Two real-world problems (content and style)

1. JB Tenenbaum, W. T. Freeman, Separating Style and Content, NIPS, 1997
2. W. T. Freeman, JB Tenenbaum, Learning bilinear models for two-factor problems in vision, CVPR, 1997

At the Very Beginning of Bilinear

Two real-world problems (content and style)

1. letters with different fonts

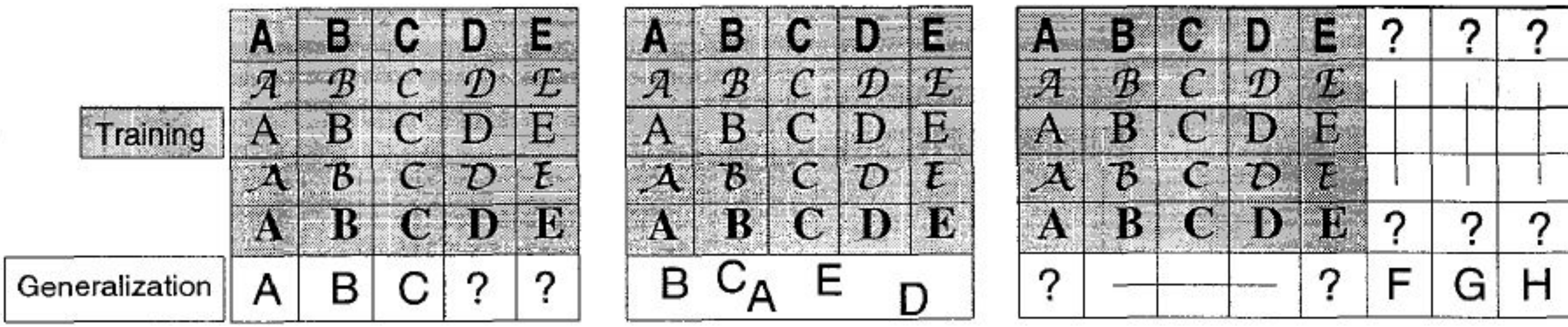


1. JB Tenenbaum, W. T. Freeman, Separating Style and Content, NIPS, 1997
2. W. T. Freeman, JB Tenenbaum, Learning bilinear models for two-factor problems in vision, CVPR, 1997

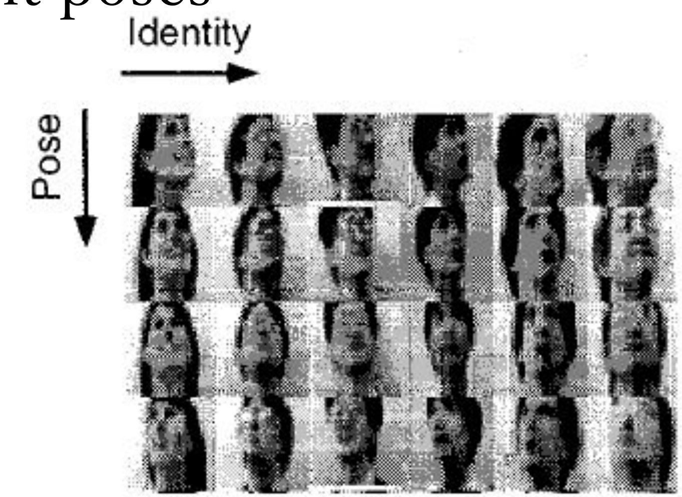
At the Very Beginning of Bilinear

Two real-world problems (content and style)

1. letters with different fonts



2. individual face images with different poses



1. JB Tenenbaum, W. T. Freeman, Separating Style and Content, NIPS, 1997
2. W. T. Freeman, JB Tenenbaum, Learning bilinear models for two-factor problems in vision, CVPR, 1997

Original Bilinear Model

Observation $\mathbf{y}^{sc}$ can be characterized by content code $\mathbf{b}^c$ and style code $\mathbf{a}^s$, parameters W are independent of content and style but govern their interactions

Mathematically, the k-th element in observation $\mathbf{y}^{sc}$

$$y_k^{sc} = \mathbf{a}^{s^{\mathrm{T}}} \mathbf{W}_k \mathbf{b}^c = \sum_{ij} a_i^s b_j^c W_{ijk}$$

Original Bilinear Model

Observation $\mathbf{y}^{sc}$ can be characterized by content code $\mathbf{b}^c$ and style code $\mathbf{a}^s$, parameters W are independent of content and style but govern their interactions

Mathematically, the k-th element in observation $\mathbf{y}^{sc}$

$$y_k^{sc} = \mathbf{a}^{s\mathrm{T}} \mathbf{W}_k \mathbf{b}^c = \sum_{ij} a_i^s b_j^c W_{ijk}$$

Or…

$$\mathbf{y}^{sc} = \mathbf{A}^s \mathbf{b}^c \qquad A_{jk}^s \equiv \sum_i a_i^s W_{ijk}^s$$

$$\mathbf{y}^{sc} = \mathbf{B}^c \mathbf{a}^s$$

Training

By minimizing the least square fitting, the parameters can be learned iteratively.

$$\mathbf{y}^{sc} = \mathbf{A}^{s}\mathbf{b}^{c} \qquad A^{s}_{jk} \equiv \sum_{i} a^{s}_{i} W^{s}_{ijk}$$

$$\mathbf{y}^{sc} = \mathbf{B}^{c}\mathbf{a}^{s}$$

With learned codes, W can also be learned.

Bilinear SVM

Motivation -- image region can be naturally represented by a matrix, which is a 2D data, then why do people vectorize the matrix to train a linear SVM?

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Bilinear SVM

Motivation -- image region can be naturally represented by a matrix, which is a 2D data, then why do people vectorize the matrix to train a linear SVM?

Generalize predictor from vector to matrix, and consider low-rank constraint to reduce the degrees of freedom in the matrix W

$$f_w(x) = w^T x$$

$$f_W(X) = \mathrm{Tr}(W^T X)$$

$$f_{W_y,W_x}(X) = \mathrm{Tr}(W_x W_y^T X) = \mathrm{Tr}(W_y^T X W_x) \qquad W = W_y W_x^T$$

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Bilinear SVM

from linear SVM to bilinear SVM

$$L(w) = \frac{1}{2} w^T w + C \sum_n \max(0, 1 - y_n w^T x_n)$$

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Bilinear SVM

from linear SVM to bilinear SVM

$$L(w) = \frac{1}{2} w^T w + C \sum_n \max(0, 1 - y_n w^T x_n)$$

$$L(W) = \frac{1}{2} \mathrm{Tr}(W^T W) + C \sum_n \max(0, 1 - y_n \mathrm{Tr}(W^T X_n))$$

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Bilinear SVM

from linear SVM to bilinear SVM

$$L(w) = \frac{1}{2}w^T w + C\sum_n \max(0, 1 - y_n w^T x_n)$$

$$L(W) = \frac{1}{2}\,\mathrm{Tr}(W^T W) + C\sum_n \max(0, 1 - y_n\,\mathrm{Tr}(W^T X_n))$$

$$L(W_y, W_x) = \frac{1}{2}\,\mathrm{Tr}(W_x W_y^T W_y W_x^T) + C\sum_n \max(0, 1 - y_n\,\mathrm{Tr}(W_x W_y^T X_n))$$

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Bilinear SVM

from linear SVM to bilinear SVM

$$L(w) = \frac{1}{2}w^T w + C\sum_n \max(0, 1 - y_n w^T x_n)$$

$$L(W) = \frac{1}{2}\operatorname{Tr}(W^T W) + C\sum_n \max(0, 1 - y_n \operatorname{Tr}(W^T X_n))$$

$$L(W_y, W_x) = \frac{1}{2}\operatorname{Tr}(W_x W_y^T W_y W_x^T) + C\sum_n \max(0, 1 - y_n \operatorname{Tr}(W_x W_y^T X_n))$$

Focusing on W_y

$$\min_{\tilde{W}_y} L(\tilde{W}_y, W_x) = \frac{1}{2}\operatorname{Tr}(\tilde{W}_y^T \tilde{W}_y) + C\sum_n \max(0, 1 - y_n \operatorname{Tr}(\tilde{W}_y^T \tilde{X}_n))$$

where $\quad \tilde{W}_y = W_y A^{\frac{1}{2}} \quad$ and $\quad \tilde{X}_n = X_n W_x A^{-\frac{1}{2}} \quad$ and $\quad A = W_x^T W_x.$

1. Hamed Pirsiavash, Deva Ramanan, Charless Fowlkes, Bilinear classifiers for visual recognition, NIPS 2009

Back to Bilinear CNN

Does bilinear CNN have meaningful explanation -- previous papers have good motivation to choose bilinear.

Back to Bilinear CNN

Does bilinear CNN have meaningful explanation -- previous papers have good motivation to choose bilinear.

Is it easy to extend bilinear CNN to multilinear version? Two streams/nets work better than a single net, how about three nets? Tensor product?

Back to Bilinear CNN

Does bilinear CNN have meaningful explanation -- previous papers have good motivation to choose bilinear.

Is bilinear CNN easy to be extended to multilinear? Two streams/nets work better than a single net? How about three nets? Tensor product?

Can we see it as a new way to fuse features from two sources?

Beyond Bilinear -- Compact

Compact Bilinear Pooling -- reducing the dimensionality of outer-product features

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Compact Bilinear Pooling -- reducing the dimensionality of outer-product features

The authors present this from the kernelized view.

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Compact Bilinear Pooling -- reducing the dimensionality of outer-product features

The authors present this from the kernelized view.

Recall bilinear pooling resulting into a c^2 length vector

$$B(\mathcal{X}) = \sum_{s \in \mathcal{S}} x_s x_s^T \qquad \mathcal{X} = (x_1, \ldots, x_{|\mathcal{S}|}, x_s \in \mathbb{R}^c)$$

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Compact Bilinear Pooling -- reducing the dimensionality of outer-product features

The authors present this from the kernelized view.

Recall bilinear pooling resulting into a c^2 length vector

$$B(\mathcal{X}) = \sum_{s \in \mathcal{S}} x_s x_s^T \qquad \mathcal{X} = (x_1, \ldots, x_{|\mathcal{S}|}, x_s \in \mathbb{R}^c)$$

linear kernel

$$\begin{aligned} \langle B(\mathcal{X}), B(\mathcal{Y}) \rangle &= \langle \sum_{s \in \mathcal{S}} x_s x_s^T, \sum_{u \in \mathcal{U}} y_u y_u^T \rangle \\ &= \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle x_s x_s^T, y_u y_u^T \rangle \\ &= \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle x_s, y_u \rangle^2 \end{aligned}$$

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Recall bilinear pooling resulting into a c^2 length vector

$$B(\mathcal{X}) = \sum_{s \in \mathcal{S}} x_s x_s^T \qquad \mathcal{X} = (x_1, \ldots, x_{|\mathcal{S}|}, x_s \in \mathbb{R}^c)$$

If a low-dimension feature $\phi(x) \in R^d$ can be found ($d << c^2$) to approximate $\langle \phi(x), \phi(y) \rangle \approx \langle x, y \rangle^2$

then...

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Recall bilinear pooling resulting into a c^2 length vector

$$B(\mathcal{X}) = \sum_{s \in \mathcal{S}} x_s x_s^T \qquad \mathcal{X} = (x_1, \ldots, x_{|\mathcal{S}|}, x_s \in \mathbb{R}^c)$$

If a low-dimension feature $\phi(x) \in R^d$ can be found ($d << c^2$) to approximate $\langle \phi(x), \phi(y) \rangle \approx \langle x, y \rangle^2$ (second-order pooling approx.)

then...

$$\begin{aligned} \langle B(\mathcal{X}), B(\mathcal{Y}) \rangle &= \langle \sum_{s \in \mathcal{S}} x_s x_s^T, \sum_{u \in \mathcal{U}} y_u y_u^T \rangle \\ &= \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle x_s x_s^T, y_u y_u^T \rangle \\ &= \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle x_s, y_u \rangle^2 \end{aligned}$$

$$\Longrightarrow$$

$$\begin{aligned} \langle B(\mathcal{X}), B(\mathcal{Y}) \rangle &= \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle x_s, y_u \rangle^2 \\ &\approx \sum_{s \in \mathcal{S}} \sum_{u \in \mathcal{U}} \langle \phi(x), \phi(y) \rangle \\ &= \langle \sum_{s \in \mathcal{S}} \phi(x_s), \sum_{u \in \mathcal{U}} \phi(y_u) \rangle \\ &\equiv \langle C(\mathcal{X}), C(\mathcal{Y}) \rangle, \end{aligned}$$

1. Yang Gao, Oscar Beijbom, Ning Zhang, Trevor Darrell, Compact Bilinear Pooling, arxiv 2015

Beyond Bilinear -- Compact

Then the compact bilinear pooling is

$$C(\mathcal{X}) := \sum_{s \in \mathcal{S}} \phi(x_s)$$

1. P. Kar and H. Karnick. Random feature maps for dot product kernels, AISTATS 2012
2. N. Pham and R. Pagh. Fast and scalable polynomial kernels via explicit feature maps, SIGKDD 2013

Beyond Bilinear -- Compact

Then the compact bilinear pooling is

$$C(\mathcal{X}) := \sum_{s \in \mathcal{S}} \phi(x_s)$$

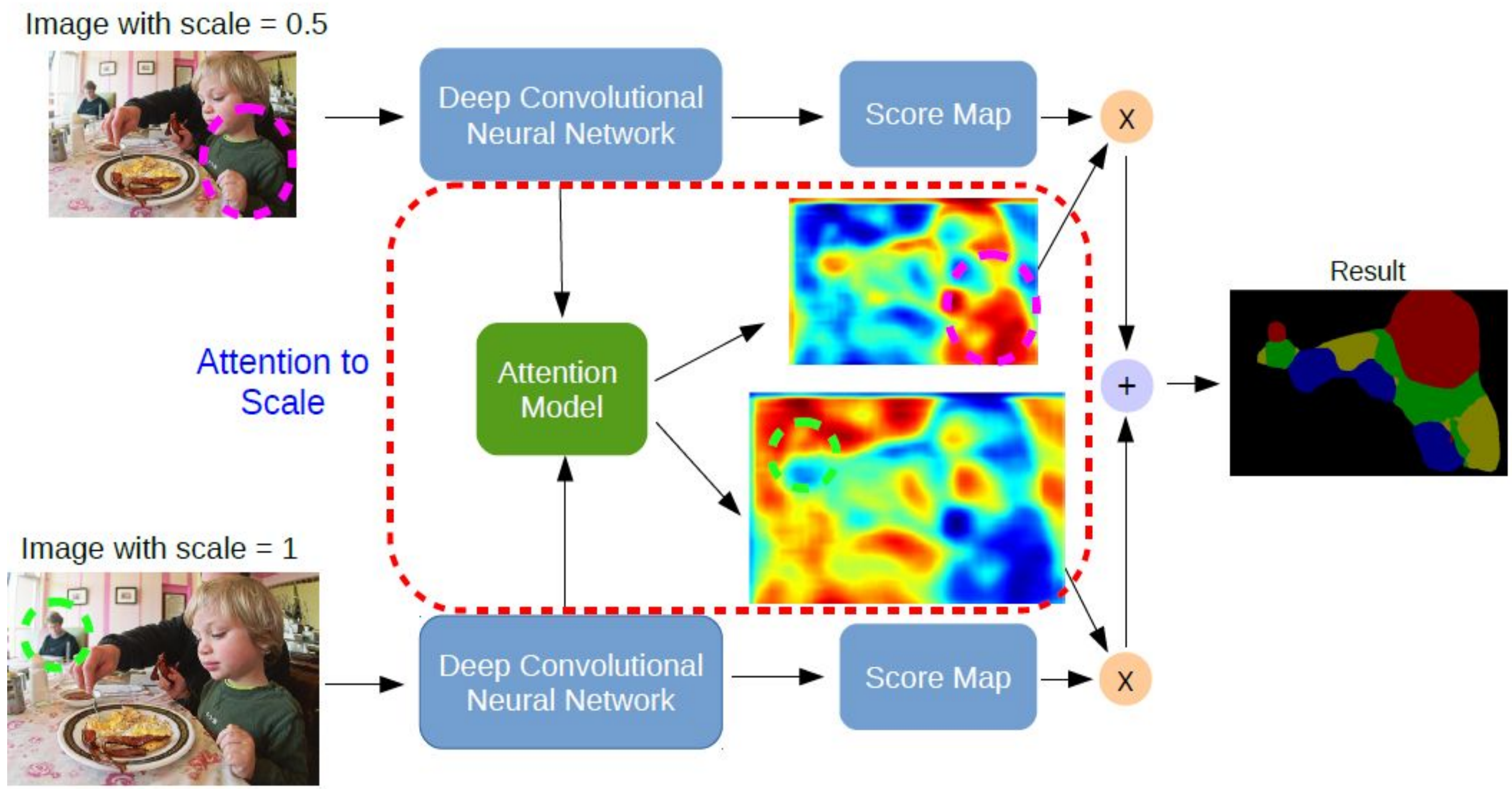
Two approximations are presented:

1. Random Maclaurin
2. Tensor Sketch

1. P. Kar and H. Karnick. Random feature maps for dot product kernels, AISTATS 2012
2. N. Pham and R. Pagh. Fast and scalable polynomial kernels via explicit feature maps, SIGKDD 2013

Beyond Bilinear -- Multiplicative Pooling

Motivation -- consider multiple scales, and softly weight features from different input scales when predicting the semantic label of a pixel.



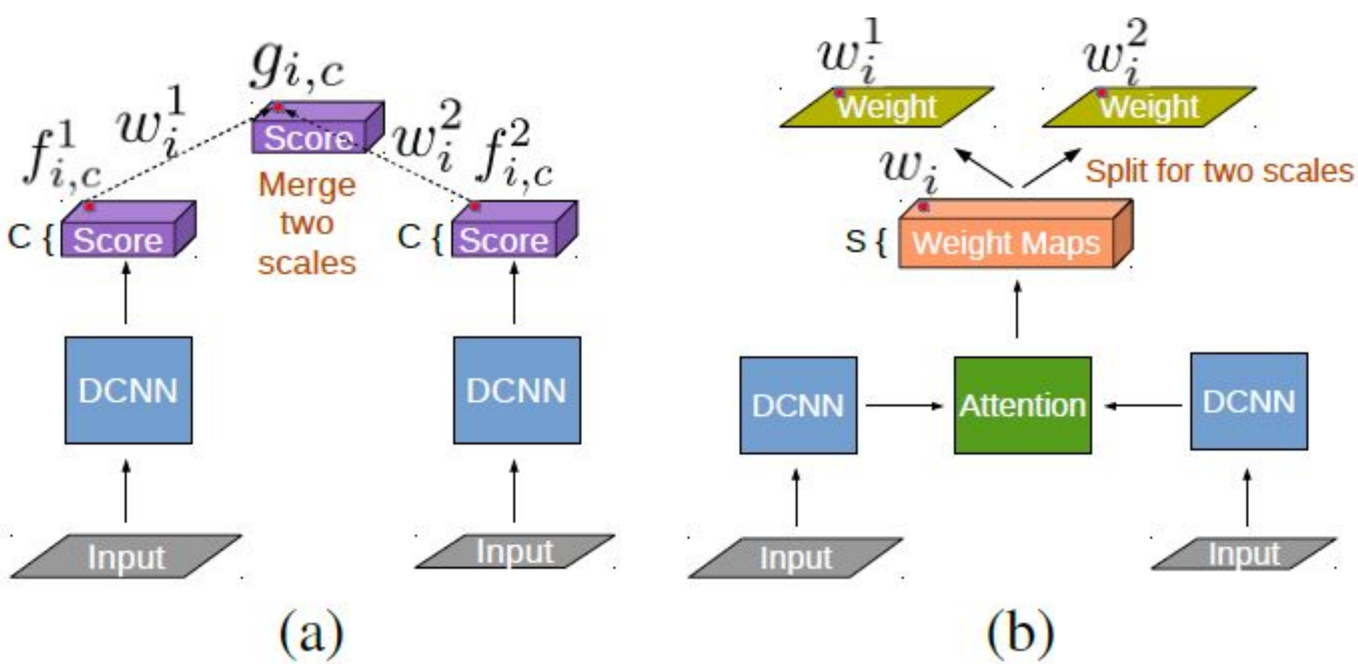
1. L-C Chen, Y Yang, J. Wang, W. Xu, A. Yuille, Attention to Scale: Scale-aware Semantic Image Segmentation, arxiv, 2015

Beyond Bilinear -- Multiplicative Pooling

Look into each pixel:

$f_{i,c}^{s}$ "i" index the spatial location of this pixel, at scale "s", for class-"c"

$$g_{i,c} = \sum_{s=1}^{S} w_i^s \cdot f_{i,c}^s \qquad w_i^s = \frac{\exp(h_i^s)}{\sum_{t=1}^{S} \exp(h_i^t)}$$



1. L-C Chen, Y Yang, J. Wang, W. Xu, A. Yuille, Attention to Scale: Scale-aware Semantic Image Segmentation, arxiv, 2015

More to think

A trend is emerging -- bilinear-, multilinear- and multiplicative-operation allow models to produce instance-adaptive features and weights in specific problems.

More to think

A trend is emerging -- bilinear-, multilinear- and multiplicative-operation allow models to produce instance-adaptive features and weights in specific problems.

Some typical problems -

1. detection and recognition
2. style, content, and photo aesthetics
3. human pose, human size, human detection
4. face viewpoint, keypoint, face detection
5. ......

Conclusion

1. Bilinear CNN Model
2. At the very beginning of bi-linear idea
3. Bilinear SVM
4. Back to Bilinear CNN
5. Beyond bilinear
6. More!

Thanks

Reference

1. JB Tenenbaum, W. T. Freeman, Separating Style and Content, NIPS, 1997

2. W. T. Freeman, JB Tenenbaum, Learning bilinear models for two-factor problems in vision, CVPR, 1997

3. H. Pirsiavash, D. Ramanan, C. Fowlkes, Bilinear classifier for visual recognition, NIPS 2009

4. T-Y Lin, A. RoyChowdhury, S. Maji, Bilinear CNN Models for Fine-grained Visual Recognition, ICCV 2015

5. Y. Gao, Oscar Beijbom, N. Zhang, T. Darrell, Compact Bilinear Pooling, arxiv, 2015

6. L-C Chen, Y Yang, J. Wang, W. Xu, A. Yuille, Attention to Scale: Scale-aware Semantic Image Segmentation, arxiv, 2015