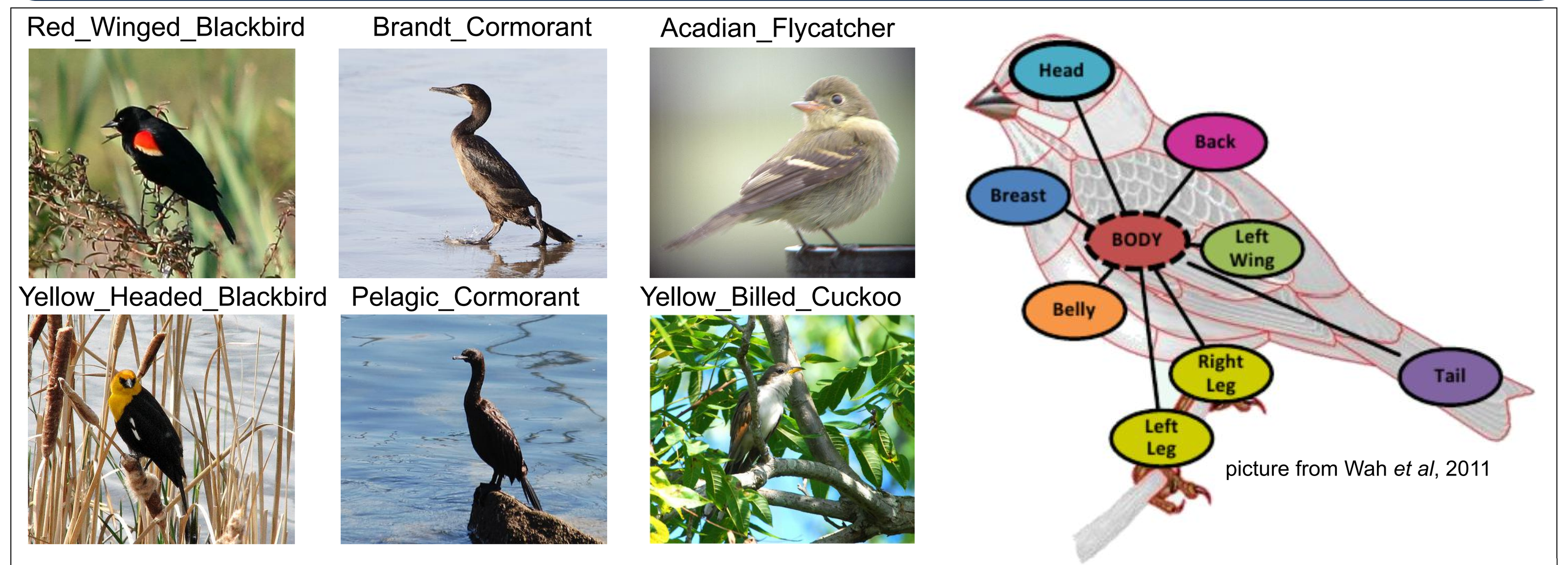
Low-Rank Bilinear Pooling for Fine-Grained Classification

Shu Kong, Charless Fowlkes

Department of Computer Science, University of California Irvine, Irvine, California, USA

Abstract



capturing subtle difference by correlation of part features

Highlights

1. Pooling second-order statistics of local features, represented by **matrix** vs. ~~vector~~
2. Using bilinear classifier for classification, other than the linear one **"Bilinear" Classifier** vs. ~~Linear Classifier~~
3. Coupling bilinear feature and bilinear classifier, and showing the classification score is essentially the Frobenius norm of local features

$$\mathbf{w}^T vec(\mathbf{X}\mathbf{X}^T) \Leftrightarrow tr(\mathbf{W}^T\mathbf{X}\mathbf{X}^T) \Leftrightarrow tr(\mathbf{U}\mathbf{U}^T\mathbf{X}_i\mathbf{X}_i^T)$$

$$\|\mathbf{U}^T\mathbf{X}\|_F^2 \Leftrightarrow tr(\mathbf{U}^T\mathbf{X}\mathbf{X}^T\mathbf{U})$$

4. Co-decompose of classifiers to further compress the model, yet allowing to train in an end-to-end manner without requiring learning the classifier first (*k*-th classifier corresponding to the *k*-th class)

$$\mathbf{U}_k^T \approx \mathbf{V}_k^T \times \mathbf{P}$$

Real-World Low-Rank Observation

Reshaping the learned linear SVM classifiers into matrix form, decomposing each one by PCA and plotting the sorted eigenvalue with standard deviation versus the eigenvalue index, and accuracy versus rank of the classifiers.

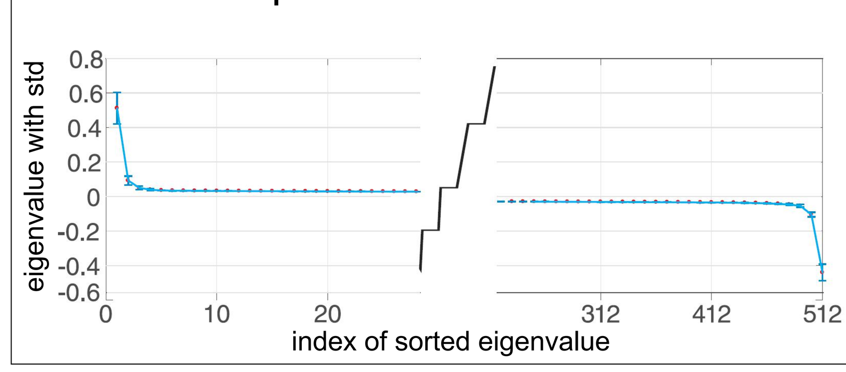
linear SVM $\max(0, 1 - y_i \mathbf{w}^T \mathbf{z}_i + b)$

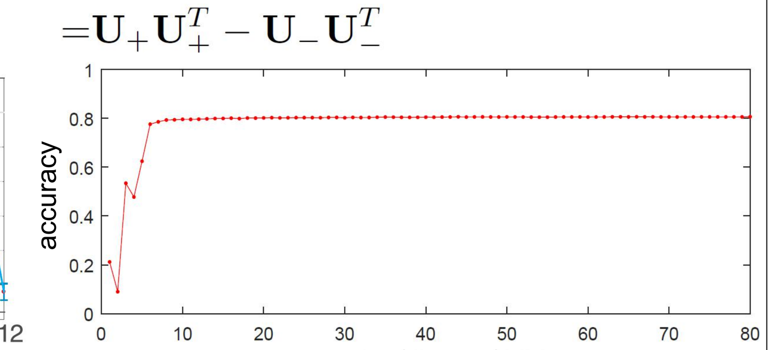
linear SVM in matrix $\max(0, 1 - y_i \mathrm{tr}(\mathbf{W}^T\mathbf{X}_i\mathbf{X}_i^T) + b)$

rank-*r* SVM $\max(0, 1 - y_i \mathrm{tr}(\mathbf{W}_r^T\mathbf{X}_i\mathbf{X}_i^T) + b)$

where we compute the low-rank counterpart through PCA on the classifier parameters

$$\mathbf{W}_r = \mathbf{\Psi}\mathbf{\Sigma}_r\mathbf{\Psi}^T = \mathbf{\Psi}_+\mathbf{\Sigma}_{r/2}^+\mathbf{\Psi}_+^T + \mathbf{\Psi}_-\mathbf{\Sigma}_{r/2}^-\mathbf{\Psi}_-^T$$
$$= \mathbf{\Psi}_+\mathbf{\Sigma}_{r/2}^+\mathbf{\Psi}_+^T - \mathbf{\Psi}_-|\mathbf{\Sigma}_{r/2}^-|\mathbf{\Psi}_-^T$$
$$= \mathbf{U}_+\mathbf{U}_+^T - \mathbf{U}_-\mathbf{U}_-^T$$





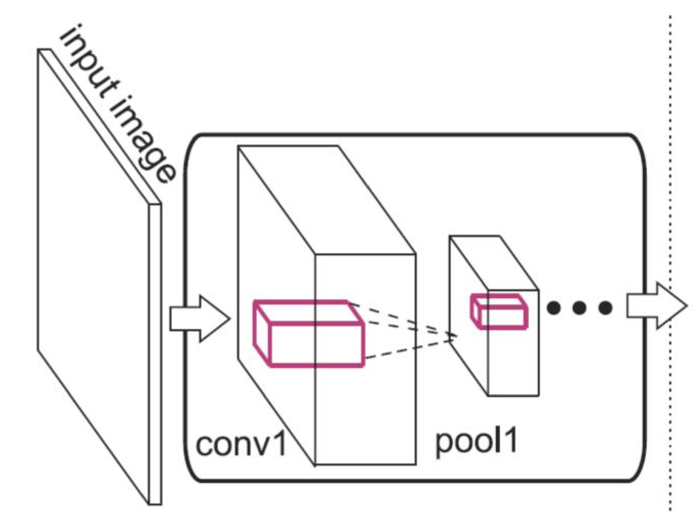
When Bilinear Feature Meets Bilinear Classifier

CNN for local feature extraction

Using **VGG16** as base model to extract local features at the *conv5_3* layer

input image size
448x448 pixel resolution

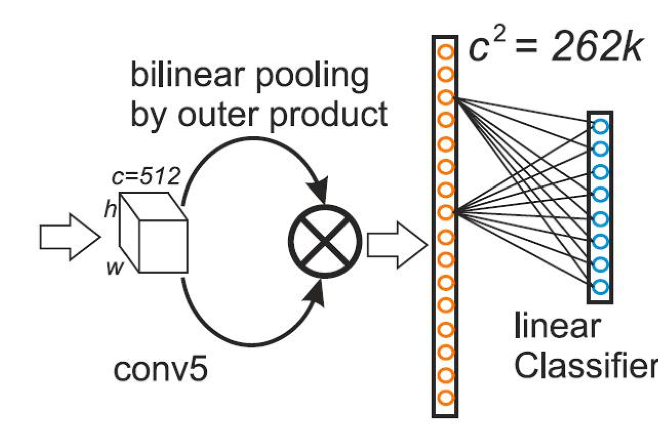
feature size
height and width *h*=*w*=28
channel *c*=512



Full Bilinear Model

Full bilinear CNN model uses linear SVM classifier over vectorized of the bilinear feature, which sums the outer product of local features

feature dimension
*c***c*=512*512=262,144

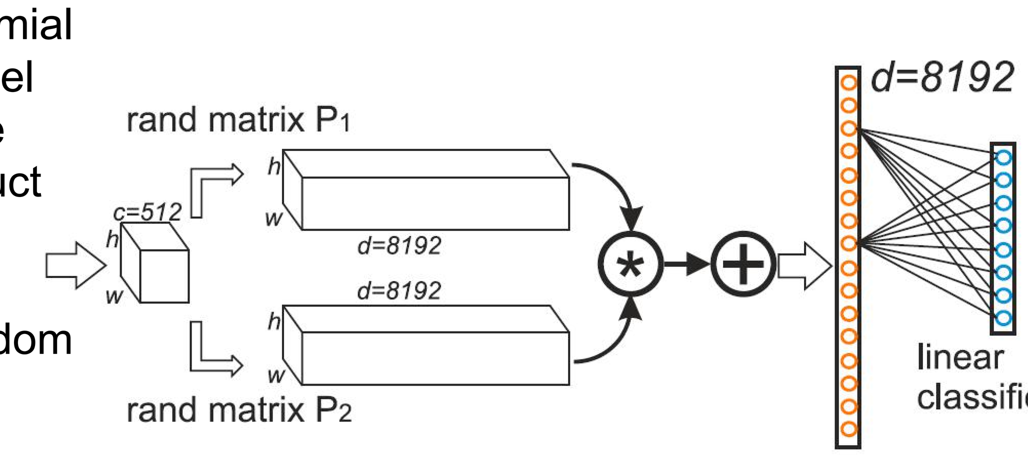


Compact Bilinear Model

By approximating the polynomial kernel, compact bilinear model computes the bilinear feature as summed Hadamard product of higher-dimensional r andom projection of local features, with two binary random matrix

P1 and P2 of size
c×*d*=512×8192

feature dimension
d=8192

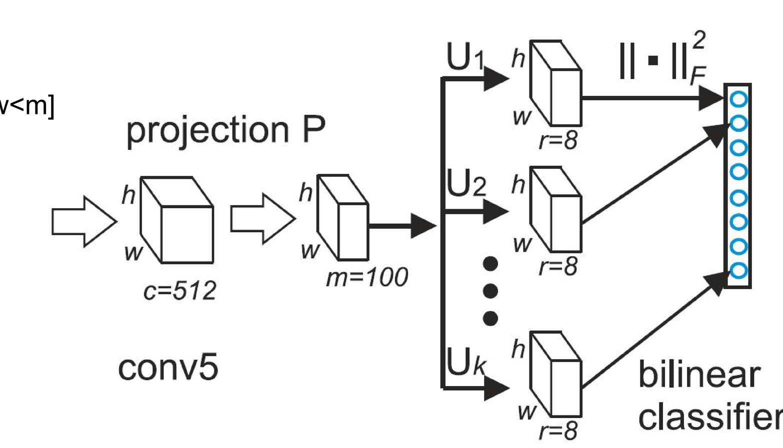


Our Model (LRBP-I) [more efficient useful when hw<m]

using Frobenious norm of local features as the classification score, avoiding explicitly computing bilinear feature

P of size
c×*m*=512×100

feature length
m×*hw*=100×28*28

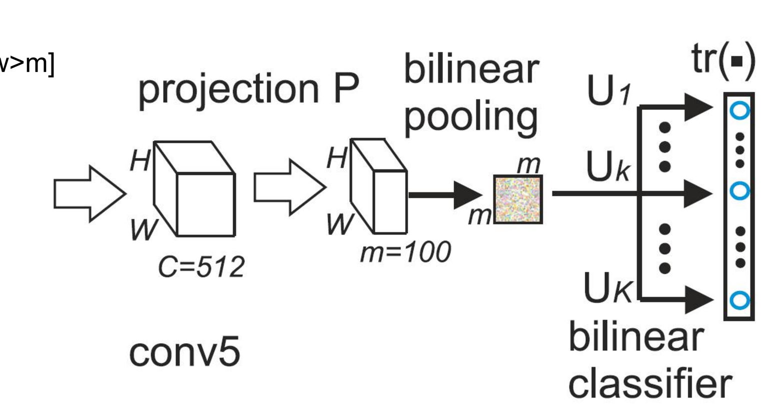


Our Model (LRBP-II) [more efficient useful when hw>m]

computing bilinear feature on reduced local features

P of size
c×*m*=512×100

feature length
m×*m*=100×100



Demo, code and model can be found at the project webpage under author's webpage
http://www.ics.uci.edu/~skong2/lr_bilinear.html

Acknowledgements: This work was supported by NSF grants DBI-1262547, IIS-1253538, hardware donation from NVIDIA.

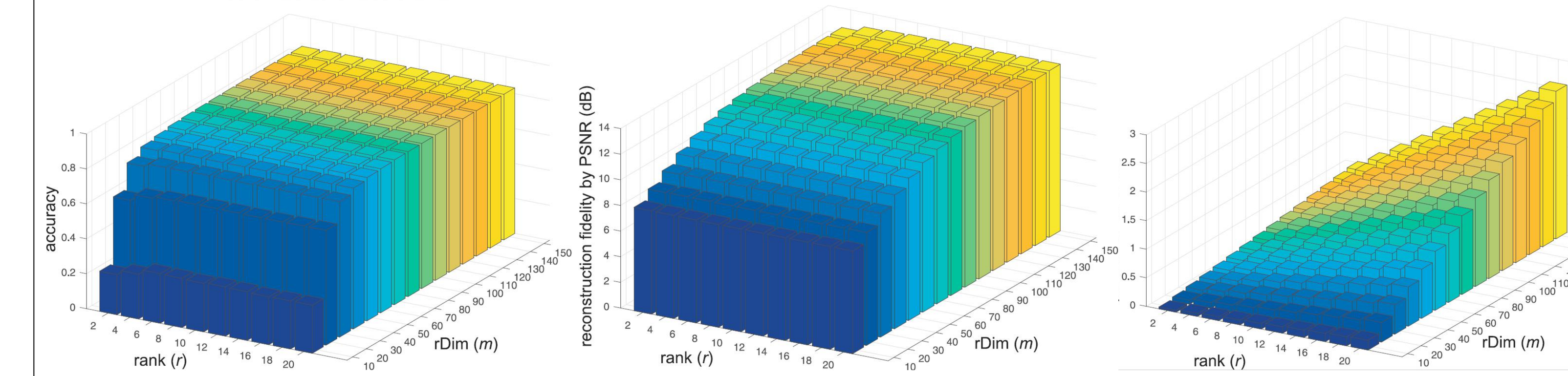
Classifier Co-Decomposition

Assumption -- the channels can be merged as not all channels contribute equally to classification

Idea -- reconstructing classifiers using low-dimensional factors

$$\min_{\mathbf{V}_k, \mathbf{P}} \sum_{k=1}^{K} \|\mathbf{U}_k - \mathbf{P}\mathbf{V}_k\|_F^2$$

Note -- this also allows end-to-end training without learning classifiers in advance



Classifier accuracy vs. reduced dimension (*m*) and rank (*r*) — Reconstruction fidelity (PSNR) of classifier parameters vs. reduced dimension (*m*) and rank (*r*) — learned parameter size vs. reduced dimension (*m*) and rank (*r*)

Experiment

Computational efficiency analysis
on CUB dataset, *K*=200, *c*=512, *h*=*w*=28, *d*=8192, *m*=100, *r*=8

	Full Bilinear	Random Maclaurin	Tensor Sketch	LRBP-I	LRBP-II
Fea. Dimension	c^2 [262K]	d [10K]	d [10K]	mhw [78K]	m^2 [10K]
Fea. Para. Mem.	0	$2cd$ [40MB]	$2c$ [4KB]	cm [200KB]	cm [200KB]
Cls. Para. Mem.	Kc^2 [KMB]	Kd [32KKB]	Kd [32KKB]	Krm [3KKB]	Krm [3KKB]
total Para. Mem.	Kc^2	$2cd + Kd$	$2c + Kd$	$cm + Krm$	$cm + Krm$
Computation fea.	$O(hwc^2)$	$O(hwcd)$	$O(hw(c + d\log d))$	$O(hwmc)$	$O(hwmc + hwm^2)$
Computation cls.	$O(Kc^2)$	$O(Kd)$	$O(Kd)$	$O(Krmhw)$	$O(Krm^2)$
total Para. Mem. ($K = 200$)	200MB	48MB	8MB	0.8MB	0.8MB

FC-VGG16 -- fully connected layer on VGG16
Fisher -- improved Fisher Encoding on activations at *conv5_3* of VGG16 as local features
Full Bilinear -- full bilinear CNN model
Random Maclaurin approach used for approximating polynomial kernel
Tensor Sketch approach used for approximating polynomial kernel
Ours -- the proposed method in our paper

summary statistics of datasets

	# train img.	# test img.	# class
CUB	5994	5794	200
DTD	1880	3760	47
Car	8144	8041	196
Airplane	6667	3333	100

Classification accuracy on benchmark datasets

	FC-VGG16	Fisher	Full Bilinear	Random Maclaurin	Tensor Sketch	LRBP (Ours)
CUB	70.40	74.7	84.01	83.86	84.00	**84.21**
DTD	59.89	65.53	64.96	65.57	64.51	**65.80**
Car	76.80	85.70	91.18	89.54	90.19	**90.92**
Airplane	74.10	77.60	87.09	87.10	87.18	**87.31**
param. size (CUB)	67MB	50MB	200MB	48MB	8MB	0.8MB

visualization

1. gradient map
2. average activation map
3. simplying input image by removing superpixels