Improved Bilinear Pooling with Pseudo Square-Rooted Matrix

Sixiang Xu, Damien Muselet, Alain Trémeau and Licheng Jiao, *Fellow, IEEE*

***Abstract*—Bilinear pooling is a feature aggregation step applied after the convolutional layers of a deep network and encodes a matrix of local features into a fixed-size bilinear representation. It improves performance in many image classification tasks. Since its emergence, this pooling has seen two major improvements: Compact Bilinear Pooling (CBP) and square-root normalization. Recently, the combination of these two elements has been widely studied. However, due to the lack of good normalization solutions, existing combination approaches showed less efficiency when they are plugged into different networks and less compatibility when they work with existing CBP techniques. To solve this problem, in this paper, we propose to apply Newton iterations, a fast square-root normalization method, to produce a new normalized matrix called *pseudo square-rooted matrix*. Subsequently, the new matrix allows a CBP technique to encode itself into a compact and normalized bilinear representation. In order to further accelerate the normalization process, our approach has two variants which can handle feature matrix extracted by different networks. Tested on three fine-grained image classification datasets, it provides competitive classification performance while consuming less computational time than other prior works.**

***Index Terms*—Bilinear pooling, Matrix square-root, Fine-grained image classification, Efficiency, Newton iterations.**

I. Introduction

A. Bilinear pooling

Bilinear model, firstly introduced by Tenenbaum et al. in [1] for separating style and contents, has been extended to a non-parametric pooling layer to aggregate deep learning features in [2]. Starting from deep features extracted with a Convolutional Neural Network (CNN) and arranged in a matrix $\mathbf{X} \in \mathbb{R}^{D \times N}$, where N is the number of feature vectors, and D their dimension, the bilinear pooling layer aggregates the vectors in a $D \times D$ matrix as:

$$\mathbf{A} = \frac{1}{N}\mathbf{X}\mathbf{X}^T + \epsilon\mathbf{I} = (\frac{1}{\sqrt{N}}\mathbf{X})(\frac{1}{\sqrt{N}}\mathbf{X})^T + \epsilon\mathbf{I}. \quad (1)$$

where ϵ is a small positive value, e.g. 1e-7, added to diagonal values in matrix $\mathbf{A}$. Compared with average or max pooling, widely used in the CNN, bilinear pooling records richer second-order statistics between different feature dimensions [2]. Furthermore, it is an orderless pooling approach and is robust to spatial variance of the inputs. With these advantageous properties, bilinear pooling has shown its effectiveness for multiple computer vision tasks, such as texture synthesis [3], style transfer [4], segmentation [5] or visual question answering [6], [7]. In particular, bilinear pooling shows superior performance for fine-grained image classification [8] on which we concentrate in our experiments.

Following the pioneer work of [8], some works such as [9], [10] extended bilinear pooling for higher performance but those studies are out of our scope. Instead, in this paper, we focus on two improvements dedicated to the pooling itself, i.e. compact bilinear pooling and square root normalization.

B. Compact Bilinear pooling

Original bilinear representations are of cumbersome size. In [8], since the feature vector dimension D is equal to 512 for the VGG-16 network, $\mathbf{A} \in \mathbb{R}^{D \times D}$ has more than 250.000 elements. Such a big representation is not practical in many aspects. One of them is leading to a heavy classifier, containing numerous parameters to be trained. Even worse, for ResNet50 [11], D is equal to 2048. Therefore, a more compact bilinear representation should be considered. Gao et al. applied existing kernel approximations such as Random Maclaurin and Tensor Sketch, to produce compact bilinear representations [12]. They show that the size can be 32 times smaller while keeping almost identical performance. Kong and Fowlkes considered a low-rank bilinear SVM to run classification based on bilinear representations [13]. The key advantage of this approach is that it avoids to explicitly compute the bilinear representation.

C. Square root normalization

Besides the size issue, the representation $\mathbf{A}$ is a Symmetric Positive Definite (SPD) matrix. It lies in the Riemannian manifold and training a linear classifier in such a non-Euclidean space, as done by the original work [8], is clearly sub-optimal. In order to map the SPD matrix into a Euclidean space, multiple works [14], [15], [16] suggested normalizing $\mathbf{A}$ into its square-rooted matrix $\mathbf{A}^{1/2}$ so that $\mathbf{A} = \mathbf{A}^{1/2}\mathbf{A}^{1/2}$. Lin and Maji found that $\mathbf{A}^{1/2}$ can be approximated by a variant of Newton iterations which speeds up this normalization process on GPUs [15]. Li et al. further introduced Newton iterations into gradient propagation of the normalization for achieving a faster training process [16].

D. Combination

Compact bilinear pooling and square root normalization improve bilinear pooling in different manners but it is not trivial to combine them [17], [18]. To solve this combination

Sixiang Xu and Licheng Jiao are with the Key Laboratory of Intelligent Perception and Image Understanding of the Ministry of Education of China, International Research Center of Intelligent Perception and Computation, School of Artificial Intelligence, Xidian University, Xi'an 710071, China (e-mail: xusixiang@xidian.edu.cn, lchjiao@mail.xidian.edu.cn).

Damien Muselet and Alain Trémeau are with Université de Lyon, UJM-Saint-Etienne, CNRS, UMR5516, Laboratoire Hubert Curien, F-42023, Saint-Etienne, France (e-mail: damien.muselet, alain.tremeau@univ-st-etienne.fr).

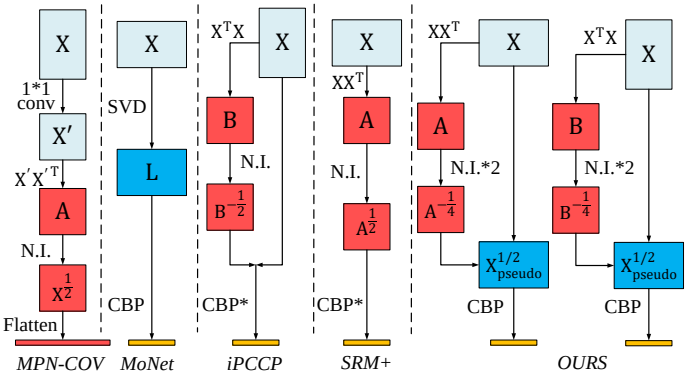

Fig. 1. Overview of the approaches combining Compact Bilinear Pooling (CBP) and square root normalization, e.g. Newton Iterations (N.I.). Input feature matrix X is $D \times N$, where D is the feature dimension and N, the number of feature vectors. CBP with '*' means that the considered approach uses its own CBP technique.

problem, there exist several works whose workflows and properties are respectively summarized in Fig. 1 and Table I.

Instead of using existing CBP techniques, *MPN-COV* [19] applies a 1*1 convolutional layer to reduce the feature dimension before bilinear pooling. The resultant compact representation is then normalized via Newton iterations, a fast way to obtain square root matrix from a SPD matrix. The problem of such an approach is that the convolutional layer for dimension reduction should be pre-trained (as well as the whole network) on a large scale datasets such as ILSVRC2012. Gou et al. [18] proposed an approach called *MoNet* that normalizes the feature vectors $\mathbf{X}$ into $\mathbf{L} \in \mathbb{R}^{D \times N}$ so that $\mathbf{A}^{1/2} = \mathbf{L}\mathbf{L}^T$. Then, the matrix $\mathbf{L}$ is transformed into a compact representation of $\mathbf{A}^{1/2}$ with an existing CBP techniques [12], [20]. Unfortunately, this approach needs a Singular Value Decomposition (SVD) computation which is not efficient in deep networks. In order to solve the SVD issue, *iPCCP* suggests to produce a compact and normalized bilinear representation via Newton iterations [21]. The authors calculated the square root of a Gram matrix, i.e. $\mathbf{X}^T\mathbf{X}$, and used it as an additional input in the following CBP module. However, this behavior is considered as one inconvenience of *iPCCP* because it needs a re-implementation of every existing CBP techniques. Moreover, *iPCCP* runs efficiently only when $N \ll D$. More recently, Yu et al. upgrade the classical CBP's Random Maclaurin (RM) with the Shifted Random Maclaurin (SRM) [20]. SRM needs smaller binary $(+1, -1)$ projection matrices and makes the CBP faster. To go one step further, the authors propose *SRM+* which is able to encode a compact normalized bilinear representation with SRM. However, it requires the processing of the matrix $\mathbf{A}^{1/2}$ which can't be derived efficiently from $\mathbf{A}$ when the feature dimension D is large.

As summarized in Table I, all these approaches partially solve the combination problem and there is still room to conceive a better solution. We need a solution that can use a general CBP module, without using any SVD and that is efficient both when $N \ll D$ and $N \gg D$. To this end, in this paper, we introduce a new normalized matrix: pseudo square-rooted matrix which can perfectly run with most kinds of existing CBP techniques and produce compact and normalized

TABLE I
BILINEAR POOLING VARIANTS OVERVIEW.

Approaches	General CBP	Square Root Norm. Efficiency		
		No SVD	N>>D	N<<D
MPN-COV		✓	✓	
MoNet	✓			
iPCCP		✓		✓
SRM+		✓	✓	
Ours	✓	✓	✓	✓

representation. To calculate our pseudo square-rooted matrix more efficiently, we leverage the normalization approach from [22] which was applied for Fisher score representation. Different from the original one, the new approach has two variants and is more flexible. It can hold its efficiency when the size of input features falls into two cases: $N \ll D$ and $N \gg D$. We test our approach on three fine-grained image classification datasets and three deep networks. The results show that our approach achieves competitive classification accuracy and runs more efficiently than other state-of-the arts. Lastly, we run an experiment to visualize the heatmap of the normalized matrix.

Our contributions are multiple:

- We propose a complete solution that combines compact bilinear pooling with matrix normalization.
- We propose two variants in order to reach efficiency in the two cases: $N \ll D$ and $N \gg D$. And the efficiency is confirmed in the later tests by comparing computation time with recent alternatives.
- We run extensive tests on three datasets with different architectures and different recent CBP techniques showing the flexibility of the proposed solution,

II. Our Approach

For simplicity, in the rest of the paper, we denote $\frac{1}{\sqrt{N}}\mathbf{X}$ as $\mathbf{X}$ and omit small values $\epsilon\mathbf{I}$, so that Eq. 1 is $\mathbf{A} = \mathbf{X}\mathbf{X}^T$. Given the matrix $\mathbf{X}$, its singular value decomposition (SVD) is expressed as:

$$\mathbf{X} = \mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T \tag{2}$$

where $\mathbf{Q}$ and $\mathbf{V}$ are unitary matrices and $\mathbf{\Sigma} \in \mathbb{R}^{D \times N}$ is a rectangular diagonal matrix. Inspired by [18], we look for a matrix $\mathbf{L}$ such as $\mathbf{A}^{1/2} = \mathbf{L}\mathbf{L}^T$, but we define it differently as:

$$\mathbf{L} = \mathbf{X}^{1/2}_{pseudo} = \mathbf{Q}\mathbf{\Sigma}^{1/2}_{pseudo}\mathbf{V}^T \tag{3}$$

where $\mathbf{\Sigma}^{1/2}_{pseudo}$ is calculated by square rooting the diagonal elements of $\mathbf{\Sigma}$. Since $\mathbf{X}$ is not a SPD matrix, there is no square rooted matrix $\mathbf{X}^{1/2}$. But like square root normalization, we scale its diagonal elements with square root and obtain a normalized matrix $\mathbf{X}^{1/2}_{pseudo}$. Hence, we call $\mathbf{X}^{1/2}_{pseudo}$ the **pseudo square-rooted matrix**.

Let us now explain how to efficiently obtain $\mathbf{X}^{1/2}_{pseudo}$ from $\mathbf{X} \in \mathbb{R}^{D \times N}$ for the two cases: $N \gg D$ and $N \ll D$. Since the fast Newton iterations can not be applied to non-SPD matrices, such as $\mathbf{X}$, we use smart alternatives by applying it two times to either $\mathbf{X}\mathbf{X}^T$ or $\mathbf{X}^T\mathbf{X}$.

A. Case 1 ($N \gg D$)

As illustrated in Fig 1 (Ours-left), we start by evaluating $\mathbf{A} = \mathbf{X}\mathbf{X}^T$. By using the SVD notations, we have:

$$\begin{aligned}\mathbf{A} &= \mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T(\mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T)^T = \mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\mathbf{\Sigma}^T\mathbf{Q}^T \\ &= \mathbf{Q}\mathbf{\Sigma}\mathbf{\Sigma}^T\mathbf{Q}^T = \mathbf{Q}\tilde{\mathbf{\Sigma}}\mathbf{H}\mathbf{H}^T\tilde{\mathbf{\Sigma}}^T\mathbf{Q}^T \\ &= \mathbf{Q}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{\Sigma}}^T\mathbf{Q}^T = \mathbf{Q}\tilde{\mathbf{\Sigma}}^2\mathbf{Q}^T,\end{aligned} \tag{4}$$

where a temporary matrix $\mathbf{H} = [\mathbf{I}_D|0] \in \mathbb{R}^{D\times N}$ ($\mathbf{I}_D$ is a $D \times D$ identity matrix) is introduced such that $\mathbf{\Sigma} = \tilde{\mathbf{\Sigma}}\mathbf{H}$. Then, we apply Newton iterations on the SPD matrix $\mathbf{A}$ and the outputs converge to $\mathbf{A}^{1/2} = \mathbf{Q}\tilde{\mathbf{\Sigma}}\mathbf{Q}^T$ and $\mathbf{A}^{-1/2}$ [23]. Then, we apply again Newton iterations on the input $\mathbf{A}^{1/2}$ to obtain $\mathbf{A}^{1/4}$ and $\mathbf{A}^{-1/4} = \mathbf{Q}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{Q}^T$. Finally, we have access to $\mathbf{X}^{1/2}_{pseudo}$ according to:

$$\begin{aligned}\mathbf{X}^{1/2}_{pseudo} &= \mathbf{A}^{-1/4}\mathbf{X} = \mathbf{Q}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{Q}^T\mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T \\ &= \mathbf{Q}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{\Sigma}\mathbf{V}^T = \mathbf{Q}\tilde{\mathbf{\Sigma}}^{-1/2}\tilde{\mathbf{\Sigma}}\mathbf{H}\mathbf{V}^T \\ &= \mathbf{Q}\tilde{\mathbf{\Sigma}}^{1/2}\mathbf{H}\mathbf{V}^T = \mathbf{Q}\mathbf{\Sigma}^{1/2}_{pseudo}\mathbf{V}^T\end{aligned} \tag{5}$$

B. Case 2 ($N \ll D$)

As illustrated in Fig 1 (Ours - right), we start by evaluating the Gram matrix $\mathbf{B} = \mathbf{X}^T\mathbf{X} \in \mathbb{R}^{N\times N}$ [24] which is equal to:

$$\begin{aligned}\mathbf{B} &= (\mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T)^T(\mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T) = \mathbf{V}\mathbf{\Sigma}^T\mathbf{Q}^T\mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T \\ &= \mathbf{V}\mathbf{\Sigma}^T\mathbf{\Sigma}\mathbf{V}^T = \mathbf{V}\tilde{\mathbf{\Sigma}}^T\mathbf{H}^T\mathbf{H}\tilde{\mathbf{\Sigma}}\mathbf{V}^T \\ &= \mathbf{V}\tilde{\mathbf{\Sigma}}^T\tilde{\mathbf{\Sigma}}\mathbf{V}^T = \mathbf{V}\tilde{\mathbf{\Sigma}}^2\mathbf{V}^T\end{aligned} \tag{6}$$

where $\mathbf{H} = [\mathbf{I}_N|0]^T \in \mathbb{R}^{D\times N}$ and $\mathbf{\Sigma} = \mathbf{H}\tilde{\mathbf{\Sigma}}$. Similarly as Case 1, after two consecutive Newton iterations, we obtain $\mathbf{B}^{-1/4} = \mathbf{V}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{V}^T$ and evaluate $\mathbf{X}^{1/2}_{pseudo}$ as:

$$\begin{aligned}\mathbf{X}^{1/2}_{pseudo} &= \mathbf{X}\mathbf{B}^{-1/4} = \mathbf{Q}\mathbf{\Sigma}\mathbf{V}^T\mathbf{V}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{V}^T \\ &= \mathbf{Q}\mathbf{\Sigma}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{V}^T = \mathbf{Q}\mathbf{H}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{\Sigma}}^{-1/2}\mathbf{V}^T \\ &= \mathbf{Q}\mathbf{H}\tilde{\mathbf{\Sigma}}^{1/2}\mathbf{V}^T = \mathbf{Q}\tilde{\mathbf{\Sigma}}^{1/2}_{pseudo}\mathbf{V}^T\end{aligned} \tag{7}$$

In both cases, Newton iterations can always process a smaller SPD matrix and thus, speed up the normalization step. Furthermore, in the two cases, $\mathbf{X}^{1/2}_{pseudo}\mathbf{X}^{1/2}_{pseudo}{}^T = \mathbf{Q}\tilde{\mathbf{\Sigma}}\mathbf{Q}^T = \mathbf{A}^{1/2}$. Hence, like MoNet [18], applying CBP techniques to $\mathbf{X}^{1/2}_{pseudo}$ can output a compact bilinear representation which approximates $\mathbf{A}^{1/2}$.

III. Experiments

A. Datasets

We run experiments on three fine-grained image classification benchmarks: Caltech-UCSD Birds-200-2011 [25] (Bird), FGVC-Aircraft Benchmark [26] (Aircraft) and Stanford Cars Dataset [27] (Car). The Bird dataset provides 200 bird species images where 5994 images compose the training dataset and 5794 images are used as test dataset. The Aircraft dataset contains 100 aircraft variant models with 67 training images and 33 test images per category. The Car dataset is composed of 16185 images of 196 classes of cars. Following a roughly 50-50 split per class, there are 8144 training images and 8041 test images. Note that we always make use of official training-test splits released along with the three datasets.

TABLE II
Comparison of the classification accuracy (%) with alternatives using VGG and ResNet backbones. In each column, the best result is color coded: Red means there exists a statistical significance (p-value< 0.05) with the second best of the column and Blue means there is no significance.

Approaches	Dimensions		Bird		Aircraft		Car	
	VGG	ResNet	VGG	ResNet	VGG	ResNet	VGG	ResNet
BCNN	262K	-	84.0	-	86.9	-	90.6	-
iBCNN	262K	-	85.8	-	88.5	-	92.1	-
Compact	8K	-	84.0	-	87.2	-	90.2	-
MoNet	10K	-	85.7	-	86.7	-	90.3	-
SRM+	8K	-	85.5	-	**90.5**	-	-	-
MPN-COV	32K	32K	86.1	86.7	88.9	89.6	91.8	92.2
Ours (TS)	8K	8K	**86.3**	**87.0**	90.0	**92.3**	**92.3**	**94.1**

TABLE III
Comparison of the classification accuracy (%) with alternatives using the iSQRT backbone. Color blue: no statistical significance between our best approach and the best alternative.

Approaches	Dimensions	Bird	Aircraft	Car
MoNet (TS)	8K	87.3	90.9	94.0
MPN-COV	32K	**88.1**	90.0	92.8
iPCCP (TS)	8K	87.3	90.5	93.7
Ours (TS)	8K	88.0	**91.0**	**94.1**
Ours (RM)	8K	87.9	90.9	94.1
Ours (SRM)	8K	88.0	90.9	94.0

B. Experimental settings

Models We use VGG-16 (VGG) and ResNet-50 (ResNet) pretrained on ILSVRC2012 [28] and extract feature vectors $\mathbf{X}$ after the layer $relu5\_3$ in the VGG and before the global average pooling in ResNet-50. Li et al. also trained their own ResNet-50 on ILSVRC2012, called iSQRT, where global average pooling is replaced with MPN-COV [16]. Following the implementation in [21], we extract local features $\mathbf{X} \in \mathbb{R}^{256\times 784}$ after the last convolutional layer of this network and center each feature vector $\mathbf{x}_i \in \mathbf{X}$ to a zero mean: $\mathbf{x}_i - \frac{1}{N}\left(\sum_{i=1}^{N}\mathbf{x}_i\right)$ before operating our pooling. Depending on the experiments, we provide results for our method with different recent CBP techniques: Tensor Sketch (TS), Random Maclaurin (RM) and Shifted Random Maclaurin (SRM). For square root normalization process, we always use 5 Newton iterations.

Training and evaluation During training and evaluation, we adopt the same image pre-processings as [16], [21] for fair comparison. On the Bird and Car datasets, we resize input images to 448×488 for training and evaluation. For the Aircraft dataset, a center crop of 448×488 from a resized image of 512×512 is considered as input image. The only data augmentation applied during training is a random horizontal flip. At test time, we either average the predictions on the input image and on its flipped version or consider the results on the input image as final prediction. If not specified, all the results are averaged over 5 runs.

For the experiments with iSQRT backbone, we follow the fine-tuning strategy from [16], [21] where the optimization algorithm corresponds to a stochastic gradient descent with a mini-batch size of 10, a weight decay of 0.001 and a momentum of 0.9. The learning rate for the classifier is $6e^{-3}$ and for the rest of layers is $1.2e^{-3}$. All the experiments stop after 100 epochs.

For the experiments with ResNet-50 and VGG-16, like other

TABLE IV
COMPARISON OF TRAINING/EVALUATION COMPUTATION TIME (SECONDS) FOR ONE MINI BATCH ON RTX3090

Approaches	N	D	Batch(Train./Val.)	TS	RM	SRM
SRM+*	784	256	10/20	-	-	**0.13/0.08**
MoNet*				0.34/0.40	0.28/0.30	0.30/0.32
iPCCP*				-	0.18/0.10	0.17/0.09
Ours*				**0.16/0.09**	**0.15/0.08**	0.14/0.08
SRM+	196	2048	32/64	-	-	0.68/0.38*
MoNet				1.11/1.20	0.91/1.25	0.91/1.22
iPCCP				-	**0.34/0.22**	**0.32/0.20**
Ours				**0.33/0.21**	**0.34/0.22**	**0.32/0.20**

* *Due to the limit of GPU memory space, mini batch size is 16/32.*

TABLE V
IMPACT OF THE NUMBER OF NEWTON ITERATIONS OR THE SVD. RESULTS ARE BASED ON 1-TIME RUN.

Network	Metric	N_iterations			SVD
		1	5	10	
iSQRT	Accuracy	87.3	91.0	90.9	91.2
	Time	0.0925	0.0935	0.0955	0.402
VGG16	Accuracy	88.2	90.4	90.5	90.4
	Time	0.424	0.434	0.467	2.001
ResNet50	Accuracy	90.8	92.1	92.1	92.2
	Time	0.208	0.209	0.215	1.203

approaches [15], [18], [2], [20], the training phase is composed of 2 steps. In the first step, we only train newly-added classifier for 100 epochs with learning rate of 1, a mini-batch size of 32, weight decay of $1e^{-5}$ and a momentum of 0.9 while parameters in the rest of layers are frozen. Then, we finetune the whole network for another 100 epochs with learning rate of $2.7e^{-2}$ and weight decay of $1e^{-4}$. For both steps, learning rate is divided by 10 when the training loss is not smaller than 0.99 times actual minimum value for 10 epochs.

C. Results

Tables II and III display classification accuracy and statistical significance of our approach with the three backbones as well as the results provided in the reference papers for comparison. **Note that the highest accuracy is not our goal. The comparison with other methods is meant to prove that our approach is compatible with various backbones and shows competitive performance.**

Compatibility with different backbones According to Table II, our method overall outperforms the classical bilinear polling (noted BCNN) and compact bilinear pooling (noted Compact) thanks to the normalization step. For other related works which produce compact and normalized bilinear representations, MoNet works poorly with VGG-16. We suspect that it is due to the use of a less appropriate data augmentation strategy [18]. Yu et al. reported results for SRM+ only with a VGG backbone [20]. A probable reason is that SRM+ is not efficient enough to be used with high dimensional features such as those from ResNet-50 (see Table IV). The poor performance of MPNCOV is probably due to the high number of layers (dimension reduction + classifier) that require more training data to avoid overfitting. iPCCP does not have any reported accuracy on these datasets. Table III provides the results with the iSQRT backbone. Our method obtains similar or better results than alternatives based on this backbone.

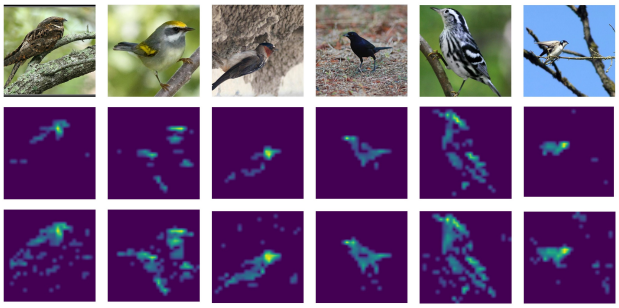

Fig. 2. $L2$-norm of the features before and after normalization. Top: input image. Middle: Heatmap before normalization. Bottom: Heatmap after normalization. Lighter regions in the heatmaps indicate more relevant features for the final classification decision. Best viewed in color.

Compatibility with CBP techniques In table III, we test our method with different CBP solutions (TS, RM and SRM), showing that our approach can work perfectly with a wide range of CBP techniques.

Efficiency To show efficiency of related works which produce compact and normalized bilinear representations, we measure computation time to run training/evaluation for one mini batch with three CBP techniques (see Table IV). ResNet-50 and iSQRT backbones are selected because their feature vectors before bilinear pooling respectively satisfy the two studied cases: $N \ll D$ and $N \gg D$.

When $N \ll D$, MoNet and SRM+ are much less efficient because the former runs the SVD and the later feeds Newton iterations with big SPD matrices of size 2048×2048. On the contrary, our solution and iPCCP avoid the SVD computation and the input matrix is much smaller (196×196).

When $N \gg D$, our approach and SRM+ are clearly more efficient because, for iPCCP, the size of SPD matrices before Newton iterations are 784×784 while our approach and SRM+ are using smaller SPD matrices (256×256).

We notice that the previous approaches are efficient in a single case at most, while our approach can handle both cases with its flexible structure, because it satisfies the three efficiency elements reported in Table I.

Number of Newton Iterations. As shown in the Table V, 5 iterations keep good balance between running efficiency and classification performance. SVD can also produce plausible accuracy, but at the cost of low efficiency.

Visualization In Figure 2, we visualize the $L2$-norm of the features before ($\mathbf{X}$) and after ($\mathbf{X}^{1/2}_{pseudo}$) normalization for some images. These illustrations suggest that after our normalization, more discriminative parts are discovered since corresponding feature vectors are enhanced. With more detail features, the next bilinear representation is more powerful.

IV. CONCLUSION

We propose an approach to efficiently combine compact bilinear pooling and square root normalization, considered as two essential elements to improve bilinear pooling. Inspired by the previous approach MoNet, our approach can efficiently normalize input feature vectors into a pseudo square-root matrix. The normalized matrix can then be encoded by most of the existing CBP techniques. Through the tests, our approach shows strong compatibility with different deep networks and needs less computation time than recent alternatives.

REFERENCES

[1] J. B. Tenenbaum and W. T. Freeman, "Separating style and content with bilinear models," *Neural computation*, vol. 12, no. 6, pp. 1247–1283, 2000.

[2] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear convolutional neural networks for fine-grained visual recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1309–1322, 2017.

[3] L. A. Gatys, A. S. Ecker, and M. Bethge, "Texture synthesis using convolutional neural networks," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, Eds., 2015, pp. 262–270. [Online]. Available: https://proceedings.neurips.cc/paper/2015/hash/a5e00132373a7031000fd987a3c9f87b-Abstract.html

[4] ——, "Image style transfer using convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2414–2423.

[5] C. Ionescu, O. Vantzos, and C. Sminchisescu, "Matrix backpropagation for deep networks with structured layers," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2965–2973.

[6] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, "Multimodal compact bilinear pooling for visual question answering and visual grounding," *arXiv preprint arXiv:1606.01847*, 2016.

[7] Z. Yu, J. Yu, J. Fan, and D. Tao, "Multi-modal factorized bilinear pooling with co-attention learning for visual question answering," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1821–1830.

[8] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1449–1457.

[9] Q. Wang, P. Li, and L. Zhang, "G2denet: Global gaussian distribution embedding network and its application to visual recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2730–2739.

[10] Y. Cui, F. Zhou, J. Wang, X. Liu, Y. Lin, and S. Belongie, "Kernel pooling for convolutional neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2921–2930.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[12] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, "Compact bilinear pooling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 317–326.

[13] S. Kong and C. Fowlkes, "Low-rank bilinear pooling for fine-grained classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 365–374.

[14] I. L. Dryden, A. Koloydenko, and D. Zhou, "Non-euclidean statistics for covariance matrices, with applications to diffusion tensor imaging," *The Annals of Applied Statistics*, vol. 3, no. 3, pp. 1102–1123, 2009.

[15] T.-Y. Lin and S. Maji, "Improved bilinear pooling with cnns," in *Proceedings of the British Machine Vision Conference(BMVC)*, G. B. Tae-Kyun Kim, Stefanos Zafeiriou and K. Mikolajczyk, Eds. BMVA Press, September 2017, pp. 117.1–117.12. [Online]. Available: https://dx.doi.org/10.5244/C.31.117

[16] P. Li, J. Xie, Q. Wang, and Z. Gao, "Towards faster training of global covariance pooling networks by iterative matrix square root normalization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 947–955.

[17] T.-Y. Lin, S. Maji, and P. Koniusz, "Second-order democratic aggregation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 620–636.

[18] M. Gou, F. Xiong, O. Camps, and M. Sznaier, "Monet: Moments embedding network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3175–3183.

[19] Q. Wang, J. Xie, W. Zuo, L. Zhang, and P. Li, "Deep cnns meet global covariance pooling: Better representation and generalization," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[20] T. Yu, X. Li, and P. Li, "Fast and compact bilinear pooling by shifted random maclaurin," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 4, 2021, pp. 3243–3251.

[21] Y. Mukuta, T. Machida, and T. Harada, "Compact approximation for polynomial of covariance feature," *arXiv preprint arXiv:1906.01851*, 2019.

[22] S. Xu, D. Muselet, and A. Trémeau, "Sparse coding and normalization for deep fisher score representation," *Computer Vision and Image Understanding*, p. 103436, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1077314222000571

[23] T.-Y. Lin and S. Maji, "Improved bilinear pooling with cnns," *arXiv preprint arXiv:1707.06772*, 2017.

[24] A. Ali, H. Touvron, M. Caron, P. Bojanowski, M. Douze, A. Joulin, I. Laptev, N. Neverova, G. Synnaeve, J. Verbeek *et al.*, "Xcit: Cross-covariance image transformers," *Advances in neural information processing systems*, vol. 34, 2021.

[25] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The Caltech-UCSD Birds-200-2011 Dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[26] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," Tech. Rep., 2013.

[27] J. Krause, M. Stark, J. Deng, and L. Fei-Fei, "3d object representations for fine-grained categorization," in *Proceedings of the IEEE international conference on computer vision workshops*, 2013, pp. 554–561.

[28] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, no. 3, pp. 211–252, 2015.