A Deeper Look at Power Normalizations

Piotr Koniusz[*,1,2] Hongguang Zhang[*,2,1] Fatih Porikli[2]
[1]Data61/CSIRO, [2]Australian National University
firstname.lastname@{data61.csiro.au[1], anu.edu.au[2]}

Abstract

Power Normalizations (PN) are very useful non-linear operators in the context of Bag-of-Words data representations as they tackle problems such as feature imbalance. In this paper, we reconsider these operators in the deep learning setup by introducing a novel layer that implements PN for non-linear pooling of feature maps. Specifically, by using a kernel formulation, our layer combines the feature vectors and their respective spatial locations in the feature maps produced by the last convolutional layer of CNN. Linearization of such a kernel results in a positive definite matrix capturing the second-order statistics of the feature vectors, to which PN operators are applied. We study two types of PN functions, namely (i) MaxExp and (ii) Gamma, addressing their role and meaning in the context of non-linear pooling. We also provide a probabilistic interpretation of these operators and derive their surrogates with well-behaved gradients for end-to-end CNN learning. We apply our theory to practice by implementing the PN layer on a ResNet-50 model and showcase experiments on four benchmarks for fine-grained recognition, scene recognition, and material classification. Our results demonstrate state-of-the-part performance across all these tasks.

1. Introduction

Second-order statistics of data features have played a pivotal role in advancing the state of the art on several problems in computer vision, including object recognition, texture categorization, action representation, and human tracking, to name a few of applications [54, 47, 58, 38, 16, 9, 34]. For example, in the popular region covariance descriptors [54], a covariance matrix, which is computed over multi-modal features from image regions, is used as an object representation for recognition and tracking, and has been extended to several other applications [54, 47, 58, 38, 16]. Given Bag-of-Words histograms or local descriptor vectors from an image, a second-order co-occurrence pooling of these vectors captures the occurrences of two features together. Such a strategy has been recently shown to result in a superior performance in semantic segmentation and visual concept detection, compared to their first-order counterparts [9, 33, 34]. A natural extension led to higher-order pooling operators [33, 34, 30] on third-order supersymmetric tensors which improve results over the second-order descriptors over 7% MAP on PASCAL VOC07.

However, second and higher-order statistics require appropriate aggregation and pooling mechanisms to obtain the highest classification results [9, 33, 34]. Once the statistics are captured in the matrix form, they undergo next a non-linearity such as Power Normalization [35] which role is to reduce/boost contributions from frequent/infrequent visual stimuli in an image, respectively. A significant progress made by the Bag-of-Words model provides numerous insights into the role played by pooling during the aggregation step. The theoretical relation between Average and Max-pooling was studied in [7]. A detailed likelihood-based analysis of feature pooling was conducted in [8] which led to a *theoretical expectation of Max-pooling*, improving overall classification results. Power Normalization has also been applied to Average pooling by Fisher Kernels [46]. Max-pooling has been recognized as a lower bound of the likelihood of *'at least one particular visual word being present in an image'* [42]. According to an evaluation [35], these pooling methods are all closely related. However, evaluations [35] do not consider the second-order pooling scenario or end-to-end learning. In the context of second-order pooling, element-wise and eigenvalue Power Normalization (ePN) were both first proposed in [33] in 2013.

In this paper, we aim to revisit the above pooling methods in end-to-end setting and shed further light on their interpretation in the context of second-order matrices. Firstly, we propose a kernel formulation which combines feature vectors collected from the last convolutional layer of ResNet-50 together with so-called spatial location vectors, previously explored in [31, 35, 33] around 2011–2013, which contain spatial locations corresponding to feature vectors in the CNN feature maps. A linearization of such a kernel results in a second-order matrix which contains ag-

[*]Both authors contributed equally.

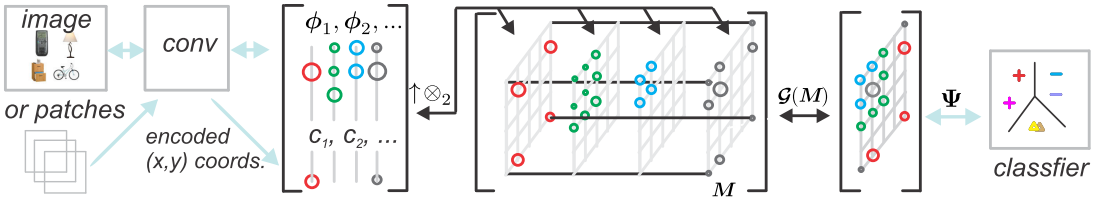

Figure 1: Our end-to-end pipeline. We pass an image (or patches) to CNN and extract feature vectors ϕ from its last conv. layer and augment them by encoded spatial coordinates $\mathbf{c}$. We perform pooling on second-order matrix $\boldsymbol{M}$ by the Power Normalization function $\boldsymbol{\mathcal{G}}$.

gregated second-order statistics of these combined vectors. Subsequently, we focus on the role of the Power Normalization family in end-to-end setting. We show that these functions have a well-founded probabilistic interpretation in the context of second-order statistics. Moreover, we propose PN surrogates which have well-behaved derivatives suitable for back-propagation unlike typical PN functions.

Our contributions are three-fold: (i) we propose to aggregate feature vectors extracted from CNNs and their spatial coordinates into a second-order matrix by principled derivations in end-to-end manner, (ii) we revisit Power Normalization functions, derive them for second-order representations and show that they follow Binomial or Multinomial distributions if features are drawn from the Brenoulli distribution, (iii) we propose PN surrogates with well-behaved derivatives for end-to-end learning, (iv) we propose new spectral variants of pooling. Figure 1 shows our pipeline.

We perform evaluations on ResNet-50 and four image classification benchmarks such as Flower102, MIT67, FMD and Food101 where we demonstrate state-of-the-art results.

2. Related Work

Second-order statistics have been extensively studied in the context of texture recognition [54, 55, 49] by the use of so-called Region Covariance Descriptors (RCD).

Region Covariance Descriptors (RCD). Such methods use a representation which typically captures co-occurrences of luminance, first- and/or second-order derivatives of texture patterns. Alternatively, co-occurrences in Local Binary Patterns (LBP) are captured to build second-order matrices [49]. RCD approaches have also been successfully applied to tracking [47], semantic segmentation [9] and object category recognition [34], to name but a few of applications. The design of RCD typically requires a decision on what signals need to be aggregated into the second-order representation and how to compare positive (semi-)definite datapoints resulting from such an aggregation step. There exist several non-Euclidean distances often applied to positive definite matrices which we list next.

Non-Euclidean distances. The distance between two positive definite datapoints is typically measured according to the Riemannian geometry while Power-Euclidean distances [14] extend to positive semi-definite distances. In particular, Affine-Invariant Riemannian Metric [45, 4], KL-Divergence Metric (*KLDM*) [59], Jensen-Bregman LogDet Divergence (*JBLD*) [10] and Log-Euclidean (*LogE*) [2] have been used in the context of diffusion imaging and the RCD-based methods. Dictionary and metric learning methods also use non-Euclidean distances [17, 18, 19, 37, 20].

Our approach differs in that we perform end-to-end learning in the CNN setting while RCD and dictionary learning constitute shallow architectures that perform worse than CNNs on the majority of classification tasks.

We note that the Log-Euclidean distance and Power Normalization have been implemented in the CNN setting [25, 24, 39, 41] for the purpose of region classification. These methods employ back-propagation which requires costly eigenvalue decomposition for computations of derivatives deeming them computationally inefficient. Note that the cost of a single eigenvalue decomposition is at least $\mathcal{O}(d^\omega)$, where constant $2<\omega<2.376$[1]. The typical bottleneck in using non-Euclidean distances in end-to-end setting lies in their costly back-propagation rules.

Our work differs in that we make an i.i.d. assumption on our co-occurrence features in our second-order representation. Thus, we require only element-wise rather than spectral operations. This reduces the complexity and relies on trivial arithmetic operations easy to implement on GPU.

Pooling and CNNs. There exist several approaches for image retrieval and recognition which perform some form of aggregation over first-order statistics extracted from the CNN maps *e.g.*, [15, 61, 1]. In [15], the authors propose to extract multiple regions from an image and aggregate CNN responses into an image representations. In [61], the authors aggregate local deep features for the task of image retrieval. In [1], the authors extend Vector of Locally Aggregated Descriptors (VLAD) to an end-to-end trainable system.

Our approach differs in that we use co-occurrences in end-to-end setting and take an analytical look at how to interpret Power Normalization functions in this setting.

There has been also a revived interest in creating co-occurrence patterns in CNN setting similar in spirit to RCD. Approach [40] applies a fusion of two CNN streams via outer product in the context of the fine-grained image recognition. Another approach for face recognition [23] uses co-occurrences of CNN feature vectors and facial attribute vec-

[1]We assume that the eigenvalue decomposition of large matrices ($d=4096$) in CUDA BLAS is fast and efficient–which is not the case.

tors to obtain state-of-the-art face recognition results. A recent approach [52] extracts feature vectors at two separate locations in feature maps and performs an outer product to form a CNN co-occurrence layer.

In contrast to these papers, we use symmetric positive (semi-)definite matrices rather than negative definite ones.

Power Normalizations. Practical image representations have to deal with the so-called burstiness which is '*the property that a given visual element appears more times in an image than a statistically independent model would predict*' [28]. Power Normalization [6, 46, 28] is known to suppress this burstiness and has been extensively studied and evaluated in the context of Bag-of-Words [35, 34]. The theoretical relation between Average and Max-pooling was studied in [7] which highlighted the underlying statistical reasons for the superior performance of Max-pooling compared to a mere average of feature vectors. An analysis of feature pooling was conducted in [8] under specific assumptions on distributions from which the aggregated features are drawn. A relationship between the likelihood of '*at least one particular visual word being present in an image*' and Max-pooling was studied in [42]. According to a survey [35], these Power Normalization functions are closely related.

We take a similar view on PN functions, however, we devise an end-to-end trainable CNN layer and derive new pooling functions with well-behaved derivatives. We follow theoretical foundations of the Power Normalization family.

3. Background

Below we review our notations and the background on kernel linearizations and the Power Normalization family.

3.1. Notations

Let $\mathbf{x} \in \mathbb{R}^d$ be a d-dimensional feature vector. Then we use $\boldsymbol{\mathcal{X}} = \uparrow\otimes_r \mathbf{x}$ to denote the r-mode super-symmetric rank-one tensor $\boldsymbol{\mathcal{X}}$ generated by the r-th order outer-product of $\mathbf{x}$, where the element of $\boldsymbol{\mathcal{X}} \in \mathfrak{S}^d_{\times^r}$ at the $(i_1, i_2, ..., i_r)$-th index is given by $\Pi_{j=1}^r x_{i_j}$. $\mathcal{I}_N$ stands for the index set $\{1, 2, ..., N\}$. The spaces of symmetric positive semidefinite and definite matrices are $\mathcal{S}^d_+$ and $\mathcal{S}^d_{++}$. Moreover, $\mathrm{Sym}(\mathbf{X}) = \frac{1}{2}(\mathbf{X}+\mathbf{X}^T)$. A vector with all coefficients equal one is denoted by $\mathbb{1}$, $\boldsymbol{j}_m$ is a vector of all zeros except for the m-th coefficient which is equal one, and $\mathbf{J}_{mn}$ is a matrix of all zeros with a value of one at the position (m, n). Moreover, $\odot$ is the Hadamard product (element-wise multiplication). We use the MATLAB notation $\mathbf{v} = [\text{begin} : \text{step} : \text{end}]$ to generate a vector $\mathbf{v}$ with elements starting as *begin*, ending as *end*, with stepping equal *step*. Operator ';' in $[\mathbf{x}; \mathbf{y}]$ denotes the concatenation of vectors $\mathbf{x}$ and $\mathbf{y}$ (or scalars).

3.2. Kernel Linearization

In the sequel, we will use kernel feature maps detailed below to embed (x, y) locations of feature vectors extracted from conv. CNN maps at (x, y) into a non-linear Hilbert space. Such locations are called *spatial coordinates* [31, 34].

Proposition 1. *Let $G_\sigma(\mathbf{x}-\mathbf{y}) = \exp(-\|\mathbf{x}-\mathbf{y}\|_2^2/2\sigma^2)$ denote a Gaussian RBF kernel centered at $\mathbf{y}$ and having a bandwidth σ. Kernel linearization refers to rewriting G_σ as an inner-product of two (in)finite-dimensional feature maps which we obtain via probability product kernels [26]. Specifically, we employ the inner product of d'-dimensional isotropic Gaussians given $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{d'}$ as follows:*

$$G_\sigma(\mathbf{x}-\mathbf{y}) = \left(\frac{2}{\pi\sigma^2}\right)^{\frac{d'}{2}} \int_{\boldsymbol{\zeta}\in\mathbb{R}^{d'}} G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta})\, G_{\sigma/\sqrt{2}}(\mathbf{y}-\boldsymbol{\zeta})\, \mathrm{d}\boldsymbol{\zeta}. \quad (1)$$

Eq. (1) can be approximated by replacing the integral with the sum over Z pivots $\boldsymbol{\zeta}_1, ..., \boldsymbol{\zeta}_Z$. Thus, we obtain:

$$\boldsymbol{\varphi}(\mathbf{x}) = \left[G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta}_1), ..., G_{\sigma/\sqrt{2}}(\mathbf{x}-\boldsymbol{\zeta}_Z)\right]^T, \quad (2)$$

$$\text{and } G_\sigma(\mathbf{x}-\mathbf{y}) \approx \left\langle \sqrt{c}\boldsymbol{\varphi}(\mathbf{x}), \sqrt{c}\boldsymbol{\varphi}(\mathbf{y}) \right\rangle, \quad (3)$$

where c is a constant. We refer to (2) as a (kernel) feature map[3] and to (3) as the linearization of the RBF kernel.

Proof. The Gaussian kernel can be rewritten as a probability product kernel. See [26] (Section 3.1) for derivations. □

3.3. Second- and Higher-order Tensors

Below we show that second- or higher-order tensors emerge from a linearization of sum of Polynomial kernels.

Proposition 2. *Let $\boldsymbol{\Phi}_A \equiv \{\boldsymbol{\phi}_n\}_{n\in\mathcal{N}_A}$, $\boldsymbol{\Phi}_B \equiv \{\boldsymbol{\phi}^*_n\}_{n\in\mathcal{N}_B}$ be datapoints from two images Π_A and Π_B, and $N = |\mathcal{N}_A|$ and $N^* = |\mathcal{N}_B|$ be the numbers of data vectors* e.g.*, obtained from the last convolutional feature map of CNN for images Π_A and Π_B. Tensor feature maps result from a linearization of the sum of Polynomial kernels of degree r:*

$$K(\boldsymbol{\Phi}_A, \boldsymbol{\Phi}_B) = \langle \boldsymbol{\Psi}(\boldsymbol{\Phi}_A), \boldsymbol{\Psi}(\boldsymbol{\Phi}_B) \rangle = \quad (4)$$

$$\frac{1}{NN^*} \sum_{n\in\mathcal{N}_A} \sum_{n'\in\mathcal{N}_B} \langle \boldsymbol{\phi}_n, \boldsymbol{\phi}^*_{n'} \rangle^r \text{ where } \boldsymbol{\Psi}(\boldsymbol{\Phi}) = \frac{1}{N} \sum_{n\in\mathcal{N}} \uparrow\otimes_r \boldsymbol{\phi}_n.$$

Proof. See [32] for the details of such an expansion. □

Remark 1. *In what follows, we will use second-order matrices obtained from the above expansion for $r = 2$, that is:*

$$\frac{1}{NN^*} \sum_{n\in\mathcal{N}_A} \sum_{n'\in\mathcal{N}_B} \langle \boldsymbol{\phi}_n, \boldsymbol{\phi}^*_{n'} \rangle^2 = \left\langle \frac{1}{N} \sum_{n\in\mathcal{N}_A} \boldsymbol{\phi}_n \boldsymbol{\phi}_n^T, \frac{1}{N^*} \sum_{n\in\mathcal{N}_B} \boldsymbol{\phi}^*_{n'} \boldsymbol{\phi}^{*T}_{n'} \right\rangle. \quad (5)$$

Thus, we obtain the following (kernel) feature map[3]:

$$\boldsymbol{\Psi}\left(\{\boldsymbol{\phi}_n\}_{n\in\mathcal{N}}\right) = \boldsymbol{\mathcal{G}}\left(\frac{1}{N} \sum_{n\in\mathcal{N}} \boldsymbol{\phi}_n \boldsymbol{\phi}_n^T\right), \quad (6)$$

[3]Note that (kernel) feature maps are not conv. CNN maps. They are two separate notions that happen to share the same name.

where $\boldsymbol{\mathcal{G}}(\mathbf{X}) = \mathbf{X}$ will be later replaced by various Power Normalization functions.

3.4. Power Normalization Family

Max-pooling [7] can be derived by drawing features from the Bernoulli distribution under the i.i.d. assumption [8] which leads to so-called *Theoretical Expectation of Max-pooling* (*MaxExp*) operator [35] detailed below.

Proposition 3. *Assume a vector $\boldsymbol{\phi} \in \{0,1\}^N$ which stores N outcomes of drawing from Bernoulli distribution under the i.i.d. assumption for which the probability p of an event $(\phi_n = 1)$ and $1-p$ for $(\phi_n = 0)$ can be estimated as an expected value* e.g., *$p = \text{avg}_n\, \phi_n$. Then the probability of at least one positive event in $\boldsymbol{\phi}$ from N trials becomes:*

$$\psi = 1-(1-p)^N. \tag{7}$$

Proof. The proof follows the school syllabus for a fair coin toss. The probability of all N outcomes to be $\{(\phi_1 = 0), ..., (\phi_N = 0)\}$ amounts to $(1-p)^N$. The probability of at least one positive outcome $(\phi_n = 1)$ amounts to applying the logical 'or' $\{(\phi_1 = 1)\,|...|\,(\phi_N = 1)\}$ and leads to:

$$1-(1-p)^N = \sum_{n=1}^{N} \binom{N}{n} p^n (1-p)^{N-n}. \tag{8}$$

□

Remark 2. *A practical implementation of this pooling strategy [35] is given by $\psi_k = 1-(1-\text{avg}_n\, \phi_{kn})^\eta$, where $0 < \eta \approx N$ is an adjustable parameter and ϕ_{kn} is a k-th feature of an n-th feature vector* e.g., *as defined in Prop. 2, which is normalized to range 0–1.*

Remark 3. *It was shown in [35] that Power Normalization (Gamma) given by $\psi_k = (\text{avg}_n\, \phi_{kn})^\gamma$, where $0 < \gamma \leq 1$ is an adjustable parameter, is in fact an approximation of MaxExp.*

4. Problem Formulation

We start by devising our co-occurrence and pooling layers. We show that the Power Normalization (*Gamma*) has an ill-behaved derivative. Thus, we generalize MaxExp and Gamma [35, 34] to Logistic a.k.a. Sigmoid (*SigmE*) and the Arcsin hyperbolic (*AsinhE*) functions.

4.1. Co-occurrence matrix

As in Prop. 2, assume that datapoints $\boldsymbol{\Phi}_A \equiv \{\boldsymbol{\phi}_n\}_{n\in\mathcal{N}_A}$ and $\boldsymbol{\Phi}_B \equiv \{\boldsymbol{\phi}^*_n\}_{n\in\mathcal{N}_B}$ from two images $\mathit{\Pi}_A$ and $\mathit{\Pi}_B$ are given, $N = |\mathcal{N}_A|$ and $N^* = |\mathcal{N}_B|$ are the numbers of data vectors obtained from the last convolutional feature map of CNN for images $\mathit{\Pi}_A$ and $\mathit{\Pi}_B$. Moreover, assume that all $\boldsymbol{\phi}$ and $\boldsymbol{\phi}^*$ are rectified *e.g.*, $\boldsymbol{\phi}_n := \max(0, \boldsymbol{\phi}_n)$, $\boldsymbol{\phi}^*_n := \max(0, \boldsymbol{\phi}^*_n)$, and subsequently β-centered w.r.t. the means $\boldsymbol{\mu} = \text{avg}_{n\in\mathcal{N}_A}\, \boldsymbol{\phi}_n$ and $\boldsymbol{\mu}^* = \text{avg}_{n\in\mathcal{N}_B}\, \boldsymbol{\phi}^*_n$ so that $\boldsymbol{\phi}_n := \boldsymbol{\phi}_n - \beta\boldsymbol{\mu}$ and $\boldsymbol{\phi}^*_n := \boldsymbol{\phi}^*_n - \beta\boldsymbol{\mu}^*$ for $0 \leq \beta \leq 1$.

The role of β-centering is to address anti-occurrences. Specifically, sophisticated models of Bag-of-Words utilize so-called negative visual words which are the evidence of lack of a given visual stimulus in an image. For instance, the authors of [27] define it as '*the negative evidence, i.e., a visual word that is mutually missing in two descriptions being compared*'. Lack of certain visual stimuli may correlate with certain visual classes *e.g.*, lack of the sky may imply an indoor scene. Thus, the role of β is to offset vectors $\boldsymbol{\phi}$ by their per-image averages $\boldsymbol{\mu}$ so that the positive/negative values yield correlations/anti-correlations, respectively.

Next, let $x_n := x_n/(W-1)$ and $y_n := y_n/(H-1)$ be spatial coordinates normalized w.r.t. the width W and height H of conv. feature maps. We form the following kernel and its linearization by the use of Proposition 1:

$$\begin{aligned}&\langle \alpha\boldsymbol{\varphi}(x_n, \boldsymbol{\zeta}), \alpha\boldsymbol{\varphi}(x^*_{n'}, \boldsymbol{\zeta})\rangle + \langle \alpha\boldsymbol{\varphi}(y_n, \boldsymbol{\zeta}), \alpha\boldsymbol{\varphi}(y^*_{n'}, \boldsymbol{\zeta})\rangle \approx \\ &\quad \alpha^2 G_\sigma(x_n - x^*_{n'}) + \alpha^2 G_\sigma(y_n - y^*_{n'}).\end{aligned} \tag{9}$$

For Z pivots $\boldsymbol{\zeta}$, we use Z in range 3–10 and equally spaced intervals *e.g.*, $\boldsymbol{\zeta} = [-0.2 : 1.4/(Z-1) : 1.2]$ to encode the spatial coordinates x_n and y_n. The above formulation extends to the aggregation over patches extracted from images as shown in Figure 1. We form vectors $\bar{\boldsymbol{\phi}}_n = [\boldsymbol{\phi}_n; \mathbf{c}_n]$ which are augmented by encoded spatial coordinates $\mathbf{c}_n = [\alpha\boldsymbol{\varphi}(x_n, \boldsymbol{\zeta}); \alpha\boldsymbol{\varphi}(y_n, \boldsymbol{\zeta})]$. Thus, we define the total length of $\mathbf{c}_n$ as $Z' = 2Z$. Combining the augmented vectors with the Proposition 2 and Eq. (6) yields:

$$\boldsymbol{\Psi}\left(\{\bar{\boldsymbol{\phi}}_n\}_{n\in\mathcal{N}}\right) = \boldsymbol{\mathcal{G}}(\boldsymbol{M}), \quad \boldsymbol{M} = \frac{1}{N}\sum_{n\in\mathcal{N}} \bar{\boldsymbol{\phi}}_n \bar{\boldsymbol{\phi}}_n^T. \tag{10}$$

Gamma pooling follows Remark 3 and is simply defined by setting $\boldsymbol{\mathcal{G}}(\mathbf{X}) = (\lambda + \mathbf{X})^\gamma$, where rising $\boldsymbol{M}$ to the power of γ is element-wise and γ is a small regularization constant:

$$\boldsymbol{\Psi}\left(\{\bar{\boldsymbol{\phi}}_n\}_{n\in\mathcal{N}}\right) = \left(\lambda + \frac{1}{N}\sum_{n\in\mathcal{N}} \bar{\boldsymbol{\phi}}_n \bar{\boldsymbol{\phi}}_n^T\right)^\gamma. \tag{11}$$

4.2. Well-motivated Pooling Approaches

Prop. 3 states that quantity $1-(1-p)^N$ is the probability of at least one success being detected in the pool of the N i.i.d. trials performed according to the Bernoulli distribution with the success probability p and stored in $\boldsymbol{\phi} \in \{0,1\}^N$. Below we extend this simple theory to the case of co-occurrences.

Proposition 4. *Assume two event vectors $\boldsymbol{\phi}, \boldsymbol{\phi}' \in \{0,1\}^N$ which store the N trials each, performed according to the Bernoulli distribution under i.i.d. assumption, for which the probability p of an event $(\phi_n \cap \phi'_n = 1)$ denotes a co-occurrence and $1-p$, for $(\phi_n \cap \phi'_n = 0)$, denotes the lack of it, and p is estimated as an expected value $p = \text{avg}_n\, \phi_n \phi'_n$. Then the probability of at least one co-occurrence event $(\phi_n \cap \phi'_n = 1)$ in ϕ_n and ϕ'_n simultaneously in N trials becomes:*

$$\psi = 1-(1-p)^N. \tag{12}$$

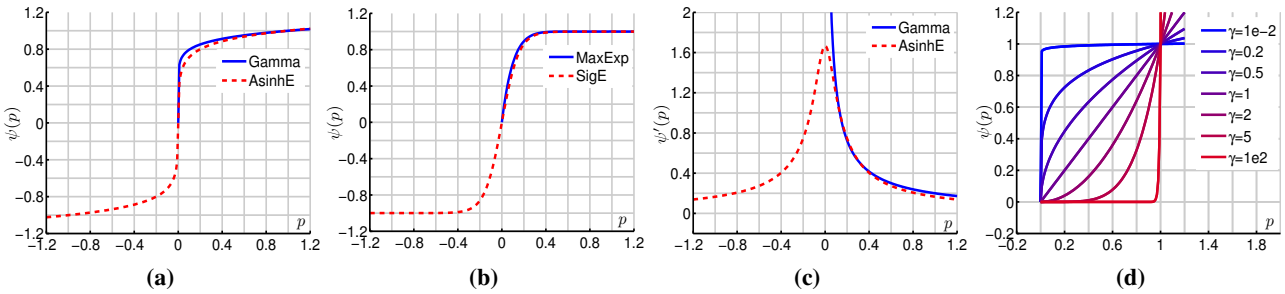

Figure 2: Gamma, AsinhE, MaxExp and SigmE are illustrated in Figures 2a and 2b while derivatives of Gamma and AsinhE are shown in Figure 2c. Lastly, Gamma for several values of γ is shown in Figure 2d from which its similarity to MaxExp in range 0–1 is clear.

Proof. The probability of all N outcomes to be $\{(\phi_1 \cap \phi'_1 = 0), ..., (\phi_N \cap \phi'_N = 0)\}$ amounts to $(1-p)^N$. The probability of at least one positive outcome $(\phi_1 \cap \phi'_1 = 1)$ amounts to applying the logical 'or' $\{(\phi_1 \cap \phi'_1 = 1) \,|...|\, (\phi_N \cap \phi'_N = 1)\}$ and leads to $1-(1-p)^N$, where $p = \text{avg}_n\, \phi_n \phi'_n$.

A stricter proof uses a Multinomial distribution model with four events for (ϕ_n) and (ϕ'_n) which describe all possible outcomes. Let probabilities p, q, s and $1–p–q–s$ add up to 1 and correspond to events $(\phi_n \cap \phi'_n = 1)$, $(\phi_n = 1, \phi'_n = 0)$, $(\phi_n = 0, \phi'_n = 1)$ and $(\phi_n \cup \phi'_n = 0)$. The first event is a co-occurrence, the latter two are occurrences only and the last event is the lack of the first three events. The probability of at least one co-occurrence $(\phi_n \cap \phi'_n = 1)$ in N trials becomes:

$$\sum_{n=1}^{N} \sum_{n'=0}^{N-n} \sum_{n''=0}^{N-n-n'} \binom{N}{n,n',n'',N-n-n'-n''} p^n q^{n'} s^{n''} (1–p–q–s)^{N-n-n'-n''}. \tag{13}$$

One can verify algebraically/numerically that Eq. (13) and (12) are equivalent w.r.t. p which completes the proof. □

Remark 4. *A practical implementation of this pooling strategy is given by* $\psi_{kl} = 1-(1-\text{avg}_n\, \phi_{kn}\phi_{ln})^\eta$, *where* $0 < \eta \approx N$ *is an adjustable parameter, and* ϕ_{kn} *and* ϕ_{ln} *are* k*-th and* l*-th features of an* n*-th feature vector* e.g., *as defined in Prop. 2, which is normalized as detailed next.*

Remark 5. *In practice,* p *is an expected value over* N *rectified co-occurring responses of pairs of convolutional filters rather than binary variables. A similar strategy is used with success in the BoW model [34]. In matrix form, we have:*

$$\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}, \eta) = 1 - \left(1 - \frac{\boldsymbol{M}}{\text{Tr}(\boldsymbol{M}) + \lambda}\right)^\eta, \tag{14}$$

where $\text{Tr}(\boldsymbol{M})$ *prevents elements of co-occurrence matrix* $\boldsymbol{M}$ *in enumerator of Eq. (14) from exceeding value of one, constant* $\lambda \approx$*1e-6 deals with the vanishing trace and* η *is chosen via cross-validation.*

Remark 6. $\boldsymbol{\mathcal{G}}^*(\boldsymbol{M}, \eta) = \boldsymbol{\mathcal{G}}(\boldsymbol{M}, \eta)(\text{Tr}(\boldsymbol{M}) + \lambda)^\gamma$ *compensates for the trace in (14) which affected the input-output ratio of norms.* $\boldsymbol{\mathcal{G}}^\ddagger(\boldsymbol{M}, \eta) = \boldsymbol{\mathcal{G}}(\boldsymbol{M}, \eta) + \kappa \boldsymbol{M}$ *prevents vanishing gradients in pooling. Both terms can be combined.*

We note that matrix $\boldsymbol{M}$ contains co-occurrences created from feature vectors $\boldsymbol{\phi}$ which were β-centered. Therefore, some entries of $\boldsymbol{M}$ may be negative. This breaks down pooling models such as Gamma and MaxExp for which we strictly use $\beta = 0$ that disables the anti-correlation mechanism. Nevertheless, we list detailed derivatives of these pooling functions w.r.t. the feature vectors in Appendix A.

4.3. Well-behaved Power Normalizations

Power Normalizations in Eq. (11) and (14) have infinite or undetermined gradients if coefficients $M_{mn} \to 0$ and $\lambda \to 0$. If regularization $\lambda > 0$, both power normalizations are somewhat compromised as their role is to magnify weak signals $\phi \approx 0$. Moreover, these pooling schemes break down in presence of negative entries $M_{mn} < 0$. Therefore, we propose the following poolings extensions.

SigmE pooling, used in lieu of MaxExp in Eq. (12) and (14), is given by Logistic a.k.a. Sigmoid (*SigmE*) functions:

$$\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}, \eta) = \frac{2}{1+\text{e}^{-\eta' \boldsymbol{M}}} - 1 \text{ and } \frac{2}{1+\text{e}^{\frac{-\eta' \boldsymbol{M}}{\text{Tr}(\boldsymbol{M})+\lambda}}} - 1. \tag{15}$$

AsinhE pooling is an alternative to Gamma function in Eq. 11. It is defined as the Arcsin hyperbolic function:

$$\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}, \eta) = \text{arcsinh}(\gamma' \boldsymbol{M}) = \log(\gamma' \boldsymbol{M} + \sqrt{1 + \gamma'^2 \boldsymbol{M}^2}), \tag{16}$$

Pooling function	$\psi(p)$ if $p<0$	$\psi'(p)$ if $p=0$	$\psi(p)$	$\psi'(p)$
Gamma [34]	inv.	∞	p^γ	$\gamma p^{\gamma-1}$
MaxExp [34]	inv.	fin.	$1-(1-p)^\eta$	$\eta(1-p)^{\eta-1}$
AsinhE	ok	fin.	$\text{Asinh}(\gamma' p)$	$\frac{\gamma'}{\sqrt{1+\gamma'^2 p^2}}$
SigmE	ok	fin.	$\frac{2}{1+e^{-\eta' p}} - 1$	$\frac{2\eta' e^{-\eta' p}}{(1+e^{-\eta' p})^2}$

Table 1: A collection of Power Normalization functions. Variables $\gamma > 0$, $\gamma' > 0$, $\eta \geq 1$, and $\eta' \geq 1$ control the level of power normalization. We indicate properties of ψ such as finite (*fin.*) or infinite (∞) derivative of ψ w.r.t. p at $p = 0$ and invalid (*inv.*) or valid (*ok*) power normalization for $p < 0$.

Figure 2 illustrates MaxExp and SigmE as well as Gamma and AsinhE functions from which it is clear that, for negative p, SigmE and AsinhE are natural extensions of MaxExp and Gamma, respectively. The derivative of AsinhE is smooth and finite (the same holds for SigmE) unlike the derivative of Gamma. Due to the above findings, we will perform our experiments on SigmE and AsinhE only. Table 1 lists various properties of the Power Normalization functions. Moreover, Appendix B provides detailed derivatives of these pooling functions w.r.t. the feature vectors. We used these derivatives in our end-to-end learning of CNNs.

Power Normalization functions have a whitening effect on features *i.e.*, the frequent bursts of the same kind of feature are reduced while the responses of rarely occurring features are magnified [34]. For co-occurrences of visual features, we showed in Prop. 4 that Power Normalizations act as detectors of co-occurring combinations of patterns *i.e.*, they capture if at least one co-occurrence of features takes place but they discard the quantity of such co-occurrences which otherwise would be a source of nuisance/noise.

4.4. Spectral Power Normalizations

Spectral versions of our pooling methods and their derivatives can be obtained by performing an SVD on $\boldsymbol{M}$, substituting eigenvalues λ_{ii} according to Table 1 such that $\lambda^{\star}_{ii} := \psi(\lambda_{ii})$ and computing $\boldsymbol{\mathcal{G}}(\boldsymbol{M}) = \boldsymbol{U}\boldsymbol{\lambda}^{\star}\boldsymbol{U}^T$. For derivatives, $\lambda^{\diamond}_{ii} := \psi'(\lambda_{ii})$ can be applied in back-propagation via SVD [25]. Table 2 shows that the spectral MaxExp and its derivative may be computed via matrix multiplications.

5. Experiments

Below we demonstrate experimentally merits of our second-order pooling with Power Normalizations.

Datasets. We employ four publicly available datasets and report the mean top-1 accuracy on each of them. The Flower102 dataset [44] is a fine-grained category recognition dataset that contains 102 categories of various flowers. Each class consists of between 40 and 258 images. The MIT67 dataset [48] contains a total of 15620 images belonging to 67 indoor scene classes. We follow the standard evaluation protocol, which uses a train and test split of 80% and 20% of images per class. The FMD dataset contains in total 100 images per category belonging to 10 categories of materials (*e.g.*, glass, plastic, leather) collected from the Flickr website. Lastly, the Food-101 dataset [5] has 101000 images in total and 1000 images per category.

	Gamma	*MaxExp*	*AsinhE*	*SigmE*
$\boldsymbol{\mathcal{G}}(\boldsymbol{M})$	$\boldsymbol{M}^{\gamma}$	$\mathbb{I}-(\mathbb{I}-\frac{\boldsymbol{M}}{\mathrm{Tr}(\boldsymbol{M})+\lambda})^{\eta}$	$\log\left(\gamma'\boldsymbol{M}+(\mathbb{I}+\gamma'^2\boldsymbol{M}^2)^{\frac{1}{2}}\right)$	$2\left(\mathbb{I}+e^{\frac{-\gamma'\boldsymbol{M}}{\mathrm{Tr}(\boldsymbol{M})+\lambda}}\right)^{-1}-\mathbb{I}$
der.	Eq. (24) /SVD	Eq. (25) /SVD	SVD	SVD

Table 2: A collection of spectral Power Normalization functions. The square, square root, power, log and exp are matrix operations.

Experimental setup. For Flower102 [44], we extract 12 cropped 224×224 patches per image and use mini-batch of size 5 to fine-tune the ResNet-50 model [21] pre-trained on ImageNet [50]. We obtain 2048 dim. $12\times7\times7$ conv. feature vectors from the last conv. layer for our second-order pooling layer. For MIT67 [48], we resize original images to 336×336 and use mini-batch of size 32, then fine-tune it on the ResNet-50 model [21] pre-trained on the Places-205 dataset [63]. With 336×336 image size, we obtain 2048 dim. 11×11 conv. feature vectors from the last conv. layer for our second-order pooling layer. For FMD [51] and Food101 [5], we resize images to 448×448, use mini-batch of size 32 and fine-tune ResNet-50 [21] pre-trained on ImageNet [50]. We use the 2048 dim. 14×14 conv. feature vectors from the last conv. layer. For ResNet-50, we fine-tune all layers for ∼20 epochs with learning rates 1e-4–1e-6. We use the Root Mean Square Propagation (RMSprop) [22] with the moving average 0.99. Where stated, we use AlexNet [36] with fine-tuned last two conv. layers. We use 256 dim. 6×6 conv. feature vectors from the last convolutional layer.

Our methods. We evaluate the generalizations of MaxExp and Gamma which are Logistic a.k.a. Sigmoid (*SigmE*) and the Arcsin hyperbolic (*AsinhE*) pooling functions. We focus mainly on our second-order representation (*SOP*) but we also occasionally report results for the first-order approach (*FOP*). For the baseline, we use the classifier on top of the *fc* layer (*Baseline*). The hyperparameters of our model are selected via cross-validation. The use of spatial coordinates is indicated by (*SC*) and spectral operators by (*Spec*).

5.1. Evaluations

We start by combining first- and second-order representations with SigmE and AsinhE pooling. We also investigate the impact of AlexNet and ResNet-50 on our approach.

Flower102. Table 3 shows that AlexNet performs worse than ResNet-50 which is consistent with the literature. For the standard ResNet-50 fine-tuned on Flower102, we ob-

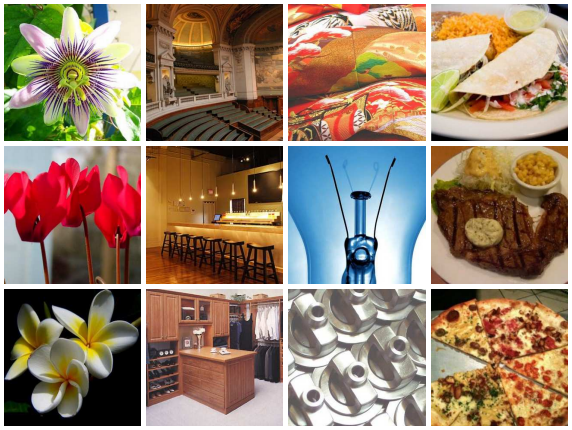

Figure 3: Each column shows examples of images from the Flower102, MIT67 FMD and Food101 dataset, respectively.

Method		top-1 accuracy
Second-order Bag-of-Words	[34]	90.2
Factors of Transferability	[3]	91.3
Reversal-inv. Image Repr.	[60]	94.0
Optimal two-stream fusion	[43]	94.5
Neural act. constellations	[53]	95.3

Method	Alexnet	ResNet-50
Baseline	82.00	94.06
FOP	85.40	94.08
FOP+AsinhE	85.64	94.60
SOP	87.20	94.70
SOP+AsinhE	88.40	95.12
SOP+SC+AsinhE	90.70	95.74
SOP+SC+SigmE	91.71	**96.78**
SOP+SC+Spec. Gamma	-	96.88
SOP+SC+Spec. MaxExp	-	**97.28**

Table 3: The Flower102 dataset. The bottom part shows our results for Alexnet and ResNet-50. The top part of the table lists state-of-the-art results from the literature.

tain 94.06% accuracy. The first-order Average and AsinhE pooling (*FOP*) and (*FOP+AsinhE*) score 94.08 and 94.6% accuracy. The second-order pooling (*SOP+AsinhE*) outperforms (*FOP+AsinhE*). We obtain the best result of **96.78**% for the second-order representation combined with spatial coordinates and SigmE pooling (*SOP+SC+SigmE*) which is 2.72% higher than our baseline. In contrast, a recent more complex state-of-the-art method [53] obtained 95.3% accuracy. Our scores highlight that capturing co-occurrences of visual features and passing them via a well-defined Power Normalization function such as SigmE works well for our fine-grained problem. We attribute the good performance of SigmE to its ability to act as a detector of co-occurrences. The role of the Hyperbolic Tangent non-linearity popular in deep learning may be explained by its similarity to SigmE. Lastly, our spectral MaxExp (*SOP+SC+Spec. MaxExp*) yields **97.28**% accuracy.

Scene recognition. Next, we validate our approach on MIT67–a larger dataset for scene recognition. Table 4

Method		top-1 accuracy
CNNs with Deep Supervision	[57]	76.1
Places-205	[56]	80.9
Deep Filter Banks	[12]	81.0
Spectral Features	[29]	84.3
Baseline		84.0
SOP+AsinhE		85.3
SOP+SigmE		85.6
SOP+SC+AsinhE		85.9
SOP+SC+SigmE		**86.3**

Table 4: The MIT67 dataset. The bottom part shows our results for ResNet-50 pre-trained on the Places-205 dataset. The top part of the table lists state-of-the-art results from the literature.

Method		acc.	Method	acc.
IFV+DeCAF	[11]	65.5	*Baseline*	83.4
FV+FC+CNN	[12]	82.2	*SOP+SC+AsinhE*	85.0
SMO Task	[62]	82.3	*SOP+SC+SigmE*	**85.5**

Table 5: The FMD dataset. Our (right) vs. other methods (left).

shows that all second-order approaches (*SOP*) outperform the standard ResNet-50 network (*Baseline*) pre-trained on the Places-205 dataset and fine-tuned on MIT67. Moreover, (*SigmE*) yields marginally better results than (*AsinhE*). Using spatial coordinates (*SC*) also results in additional gain in the classification performance. The second-order representation combined with spatial coordinates and SigmE pooling (*SOP+SC+SigmE*) yields **86.3**% accuracy and outperforms our baseline and [29] by 2.3 and 2%, respectively.

Material classification. Next, we quantify our performance on the FMD dataset for material/texture recognition. Table 5 demonstrates that our second-order representation (*SOP+SC+SigmE*) scores **85.5**% accuracy and outperforms our baseline approach by 2.1%. We note that our approach and the baseline use the same testbed. The only difference is our second-order representations, spatial coordinates and Power Normalization components in our last layer.

Food101. We apply our strongest second-order representations (*SOP+SC+SigmE*) and (*SOP+SC+Spec. MaxExp*) to this dataset and obtain **87.5**% and **87.8**% accuracy. In contrast, a recent more involved kernel pooling [13] reports 85.5% accuracy while the baseline approach scores only 81.9% in the same testbed. This demonstrates the strength of our approach on fine-grained problems.

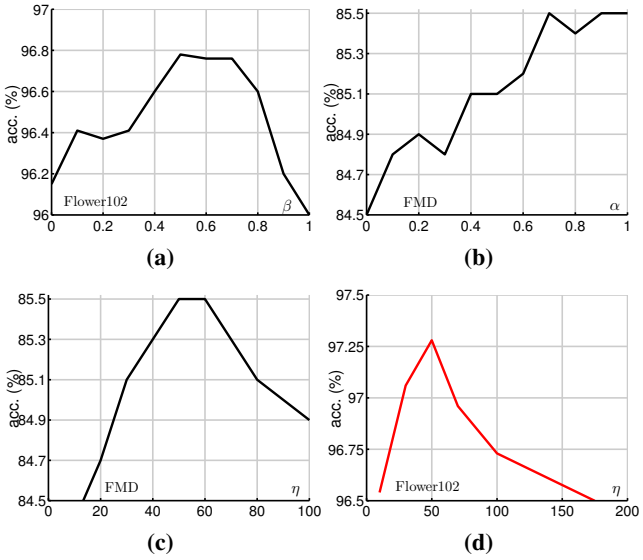

Figure 4: Performance w.r.t. hyperparameters. Figures 4a and 4b: β-centering on Flower102 and α for spatial coordinate encoding on FMD. Figures 4c and 4d: the accuracy w.r.t. the η' and η parameters given SigmE and the spectral MaxExp.

Performance w.r.t. hyperparameters. Figure 4a demonstrates that β-centering has a positive impact on image classification with ResNet-50. This strategy, detailed in Section 4.1, is trivial to combine with our pooling. Figure 4b shows that setting non-zero α, which lets encode spatial coordinates according to Eq. (9), brings additional gain in accuracy at no extra cost. Figure 4c demonstrates that over 1% accuracy can be gained by tuning our SigmE pooling. Moreover, Figure 4d shows that the spectral MaxExp can yield further gains over element-wise SigmE and MaxExp for carefully chosen η. Lastly, we have observed that our spectral and element-wise MaxExp converged in 3–12 and 15–25 iterations, resp. This shows that both spectral and element-wise pooling have their strong and weak points.

6. Conclusions

We have studied Power Normalizations in the context of co-occurrence representations and demonstrated their theoretical role which is to 'detect' co-occurring pairs of features. We have proposed surrogate functions SigmE and AsinhE which can handle so-called negative evidence and have well-behaved derivatives for end-to-end learning which we performed. Our pooling operators are element-wise therefore they are cheap to implement in GPU. Moreover, our pooling operators easily extend to spectral pooling. We have demonstrated state-of-the-art results on four popular benchmarks and sensible gains on powerful ResNet-50.

Appendices

A. Derivatives of Average, Gamma and MaxExp functions

Let $\boldsymbol{\Phi} = [\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_N] \in \mathbb{R}^{d\times N}$, $\mathbf{C} = [\mathbf{c}_1, ..., \mathbf{c}_N] \in \mathbb{R}^{Z'\times N}$, and some class. loss $\ell(\boldsymbol{\Psi}, \boldsymbol{W})$, where $\boldsymbol{\Psi} \in \mathcal{S}_+^{d+Z'}$ (or $\mathcal{S}_{++}$) and $\boldsymbol{W}$ are our descriptor and a hyperplane. Eq. (10) yields:

$$\frac{\partial \sum_n \bar{\boldsymbol{\phi}}_n \bar{\boldsymbol{\phi}}_n^T}{\partial \phi_{m'n'}} = \begin{bmatrix} \boldsymbol{j}_{m'}\boldsymbol{\phi}_{n'}^T + \boldsymbol{\phi}_{n'}\boldsymbol{j}_{m'}^T & \boldsymbol{j}_{m'}\mathbf{c}_{n'}^T \\ \mathbf{c}_{n'}\boldsymbol{j}_{m'}^T & [0]_{Z'\times Z'} \end{bmatrix}, \quad (17)$$

where $[0]_{Z'\times Z'}$ denotes array of size $Z'\times Z'$ filled with zeros.
Average pooling is set by $\boldsymbol{\mathcal{G}}(\boldsymbol{M}) = \boldsymbol{M}$ and $\boldsymbol{D} = \mathbb{1}\mathbb{1}^T$ so that $\boldsymbol{\Psi} = \boldsymbol{M} = \frac{1}{N}\sum_n \bar{\boldsymbol{\phi}}_n \bar{\boldsymbol{\phi}}_n^T$. Thus, the full derivative becomes:

$$\sum_{k,l} \frac{\partial \ell(\boldsymbol{\Psi}, \boldsymbol{W})}{\partial \Psi_{kl}} \frac{\partial \Psi_{kl}}{\partial \boldsymbol{\Phi}} = \frac{2}{N} \operatorname{Sym}\Big(\frac{\partial \ell(\boldsymbol{\Psi}, \boldsymbol{W})}{\partial \boldsymbol{\Psi}} \odot \boldsymbol{D}\Big)_{(1:d,:)} \begin{bmatrix} \boldsymbol{\Phi} \\ \mathbf{C} \end{bmatrix}. \quad (18)$$

Gamma pooling is set by $\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}) = (\lambda + \boldsymbol{M})^\gamma$, where rising $\boldsymbol{M}$ to the power of γ is element-wise and λ is a reg. constant. Thus, we obtain:

$$\frac{\partial \boldsymbol{\Psi}}{\partial \phi_{m'n'}} = \frac{1}{N}\gamma(\lambda + \boldsymbol{M})^{\gamma-1} \odot \frac{\partial \sum_n \bar{\boldsymbol{\phi}}_n \bar{\boldsymbol{\phi}}_n^T}{\partial \phi_{m'n'}}. \quad (19)$$

The derivative is given by Eq. (18) if $\boldsymbol{D} = \gamma(\lambda + \boldsymbol{M})^{\gamma-1}$.
MaxExp pooling $\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}) = 1 - (1 - \boldsymbol{M}/(\operatorname{Tr}(\boldsymbol{M}) + \lambda))^\eta$ has the derivative given by Eq. (18) with the following $\boldsymbol{D}$:

$$\boldsymbol{D} = \eta\left(1 - \frac{\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M}) + \lambda}\right)^{\eta-1} \odot \boldsymbol{T} \text{ and } \boldsymbol{T} = \left(\frac{1}{\operatorname{Tr}(\boldsymbol{M})+\lambda} - \frac{\boldsymbol{M}\odot\mathbb{I}}{(\operatorname{Tr}(\boldsymbol{M})+\lambda)^2}\right), \quad (20)$$

where multiplication $\odot$, division, rising to the power *etc.* are all element-wise operations.

B. Derivatives of SigmE and AsinhE pooling

SigmE pooling is set by $\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}) = \frac{2}{1+\mathrm{e}^{-\eta'\boldsymbol{M}}} - 1$ or trace-normalized $\frac{2}{1+\mathrm{e}^{\frac{-\eta'\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})+\lambda}}} - 1$. The first expression yields:

$$\frac{\partial \boldsymbol{\Psi}}{\partial \phi_{m'n'}} = \frac{1}{N}\frac{2\eta'\mathrm{e}^{-\eta'\boldsymbol{M}}}{(1+\mathrm{e}^{-\eta'\boldsymbol{M}})^2} \odot (\boldsymbol{j}_{m'}\boldsymbol{\phi}_{n'}^T + \boldsymbol{\phi}_{n'}\boldsymbol{j}_{m'}^T), \quad (21)$$

where multiplication $\odot$, division, and exponentiation are all element-wise operations.
AsinhE pooling is set by $\boldsymbol{\Psi} = \boldsymbol{\mathcal{G}}(\boldsymbol{M}) = \operatorname{arcsinh}(\gamma'\boldsymbol{M}) = \log(\gamma'\boldsymbol{M} + \sqrt{1+\gamma'^2\boldsymbol{M}^2})$ which yields the following:

$$\frac{\partial \boldsymbol{\Psi}}{\partial \phi_{m'n'}} = \frac{1}{N}\frac{\gamma'}{\sqrt{\gamma'^2\boldsymbol{M}^2+1}} \odot (\boldsymbol{j}_{m'}\boldsymbol{\phi}_{n'}^T + \boldsymbol{\phi}_{n'}\boldsymbol{j}_{m'}^T), \quad (22)$$

where multiplication $\odot$, division, square root and the square are all element-wise operations.

For SigmE, trace-normalized SigmE and AsinhE pooling methods, the final derivatives are given by Eq. (18) with the following $\boldsymbol{D}$, respectively:

$$\boldsymbol{D} = \frac{2\eta'\mathrm{e}^{-\eta'\boldsymbol{M}}}{(1+\mathrm{e}^{-\eta'\boldsymbol{M}})^2} \text{ or } \boldsymbol{D} = \frac{2\eta'\mathrm{e}^{\frac{-\eta'\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})+\lambda}}}{(1+\mathrm{e}^{\frac{-\eta'\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})+\lambda}})^2} \odot \boldsymbol{T} \text{ and } \boldsymbol{D} = \frac{\gamma'}{\sqrt{\gamma'^2\boldsymbol{M}^2+1}}. \quad (23)$$

Moreover, for SigmE and AsinhE we allow β-centering so that $\boldsymbol{\phi}_n := \boldsymbol{\phi}_n - \beta\boldsymbol{\mu}$ and $\boldsymbol{\phi}_n^* := \boldsymbol{\phi}_n^* - \beta\boldsymbol{\mu}^*$. Thus, the derivative of this substitution has to be included in the chain rule.

C. Derivatives of Spectral Gamma and MaxExp

Gamma pooling has derivative which can be solved by the SVD back-propagation or the Sylvester equation if $\gamma = 0.5$:

$$2\operatorname{Res}\Big(\operatorname{Sym}\Big(\frac{\partial \ell(\boldsymbol{\Psi}, \boldsymbol{W})}{\partial \boldsymbol{\Psi}}\Big)_{(:)}^T \boldsymbol{M}^*\Big)_{d+Z'\times d+Z'} \text{ and } \boldsymbol{M}^* = (\mathbb{I}\otimes\boldsymbol{M}^{\frac{1}{2}} + \boldsymbol{M}^{\frac{1}{2}}\otimes\mathbb{I})^\dagger, \quad (24)$$

where $\otimes$ and $\dagger$ are the Kronecker product and the pseudo-inverse. Matrix vectorization and reshaping to the size $m\times n$ are denoted by $(:)$ and $\operatorname{Res}(\mathbf{X})_{m\times n}$.
MaxExp has a closed-form derivative which requires the following chain rule:

$$\frac{\partial \boldsymbol{\mathcal{G}}(\boldsymbol{M})}{\partial M_{kl}} = \frac{1}{\operatorname{Tr}(\boldsymbol{M})}\sum_{n=0}^{\eta-1}\Big(\mathbb{I} - \frac{\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})}\Big)^n\Big(\mathbf{J}_{kl} - \frac{\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})}\mathbb{I}_{kl}\Big)\Big(\mathbb{I} - \frac{\boldsymbol{M}}{\operatorname{Tr}(\boldsymbol{M})}\Big)^{\eta-1-n}. \quad (25)$$

References

[1] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. *CVPR*, 2016. 2

[2] V. Arsigny, P. Fillard, X. Pennec, and N. Ayache. Log-euclidean metrics for fast and simple calculus on diffusion tensors. *Magnetic resonance in medicine*, 56(2):411–421, 2006. 2

[3] H. Azizpour, A. S. Razavian, J. Sullivan, A. Maki, and S. Carlsson. Factors of transferability for a generic convnet representation. *CoRR*, abs/1406.5774, 2015. 7

[4] R. Bhatia. Positive definite matrices. *Princeton Univ Press*, 2007. 2

[5] L. Bossard, M. Guillaumin, and L. J. V. Gool. Food-101 - mining discriminative components with random forests. *ECCV*, pages 446–461, 2014. 6

[6] S. Boughorbel, J.-P. Tarel, and N. Boujemaa. Generalized Histogram Intersection Kernel for Image Recognition. *ICIP*, 2005. 3

[7] Y. Boureau, F. Bach, Y. LeCun, and J. Ponce. Learning Mid-Level Features for Recognition. *CVPR*, 2010. 1, 3, 4

[8] Y. Boureau, J. Ponce, and Y. LeCun. A Theoretical Analysis of Feature Pooling in Vision Algorithms. *ICML*, 2010. 1, 3, 4

[9] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic Segmentation with Second-Order Pooling. *ECCV*, 2012. 1, 2

[10] A. Cherian, S. Sra, A. Banerjee, and N. Papanikolopoulos. Jensen-Bregman LogDet Divergence with Application to Efficient Similarity Search for Covariance Matrices. *TPAMI*, 35(9):2161–2174, 2013. 2

[11] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, and A. Vedaldi. Describing textures in the wild. *CVPR*, pages 3606–3613, 2014. 7

[12] M. Cimpoi, S. Maji, and A. Vedaldi. Deep filter banks for texture recognition and segmentation. *CVPR*, pages 3828–3836, 2015. 7

[13] Y. Cui, F. Zhou, J. Wang, X. Liu, Y. Lin, and S. Belongie. Kernel pooling for convolutional neural networks. *CVPR*, 2017. 7

[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*, 3(3):1102–1123, 2009. 2

[15] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. *ECCV*, pages 392–407, 2014. 2

[16] K. Guo, P. Ishwar, and J. Konrad. Action recognition from video using feature covariance matrices. *Trans. Img. Proc.*, 22(6):2479–2494, 2013. 1

[17] M. Harandi, R. Hartley, C. Shen, B. Lovell, and C. Sanderson. Extrinsic methods for coding and dictionary learning on grassmann manifolds. *IJCV*, 2015. 2

[18] M. Harandi and M. Salzmann. Riemannian coding and dictionary learning: Kernels to the rescue. *CVPR*, 2015. 2

[19] M. Harandi, M. Salzmann, and M. Baktashmotlagh. Beyond gauss: Image-set matching on the riemannian manifold of pdfs. *ICCV*, 2015. 2

[20] M. Harandi, M. Salzmann, and R. Hartley. Joint dimensionality reduction and metric learning: A geometric take. *ICML*, page 14041413, 2017. 2

[21] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CVPR*, June 2016. 6

[22] G. Hinton. Neural Networks for Machine Learning Lecture 6a: Overview of mini-batch gradient descent Reminder: The error surface for a linear neuron. Lecture notes, https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf, 2017. Accessed: 10-11-2017. 6

[23] G. Hu, Y. Hua, Y. Yuan, Z. Zhang, Z. Lu, S. S. Mukherjee, T. M. Hospedales, N. M. Robertson, and Y. Yang. Attribute-enhanced face recognition with neural tensor fusion networks. *ICCV*, 2017. 2

[24] Z. Huang and L. V. Gool. A riemannian network for spd matrix learning. *AAAI*, pages 2036–2042, 2017. 2

[25] C. Ionescu, O. Vantzos, and C. Sminchisescu. Matrix backpropagation for deep networks with structured layers. *ICCV*, 2015. 2, 6

[26] T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *JMLR*, 5:819–844, 2004. 3

[27] H. Jegou and O. Chum. Negative evidences and cooccurrences in image retrieval: the benefit of pca and whitening. *ECCV*, 2012. 4

[28] H. Jégou, M. Douze, and C. Schmid. On the Burstiness of Visual Elements. *CVPR*, pages 1169–1176, 2009. 3

[29] S. H. Khan, M. Hayat, and F. Porikli. Scene categorization with spectral features. *ICCV*, pages 5638–5648, 2017. 7

[30] P. Koniusz and A. Cherian. Sparse coding for third-order super-symmetric tensor descriptors with application to texture recognition. *CVPR*, 2016. 1

[31] P. Koniusz and K. Mikolajczyk. Spatial coordinate coding to reduce histogram representations, dominant angle and colour pyramid match. *ICIP*, 2011. 1, 3

[32] P. Koniusz, Y. Tas, and F. Porikli. Domain adaptation by mixture of alignments of second- or higher-order scatter tensors. *CoRR*, abs/1409.1556, 2016. 3

[33] P. Koniusz, F. Yan, P. Gosselin, and K. Mikolajczyk. Higher-order Occurrence Pooling on Mid- and Low-level Features: Visual Concept Detection. *Technical Report*, 2013. 1

[34] P. Koniusz, F. Yan, P. Gosselin, and K. Mikolajczyk. Higher-order occurrence pooling for bags-of-words: Visual concept detection. *PAMI*, 2016. 1, 2, 3, 4, 5, 6, 7

[35] P. Koniusz, F. Yan, and K. Mikolajczyk. Comparison of Mid-Level Feature Coding Approaches And Pooling Strategies in Visual Concept Detection. *CVIU*, 2012. 1, 3, 4

[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. *NIPS*, pages 1106–1114, 2012. 6

[37] S. Kumar Roy, Z. Mhammedi, and M. Harandi. Geometry aware constrained optimization techniques for deep learning. *CVPR*, 2018. 2

[38] P. Li and Q. Wang. Local log-euclidean covariance matrix (l^2 ecm) for image representation and its applications. *ECCV*, 2012. 1

[39] P. Li, J. Xie, Q. Wang, and W. Zuo. Is second-order information helpful for large-scale visual recognition? *ICCV*, 2017. 2

[40] T.-Y. Lin, A. R. Chowdhury, and S. Maji. Bilinear cnn models for fine-grained visual recognition. *ICCV*, 2017. 2

[41] T.-Y. Lin and S. Maji. Improved Bilinear Pooling with CNNs. *BMVC*, 2017. 2
[42] L. Lingqiao, L. Wang, and X. Liu. In Defence of Soft-assignment Coding. *ICCV*, 2011. 1, 3
[43] J. Liu, C. Gao, D. Meng, and W. Zuo. Two-stream contextualized cnn for fine-grained image classification. *AAAI*, 2016. 7
[44] M.-E. Nilsback and A. Zisserman. Automated Flower Classification over a Large Number of Classes. *ICVGIP*, Dec 2008. 6
[45] X. Pennec, P. Fillard, and N. Ayache. A Riemannian Framework for Tensor Computing. *IJCV*, 66(1):41–66, 2006. 2
[46] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher Kernel for Large-Scale Image Classification. *ECCV*, pages 143–156, 2010. 1, 3
[47] F. Porikli and O. Tuzel. Covariance tracker. *CVPR*, 2006. 1, 2
[48] A. Quattoni and A. Torralba. Recognizing indoor scenes. *CVPR*, 2009. 6
[49] A. Romero, M. Y. Terán, M. Gouiffès, and L. Lacassagne. Enhanced local binary covariance matrices (ELBCM) for texture analysis and object tracking. *MIRAGE*, pages 10:1–10:8, 2013. 2
[50] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 6
[51] L. Sharan, R. Rosenholtz, and E. H. Adelson. Material perception: What can you see in a brief glance? *Journal of Vision*, 14(9), 2014. 6
[52] Y.-F. Shih, Y.-M. Yeh, Y.-Y. Lin, M.-F. Weng, Y.-C. Lu, and Y.-Y. Chuang. Deep co-occurrence feature learning for visual object recognition. *CVPR*, 2017. 3
[53] M. Simon and E. Rodner. Neural activation constellations: Unsupervised part model discovery with convolutional networks. *ICCV*, pages 1143–1151, 2015. 7
[54] O. Tuzel, F. Porikli, and P. Meer. Region covariance: A fast descriptor for detection and classification. *ECCV*, 2006. 1, 2
[55] O. Tuzel, F. Porikli, and P. Meer. Pedestrian detection via classification on riemannian manifolds. *PAMI*, 30(10):1713–1727, 2008. 2
[56] L. Wang, S. Guo, W. Huang, and Y. Qiao. Places205-vggnet models for scene recognition. *CoRR*, abs/1508.01667, 2015. 7
[57] L. Wang, C.-Y. Lee, Z. Tu, and S. Lazebnik. Training deeper convolutional networks with deep supervision. *CoRR*, abs/1505.02496, 2015. 7
[58] Q. Wang, F. Chen, and W. Xu. Tracking by third-order tensor representation. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 41(2):385–396, 2011. 1
[59] Z. Wang and B. C. Vemuri. An affine invariant tensor dissimilarity measure and its applications to tensor-valued image segmentation. *CVPR*, 2004. 2
[60] L. Xie, J. Wang, W. Lin, B. Zhang, and Q. Tian. Towards reversal-invariant image representation. *IJCV*, 123(2):226–250, 2017. 7
[61] A. B. Yandex and V. Lempitsky. Aggregating local deep features for image retrieval. *ICCV*, pages 1269–1277, 2015. 2
[62] Y. Zhang, M. Ozay, X. Liu, and T. Okatani. Integrating deep features for material recognition. *ICPR*, pages 3697–3702, 2016. 7
[63] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. *NIPS*, 2014. 6