Semantic Fisher Scores for Task Transfer: Using Objects to Classify Scenes

Mandar Dixit, *Student Member, IEEE,* Yunsheng Li, *Student Member, IEEE,*
Nuno Vasconcelos, *Fellow, IEEE,*

Abstract—The tranfer of a neural network (CNN) trained to recognize objects to the task of scene classification is considered. A Bag-of-Semantics (BoS) representation is first induced, by feeding scene image patches to the object CNN, and representing the scene image by the ensuing bag of posterior class probability vectors (semantic posteriors). The encoding of the BoS with a Fisher vector (FV) is then studied. A link is established between the FV of any probabilistic model and the Q-function of the expectation-maximization (EM) algorithm used to estimate its parameters by maximum likelihood. This enables 1) imediate derivation of FVs for any model for which an EM algorithm exists, and 2) leveraging efficient implementations from the EM literature for the computation of FVs. It is then shown that standard FVs, such as those derived from Gaussian or even Dirichelet mixtures, are unsuccessful for the transfer of semantic posteriors, due to the highly non-linear nature of the probability simplex. The analysis of these FVs shows that significant benefits can ensue by 1) designing FVs in the natural parameter space of the multinomial distribution, and 2) adopting sophisticated probabilistic models of semantic feature covariance. The combination of these two insights leads to the encoding of the BoS in the natural parameter space of the multinomial, using a vector of Fisher scores derived from a mixture of factor analyzers (MFA). A network implementation of the MFA Fisher Score (MFA-FS), denoted as the MFAFSNet, is finally proposed to enable end-to-end training. Experiments with various object CNNs and datasets show that the approach has state-of-the-art transfer performance. Somewhat surprisingly, the scene classification results are superior to those of a CNN explicitly trained for scene classification, using a large scene dataset (Places). This suggests that holistic analysis is insufficient for scene classification. The modeling of local object semantics appears to be at least equally important. The two approaches are also shown to be strongly complementary, leading to very large scene classification gains when combined, and outperforming all previous scene classification approaches by a sizeable margin.

Index Terms—Deep Neural Network, Scene Classification, Fisher Vector, MFA

1 Introduction

Convolutional neural networks (CNNs) have achieved remarkable performance on vision problems such as image classification [1–3] or object detection and localization [4–6]. Beyond impressive results, they have an unmatched resilience to dataset bias [7]. It is now well known that a network trained to solve a task on a certain dataset (e.g. object recognition on ImageNet [8]) can be easily fine-tuned to a related problem on another dataset (e.g. object detection on MS-COCO). Less studied is robustness to task bias, i.e. generalization across tasks. In this work, we consider an important class of such problems, where a classifier trained on a set of semantics is transferred to a second set of semantics, which are loose combinations of the original ones. We consider the particular case where original semantics are object classes and target semantics are scene classes that somehow depend on those objects.

Task transfer has been a topic of significant interest in computer vision. Prominent examples of cross-task transfer include object detectors learned from object recognition models [4, 9], object recognizers based on attribute detectors [10, 11] and complex activity recognition methods based on attribute detection [12, 13] or object recognition [14, 15]. Our particular interest in object to scene transfer stems from the complex relation between the two domains. A scene can be described as a collection of multiple objects occurring in an unpredictable layout. Localizing the scene semantics is already a difficult task. This is compounded by the difficulty of mapping localized semantics into a holistic scene representation. The problem of knowledge transfer from object to scene recognizers is therefore very challenging.

One might argue that instead of using transfer, a scene classifier CNN can be trained directly from a large dataset of scene images. This approach has two major limitations. First, it does not leverage all the work already devoted to object recognition in the literature. Both datasets and models have to be designed from scratch, which is time consuming. Second, the "directly learned" CNN does not necessarily model relations between holistic scene descriptions and scene objects. This can degrade classification performance. We consider instead the prediction of holistic scene tags from the scores produced by an object CNN classifier. Since it leverages available object recognition CNNs this type of transfer is more efficient in terms of data collection and training. We show that it can also produce better scene classification results. This is because a scene classifier can leverage the recognition of certain types of rocks, tree stumps, or lizard species to distinguish between "Arizona Desert" and "Joshua Tree National Park". A holistically trained CNN can have difficulty honing in on these objects as discriminators between the two classes.

The proposed object-to-scene transfer is based on the

- *M. Dixit is with the Microsoft, Redmond, WA, 98052. E-mail: madixit@microsoft.com*
- *Y. Li and N. Vasconcelos are with Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA 92093.*

Manuscript received Oct -, 2018; revised Jan -, 2019.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2921960, IEEE Transactions on Pattern Analysis and Machine Intelligence

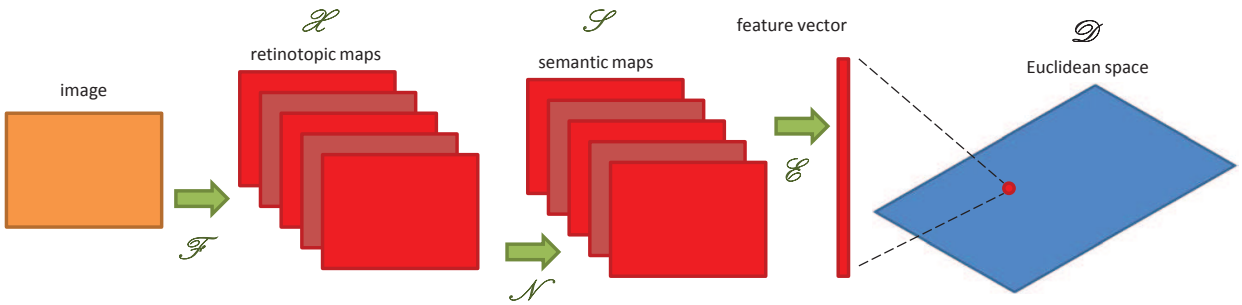


Fig. 1: The bag of semantics (BoS) classifier consists of a retinotopic feature mapping $\mathcal{F}$ followed by a semantic mapping $\mathcal{N}$. A non-linear embedding $\mathcal{E}$ of the semantic maps is then used to generate a feature vector on a Euclidean space D.

bag of semantics (BoS) representation. It derives a scene representation by scoring a set of image patches with a pre-trained object classifier. The probabilities of different objects are the *scene semantics* and the set of probability vectors the BoS. A holistic scene classifier is then applied to the BoS, to transfer knowledge from objects to scenes. Several authors have argued for semantic image representations in vision [16–22]. They have been used to describe objects by their attributes [20], represent scenes as collections of objects [22, 23] and capture contextual relations between classes [24]. For tasks such as hashing or large scale retrieval, a global semantic descriptor is usually preferred [25, 26]. Works on zero-shot object based scene representation [23] also use a global semantic image descriptor, mainly because the object-to-scene transfer functions used in these problems require dimensions of the descriptor to be interpretable as object scores. Proposals for scene classification, on the other hand, tend to rely on the BoS [18, 19, 22]. However, while the BoS outperforms low-level features in low dimensions [19], it has been less effective for high dimensional descriptors such as the Fisher vector (FV) [27]. This is because region semantics can be noisy, and it is hard to map a bag of probability vectors into a high dimensional scene representation, such as a FV [27].

In this work, we leverage the high accuracy of ImageNet trained CNNs [1, 2] to overcome the first problem. We obtain a BoS by using these networks to extract semantic descriptors (object class posterior probability vectors) from local image patches. We then extend the FV to this BoS. This *semantic Fisher vector* amounts to a large set of non-linear pooling operators that act on high-dimensional probability vectors. We show that, unlike low-level features, this extension cannot be implemented by the classical Gaussian mixture model FV (GMM-FV). We simplify the derivation of FVs for other models, by linking the FV of any probabilistic model to the Q-function of the expectation-maximization (EM) algorithm used to estimate its parameters. It is shown that the FV can be trivially computed as a combination of the E and M steps of EM. This link also enables the leveraging of efficient EM implementations to compute FVs. It is, however, shown that even a more natural distribution for probability vectors, the Dirichlet mixture model (DMM), fails to generate an effective FV for the BoS.

We hypothesize that this is due to the non-Euclidean nature of the probability simplex, which makes the modeling of probability distributions quite complex. Since the FV is always defined with respect to a reference probability distribution, this hurts classification performance. For the GMM-FV, the problem is that the assumption of a locally Euclidean geometry is not well suited for image semantics, which are defined on the simplex. For the DMM-FV, the problem is a lack of explicit modeling of second order statistics of these semantics. Nevertheless, an analysis of the DMM-FV reveals a non-linear log embedding that maps a multinomial distribution to its natural parameter space, where the Euclidean assumption is effective. This suggests using a GMM model on the natural parameter space of the semantic multinomial (SMN), leading to the logSMN-FV. In fact, because the multinomial has various natural space parametrization, we seek the one best suited for CNN semantics. This turns out to be the inverse of the softmax implemented at the network output. Since the CNN is optimized for these semantics, this parameterization has the benefits of end-to-end training. It is shown that a GMM-FV of the pre-softmax CNN outputs significantly outperforms the GMM-FV and the DMM-FV.

While these results show an advantage for modeling second order statistics, the use of a GMM of diagonal covariances limits the ability of the GMM-FV to approximate the non-linear manifold of CNN natural parameter features. For this, we resort to a richer generative model, the mixture of factor analyzers (MFA) [28, 29], which locally approximates the natural-space BoS manifold by a set of low-dimensional linear subspaces, derived from covariance information. We derive the MFA Fisher score (MFA-FS) and corresponding MFA-FV and show that the covariance statistics captured by these descriptors are highly discriminant for CNN semantics, significantly outperforming the GMM-FV. To allow end-to-end training, the MFA-FS is finally implemented as a neural network layer. The resulting MFAFSNet is an object to scene transfer network that can be fine-tuned for scene classification by backpropagation. This further improves scene classification performance.

Experiments on the SUN [30] and MIT Indoor [31] datasets show that the MFA representations (MFA-FS and MFAFSNet) outperform scene classifiers based on lower level CNN features [32–35], alternative approaches for second order pooling of CNN semantics [36, 37], and even CNNs learned directly from scene datasets [6, 38, 39]. This is surprising, since the MFA representations perform task transfer, applying object recognition CNNs to scene classification, and require little scene training data. This is unlike direct CNN training, which requires a much larger scene dataset, such as Places [6]. Furthermore, the two representations are complementary: combination of the MFA-FS and the scene CNN significantly outperforms the methods in isolation. The combined classifier has state-of-the-art scene classification performance, achieving sizable improvements

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2921960, IEEE Transactions on Pattern Analysis and Machine Intelligence

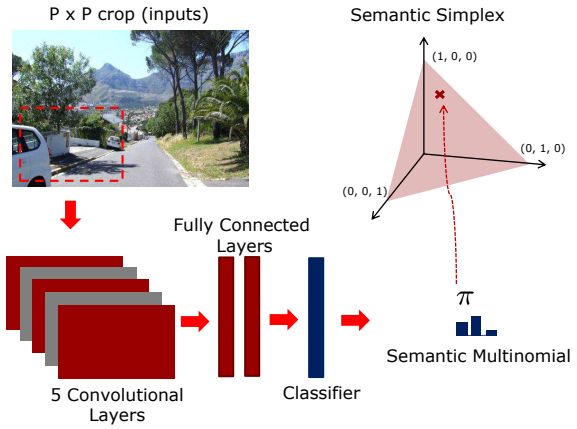


Fig. 2: CNN based semantic image representation. Each image patch is mapped into an SMN π.

over all previous approaches.

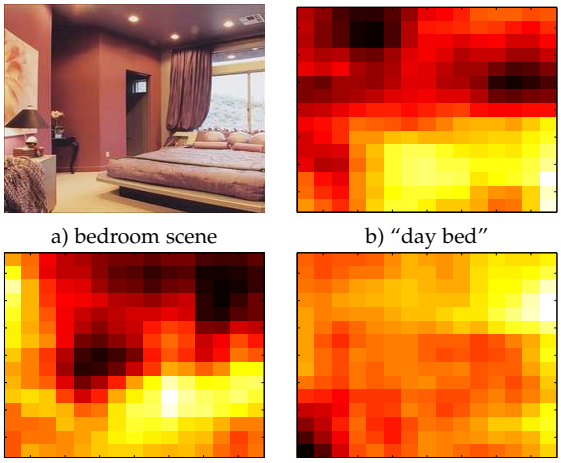
2 Bag of Semantics Classification

We start by reviewing the foundations of BoS classification.

2.1 Prior work

Figure 1 presents the architecture of the BoS classifier. Given an image $I(l)$, where l denotes spatial location, it defines an initial mapping $\mathcal{F}$ into a set of *retinotopic* feature maps $f_k(l)$. These preserve spatial topology of the image and encode local visual information. They have been implemented with handcrafted descriptors such as SIFT, HoG or the convolutional layers of a CNN. The next stage is a second retinotopic mapping $\mathcal{N}$ into the space of classifier outputs $\mathcal{S}$. Classifiers that define this mapping are pre-trained on an auxiliary set of *semantic concepts*, e.g. objects [22, 40] or themes [19, 24, 41], that occur locally within images. At each location l, they map the descriptors extracted at l into a *semantic vector* in $\mathcal{S}$, whose entries are probabilities of occurrence of the individual semantic concepts. The image is thus, transformed into a collection or a "bag" of semantics. However, due to their retinotopic nature, a BoS is sensitive to variations in scene layout. If an object changes position in the field of view, the semantic feature map will change completely. To guarantee *invariance*, the BoS is embedded into a fixed length non-retinotopic representation, using a non-linear mapping $\mathcal{E}$ into a high dimensional feature space $\mathcal{D}$. The space $\mathcal{D}$ must have a Euclidean structure that supports classification with linear decision boundaries.

Prior BoS scene classifiers [19, 22, 24, 40, 41] had limited success, for two reasons. First, scene semantics are non-trivial to *localize*. Scenes are collections of objects and stuff [42] in diverse layouts. Detecting these entities can be challenging. Object detectors based on handcrafted features, such as SIFT or HoG, lacked discriminative power, producing mappings $\mathcal{N}$ riddled with *semantic noise* [24]. Second, it can be difficult to design an invariant scene descriptor (embedding $\mathcal{E}$). The classical pooling of the bag of descriptors into a vector of statistics works well for low and mid-level features [43–45] but is far less effective for the BoS. Semantic features are class probabilities that inhabit a very non-Euclidean simplex. Commonly used statistics, such as average or max pooling [19, 22], do not perform well in this space. Our experiments show that even sophisticated non-linear embeddings, such as FVs [27], can perform poorly.



a) bedroom scene b) "day bed" c) "quilt, comforter" d) "window screen"

Fig. 3: ImageNet based BoS for a) bedroom image. Object recognition channels for b) "day bed" c) "comforter" and d) "window screen."

The introduction of deep CNNs [1–3] has all but solved the problem of noisy semantics. These models learn highly discriminative and non-linear image mappings $\mathcal{F}$ that are far superior to handcrafted features. Their top layers have been shown selective of semantics such as faces and object parts [46]. As discussed in the following section, scoring the local regions of a scene with an object recognition CNN produces a robust BoS. It remains to design the embedding $\mathcal{E}$. This is discussed in the remainder of the paper.

2.2 CNN semantics

Given a vocabulary $\mathcal{V} = \{v_1, \ldots, v_S\}$ of S *semantic concepts*, an image I can be described as a bag of instances from these concepts, localized within image patches/regions. Defining an S-dimensional binary indicator vector s_i, such that $s_{ir} = 1$ and $s_{ik} = 0, k \neq r$, when the i^{th} image patch b_i depicts the semantic class r, the image can be represented as $I = \{s_1, s_2, \ldots, s_n\}$, where n is the total number of patches. Assuming that s_i is sampled from a multinomial distribution of parameter π_i, the log-likelihood of I is

$$\mathcal{L} = \log \prod_{i=1}^{n} \prod_{r=1}^{S} {\pi_{ir}}^{s_{ir}} = \sum_{i=1}^{N} \sum_{r=1}^{S} s_{ir} \log \pi_{ir}. \quad (1)$$

Since the semantic labels s_i for image regions are unknown, it is common to rely instead on the expected log-likelihood

$$E[\mathcal{L}] = \sum_{i=1}^{n} \sum_{r=1}^{S} E[s_{ir}] \log \pi_{ir} \quad (2)$$

where $E[s_{ir}] = P(r|b_i) = \pi_{ir}$ are the scene semantics for patch i, and (2) depends only on the multinomial parameters π_i. This is denoted the semantic multinomial (SMN) in [17]. SMNs are computed by applying a classifier, trained on the semantics of $\mathcal{V}$, to the image patches b_i, and using the resulting posterior class probabilities as π_i. This is illustrated in Figure 2, for a CNN classifier. Each patch is mapped into the probability simplex, denoted the semantic space $\mathcal{S}$ in Figure 1. The image is finally represented by the SMN collection $I = \{\pi_1, \ldots, \pi_n\}$. This is the BoS.

Throughout this work, we use ImageNet classes as $\mathcal{V}$ and object recognition CNNs to estimate the π_i. For efficient BoS extraction, the CNN is implemented as a fully convolutional network, generating the BoS with a single forward pass per image. This requires changing fully connected into 1x1 convolutional layers. The receptive field of a fully convolutional CNN can be altered by reshaping the size of the input image. E.g. for 512x512 images, the fully convolutional implementation of [1] extracts SMNs from 128x128 pixel patches 32 pixels apart. Figure 3 illustrates the high quality of the resulting semantics. Recognizers of the "bed", "window" and "quilt" objects exhibit are highly active in the regions where they appear in a bedroom scene.

3 Semantic Embedding

To design the invariant embedding $\mathcal{E}$ we rely on the Fisher vector (FV) [27, 47]. In this section, we review the FV and discuss its computation using the EM algorithm.

3.1 Fisher Vectors

Images are frequently represented by a bag of descriptors $\mathcal{D} = \{\pi_1, \ldots, \pi_n\}$ sampled independently from some generative model $p(\pi;\theta)$. An embedding is used to map this representation into a fixed-length vector suitable for classification. A popular mapping is the gradient (with respect to θ) of the log-likelihood $\nabla_\theta L(\theta) = \frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta)$ evaluated at a *background model* θ^b. This is known as the *Fisher score* of θ. This gradient vector is often normalized by $\mathcal{F}^{-\frac{1}{2}}\nabla_\theta L(\theta)$, where $\mathcal{F}^{-\frac{1}{2}}$ is the square root of the Fisher information matrix $\mathcal{F}$ of $p(\pi;\theta)$. This is the FV of $\mathcal{D}$ [27, 48].

Since, for independent sampling, $\log p(\mathcal{D};\theta)$ is a sum of the log-likelihoods $\log p(\pi_i;\theta)$, the FV is a vector of pooling operators, whose strength depends on the expressiveness of the generative model $p(\pi;\theta)$. The FV based on a large Gaussian mixture model (GMM) is known to be a strong descriptor of image context [47, 49]. However, for models like GMMs or hidden Markov models, the FV can have various implementations of very different complexity and deriving an efficient implementation is not always easy. We next show that Fisher scores can be trivially obtained using a single step of the expectation maximization (EM) algorithm commonly used to learn such models. This unifies the EM and FV computations, enabling the use of many efficient implementations previously uncovered in the EM literature to implement FVs.

3.2 Fisher Scores from EM

Consider the log-likelihood of $\mathcal{D}$ under a latent-variable model $\log p(\mathcal{D};\theta) = \log\int p(\mathcal{D},z;\theta)dz$ of hidden variable z. Since the left-hand side is independent of the hidden variable, this can be written in an alternate form [50]

$$\begin{aligned}
&\log p(\mathcal{D};\theta) = \log p(\mathcal{D},z;\theta) - \log p(z|\mathcal{D};\theta)\\
=\;& \int q(z)\log p(\mathcal{D},z;\theta)dz - \int q(z)\log p(z|\mathcal{D};\theta)dz\\
=\;& \int q(z)\log p(\mathcal{D},z;\theta)dz - \int q(z)\log q(z)dz\\
&+\int q(z)\log\frac{q(z)}{p(z|\mathcal{D};\theta)}dz\\
=\;& Q(q;\theta) + H(q) + KL(q||p;\theta) \qquad (3)
\end{aligned}$$

where $Q(q;\theta)$ is the "Q" function of the EM algorithm, $q(z)$ a general probability distribution, $H(q)$ its differential entropy and $KL(q||p;\theta)$ the Kullback Liebler divergence between the posterior $p(z|\mathcal{D};\theta)$ and $q(z)$. It follows that

$$\frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta) = \frac{\partial}{\partial\theta}Q(q;\theta) + \frac{\partial}{\partial\theta}KL(q||p;\theta) \qquad (4)$$

where

$$\frac{\partial}{\partial\theta}KL(q||p;\theta) = -\int\frac{q(z)}{p(z|\mathcal{D};\theta)}\frac{\partial}{\partial\theta}p(z|\mathcal{D};\theta)dz. \qquad (5)$$

Each iteration of the EM algorithm chooses the q distribution $q(z) = p(z|\mathcal{D};\theta^b)$, where θ^b is a reference parameter vector (parameter estimates from previous iteration). In this case,

$$\begin{aligned}
Q(q;\theta) &= \int p(z|\mathcal{D};\theta^b)\log p(\mathcal{D},z;\theta)dz \qquad (6)\\
&= E_{z|\mathcal{D};\theta^b}[\log p(\mathcal{D},z;\theta)] \qquad (7)
\end{aligned}$$

and

$$\begin{aligned}
\frac{\partial}{\partial\theta}KL(q||p;\theta)\Big|_{\theta=\theta^b} &= -\int\frac{p(z|\mathcal{D};\theta^b)}{p(z|\mathcal{D};\theta^b)}\frac{\partial}{\partial\theta}p(z|\mathcal{D};\theta)\Big|_{\theta=\theta^b}dz\\
&= -\frac{\partial}{\partial\theta}\int p(z|\mathcal{D};\theta)\Big|_{\theta=\theta^b}dz = 0.
\end{aligned}$$

It follows from (4) that

$$\frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta)\Big|_{\theta=\theta^b} = \frac{\partial}{\partial\theta}Q(p(z|\mathcal{D};\theta^b);\theta).\Big|_{\theta=\theta^b} \qquad (8)$$

In summary, the Fisher score $\nabla_\theta L(\theta)|_{\{\theta=\theta^b\}}$ of background model θ^b is the gradient of the Q-function of EM evaluated at reference model θ^b. The computation of the Fisher score thus simplifies into the two steps of EM. First, the E step computes the Q function $Q(p(z|\mathcal{D};\theta^b);\theta)$ at the reference θ^b. Second, the M-step evaluates the gradient Q with respect to θ at $\theta = \theta^b$. Since latent variable models are learned with EM, efficient implementations of these steps are usually already available in the literature, e.g. the Baum-Welch algorithm used to learn hidden Markov models [51]. Hence, the connection to EM makes the derivation of the Fisher score trivial for most models of interest.

4 Semantic Fisher vectors

In this section, we discuss the encoding of the Image BoS into semantic FVs.

4.1 Gaussian Mixture FVs

The most popular model in the FV literature is the GMM of diagonal covariance [27, 47, 49], here denoted the variance-GMM. Under this generative model, a mixture component z_i is first sampled from a hidden variable z of categorical distribution $p(z=k) = w_k$. A descriptor π_i is then sampled from the Gaussian component $p(\pi|z=k) \sim G(\pi,\mu_k,\sigma_k)$ of mean μ_k and variance σ_k, which is a diagonal matrix. Both hidden and observed variables are sampled independently. The Q function is

$$\begin{aligned}
Q(p(z|\mathcal{D};\theta^b);\theta) &= \textstyle\sum_i E_{z_i|\pi_i;\theta^b}[\log p(\pi_i,z_i;\theta)]\\
&= \textstyle\sum_i E_{z_i|\pi_i;\theta^b}\left[\sum_k I(z_i,k)\log p(\pi_i,k;\theta)\right]\\
&= \textstyle\sum_{i,k} p(k|\pi_i;\theta^b)\log p(\pi_i|z_i=k;\theta)w_k \qquad (9)
\end{aligned}$$

where $I(.)$ is the indicator function. The probabilities $p(k|\pi_i;\theta^b)$ are the only quantities computed in the E-step. The M-step then computes the gradient with respect to parameters $\theta = \{\mu_k, \sigma_k\}$

$$\mathcal{G}_{\mu_k^d}(\mathcal{I}) = \frac{\partial}{\partial \mu_k^d} Q = \sum\nolimits_i p(k|\pi_i) \left(\frac{\pi_i^d - \mu_k^d}{(\sigma_k^d)^2} \right) \quad (10)$$

$$\mathcal{G}_{\sigma_k^d}(\mathcal{I}) = \frac{\partial}{\partial \sigma_k^d} Q = \sum\nolimits_i p(k|\pi_i) \left[\frac{(\pi_i^d - \mu_k^d)^2}{(\sigma_k^d)^3} - \frac{1}{\sigma_k^d} \right], \quad (11)$$

where Q indicates the log-likelihood of the image and π_i^d is the d^{th} entry of vector π_i.

These are also the components of the Fisher score, when evaluated using a reference model $\theta^b = \{\mu_k^b, \sigma_k^b\}$ learned (with EM) from all training data. The FV is obtained by scaling the gradient vectors by an approximate Fisher information matrix, as detailed in [47]. This leads to the following mean and variance components of the GMM-FV

$$\mathcal{V}_{\mu_k}(\mathcal{I}) = \frac{1}{n\sqrt{w_k}} \sum\nolimits_i p(k|\pi_i) \left(\frac{\pi_i - \mu_k}{\sigma_k} \right) \quad (12)$$

$$\mathcal{V}_{\sigma_k}(\mathcal{I}) = \frac{1}{n\sqrt{2w_k}} \sum\nolimits_i p(k|\pi_i) \left[\frac{(\pi_i - \mu_k)^2}{\sigma_k^2} - 1 \right]. \quad (13)$$

For a single Gaussian component of zero mean, (12) reduces to the average pooling operator. For mixtures of many components, (12) implements a pooling operator per component, restricting each operator to descriptors of large probability $p(k|\pi_i)$ under the component. The FV can also implement other pooling operations, e.g. capturing higher order statistics as in (13). Many variations of the GMM-FV have been proposed to enable discriminative learning [52], spatial feature encoding [53] or non-iid mixture modeling [54]. However, for low-level features and large enough mixtures, the classical FV of (12) and (13) is still considered state-of-the-art.

4.2 Dirichlet Mixture FVs

The variance-GMM is a default model for low-level visual descriptors [43, 47, 55, 56]. However, SMNs, which inhabit a probability simplex, are more naturally modeled by the Dirichlet mixture (DMM). This follows from the fact that the Dirichlet distribution is the most popular model for probability vectors [57]. For example, it is widely used for text modeling [58], as a prior of the latent Dirichlet allocation model, and for SIFT based image categorization [54, 59]. The DMM was previously used to model "theme" based SMNs in [24]. It is defined as

$$P(\pi|\{\alpha_k, w_k\}_{k=1}^K) \quad = \quad \frac{1}{Z(\alpha_k)} e^{\sum_l (\alpha_{kl}-1)\log \pi_l}. \quad (14)$$

where α_k is the Dirichlet parameter of the k^{th} mixture component and w_k denotes the mixture weight. $Z(\alpha_k)$ is the normalizing constant $\frac{\gamma(\sum_l \alpha_{kl})}{\prod_l \gamma(\alpha_{kl})}$, where $\gamma(x) = \int_0^\infty x^{t-1} e^{-x} dx$ is the Gamma function. The generative process is as follows. A mixture component z is sampled from a categorical distribution $p(z = k) = w_k$. An observation π is then sampled from the selected Dirichlet component $P(\pi|\alpha_k)$. This makes the observation π a multinomial distribution that resides on the probability simplex.

TABLE 1: Parameters of (18) for the GMM-FV and DMM-FV. $\mathcal{F}_k$ is given by (17), $h_k(\pi;\mu,\Sigma,w)$ by (21) and $q_k(\pi;\alpha,w)$ by (22).

	GMM-FV	DMM-FV
θ_k	μ_k, $\Sigma_k = \sigma_k I$	α_k
$\nu(\pi_i)$	π_i	$\log(\pi_i)$
$\xi(\theta_k)$	μ_k	$f(\alpha_k) = \psi(\alpha_k) - \psi(\sum_l \alpha_{kl})$
$\gamma(\theta_k)$	$\frac{1}{\sqrt{w_k}\sigma_k}$	$\mathcal{F}_k^{-1/2}$
$p(k\|\pi)$	$h_k(\pi;\mu,\Sigma,w)$	$q_k(\pi;\alpha,w)$

The EM algorithm for DMM learning has Q function

$$Q(p(z|\mathcal{D};\alpha^b);\alpha) = \sum_{i,k} h_{ik} \left(\sum_l \alpha_{kl} \log \pi_l - \log Z(\alpha_k) \right) \quad (15)$$

where h_{ik} is the posterior probability $p(k|\pi_i;\theta_b)$ of the sample π_i being under the k^{th} components and we ignore terms that do not depend on the α parameters[1]. The expression for the Fisher scores $\mathcal{G}_{\alpha_k}(\mathcal{I}) = \frac{\partial L(\theta)}{\partial \alpha_k}$ of a DMM is

$$\mathcal{G}_{\alpha_k}(\mathcal{I}) = \frac{1}{n} \sum_{i=1}^N h_{ik} \left(\log \pi_i - \psi(\alpha_k) + \psi(\sum_l \alpha_{kl}) \right), \quad (16)$$

where $\psi(x) = \frac{\partial \gamma(x)}{\partial x}$. As usual in the FV literature [27], we approximate the Fisher information $\mathcal{F}$ by the component-wise block diagonal matrix

$$\begin{aligned} (\mathcal{F}_k)_{lm} &= E\left[-\frac{\partial^2 \log P(\pi|\{\alpha_k, w_k\}_{k=1}^K)}{\partial \alpha_{kl} \partial \alpha_{km}} \right] \\ &\approx w_k \left(\psi'(\alpha_{kl})\delta(l,m) - \psi'(\sum_l \alpha_{kl})) \right) \end{aligned} \quad (17)$$

where $\delta(l,m) = 1$ if $l = m$. The DMM Fisher vector for image I is finally obtained from (16) and (17) as $\mathcal{F}_k^{-1/2} \mathcal{G}_{\alpha_k}(\mathcal{I})$.

4.3 The logSMN-FV

To understand the benefits and limitations of the GMM-FV and DMM-FV it helps to investigate their relationships. Consider the application of the two FVs to the set of SMNs $\{\pi_1, \ldots, \pi_n\}$ extracted from image $\mathcal{I}$. In both cases, the FV can be written as

$$\mathcal{V}_{\theta_k}(\mathcal{I}) = \frac{1}{n} \sum_{i=1}^N p(k|\pi_i) \gamma(\theta_k) \left(\nu(\pi_i) - \xi(\theta_k) \right) \quad (18)$$

where $\gamma(.), \nu(.)$, and $\xi(.)$ are defined in Table 1. This is a pooling mechanism that combines four operations: $p(k|\pi_i)$ *assigns* the SMNs π_i to the components k, $\nu(.)$ *embeds* each SMN into the space where pooling takes place, $\xi(.)$ defines a *centroid* with respect to which the residuals $\nu(\pi_i) - \xi(\theta_k)$ are computed, and $\gamma(\theta_k)$ *scales* or normalizes that residual.

There are three main differences between the FVs. First, while the GMM-FV lacks an embedding, the DMM-FV uses $\nu(.) = \log(.)$. Second, while the GMM-FV has independent parameters to define centroids (μ_k) and scaling (σ_k), the parameters of the DMM-FV are coupled, since the centroids $f(\alpha_k)$ and the scaling parameter $\mathcal{F}_k^{-1/2}$ are both determined by the DMM parameters α_k. Finally, the two FVs differ in the assignments $p(k|\pi_i)$ and centroids $\xi(\theta_k)$. However, the centroids are closely related. Assuming a background mixture model learned from a training set $\{\pi_1^b, \ldots, \pi_N^b\}$

1. Gradients w.r.t mixture weights w_k are less informative than w.r.t other parameters and ignored in the FV literature [27, 47, 49].

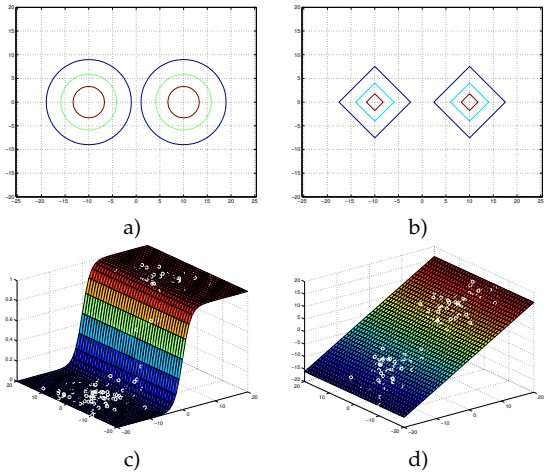


Fig. 4: Top: Two classifiers in an Euclidean space $\mathcal{X}$, a) L_2 and b) L_1 merics. Bottom: c) projection of a sample from a) into the semantic space $\mathcal{S}$ (only $P(y=1|x)$ shown). d) natural parameter space mapping of c).

they are the parameters that set (12) and (16) to zero upon convergence of EM. This leads to the expressions

$$\mu_k = \frac{\sum_i p(k|\pi_i^b)\pi_i^b}{\sum_i p(k|\pi_i^b)} \quad (19)$$

$$f(\alpha_k) = \frac{\sum_i p(k|\pi_i^b)\log \pi_i^b}{\sum_i p(k|\pi_i^b)}. \quad (20)$$

The differences in the assignments are also mostly of detail, since

$$h_k(\pi;\mu,\Sigma,w) = \frac{w_k e^{||\pi-\mu_k||_{\Sigma_k}}}{\sum_j w_j e^{||\pi-\mu_j||_{\Sigma_j}}} \quad (21)$$

$$q_k(\pi;\alpha,w) = \frac{w_k e^{(\alpha_k-1)^T\log\pi}}{\sum_j w_j e^{(\alpha_j-1)^T\log\pi}} \quad (22)$$

are both softmax type non-linearities. For both assignments and centroids, the most significant difference is the use of the $\log\pi$ embedding in the DMM-FV.

In summary, the two FVs differ mostly in the use of the $\log\pi$ embedding by the DMM-FV and the greater modeling flexibility of the GMM-FV, due to the availability of independent localization (centroid) μ_k and scale σ_k parameters. This suggests the possibility of combining the strengths of the two FVs by applying the GMM-FV *after* this embedding. We refer to this as the logSMN-FV

$$\mathcal{V}_{\mu_k}(\mathcal{I}) = \frac{1}{n\sqrt{w_k}}\sum\nolimits_i p(k|\pi_i)\left(\frac{\log\pi_i-\mu_k}{\sigma_k}\right). \quad (23)$$

Our experiments, see Section 6.2 (Table 2), show that this simple transformation leads to a large improvement in classification accuracy.

4.4 FVs in Natural Parameter Space

The gains of the log embedding can be explained by the non-Euclidean nature of the probability simplex. For some insight on this, consider the two binary classification problems of Figures 4 a) and b). In a) the two classes are Gaussian, in b) Laplacian. Both problems have class-conditional distributions $P(x|y)\propto\exp\{-d(x,\mu_y)\}$ where $Y\in\{0,1\}$ is the class label and $d(x,\mu)=||x-\mu||_p$, with $p=1$ for Laplacian and $p=2$ for Gaussian. Figures 4 a) and b) show the iso-contours of the probability distributions under the two scenarios. Note that the two classifiers use very different metrics.

The posterior distribution of class $Y=1$ is, in both cases,

$$\pi(x)=P(y=1|x)=\sigma(d(x,\mu_0)-d(x,\mu_1)) \quad (24)$$

where $\sigma(v)=(1+e^{-v})^{-1}$ is the sigmoid. Since this is very non-linear, the projection $x\rightarrow(\pi(x),1-\pi(x))$ of the samples x_i into the semantic space destroys the Euclidean structure of the original spaces $\mathcal{X}$. This is illustrated in c), which shows the posterior surface and the projections $\pi(x_i)$ for Gaussian x_i. In this space, the shortest path between two samples is not a line. The sigmoid also makes the posterior surfaces of the two problems very similar. The surface of the Laplacian problem in b) is visually indistinguishable from c). In summary, Euclidean classifiers with two very different metrics transform the data into highly non-Euclidean semantic spaces that are almost indistinguishable. This reduces the effectiveness of modeling probabilities directly with GMMs or DMMs, producing weak FV embeddings.

The problem can be avoided by noting that SMNs are the parameters of the multinomial, which is a member of the exponential family of distributions

$$P_S(s;\pi)=h(s)g(\pi)\exp\left(\eta^T(\pi)T(s)\right), \quad (25)$$

where $T(s)$ is denoted a sufficient statistic. In this family, the re-parametrization $\nu=\eta(\pi)$ makes the (log) probability distribution *linear* in the sufficient statistic

$$P_S(s;\nu)=h(s)g(\eta^{-1}(\nu))\exp\left(\nu^T T(s)\right). \quad (26)$$

This is called the *natural parameterization* of the distribution. Under this parametrization, the multinomial log-likelihood of the BoS in (2) yields a natural parameter vector $\nu_i=\eta(E\{s_i\})$ for each patch x_i, instead of a probability vector. For the binary semantics of Figure 4, $\eta(.)$ is the logit transform $\nu=\log\frac{\pi}{1-\pi}$. This maps the high-nonlinear semantic space of Figure 4 c) into the linear space of d), which preserves the Euclidean structure of a) and b). Hence, while the variance-GMM is not well matched to the geometry of the probability simplex where π is defined, it is a good model for distributions on the (Euclidean) natural parameter space defined by $\nu(\pi)$.

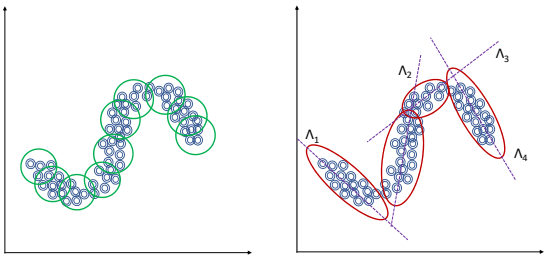
Similarly, for multiclass semantics, the mapping from multinomial to natural parameter space is a one-to-one transformation into a space with Euclidean structure. In fact, the multinomial of parameter vector $\pi=(\pi_1,\ldots,\pi_S)$ has three possible natural parametrization

$$\nu_k^{(1)} = \log\pi_k \quad (27)$$

$$\nu_k^{(2)} = \log\pi_k + C \quad (28)$$

$$\nu_k^{(3)} = \log\frac{\pi_k}{\pi_S} \quad (29)$$

where ν_k and π_k are the k^{th} entries of ν and π, respectively. The fact that logSMNs implement $\nu^{(1)}$ explains the good performance of the logSMN-FV. However, the existence of two alternative embeddings raises the question of whether this is the best natural parameter space embedding for the BoS produced by a CNN. Note that, under $\nu^{(2)}$, $\pi_k=\frac{1}{C}e^{\nu_k}$ defines a probability vector if and only if $C=\sum_i e^{\nu_i}$.



a) variance GMM b) MFA

Fig. 5: Modeling data on a manifold. The variance-GMM requires many Gaussians. By fitting locally linear subspaces, the MFA requires few Gaussians.

Hence, the mapping from $\nu^{(2)}$ to π is the softmax function commonly implemented at the CNN output. This implies that CNNs learn to optimally discriminate data in the natural parameter space defined by $\nu_k^{(2)}$ and, for CNN semantics, $\nu_k^{(2)}$ should enable better scene classification.

4.5 The MFA-FV

The models introduced so far mostly disregard semantic feature covariance. The Dirichlet mixture is, by design, incapable of modeling second order statistics. As usual in the FV literature [27, 47, 49], the GMM-based FVs assume a diagonal covariance per mixture component. While standard for SIFT descriptors [47], this is not suitable for the much higher dimensional CNN features, more likely to populate a low-dimensional manifold of the ambient semantic space. As illustrated in Figure 5, the variance-GMM requires many components to cover such a distribution. While a full covariance GMM could be substantially more efficient, covariance modeling is difficult in high dimensions. The data available for transfer learning is rarely sufficient to learn full covariances.

In this work, we explore approximate covariance modeling using mixtures of factor analyzers (MFAs) [28]. As illustrated in Figure 5, the MFA approximates a non-linear data manifold by a set of locally linear subspaces. Each mixture component generates Gaussian data in a low dimensional latent space, which is then projected linearly into the high dimensional observation space. This is a low rank approximation of the full covariance Gaussian, which can be learned with the small amounts of data available for transfer learning. It generates high-dimensional covariance statistics that can be exploited by a FV for better classification.

4.5.1 MFA Fisher scores

A factor analyzer (FA) models high dimensional observations $x \in \mathbb{R}^D$ in terms of latent "factors" $z \in \mathbb{R}^R$ defined on a low-dimensional subspace $R << D$ [28]. Specifically, $x = \Lambda z + \epsilon$, where Λ is the factor loading matrix and ϵ additive noise. Factors z are distributed as $G(z, 0, I)$ and noise as $G(\epsilon, 0, \psi)$, where ψ is a diagonal matrix. It can be shown that x follows a Gaussian distribution $G(x, 0, S)$ of covariance $S = \Lambda\Lambda^T + \psi$. Since this is a full covariance matrix, the FA is better suited for high dimensional data than a Gaussian of diagonal covariance.

The MFA extends the FA so as to allow a piece-wise linear approximation of a non-linear data manifold. It has two hidden variables: a discrete variable s, $p(s = k) = w_k$, which determines the mixture assignments and a continuous latent variable $z \in \mathbb{R}^R$, $p(z|s = k) = G(z, 0, I)$, which is a low dimensional projection of the observation variable $x \in \mathbb{R}^D$, $p(x|z, s = k) = G(x, \Lambda_k z + \mu_k, \psi)$. Hence, the k^{th} MFA component is a FA of mean μ_k and subspace defined by Λ_k. As illustrated in Figure 5, the MFA components approximate the distribution of x by a set of sub-spaces. The MFA can be learned with an EM algorithm of Q function

$$\begin{aligned} Q(\theta^b; \theta) = & \\ = & \sum_i E_{z_i, s_i|x_i;\theta^b}\left[\sum\nolimits_k I(s_i, k) \log p(x_i, z_i, s_i = k; \theta)\right] \\ = & \sum_{i,k} h_{ik} E_{z_i|x_i;\theta^b}\Big[\log G(x_i, \Lambda_k z_i + \mu_k, \psi) \\ + & \log G(z_i, 0, I) + \log w_k\Big] \end{aligned}$$

where $h_{ik} = p(s_i = k|x_i; \theta^b)$. After some simplifications, defining

$$S_k^b = \Lambda_k^b \Lambda_k^{b^T} + \psi^b \quad (30)$$

$$\beta_k^b = \Lambda_k^{b^T} \left(S_k^b\right)^{-1}, \quad (31)$$

the E step reduces to computing

$$h_{ik} = p(k|x_i; \theta^b) \propto w_k^b G(x_i, \mu_k^b, S_k^b) \quad (32)$$

$$E_{z_i|x_i;\theta^b}[z_i] = \beta_k^b (x_i - \mu_k^b) \quad (33)$$

$$E_{z_i|x_i;\theta^b}[z_i z_i^T] = \beta_k^b (x_i - \mu_k^b)(x_i - \mu_k^b)^T \beta_k^{b^T} \quad (34)$$

$$- \left(\beta_k^b \Lambda_k^b - I\right). \quad (35)$$

The M-step computes the Fisher scores of $\theta = \{\mu_k^b, \Lambda_k^b\}$. After some algebraic manipulations, these can be written as

$$\mathcal{G}_{\mu_k}(\mathcal{I}) = \sum\nolimits_i h_{ik} \{S_k^b\}^{-1} \left(x_i - \mu_k^b\right) \quad (36)$$

$$\mathcal{G}_{\Lambda_k}(\mathcal{I}) = \sum\nolimits_i h_{ik} \Big[\{S_k^b\}^{-1} (x_i - \mu_k^b)(x_i - \mu_k^b)^T \beta_k^{b^T} - \{S_k^b\}^{-1} \Lambda_k^b\Big] \quad (37)$$

For a detailed discussion of the Q function, the reader is referred to the EM derivation in [28]. Note that the scores with respect to the means are functionally similar to the first order residuals of (10). However, the scores with respect to the factor loading matrices Λ_k account for covariance statistics of the observations x_i, not just variances. We refer to (36) and (37) as the MFA Fisher scores (MFA-FS).

4.5.2 MFA Fisher Information

The MFA-FV is obtained by scaling the MFA-FS by the Fisher information matrix. As before, this is approximated by a block-diagonal matrix that scales the Fisher scores of the k^{th} mixture component by the inverse square-root of

$$\mathcal{F}_k = w_k Cov_k\left(\mathcal{G}_k(x)\right). \quad (38)$$

Here w_k is the weight of the k^{th} mixture, $\mathcal{G}_k(x)$ the data term of its Fisher score, and Cov_k the covariance with respect to the k^{th} mixture component. For the mean scores of (36) this is simply the component covariance S_k^b. For the factor loading scores it is the covariance of the data term

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2019.2921960, IEEE Transactions on Pattern Analysis and Machine Intelligence

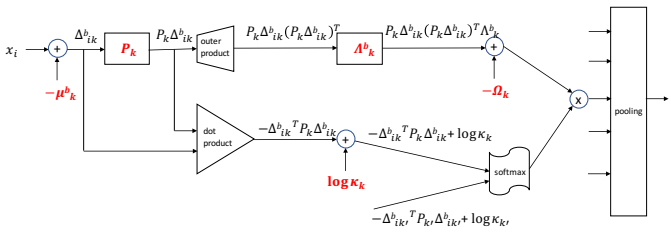


Fig. 6: The MFA-FS(Λ) layer implements (49) as a network layer. The bottom branch computes the posterior probability of (50). The top branch computes the remainder of the summation argument. Note that circles denote entry-wise operations, boxes implement matrix multiplications (weight layers), the outer product layer is similar to [36], and the dot-product layer a combination of elementwise multiplication and a sum. Expressions in red are parameters of the k^{th} MFA component, in black the computations made by the network.

of (37). This is a $D \times R$ matrix, whose entry (i, j) is the product of two Gaussian random variables

$$\begin{aligned} \mathcal{G}_k^{(i,j)}(x) &= f_i g_j \\ &= \lfloor \{S_k^b\}^{-1}(x-\mu_k^b) \rfloor_i \lfloor \beta_k^b (x-\mu_k^b) \rfloor_j \end{aligned}$$

where $\lfloor w \rfloor_i$ is the i^{th} element of vector w. The covariance matrix of the vectorized Fisher score is then

$$Cov_k(\mathcal{G}_k(x))_{(i,j),(l,m)} = E\left[f_i g_j f_l g_m\right] - E[f_i g_j] E[f_l g_m].$$

This can be simplified by using Isserlis' theorem, which states that, for zero-mean Gaussian random variables $\{x_1, x_2, x_3, x_4\}$, $E[x_1 x_2 x_3 x_4] = E[x_1 x_2] E[x_3 x_4] + E[x_1 x_3] E[x_2 x_4] + E[x_1 x_4] E[x_2 x_3]$. It follows that

$$\begin{aligned} &Cov_k(\mathcal{G}_k(x))_{(i,j),(l,m)} = \\ &= E\left[f_i g_m\right] E\left[f_l g_j\right] + E\left[f_i f_l\right] E\left[g_j g_m\right] \end{aligned} \tag{39}$$

with

$$\begin{aligned} E\left[f_i g_m\right] E\left[f_l g_j\right] &= \lfloor (S_k^b)^{-1} \Lambda_k^b \rfloor_{i,m} \lfloor (S_k^b)^{-1} \Lambda_k^b \rfloor_{l,j} \\ E\left[f_i f_l\right] E\left[g_j g_m\right] &= \lfloor (S_k^b)^{-1} \rfloor_{i,l} \lfloor \beta_k^b \Lambda_k^b \rfloor_{j,m} \end{aligned} \tag{40}$$

The Fisher scaling of the k^{th} MFA component is obtained by combining (38), (39) and (40).

Note that, like the GMM-FV, the MFA-FV can be applied with or without the embedding into natural parameter space. However, because it is still a Gaussian model, all arguments above suggest that it should be more effective when applied after the embeddings of (27)-(29).

5 Neural Network Embedding

The FVs above are implemented independently of the CNN used to extract the SMNs. The mixture model is learned from the extracted BoS and the FV derived from its parameters. In this section, we redesign the MFA-FS embedding as a CNN layer, to enable end-to-end training.

5.1 The MFA-FS Layer

To implement (36) and (37) in a CNN, we start by defining

$$\Delta_{ik}^b = x_i - \mu_k^b. \tag{41}$$

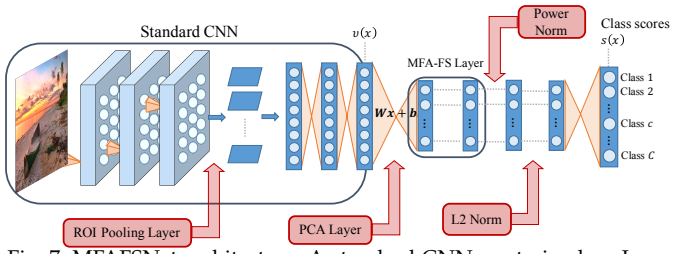


Fig. 7: MFAFSNet architecture. A standard CNN, pretrained on ImageNet, is used to extract a vector $\nu(x)$ of image features. The network is applied to image patches, which are combined with a RoI pooling layer. The vector $\nu(x)$ is fed to a dimensionality reduction layer, then to the MFA-FS layer, and finally power and L_2 normalized, before application of a linear classification layer.

Combining this with (30) and (31), (36) and (37) can be written as

$$\mathcal{G}_{\mu_k}(\mathcal{I}) = \sum_i p(k|x_i;\theta^b) \{S_k^b\}^{-1} \Delta_{ik}^b \tag{42}$$

$$\begin{aligned} \mathcal{G}_{\Lambda_k}(\mathcal{I}) &= -\sum_i p(k|x_i;\theta^b) \{S_k^b\}^{-1} [\Delta_{ik}^b \Delta_{ik}^{b^T} \{S_k^b\}^{-1} \Lambda_k^b - \Lambda_k^b] \\ &= -\sum_i p(k|x_i;\theta^b) \{S_k^b\}^{-1} \Delta_{ik}^b [\{S_k^b\}^{-1} \Delta_{ik}^b]^T \Lambda_k^b \\ &\quad + \sum_i p(k|x_i;\theta^b) \{S_k^b\}^{-1} \Lambda_k^b \end{aligned} \tag{43}$$

Since the k_{th} mixture component $p(x|s=k)$ has distribution $G(x, \mu_k, S_k^b)$, it follows that

$$\begin{aligned} p(k|x_i;\theta^b) &= \frac{w_k G(x_i;\mu_k^b, S_k^b)}{\sum_k w_k G(x_i;\mu_k^b, S_k^b)} \\ &= \frac{\frac{w_k}{|S_k^b|^{\frac{1}{2}}} \exp\{-\frac{1}{2}\Delta_{ik}^{b^T} S_k^{b^{-1}} \Delta_{ik}^b\}}{\sum_k \frac{w_k}{|S_k^b|^{\frac{1}{2}}} \exp\{-\frac{1}{2}\Delta_{ik}^{b^T} S_k^{b^{-1}} \Delta_{ik}^b\}} \end{aligned} \tag{44}$$

and denoting

$$P_k = S_k^{b^{-1}}, \tag{45}$$

$$\Omega_k = S_k^{b^{-1}} \Lambda_k^b, \tag{46}$$

$$\kappa_k = \frac{w_k}{|S_k^b|^{\frac{1}{2}}}, \tag{47}$$

finally leads to

$$\mathcal{G}_{\mu_k}(\mathcal{I}) = \sum_i p(k|x_i;\theta^b) P_k \Delta_{ik}^b \tag{48}$$

$$\mathcal{G}_{\Lambda_k}(\mathcal{I}) = -\sum_i p(k|x_i;\theta^b) \{P_k \Delta_{ik}^b (P_k \Delta_{ik}^b)^T \Lambda_k^b - \Omega_k\} \tag{49}$$

$$p(k|x_i;\theta^b) = \frac{\kappa_k \exp\{-\frac{1}{2}\Delta_{ik}^{b^T} P_k \Delta_{ik}^b\}}{\sum_{k'} \kappa_{k'} \exp\{-\frac{1}{2}\Delta_{ik'}^{b^T} P_{k'} \Delta_{ik'}^b\}} \tag{50}$$

Figure 6 shows how (49) can be implemented as a network layer. The bottom branch computes the posterior probability of (50). The top branch computes the remainder of the summation argument. The computation of (48) is similar. The bottom branch is identical, the top branch omits the operations beyond $P_k \Delta_{ik}^b$. However, because the benefits of this component are small, we only use the layer of Figure 6.

5.2 Network Architecture

The overall architecture of the MFAFSNet is shown in Figure 7. A model pretrained on ImageNet is used to extract

a vector $\nu(x)$ of image features. This network is applied to image patches, producing multiple feature maps per image to classify. When the patches are of a single scale, the model is converted to a fully convolutional network. For patches of multiple scales, the final pooling layer is replaced with a region-of-interest (ROI) pooling layer, which accepts feature maps of multiple sizes and produces a fixed size output. This is a standard practice in object detection [4, 60]. The feature vector $\nu(x)$ is dimensionality reduced by a fc layer of appropriate dimensions, and fed to the MFA-FS layer of Figure 6. Note that this layer pools multiple local features, corresponding to objects of different sizes and in different image locations, generating a single feature vector for the whole image. This is fed to a power and a L2 normalization layers, and finally to a linear classifier layer.

5.3 Loss Function

While the parameters μ_k^b, P_k, λ_k^b, Ω_k and $\log \kappa_k$ are learned by back-propagation, they must maintain their interpretation as statistical quantities. This requires that (30) and (45)-(47) hold. Some of these constraints do not need to be enforced. For example, since (47) is the only to involve w_k, there is a one to one relationship between $\log \kappa_k$ and w_k, independently of the value of $|S_k^b|^{\frac{1}{2}}$. In result, it is equivalent to learn w_k under the constraint of (47) or simply learn $\log \kappa_k$, which leads to a simpler optimization. A similar observation holds for (30), which is the only constraint on ψ^b. On the other hand, some of the relationships must be enforced to maintain the MFA-FS interpretation. These are (45), (46), and the symmetry of matrix P_k. They are enforced by adding regularization terms to the loss function. For training set $\mathcal{D} = \{(x_i, y_i)\}$ and classification loss $L_c(.)$, this leads to a loss function

$$\begin{aligned} L(\mathcal{D}) &= L_c(\mathcal{D}) + \lambda_1 \sum_k ||\Omega_k - P_k \Lambda_k^b||_F^2 \\ &+ \lambda_2 \sum_k ||P_k - P_k^T||_F^2 \end{aligned} \quad (51)$$

where $||A||_F$ is the Frobenius norm of A, and λ_1, λ_2 control the regularization strength. We use the hinge loss

$$L_c(\mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} \left[\max(0, 1 - I(y_i, k) s_k(x_i)\right]^2 \quad (52)$$

where $s(x)$ is the input to the softmax at the top of the network and $I(\cdot)$ the indicator function. This is for consistency with the FV literature, which is based on SVMs. Any other classification loss could be used.

6 Experimental Results

In this section, we present the results of an extensive evaluation of all the FVs discussed above.

6.1 Experimental set-up

All experiments are based on the MIT Indoor [31] and SUN [30] scene datasets. MIT Indoor consists of 100 images each from 67 indoor scene classes. Following the standard protocol, we use 80 images per class for training and the remaining 20 for testing. MIT SUN has about 100K images from 397 indoor and outdoor scene categories. It provides randomly sampled image sets each with 50 images per class for training as well as test. Performance, on both datasets, is reported as average per class classification accuracy.

TABLE 2: Comparison of FV encodings. SIFT-FV is a GMM-FV applied to a bag of SIFT features, and conv5-FV a GMM-FV applied at layer conv5.

Method	MIT Indoor	SUN
SIFT-FV	60.0	43.3
conv5-FV	61.4	-
SMN-FV	55.3	36.87
DMM-FV	58.8	40.86
logSMN-FV	**67.7**	**49.86**

TABLE 3: Ablation analysis on MIT Indoor, for the role of centroids: learned, transferred (T), or random (R).

FV	Accuracy
DMM	58.8
DMM(T)	58.4
DMM(R)	58.0
logSMN	68.5
logSMN(T)	**68.6**
logSMN(R)	61.3

For the FVs implemented independently of the object recognition CNN, the BoS is extracted with the CNNs of [1] or [61], pre-trained on ImageNet. The networks are applied convolutionally, generating a 1,000 dimensional SMN for roughly every 128x128 pixel region. The 1,000 dimensional probability vectors are reduced to 500 dimensions using PCA. These are the descriptors π_i used to build the FV. Unless otherwise noted, GMM-FVs, DMM-FVs, and logSMN-FVs use a 100-component mixture model. All FVs are power and L_2 normalized [27], and classified with a linear SVM. Unless otherwise noted, the MFA is computed with $K = 50$ components and latent space dimension $R = 10$.

The MFAFSNet is implemented with the object recognition networks of [1, 61, 62], trained on ImageNet. The vector $\nu(x)$ of Figure 7 is the input to the softmax at the top of these networks, i.e. we use the $\nu^{(2)}$ embedding of (28). A vector $\nu(x)$ is produced per $l \times l$ image patch, mapped into 500 dimensions by the PCA layer, and fed to the MFA-FS layer. Images are resized, reducing smaller side to 512-pixels and maintaining aspect ratio. Three patch sizes, $l \in \{96, 128, 160\}$ were used, producing between 590 and 1000 patches per image. The MFA-FS layer uses $K = 50$ mixture components and $R = 10$ subspace dimensions, outputting a vector of $500 \times 50 \times 10$ dimensions. The last fc layer is learned with a learning rate of 0.001, while 0.00001 is used for all others. Momentum and weight decay were set to 0.9 and 0.0005 respectively and the network trained on 10 epochs. For simplicity, we set $\lambda_1 = \lambda_2 = \lambda$ in (51).

6.2 Benefits of Natural Parameter Space

We begin by comparing FVs that embed the BoS distribution, the GMM-FV (denoted SMN-FV) and DMM-FV, to the logSMN-FV, which embeds natural parameter descriptors. For completeness, we also tested the classical SIFT-FV of [47] and a GMM-FV that embeds the features of CNN convolutional layer 5, denoted conv5-FV. The CNN is that of [1]. All GMM-FVs use a reference GMM θ^b and are computed with (12). We ignore the variance component of (13), since it did not improve the performance. The DMM-FV uses a reference DMM $\theta^b = \{\alpha_k^b, w_k^b\}_{k=1}^K$ and is computed with (16) and (17). The logSMN-FV is computed with (23).

Table 2 summarizes the performance of all methods. The SMN-FV is a poor classifier, underperforming the SIFT and conv5 FVs by $5 - 6\%$ points. This is surprising, given the now well documented advantage of CNN over SIFT features and the increased discriminant power of SMN over

TABLE 4: Ablation analysis on MIT Indoor for the role of scaling and assignments. Tilde indicates transferred parameters.

Mixture Model	FV Encoding		
	Scaling	Assignment	Accuracy
DMM	$\mathcal{F}_k^{-1/2}(\alpha)$	$q_k(\pi;\alpha,w)$	58.8
		$h_k(\log\pi;\tilde{\mu},\Sigma,w)$	57.7
	$\frac{1}{\sqrt{w_k}\sigma_k}$	$q_k(\pi;\alpha,w)$	67.1
		$h_k(\log\pi;\tilde{\mu},\Sigma,w)$	68.6
GMM	$\frac{1}{\sqrt{w_k}\sigma_k}$	$h_k(\log\pi;\mu,\Sigma,w)$	68.5
		$q(\pi;\tilde{\alpha},w)$	68.7
	$\mathcal{F}_k^{-1/2}(\tilde{\alpha})$	$h_k(\log\pi;\mu,\Sigma,w)$	57.7
		$q(\pi;\tilde{\alpha},w)$	58.4

conv5 features. The DMM-FV outperforms the SMN-FV but still underperforms the other two approaches. The limited improvement over the SMN-FV can be partially explained by the somewhat surprising observation that best performance is achieved with a single component mixture. This is unlike GMM, which tends to improve significantly with the number of components, and suggests that the DMM lacks the flexibility to fit complex distributions. On the other hand, the logSMN-FV leads to a staggering improvement in classification accuracy, beating the SMN-FV by around 9% on both datasets! This is surprising, since it is identical to the SMN-FV, up to the log mapping into the natural parameter space. It is also the first BoS FV to outperform the SIFT-FV.

6.3 Ablation Analysis

To better understand these differences, we did an ablation analysis of the parameters of Table 1 on MIT Indoor.

Centroids: Consider a DMM $\{w_k,\alpha_k\}_{k=1}^K$ learned from SMNs π_i and a GMM $\{w_k,\mu_k,\Sigma_k\}_{k=1}^K$ learned from logSMNs $\log\pi_i$. Since, in this case, (19) and (20) are identical, the GMM centroids can be used to construct a DMM and vice versa. For the former, it suffices to map the Gaussian means $\{\mu_k\}_{k=1}^K$ in $\log\pi$ space to the Dirichlet parameters, by solving $\mu_k = \psi(\tilde{\alpha}_k) - \psi(\sum_k \tilde{\alpha}_k)$ for $\tilde{\alpha}_k$. The GMM $\{w_k,\mu_k,\Sigma_k\}_{k=1}^K$ can then be mapped to a DMM $\{w_k,\tilde{\alpha}_k\}_{k=1}^K$, using the estimated $\tilde{\alpha}_k$ and copying weights w_k. For the latter, a GMM is anchored at the DMM centroids $\tilde{\mu}_k = f_k(\alpha)$, the weights w_k copied from the latter, and the Gaussian covariances Σ_k set to the global covariance Σ of the training data. We refer to this process as model transfer.

We trained a DMM $\{w_k,\alpha_k\}_{k=1}^K$ and a GMM $\{w_k,\mu_k,\Sigma\}_{k=1}^K$ in $\log\pi$ space. A second Dirichlet model, DMM(T), of parameters $\{w_k,\tilde{\alpha}_k\}_{k=1}^K$ was obtained by transferring the GMM means. Similarly, a second Gauss mixture, GMM(T), of parameters $\{w_k,\tilde{\mu}_k,\Sigma\}_{k=1}^K$ was transferred from the DMM. These models were compared to a DMM of random centroids, DMM(R), and a GMM of random centroids in $\log\pi$ space and covariances Σ_k set to the global covariance Σ of the training data, GMM(R). The mixture weights of DMM(R) and GMM(R) were set to uniform.

Table 3 summarizes the performance of all FVs. The DMM-FV is always substantially weaker than the logSMN-FV. Nevertheless, the logSMN(T) result shows that the DMM can be used to estimate the mixture parameters. In fact, the GMM with centroids transferred from the DMM has slightly better performance than the original GMM. On the other hand, learning the logSMN GMM and transferring to the DMM-FV form has weak performance. These results show that the *accuracy of the estimation of the mixture model is much less important than the FV encoding itself.* This is further confirmed by the results of the models with random parameters. For the weaker DMM-FV, random centroids perform as well as original or transferred centroids. In fact, all these FVs underperform the stronger logSMN-FV encoding with no learning (random centroids).

In summary, it is more important to use a stronger FV encoding (logSMN-FV) than carefully learn parameters of a weaker FV encoding (DMM-FV). However, for the stronger logSMN-FV, there is a non-trivial gain in using learned centroids. On the other hand, it does not matter if they are the GMM centroids or transfered from a DMM. All of this follows from the equality of (19) and (20) for the logSMN-FV and the DMM-FV. Since this equality also holds for the residuals $\nu(\pi_i) - \xi(\theta_k)$ of (18), the two FV encodings only differ in the scaling $\gamma(\theta_k)$ and assignments ($h_k(\log\pi;\mu,\Sigma,w)$ vs $q_k(\pi;\alpha,w)$).

Scaling and assignments: The impact of the two factors can be studied by starting from the logSMN-FV and changing the scaling function to $\mathcal{F}_k^{-1/2}$ or the assignment function to $q_k(\pi;\tilde{\alpha},w)$. The combination of the two modifications leads to the DMM(T)-FV. Conversely, it is possible to start from the DMM-FV and change each of the functions or both, in which case we obtain the logSMN(T)-FV. Table 4 shows that the assignment function has a small effect. In all cases, the gain of changing assignment is at most 1.5%, and no assignment function is clearly better. What seems to matter is that it *matches the scaling function.* The DMM performs better with $q_k(.)$ for the original $\mathcal{F}_k^{-1/2}$ scaling, but with $h_k(.)$ for Gaussian scaling. The GMM has equivalent performance with the two assignments for Gaussian scaling, but performs better with $q_k(.)$ for $\mathcal{F}_k^{-1/2}$ scaling.

On the other hand, *the scaling function has a significant effect on classification accuracy.* For both mixtures and assignment functions, Gaussian scaling is 10% better than $\mathcal{F}_k^{-1/2}$ scaling. These results are not totally surprising, since Gaussian scaling has an additional degree of freedom (variance σ_k), while $\mathcal{F}_k^{-1/2}$ is determined by the α parameters already used to determine centroids. The normalization of Gaussian scaling, which produces normal residuals of zero mean and unit variance, is akin to batch normalization [63], which is well known to benefit learning. It is also known that the most important effect of Fisher information scaling is the decorrelation of the FV, which improves its performance significantly [47]. On the other hand, (17) shows that the scaling matrix $\mathcal{F}_k$ has a restrictive structure, in that its off-diagonal elements are equal to $-\psi'(\sum_l \alpha_{kl})$. Hence, $\mathcal{F}_k$ resides in a subspace of the space of symmetric positive definite matrices $\mathbb{S}_d^+$ and affords very few degrees of freedom (roughly equal to the dimensionality of α).

Since scaling is determined by the Fisher information and this defines the local metric on the tangent space $\mathcal{T}_{\theta^b}$ to the model manifold, these results suggest that the Dirichlet manifold is not suited for classification. It is, however, interesting that classification is so strongly affected by the choice of model manifold. This is particularly surprising because, as shown in Table 2, Gaussian scaling is not a top performer in the absence of the $\log\pi$ embedding. In summary, while effective classification requires logSMNs, the modeling manifold is well captured by the GMM.

TABLE 5: Accuracy of GMM-FVs implemented with the natural parameter embeddings of (27)-(29).

NP embedding	MIT Indoor	SUN
$\nu^{(1)}$	67.7	50.87
$\nu^{(2)}$	**68.5**	**51.17**
$\nu^{(3)}$	67.6	50.47

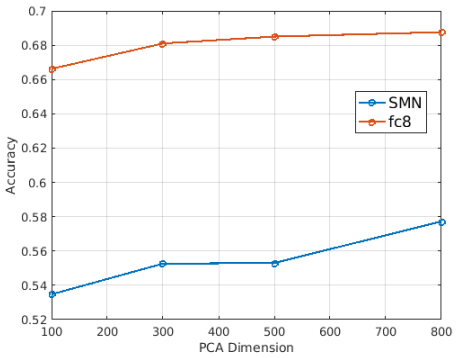


Fig. 8: Performance variation of the $\nu^{(2)}$-FV and the SMN-FV with PCA dimension.

6.4 Natural Parameter Embedding

The vastly superior performance of the logSMN-FV in Table 2 suggests that the FV should be computed in natural parameter space. In fact, it corresponds to the natural parameter transformation of (27). We next compared the transformations of (27)-(29). Table 5 summarizes the performance of the FV implemented with each mapping. $\nu^{(2)}$ has the best performance, followed by the $\nu^{(1)}$ (the logSMN-FV) and $\nu^{(3)}$. This is consistent with the fact that the softmax CNN is trained to maximize discrimination in the space of $\nu^{(2)}$, as discussed in Section 4.4, confirming that 1) probability modeling is difficult in the simplex, and 2) natural parameter transformations alleviate the problem. The performance of the $\nu^{(2)}$ embedding does not vary drastically with the dimensionality of the PCA transformation used before FV encoding. As shown in fig 8, even with 100 dimensional PCA projection, the $\nu^{(2)}$-FV achieves an accuracy of 66%, which is much higher than the SMN-FV.

Finally, to ensure that the gains of the FV are not just due to the use of a log non-linearity, we applied the log transformation to the activations of the penultimate CNN layer (often referred to as fc7) and $\nu^{(2)}$ features in (28), that already reside in the NP space. Rather than a gain, this resulted in a substantial decrease in performance (58% with an log fc7-FV vs 65.1% with an fc7-FV and 60.97% with a $\log \nu^{(2)}$-FV vs 68.5% with a $\nu^{(2)}$-FV, on MIT Indoor scenes). This was expected, since the role of log is natural parameter transformation and the argument does not apply for spaces other than the probability simplex.

6.5 Comparison to previous embeddings

Other embeddings have been proposed for the classification of semantic vectors. [19] projects SMNs on the great circle, using a square root embedding $\sqrt{\pi}$. The non-Euclidean nature of the simplex and the non-linearity of its geodesics are noted as a major difficulty for SMN based classification. The square root embedding was also used in [64] for SIFT descriptors. Rather than L_2 normalization, the authors propose L_1 normalization of the SIFT histogram to produce a probability vector. The SIFT probabilities are then transformed into "Root-SIFT" descriptors and encoded with a GMM-FV. This achieved moderate improvements over standard SIFT. We applied the square root embedding to the SMNs, achieving 58.95% accuracy on MIT Indoor and 40.6% on SUN. This is close to the DMM-FV results of Table 2, but drastically inferior to the $\nu^{(2)}$-FV results of Table 5. The inability of the root embedding to replicate the linearization of Figure 4 d) limits the performance of the GMM-FV after the embedding.

An alternative embedding of probability descriptors was introduced in [65]. It uses a $\log$ transformation on L_1 normalized SIFT descriptors and is also inspired by Dirichlet sufficient statistics. A SIFT probability vector p is subjected to a von-Mises transformation $\nu^{(5)} = \frac{\log(p+\epsilon)-\log \epsilon}{\|\log(p+\epsilon)-\log \epsilon\|_2}$ and encoded by a GMM-FV. This was shown to improve on the rootSIFT-FV of [64]. Except for its L_2 normalization, the von-Mises embedding is somewhat similar to natural parameter transformation $\nu^{(3)} = \log p_i - \log p_N$. When we applied it to SMNs, it achieved 63.4% on MIT Indoor and 46.1% on SUN. This was better than the square root embedding, but underperformed all three embeddings of (27)-(29). The most likely reason is the projection onto the great circle (L_2 normalization), which may work for SIFT but does not help for CNN semantics.

6.6 Covariance Modeling

We next evaluate the importance of covariance modeling, by studying the MFA-FV derived from a MFA learned in the natural parameter space. Unless otherwise noted, results refer to MIT Indoor.

6.6.1 *Importance of Covariance Modeling*

The MFA was compared to the variance-GMM, using the set-up of Section 6.1, embedding $\nu^{(2)}$ of (28), $K = 50$ components and latent space dimension $R = 10$. Table 6 compares the GMM-FV(μ) of (12), the GMM-FV(σ) of (13), the MFA-FS(μ) of (36), the MFA-FS(Λ) of (37), the MFA-FV(μ) which scales the MFA-FS(μ) with $(w_k S_k^b)^{-1/2}$ and the MFA-FV(Λ) which scales the MFA-FS(Λ) with the Fisher information of (38)-(40). GMM-FV(σ) was the weakest performer, underperforming the GMM-FV(μ) by more than 10%. This difference is much larger for CNN features than for the lower dimensional SIFT features [27] and the reason why CNN FVs only consider gradients w.r.t. means [32, 66].

The improved covariance modeling of the MFA solves this problem. The MFA-FS(Λ) significantly outperforms both GMM-FVs and the MFA-FS(μ). A related covariance modeling was used in [67] to obtain FVs w.r.t. Gaussian means and local subspace variances (covariance eigenvalues). In our experiments, this subspace variance FV (60.7% on MIT Indoor) outperformed the variance GMM-FV(σ) but was clearly inferior to the MFA-FS(Λ), which captures full covariance. In summary, full covariance modeling appears to be essential for FV-style pooling of CNN features.

On the other hand, the MFA-FV(μ) and MFA-FV(Λ) have similar performance. Unlike the GMM-FV, where variance gradients are uninformative but Fisher scaling has large gains, the MFA-FV derives most of its power from the covariance gradients (MFA-FS(Λ)) and gains little from Fisher

TABLE 6: Scene classification accuracy of various embeddings.

Descriptor	MIT Indoor	SUN
Object-based		
GMM FV (μ)	66.08	50.01
GMM FV (σ)	53.86	37.71
MFA FS (μ)	67.68	51.43
MFA FS (Λ)	**71.11**	**53.38**
MFA FV (μ)	66.73	51.37
MFA FV (Λ)	70.89	53.56
Gist-based		
BoS-fc1	64.84	47.47
BoS-fc2	69.36	50.9
BoS-fc3	70.6	53.12

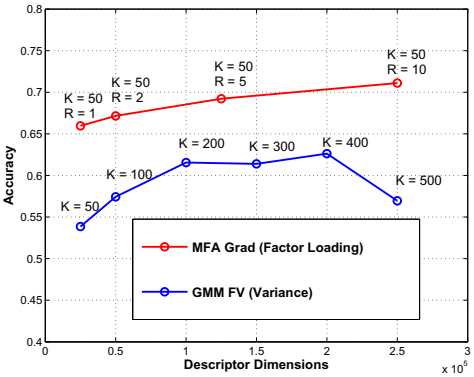


Fig. 9: Accuracy vs. descriptor size for MFA-FS(Λ) of 50 components and R factor dimensions and GMM-FV(σ) of K components.

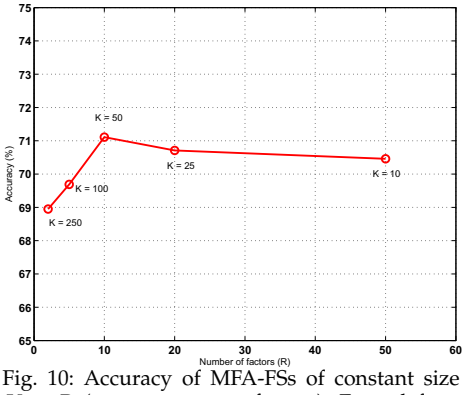


Fig. 10: Accuracy of MFA-FSs of constant size $K \times R$ (components vs. factors). From left to right, R incraeses while K decreases.

scaling. In fact, even the concatenation (MFA-FS(μ), MFA-FS(Λ)) gave small improvement ($\sim 1\%$), which does not justify the increased computation. We use the MFA-FS(Λ) alone in the following sections.

6.6.2 Covariance Modeling vs Subspace Dimensions

The comparison above is somewhat unfair because, for fixed number of components K, the GMM-FV has less parameters than the MFA-FS. Fig. 9 compares the GMM-FV(σ) and MFA-FS(Λ) when K varies in $\{50, \ldots, 500\}$ and the MFA latent space dimensions R in $\{1, \ldots, 10\}$. For comparable dimensions, the covariance based scores significantly outperform the variance statistics. Fixed-size MFA-FS (Λ) descriptors (250K dimensions) were next used to evaluate the relative importance of local covariance modeling (dimensionality R) and global flexibility of the mixture (components K). MFA models were learned with K decreasing from 250 to 10, each reduction in K being traded for an increase in R, from 2 to 50. As shown in Figure 10, classification accuracy increased steadily as K decreased from 250 to 50 (R increasing from 2 to 10). This shows that there is a trade-off between local covariance modeling and global manifold approximation, as suggested by Figure 5. With better covariance modeling, fewer mixture components are required to cover the manifold of semantic features. Obviously, if R is too large the number of components may not be enough to enable a global approximation of the manifold. It appears, however, that low subspace dimensionality is more costly than few components. For small R, even models with large K ($K = 250$) perform poorly. Best results are achieved when the MFA has a sufficient number of Gaussians that implement a reasonable linear approximation of the local manifold. In our experiments, this corresponded to $K = 50, R = 10$.

6.6.3 Gist Descriptors

The MFA-FS follows a tradition of embeddings that pool a bag-of-descriptors [27, 43, 44, 55]. The underlying i.i.d. assumptions make the embedding flexible, with no template-like rigidity. An alternative scene representation is a holistic "gist" descriptor, e.g., produced by a fully connected (fc) neural network layer. The MFA-FS was compared to gist embeddings based on a network of one or more fc layers, interspersed with layers of ReLu non-linearities. The BoS was used to produce a tensor of 10 x 10 x 1000 responses, containing 1000 dimensional $\nu^{(2)}$ descriptors extracted from roughly every 128x128 image region. An fc layer was then used to map the tensor into a 4096 dimensional vector and followed by a ReLU layer. Successive fc layers of 4096 input and 4096 output channels and ReLU stages were optionally added to create a deeper embedding. The final 4096 dimensional vector was fed to a linear classifier, trained with a scene classification loss. All fc layers were learned with "drop-out" of probability 0.2. The embeddings are denoted BoS-fc1 to BoS-fc3 based on the number of fc layers used.

A problem for this approach is the limited amount of transfer data. The number of parameters in the largest embedding was almost equal to that of the ImageNet CNN of [1, 2]. MIT Indoor is too small to train such an embedding, leading to a scene classification accuracy of 33%. To overcome the problem, we used the Places scene dataset [6], which contains 2.4 M training images of 200 scene categories. This, however, made the training of gist embeddings last several days on a GPU, as opposed to two hours for the MFA-FS. Nevertheless, as shown in Table 6, the BoS-fc embeddings still underperformed the MFA-FS.

6.7 MFAFSNet

We finish with a set of experiments on the MFAFSNet.

6.7.1 Relevance of statistical interpretation

A set of experiments was conducted to test the need to enforce the statistical interpretation of the MFAFSNet[2]. The first addressed parameter initialization, comparing random initialization of the MFA-FS parameters (zero mean Gaussian of standard deviation 0.01) to initialization with the MFA-FS (PCA matrix learned from all patches $\nu(x)$ and MFA layer learned by EM [28]). A strength $\lambda = 1$ was used in (51). Table 7 shows that random initialization was weaker by $2-3\%$ on MIT Indoor and $4-6\%$ on SUN. The importance of regularization was next investigated by varying λ. For small λ, the learning algorithm is free to ignore the MFA-FS constraints. For larger λ, the network has stronger statistical interpretation. Figure 11 shows an improvement of up to 1% when λ increases from 0.01 to 1. These experiments show that it is important to enforce the statistical interpretation of the MFAFSNet. In all remaining experiments we use MFA-FS initialization and $\lambda = 1$.

2. Results are reported for a single patch size of 96, but similar behavior was observed for other configurations.

TABLE 7: Effect of initialization on classification accuracy.

	MIT Indoor	SUN
	AlexNet	
Random	69.82	50.23
MFA-FS	71.44	54.14
	VGG-16	
Random	77.3	56.2
MFA-FS	80.3	62.51

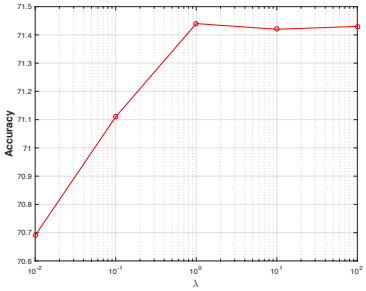


Fig. 11: Effect of regularization on classification accuracy.

6.7.2 Multi-scale and end-to-end learning

A set of experiments then investigated the impact of multiple patch sizes. Table 8 compares the accuracies of the MFA-FS(Λ) and MFAFSNet with 96x96, 128x128 and 160x160 patches, as well as their combination (3 scales). For the MFA-FS, the three vectors were concatenated. For the MFAFSNet, a mixture of bounding boxes of the three sizes was fed to the ROI pooling layer. This generated a shorter vector, better suited for the available GPU memory. The multi-scale combination achieved the best performance for all CNNs and datasets. This is not surprising, as it accounts for multiple object sizes within the scenes. The consistently better performance of the MFAFSNet, over the MFA-FS, also confirms the benefits of end-to-end learning.

6.7.3 Comparison to previous transfer-based methods

Various methods have been proposed to transfer ImageNet object classifiers to scenes [34, 37, 66, 68]. Since they only report results for MIT Indoor, we compare results for this dataset only on Table 9. The GMM FV of [68] uses convolutional features from AlexNet or VGG-16 extracted in a large multi-scale setting. [34] proposed a gradient representation based on sparse codes and reported results for a single patch scale of 128x128 and AlexNet features. An improved H-Sparse representation, combining multiple scales and VGG features was later proposed in [35]. The recent bilinear (BN) pooling method of [36] is similar to the MFA-FS in that it captures global second order descriptor statistics. The simplicity of these descriptors enables fine-tuning of CNN layers to scene classification. However, as shown in [37] for VGG-16 features, the results are clearly inferior to those of the MFA-FS without fine-tuning and about 5% worse than the MFAFSNet. [37] proposes to compress these bilinear statistics with trainable transformations. However, the resulting image representation of size $8K$ has accuracy inferior to combining the MFA-FS with a PCA of $5K$ dimensions. In summary, the MFA-FS and MFAFSNet are state of the art procedures for task transfer from object recognition (on ImageNet) to scene classification (on MIT Indoor/SUN). The closest competitor [68] combines CNN features in a massive multiscale setting (10 image sizes). The MFA-FS and MFAFSNet outperform it with only 3 scales.

6.7.4 Task vs. dataset transfer

The MFA based classifiers implement task transfer, using an object recognition network to classify scenes. This is an alternative to the standard dataset transfer, where a network trained to classify scenes is applied to a different scene dataset. This approach is simpler but much more intensive, requiring the collection and annotation of a large scene dataset. It was pursued in [6], which assembled a scene dataset (Places) of 2.4M images and used it to train scene classification CNNs. These were then transfered to MIT Indoor and SUN, by using the the CNN as a features extractor and linearly classifying these features.

TABLE 8: Classification accuracy as a function of patch size $p \times p$. 'All' denotes combination of three sizes.

		MIT Indoor				SUN			
		160	128	96	All	160	128	96	All
MFA-FS	AlexNet	69.8	71.1	70.5	73.6	52.4	53.4	53.5	56.0
MFAFSNet		70.1	71.9	71.5	**75.3**	52.6	54.5	54.2	**57.3**
MFA-FS	VGG-16	77.3	77.3	80.0	80.1	59.8	61.0	61.7	63.3
MFAFSNet		78.3	78.8	80.5	**81.3**	61.5	62.0	61.7	**64.8**
MFA-FS	ResNet-50	81.2	82.0	82.4	83.4	63.5	65.2	64.8	65.7
MFAFSNet		81.5	82.7	83.0	**84.0**	63.4	65.5	65.7	**66.3**

TABLE 9: Task transfer performance on MIT Indoor. 'All' denotes combined sizes $96, 128, 160$.

Method	Scales 128	Scales All	Scales 128	Scales All
	AlexNet		VGG	
MFAFSNet	**71.85**	**75.31**	**80.48**	**81.32**
MFA-FS	71.11	73.58	79.9	81.43
FV+FC [68]	-	71.6	-	81.0
Sp. Code [34, 35]	68.2	-	-	77.6
H-Sparse [35]	-	-	-	79.5
BN [37]	-	-	77.55	-
	VGG + dim. reduct.			
MFA-FS + PCA (5k)	-	-	**79.3**	-
BN (8k) [37]	-	-	76.17	-

TABLE 10: Task vs. dataset transfer. 'Both' referes to the combination of MFAFSNet with Places CNN.

	SUN	Indoor
	AlexNet	
Places	54.3	68.24
Places ft	56.8	72.16
MFAFSNet	57.29	75.31
Both	**64.47**	**80.49**
	VGG	
Places	61.32	79.47
Places ft	65.25	81.34
MFAFSNet	64.81	81.32
Both	**72.43**	**88.05**
	Resnet-50	
Places	63.51	79.05
Places ft	63.80	82.61
MFAFSNet	66.28	83.99
Both	**73.35**	**88.06**

Table 10 compares the performance of the two transfer approaches. Somewhat surprisingly, task transfer with the MFAFSNet *outperformed* dataset transfer with the pre-trained Places CNN, on both datasets, for all networks. We include an additional baseline, denoted Places ft, where in addition to large scale training on scenes, the CNN is fine-tuned to Indoor and SUN as well. The transfer based MFAFSNet beats this strong baseline for AlexNet and the deeper Resnet-50 architectures, and is only slightly worse for the VGG architecture. An ensuing question is whether there is any complementarity between the object-based MFAFSNet and the holistic representation learned by the Places CNN. This was tested by training a classifier on the concatenation of the two descriptors. As shown in Table 10, it lead to a substantial increase in performance $(6 - 8\%)$, suggesting that the representations are indeed *very complementary*. To the best of our knowledge, no method using these or deeper CNNs has reported better results on these datasets. This is detailed in Table 11, which compares results to recent scene classification methods in the literature. The MFAFSNet + Places combination is a state-of-the-art classifier with substantial gains over all other approaches.

7 Conclusion

This work makes several contributions to computer vision. First, we introduced a new task transfer architecture based on sophisticated pooling operators for CNN features, implemented under the FV paradigm. While good performance

TABLE 11: Performance of scene classification methods.

	Indoor	SUN		Indoor	SUN
AlexNet			VGG		
Without Places					
Sparse Cod. [34]	68.2	-	DAG-CNN [71]	77.5	56.2
VLAD [32]	68.88	51.98	Sparse Cod. [35]	77.6	-
Mid Level [72]	70.46	-	Compact BN [37]	76.17	-
FV+FC [68]	71.6	-	Full BN [37]	77.55	-
MFA-FS	73.58	55.95	H-Sparse [35]	79.5	-
MFAFSNet	75.01	57.15	FV+FC [68]	81.0	-
			MFA-FS	81.43	63.31
			MFAFSNet	82.66	64.59
With Places					
MetaClass [73]	78.9	58.11	Places+SF [38]	84.3	67.6
			LS-DHM [39]	83.75	67.56
MFA-FS	79.86	63.16	MFA-FS	87.23	71.06
MFAFSNet	**80.49**	**64.47**	MFAFSNet	**88.05**	**72.43**

was demonstrated for object-to-scene transfer, the architecture is applicable to any problems involving the transfer of a set of source semantics into a set of target semantics that are loose combinations of them. Image captioning [69] or visual question-answering [70] are examples of vision problems that could leverage such transfer.

Second, we demonstrated the importance of semantic representations for this type of transfer. While others had argued for this in the past [70], the semantic noise of pre-CNN semantic spaces prevented the implementation of effective semantic transfer systems. We have shown that the combination of the BoS produced by a CNN and a sophisticated transfer architecture enable state of the art performance in problems like scene classification. In fact, this transfer was shown to *outperform* the direct learning of CNNs from much larger scene datasets. While transfer learning has been pursued as a vehicle for efficient training, the results above indicate that task transfer could be essential to the solution of complex vision problems. This points towards modular vision systems and is an agreement with human cognition, which is highly modular and rich in interactions of modules specialized in different semantics.

Third, we have contributed evidence to the long standing debate on whether scene understanding is based on objects or gist. Here, the most significant finding was the amplitude of the gains of combining the two representations. While this could be due to sub-optimality of our object or gist-based solutions, it is unlikely that better training or larger datasets would suffice to overcome the large gap between the individual and joint performances. This makes intuitive sense, since an object-representation of scenes must combine localized detections in a manner invariant to object configurations. As we have shown, this requires very non-linear pooling operators that are complicated to learn. In the absence of explicit object supervision, a CNN could find it difficult to uncover them. On the other hand, a holistic gist component appears to be critical as well. For example, it accounts for the relative placement of objects in the scene.

ACKNOWLEDGMENTS

This work was supported by NSF awards IIS-1208522 and IIS-1637941 and GPU donations by Nvidia.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[2] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, vol. abs/1409.1556, 2014.

[3] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *CVPR*, 2015. [Online]. Available: http://arxiv.org/abs/1409.4842

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *CVPR*, 2014.

[5] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[6] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning Deep Features for Scene Recognition using Places Database." *NIPS*, 2014.

[7] A. Torralba and A. A. Efros, "Unbiased look at dataset bias," in *CVPR'11*, June 2011.

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, June 2009, pp. 248–255.

[9] K. He, X. Zhang, S. Ren, and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," in *ECCV*, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds. Cham: Springer International Publishing, 2014, pp. 346–361.

[10] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by between-class attribute transfer," in *CVPR*, June 2009, pp. 951–958.

[11] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid, "Label-embedding for image classification," *TPAMI*, vol. 38, no. 7, pp. 1425–1438, 2016.

[12] J. Liu, B. Kuipers, and S. Savarese, "Recognizing human actions by attributes," in *CVPR 2011*, June 2011, pp. 3337–3344.

[13] W.-X. Li and N. Vasconcelos, "Complex activity recognition via attribute dynamics," *IJCV*, vol. 122, no. 2, pp. 334–370, Apr 2017. [Online]. Available: https://doi.org/10.1007/s11263-016-0918-1

[14] M. Jain, J. C. van Gemert, and C. G. M. Snoek, "What do 15,000 object categories tell us about classifying and localizing actions?" in *CVPR*, June 2015, pp. 46–55.

[15] M. Jain, J. C. van Gemert, T. Mensink, and C. G. M. Snoek, "Objects2action: Classifying and localizing actions without any video example," *CoRR*, vol. abs/1510.06939, 2015.

[16] J. Vogel and B. Schiele, "Semantic modeling of natural scenes for content-based image retrieval," *IJCV*, vol. 72, no. 2, pp. 133–157, Apr. 2007. [Online]. Available: http://dx.doi.org/10.1007/s11263-006-8614-1

[17] N. Rasiwasia, P. Moreno, and N. Vasconcelos, "Bridging the gap: Query by semantic example," *Multimedia, IEEE Transactions on*, vol. 9, no. 5, pp. 923–938, 2007.

[18] Y. Su and F. Jurie, "Improving image classification using semantic attributes," *IJCV*, vol. 100, no. 1, pp. 59–77, 2012.

[19] R. Kwitt, N. Vasconcelos, and N. Rasiwasia, "Scene recognition on the semantic manifold," in *ECCV*, ser. ECCV'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 359–372.

[20] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *TPAMI*, vol. 36, no. 3, pp. 453–465, 2014.

[21] A. Bergamo and L. Torresani, "Classemes and other classifier-based features for efficient object categorization," *TPAMI*, p. 1, 2014.

[22] L.-J. Li, H. Su, Y. Lim, and F.-F. Li, "Object bank: An object-level image representation for high-level visual recognition," *IJCV*, vol. 107, no. 1, pp. 20–39, 2014.

[23] S. Kordumova, T. Mensink, and C. G. Snoek, "Pooling objects for recognizing scenes without examples," in *Proceedings of the 2016 ACM on international conference on multimedia retrieval*. ACM, 2016, pp. 143–150.

[24] N. Rasiwasia and N. Vasconcelos, "Holistic context models for visual recognition," *TPAMI*, vol. 34, no. 5, pp. 902–917, May 2012.

[25] L. Torresani, M. Szummer, and A. Fitzgibbon, "Efficient object category recognition using classemes," in *ECCV*, Sep. 2010, pp. 776–789. [Online]. Available: \url{http://research.microsoft.com/pubs/136846/TorresaniSzummerFitzgibbon-classemes-eccv10.pdf}

[26] A. Bergamo, L. Torresani, and A. Fitzgibbon, "Picodes: Learning a compact code for novel-category recognition," in *NIPS*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 2088–2096.

[27] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, ser. ECCV'10, 2010, pp. 143–156.

[28] Z. Ghahramani and G. E. Hinton, "The em algorithm for mixtures of factor analyzers," Tech. Rep., 1997.

[29] J. Verbeek, "Learning nonlinear image manifolds by global alignment of local linear models," *TPAMI*, vol. 28, no. 8, pp. 1236–1250, Aug. 2006.

[30] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *CVPR*, 2010, pp. 3485–3492.

[31] A. Quattoni and A. Torralba, "Recognizing indoor scenes," *CVPR*, vol. 0, pp. 413–420, 2009.

[32] Y. Gong, L. Wang, R. Guo, and S. Lazebnik, "Multi-scale orderless pooling of deep convolutional activation features," in *ECCV*, vol. 8695, 2014, pp. 392–407.

[33] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *CVPR*, June 2015.

[34] L. Liu, C. Shen, L. Wang, A. Hengel, and C. Wang, "Encoding high dimensional local features by sparse coding based fisher vectors," in *Advances in Neural Information Processing Systems 27*, 2014, pp. 1143–1151.

[35] L. Liu, P. Wang, C. Shen, L. Wang, A. van den Hengel, C. Wang, and H. T. Shen, "Compositional model based fisher vector coding for image classification," *CoRR*, vol. abs/1601.04143, 2016.

[36] T.-Y. Lin, A. RoyChowdhury, and S. Maji, "Bilinear cnn models for fine-grained visual recognition," in *International Conference on Computer Vision (ICCV)*, 2015.

[37] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, "Compact bilinear pooling," *CoRR*, vol. abs/1511.06062, 2015.

[38] S. H. Khan, M. Hayat, and F. Porikli, "Scene categorization with spectral features," in *CVPR*, 2017, pp. 5638–5648.

[39] S. Guo, W. Huang, L. Wang, and Y. Qiao, "Locally supervised deep hybrid model for scene recognition," *TIP*, vol. 26, no. 2, pp. 808–820, 2017.

[40] E. P. X. Li-Jia Li, Hao Su and L. Fei-Fei, "Object bank: A high-level image representation for scene classification and semantic feature sparsification," in *NIPS*, Vancouver, Canada, December 2010.

[41] N. Rasiwasia and N. Vasconcelos, "Scene classification with low-dimensional semantic spaces and weak supervision," in *CVPR*, 2008.

[42] E. H. Adelson, "On seeing stuff: the perception of materials by humans and machines," *Proc. SPIE*, vol. 4299, pp. 1–12, 2001. [Online]. Available: http://dx.doi.org/10.1117/12.429489

[43] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1–22.

[44] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *CVPR*, vol. 2, 2006, pp. 2169 – 2178.

[45] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *CVPR*, 2009.

[46] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014, pp. 818–833. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10590-1_53

[47] J. Sánchez, F. Perronnin, T. Mensink, and J. J. Verbeek, "Image classification with the fisher vector: Theory and practice," *IJCV*, vol. 105, no. 3, pp. 222–245, 2013.

[48] T. S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *NIPS*, 1999, pp. 487–493.

[49] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *CVPR*, 2007, pp. 1–8.

[50] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society, B, 39, 1-38*, 1977.

[51] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, Feb 1989.

[52] L. Getoor and T. Scheffer, Eds., *ICML*. Omnipress, 2011.

[53] J. Krapac, J. Verbeek, and F. Jurie, "Modeling Spatial Layout with Fisher Vectors for Image Categorization," in *ICCV*. Barcelona, Spain: IEEE, Nov. 2011, pp. 1487–1494. [Online]. Available: http://hal.inria.fr/inria-00612277

[54] R. Cinbis, J. Verbeek, and C. Schmid, "Image categorization using fisher kernels of non-iid image models," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, 2012, pp. 2184–2191.
[55] N. Vasconcelos and A. Lippman, "A probabilistic architecture for content-based image retrieval," in *CVPR*, 2000, pp. 216–221.
[56] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *CVPR*, jun 2010, pp. 3304–3311. [Online]. Available: http://lear.inrialpes.fr/pubs/2010/JDSP10
[57] T. P. Minka, "Estimating a dirichlet distribution," Tech. Rep., 2000.
[58] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, January 2003. [Online]. Available: http://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf
[59] L. Fei-Fei and P. Perona, "A bayesian hierarchical model for learning natural scene categories," in *CVPR*, vol. 2, jun. 2005, pp. 524 – 531 vol. 2.
[60] R. Girshick, "Fast r-cnn," in *ICCV*, December 2015.
[61] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.
[62] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.
[63] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015, pp. 448–456.
[64] J. Delhumeau, P. H. Gosselin, H. Jégou, and P. Pérez, "Revisiting the vlad image representation," in *ACM Multimedia*, 2013, pp. 653–656.
[65] T. Kobayashi, "Dirichlet-based histogram feature transform for image classification," in *CVPR*, June 2014.
[66] M. Dixit, S. Chen, D. Gao, N. Rasiwasia, and N. Vasconcelos, "Scene classification with semantic fisher vectors," in *CVPR*, June 2015.
[67] M. Tanaka, A. Torii, and M. Okutomi, "Fisher vector based on full-covariance gaussian mixture model," *IPSJ Transactions on Computer Vision and Applications*, vol. 5, pp. 50–54, 2013.
[68] M. Cimpoi, S. Maji, I. Kokkinos, and A. Vedaldi, "Deep filter banks for texture recognition, description, and segmentation," *IJCV*, 2015.
[69] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *ICML*, 2015, pp. 2048–2057.
[70] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, and D. Parikh, "Vqa: Visual question answering," in *ICCV*, 2015, pp. 2425–2433.
[71] S. Yang and D. Ramanan, "Multi-scale recognition with dag-cnns," in *ICCV*, December 2015.
[72] Y. Li, L. Liu, C. Shen, and A. van den Hengel, "Mid-level deep pattern mining," in *CVPR*, June 2015.
[73] R. Wu, B. Wang, W. Wang, and Y. Yu, "Harvesting discriminative meta objects with deep cnn features for scene classification," in *ICCV*, December 2015.

Mandar Dixit Mandar Dixit received his PhD from the University of California at San Diego in 2018. He is currently a researcher in the computer vision technology group at Microsoft, Redmond. His research interests include in large scale visual recognition, transfer learning and meta learning.

Yunsheng Li Yunsheng Li received his Master degreee from the University of California at San Diego in 2018. He is currently a Ph.D. student in the Statistical Visual Computing Lab of UC San Diego. His research interests include transfer learning and meta learning.

Nuno Vasconcelos Nuno Vasconcelos received the licenciatura in electrical engineering and computer science from the Universidade do Porto, Portugal, and the MS and PhD degrees from the Massachusetts Institute of Technology. He is a Professor in the Electrical and Computer Engineering Department at the University of California, San Diego, where he heads the Statistical Visual Computing Laboratory. He has received a NSF CAREER award, a Hellman Fellowship, several best paper awards, and has authored more than 150 peer-reviewed publications. He has been Area Chair of multiple computer vision conferences, and is currently an Associate Editor of the IEEE Transactions on PAMI. In 2017, he was ellected Fellow of the IEEE.