UNIVERSITY OF CALIFORNIA, SAN DIEGO

Semantic transfer with deep neural networks

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy
in
Electrical Engineering (Intelligent Systems, Robotics, and Control)

by

Mandar Dixit

Committee in charge:

Professor Nuno Vasconcelos, Chair
Professor Manmohan Chandraker
Professor Kenneth Kreutz-Delgado
Professor Gert Lanckriet
Professor Zhuowen Tu

2017

DEDICATION

Dedicated to my mother

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my sincerest gratitude to my advisor Prof. Nuno Vasconcelos. It was under his guidance, that I learned how to think about complex problems, how to ask the right questions and how to conduct research. I would also like to thank the members of my doctoral committee, Prof. Kenneth Kreutz-Delgado, Prof. Gert Lanckriet, Prof. Zhuowen Tu and Prof. Manmohan Chandraker for their enlightening lectures, invaluable discussions and advice all of which helped me progress in my research.

I have had the privilege to know and work with many talented people while at the Statistical Visual Computing Lab (SVCL): Dashan Gao, Antoni Chan, Nikhil Rasiwasia, Vijay Mahadevan, Hamed Masnadi-Shirazi, Sunhyoung Han, Weixin Li, Ehsan Saberian, Jose Costa Pereira, Kritika Murlidharan, Can Xu, Song Lu, Si Chen, Zhaowei Cai, Bo Liu, Yingwei Li, Pedro Morgado and Yunsheng Li. I am very thankful to have met all of them during my stay at UCSD. In the initial years of my PhD, the advice I received from Dr. Nikhil Rasiwasia and Dr. Vijay Mahadevan was helpful in getting me started as a researcher in computer vision. I had several stimulating discussions, along the way, with Dr. Weixin Li, Dr. Ehsan Saberian and Dr. Jose Costa Pereira. In the middle years of my PhD, I had the good fortune of collaborating with Dr. Dashan Gao and learning from his vast experience. Some of the younger members of the lab have also helped me a lot in my past projects. Among them, I am particularly thankful to Si Chen and Yunsheng Li who are co-authors on a couple of papers with me.

Outside of SVCL, Dr. Roland Kwitt of the University of Salzburg has been my collaborator for many years. Dr. Kwitt and I continue to work together on some very interesting topics. One of our joint projects has recently resulted in a high impact publication. I am very grateful for his contributions to my research as well as my understanding of various aspects of computer vision. I would also like to thank Dr. Marc Neithammer of the University of North Carolina at Chapel Hill, who advises Dr. Kwitt and I on our projects from time to time.

My stay at UCSD was made enjoyable by the company of many fellow Indian students, some of whom became my very good friends. I am thankful, in particular, to Aman Bhatia, Siddharth Joshi and Joshal Daftari with whom I shared many cherishable moments.

In the third year of my PhD, I met my dear Varshita, who has, since, become my wife and my best friend. Throughout this arduous journey, she has been a constant source of encouragement for me. During the most trying of times, it was her unwavering support that kept me going. I couldn't thank her more for all that she has done for me.

Finally I would like to thank my family back in India. I owe my parents, Dilip and Jyoti Dixit, an enormous debt of gratitude for the unconditional love and encouragement that they have provided me over the years. I would also like to thank my younger brother Aniruddha with whom I spent many years of carefree childhood.

The text of Chapter II, is based on material as it appears in: M. Dixit, S. Chen, D. Gao, N. Rasiwasia and N. Vasconcelos, "Scene classification with semantic Fisher vectors", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, 2015 and M. Dixit, Y. Li and N. Vasconcelos, "Bag-of-Semantics representation for object-to-scene transfer", [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material. The author would like to thank Mr. Si Chen, Dr. Dashan Gao and Dr. Nikhil Rasiwasia for their helpful contributions to this project.

The text of Chapter III, is based on material as it appears in: M. Dixit and N. Vasconcelos, "Object based scene representations using Fisher scores of local subspace projections", Neural Information Processing Systems (NIPS), Barcelona, Spain, 2016. and M. Dixit, Y. Li and N. Vasconcelos, "Bag-of-Semantics representation for object-to-scene transfer", [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material. The author would also like to thank Dr. Weixin Li for helpful discussions during this project.

The text of Chapter IV, is based on material as it appears in: M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, "AGA: Attribute-Guided Augmentation", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, 2017 and M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, "Data augmentation with attribute trajectory transfer", [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material. The author would like to thank Dr. Roland Kwitt and Dr. Marc Neithammer for their helpful contributions to this project.

VITA

2007	Bachelor of Technology ECE, Visvesvaraya National Institute of Technology, Nagpur, India
2009	Master of Technology EE, Indian Institute of Technology, Kanpur, India
2009–2017	Research Assistant Statistical and Visual Computing Laboratory Department of Electrical and Computer Engineering University of California, San Diego
2015	Master of Science Electrical and Computer Engineering, University of California, San Diego
2017	Doctor of Philosophy Electrical and Computer Engineering, University of California, San Diego

PUBLICATIONS

M. Dixit, Y. Li and N. Vasconcelos, Bag-of-Semantics representation for object-to-scene transfer. To be submitted for publication, *IEEE Trans. on Pattern Analysis and Machine Intelligence.*

M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, Data augmentation with attribute trajectory transfer. To be submitted for publication, *IEEE Trans. on Pattern Analysis and Machine Intelligence.*

Y. Li, M. Dixit and N. Vasconcelos, Deep scene image classification with the MFAFVNet. Accepted for publication in *IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy, 2017.

M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, AGA: Attribute-Guided Augmentation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, Hawaii, 2017.

M. Dixit and N. Vasconcelos, Object based scene representations using Fisher scores of local subspace projections. In *Proc. Neural Information Processing Systems (NIPS)*, Barcelona, Spain, 2016.

M. George, M. Dixit, G. Zogg and N. Vasconcelos, Semantic clustering for robust fine-grained scene recognition. In *Proc. European Conference on Computer Vision (ECCV)*, Amsterdam, Netherlands, 2016.

M. Dixit, S. Chen, D. Gao, N. Rasiwasia and N. Vasconcelos, Scene classification with semantic Fisher vectors. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Boston, 2015.

M. Dixit*, N. Rasiwasia* and N. Vasconcelos, Class specific simplex-latent Dirichlet allocation for image classification. In *Proc. IEEE International Conference on Computer Vision (ICCV)*, Sydney, Australia, 2013. ($*$) - indicates equal contribution

M. Dixit, N. Rasiwasia and N. Vasconcelos, Adapted Gaussian models for image classification. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Colorado Springs, 2011.

ABSTRACT OF THE DISSERTATION

Semantic transfer with deep neural networks

by

Mandar Dixit
Doctor of Philosophy in Electrical Engineering
(Intelligent Systems, Robotics, and Control)
University of California, San Diego, 2017
Professor Nuno Vasconcelos, Chair

Visual recognition is a problem of significant interest in computer vision. The current solution to this problem involves training a very deep neural network using a dataset with millions of images. Despite the recent success of this approach on classical problems like object recognition, it seems impractical to train a large scale neural network for every new vision task. Collecting and correctly labeling a large amount of images is a big project in itself. The process of training a deep network is also fraught with excessive trial and error and may require many weeks with relatively modest hardware infrastructure. Alternatively one could leverage the information already stored in a trained network for several other visual tasks using *transfer learning*.

In this work we consider two novel scenarios of visual learning where knowledge transfer is affected from off-the-shelf convolutional neural networks (CNNs).

In the first case we propose a holistic scene representation derived with the help of pre-trained object recognition neural nets. The object CNNs are used to generate a *bag of semantics* (BoS) description of a scene, which accurately identifies object occurrences (semantics) in image regions. The BoS of an image is, then, summarized into a fixed length vector with the help of the sophisticated *Fisher vector* embedding from the classical vision literature. The high *selectivity* of object CNNs and the natural *invariance* of their semantic scores facilitate the transfer of knowledge for holitistic scene level reasoning. Embedding the CNN semantics, however, is shown to be a difficult problem. Semantics are probability multinomials that reside in a highly non-Euclidean simplex. The difficulty of modeling in this space is shown to be a bottle-neck to implementing a discriminative Fisher vector embedding. This problem is overcome by reversing the probability mapping of CNNs with a *natural parameter* transformation. In the natural parameter space, the object CNN semantics are efficiently combined with a Fisher vector embedding and used for scene level inference. The resulting *semantic Fisher vector* achieves state-of-the-art scene classification indicating the benefits of BoS based object-to-scene transfer.

To improve the efficacy of object-to-scene transfer, we propose an extension of the Fisher vector embedding. Traditionally, this is implemented as a natural gradient of Gaussian mixture models (GMMs) with diagonal covariance. A significant amount of information is lost due to the inability of these models to capture covariance information. A mixture of Factor analyzers (MFAs) are used instead to allow efficient modeling of a potentially non-linear data distribution in the semantic manifold. The Fisher vectors derived using MFAs are shown to improve substantially over the GMM based embedding of object CNN semantics. The improved transfer-based semantic Fisher vectors are shown to outperform even the CNNs trained on large scale scene datasets.

Next we consider a special case of transfer learning, known as few-shot learning, where the training images available for the new task are very few in number (typically less than 10). Extreme scarcity of data points prevents learning a generalize-able model even in the rich feature space of pre-trained CNNs. We present a novel approach of *attribute guided data augmentation* to solve this problem. Using an auxiliary dataset of object images labeled with 3D depth and pose, we learn trajectories of variations along these *attributes*. To the training examples in a few-shot dataset, we transfer these learned attribute trajectories and generate synthetic data points. Along with the original few-shot examples, the additional synthesized data can also be used for the target task. The proposed guided data augmentation strategy is shown to improve both few-shot object recognition and scene recognition performance.

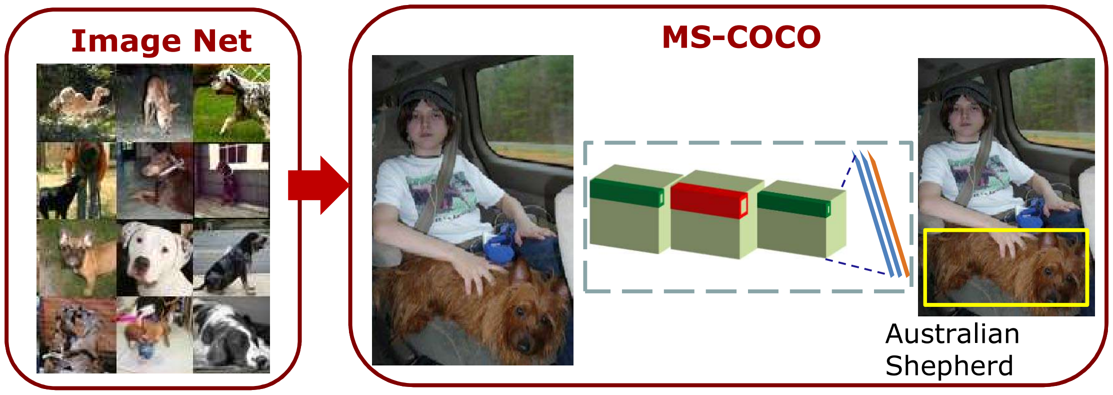
Chapter I

Introduction

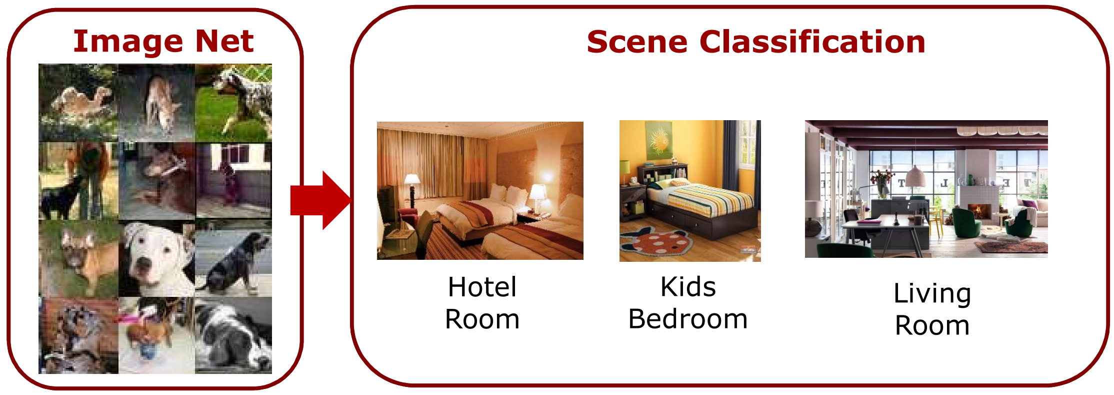
I.A Visual Recognition

Ability to recognize visual semantics such as objects, people and stuff [1] in scenes is critical for any autonomous intelligent system, e.g. a self driving car or a self navigating robot. The development of visual recognition systems, therefore, receives a significant amount of attention in computer vision community. Earlier proposals of visual recognition, relied heavily on carefully *designed* feature extractors that summarized low-level edge or texture information within images [57, 15, 2]. Edge based descriptors were extracted from image regions and often subject to non-linear encodings [14, 91, 63] and subsequent pooling to generate a reasonably invariant image representation. This representation was used with a discriminative classifier to perform various tasks such as scene classification, object recognition and object detection with a reasonable degree of success [50, 8, 64, 73, 24, 15]. Low-level edge and texture features, however, have a limited discriminative power as well as invariance. Templates of gradient orientations are largely unable to detect and describe meaningful semantics such as objects or object-parts that may be related to the high-level visual tasks of interest. Features such as SIFT and HoG, therefore, present a significant bottleneck for the visual recognition systems that rely on them.

An alternative to feature design, was the technique of visual feature learning or *deep learning*, where a sequential hierarchy of filters or templates were learned end-to-end specifically to optimize the performance on a given high-level task [51]. The filter units in these *deep networks* were highly non-linear and learned with strong supervision using the technique called *back-propagation*. Recently, due to the availability of large scale labeled datasets like ImageNet [17], deep neural networks have achieved major breakthroughs in visual recognition. Krizhevsky et. al. [45] were able to successfully train a deep convolutional neural network (CNN) using millions of images from ImageNet and achieve remarkable results on object recognition. Simonyan et. al. [76] reduced their error by more than half, with the



a) Object recognition to Object detection



b) Object recognition to Scene classification

Figure I.1: Types of Transfer Learning. a) depicts an example of Across dataset transfer, b) depicts an example of Across domain transfer.

help of a CNN with an even deeper architecture. More recently, He et. al. [35] have claimed recognition accuracies higher than those of human experts on object recognition using a CNN that is hundreds of layers deep.

The successes of deep learning seem to have rendered the feature design frameworks obsolete. Today the best approach to build an accurate visual recognition system is to i) collect millions of labeled images, and ii) train a CNN that is deep enough. Given the large number of possible recognition tasks, using this recipe for each one of them seems unfeasible. Collecting datasets with tens of millions of images and having them labeled using experts is a big project in itself. The configuration of a deep neural network may also require extensive train-and-error and the training may require months to finish on normal hardware. Instead of training a CNN for each new task, therefore, it may be worthwhile to develop techniques of knowledge transfer from the CNNs that are already trained for other tasks.

I.B Transfer Learning

The remarkable performance of deep CNNs can be attributed to a high semantic selectivity exhibited by their units. For instance filters in the higher layers of an object recognition CNNs are found to detect relevant semantics such as faces and object parts [93]. The feature responses of such CNNs, therefore, are clearly more discriminative than edge-based histograms, and, at the same time, generic enough to be used in other vision tasks.

Many recent works have have leveraged the publicly available, ImageNet trained, object recognition CNNs for other related tasks such as object detection [30, 29, 71], fine-grained recognition [54] and object localization [95] with reasonable success. Since these proposals affect knowledge transfer within the same visual domain (of objects) but across different datasets, we refer to their framework as *across-dataset* transfer. An example of this is depicted in fig Figure I.1 a) where the transfer occurs between ImageNet object recognition and MS-COCO object detection. Across dataset transfers are achieved by gently adapting the ImageNet CNN on the new dataset using a modified loss and a few iterations of back-propagation. This technique is commonly refered to as *finetuning* in the recent literature [30]. Achieving knowledge transfer across dissimilar domains, however, is not as straightforward. Consider for example, the case depicted in fig Figure I.1 b) where a transfer is desired between object recognition CNNs and holistic scene level inference. Most scenes are not defined by presence or absence of one object but by co-occurrence of multiple objects in the field of view. To achieve *across domain* transfer in these circumstances, we need the CNN to identify objects and an additional embedding to model their contextual co-occurrence. This embedding is not directly available from any ImageNet trained CNN and needs to be learned on the limited scene dataset that is available for transfer.

Additionally the efficacy of any transfer learning method also depends on the cardinality of the new data set available to learn, often called the *target* dataset. When the available data points per class is very few, the problem is referred to as that of *few-shot* learning. Few-shot learning is not easy even in the regime of deep neural networks. This is because one cannot finetune a large network, to a handful of examples, without overfitting. The only way out therefore, is to either collect more data or learn to generate synthetic examples that can be used to augment the target set.

I.C Contributions

In this thesis we consider two important cases of transfer learning. First is the problem of across-domain transfer learning, where an object recognition CNN is used to transfer knowledge to the domain of scenes. For this we design a bag-of-semantics (BoS) representation of scenes generated by the object recognition network. We design and test several embeddings of the BoS representation that summarize the contextual interactions of objects in scenes. The second problem we try to solve is that of one-shot or few-shot transfer. Specifically, we propose a method to augment a dataset of very few examples using synthetic samples generated by a network. This alleviates, to some extent, the issues of transfer in severe data constraints.

I.C.1 Object-to-Scene Semantic Transfer

Scenes are often described as collection of objects and stuff [1]. An object based CNN, therefore, can be used to identify the semantics present in a scene. In our work we propose to describe a scene image, on similar lines, with a bag-of-semantics (BoS) representation generated by a pre-trained object recognition

CNN. A BoS consists of an orderless collection of object probability multinomials generated by the CNN from local regions or patches of the scene. It is common practice in vision, to summarize such representations into a fixed length image representation typically using a non-linear embedding [91, 8, 64]. The most effective embedding in classical visual recognition literature is known as the Fisher vector (FV) [38, 64]. Although an FV is known to work quite well for many low-level features, with CNN generated probabilities, we show that it performs poorly. This is primarily due to the non-Euclidean nature of the space of probability vectors which makes it difficult to design an FV that generalizes well. We show that this problem can be alleviated by simply reversing the probability mapping implemented by the object classifiers and working with the natural parameter form of multinomials. For discriminative classifiers such as a CNN, a natural parameter transformation can be easily achieved by simply dropping the final softmax layer that produces probabilities. In the Euclidean space of natural parameters, we shown that a FV can be designed very easily using standard Fisher recipe. This representation, when used with a simple linear classifier, forms a conduit for transfer between object and scene domains. For the task of transfer based scene classification, it is shown to achieve very competitive results using relatively few scene images.

I.C.2 Semantic Covariance Modeling

While a classical FV embedding learned on a natural parameter form of object semantics produces a strong enough scene classifier, we show that it can be made even more accurate using an FV tailored for high dimensional data. The standard FV [64] derives from a Gaussian mixture model (GMM) that assumes a diagonal covariance per component. We argue that this approach is inefficient for the space of CNN features and that it would be better to use a GMM that is capable of modeling local covariances of the data manifold. Learning full covariance GMMs is impractical in large spaces due to the lack of enough data to estimate the parameters. Therefore, we propose a model that approximates the data distribution in several local low-dimensional linear sub-spaces. This model known as a mixture of factor analyzers (MFA) model learns a collection of local Gaussians with approximate covariance structure that cover the data distribution more efficiently compared to a variance-GMM. We derive a FV embedding for the MFA model and use it to encode our natural parameter BoS for transfer based scene classification. The ability to model covariance within an MFA, results in substantial improvements in the final scene classification. The transfer based MFA FV scene representation, is also shown to be better than a scene classifier trained directly from millions of labeled scene images. Upon combination with the scene CNN, the MFA-FS is shown to improve the performance further by a non-trivial margin.

I.C.3 Attribute Guided Data augmentation

Transfer learning becomes challenging especially when the amount of new data available to learn is very limited. Under extreme circumstances, this could be as little as 1-10 examples per class. We propose a solution to this so called *few-shot* transfer learning problem. Generally when a classifier has little data to train from, many works resort to cropping, flipping, rotating images to simulate the presence of adequate data. This is however, not the same as adding new information and the method seldom results in stable improvements. In our work we propose to generate non-trivial variations of available data points by attempting to modify their attributes (properties). We try to learn the trajectories of 3D pose and depth attributes of object images using a small auxiliary dataset that provides such information. In a one-shot or few-shot transfer scenario, then, for each available image, we generate its representation using an object CNN and regress it along the learned trajectories of poses and depths, thereby hallucinating changes in these properties and at the same time generating new synthetic features. The synthetic features correspond to the objects in the image changing their pose or depth by a specified amount. The transfer dataset with very few images is thus augmented with the additional samples generated by simulating attribute (pose/depth) variation. We show that the presence of additional examples improve the performance of one-shot or few-shot transfer based object as well as scene recognition. Since, the data augmentation is achieved using attributes as a supervisory signal, we refer to it as *attribute guided augmentation*. Alternatively, the data is generated by transferring a learned trajectory of variations in pose/depth, to the few-shot examples. Therefore, the proposed method can also be called *attribute trajectory transfer* based augmentation.

I.D Organization of the thesis

The organization of this thesis is as follows. In Chapter II we first review the existing bag-of-features (BoF) approach for scene classification. We then introduce the BoS image representation obtained using ImageNet trained CNNs. The design of basic FV embeddings for the ImageNet BoS is discussed in the rest of the chapter. In Chapter III we show that revisit the classical Fisher vector and show that it can be derived conveniently using the EM algorithm. We then introduce the Mixture of factor analyzers (MFA) model that can model covariances in high dimensional spaces unlike a variance GMM often preferred in FV literature. We derive the Fisher embedding using EM for MFA model and evaluate it for BoS based scene classification. In Chapter IV, we describe a system to generate synthetic data given very few examples of real data in a transfer scenario. This is shown to be helpful in one-shot and few-shot recognition scenarios. Final summary of the work and conclusions are presented in Chapter V.

Chapter II

Semantic Image Representations for Object to Scene Transfer

II.A Scene Classification

Natural scene classification is a challenging problem for computer vision, since most scenes are collections of entities (e.g. objects) organized in a highly variable layout. This high variability in appearance has made flexible visual representations quite popular for this problem. Many works have proposed to represent scene images as orderless collections, or "bags," of locally extracted visual features, such as SIFT or HoG [57, 15]. This is known as the bag-of-features (BoF) representation. For the purpose of classification, these features are pooled into an invariant image representation known as the Fisher vector (FV) [38, 64], which is then used for discriminant learning. Until very recently, bag-of-SIFT FV achieved state-of-the-art results for scene classification [73].

Recently, there has been much excitement about alternative image representations, learned with convolutional neural networks (CNNs) [51], which have demonstrated impressive results on large scale object recognition [45]. This has prompted many researchers to extend CNNs to problems such as action recognition [42], object localization [30], scene classification [31, 96] and domain adaptation [21]. Current multi-layer CNNs can be decomposed into a first stage of convolutional layers, a second fully-connected stage, and a final classification stage. The convolutional layers perform pixel wise transformations, followed by localized pooling, and can be thought of as extractors of visual features. Hence, the convolutional layer outputs are a BoF representation. The fully connected layers then map these features into a vector amenable to linear classification. This is the CNN analog of a Fisher vector mapping.

Beyond SIFT Fisher vectors and CNN layers, there exists a different class of image mappings known as *semantic representations*. These mappings require vectors of classifier outputs, or *semantic descriptors,* extracted from an image. Several authors have argued for the potential of such representations [88, 67, 80, 48, 49, 5, 52]. For example, semantic representations have been used to describe objects by their attributes [49], represent scenes as collections of objects [52] and capture contextual relations between classes [68]. For some visual tasks, such as hashing or large scale retrieval, a global semantic descriptor is usually preferred [84, 6]. Proposals for scene classification, on the other hand, tend to rely on a collection of locally extracted semantic image descriptors, which we refer to as bag of semantics (BoS) [80, 48, 52]. While a BoS based scene representation has outperformed low-dimensional BoF representations [48], it is usually less effective than the high dimensional BoF-FV. This is due to the fact that, 1) local or patch-based semantic features can be very noisy, and 2) it is harder to combine them into a global image representation, akin to the Fisher vector.

In this work, we argue that highly accurate classifiers, such as the ImageNET trained CNN of [45] eliminate the first problem. We obtain a BoS image representation using this network by extracting semantic descriptors (object class posterior probability vectors) from local image patches. We then consider the design of a *semantic Fisher vector*, which is an extension of the standard Fisher vector to this BoS. We show that this is difficult to implement directly on the space of probability vectors, because of its non-Euclidean nature. On the other hand, if semantic descriptors from an image are seen as parameters of a multinomial distribution and subsequently mapped into their natural parameter space, a robust semantic FV can be obtained simply using the standard Gaussian mixture based encoding of the transformed descriptors [64]. In case of a CNN, this natural parameter mapping is shown equivalent to the inverse of its soft-max function. It follows that the semantic FV can be implemented as a classic (Gaussian Mixture) FV of pre-softmax CNN outputs.

The semantic FV, represents a strong embedding of features that are fairly *abstract* in nature. Due to the invariance of this representation, which is a direct result of semantic abstraction, it is shown to outperform Fisher vectors of lower layer CNN features [31] as well as a classifier obtained by fine-tuning the CNN itself [30]. Finally, since object semantics are used to produce our image

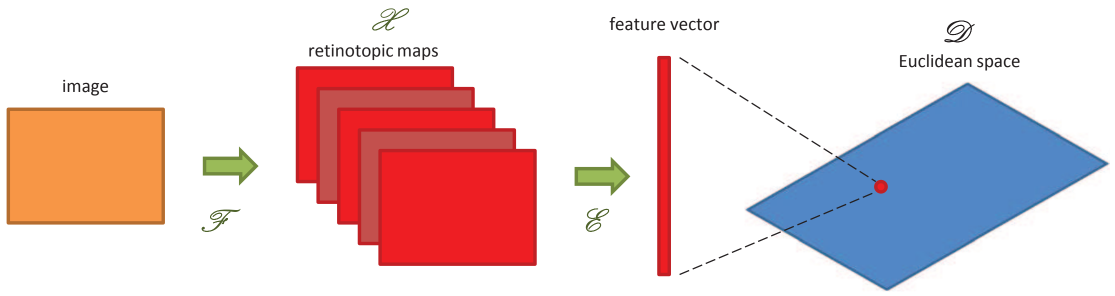


Figure II.1: The Bag of Features (BoF) embedding. A preliminary feature mapping $\mathcal{F}$, maps an image into a space $\mathcal{X}$ of retinotopic features. A non-linear embedding $\mathcal{E}$ is then used to map this intermediate representation into a feature vector on an Euclidean space $\mathcal{D}$.

representation, it is complementary to the features of the scene classification network (Places CNN) proposed in [96]. Experiments show that a simple combination of the two descriptors, produces a state-of-the-art scene classifier on MIT Indoor and MIT SUN benchmarks.

II.B Image representations

In this section, we briefly review BoF and BoS based image classification.

II.B.1 Bag of Features

Both the SIFT-FV classifier and the CNN are special cases of the general architecture in Figure Figure II.1, commonly known as the bag of features (BoF) classifier. For 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 maps preserve the spatial topology of the image. Common examples of mapping $\mathcal{F}$ include dense SIFT, HoG and the convolutional layers of a CNN. The BoF produced by $\mathcal{F}$ is subject to a highly nonlinear *embedding* $\mathcal{E}$ into a high dimensional feature space $\mathcal{D}$. This is a space with Euclidean structure, where a linear classifier $\mathcal{C}$ suffices for good performance.

It could be argued that this architecture is likely to always be needed for scene classification. The feature mapping $\mathcal{F}$ can be seen as a (potentially non-linear) local convolution of the input image with filters, such as edge detectors or object parts. This enables the classifier to be highly selective, e.g. distinguish pedestrians from cars. However, due to its retinotopic nature, the outputs of $\mathcal{F}$ are sensitive to variations in scene layout. The embedding $\mathcal{E}$ into the non-retinotopic space $\mathcal{D}$ is, therefore, necessary for *invariance* to such changes. Also, the space $\mathcal{D}$ must have a Euclidean structure to support classification with a linear decision boundary.

CNN based classifiers have recently achieved spectacular results on the ImageNET object recognition challenge [45, 74]. Their success has encouraged many researchers to use the features and embeddings learned by these networks for scene classification, replacing the traditional SIFT-FV based architecture [21, 75, 31, 55]. It appears undisputable that their retinotopic mapping $\mathcal{F}$, which is strongly non-linear (multiple iterations of pooling and rectification) and discriminant in nature (due to back-propagation) [93], has a degree of selectivity that cannot be matched by shallower mappings, such as SIFT. Less clear, however, is the advantage of using embeddings learned on ImageNET in place of the Fisher vectors for scene representation. As scene images exhibit a greater degree of intra class variation compared to object images, the ability to trade-off selectivity with invariance is critical for scene classification. While Fisher vectors derived using mixture based encoding are invariant by design, a CNN embedding learned from almost centered object images is unlikely to cope with the variability in scenes.

II.B.2 Bag of Semantics

Semantic representations are an alternative to the architecture of Figure Figure II.1. They simply map each image into a set of classifier outputs, using these as features for subsequent processing. The resulting feature space $\mathcal{S}$ is commonly known as the *semantic feature space.* Since scene semantics vary across image regions, scene classification requires a spatially localized semantic mapping. This is denoted as the *bag-of-semantics* (BoS) representation.

As illustrated in Figure Figure II.2, the BoS is akin to the BoF, but based on semantic descriptors. Its first step is the retinotopic maping $\mathcal{F}$. However, instead of the embedding $\mathcal{E}$, this is followed by another *retinotopic* mapping $\mathcal{N}$ into $\mathcal{S}$. At each location l, $\mathcal{N}$ maps the BoF descriptors extracted from a neighborhood of l into a *semantic descriptor*. The dimensions of this descriptor are probabilities of occurrence of visual classes (e.g. object classes, attributes, etc.). A BoS is an ensemble of retinotopic maps of these probabilities. An embedding $\mathcal{E}$ is used to finally map the BoS features into a Euclidean space $\mathcal{D}$.

While holistic semantic representations have been successful for applications like image retrieval or hashing, localized representations, such as the BoS, have proven less effective for scene classification, for a couple of reasons. First, the scene semantics are *hard to localize*. They vary from image patch to image patch and it has been difficult to build reliable scene patch classifiers. Hence, local semantics tend to be noisy [70, 52] and most works use a single global semantic descriptor per image [84, 4, 5]. This may be good for hashing, but it is not expressive enough for scene classification. Second, when semantics are extracted locally, the embedding $\mathcal{E}$ into an Euclidean space has been difficult to implement [48]. This is because semantic descriptors are probability vectors, and thus inhabit a very non-Euclidean space, the probability simplex, where commonly used descriptor statistics lose their effectiveness. In our results we show that even the sophisticated Fisher vector encoding [64], when directly implemented, has poor performance on this space.

We argue, that the recent availability of robust classifiers such as the CNN of [45], trained on large scale datasets, such as ImageNET [17], effectively solves the problem of noisy semantics. This is because an ImageNET CNN is, in fact, trained to classify objects which may occur in local regions or patches of a scene image. The problem of implementing an invariant embedding $\mathcal{E}$ in the semantic space, however, remains to be solved.

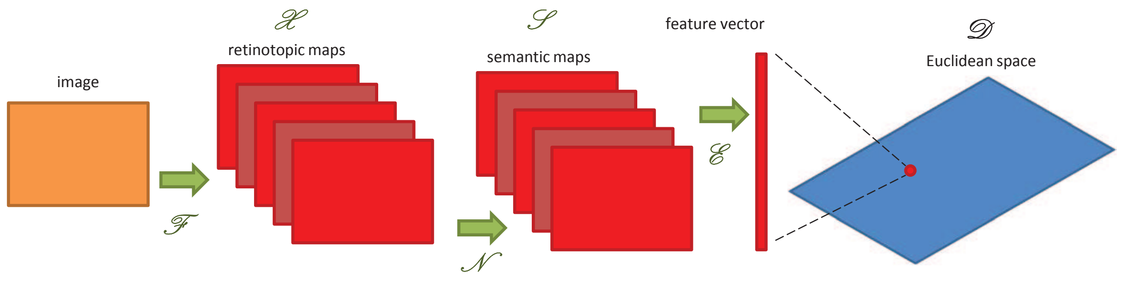


Figure II.2: The Bag of Semantics (BoS) embedding. The space $\mathcal{X}$ of retinotopic features is first mapped into a retinotopic semantic space $\mathcal{S}$, using a classifier of image patches. A non-linear embedding $\mathcal{E}$ is then used to map this representation into a feature vector on an Euclidean space $\mathcal{D}$.

II.C BoF embeddings

We first try to analyze, the suitability for scene classification, of the known BoF embeddings, namely the Fisher vector and the fully connected layers of ImageNET CNNs.

II.C.1 CNN embedding

For the CNN of [45], the mapping $\mathcal{F}$ consists of 5 convolutional layers. These produce an image BoF $\mathcal{I} = \{x_1, x_2, \ldots x_N\}$, where x_i's are referred to as the *conv5* descriptors. The descriptors are max pooled in their local neighborhood and transformed by the embedding $\mathcal{E}$. The embedding is implemented using two fully connected network stages, each performing a linear projection, and a non-linear *ReLu* transformation $\{W \times (.)\}_+$. The resulting outputs of layer 7, which we denote as *fc7*, are the features of space $\mathcal{D}$, in Figure Figure II.1.

II.C.2 FV embedding

Alternatively, a FV embedding can be implemented for the BoF of *conv5* descriptors. This consists of a preliminary projection into a principal component analysis (PCA) subspace,

$$x = Cz + \mu, \tag{II.1}$$

where C is a low-dimensional PCA basis and z are the coefficients of projection of the *conv 5* descriptors x on it. z's are assumed Gaussian mixture distributed.

$$z \sim \sum_k w_k N(\mu_k, \sigma_k). \tag{II.2}$$

A central component of the FV is the natural gradient with respect to parameters (mean, variance and weights) of this model [73]. For *conv5* features, we have found that the gradient with respect to the mean [64]

$$\mathcal{G}^{\mathcal{I}}_{\mu_k} = \frac{1}{N\sqrt{w_k}} \sum_{i=1}^{N} p(k|z_i) \left(\frac{z_i - \mu_k}{\sigma_k} \right) \tag{II.3}$$

suffices for good performance. Note that this gradient is an encoding and pooling operation over the z_i. It destroys the retinotopic topology of the BoF and guarantees invariance to variations of scene layout.

II.C.3 Comparison

We compared the CNN and FV embeddings, on two popular object recognition (Caltech 256 [33]) and scene classification (MIT Indoors [65]) datasets, with the results shown in the top half of Table Table II.1. For the CNN embedding, 7^{th} fully connected layer features were obtained with "Caffe" [41]. Following [21], this 4096 dimensional feature vector was extracted globally from each image. It was subsequently power normalized (square rooted), and L_2 normalized, for better performance [75]. The classifier trained with this representation is denoted "fc 7" in the table. For the FV embedding, the 256-dimensional *conv5* descriptors were PCA reduced to 200 dimensions and pooled with (II.3), using a 100-Gaussian mixture. This was followed by a square root and L_2 normalization, plus a second PCA to reduce dimensionality to 4096 and is denoted "conv5 + FV" in the table. Both representations were used with a linear SVM classifier.

The results of this experiment highlight the strengths and limitations of

Table II.1: Comparison of the ImageNET CNN and FV embeddings on scene and object classification tasks.

Method	MIT Indoor	Caltech 256
fc 7	59.5	68.26
conv5 + FV	61.43	56.37
fc7 + FV	65.1	60.97

the two embeddings. While *fc7* is vastly superior to the FV for object recognition (a gain of almost 12% on Caltech), it is clearly worse for scene classification (a loss of 2% on MIT Indoor). This suggests that, although invariant enough to represent images containing single objects, the CNN embedding cannot cope with the variability of the scene images. On the other hand, the mixture based encoding mechanism of the FV is quite effective on the scene dataset.

FV over *conv 5*, however, is an embedding of low-level CNN features. In principle, an equivalent embedding of BoS features should have better performance, since semantic descriptors have a higher level of abstraction than *conv5*, and thus exhibit greater invariance to changes in visual appearance. To some extent, the image representation proposed by Gong et. al. [31] shows the benefits of such invariance, albeit using an embedding of the intermediate 7^{th} layer activations, not the semantic descriptors at the network output. They represent a scene image as a collection of *fc7* activations extracted from local crops or patches. These are summarized using an approximate form of (II.3), known as VLAD [40]. The resulting embedding, denoted as "fc7 + FV" in Table Table II.1, is very effective for scene classification[1]. However, since the representation does not derive from semantic features, it is likely to be both less discriminant and less abstract than the truly semantic embedding of Figure Figure II.2. The implementation of an effective semantic embedding, on the other hand, is not trivial. We consider this problem in the remainder of this work.

[1]The results reported here are based on (II.3) and 128x128 image patches. They are slightly superior to those of VLAD, in our experiments

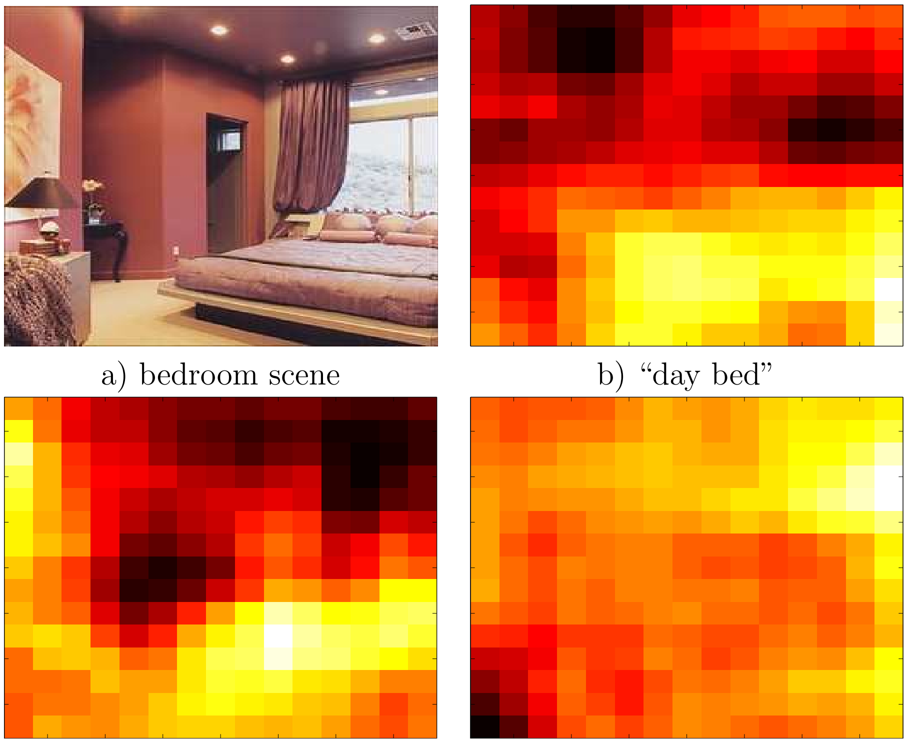


Figure II.3: Example of an imageNet based Bag of Semantics. a) shows the original image of a bedroom. The object recognition channels in ImageNet CNN related to b) “day bed” c) “comforter” and d) “window screen” show high affinity towards relevant local semantics of the scene

II.D Semantic FV embedding

We start with a brief review of a BoS image representation and then propose suitable embeddings for them.

II.D.1 The BoS

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 x_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

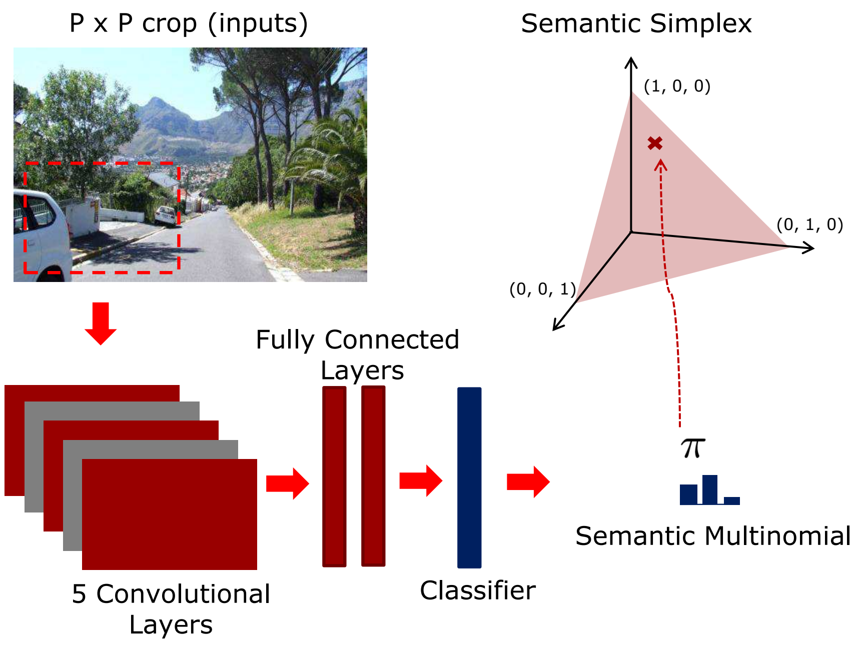


Figure II.4: Convolutional Neural Net based semantic image representation. Each image patch is mapped into an SMN π on the semantic space $\mathcal{S}$, by combination of a convolutional BoF mapping $\mathcal{F}$ and a secondary mapping $\mathcal{N}$ by the fully connected network stage. The resulting BoS is a retinotopic representation, i.e. one SMN per image patch.

of parameter π_i, the log-likelihood of image I can be expressed as,

$$\mathcal{L} = \log \prod_{i=1}^{n} \prod_{r=1}^{S} {\pi_{ir}}^{s_{ir}} = \sum_{i=1}^{N} \sum_{r=1}^{S} s_{ir} \log \pi_{ir}. \tag{II.4}$$

Since the precise semantic labels s_i for image regions are usually not known, it is common to rely instead on the expected log-likelihood

$$E[\mathcal{L}] = \sum_{i=1}^{n} \sum_{r=1}^{S} E[s_{ir}] \log \pi_{ir} \tag{II.5}$$

Using the fact that $\pi_{ir} = E[s_{ir}]$ or $P(r|x_i)$, it follows that the expected image log-likelihood is fully determined by the multinomial parameter vectors π_i. This is denoted the semantic multinomial (SMN) in [67]. They are usually computed by 1) applying a classifier, trained on the semantics of $\mathcal{V}$, to the image patches, and 2) using the resulting posterior class probabilities as SMNs π_i. The process is illustrated in Figure Figure II.4 for a CNN classifier. Each patch is thus mapped into the probability simplex, which is denoted the semantic space $\mathcal{S}$ in Figure Figure II.2. The image is finally represented by the SMN collection $I = \{\pi_1, \ldots, \pi_n\}$. This is the bag-of-semantics (BoS) representation.

In our implementation, we use the ImageNET classes as $\mathcal{V}$ and the object recognition CNN in [45] to estimate the SMNs π_i. Scoring patches of a scene individually, to generate these SMNs, is a simple but slow approach to semantic labeling. A faster alternative is to transform a CNN into a fully convolutional network and generate a BoS with one forward pass on the scene image. This requires changing the fully connected layers, if any, in the CNN 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. if the image is of size 512x512 pixels, the fully convolutional implementation of [45], generates SMNs from 128x128 pixel patches that are 32 pixels apart, approximately. The high quality of semantics generated by this classifier is apparent from fig. Figure II.3, where the recognizers related to “bed”, “window” and “quilt” are shown to exhibit high activity in areas where these objects appear in a bedroom scene.

II.D.2 Evaluation

We evaluate the performance of the GMM Fisher vector as a BoS embedding, for the task of scene classification. Experiments are performed on benchmark scene datasets namely MIT Indoors [65] and MIT SUN [90]. The MIT Indoor dataset consists of 100 images each from 67 indoor scene classes. The standard protocol for scene classification, on this dataset, is to use 80 images per class for training and the remaining 20 per class for testing. The MIT SUN dataset has about 100K images from 397 indoor and outdoor scene categories. The authors of this dataset provide randomly sampled image sets each with 50 images per class for training as well as test. Performance, on both datasets, is reported as average

Table II.2: Evaluation of different Fisher vector encoding techniques over imageNet BoS. Fisher vectors of fully connected layer features and handcrafted SIFT are included for reference.

Method	MIT Indoor	SUN
SIFT-FV	60.0	43.3
fc7-FV	65.1	48.3
SMN-FV	55.3	36.87
DMM-FV	58.8	40.86
$\nu^{(1)}$-FV	67.7	49.86
$\nu^{(2)}$-FV	**68.5**	**49.86**
$\nu^{(3)}$-FV	67.6	48.81
$\nu^{(4)}$-FV	58.95	40.6

per class classification accuracy.

To generate an image BoS, we use the object recognition CNN of [45], pre-trained on the ImageNet dataset. The network is applied to every scene image convolutionally, generating a 1000 dimensional SMN for every 128x128 pixel region, approximately. The 1000 dimensional probability vectors (π_i's) are reduced to 500 dimensions using PCA. A reference Gaussian mixture model θ^b, with 100 components, is trained using the PCA reduced SMNs x_i's from all training images. For each scene image, using its BoS, a Fisher vector is computed as shown in (II.3). The image FVs are power normalized and L_2 normalized, as per standard procedure [64], and used to train a linear SVM for scene classification.

The GMM-FV classifier trained on imageNet semantics is denoted as SMN-FV in Table Table II.2. Scene classification performance of the SMN-FV is found to be very poor on both MIT Indoor and SUN datasets. The classifier is in fact, significantly weaker compared to even a handcrafted SIFT-based FV. The accuracy of SMN-FV is about $5-6\%$ points lower than the SIFT-FV used in [73]. It is undisputed that the mappings of a CNN, which are strongly non-linear (multiple iterations of pooling and rectification) and discriminant (due to back-propagation), have a degree of selectivity that cannot be matched by shallower mappings, such as SIFT. The inferior performance of CNN based SMN-FV, therefore, is very surprising. Additionally, semantic (classifier) mappings of a CNN have a higher level of abstraction compared to mappings of lower network layers. CNN semantics therefore, exhibit greater *invariance* to visual appearance compared to activations from pre-classification fully connected layers. The SMN-FV, therefore, in principle, should have better performance compared to an FV embedding of fully connected (fc7) features. A comparison on scene classification, however, shows that a SMN-FV is substantially worse than a fc7-FV of similar complexity. Both the results indicate that despite the demonstrable superiority of ImageNet semantics, embedding them with a simple GMM-FV can adversely impact the final performance. In order to design an FV embedding suitable for ImageNet semantics, it is therefore, necessary to first understand the nature of the semantic space.

II.D.3 Limitations

While GMM FVs perform reasonably well with low-level feature spaces such as SIFT, HoG etc., their failure with ImageNet SMNs can be attributed to a very non-Euclidean nature of the space of probability vectors. In general, the difficulty of modeling on a data space $\mathcal{X}$ depends on its topology. Most machine learning assumes vector spaces with Euclidean structure, e.g. where the natural measure of distance between examples $x_i \in \mathcal{X}$ is a metric. This is not the case for the probability simplex, which has a non-metric Kullback-Leibler divergence as its natural distance measure, and makes model learning quite difficult.

To illustrate this issue we present two binary classification problems shown in Figures Figure II.5 a) and b). In one case, the two classes are Gaussian, and in the other they are Laplacian. The class-conditional distributions of both problems are of the form $P(x|y) \propto \exp\{-d(x, \mu_y)\}$ where $Y \in \{0, 1\}$ is the class label and

$$d(x, \mu) = ||x - \mu||_p \tag{II.6}$$

with $p = 1$ for Laplacian and $p = 2$ for Gaussian data. Figures Figure II.5 a) and

b) show the iso-contours of the probability distributions under the two scenarios. Note that both the classifiers have very different metrics.

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

$$\pi(x) = P(y = 1|x) = \sigma(d(x, \mu_0) - d(x, \mu_1)) \tag{II.7}$$

where $\sigma(v) = (1 + e^{-v})^{-1}$ is the sigmoid function. Due to the non-linearity of the sigmoid mapping, the projection $x \to (\pi(x), 1 - \pi(x))$ of the samples x_i into the semantic space destroys the Euclidean structure of their original spaces $\mathcal{X}$. This is illustrated in Figure Figure II.5 c), where we show the posterior surface and the projections $\pi(x_i)$ for samples x_i of the Guassian classes of Figure Figure II.5 a). On the semantic space, the shortest path between two points is not necessarily a line. The non-linearity of the sigmoid also makes the posterior surfaces of both classification problems very similar. The posterior surface of the Laplacian problem in Figure Figure II.5 b) is visually indistinguishable from Figure Figure II.5 c) and is omitted for brevity.

The example shows two very different classifiers transforming the data into highly non-Euclidean semantic spaces that are almost indistinguishable. This suggests that modeling directly in the space of probabilities with Gaussian Mixtures can be difficult in general. This is the most likely reason for the weakness of GMM-FVs in the semantic space of CNNs.

II.D.4 Natural parameter space

The non-Euclidean nature of a classifier's posterior surface makes the embedding $\mathcal{E}$ of Figure Figure II.2 very difficult to learn from SMNs. Note, for example, that the PCA step in (II.1) or the subsequent Gaussian encoding described in (II.3) make no sense for semantic space data, since the geodesics of the posterior surface are not lines. This problem can be avoided by noting that SMNs are the parameters of the multinomial, which is a member of the exponential family of distributions

$$P_S(s;\pi) = h(s)g(\pi)\exp\left(\eta^T(\pi)T(s)\right), \tag{II.8}$$

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

$$P_S(s;\nu) = h(s)g(\eta^{-1}(\nu))\exp\left(\nu^T T(s)\right). \tag{II.9}$$

This is called the *natural parameterization* of the distribution. Under this parametrization, the multinomial log-likelihood of an image BoS in (II.5) yields a natural parameter vector $\nu_i = \eta(E\{s_i\})$ for each patch x_i, instead of a probability vector. When the semantics are binary, the natural parameter is obtained by a logit transform $\nu = \log\frac{\pi}{1-\pi}$ of SMNs. This maps the high-nonlinear semantic space of Figure Figure II.5 c) into the linear space of Figure Figure II.5 d). Similarly, by mapping the multinomial distribution to its natural parameter space, it is possible to obtain a one-to-one transformation of the semantic space into a space with Euclidean structure. This makes the embedding $\mathcal{E}$ of Figure Figure II.2 substantially easier.

II.D.5 Natural Parameter FVs

Natural parameter transformation maps an image BoS $I = \{\pi_1, \dots \pi_n\}$ into the *natural parameter space BoS* (NP-BoS) $I = \{\nu_1, \dots \nu_n\}$. The main advantage of the natural parameter space is its Euclidean nature, which allows the design an embedding using the standard GMM FV machinery. For a multinomial distribution of parameter vector $\pi = (\pi_1, \dots, \pi_S)$ there are actually three possible natural parametrizations

$$\nu_k^{(1)} = \log\pi_k \tag{II.10}$$

$$\nu_k^{(2)} = \log\pi_k + C \tag{II.11}$$

$$\nu_k^{(3)} = \log\frac{\pi_k}{\pi_S} \tag{II.12}$$

where ν_k and π_k are the k^{th} entries of ν and π, respectively. The performance of these parametrizations is likely to depend on the implementation of the semantic classifier that generates the SMNs. For a discriminant classifier such as the CNN, $\nu^{(2)}$ will likely be the best parameterization. Note that, in this case, the vector of entries $\pi_k = \frac{1}{C}e^{\nu}$, is a probability vector if and only if $C = \sum_i e^{\nu_i}$. Hence, the mapping from ν to π is the softmax transformation commonly implemented at the CNN output. This implies that the CNN is learning how to discriminate the data in the natural parameter space of the multinomial distribution, which is a generalization of a natural binomial space shown in Figure Figure II.5 d). We test this assertion in our experiments by comparing the parametrizations of (II.10)-(II.12) for GMM-FV based scene classification.

II.D.6 Dirichlet Mixture FVs

An alternative to GMM based embedding of natural parameter features, is the use of a Dirichlet mixture models (DMM) in the space of SMNs. A DMM is the most popular model for multinomial probability vectors [59]. It was previously proposed by Rasiwasia et. al. [68] to model scene class-specific distributions of "theme" SMNs. The probability distribution function of a DMM is defined as,

$$P(\pi|\{\alpha_k, w_k\}_{k=1}^{K}) = \frac{1}{Z(\alpha_k)} e^{\sum_l (\alpha_{kl}-1)\log \pi_l}. \tag{II.13}$$

where α_k is known as the Dirichlet parameter of the k^{th} mixture component and w_k denotes the mixture weight. $Z(\alpha_k)$ is a normalizing constant of the distribution component and has the mathematical form $\frac{\gamma(\sum_l \alpha_{kl})}{\prod_l \gamma(\alpha_{kl})}$, where $\gamma(x) = \int_0^\infty x^{t-1}e^{-x}dx$ denotes a Gamma function. Note that the sufficient statistic of a Dirichlet distribution is $\log \pi$. A DMM, therefore, inherently operates in the space of the natural parameter $\nu^{(1)}$ shown in (II.10). The log-likelihood of an image BoS $I = \{\pi_1, \ldots \pi_n\}$

under the DMM is

$$\begin{aligned}\mathcal{L} &= \log P(\{\pi_i\}_{i=1}^{n}|\{\alpha_k, w_k\}_{k=1}^{K}) && \text{(II.14)}\\ &= \log \prod_{i=1}^{n}\sum_{k=1}^{K} w_k \frac{\gamma\left(\sum_l \alpha_{kl}\right)}{\prod_l \gamma(\alpha_{kl})} e^{\sum_l (\alpha_{kl}-1)\log \pi_{il}}. && \text{(II.15)}\end{aligned}$$

The Fisher scores of this log-likelihood are

$$\begin{aligned}\mathcal{G}_{\alpha_k}^{I} &= \frac{1}{n}\frac{\partial \mathcal{L}}{\partial \alpha_k}\\ &= \frac{1}{n}\sum_{i=1}^{N} p(k|\pi_i)\left(\psi(\sum_l \alpha_{kl}) - \psi(\alpha_k) + \log \pi_i\right) && \text{(II.16)}\end{aligned}$$

where $\psi(x) = \frac{\partial \gamma(x)}{\partial x}$. Using some common assumptions in the FV literature [64], we approximate the Fisher information $\mathcal{F}$ by the block diagonal matrix

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

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

II.D.7 Evaluation of NP Embeddings

We design a scene classification experiment to compare the performance of the DMM-FV from section II.D.6 with the Gaussian Mixture FVs trained over parameter mappings (II.10)-(II.12). The latter are denoted $\nu^{(i)}$-FVs based on the NP mapping $\nu^{(i)}$ used to learn them. The experimental protocol is the same as described in section II.D.2. Local SMNs are extracted using the ImageNet CNN of [45] from evenly spaced, 128x128 pixel patches of scene images from MIT Indoor and SUN datasets. To obtain the $\nu^{(i)}$-FVs, the image SMNs are transformed into the appropriate natural parameters and reduced to 500 dimensions using PCA.

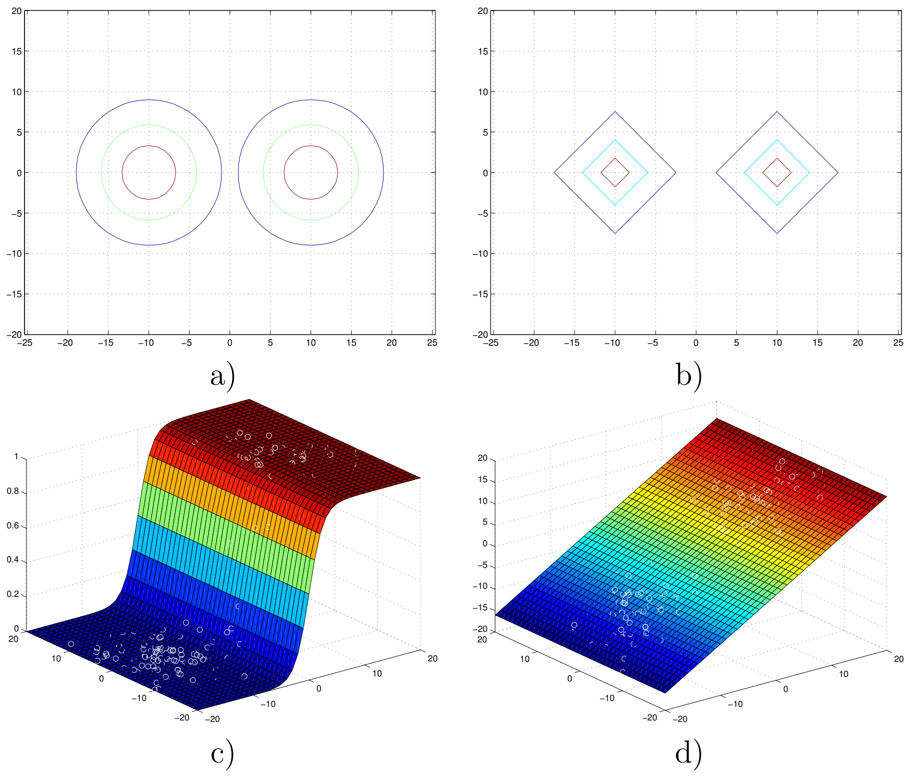


Figure II.5: Effects of probability mapping on Euclidean spaces. Top: Two classifiers in an Euclidean feature space $\mathcal{X}$, with metrics a) the L_2 or b) L_1 norms. Bottom: c) projection of a sample from a) into the semantic space $\mathcal{S}$ (only $P(y = 1|x)$ shown). The posterior surface destroys the Euclidean structure of $\mathcal{X}$ and is very similar for the Gaussian and Laplacian samples (Lapalacian omitted for brevity). d) natural parameter space mapping of c).

Reference or background GMMs $\theta^b = \{\mu_k^b, \sigma_k^b, w_k\}_{k=1}^K$ with $K = 100$ components, are learned using the PCA reduced natural parameters obtained from the training images. Natural parameter FVs are computed for these models using (II.3). For the DMM-FV, a reference Dirichlet model $\theta^b = \{\alpha_k^b, w_k^b\}_{k=1}^K$ with 100 components, is trained on the space of training SMNs π_i's. The FV embedding is computed using gradient scores in (II.16) and the Fisher scaling of (II.17). The DMM-FV and the $\nu^{(i)} - FV$s are Power normalized, L2 normalized and used to train linear SVMs for scene classification.

Results reported in table Table II.2, show that the natural parameter $\nu^{(i)}$-FVs improve significantly over GMM-FVs obtained directly from SMNs (denoted

SMN-FVs). The gain of about $13 - 14\%$ points in accuracy demonstrates the impact of learning an FV embedding in a Euclidean space, or, conversely, highlights the adverse effects of working directly in the probability space. Unlike the SMN-FV, the $\nu^{(i)}$-FVs are able to leverage the full discriminative power as well as the invariance properties of CNN semantics. Due to these qualities, they easily outperform the baseline SIFT-FVs [73] or FV's of lower CNN layer features (e.g. conv5-FV). The DMM-FV, on the other hand, performs very poorly. Despite acting in the same natural parameter space $\nu^{(1)} = \log \pi$, the DMM-FV is outperformed by the $\nu^{(1)}$-FV by a margin of about 9-10 % points on both datasets. This suggests that it is better to learn a simple Gaussian mixture in the space of Dirichlet sufficient statistics, instead of directly learning a Dirichlet mixture. Failure of the DMM-FV perhaps stems from the in-flexibility inherent to Dirichlet modeling. The discriminative power of a GMM and it's FV is known to increase proportionally with the mixture size. A DMM-FV, on the other hand, deteriorates with increasing mixture cardinality. The best performance, using a DMM-FV, is achieved when the distribution is unimodal, as seen in fig Figure II.6. To understand the reasons behind the disparity between the $\nu^{(i)}$-FVs and the DMM-FV, we perform an ablation study in the following section.

II.D.8 Ablation Analysis

Consider a Dirichlet mixture $\{w_k, \alpha_k\}_{k=1}^{K}$ learned from the SMNs and a Gaussian mixture $\{w_k, \mu_k, \Sigma_k\}_{k=1}^{K}$ learned using natural parameter points $\log \pi$. Both the distributions specify a distinct set of K codewords $\{c_k\}_{k=1}^{K}$ in the common space of $\nu^{(1)}$s. From the M-step of the EM algorithm, it is evident that the both sets of codewords are of the following form.

$$c_k = \frac{1}{n_k} \sum_{\{\pi_i \in \mathcal{D}\}} q_{ik} \log \pi_i \tag{II.18}$$

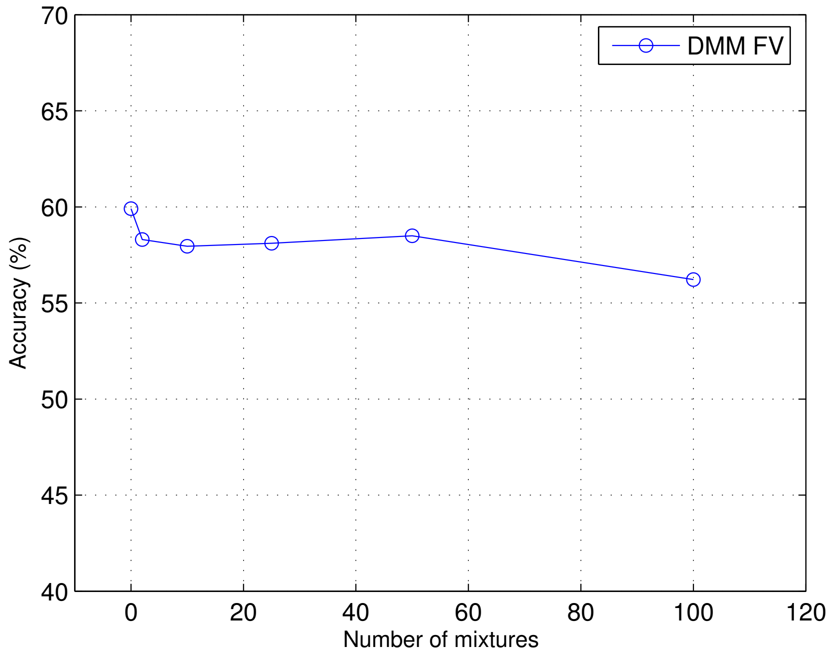


Figure II.6: The scene classification performance of a DMM-FV varying with the number of mixture components. The experiment is performed on MIT Indoor scenes.

The weights q_{ik} are the posterior assignment probabilities of points π_i from the training set $\mathcal{D}$ and n_k equals the sum of these weights $\sum_i q_{ik}$, estimated during the E-step. In case of a GMM, the centroids c_k equal the maximum likelihood (ML) estimates of its means μ_k, upon convergence of EM. For the Dirichlet mixture, these centroids provide an ML estimate of the function $f_k(\alpha) = \psi(\alpha_k) - \psi(\sum_k \alpha_k)$.

Since both the codebooks reside in the same natural parameter space, the Gaussian codewords can be used to construct a valid Dirichlet model and vice versa. The means $\{\mu_k\}_{k=1}^K$ of a Gaussian in the $\log \pi$ space, can be mapped to a set of Dirichlet parameters by solving the equation $\mu_k = \psi(\tilde{\alpha}_k) - \psi(\sum_k \tilde{\alpha}_k)$ for $\tilde{\alpha}_k$'s. Thus, a GMM $\{w_k, \mu_k, \Sigma_k\}_{k=1}^K$ in $\nu^{(1)}$ can be mapped to a DMM $\{w_k, \tilde{\alpha}_k\}_{k=1}^K$ by estimating the $\tilde{\alpha}_k$'s and copying the mixture weights w_k. Similarly, a mixture of Gaussians can be anchored at the centroids $\tilde{\mu}_k = f_k(\alpha)$ of a known DMM. The mixture weights w_k can be copied from the Dirichlet model. The covariances of the Gaussian components Σ_k can be initialized to the global covariance matrix Σ

estimated from the training data.

The ability to map a Dirichlet mixture into a Gaussian mixture and vice versa, helps us design experiments to probe the Fisher vector representations of the two models. In the first of these experiments, we evaluate the impact of a codebook on the performance of a GMM or a DMM FV. Using SMNs extracted from MIT Indoor scenes, we train a Dirichlet Mixture model $\{w_k, \alpha_k\}_{k=1}^{K}$ of $K = 100$ components as well as a Gaussian mixture model $\{w_k, \mu_k, \Sigma\}_{k=1}^{K}$ of the same size in the $\log \pi$ space. We denote these models as DMM_1 and GMM_1 respectively. A second Dirichlet model, DMM_2 is constructed using means of GMM_1 as $\{w_k, \tilde{\alpha}_k\}_{k=1}^{K}$. A Fisher vector is derived using both the learned and the constructed Dirichlet models and evaluated independently for classification. Similarly, using the centroids of the Dirichlet model DMM_1, we construct a second Gaussian mixture model GMM_2 with parameters $\{w_k, \tilde{\mu}_k, \Sigma\}_{k=1}^{K}$. Both the Gaussian models, GMM_1 and GMM_2 are used to obtain FVs in the $\nu^{(1)}$ space that are tested for classification. Results in table Table II.3 indicate that the performance of a Dirichlet model (DMM_1) trained from scratch is virtually similar to the performance of a Dirichlet model (DMM_2) derived from a GMM. The DMM_1-FV and DMM_2-FV embeddings achieve accuracies of within 0.5% of each other (58.8% v 58.4%). Same is the case with the Gaussian mixtures, GMM_1 and GMM_2. FV of the trained GMM_1 achieves an accuracy of 68.5% whereas an FV of the derived GMM_2 performs at 68.6%. These results suggest that the mixture model quality is perhaps not as important as the FV encoding it is used to generate. A GMM in the natural parameter space, seems to produce a better classifier than a Dirichlet mixture primarily, perhaps due to the way it encodes a descriptor into an FV. If the same Gaussian mixture is used to generate a DMM-FV encoding, its performance is seen dropping drastically (by about 10% points). On the other hand, if a learned Dirichlet mixture model is used to produce a GMM-FV instead of its natural DMM-FV encoding, its performance improves by the same margin. To further examine the impact of model quality on the power of an FV derived from it, we perform an ad-

Table II.3: Impact of codebook selection on DMM and GMM FVs in $\nu^{(1)}$ space. Table reports a comparison of FVs obtained using a learned DMM, denoted DMM_1, a DMM constructed from a learned GMM, denoted DMM_2 and one that is initialized from randomly sampled data points, denoted DMM_3 is shown above. A similar comparison is reported between GMM_1 trained on $\nu^{(1)}$, a GMM_2 constructed using DMM_1 and a GMM_3 initialized from randomly sampled datapoints in $\nu^{(1)}$. The experiments are performed on MIT Indoor.

Fisher Vector	Accuracy
DMM_1-FV	58.8
DMM_2-FV	58.4
DMM_3-FV	58.0
GMM_1-FV	68.5
GMM_2-FV	68.6
GMM_3-FV	61.3

ditional experiment with randomly initialized model centroids. A set of K natural parameter points $\{\log \pi_i\}_{i=1}^{K}$ are sampled randomly from the training set and used as our codewords $\{c_k\}_{k=1}^{K}$. Using this clearly sub-optimal codebook, we construct a Dirichlet mixture DMM_3 and a Gaussian mixture GMM_3 with uniform mixture weights. For the GMM, the component covariances are again set to Σ. Both the models are used to generate Fisher vectors denoted, DMM_3-FV and GMM_3-FV, and tested for scene classification. The accuracy of the DMM_3-FV from table Table II.3 is 58.0%, which is not very far from the Dirichlet-FVs obtained from learned codebooks (Gaussian or Dirichlet). The performance of the Gaussian mixture GMM_3-FV is, however, found to be much worse than GMM-FVs derived using properly learned mixtures. The poor performance of a GMM initialized with random means is no surprise. The centroids are not at all adapted to the data distribution and the model is not expected to informative at all. The total lack of impact this has on the performance of a DMM-FV is perhaps the more surprising result. Even a randomly initialized Dirichlet model produces a DMM-FV that is on par with the DMM-FVs of more systematically mixtures. This must mean that the Dirichlet mixture FV is inherently deficient as a descriptor encoding, compared to say a Gaussian mixture FV.

To understand what makes it so weak, we perform a component wise analysis of the DMM Fisher vector encoding. For this, we consider the learned Gaussian mixture GMM_1 in $\log \pi$ space and the Dirichlet mixture DMM_2 constructed from it. The FVs of these models produce a non-linear transformation of a data point π, that can be summarized as,

$$\mathcal{G}_k(\pi) = \mathcal{F}_k^{-\frac{1}{2}} q_k \{T(\pi) - c_k\} \tag{II.19}$$

Here, $\mathcal{F}_k$ denotes the Fisher information matrix (FIM), q_k the posterior assignment probability and $r_k(\pi) = T(\pi) - c_k$ the residual of the k^{th} component of the underlying mixture. By construction, the codewords of the Gaussian mixture GMM_1 are the same as those used by the Dirichlet model DMM_2 ($c_k = \mu_k = f_k(\tilde{\alpha})$). The descriptor residuals $\log \pi - c_k$ generated by the models, therefore, are exactly the same. The difference between the two encodings stems from their different mixture assignment functions q_k and Fisher scalings $\mathcal{F}_k$. The DMM_2 model uses a Fisher matrix of the form (II.17), which we denote as $H_k(\tilde{\alpha})$, whereas GMM_1's Fisher information equals Σ. The point assignment probability under the GMM_1 is $q(\log \pi; \mu, \Sigma, w) \propto w_k \exp\{\| \log \pi - \mu_k \|_\Sigma\}$, whereas under the DMM_2 model, it is $q(\pi; \tilde{\alpha}, w) \propto w_k \exp\{\tilde{\alpha}_k^T \log \pi\}$. Both the differences contribute to a cumulative gap of about 10% points in their classification performance. Starting from a GMM_1-FV, therefore, if we change its assignment function to $q(\pi; \tilde{\alpha}, w)$ or Fisher scaling to $H_k(\tilde{\alpha})$, we can verify the individual impact of the two components. Note that if we change both, the GMM_1-FV transforms into a DMM_2-FV. The results of this ablation are reported in table Table II.4. When the assignment function of the GMM_1-FV is changed to Dirichlet, its performance reduces by %. One the other hand, when the fisher scaling is changed from Σ to $H_k(\tilde{\alpha})$, a substantial loss of % is observed. We perform a similar experient, trying to replace the scaling and assignment of the DMM_1-FV with those of the constructed model GMM_2. The results again indicate only a moderate change in performance due to change of assignment

Table II.4: Ablation analysis of the DMM-FV and the GMM-FV embeddings on $\nu^{(1)}$ space.

Mixture	FV Encoding		
Model	Scaling	Assignment	Accuracy
DMM	$H(\alpha)$	$q(\pi;\alpha,w)$	58.8
		$q(\log\pi;\tilde{(\mu)},\Sigma,w)$	57.7
	Σ	$q(\pi;\alpha,w)$	67.1
		$q(\log\pi;\tilde{(\mu)},\Sigma,w)$	68.6
GMM	Σ	$q(\log\pi;\mu,\Sigma,w)$	68.5
		$q(\pi;\tilde{\alpha},w)$	68.7
	$H(\tilde{\alpha})$	$q(\log\pi;\mu,\Sigma,w)$	57.7
		$q(\pi;\tilde{\alpha},w)$	58.4

function. A big improvement is observed, however, when the Gaussian scaling Σ is used in place of the Dirichlet scaling of $H_k(\alpha)$. The evidence, therefore, support the conclusion that its Fisher information based scaling is the biggest drawback of the DMM-FV. The structure of this matrix as shown in (II.17) is restrictive, in that its off-diagonal elements are a scalar constant equal to $-\psi'(\sum_l \alpha_{kl})$. Therefore, it 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). The most important effect of any Fisher information based scaling is the decorrelation of an FV, which is known to improve its performance significantly [73]. The DMM Fisher matrix, however, seems to fail in producing this effect. Fisher information of a model represents the local metric on the tangent space $\mathcal{T}_{\theta^b}$ to the model manifold. Using the analysis, one can say that a Dirichlet model manifold is not conducive for FV based classification.

II.D.9 NP-BoS v.s. Comparable Transformations

For our BoS based scene classifier, a simple Gaussian Mixture FV constructed on one of the natural parameter spaces (II.10)-(II.12) seems to be the best candidate for a Euclidean embedding $\mathcal{E}$. As seen in table Table II.2, the performance of all three $\nu^{(i)}$ Fisher vectors is very close to each other, although

$\nu^{(2)}$ is slightly better than the other two. This is perhaps, because $\nu^{(2)}$ represents the natural parameter space that almost all known discriminant classifiers learn in, e.g. CNN (see explanation in II.D.5). We, therefore, prefer $\nu^{(2)}$ over others to learn our scene classifiers.

The proposed natural parameter transformations $\nu^{(i)}$, are the best known solution for the problems of semantic space classification. There have been proposals, in the past, along similar lines to transform semantic or non-semantic probability vectors for bag-of-features style classification. Kwitt et. al. [48], propose projecting the SMNs on the great circle using a square root embedding $\sqrt{\pi}$. This helps alleviate the difficulty of modeling on the complex semantic simplex, to some extent. The non-Euclidean nature of the simplex and the resulting non-linearity of its geodesics is also noted by the authors as a major source of difficulty for SMN based classification. The use of square root embedding was also proposed in [16] over low-level SIFT descriptors. Instead of L_2 normalizing the classical SIFT histogram, the authors propose to L_1 normalize it into a probability multinomial. The, L_1 normalized SIFT probabilities are transformed into "Root-SIFT" descriptors and encoded using a Gaussian mixture FV. It is unclear whether the distribution of low-level probabilities such as "Root-SIFT" are similar to semantic multinomials. The square-root trick, nevertheless, helps them achieve some moderate improvements over standard SIFT. We compare the square root embedding of SMNs with our NP transformations for FV based scene classification. The Root-BoS scene classifier is denoted $\nu^{(4)}$ in table Table II.2. Its performance on MIT Indoor and MIT SUN scene datasets , at 58.95% and 40.6% respectively, is quite close to that of the DMM-FV but much worse than any of the natural parameter embeddings. This indicates that a square root transformation is perhaps not enough to reverse the effects of non-linear probability transformations such as a sigmoid or a softmax. Another approach for transforming low-level probability descriptors was proposed by Kobayashi et. al. in [44]. Their use a log based transformation on L_1 normalized SIFT descriptors is also inspired from the Dirichlet sufficient statistics. In their work, 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 summarized using a Gaussian Mixture FV. The resulting image classifier was shown to improve over the Root-SIFT FV of [16]. Except for its L_2 normalization, the von-Mises embedding proposed in [44] is somewhat similar to one of our natural parameter transformations $\nu^{(3)} = \log p_i - \log p_N$. When used with our SMNs for FV based classification, we find that its performance is better than the square root embedding $\nu^{(4)}$ but worse than all three of our NP transformations (II.10)-(II.12). The most likely reason for this is the projection onto the great circle through an L_2 normalization, which may work for SIFT, but doesn't quite help CNN based semantics.

II.E Related work

The proposed semantic FV has relations with a number of works in the recent literature.

II.E.1 FVs of lower layer activations

The proposed representation, when computed with mapping $\nu^{(2)}$ of (II.11), as discussed above, acts directly on the outputs of the 8^{th} layer (fc8) of ImageNET CNN [45]. In that sense, it is similar to the Fisher vectors of [55, 31], which are computed using the activations from the fully connected 6^{th} (fc6) and 7^{th} layers (fc7). The most important difference between these and fc8 outputs, however, is that the latter are semantic features obtained as a result of a discriminant projection on fc7 and fc6. They are, therefore, likely to be more selective. Besides their explicit semantic nature also ensures a higher level of abstraction, as a result of which they can generalize better than lower CNN layer features. We compare with the two representations to validate these assertions.

II.E.2 Fine Tuning

Beyond its success on ImageNET classification, the CNN of [45] has been shown to be highly adaptable to other classification tasks. A popular adaptation strategy, known as “fine tuning” [30], involves performing additional iterations of back-propagation on the new datasets. This is, however, an heuristic and time consuming process, which needs to be monitored carefully in order to prevent the network from over-fitting. The proposed semantic Fisher vector can also be seen as an adaptation mechanism that fully leverages the original CNN, to extract features, augmenting it with a Fisher vector layer that enables its application to other tasks. This process is without heuristics and consumes much less time than “fine-tuning”. We compare the performance of the two in Section II.F.3.

II.E.3 The Places CNN

Recent efforts of improving scene classification have relied on a pre-trained imageNET CNN [21, 75, 31, 55]. mainly because of the superior quality of its feature responses [93]. Our work focusses on using object semantics generated by this network to obtain a high level representation for scene images. Zhou et. al. propose a more direct approach that does not rely on the ImageNET CNN at all. They simply learn a new CNN on a large scale database of scene images known as the “Places” dataset [96]. Although the basic architecture of their Places CNN is the same as that of the ImageNET CNN, the type of features learned are very different. While the convolutional units of ImageNET CNN respond to object-like occurrences, those in Places CNN are selective of landscapes with more spatial features. The embedding of the Places CNN, therefore, produces a holistic representation of scenes that is complementary to our semantic FV. We demonstrate the effect of combining the two representations in our classification experiments.

Table II.5: Impact of semantic feature extraction at different scales.

Dataset	Feat	full img	160	128	96	80	Best 3	Best 4
Indoor	fc8	48.5	**66.6**	**68.5**	**67.8**	**67.38**	**71.24**	**72.86**
	fc7	**59.5**	64.7	65.1	65.4	65.37	68.8	69.7
SUN	fc8	32.6	47.5	**49.61**	**50.03**	**49.39**	**53.24**	**54.4**
	fc7	**43.76**	48.08	48.3	48.46	47.32	51.8	53.0

II.F Evaluation

In this section we report on a number of experiments designed to evaluate the performance of the semantic FV.

II.F.1 Experimental setup

All experiments were conducted on the 67 class MIT Indoor Scenes [65] and the 397 class SUN Scenes [90] datasets. The CNN features were extracted with the Caffe library [41]. For FVs, the relevant CNN features (fc7 or fc8) were extracted from local $P \times P$ image patches on a uniform grid. For simplicity, the preliminary experiments were performed with $P = 128$. A final round of experiments used multiple scale features, with $P \in \{96, 80, 128, 160\}$. For all GMM-FVs the local features were first reduced to 500 dimensions, using a PCA, and then pooled using (II.3) and a 100 component mixture. The DMM-FV of Section II.D.6 was learned with a 50 component mixture on the 1,000 dimensional SMN space. As is common in the literature,all Fisher vectors were power normalized and L_2 normalized. This resulted in DMM and GMM FVs of size of 50000, dimensions of which were further reduced to 4096, by PCA. In some experiments, we also evaluate classifiers based on fc7 and SMN features extracted globally, as in [21]. The global fc7 features were square-rooted and L_2 normalized, whereas the global SMNs were simply square rooted. Scene classifiers trained on all image representations were implemented with a linear SVM.

II.F.2 The role of invariance

To test the hypothesis that the semantic FV is both more discriminant and invariant than FVs extracted from lower network layers, we compared its performance with that of the fc7 FV of [31]. In this experiment, the CNN features were extracted at multiple scales (globally as well as from patches of size 80, 96, 128 and 160 pixels). Table Table II.5 shows the results of the semantic FV (denoted fc8) and the fc7 FV. Several remarks are worth making, in light of previous reports on similar experiments [30, 21, 31]. First, when compared to the approach of extracting CNN features globally [21], the localized representations have far better performance. Second, while fc7 features extracted globally are known to perform poorly [31], the use of global 8^{th} layer features leads to even worse performance. This could suggest the conclusion that layer 8 somehow extracts "worse features" for scene classification. The remaining columns, however, clearly indicate otherwise. When extracted locally, semantic descriptors are highly effective, achieving a gain of up to 3 points with respect to the fc7 features. The gap in performance between the localized and global semantic descriptors is explained by the localized nature of scene semantics, which vary from patch to patch. A global semantic descriptors is just not expressive enough to capture this diversity. Third, recent arguments for the use of intermediate CNN features should be revised. On the contrary, the results of the table support the conclusion that these features are both less discriminant and invariant than semantic descriptors. When combined with a proper encoding, such as the semantic FV, the latter achieve the best scene classification results.

Finally, to ensure that the gains of the semantic FV are not just due to the use of the transformation of (II.11), we applied the transformation to the fc7 features as well. Rather than a gain, this resulted in a substantial performance decrease (58% compared to the 65.1% of the fc7-FV on MIT Indoors at patch size 128). This was expected, since the natural parameter space arguments do not apply in this case.

II.F.3 Comparison to the state of the art

Concatenating Fisher vectors of fc7 features computed at multiple patch scales was shown to produce substantial gains in [31]. We implemented this strategy for both the fc7-FV and the semantic FV, with the results shown in Table Table II.5. Combining the fc7-FVs at three patch scales resulted in classification accuracies of 68.8% on MIT Indoor and 51.8% on SUN. While this is a non-trivial improvement over any of the single-scale classifiers, the concatenation of semantic-FVs at 3 scales produced even better results (accuracies of 71.24% on MIT Indoor and 53.24% on SUN). Similar gains were observed when using 4 patch scales, as reported in the table.

A comparison of our multiscale semantic FV with other leading representations derived from the ImageNET CNNs is shown in table Table II.6. As expected, the pioneering DeCaf [21] representation is vastly inferior to all other methods since it describes complex scene images with a globally extracted descriptor using an object CNN [45]. Among techniques that rely on local feature extraction are the proposals of Liu et. al. [55] and Razavian et. al. [75]. The scene representation in [55] is a sparse code derived from the 6^{th} layer activations (fc6) of the CNN. Razavian et. al. [75], use features from the penultimate layer of the OverFeat network [74] extracted from coarser spatial scales.Since, the features used in both [55] and [75] lack the invariance of semantics, their classifiers are easily outperformed by our semantic FV classifier. We also compare with the technique referred to as fine-tuning [30] which adapts the imageNET CNNs directly to the task of scene classification. The process requires a few tens of thousands of back-propagation iterations on the scene dataset of interest and lasts about 5-10 hours on a single GPU. The resulting classifier, however, is significantly worse than our semantic FV classifier.

An alternative to using pre-trained object classification CNNs [45, 74] for scenes is to learn a CNN directly on a large scale scene dataset. This was recently performed by Zhou et. al. using a 2 million image Places dataset [96]. Table Table

Table II.6: Comparison with the state-of-the-art methods using ImageNET trained features. *-Indicates our implementation.

Method	MIT Indoor	MIT SUN
fc8-FV (Our)	**72.86**	**54.4 $\pm$ 0.3**
fc7-FV [31]*	69.7	53.0 $\pm$ 0.4
fc7-VLAD [31]	68.88	51.98
ImgNET finetune	63.9	48.04 $\pm$ 0.19
OverFeat + SVM [75]	69	-
fc6 + SC [55]	68.2	-
MidLevel [20]	66.87	-
DeCaF [21]*	59.5	43.76

Table II.7: Comparison with a CNN trained on Scenes [96]

Method	MIT Indoor	MIT SUN
ImgNET fc8-FV (Our)	72.86	54.4 $\pm$ 0.3
Places fc7 [96]	68.24	54.34 $\pm$ 0.14
Combined	**79.0**	**61.72 $\pm$ 0.13**

II.7 indicates a comparison between a scene representation obtained with the Places CNN and our ImageNET based semantic FV. The results of semantic FV are slightly better that theirs on the Indoor scenes dataset, whereas, on SUN, both the descriptors perform comparably. More importantly, a simple concatenation of the two produces a gain of almost 7% in accuracy on both datasets, indicating that the embeddings are, in fact, complimentary. These results are, to the best of our knowledge, state-of-the-art on scene classification.

II.G Conclusions

In this chapter we discussed the benefits of modeling scene images as bags of object semantics from an ImageNET CNN instead of its lower layer activations. To leverage the superior quality of semantic descriptors, we propose an effective approach to summarize them with a Fisher vector, which is non-trivial. The semantic FV provides a better classification architecture than an FV of low-level features or a even fine-tuned classifier. When combined with features from a scene classification CNN, our semantic FV produces state-of-the-art results.

II.H Acknowledgements

The text of Chapter II, is based on material as it appears in: M. Dixit, S. Chen, D. Gao, N. Rasiwasia and N. Vasconcelos, ”Scene classification with semantic Fisher vectors”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, 2015 and M. Dixit, Y. Li and N. Vasconcelos, ”Bag-of-Semantics representation for object-to-scene transfer”, [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material.

Chapter III

Improving Transfer with semantic covariance model

III.A Introduction

In this chapter we propose improvements to the ImageNet based semantic Fisher vectors introduced in Chapter II for scene classification. Like in the classifical FV literature [63], the semantic FVs were derived using Gaussian mixtures learned with a diagonal covariance per component. We refer to such a model as a variance-GMM. While variance only modeling in GMMs has been sufficient for lower dimensional SIFT descriptors [73], it may not necessarily be the case for much higher dimensional CNN features. The ImageNet semantics or natural parameter features, for instance, most likely reside in a non-linear manifold of the ambient high dimensional semantic space. A relatively inflexible variance-GMM will require large number of components to cover such a data distribution. On the other hand, a GMM that is capable of modeling full covariance, locally, will certainly be more efficient. However, full covariance modeling in large spaces is not very easy. The number of parameters to be estimated increase quadratically with the dimensionality of the space. The amount of data available is always insufficient for learning such large mixture models. One could, however, resort to approximate covariance modeling using models such as a mixture of factor analyzers (MFA) [27, 87]. The MFA model covers a non-linear data manifold with local linear approximations. The data is assumed to be generated by a simpler Gaussian distribution in localized low dimensional latent spaces and then projected linearly into the high dimensional observation space. The MFA effectively provides a low rank approximation for the full covariance of a Gaussian and can, therefore, be learned with reasonably less data available in transfer learning scenarios. It also generates higher dimensional covariance statistics, which we show to be better than the gradient scores with respect to variances, for semantic classification. The MFA generated Fisher scores, in fact, outperform the semantic fisher vectors of Chapter II further improving the efficacy of CNN based cross-domain (object to scene) knowledge transfer.

We first begin with a re-interpretation of Fisher vectors that were introduced in Chapter II and show that they are in fact obtained in the popular Expectation maximization (EM) algorithm. This is result is used to obtain Fisher scores for the MFA model. The MFA Fisher information is derived using an old result related to higher order moments of zero mean Gaussian random variables. The MFA based Fisher vectors are then evaluated for transfer based scene classification on the MIT Indoor and SUN benchmarks.

III.B Fisher scores

In computer vision, an image is frequently interpreted as a set of descriptors $\mathcal{D} = \{x_1, \ldots, x_n\}$ sampled i.i.d. from some generative model $p(x;\theta)$. Since most classifiers require fixed-length inputs, it is common to map the bag of descriptors $\mathcal{I}$ into a fixed-length vector. A popular mapping consists of computing the gradient (with respect to θ) of the log-likelihood $\nabla_\theta L(\theta) = \frac{\partial}{\partial\theta} \log p(\mathcal{D};\theta)$ for a model θ^b. This is known as the *Fisher score* of θ. This gradient vector is often normalized by the square root of the Fisher information matrix $\mathcal{F}$ of $p(x;\theta)$, according to $\mathcal{F}^{-\frac{1}{2}}\nabla_\theta L(\theta)$. This is referred to as a Fisher vector (FV) [38] representation of $\mathcal{D}$.

Strength of a Fisher vector depends on the expressiveness of the generative model $p(x;\theta)$. An FV derived for a sophisticated probabilistic model can capture higher order trends of the feature distribution within images. E.g., a Fisher vector of a large enough mixture of Gaussians (GMM) is known to be a strong descriptor of image context [63, 73]. For complex distributions like GMMs and hidden Markov models, directly deriving Fisher scores is not always easy. We show, however, that scores can be trivially obtained using a single step of the expectation maximization (EM) algorithm commonly used to learn such models.

III.B.1 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 does not depend on the hidden variable, this can be written in an alternate form, which is widely used in the EM literature,

$$\begin{aligned}
& \log p(\mathcal{D};\theta) = \log p(\mathcal{D}, z;\theta) - \log p(z|\mathcal{D};\theta) \\
= & \int q(z)\log p(\mathcal{D}, z;\theta)dz - \int q(z)\log p(z|\mathcal{D};\theta)dz \\
= & \int q(z)\log p(\mathcal{D}, z;\theta)dz - \int q(z)\log q(z)dz \\
& + \int q(z)\log \frac{q(z)}{p(z|\mathcal{D};\theta)}dz \\
= & Q(q;\theta) + H(q) + KL(q||p;\theta) \qquad \text{(III.1)}
\end{aligned}$$

where $Q(q;\theta)$ is the "Q" function, $q(z)$ 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 $q(z)$. Hence,

$$\frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta) = \frac{\partial}{\partial\theta}Q(q;\theta) + \frac{\partial}{\partial\theta}KL(q||p;\theta) \qquad \text{(III.2)}$$

where

$$\frac{\partial}{\partial\theta}KL(q||p;\theta) = -\int \frac{q(z)}{p(z|\mathcal{D};\theta)}\frac{\partial}{\partial\theta}p(z|\mathcal{D};\theta)dz. \qquad \text{(III.3)}$$

In each iteration of the EM algorithm the q distribution is chosen as $q(z) = p(z|\mathcal{D};\theta^b)$, where θ^b is a reference parameter vector (the parameter estimates from the previous EM iteration) and

$$\begin{aligned}
Q(q;\theta) &= \int p(z|\mathcal{D};\theta^b)\log p(\mathcal{D}, z;\theta)dz \qquad \text{(III.4)} \\
&= E_{z|\mathcal{D};\theta^b}[\log p(\mathcal{D}, z;\theta)]. \qquad \text{(III.5)}
\end{aligned}$$

It follows that

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

and

$$\frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta)\bigg|_{\theta^b} = \frac{\partial}{\partial\theta}Q(p(z|\mathcal{D};\theta^b);\theta)\bigg|_{\theta^b} \qquad \text{(III.6)}$$

Thus, 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.

An alternative approach to the same result, requires analysis of the KL divergence between $q = p(z|\mathcal{D};\theta^b)$ and $p(z|\mathcal{D};\theta)$. By rearranging terms in (III.1) it can be written as,

$$KL(\theta^b;\theta) = \log p(\mathcal{D};\theta) - Q(\theta^b;\theta) - H_q(\theta^b)$$

For models that allow exact inference, this expression is mathematically tractable and continuously differentiable with respect to θ. The divergence, therefore, smoothly reduces to 0 as θ approaches θ^b from any direction. The slope of a tangent to $KL(\theta^b;\theta)$ (it's derivative) at $\theta = \theta^b$, therefore, equals 0. It follows that

$$\begin{aligned}\frac{\partial}{\partial\theta}KL(\theta^b;\theta)\bigg|_{\theta=\theta^b} &= 0 \\ \frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta)\bigg|_{\theta^b} - \frac{\partial}{\partial\theta}Q(p;\theta)\bigg|_{\theta^b} &= 0 \\ \frac{\partial}{\partial\theta}\log p(\mathcal{D};\theta)\bigg|_{\theta^b} = \frac{\partial}{\partial\theta}Q(p;\theta)\bigg|_{\theta^b} &\end{aligned}$$

The computation of the Fisher score thus simplifies into the two steps of EM. First, the E step computes the Q function $Q(p(z|x;\theta^b);\theta)$ at the reference θ^b.

Second, the M-step evaluates the gradient of the Q function with respect to θ at $\theta = \theta^b$. This interpretation of the Fisher score is particularly helpful when efficient implementations of the EM algorithm are available, e.g. the recursive Baum-Welch computations commonly used to learn hidden Markov models [66].

III.B.2 Bag of features

Fisher scores are usually combined with the bag-of-features representation, where an image is described as an orderless collection of localized descriptors $\mathcal{D} = \{x_1, x_2, \ldots x_n\}$. These were traditionally SIFT descriptors, but have more recently been replaced with responses of object recognition CNNs [31, 12]. In this work we use the semantic features proposed in Chapter II, which are obtained by transforming softmax probability vectors p_i, obtained for image patches, into their natural parameter form.

III.B.3 Gaussian Mixture Fisher Vectors

A GMM is a model with a discrete hidden variable that determines the mixture component which explains the observed data. The generative process is as follows. A mixture component z_i is first sampled from a multinomial distribution $p(z = k) = w_k$. An observation x_i is then sampled from the Gaussian component $p(x|z = k) \sim \mathcal{G}(x, \mu_k, \sigma_k)$ of mean μ_k and variance σ_k. Both the hidden and observed variables are sampled independently, and the Q function simplifies to

$$\begin{aligned} Q(p(z|\mathcal{D}; \theta^b); \theta) &= \sum\nolimits_i E_{z_i|x_i;\theta^b}\left[\sum\nolimits_k I(z_i, k) \log p(x_i, k; \theta)\right] \\ &= \sum\nolimits_{i,k} h_{ik} \log p(x_i|z_i = k; \theta) w_k \end{aligned} \tag{III.7}$$

where $I(.)$ is the indicator function and h_{ik} is the posterior probability $p(k|x_i; \theta^b)$. The probability vectors h_i are the only quantities computed in the E-step.

In the Fisher vector literature [64, 73], the GMM is assumed to have diagonal covariances. This is denoted as the variance-GMM. Substituting the expressions of $p(x_i|z_i = k; \theta)$ and differentiating the Q function with respect to parameters $\theta = \{\mu_k, \sigma_k\}$ leads to the two components of the Fisher score

$$\mathcal{G}_{\mu_k^d}(\mathcal{I}) = \frac{\partial}{\partial \mu_k^d} L(\theta) = \sum\nolimits_i p(k|x_i) \left(\frac{x_i^d - \mu_k^d}{(\sigma_k^d)^2} \right) \tag{III.8}$$

$$\mathcal{G}_{\sigma_k^d}(\mathcal{I}) = \frac{\partial}{\partial \sigma_k^d} L(\theta) = \sum\nolimits_i p(k|x_i) \left[\frac{(x_i^d - \mu_k^d)^2}{(\sigma_k^d)^3} - \frac{1}{\sigma_k^d} \right]. \tag{III.9}$$

These quantities are evaluated using a reference model $\theta^b = \{\mu_k^b, \sigma_k^b\}$ learned (with EM) from all training data. To compute the Fisher vectors, scores in (III.8) and (III.9) are often scaled by an approximate Fisher information matrix, as detailed in [73]. When used with SIFT descriptors, these mean and variance scores usually capture complimentary discriminative information, useful for image classification [64]. Yet, FVs computed from CNN features only use the mean gradients similar to (III.8), ignoring second-order statistics [31]. In the experimental section, we show that the variance statistics of CNN features perform poorly compared to the mean gradients. This is perhaps due to the inability of the variance-GMM to accurately model data in high dimensions. We test this hypothesis by considering a model better suited for this task.

III.B.4 Fisher Scores for the Mixture of Factor Analyzers

A factor analyzer (FA) is a type of a Gaussian distribution that models high dimensional observations $x \in \mathbb{R}^D$ in terms of latent variables or "factors" $z \in \mathbb{R}^R$ defined on a low-dimensional subspace $R << D$ [27]. The process can be written as $x = \Lambda z + \epsilon$, where Λ is known as the factor loading matrix and ϵ models the additive noise in dimensions of x. Factors z are assumed distributed as $\mathcal{G}(z, 0, I)$ and the noise is assumed to be $\mathcal{G}(\epsilon, 0, \psi)$, where ψ is a diagonal matrix. It can be shown that x has full covariance $S = \Lambda\Lambda^T + \psi$, making the FA better suited for high dimensional modeling than a Gaussian of diagonal covariance.

A mixture of factor analyzers (MFA) is an extension of the FA that allows a piece-wise linear approximation of a non-linear data manifold. Unlike the GMM, it has two hidden variables: a discrete variable s, $p(s = k) = w_k$, which determines the mixture assignments and a continuous latent variable $z \in \mathbb{R}^R$, $p(z|s = k) = \mathcal{G}(z, 0, I)$, which is a low dimensional projection of the observation variable $x \in \mathbb{R}^D$, $p(x|z, s = k) = \mathcal{G}(x, \Lambda_k z + \mu_k, \psi)$. Hence, the k^{th} MFA component is a FA of mean μ_k and subspace defined by Λ_k. Overall, the MFA components approximate the distribution of the observations x by a set of sub-spaces in observation space. The Q function is

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

where $h_{ik} = p(s_i = k|x_i; \theta^b)$. After some simplifications, the E step reduces to computing

$$h_{ik} = p(k|x_i; \theta^b) \propto w_k^b \mathcal{N}(x_i, \mu_k^b, S_k^b) \qquad \text{(III.10)}$$

$$E_{z_i|x_i;\theta^b}[z_i] = \beta_k^b (x_i - \mu_k^b) \qquad \text{(III.11)}$$

$$\begin{aligned} E_{z_i|x_i;\theta^b}[z_i z_i^T] &= \beta_k^b (x_i - \mu_k^b)(x_i - \mu_k^b)^T \beta_k^{b^T} \\ &\quad - \left(\beta_k^b \Lambda_k^b - I\right) \end{aligned} \qquad \text{(III.12)}$$

with $S_k^b = \Lambda_k^b \Lambda_k^{b^T} + \psi^b$ and $\beta_k^b = \Lambda_k^{b^T} \left(S_k^b\right)^{-1}$. The M-step then evaluates the Fisher score of $\theta = \{\mu_k^b, \Lambda_k^b\}$. With some algebraic manipulations, this can be shown to have components

$$\mathcal{G}_{\mu_k}(\mathcal{I}) = \sum_i p(k|x_i; \theta^b) \{S_k^b\}^{-1} \left(x_i - \mu_k^b\right) \qquad \text{(III.13)}$$

$$\begin{aligned} \mathcal{G}_{\Lambda_k}(\mathcal{I}) = \sum_i p(k|x_i; \theta^b) \Big[&\{S_k^b\}^{-1} (x_i - \mu_k^b)(x_i - \mu_k^b)^T \beta_k^{b^T} \\ &- \{S_k^b\}^{-1} \Lambda_k^b \Big] \end{aligned} \qquad \text{(III.14)}$$

For a detailed discussion of the Q function, the reader is referred to the EM derivation in [27]. Note that the scores with respect to the means are functionally similar to the first order residuals in (III.8). 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 the representations (III.13) and (III.14) as MFA Fisher scores (MFA-FS). The mean scores (III.13) are scaled with the Fisher information matrix S_k^b which equals the component covariance. For the factor loading scores in (III.14), the derivation of Fisher information is presented in the following section.

III.B.5 Fisher Information

For a mixture distribution, the Fisher information is often approximated as a block-diagonal matrix that scales the Fisher scores of the k^{th} mixture component with inverse square-root of the sub-matrix

$$\mathcal{F}_k = w_k Cov\left(\mathcal{G}_k(x)\right) \tag{III.15}$$

Here w_k is the weight of the k^{th} mixture and $\mathcal{G}_k(x)$ is the data term of its Fisher score. Therefore, to approximate the Fisher information with respect to Λ_ks, we need to compute the covariance of the data term in (III.14). This term is a $D \times R$ matrix, every $(i,j)^{th}$ entry of which is a product of two Gaussian random variables,

$$\begin{aligned} \mathcal{G}_k^{(i,j)}(x) &= f_i g_j \\ &= \lfloor \{S_k^b\}^{-1}(x-\mu_k^b) \rfloor_i \lfloor \beta_k^b (x-\mu_k^b) \rfloor_j \end{aligned} \tag{III.16}$$

Here $\lfloor W \rfloor_i$ denotes the i^{th} element of the vector W. The covariance matrix of the vectorized Fisher score then contains the following terms,

$$Cov(\mathcal{G}(x))_{(i,j),(k,l)} = E\left[f_i g_j f_k g_l\right] - E[f_i g_j]E[f_k g_l]$$

This can be simplified using the expectation property of Gaussian random variables [1] as

$$Cov(\mathcal{G}(x))_{(i,j),(k,l)} = E\left[f_i g_l\right] E\left[f_k g_j\right] + E\left[f_i f_k\right] E\left[g_j g_l\right] \tag{III.17}$$

where the individual expectation terms can be computed as

$$E\left[f_i g_l\right] E\left[f_k g_j\right] = \lfloor (S_k^b)^{-1} \Lambda_k^b \rfloor_{(i,l)} \lfloor (S_k^b)^{-1} \Lambda_k^b \rfloor_{(k,j)} \tag{III.18}$$

$$E\left[f_i f_k\right] E\left[g_j g_l\right] = \lfloor (S_k^b)^{-1} \rfloor_{(i,k)} \lfloor \beta_k^b \Lambda_k^b \rfloor_{(j,l)} \tag{III.19}$$

The Fisher scaling for the k^{th} MFA component can be obtained by combining (III.15), (III.17) and (III.18). While this derivation of the MFA information has a tutorial value, in practice it did not result in any gains over simply using the scores of Λ as our image representation. Therefore, we avoid scaling the scores in (III.14) for our image classification experiments, although they may be useful in some other scenarios.

III.B.6 Evaluating MFA embeddings

In this section we present an extensive evaluation of the MFA based Fisher embedding for BoS classification.

Impact of Covariance Modeling

We begin with an experiment to compare the modeling power of MFAs to variance-GMMs. This was based on ImageNet SMNs extracted using the CNN in [45] on a 128x128 patch scale. An MFA of $K = 50$ components, and a latent space dimension of $R = 10$ was learned on the natural parameters $\nu^{(2)}$. To make the learning manageable, the descriptors were first reduced using PCA to

[1]For Gaussian random variables $\{x_1, x_2, x_3, x_4\}$, the property $E[x_1 x_2 x_3 x_4] = E[x_1 x_2]E[x_3 x_4] + E[x_1 x_3]E[x_2 x_4] + E[x_1 x_4]E[x_2 x_3]$ holds true.

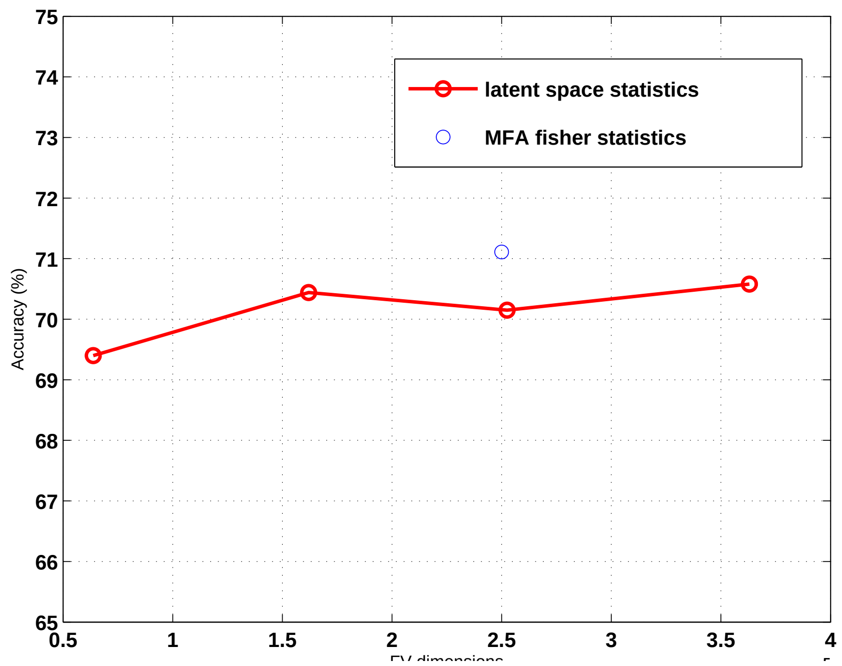


Figure III.1: Performance of Latent space statistics of (III.12) for different latent space dimensions. The accuracy of MFA-FS (III.14) for K=50, R=10 included for reference.

500 dimensions. Classification is performed on both MIT Indoor and SUN scene datasets. Table Table III.2 presents the classification accuracy of a GMM-FV that only considers the mean - GMM-FV(μ) - or variance - GMM-FV(σ) - parameters and a MFA-FS that only considers the mean - MFA-FS(μ) - or covariance - MFA-FS(Λ) - parameters. The most interesting observation is the complete failure of the GMM-FV (σ), which under-performs the GMM-FV(μ) by more than 10%. The difference between the two components of the GMM-FV is not as startling for lower dimensional SIFT features [64]. However, for CNN features, the discriminative power of variance statistics is exceptionally low. This explains why previous FV representations for CNNs [31] only consider gradients with respect to the means. A second observation of importance is that the improved modeling of covariances by the MFA eliminates this problem. In fact, MFA-FS(Λ) is significantly better than both GMM-FVs. It could be argued that a fair comparison requires an increase in the GMM modeling capacity. Fig. Table III.3 tests

this hypothesis by comparing GMM-FVs(σ) and MFA-FS (Λ) for various numbers of GMM components ($K \in \{50, \ldots, 500\}$) and MFA hidden sub-spaces dimensions ($R \in \{1, \ldots, 10\}$). For comparable vector dimensions, the covariance based scores always significantly outperforms the variance statistics on both datasets. A final observation is that, due to covariance modeling in MFAs, the MFA-FS(μ) performs better the GMM-FV(μ). The first order residuals pooled to obtain the MFA-FS(μ) (III.13) are scaled by covariance matrices instead of variances. This local de-correlation provides a non-trivial improvement for the MFA-FS(μ) over the GMM-FV(μ)($\sim$ 1.5% points). Covariance modeling was previously used in [82] to obtain FVs w.r.t. Gaussian means and local subspace variances (eigen-values of covariance). Their subspace variance FV, derived with our MFAs, performs much better than the variance GMM-FV (σ), due to a better underlying model (60.7% v 53.86% on Indoor). It is, however, still inferior to the MFA-FS(Λ) which captures full covariance within local subspaces.

While a combination of the MFA-FS(μ) and MFA-FS(Λ) produces a small improvement ($\sim$ 1%), we restrict to using the latter in the remainder of this work.

Local v Global Codebook

We experiment here with the structure of the MFA model and evaluate its impact on the MFA-FS embedding. We learn multiple MFA models such that the size of the second order MFA-FS (Λ) obtained from each of them remains fixed at 250K dimensions. Specifically, the mixture cardinality is varied from $K = 250$ to $K = 10$, and at each step a reduction in K is traded off for an increase in the latent space dimensions R (R varies from 2 to 50). The MFA-FS obtained from each of these models is evaluated for MIT Indoor scene classification.

The results in fig Figure III.2 indicate a steady increase in accuracy of the representation as K decreases from 250 to 50 and latent space dimensions increase simultaneously from 2 to 10. This confirms our earlier assumption about the MFA model, that if adequate parameters are allowed to approximate the local covariance

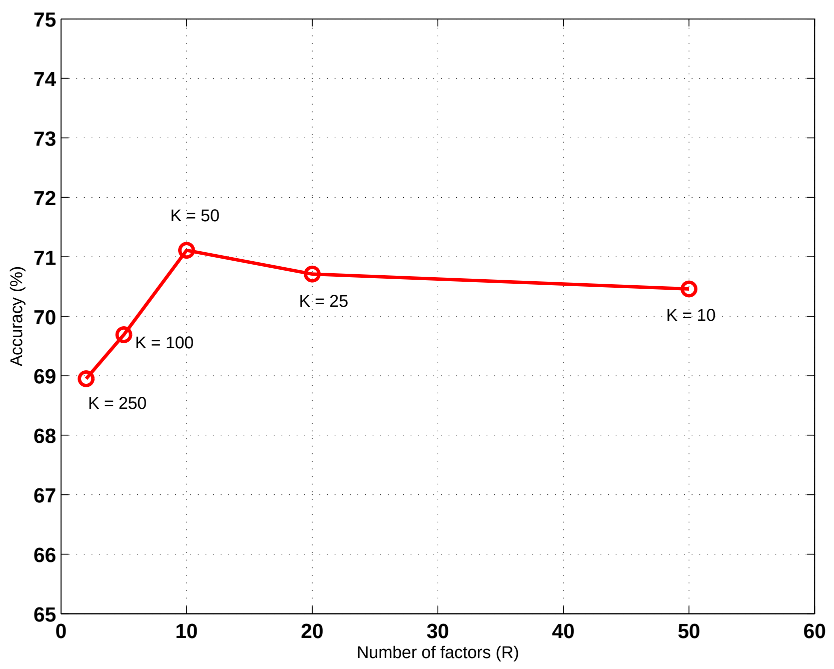


Figure III.2: Comparison of MFA-FS obtained with different mixture models. The size of the MFA-FS $(K \times R)$ is kept constant. From left to right, the latent space dimensions R are incraesed while decreasing the number of mixture components K. Optimal result is obtained when the model combines adequate representation power in the latent space as well as the ability to model spatially (K = 50, R = 10).

of data, the model can be economical in the number of mixture components. On the other hand, if the latent space dimensions R are increased further in exchange of decreasing number of mixture components, the performance of the FS decreases steadily. Trying to improving the covariance approximation by increasing R at the cost of the model's ability to cover the manifold impacts the final performance. Best results are achieved when the MFA combines a sufficient number of Gaussians that implement a reasonable linear approximation of the local manifold (e.g. $K = 50, R = 10$).

Latent Space Statistics

The scores MFA-FS(Λ) incorporate second order sufficient statistics of dimensions $K \times D \times R$, where K is the number of components, D is the size of

Table III.1: Comparison of MFA-FS with semantic "gist" embeddings learned using ImageNet BoS and the Places dataset.

Descriptor	MIT Indoor	SUN
MFA FS (Λ)	**71.11**	**53.38**
BoS-fc1	64.84	47.47
BoS-fc2	69.36	50.9
BoS-fc3	70.6	53.12

the observation space and R, the size of the latent space. Alternatively we could use the latent space second order statistics in (III.12) as an image representation. These are directly obtained using the E-step and their dimensionality is approximately $K \times \frac{R^2}{2}$. For a moderately small latent space $R << D$, the representation in (III.12) can be used as a low-dimensional alternative to MFA-FS(Λ).

We evaluate (III.12), which we refer to as Latent Fisher statistics (LFS) on MIT Indoor scene classification. As features, we again use the 128x128 patch ImageNet BoS generated by [45]. The MFA models used to obtain LFS, have a fixed mixture size of $K = 50$. The latent space size is increased from $R = 50$ to $R = 120$ to increase the dimensionality of the LFS. Results in fig Figure III.1 shows that for a moderate size of $60K$ dimensions, the LFS achieves about 69.4% accuracy which is only 1.6% worse than the accuracy of the $250K$ dimensional MFA-FS(Λ). Increasing the latent space dimensions R to $\{80, 100, 120\}$ increases the accuracy above 70%. The LFS, however, never outperforms an MFA-FS of a comparable size, perhaps, because a larger model is neede to generate the former. Under the constrains of transfer learning, the data may not be enough to properly learn such large MFAs.

MFA-FS v BoS "gist"

The MFA-FS embedding derives from a bag-of-descriptors model of an image, which was used very often in classical vision literature [86, 14, 50, 64]. Due to the i.i.d. assumptions in these models, the embeddings derived from them are very flexible and do not exhibit template like rigidity. On the other hand, neural networks have a tendency to learn “gist”-style image representations that are more holistic non-linear templates of a visual concept. To compare our MFA-FS representation with NN based “gist embeddings”, we propose to learn an MLP with one or more fully connected layers and ReLu non-linearities on top of the ImageNet BoS generated by the pre-trained network in [45]. The BoS map is of a fixed size (10 x 10 x 1000), with the ImageNet CNN generating a 1000 dimensional $\nu^{(2)}$ descriptor for roughly every 128x128 pixel region. A fully connected layer on top of this map maps it into 4096 dimensions and is followed by a ReLu non-linearlity. Successive fully connected layers with input v output channels (4096 x 4096) and ReLu stages can be added to make the embedding deeper before the final classification layer. In the fc layers we employ “drop-out” with a probability of 0.2. The network embeddings are denoted BoS-fc1, BoS-fc2 or BoS-fc3 based on the number of fc layers used on top of the semantic map. We propose to train the “gist” embeddings directly with a scene classification loss and the ImageNet.

The most immediate problem faced during training is the insufficiency of data in a transfer scenario. The total number of parameters in the embedding is almost equal to the ImageNet CNN itself. A dataset of the size of MIT Indoor, for example, proves to be highly inadequate to train such a heavy embedding. The resulting accuracy of scene classification drops as low as 33%. The only way to learn this embedding, therefore, is using a dataset that is as large as ImageNet itself. We use the large scale scene classification dataset named Places, introduces recently by Zhou et. al. [96]. It consists of 2.4 M training images distributed across 200 scene categories. The training of the “gist” embeddings on this data takes many days on a GPU, as opposed to the MFA-FS training which takes a couple hours. The complexity of these embeddings also far exceeds that of the MFA-FS which is based on a PCA and a moderately sized mixture. We use the learned BoS-fc embeddings from Places dataset, and use them as image representations for transfer based scene classification on MIT Indoor and SUN. As seen in table Table III.1, despite the complexity of these BoS-fc embeddings and the relatively large amount of data required to learn them (2.4M Places images v 5K Indoor or 20K SUN images), the performance is not much better than the MFA-FS. The BoS-fc representation can of course be made deeper by adding more layers, and the performance may become better than the MFA-FS. It is not possible, however, without millions of additional images and days of training, none of which are needed for our transfer based MFA-FS image representation. In a later section, we show that our MFA-FS classifier is also competitive with very deep CNNs are trained end-to-end on the Places dataset for scene classification.

III.C Related work

Although most methods using pre-trained ImageNet CNN features default to the GMM FV embedding [31, 12], some recent works have tried to explore different pooling strategies [55, 54, 26]. We briefly introduce these methods here and compare to their results in our experimental section.

III.C.1 Gradients based on Sparse Coding

The work of Liu et. al. [55] is motivated by the assumption that a Gaussian mixture of finite size may not be effective in modeling high dimensional features such as those generated by a CNN. Their proposal is a generative Gaussian distribution $p(x|u) \sim \mathcal{N}(Bu, \Sigma)$ with a random mean that resides in the span of basis vectors B. The mean vector is indexed by a code u sampled from a zero mean Laplace distribution. The approximate marginal log likelihood of features reduces to a standard sparse coding objective [91] which is differentiated with respect to dictionary B to obtain their final image representation. We compare their scene classification performance with our proposed MFA gradient representation.

III.C.2 Bilinear Pooling

Lin et. al. [54] have recently proposed a simple bilinear pooling of CNN features for the task of image classification. Their classifier makes predictions in the space of an image representation obtained by aggregating outer products $x_i x_i^T$ of CNN features. The representation was shown to be fine-tunable to the task along with the CNN layers. Gao et. al. [26] recently proposed a learnable low-dimensional projection for these bilinear descriptors to make them more compact. Since a bilinear pooled representation captures correlation between CNN features, it is, in some sense, similar to our MFA gradients. In our experiments, therefore, we compare with the scene classification results presented in these works.

III.D Scene Image Classification

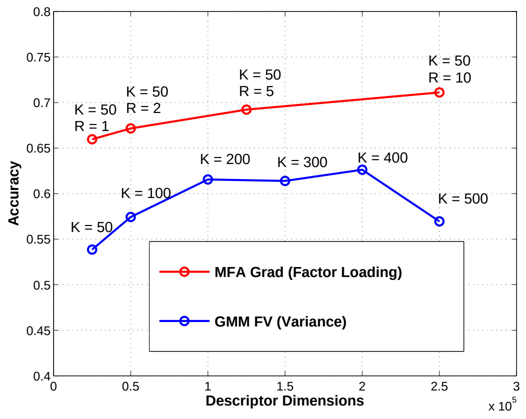
We finally compare the proposed MFA-FS representation in (III.14) with the state-of-the-art methods in scene classification. We build these classifiers using image BoS obtained from three different ImageNet CNNs, namely, the 8 layer network of [45] (denoted as AlexNet) and the deeper 16 and 19 layer networks of [76] (denoted VGG-16 and VGG-19, respectively). These CNNs assign 1000 dimensional object recognition probabilities to $P \times P$ patches (sampled on a grid of fixed spacing) of the scene images, with $P \in \{128, 160, 96\}$. For the MFA-FS representation, image SMNs are used in their natural parameter form $\nu^{(2)}$ and PCA-reduced to 500 dimensions as in Chapter II. The scene image BoS is mapped into the embedding, using (III.13), (III.14) and a background MFA of size $K = 50, R = 10$. Note that a separate MFA-FS is generated for every imageNet CNN (AlexNet, VGG16 and VGG19) for every fixed patch scale P of SMN extraction. As usual in the FV literature, the MFA-FS vectors are power normalized, $L2$ normalized, and classified with a cross-validated linear SVM.

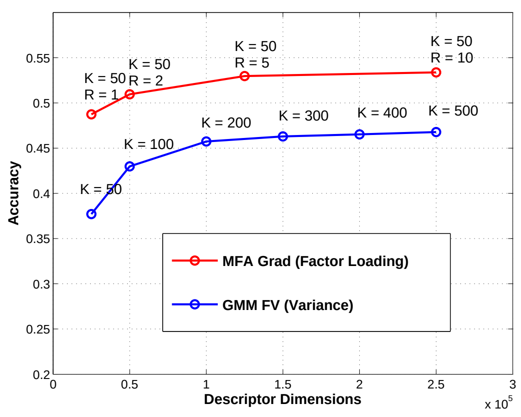
The proposed classifiers are compared to scene CNNs, trained on the large scale Places dataset. In this case, the features from the penultimate CNN layer are

Table III.2: Classification accuracy ($K = 50, R = 10$).

Descriptor	MIT Indoor	SUN
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**

Table III.3: Classification accuracy vs. descriptor size for MFA-FS(Λ) and a comparable GMM-FV(σ). The MFA model uses $K = 50$ components and R factor dimensions. Left: MIT Indoor. Right: SUN.





used as a holistic scene representation and classified with a linear SVM, as in [96]. We use the places CNNs trained with the AlexNet and the VGG-16 architectures provided by the authors. We also compare our performance with ImageNet based transfer learning methods that combine FV-like embeddings with different CNN features [32, 55, 56, 12, 89, 26, 53] for scene classification.

III.D.1 Multi-scale learning and Deep CNNs

First we employ a well known trick to improve the performance of our MFA based classifiers. Recent works have demonstrated gains due to combining deep CNN features extracted at multiple-scales. Table Table III.4 presents the classification accuracies of the MFA-FS (Λ) based on AlexNet, and 16 and 19 layer VGG features extracted from 96x96, 128x128 and 160x160 pixel image patches, as well as their concatenation (3 scales). These results confirm the benefits of multi-

Table III.4: MFA-FS classification accuracy as a function of patch scale.

	MIT Indoor	SUN
AlexNet		
160x160	69.83	52.36
128x128	71.11	53.38
96x96	70.51	53.54
3 scales	**73.58**	**55.95**
VGG-16		
160x160	77.26	59.77
128x128	77.28	60.99
96x96	79.57	61.71
3 scales	**80.1**	**63.31**
VGG-19		
160x160	77.21	-
128x128	79.39	-
96x96	79.9	-
3 scales	**81.43**	-

scale feature combination, which achieves the best performance for all CNNs and datasets.

III.D.2 Comparison with ImageNet based Classifiers

We next compared the MFA-FS to state of the art scene classifiers also based on transfer from ImageNet CNN features [55, 12, 26]. Since all these methods only report results for MIT Indoor, we limited the comparison to this dataset, with the results of Table Table III.6. The GMM FV in [12] is computed using convolutional features from AlexNet or VGG-16 extracted in a large multi-scale setting. Liu et. al. proposed a gradient representation based on sparse codes. Their initial results were reported on a single patch scale of 128x128 using AlexNet features [55]. More recently, they have proposed an improved H-Sparse representation, combined multiple patch scales and used VGG features in [56]. The recently proposed bilinear (BN) descriptor pooling of [54] is similar to the MFA-FS in the sense that it captures global second order descriptor statistics. The simplicity of these descriptors

Table III.5: Comparison with scene classification state-of-the-art. *-combination of patch scales (128, 96, 160).

Method	MIT Indoor	SUN
MFA-FS + Places (VGG)	**87.23**	**71.06**
MFA-FS + Places (AlexNet)	79.86	63.16
MFA-FS (VGG)	81.43	63.31
MFA-FS (AlexNet)	73.58	55.95
Full BN (VGG) [26]	77.55	-
Compact BN (VGG) [26]	76.17	-
H-Sparse (VGG) [56]	79.5	-
Sparse Coding (VGG) [56]	77.6	-
Sparse Coding (AlexNet) [55]	68.2	
MetaClass (AlexNet) + Places [89]	78.9	58.11
VLAD (AlexNet) [31]	68.88	51.98
FV+FC (VGG) [12]	81.0	-
FV+FC (AlexNet) [12]	71.6	-
Mid Level [53]	70.46	-
DAG-CNN (VGG) [92]	77.5	56.2

enables the fine-tuning of the CNN layers to the scene classification task. However, their results, reproduced in [26] for VGG-16 features, are clearly inferior to those of the MFA-FS without fine-tuning. Gao et. al. [26] propose a way to compress these bilinear statistics with trainable transformations. For a compact image representation of size $8K$, their accuracy is inferior to a representation of $5K$ dimensions obtained by combining the MFA-FS with a simple PCA.

These experiments show that the MFA-FS is a state of the art procedure for task transfer from object recognition (on ImageNet) to scene classification (e.g. on MIT Indoor or SUN). Its closest competitor is the classifier of [12], which combines CNN features in a massive multiscale setting (10 image sizes). While MFA-FS outperforms [12] with only 3 image scales, its performance improves even further with addition of more scales (82% with VGG, 4 patch sizes).

Table III.6: Comparison to task transfer methods using ImageNet CNNs on MIT Indoor.

Method	1 scale	mscale
AlexNet		
MFA-FS	**71.11**	**73.58**
FV+FC [12]	-	71.6
Sparse Coding [55]	68.2	-
VGG		
MFA-FS	**79.9**	**81.43**
Sparse Coding [56]	-	77.6
H-Sparse [56]	-	79.5
BN [26]	77.55	-
FV+FC [12]	-	81.0
VGG + dim. reduction		
MFA-FS + PCA (5k)	**79.3**	-
BN (8k) [26]	76.17	-

III.D.3 Task transfer performance

The next question is how object-to-scene transfer compares to the much more intensive process, pursued by [96], of collecting a large scale labeled Places scene dataset and training a deep CNN from it. The dataset, consists of 2.4M images, from which the authors train both AlexNet and VGG Net CNNs were trained for scene classification. The fully connected features from the networks are used as scene representations and classified with linear SVMs on Indoor scenes and SUN. The Places CNN features are a direct alternatives to the MFA-FS. While the use of the former is an example of dataset transfer (features trained on scenes to classify scenes) the use of the latter is an example of task transfer (features trained on objects to classify scenes).

A comparison between the two transfer approaches is shown in table Table III.7. Somewhat surprisingly, task transfer with the MFA-FS *outperformed* dataset transfer with the Places CNN, on both MIT Indoors and SUN and for both the AlexNet and VGG architectures. This supports the hypothesis that the variability of configurations of most scenes makes scene classification much harder than ob-

Table III.7: Comparison with the Places trained Scene CNNs.

Method	SUN	Indoor
AlexNet		
MFA-FS	55.95	73.58
Places	54.3	68.24
Combined	63.16	79.86
VGG		
MFA-FS	63.31	81.43
Places	61.32	79.47
Combined	**71.06**	**87.23**
AlexNet + VGG		
Places (VGG + Alex)	65.91	81.29
MFA-FS(Alex) + Places(VGG)	68.8	85.6
MFA-FS(VGG) + Places(Alex)	67.34	82.82

ject recognition, to the point where CNN architectures that have close-to or above human performance for object recognition are much less effective for scenes. It is, instead, preferable to pool object detections across the scene image, using a pooling mechanism such as the MFA-FS. We also show that the object-based MFA-FS and the scene-based representation from Places CNN are *complementary* in nature. When we train a classifier on a concatenation of the two descriptors, the performance improves by about $6-8\%$ which is significant margin. The improvements are observed while using both the AlexNet and the VGG CNN architectures on both MIT Indoor and SUN datasets. To the best of our knowledge, no method using these or deeper CNNs has reported better results than the combined MFA-FS and Places VGG features of Table Table III.7.

It could be argued that this improvement is just an effect of the often observed benefits of fusing different classifiers. Many works even resort to "bagging" of multiple CNNs to achieve performance improvements [76]. To test this hypothesis we also implemented a classifier that combines two Places CNNs with the AlexNet and VGG architectures. This is shown as Places (VGG+AlexNet) in the last section of Table Table III.7. While improving on the performance of both MFA-FS and Places, its performance is not as good as that of the combination of

Table III.8: Classification accuracy ($K = 50, R = 10$).

Descriptor	MIT Indoor	SUN
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**

the object-based and scene-based representations (MFA-FS + Places). As shown in the remainder of the last section of the table, any combination of an object CNN with MFA-FS based transfer and a scene CNN outperforms this classifier.

Finally, table Table III.5 compares results to the best recent scene classification methods in the literature. This comparison shows that MFA-FS + Places combination is a state-of-the-art classifier with substantial gains over all other proposals. The results of 71.06% on SUN and 87.23% on Indoor scenes substantially outperform the previous best results reported on both scene datasets.

III.E Conclusion

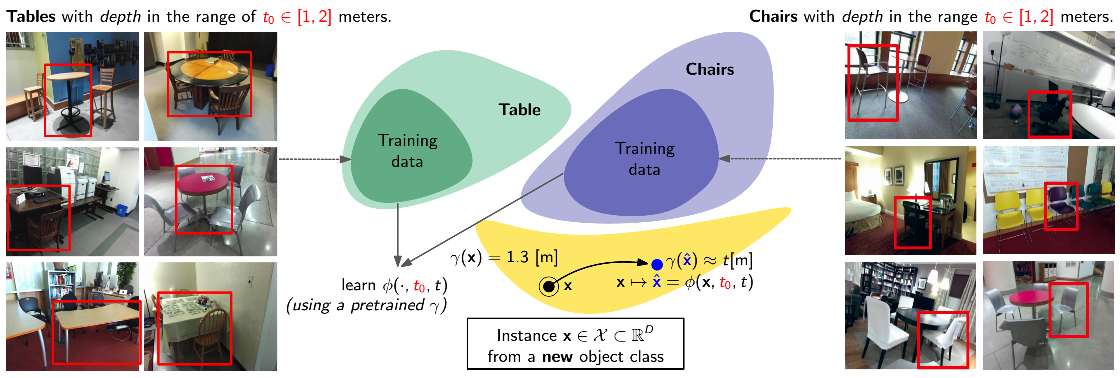
Object based scene representation was addressed with traditional GMM FV in Chapter II, which we have shown to be very ineffective for the high-dimensional CNN features. The reason is the limited flexibility of the GMM in modeling feature covariances. We have addressed this problem by adopting a better model, the MFA, which approximates the non-linear data manifold by a set of local subspaces. We then introduced the Fisher score with respect to this model, denoted as the MFA-FS. Through extensive experiments, we have shown that the MFA-FS has state of the art performance for object-to-scene transfer and this transfer actually *outperforms* a scene CNN trained on a large scene dataset. These results are significant given that 1) MFA training takes only a few hours versus training a CNN, and 2) transfer requires a much smaller scene dataset.

III.F Acknowledgements

The text of Chapter III, is based on material as it appears in: M. Dixit and N. Vasconcelos, "Object based scene representations using Fisher scores of local subspace projections", Neural Information Processing Systems (NIPS), Barcelona, Spain, 2016. and M. Dixit, Y. Li and N. Vasconcelos, "Bag-of-Semantics representation for object-to-scene transfer", [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material.

Chapter IV

Attribute Trajectory Transfer for Data Augmentation



Tables with depth in the range of $t_0 \in [1, 2]$ meters.
Chairs with depth in the range $t_0 \in [1, 2]$ meters.
Table
Training data
Chairs
Training data
$\gamma(\mathbf{x}) = 1.3$ [m]
$\gamma(\hat{\mathbf{x}}) \approx t$[m]
$\mathbf{x}$
$\mathbf{x} \mapsto \hat{\mathbf{x}} = \phi(\mathbf{x}, t_0, t)$
learn $\phi(\cdot, t_0, t)$
(using a pretrained γ)
Instance $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$
from a new object class

Figure IV.1: An illustration of the proposed trajectory learning and transfer method. Given a predictor $\gamma : \mathcal{X} \to \mathbb{R}_+$ of some object attribute (e. g., depth or pose), we propose to *learn* a mapping of object features $\mathbf{x} \in \mathcal{X}$, such that (1) the new synthetic feature $\hat{\mathbf{x}}$ is "close" to $\mathbf{x}$ (to preserve object identity) and (2) the predicted attribute value $\gamma(\hat{\mathbf{x}}) = \hat{t}$ of $\hat{\mathbf{x}}$ matches a desired object attribute value t, i. e., $t - \hat{t}$ is small. In this illustration, we learn a mapping for features with associated *depth* values in the range of 1-2 [m] to t [m] and apply this mapping to an instance of a new object class. In our approach, this mapping is learned in an *object-agnostic* manner. With respect to our example, this means that *all* training data from 'chairs' and 'tables' is used to a learn feature synthesis function ϕ.

IV.A Introduction

Cnvolutional neural networks (CNNs), trained on large scale data, have significantly advanced the state-of-the-art on traditional vision problems such as object recognition [45, 76, 81] and object detection [29, 71]. Success of these networks is mainly due to their high selectivity for semantically meaningful visual concepts, e. g., objects and object parts [93]. In addition to ensuring good performance on the problem of interest, this property of CNNs also allows for *transfer* of knowledge to several other vision tasks [21, 31, 12, 19]. The object recognition network of [45], e. g., has been successfully used for object detection [29, 71], scene classification [31, 19], texture classification [12] and domain adaptation [21], using various transfer mechanisms.

CNN-based transfer is generally achieved either by *finetuning* a pre-trained network, such as in [45], on a new image dataset or by designing a new image representation on such a dataset based on the activations of the pre-trained network layers [21, 31, 19, 12]. Recent proposals of transfer have shown highly competitive performance on different predictive tasks with a modest amount of new data (as few as 50 images per class). The effectiveness of transfer-based methods, however, has not yet been tested under more severe constraints such as in a *few-shot* or a *one-shot* learning scenario. In these problems, the number of examples available for learning may be as few as one per class. Fine-tuning a pre-trained CNN with millions of parameters to such inadequate datasets is clearly not a viable option. A one-shot classifier trained on CNN activations will also be prone to over-fitting due to the high dimensionality of the feature space. The only way to solve the problem of limited data is to *augment* the training corpus by obtaining more examples for the given classes.

While augmentation techniques can be as simple as flipping, rotating, adding noise, or extracting random crops from images [45, 11, 94], *task-specific*, or *guided* augmentation strategies [10, 34, 72, 62] have the potential to generate more realistic synthetic samples. This is a particularly important issue, since performance of CNNs heavily relies on sufficient coverage of the variability that we expect in unseen testing data. In scene recognition, we desire, for example, sufficient variability in the constellation and transient states of scene categories (c. f. [47]), whereas in object recognition, we desire variability in the specific incarnations of certain objects, lighting conditions, pose, or depth, just to name a few. Unfortunately, this variability is often dataset-specific and can cause substantial bias in recognition results [83].

An important observation in the context of our work is that augmentation is typically performed on the image, or video level. While this is not a problem with simple techniques, such as flipping or cropping, it can become computationally expensive if more elaborate augmentation techniques are used. We argue that, in specific problem settings, augmentation might as well be performed in *feature space*, especially in situations where features are input to subsequent learning steps.

This is common, e. g., in recognition tasks, where the softmax output of trained CNNs is often not used directly, but activations at earlier layers are input to an external discriminant classifier.

Contribution: We propose an approach to augment the training set with *feature descriptors* instead of images. Specifically, we advocate an augmentation technique that learns to synthesize features, guided by desired values for a set of object attributes, such as depth or pose. An illustration of this concept is shown in Fig. Figure IV.1. We first train a fast RCNN [29] detector to identify objects in 2D images. This is followed by training a neural network regressor which predicts the 3D attributes of a detected object, namely its depth from the camera plane and pose. An encoder-decoder network is then trained which, for a detected object at a certain depth and pose, will "hallucinate" the changes in its RCNN features for a set of desired depths/poses. Using this architecture, for a new image, we are able to augment existing feature descriptors by an auxiliary set of features that correspond to the object changing its 3D position.

Organization: Sec. IV.B reviews prior work. Sec. IV.C presents the basic idea of attribute trajectory transfer that motivates our work. In sec. IV.D we introduce a novel architecture that approximates attribute-trajectories with incomplete data and allows their transfer for the purpose of augmentation. The key component of this architecture is the encoder-decoder MLP which is introduced as feature regressor in this section. Sec. IV.E demonstrates that AGA in feature space improves one-shot object recognition and object-based scene recognition performance on previously unseen classes. Sec. IV.F delves deeper into the proposed feature augmentation method. Extensive analysis regarding the quality of synthetic data generated by our algorithm is provided. Sec. IV.G concludes the paper with a discussion and an outlook on potential future directions.

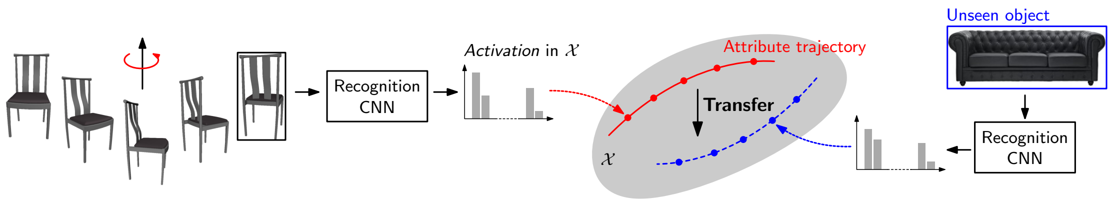


Figure IV.2: Schematic illustration of data augmentation by transferring an approximated attribute trajectory (here, for variation in object pose).

IV.B Related work

Our review of related work primarily focuses on *data augmentation* strategies. While many techniques have been proposed in the context of training deep neural networks to avoid over-fitting and to increase variability in the data, other (sometimes closely related) techniques have previously appeared in the context of one-shot and transfer learning. We can roughly group existing techniques into (1) *generic*, computationally cheap approaches and (2) task-specific, or guided approaches that are typically more computationally involved.

As a representative of the first group, Krizhevsky et. al. [45] leverage a set of label-preserving transformations, such as patch extraction + reflections, and PCA-based intensity transformations, to increase training sample size. Similar techniques are used by Zeiler and Fergus [94]. In [11], Chatfield and Zisserman demonstrate that the augmentation techniques of [45] are not only beneficial for training deep architectures, but shallow learning approaches equally benefit from such *simple* and *generic* schemes.

In the second category of guided-augmentation techniques, many approaches have recently been proposed. In [10], e. g., Charalambous and Bharath employ guided-augmentation in the context of gait recognition. The authors suggest to simulate synthetic gait video data (obtained from avatars) with respect to various confounding factors (such as clothing, hair, etc.) to extend the training corpus. Similar in spirit, Rogez and Schmid [72] propose an image-based synthesis engine for augmenting existing 2D human pose data by photorealistic images with greater pose variability. This is done by leveraging 3D motion capture (MoCap) data. In [62], Peng et. al. also use 3D data, in the form of CAD models, to render synthetic images of objects (with varying pose, texture, background) that are then used to train CNNs for object detection. It is shown that synthetic data is beneficial, especially in situations where few (or no) training instances are available, but 3D CAD models are. Su et. al. [79] follow a similar pipeline of rendering images from 3D models for viewpoint estimation, however, with substantially more synthetic data obtained, e. g., by deforming existing 3D models before rendering.

Another (data-driven) guided augmentation technique is introduced by [34]. The authors propose to *learn* class-specific transformations from external training data, instead of manually specifying transformations as in [45, 94, 11]. The learned transformations are then applied to the samples of each class. Specifically, diffeomorphisms are learned from data and encouraging results are demonstrated in the context of digit recognition on MNIST. Notably, this strategy is conceptually similar to earlier work by Miller et. al. [58] on one-shot learning, where the authors synthesize additional data for digit images via an iterative process, called *congealing*. During that process, external images of a given category are aligned by optimizing over a class of geometric transforms (e. g., affine transforms). These transformations are then applied to single instances of the new classes to increase data for one-shot learning.

Marginally related to our work, we remark that alternative approaches to implicitly learn spatial transformations have been proposed. For instance, Jaderberg et. al. [39] introduce *spatial transformer* modules that can be injected into existing deep architectures to implicitly capture spatial transformations inherent in the data, thereby improving invariance to this class of transformations.

While *all* previously discussed methods essentially propose *image-level* augmentation to train CNNs, our approach is different in that we perform augmentation in *feature space*. Along these lines, the approach of Kwitt et. al. [47] is conceptually similar to our work. In detail, the authors suggest to learn how features change as a function of the strength of certain transient attributes (such as sunny, cloudy, or foggy) in a scene-recognition context. These models are then transferred to previously unseen data for one-shot recognition. There are, however, two key differences between their approach and ours. First, they require datasets labeled with *attribute trajectories*, i. e., all variations of an attribute for every instance of a class. We, on the other hand, make use of conventional datasets that seldom carry such extensive labeling. We elaborate on this point in the next section. Second important difference is that, the augmenters in [47] are simple linear regressors trained in a *scene-class specific* manner. This imposes limits on the capacity of the model as well as complicates knowledge transfer across different classes. We, in contrast, learn deep non-linear models in a *class-agnostic* manner which enables a straightforward application to recognition in transfer settings.

IV.C Attribute Trajectory Transfer

The proposed data augmentation method implements a visual *attribute trajectory transfer* to generate non-trivial copies from a given image. A visual attribute, in this case, represents a common property of objects that manifests visually and can be quantified with a scalar value $s \in \mathbb{R}$, e.g. object pose (in degrees) or depth from camera (in meters). In a predefined visual feature space $\mathcal{X} \subset \mathbb{R}^D$, variations in object attributes can be learned as a smooth trajectory. The learned trajectory of attributes, e.g. pose, can then be *transferred* to an unseen object to simulate pose transformations. Fig Figure IV.2 demonstrates this process where pose trajectory for a "chair" is learned in $\mathcal{X}$ using a sequence of chair images captured in multiple poses. The learned trajectory of chair poses can be used to hallucinate pose transformations in a previously unseen object such as a "couch". The process of trajectory transfer, thus, naturally generates synthetic data points $\hat{\mathbf{x}} \in \mathcal{X}$ which correspond to deterministic changes in an attribute of the original image $\mathbf{x}$.

Attribute trajectory transfer is used with some success by Kwitt et. al. [47] for the purpose of guided data augmentation. They propose to generate synthetic examples from real scene images by regressing along simple trajectories of transient scene attributes such as degree of “sunny-ness”, “foggy-ness”, “amount of snow” etc in a CNN generated feature space $\mathcal{X}$ [96]. An obvious limitation of this approach, however, is the massive amount of data needed to properly learn feature trajectories. Ideally, trajectory learning requires a sufficiently large labeled dataset $\mathcal{D} = \{(\mathcal{T}_1, y_1), (\mathcal{T}_2, y_2), \ldots, (\mathcal{T}_N, y_N)\}$, where each $\mathcal{T}_i = \{(\mathcal{I}_{ik}, s_{ik})\}_{k=0}^{t}$ denotes a trajectory or sequence of images $\mathcal{I}_{ik}$ captured for an instance of class $y_i \in \{1, 2, \ldots C\}$ while steadily varying one of its attributes (e. g., orientation, illumination etc.) in steps of Δs according to $s_{ik} = s_{i0} + k\Delta s$. An example of a pose trajectory in image space is shown in Fig Figure IV.3 for a chair. Availability of image trajectories, theoretically, ensures a simple implementation of guided data augmentation. In [47], for example, trajectories captured from a set of fixed webcams are used to learn scene-specific linear mappings $\mathbf{x} = Ws + b$ from attributes s to features $\mathbf{x}$. Synthetic feature points $\hat{\mathbf{x}}$ are then obtained given a new example $\mathbf{x}^*$ for a desired shift in the attribute value $t - s^*$ using the regression function $\phi(\mathbf{x}^*, s^*, t) = W(t - s^*) + \mathbf{x}^*$. Adequately sized datasets of image trajectories, however, are very difficult to obtain. Known natural image datasets in vision seldom capture extensive attribute variations for all instances of different visual classes. Even the recently introduced large scale corpora only provide millions of still images and no trajectories for class instances. An ImageNet [17] “chair” class, for example, has a thousand images, each from a different chair captured in a single pose and position. Similarly, in the Places scene dataset [96], a “street” scene class, contains a single image each of different street scenes. Attribute trajectory learning and transfer on standard vision datasets, therefore, is not straightforward at all.

In this work we reformulate the problem of trajectory transfer so that it can be extended to standard vision datasets. Datasets of natural images are generally of the form $\mathcal{D} = \{(\mathcal{I}_1, s_1, y_1), (\mathcal{I}_2, s_2, y_2), \dots, (\mathcal{I}_N, s_N, y_N)\}$, where each image $\mathcal{I}_k$ represents an instance of a class $y_k \in \{1, 2, \dots, C\}$ captured in a condition (attribute) that can be quantified by a scalar $s_k \in \mathbb{R}$ (attribute value). Since the first step of trajectory transfer is trajectory learning, the absence of image sequences poses an immediate problem. We overcome this with the help of a two step *attribute trajectory approximation* approach. First, using a suitable feature space $\mathcal{X}$ generated by an object CNN, we train a predictor of attribute values $\gamma(\mathbf{x}), \mathbf{x} \in \mathcal{X}$. Next we train a feature synthesizer function $\phi(\mathbf{x}, \gamma(\cdot), t)$ which perturbs the image representation x such that the new data point $\hat{\mathbf{x}}$ achieves a specified change in the attribute value from $t_0 = \gamma(\mathbf{x})$ to $t = \gamma(\hat{\mathbf{x}})$ for the smallest displacement in an L_2 sense. The synthesis function $\phi(.)$, therefore, effectively learns to translate a data point $(\mathbf{x}, t_0)$ to another location $(\hat{\mathbf{x}}, t)$ on the attribute trajectory. A sufficient number of such $\phi(\cdot, t_0, t)$'s learned between different pairs of current and desired attribute values (t_0, t) can be used to traverse the entire trajectory of attribute variations. The proposed idea of attribute trajectory approximation is illustrated in Fig Figure IV.5. The architectural details of our framework as well as the learning procedure is discussed in the following section.

IV.D Architecture

The architecture for attribute trajectory approximation, as depicted in Fig. Figure IV.5, comprises of an object CNN stage, an attribute value predictor and a feature generator module. Activations from the higher layers of any, off-the-shelf, object detection or recognition network may be used as the space $\mathcal{X} \subset \mathbb{R}^D$ for image description. The representation $\mathbf{x} \in \mathcal{X}$ of an image $\mathcal{I}$ is then processed by an attribute predictor that predicts the pose or depth t_0 of the example. The feature generator implements a transformation of the current feature $\mathbf{x}$ such that the new point matches a specified pose or depth t. Implementation of the attribute predictor and the feature synthesis module is described in this

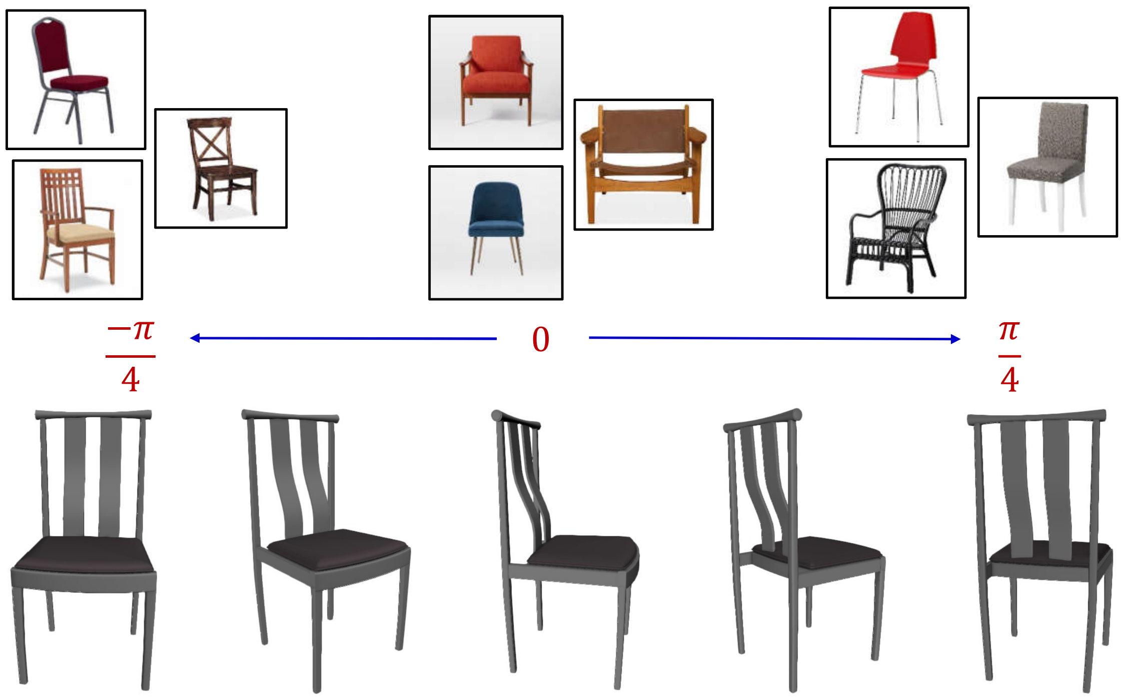


Figure IV.3: Distinction between class instances in standard natural image datasets (*top*) and image trajectories (*bottom*).

section.

IV.D.1 Attribute regression

Let $\mathcal{A}$ be the set of all attributes that are available as labels for a dataset of object images. As mentioned earlier, these could be object properties such as pose, depth, volume, height etc. An attribute $A \in \mathcal{A}$ takes on scalar values $s \in \mathbb{R}$ which, we assume, can be accurately predicted using the image representation $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$. We train an attribute regressor function $\gamma : \mathcal{X} \to \mathbb{R}$ for every attribute A in the set $\mathcal{A}$. Each regressor takes as input the image descriptor $\mathbf{x}$ and predicts its strength or value, i. e., $\gamma(x) = t_0$. While γ could, in principle, be implemented by a variety of approaches, such as support vector regression [22] or Gaussian processes [9], we use a two-layer neural network instead, to accomplish this task. This is not an arbitrary choice, as it will later enable us to easily re-use this building block in the learning stage of the synthesis function(s) ϕ. The

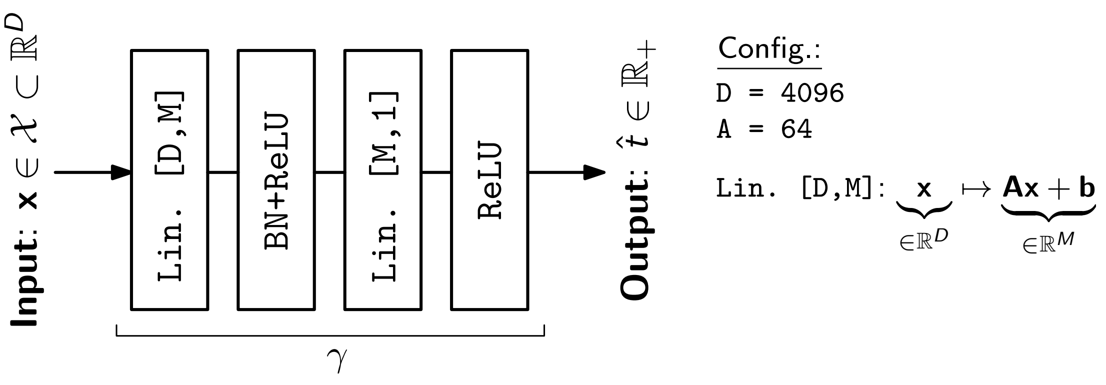

Input: $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$
Lin. [D,M]
BN+ReLU
Lin. [M,1]
ReLU
γ
Output: $\hat{t} \in \mathbb{R}_+$
Config.:
D = 4096
A = 64
Lin. [D,M]: $\underbrace{\mathbf{x}}_{\in\mathbb{R}^D} \mapsto \underbrace{\mathbf{Ax}+\mathbf{b}}_{\in\mathbb{R}^M}$

Figure IV.4: Architecture of the attribute regressor γ.

architecture of the attribute regressor is shown in Fig. Figure IV.4, consisting of two linear layers, interleaved by batch normalization (BN) [36] and rectified linear units (ReLU) [60]. While this architecture is admittedly simple, adding more layers did not lead to significantly better results in our experiments. Nevertheless, the design of this component is problem-specific and could easily be replaced by more complex variants, depending on the characteristics of the attributes that need to be predicted.

Learning. The attribute regressor can easily be trained from a collection of N training tuples $\{(x_i, s_i)\}_{i=1}^N$ for each attribute using an L_2 loss objective between $\gamma(x)$ and ground truth s.

IV.D.2 Feature regression

The key component of our data augmentation framework is the feature synthesis function ϕ which implements a mapping,

$$\phi : \mathcal{X} \times \mathbb{R} \times \mathbb{R} \to \mathcal{X}, \ (\mathbf{x}, \gamma(\mathbf{x}), t) \mapsto \hat{\mathbf{x}}, \quad \text{s.t.} \quad \gamma(\hat{\mathbf{x}}) \approx t \ . \tag{IV.1}$$

This mapping accepts, as input, the image representation $\mathbf{x} \in \mathcal{X}$, the predicted value $t_0 = \gamma(\mathbf{x})$ for an attribute A and a desired scalar value t, at which we want the attribute A to be. The function $\phi(\mathbf{x}, t_0, t)$, then, generates a new representation $\hat{\mathbf{x}}$, such that its predicted $\gamma(\hat{\mathbf{x}})$ matches the desired t. A bank of functions ϕ learned

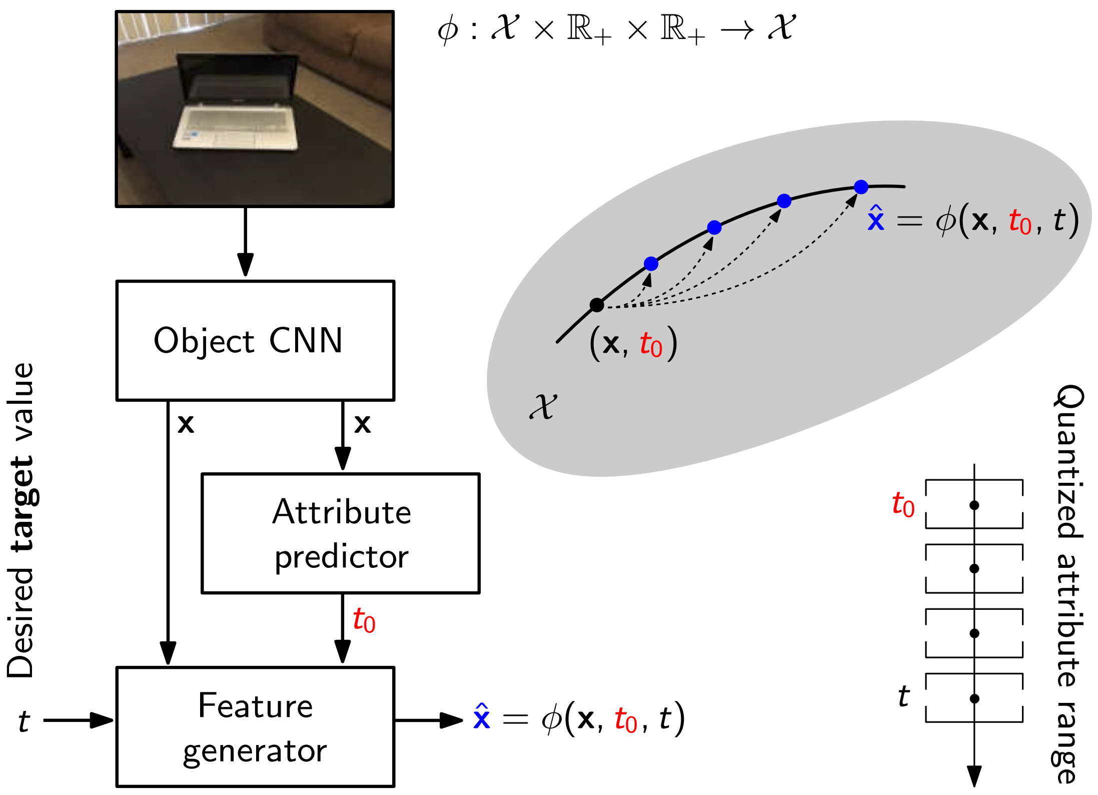

$\phi : \mathcal{X} \times \mathbb{R}_+ \times \mathbb{R}_+ \to \mathcal{X}$
Object CNN
x
x
Attribute predictor
t_0
Feature generator
t
Desired **target** value
$\hat{\mathbf{x}} = \phi(\mathbf{x}, t_0, t)$
$(\mathbf{x}, t_0)$
$\hat{\mathbf{x}} = \phi(\mathbf{x}, t_0, t)$
$\mathcal{X}$
t_0
t
Quantized attribute range

Figure IV.5: Illustration of attribute-guided trajectory approximation. For a given image, the object CNN generates a feature $\mathbf{x}$ and the attribute predictor generates the current pose/depth t_0. The feature generator function $\phi(x, t_0, t)$ translates the data point $\mathbf{x}$ to another point $\hat{\mathbf{x}}$ in the CNN feature space such that the pose/depth of the new sample matches a desired value t. The entire trajectory of pose/depth variations can be approximated by learning one feature generator ϕ for every possible pair of (t_0, t) in the attribute range.

for sufficient pairs of (t_0, t), will allow translation from a given location $\mathbf{x}$ to any other location on $\mathcal{X}$ that lies on the trajectory of variation in attribute A. For ease of implementation, therefore, we quantize the range $s \in \mathbb{R}$ of attribute A into I intervals $[l_i, h_i]$, where l_i, h_i denote the lower and upper bounds of the i-th interval. A function ϕ_i^k is trained to translate $\mathbf{x}$ such that the current attribute $t_0 \in [l_i, h_i]$ changes to the desired attribute $t \in [l_k, h_k]$. In our illustration of Fig. Figure IV.1, e. g., we show a ϕ learned for depth attribute from interval $[l, h] = [1, 2]$ to a desired value $3[m]$ representing the interval $[2.5, 3.5]$. While learning separate synthesis functions simplifies the problem, it requires a good a-priori *attribute (strength) predictor*, since, otherwise, we could not decide which ϕ_i^k to use. During testing, we (1) predict the object's attribute value from its original feature $\mathbf{x}$, i. e., $\gamma(\mathbf{x}) = t_0$, identify the input interval $t_0 \in [l_i, h_i]$ and then (2) synthesize additional features as $\hat{\mathbf{x}} = \phi_i^k(\mathbf{x})$ for translation to all other intervals $[l_k, h_k], i \neq k$.

To implement[1] ϕ, we design an encoder-decoder architecture, reminiscent of a conventional autoencoder [3]. Our objective, however, is not to encode and then reconstruct the input, but to produce an output that resembles a feature descriptor of an object at a desired attribute value. In other words, the *encoder* essentially learns to extract the essence of features; the *decoder* then takes the encoding and decodes it to the desired result. In general, we can formulate the optimization problem as

$$\min_{\phi \in \mathcal{C}} L(\mathbf{x}, t; \phi) = (\gamma(\phi(\mathbf{x})) - t)^2 \ , \tag{IV.2}$$

where the minimization is over a suitable class of functions $\mathcal{C}$. Notably, when implementing ϕ as an encoder-decoder network with an appended (pre-trained) attribute predictor (see Fig. Figure IV.6) and loss $(\gamma(\phi(\mathbf{x})) - t)^2$, we have little control over the decoding result in the sense that we cannot guarantee that the *identity* of the input is preserved. This means that features from a particular object class might map to features that are no longer recognizable as this class, as the encoder-decoder will *only* learn to "fool" the attribute predictor γ. For that reason, we add a *regularizer* to the objective of Eq. (IV.2), i. e., we require the decoding result to be close, e. g., in the 2-norm, to the input. This changes the optimization problem of Eq. (IV.2) to

$$\min_{\phi \in \mathcal{C}} L(\mathbf{x}, t; \phi) = (1 - \alpha) \underbrace{(\gamma(\phi(\mathbf{x})) - t)^2}_{\text{Mismatch penalty}} + \alpha \underbrace{\|\phi(\mathbf{x}) - \mathbf{x}\|^2}_{\text{Regularizer}} \ . \tag{IV.3}$$

[1]We omit the sub-/superscripts for readability.

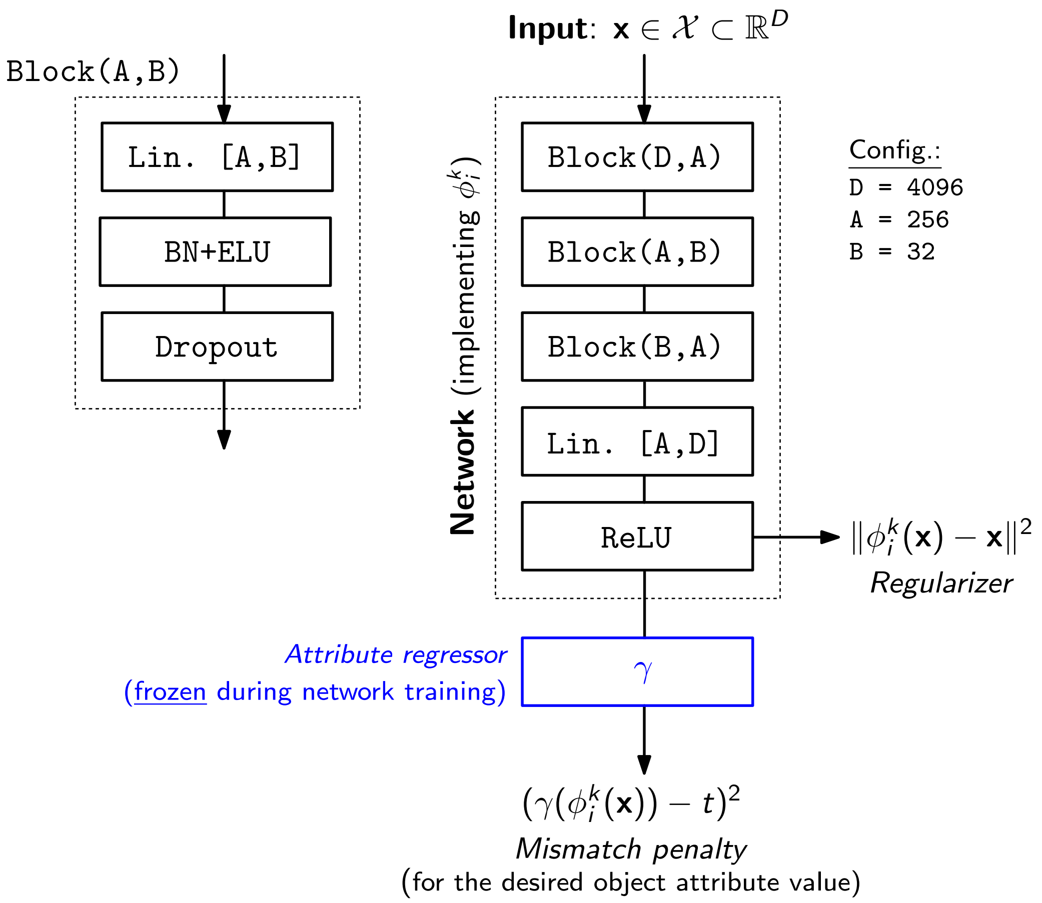

Block(A,B)
Lin. [A,B]
BN+ELU
Dropout
Input: $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^D$
Network (implementing ϕ_i^k)
Block(D,A)
Block(A,B)
Block(B,A)
Lin. [A,D]
ReLU
Config.:
D = 4096
A = 256
B = 32
$\|\phi_i^k(\mathbf{x}) - \mathbf{x}\|^2$
Regularizer
Attribute regressor
(frozen during network training)
γ
$(\gamma(\phi_i^k(\mathbf{x})) - t)^2$
Mismatch penalty
(for the desired object attribute value)

Figure IV.6: Illustration of the encoder-decoder network for AGA. During *training*, the attribute regressor γ is appended to the network, whereas, for *testing* (i. e., feature synthesis) this part is removed. When learning ϕ_i^k, the input $\mathbf{x}$ is such that the associated attribute value t_0 is within $[l_i, h_i]$ and one ϕ_i^k is learned per desired attribute value $t \in [l_k, h_k]$.

Interpreted differently, this resembles the loss of an autoencoder network with an added *target attribute mismatch* penalty. The encoder-decoder network that implements the function class $\mathcal{C}$ to learn ϕ is shown in Fig. Figure IV.6. The core building block is a combination of a linear layer, batch normalization, ELU [13], followed by dropout [78]. After the final linear layer, we add one ReLU layer to enforce $\hat{\mathbf{x}} \in \mathbb{R}_+^D$.

Learning. Training the encoder-decoder network of Fig. Figure IV.6 requires an a-priori trained attribute regressor γ for each given attribute $A \in \mathcal{A}$.

During training, this attribute regressor is appended to the network and its *weights are frozen*. Hence, only the encoder-decoder weights are updated. To train one ϕ_i^k for each interval $[l_i, h_i]$ of the object attribute range and a desired object attribute value t_k, we partition the training data from the external corpus into subsets $\mathcal{S}_i$, such that $\forall(\mathbf{x}_n, s_n) \in \mathcal{S}_i : s_n \in [l_i, h_i]$. One ϕ_i^k is learned from $\mathcal{S}_i$ for each desired object attribute value $t \in [l_k, h_k]$. As training is in feature space $\mathcal{X}$, we have no convolutional layers and consequently training is computationally cheap. For testing, the attribute regressor is removed and only the trained encoder-decoder network (implementing ϕ_i^k) is used to synthesize features. Consequently, given $|\mathcal{A}|$ attributes, I intervals per attribute, we obtain $|\mathcal{A}| \cdot I^2$ synthesis functions.

Attribute based Feature Extrapolation

The approach for training shown in (IV.3) is unique, in that it facilitates learning transformations ϕ without special image sets (pairs, trajectories etc). The primary goal of ϕ's is to learn a mapping from a data point $(\mathbf{x}, t_0)$ to another point $(\hat{\mathbf{x}}, t)$. In an ideal case, this could be framed as an alignment problem between sample pairs $(\mathbf{x}, t_0)$ and $(\hat{\mathbf{x}}, t)$ available in the training database. Our setting, however, does not assume the availability of special data with paired images or image sequences (e.g. two or more images of the same couch in different poses). We cannot, therefore, avail of the point-to-point alignment/translation formulations often used in domain adaptation problems [46, 69] and, more recently, in image-to-image generation [37]. In our case, the desired output $\hat{\mathbf{x}}$ to which original point $\mathbf{x}$ must be translated, is *hidden*. All we have, for training, is $\mathbf{x}$, its current attribute value t_0 and the final desired attribute value t. To estimate the unknown $\hat{\mathbf{x}}$ under these constraints, we leverage the trained attribute predictor $\gamma(.)$ which induces a measure on the feature space $\mathcal{X}$. The feature synthesizers ϕ essentially learn to translate the original sample $\mathbf{x}$ by the smallest amount so as to affect the specified change in its attribute, as measured by $\gamma(.)$. Functions ϕ, thus, learn to *imagine* changes to an item (an image) without having explicit images depicting these change. The transformation is, by nature, an *extrapolation* along $\mathcal{X}$ *guided* by $\gamma(\cdot)$.

Another important feature of our approach is that it allows *modularity*. The feature space $\mathcal{X}$ and the attribute predictors γ can be learned on a completely different generic dataset of objects. A pre-trained CNN space $\mathcal{X}$ as well as attribute predictors γ trained on it can be plugged in to the objective (IV.3) for feature synthesis. A γ that predicts "pose" of a generic collection of common indoor objects, can be easily used to train a pose guided data synthesizer (ϕ) for a set of "unseen" objects. The construction, thus, naturally allows *transfer* of attribute trajectories across object datasets.

Relation to Direct Gradient Ascent

The objective in (IV.2) is the same as the loss for which the attribute regressor $\gamma(.)$ is trained with on space $\mathcal{X}$. Theoretically $\mathbf{x}$ may be modified to achieve the desired attribute value of t, using backpropagated gradient updates within $\gamma(.)$. These are,

$$\frac{\partial E}{\partial \mathbf{x}} = -2(1-\alpha)\epsilon\frac{\partial \gamma}{\partial \mathbf{x}} \tag{IV.4}$$

where ϵ stands for $t - \gamma(\mathbf{x})$. Such a simplistic update rule, however, will most likely lead to spurious results. It is, after all a well known tendency among deep neural networks to be fooled by noisy inputs generated with uncontrolled gradient ascent [61]. Adding an L_2 constraint on the objective as in (IV.3) could help soften the gradient steps as follows,

$$\frac{\partial E}{\partial \mathbf{x}} = -2(1-\alpha)\epsilon\frac{\partial \gamma}{\partial \mathbf{x}} - 2\alpha\delta \tag{IV.5}$$

where $\delta = |\mathbf{x}_0 - \mathbf{x}|$, where $\mathbf{x}_0$ is an anchor point for the features. Notice that both components of the loss counteract each other. These updates may be seen as *inference* of the hidden $\hat{\mathbf{x}}$ for every given data point $\mathbf{x}$ and a scalar t. All that is required is a pre-trained $\gamma()$ that computes and propagates the gradients in (IV.4) or (IV.5). Our method introduces a feature regressor stage in between input x and the attribute predictor. Its role is to allow *learningability* of traversal in $\mathcal{X}$. The regressor is designed as an encoder decoder with input $\mathbf{x}$ and output $\hat{\mathbf{x}}$ with parameters that can be tuned for the transformation. The procedure of learning the regressor can be seen as involving *inference* of the hidden $\hat{\mathbf{x}}$ that achieves the desired pose/depth t as well as *estimation* of the parameters θ of the module. While $\hat{\mathbf{x}}$ is updated using (IV.5), the parameters are updated with,

$$\frac{\partial E}{\partial \theta} = \left[-2(1-\alpha)\epsilon\frac{\partial \gamma}{\partial \hat{\mathbf{x}}} - 2\alpha\delta\right] \odot \frac{\partial \hat{\mathbf{x}}}{\partial \theta} \tag{IV.6}$$

Advantage of a trainable system over backprop based feature updates is that the former can produce generalizable data points which are necessary for the success of the few-shot transfer task.

IV.E Experiments

We first discuss the generation of adequate training data for the encoder-decoder network, then evaluate every component of our architecture separately and eventually demonstrate its utility on (1) one-shot object recognition in a transfer learning setting and (2) one-shot scene recognition.

Dataset. We use the SUN RGB-D dataset from Song et. al. [77]. This dataset contains 10335 RGB images with depth maps, as well as detailed annotations for more than 1000 objects in the form of 2D and 3D bounding boxes. In our setup, we use object depth and pose as our attributes, i. e., $\mathcal{A} = \{\texttt{Depth}, \texttt{Pose}\}$. For each ground-truth 3D bounding box, we extract the depth value at its centroid and obtain pose information as the rotation of the 3D bounding box about the vertical y-axis. In all experiments, we use the first 5335 images as our *external database*, i. e., the database for which we assume availability of attribute annotations. The remaining 5000 images are used for testing/evaluation; more details are given in the specific experiments.

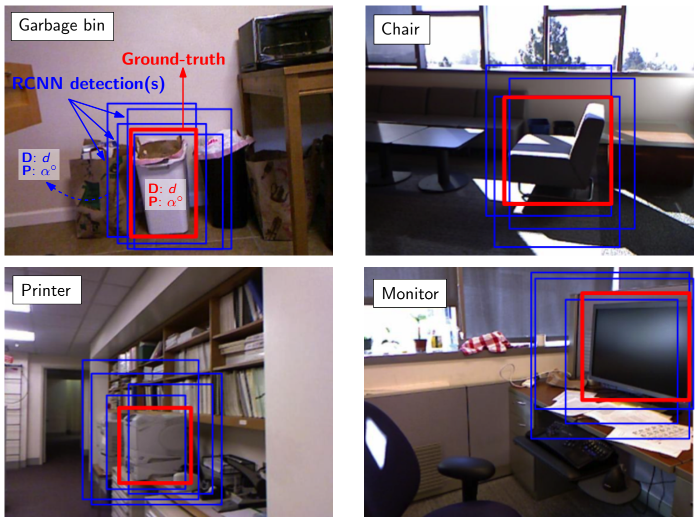


Figure IV.7: Illustration of *training data generation* for AGA training. *First*, we obtain fast RCNN [29] activations (e. g., `FC7` layer) of Selective Search [85] proposals that overlap with 2D ground-truth bounding boxes (IoU > 0.5) and scores > 0.7 (for a particular object class) to generate a sufficient amount of training data. *Second*, attribute values (i. e., depth $\mathbf{D}$ and pose $\mathbf{P}$) of the corresponding 3D ground-truth bounding boxes are associated with the proposals (best-viewed in color).

Training data. Notably, in SUN RGB-D, the number of instances of each object class are not evenly distributed, simply because this dataset was not specifically designed for object recognition tasks. Consequently, images are also not object-centric, meaning that there is substantial variation in the location of objects, as well as the depth and pose at which they occur. This makes it difficult to extract a sufficient and balanced number of feature descriptors per object class, if we would *only* use the ground-truth bounding boxes to extract training data. We circumvent this problem by leveraging the fast RCNN detector of [29] with object proposals generated by Selective Search [85]. In detail, we finetune the ImageNet model from [29] to SUN RGB-D, using the same 19 objects as in [77]. We then run the detector on all images from our training split and keep the proposals with detection scores > 0.7 and a sufficient overlap (measured by the IoU >0.5) with the 2D ground-truth bounding boxes. This is a simple augmentation technique to increase the amount of available training data. The activations for these proposals generated by a suitable layer of the RCNN are used as our features $\mathbf{x}$. Each proposal that remains after overlap and score thresholding is annotated by the attribute information of the corresponding ground-truth bounding box in 3D. As this strategy generates a larger number of descriptors (compared to the number of ground-truth bounding boxes), we can evenly balance the training data in the sense that we can select an equal number of detections per object class to train (1) the attribute regressor and (2) the encoder-decoder network. Training data generation is illustrated in Fig. Figure IV.7 on four example images.

Implementation. The attribute regressor and the encoder-decoder network are implemented in `PyTorch`[2]. All models are trained using `Adam` [43]. For the attribute regressor, we train for 150 epochs with a batch size of 300 and an initial learning rate of 0.001. The learning rate drops by 10 every 50th epoch. The encoder-decoder network is also trained for 150 epochs with the same learning rate schedule, but with a batch size of 64. The dropout probability during training is

[2]`http://pytorch.org/`

set to 0.25. No dropout is used for testing. For our classification experiments, we use a linear C-SVM, as implemented in `liblinear` [23]. On a Linux system, running Ubuntu 16.04, with 128 GB of memory and one NVIDIA Titan X, training one model (i. e., one $\phi(\cdot, t_0, t)$ for a specific choice of t, t_0) takes $\approx$ 30 seconds. The relatively low demand on computational resources highlights the advantage of AGA in feature space, as no convolutional layers need to be trained. As the feature space $\mathcal{X}$, any of the higher layers of the RCNN can be due to their semantic nature which is critical for problems of *transfer* [21, 31, 19]. In the initial few experiments, therefore, we train and evaluate our system using the last fully connected layer of the RCNN network denoted `FC7`. All trained models+source code are publicly available online[3].

IV.E.1 Attribute regression

Our strategy to augment data requires training of attribute regressors γ in a manner *agnostic* to the object class. Since attribute value γ is the only supervisory signal that guides the proposed augmentation, performance of AGA is tied to the accuracy of attribute prediction. In this section, we evaluate the performance of our *object agnostic* depth and pose predictors trained using the architecture in sec. IV.D.1.

Table Table IV.1 lists the median-absolute-error (MAE) of depth (in [m]) and pose (in [deg]) prediction per object. We train on instances of 19 object classes ($\mathcal{S}$) in our training split of SUN RGB-D and test on instances of the same object classes, but extracted from the testing split. As feature representation we use two top level activations of the RCNN namely `FC6` and `FC7`. We observe that the accuracy of depth prediction is quite high with an MAE of only 0.45 m for both features. Pose predictor on the other hand is a little more erroneous with an MAE of 45 deg. Pose estimation from 2D data, in general, is a substantially harder problem than depth estimation (which works remarkably well, even on a per-pixel

[3]`https://github.com/rkwitt/AGA`

Table IV.1: Median-Absolute-Error (MAE), for depth / pose, of the attribute regressor, evaluated on left-out (during training) instances of *19 objects* from SUN RGBD. In our setup, the pose estimation error quantifies the error in predicting a rotation around the z-axis. For reference, the range of of the object attributes in the training data is [0.2m, 7.5m] for `Depth` and [0°, 180°] for `Pose`.

Object Class	**Depth** (MAE [m])		**Pose** (MAE [deg])	
	`FC7`	`FC6`	`FC7`	`FC6`
`bathtub`	0.21	0.28	48.60	48.96
`bed`	0.43	0.46	54.76	53.77
`bookshelf`	0.62	0.73	53.56	50.88
`box`	0.77	0.71	46.84	41.50
`chair`	0.48	0.45	42.44	45.48
`counter`	0.57	0.23	48.78	39.34
`desk`	0.52	0.47	45.80	49.96
`door`	0.62	0.41	68.36	48.66
`dresser`	0.38	0.41	67.96	66.89
`garbage bin`	0.47	0.48	50.71	44.08
`lamp`	0.56	0.68	36.63	34.35
`monitor`	0.36	0.40	41.98	36.27
`night stand`	0.51	0.61	46.20	44.53
`pillow`	0.43	0.57	54.35	41.86
`sink`	0.26	0.25	60.08	54.09
`sofa`	0.48	0.43	46.42	41.94
`table`	0.47	0.43	43.29	44.12
`tv`	0.51	0.56	40.25	34.58
`toilet`	0.25	0.24	40.24	32.79
∅	0.47	0.46	44.45	44.95

level, c. f. [55]). Nevertheless, our recognition experiments (in Secs. IV.E.3 and IV.E.4) show that even with mediocre performance of the pose predictor (due to symmetry issues, etc.), augmentation along this dimension is still beneficial. Finally, we note that there isn't a significant difference between the performances using either `FC6` or `FC7` feature spaces. Therefore, unless otherwise specified, we use `FC7` as our feature representation in the remainder of this work.

IV.E.2 Feature regression

We assess the performance of our regressor(s) ϕ_i^k, shown in Fig. Figure IV.6, that are used for synthetic feature generation. In all experiments, we use an overlapping sliding window to bin the range of each attribute $A \in \mathcal{A}$ into I intervals $[l_i, h_i]$. In case of `Depth`, we set $[l_0, h_0] = [0, 1]$ and shift each interval by 0.5 meter; in case of `Pose`, we set $[l_0, h_0] = [0°, 45°]$ and shift by 25°. We generate as many intervals as needed to cover the full range of the attribute values in the training data. The bin-width / step-size were set to ensure a roughly equal number of features in each bin. For augmentation, we choose $0.5, 1, \ldots, \max(\texttt{Depth})$ as target attribute values for `Depth` and $45°, 70°, \ldots, 180°$ for `Pose`. This results in $T = 11$ target values for `Depth` and $T = 7$ for `Pose`.

We use two separate evaluation metrics to assess the performance of ϕ_i^k. *First*, we are interested in *how well* the feature regressor can generate features that correspond to the desired attribute target values. To accomplish this, we run each synthetic feature $\hat{\mathbf{x}}$ through the attribute predictor and assess the MAE, i. e., $|\gamma(\hat{\mathbf{x}}) - t|$, over all attribute targets t. Table Table IV.2 lists the average MAE, per object, for (1) features from object classes that were *seen* in the training data and (2) features from objects that we have never seen before. As wee can see from Table Table IV.2, MAE's for seen and unseen objects are similar, indicating that the encoder-decoder has learned to synthesize features, such that $\gamma(\hat{\mathbf{x}}) \approx t$.

Second, we are interested in *how much* synthesized features *differ* from original features. While we cannot evaluate this directly (as we do not have data from one particular object instance at multiple depths and poses), we can assess how "close" synthesized features are to the original features. The intuition here is that closeness in feature space is indicative of an object-identity preserving synthesis. In principle, we could simply evaluate $\|\phi_i^k(\mathbf{x}) - \mathbf{x}\|^2$, however, the 2-norm is hard to interpret. Instead, we compute the Pearson correlation coefficient ρ between each original feature and its synthesized variants, i. e., $\rho(\mathbf{x}, \phi_i^k(\mathbf{x}))$. As ρ ranges from $[-1, 1]$, high values indicate a strong linear relationship to the original features. Results are reported in Table Table IV.2. Similar to our previous results for MAE, we observe that ρ, when averaged over all objects, is slightly lower for objects that did not appear in the training data. This decrease in correlation, however, is relatively small.

In summary, we conclude that these results warrant the use of ϕ_i^k on feature descriptors from object classes that have *not* appeared in the training corpus. This enables us to test ϕ_i^k in transfer learning setups, as we will see in the following one-shot experiments of Secs. IV.E.3 and IV.E.4.

IV.E.3 One-shot object recognition

First, we demonstrate the utility of our approach on the task of one-shot object recognition in a transfer learning setup. Specifically, we aim to learn attribute-guided augmenters ϕ_i^k from instances of object classes that are available in an external, annotated database (in our case, SUN RGB-D). We denote this collection of object classes as our *source classes* $\mathcal{S}$. Given one instance from a collection of completely different object classes, denoted as the *target classes* $\mathcal{T}$, we aim to train a discriminant classifier C on $\mathcal{T}$, i. e., $C : \mathcal{X} \to \{1, \ldots, |\mathcal{T}|\}$. In this setting, $\mathcal{S} \cap \mathcal{T} = \emptyset$. Note that no attribute annotations for instances of object classes in $\mathcal{T}$ are available. This can be considered a variant of transfer learning, since we transfer knowledge from object classes in $\mathcal{S}$ to instances of object classes in $\mathcal{T}$, *without* any prior knowledge about $\mathcal{T}$.

Setup. We evaluate one-shot object recognition performance on three collections of previously unseen object classes in the following setup: First, we randomly select two sets of 10 object classes and ensure that each object class has at least 100 samples in the testing split of SUN RGB-D. We further ensure that no object class is in $\mathcal{S}$. This guarantees (1) that we have never seen the image, nor (2) the object class during training. Since, SUN RGB-D does not have object-centric images, we use the ground-truth bounding boxes to obtain the actual object crops. This allows us to tease out the benefit of augmentation without having to deal

Table IV.2: Quality assessment of feature regressor ϕ_i^k. Evaluation is presented in terms of (1) Pearson correlation (ρ) of *synthesized* and *original* features and (2) mean MAE of predicted attribute values of synthesized features, $\gamma(\phi_i^k(\mathbf{x}))$, w. r. t. the desired attribute values t. **D** indicates `Depth`-aug. features (MAE in [m]); **P** indicates `Pose`-aug. features (MAE in [deg]).

	Object	ρ	**D** (MAE)	ρ	**P** (MAE)
Seen objects	`bathtub`	0.75	0.10	0.68	3.99
	`bed`	0.81	0.07	0.82	3.30
	`bookshelf`	0.80	0.06	0.79	3.36
	`box`	0.74	0.08	0.74	4.44
	`chair`	0.73	0.07	0.71	3.93
	`counter`	0.76	0.08	0.77	3.90
	`desk`	0.75	0.07	0.74	3.93
	`door`	0.67	0.10	0.63	4.71
	`dresser`	0.79	0.08	0.77	4.12
	`garbage bin`	0.76	0.07	0.76	5.30
	`lamp`	0.82	0.08	0.79	4.83
	`monitor`	0.82	0.06	0.80	3.34
	`night stand`	0.80	0.07	0.78	4.00
	`pillow`	0.80	0.08	0.81	3.87
	`sink`	0.75	0.11	0.76	4.00
	`sofa`	0.78	0.08	0.78	4.29
	`table`	0.75	0.07	0.74	4.10
	`tv`	0.78	0.08	0.72	4.66
	`toilet`	0.80	0.10	0.81	3.70
	$\varnothing$	**0.77**	**0.08**	**0.76**	**4.10**
Unseen objects	`picture`	0.67	0.08	0.65	5.13
	`ottoman`	0.70	0.09	0.70	4.41
	`whiteboard`	0.67	0.12	0.65	4.43
	`fridge`	0.69	0.10	0.68	4.48
	`counter`	0.76	0.08	0.77	3.98
	`books`	0.74	0.08	0.73	4.26
	`stove`	0.71	0.10	0.71	4.50
	`cabinet`	0.74	0.09	0.72	3.99
	`printer`	0.73	0.08	0.72	4.59
	`computer`	0.81	0.06	0.80	3.73
	$\varnothing$	0.72	0.09	0.71	4.35

with confounding factors such as background noise. The two sets of object classes are denoted $\mathcal{T}_1$[4] and $\mathcal{T}_2$[5]. We additionally compile a third set of target classes $\mathcal{T}_3 = \mathcal{T}_1 \cup \mathcal{T}_2$ and remark that $\mathcal{T}_1 \cap \mathcal{T}_2 = \emptyset$. Consequently, we have two 10-class problems and one 20-class problem. For each object image in $\mathcal{T}_i$, we then collect RCNN `FC7` features.

As a *Baseline*, we "train" a linear C-SVM (on 1-norm normalized features) using only the single instances of each object class in $\mathcal{T}_i$ (SVM cost fixed to 10). Exactly the same parameter settings of the SVM are then used to train on the single instances + features synthesized by AGA. We repeat the selection of one-shot instances 500 times and report the average recognition accuracy. For comparison, we additionally list 5-shot recognition results in the same setup.

Remark. The design of this experiment is similar to [62, Section 4.3.], with the exceptions that we (1) *do not* detect objects, (2) augmentation is performed in feature space and (3) no object-specific information is available. The latter is important, since [62] assumes the existence of 3D CAD models for objects in $\mathcal{T}_i$ from which synthetic images can be rendered. In our case, augmentation *does not* require any a-priori information about the objects classes.

Results. Table Table IV.3 lists the classification accuracy for the different sets of one-shot training data. *First*, using original one-shot instances augmented by `Depth`-guided features (+**D**); *second*, using original features + `Pose`-guided features (+**P**) and *third*, a combination of both (+**D**, **P**); In general, we observe that adding AGA-synthesized features improves recognition accuracy over the *Baseline* in all cases. For `Depth`-augmented features, gains range from 3-5 percentage points, for `Pose`-augmented features, gains range from 2-4 percentage points on average. We attribute this effect to the difficulty in predicting object pose from 2D data, as can be seen from Table Table IV.1. Nevertheless, in both augmentation settings, the gains are statistically significant (w. r. t. the *Baseline*), as evaluated

[4] $\mathcal{T}_1$ = {`picture`, `whiteboard`, `fridge`, `counter`, `books`, `stove`, `cabinet`, `printer`, `computer`, `ottoman`}

[5] $\mathcal{T}_2$ = {`mug`, `telephone`, `bowl`, `bottle`, `scanner`, `microwave`, `coffee table`, `recycle bin`, `cart`, `bench`}

Table IV.3: One / few shot recognition accuracy (over 500 trials) for three object recognition tasks; *top*: one-shot, *bottom*: five-shot. Numbers in parentheses indicate the #classes. A '✓' indicates that the result is statistically different (at 5% sig.) from the *Baseline*. +**D** indicates adding `Depth`-aug. features to the one-shot instances; +**P** indicates addition of `Pose`-aug. features and +**D**, **P** denotes adding a combination of `Depth`-/`Pose`-aug. features.

	Baseline	`AGA`+**D**	`AGA`+**P**	`AGA`+**D**+**P**
		One-shot		
$\mathcal{T}_1$ (10)	33.74	38.32	37.25	39.10
$\mathcal{T}_2$ (10)	23.76	28.49	27.15	30.12
$\mathcal{T}_3$ (20)	22.84	25.52	24.34	26.67
		Five-shot		
$\mathcal{T}_1$ (10)	50.03	55.04	53.83	56.92
$\mathcal{T}_2$ (10)	36.76	44.57	42.68	47.04
$\mathcal{T}_3$ (20)	37.37	40.46	39.36	42.87

by a Wilcoxn rank sum test for equal medians [28] at 5% significance (indicated by '✓' in Table Table IV.3). Adding both `Depth-` *and* `Pose`-augmented features to the original one-shot features achieves the greatest improvement in recognition accuracy, ranging from 4-6 percentage points. This indicates that information from depth and pose is complementary and allows for better coverage of the feature space. Notably, we also experimented with the metric-learning approach of Fink [25] which only led to negligible gains over the *Baseline* (e. g., 33.85% on $\mathcal{T}_1$).

Feature analysis/visualization. To assess the nature of feature synthesis, we backpropagate through RCNN layers the gradient w. r. t. the 2-norm between an original and a synthesized feature vector. The strength of the input gradient indicates *how much each pixel of the object must change* to produce a proportional change in depth/pose of the sample. As can be seen in the example of Fig. Figure IV.8, a *greater* desired change in depth invokes a *stronger* gradient on the monitor. *Second*, we ran a *retrieval experiment*: we sampled 1300 instances of 10 (unseen) object classes ($\mathcal{T}_1$) and synthesized features for each instance w. r.

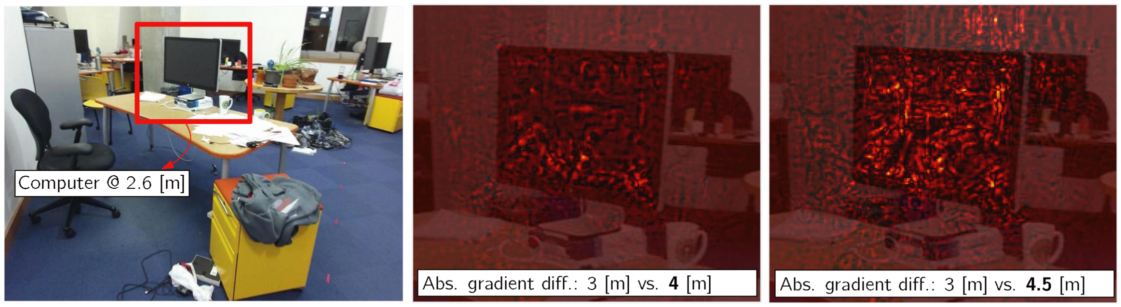

Computer @ 2.6 [m]
Abs. gradient diff.: 3 [m] vs. **4** [m]
Abs. gradient diff.: 3 [m] vs. **4.5** [m]

Figure IV.8: Visualizing gradients generated by the AGA network to affect depth change. Figure depicts the difference in *gradient magnitude* when backpropagating (through RCNN) the 2-norm of the difference between an original and a synthesized feature vector for an increasing desired change in depth, i. e., 3[m] v. s. 4[m] (*middle*) and 3[m] v. s. 4.5[m] (*right*).

t. depth. Synthesized features were then used for retrieval on the original 1300 features. This allows to assess if synthesized features (1) allow to retrieve instances of the *same class* (Top-1 acc.) and (2) of the desired attribute value. The latter is measured by the coefficient of determination (R^2). As seen in Table Table IV.5, the R^2 scores indicate that we can actually retrieve instances with the desired attribute values. Notably, even in cases where $R^2 \approx 0$ (i. e., the linear model does not explain the variability), the results still show decent Top-1 acc., revealing that synthesis *does not alter class membership*.

IV.E.4 Object-based one-shot scene recognition

We can also use AGA for a different type of transfer, namely the transfer from object detection networks to one-shot scene recognition. Although, object detection is a challenging task in itself, significant progress is made, every year, in competitions such as the ImageNet challenge. Extending the gains in object detection to other related problems, such as scene recognition, is therefore quite appealing. A system that uses an accurate object detector such as an RCNN [29] to perform scene recognition, could generate comprehensive annotations for an image in one forward pass. An object detector that supports one-shot scene recognition could do so with the least amount of additional data. It must be noted that such

Table IV.4: *One-shot classification* on indoor scenes from [65]. The classes include: {auditorium, bakery, bedroom, bookstore, children room, classroom, computer room, concert hall, corridor, dental office, dining room, hospital room, laboratory, library, living room, lobby, meeting room, movie theater, nursery, office, operating room, pantry, restaurant}. For `Sem-FV` [19], we use ImageNet CNN features extracted at one image scale.

Method	**Accuracy** [%]
`max. pool` (*Baseline*)	13.97
`AGA FV` (+**D**)	**15.13**
`AGA FV` (+**P**)	**14.63**
`AGA CL-1` (+**D**, max.)	**16.04**
`AGA CL-2` (+**P**, max.)	**15.52**
`AGA CL-3` (+**D**, +**P**, max.)	**16.32**
`Sem-FV` [19]	32.75
`AGA Sem-FV`	**34.36**
`Places` [96]	51.28
`AGA Places`	**52.11**

systems are different from object recognition based methods such as [31, 19, 12], where explicit detection of objects is not necessary. They apply filters from object recognition CNNs to several regions of images and extract features from all of them, whether or not an object is found. The data available to them is therefore enough to learn complex descriptors such as Fisher vectors (FVs). A detector, on the other hand, may produce very few features from an image, based on the number of objects found. AGA is tailor-made for such scenarios where features from an RCNN-detected object can be augmented.

Setup. To evaluate AGA in this setting, we select a 25-class subset of MIT Indoor [65], which may contain objects that the RCNN is trained for. The reason for this choice is our reliance on a detection CNN, which has a vocabulary of 19 objects from SUN RGB-D. At present, this is the largest such dataset that provides objects and their 3D attributes. The system can be extended easily to accommodate more scene classes if a larger RGB-D object dataset becomes available. As the RCNN produces very few detections per scene image, the best approach, without

Table IV.5: *Retrieval results* for unseen objects ($\mathcal{T}_1$) when querying with synthesized features of varying depth. Larger R^2 values indicate a stronger linear relationship ($R^2 \in [0, 1]$) to the depth values of retrieved instances.

Object	Top-1	R^2	**Object**	Top-1	R^2
`picture`	0.33	0.36	`whiteboard`	0.12	0.30
`fridge`	0.26	0.08	`counter`	0.64	0.18
`books`	0.52	0.07	`stove`	0.20	0.13
`cabinet`	0.57	0.27	`printer`	0.31	0.02
`computer`	0.94	0.26	`ottoman`	0.60	0.12

augmentation, is to perform pooling of RCNN features from proposals into a fixed-size representation. We used max-pooling as our *baseline*. Upon augmentation, using predicted depth/ pose, an image has enough RCNN features to compute a GMM-based FV. For this, we use the experimental settings in [19]. The FVs are denoted as `AGA FV(+D)` and `AGA FV(+P)`, based on the attribute used to guide the augmentation. As classifier, we use a linear C-SVM with fixed parameter (C).

Results. Table Table IV.4 lists the avgerage one-shot recognition accuracy over multiple iterations. The benefits of AGA are clear, as both aug. FVs perform better than the max-pooling baseline by 0.5-1% points. Training on a combination (concatenated vector) of the augmented FVs and max-pooling, denoted as `AGA CL-1`, `AGA CL-2` and `AGA CL-3` further improves by about 1-2% points. Finally, we combined our aug. FVs with the state-of-the-art semantic FV of [19] and Places CNN features [96] for one-shot classification. Both combinations, denoted `AGA Sem-FV` and `AGA Places`, improved by a non-trivial margin ($\sim$1% points).

IV.F Analyzing AGA

In the previous section, we demonstrated the use of our AGA method to improve one-shot learning in case of object recognition. Here, we provide a detailed analysis of the parameter choices made during system training and their effect on the performance. Specifically, we highlight the interplay between the terms of the loss used in (IV.3) and the impact of trading one off for the other on the quality of synthetic data. Additionally, we explicitly demonstrate the selectivity of the synthesized data points by using them directly for classification. Finally, we provide ways to improve the AGA baseline by exploiting multiple levels of information from the object CNN.

Impact of regularization

The training objective for synthesis functions ϕ, shown in (IV.3), consists of two opposing losses: (1) the *regularization loss* penalizes displacement of the synthetic point $\hat{\mathbf{x}}$ from the original point $\mathbf{x}$. The *attribute mismatch term*, on the other hand, encourages $\hat{\mathbf{x}}$ to move so that the attribute changes from the current $t_0 = \gamma(\mathbf{x})$ to t. The two, therefore, act as *adversaries* during the optimization steps. The regularization parameter α controls the trade-off between the two losses and its value affects the *selectivity* of the synthesized data points. To study this effect, as well as to determine the optimal value for α, we perform more experiments in the object recognition framework described in Sec. IV.E.3. Specifically, for the dataset $\mathcal{T}_1$, we use our augmentation strategy to produce synthetic data points with different values of α in (IV.3). For every *alpha*, the resulting augmented dataset is used for one-shot object recognition. To get a better understanding of the quality of augmented data, in this experiment, we use a nearest neighbor (NN), as well as a linear SVM (as in Sec. IV.E.3).

Fig. Figure IV.9 shows the variation in results achieved with pose, depth, and pose+depth based augmentation for different values of parameter α. It is clear that for the SVM classifier, increasing α improves the performance of AGA based object recognition over the one-shot baseline. For 1-NN classification, less regularization tends to work better. Overall, the SVM classifier obtains the best accuracy when synthetic features are combined. Also, the results indicate that large-margin classification clearly prefers points closer to the original samples. If a “chair” image is used to produce a synthetic point that is nearby in $\mathcal{X}$, the

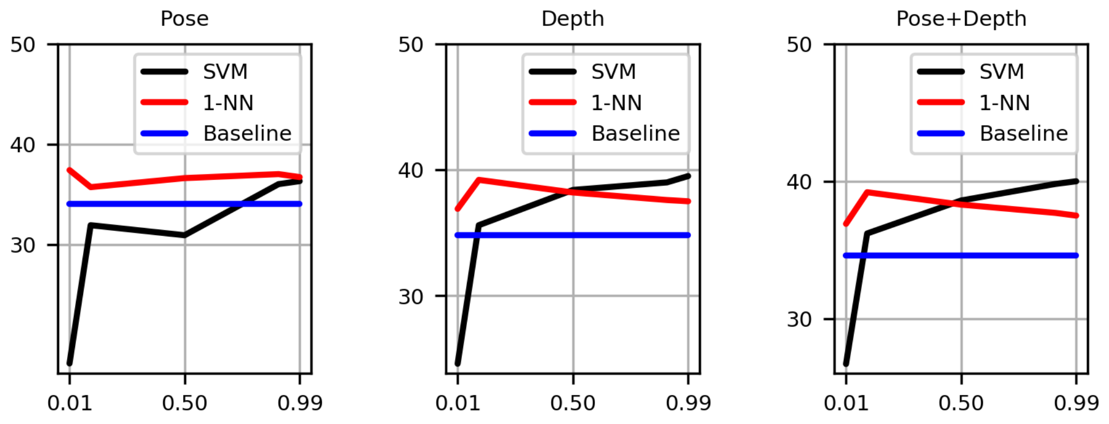


Figure IV.9: One-shot regularization experiment where the weight α is varied from 0.01 to 0.99. Lower α prefers minimizing attribute mismatch but generates a (possible) less informative data-point. Higher α maintains the discriminative power of the synthetic data. The *Baseline* represents a linear SVM trained on the 1-shot instances only.

generated example is likely to also be a "chair". For a very low α, the discriminative power of the synthetic examples seems to be very low. As a result the final SVM classifier fails to generalize, resulting in poor performance. In a one-shot scenario it is impossible for any classifier to correctly estimate class distributions (or boundaries). Augmentation, therefore, can be seen as equivalent to introducing *pseudo-examples* to regularize them. Using a lower weight α in (IV.3), results in such examples being largely *non-informative*. Another key observation is the disparity between the performance of the NN and the SVM classifiers. An SVM is learned discriminatively and is known to be more robust in limited data scenarios. A NN classifier, which relies on a generative model, performs well when sufficient amount of data is available. The results of our experiments, however, indicate that the NN is more robust than an SVM for a range of different α's. This is perhaps because of the tendency of the latter to *overfit*. Especially at lower α's, the SVM is seen overfitting to the poor collection of synthetic data points generated by our algorithm and performing much worse than the parameter-less NN classifier.

Quality of synthetic features

Next, we assess the discriminative power of features synthesized by our algorithm more explicitly. In Sec. IV.E.3, we tried to do so by evaluating the synthesized data for retrieval of real image features from $\mathcal{T}_1$. The results show that AGA generated data correlates semantically with the examples retrieved. Instead of proxy-retrieval, here, we perform a proxy recognition experiment to evaluate the quality of our data synthesis. Using the original one-shot training image from each class of $\mathcal{T}_1$, we first synthesize additional data points with depth and pose attribute guidance. We remove the original one-shot example and use, simply, *only* the synthesized data to train a classifier for object recognition. The same procedure is repeated for 5-shot recognition.

The results of proxy based recognition are shown in Fig Figure IV.10. Many conclusions are possible from these. First, it is clear that the performance of the proxy classifier is not much worse than the classifier trained with original as well as synthesized data. It is also, in some cases, better than the one-shot or few-shot baseline. This finding is very significant in itself. The implication is that AGA-based synthetic examples carry enough "clean" information from the original data-point and the latter can, in some cases, be even omitted without a huge loss. It is, thus, apparent that the synthetic data does not simply serve as a non-informative prior that regularizes the one-shot or low-shot SVM. Another interesting observation from Fig Figure IV.10 is that the proxy classifier performs better when the weight α in the AGA objective is higher, consistent with the experiments of Sec. IV.F. Since, a higher α incentivizes the synthetic data to be close to the original in an L_2 sense, the former seems to *distill* semantic information from the latter. The improved disciminative power of synthetic examples for a higher α also vindicates our claim that the L_2 loss term in (IV.3) preserves object identity.

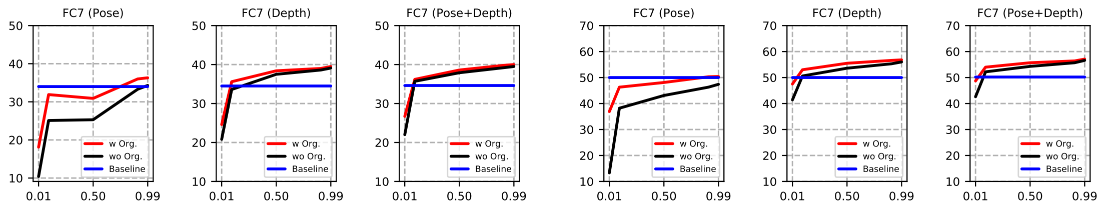

(a) 1-shot (b) 5-shot

Figure IV.10: One-shot and five-shot recognition with only augmented data (original omitted). Recognition performance reported (on y-axis in [%]) *with* (w Org.) and *without* (wo Org.) original features (`FC7`) for a linear SVM classifier, depending on the regularization weight α (x-axis). The *Baseline* result denotes the recognition performance of a linear SVM trained on the original features only (i.e., 1 example/class in case of 1-shot and 5 examples/class in case of 5-shot.

Few-shot performance

To understand the limits of our algorithm, we steadily increased the number of training points available to the (linear-SVM) few-shot classifier (from 1 to 10). In each case, using our method, we generated synthetic points (`FC7`) from each example in the training set. The result is shown in Fig. Figure IV.11 (gray and red bars). We find that the improvements due to augmentation sustain until we increase the training examples over 10 per class. It is also worth noting that AGA does not quite match having real image data. In particular, for each real data point, our augmentation method generates ≈ 18 synthetic points. The performance achieved using one real example and 18 AGA generated ones is much less than using 10 real examples. However, AGA clearly helps to achieve meaningful improvements when real data is not available.

Optimizing the feature space $\mathcal{X}$

The main motivation behind the proposed guided augmentation strategy is the unavailability of adequate training data in few-shot *transfer learning* scenarios. For transfer problems, in general, it is also essential to avail of a semantic feature representation for data that acts as a conduit for the said transfer. E.g. the activations of top layers in a pre-trained object recognition CNN, such as [45], are shown to be highly selective of meaningful visual concepts. In many recent works, therefore, top layers of a CNN have been used for image representation for various vision tasks with remarkable success [21, 31, 19, 18]. In the proposed framework also, we use the activations of the penultimate layer of the RCNN (before the classifier), denoted `FC7`, for data representation as well as generation. One may argue that the higher layers of a network are too *invariant* for pose/depth modeling. Our main objective, however, is to generate discriminative data points in a feature space that is conducive to knowledge transfer. Therefore, the use of higher CNN layer activations is imperative for good performance. Initial layers of a network may preserve detailed visual variations, however, they lack the semantic selectivity necessary to help transfer of high level information such as detected objects in an image. In fact, the experimental analysis with synthesized `FC7` features presented in section IV.E, shows that despite its supposed invariance, the feature space still allows AGA to sensibly capture attribute variations. First, the accuracy of attribute prediction is reasonable as shown in table Table IV.1. Second, according to the retrieval experiment in IV.E.3, synthesized `FC7` feature points are able to recall database examples that match it in object class as well as attribute value (pose/depth) accurately.

Although we use `FC7` as the representation space $\mathcal{X}$ in our few-shot experiments, it is certainly not the only choice as far as semantic activations of a CNN are concerned. In some works, `conv5` outputs [12], `fc6` outputs [55] and even fc8 features [19] have been used for transfer. All these layers appear in the final stage of a CNN and their units are demonstrably semantic in nature. In order to improve our AGA system we experimented with combinations of these features. In particular, we found the addition of `FC6` activations to `FC7` helped improve the quality of the synthetic data as well as the few-shot performance substantially. We achieved the combination in two different ways. The first approach is referred to as *late fusion*, where an AGA system is trained for each representation inde-

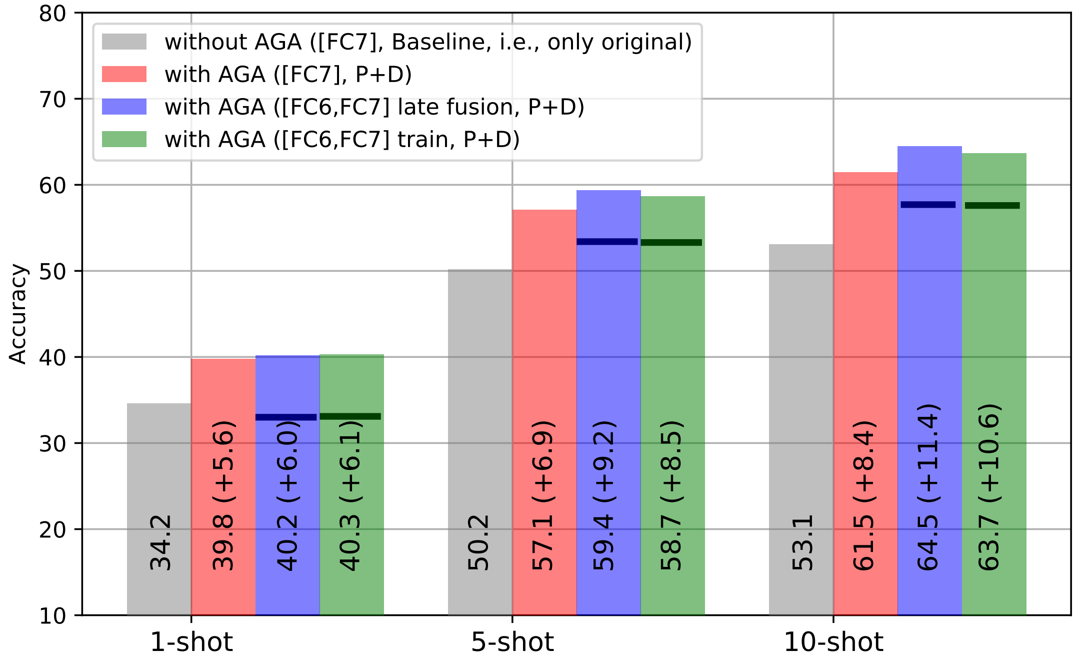


Figure IV.11: Performance of few shot (1-10) object recognition on $\mathcal{T}_1$ using AGA. Shown are the results of a linear SVM classifier trained (a) on `FC7` features of the original examples; (b) on `FC7` features for pose & depth; (c) on a concatenation of `FC6` and `FC7` features for depth & pose (each AGA synthesis function trained separately; (d) on `FC6` and `FC7` features for depth & pose when the AGA synthesis function is retrained on the concatenation of `FC6` and `FC7`. The bold horizontal lines show the baseline(s) for (c) and (d).

pendently. At the one-shot/few-shot transfer learning stage, then, the `FC7` and `FC6` descriptors synthesized from an image for the same desired attribute value t are concatenated. The extended feature representation is used instead of just `FC7` for few-shot SVM training and testing. The second approach of feature fusion is referred to as *paired training*. This entails relearning the AGA system, which involves re-training the attribute predictor γ as well as feature synthesizers ϕ, on the joint $[\texttt{FC7}, \texttt{FC6}]$ space. Augmentation as well as one-shot/few-shot learning is performed on this larger space. The performance is shown in fig Figure IV.11.

Results indicate very little to no improvement due to both feature fusion methods over `FC7` for one shot AGA. Perhaps a large feature space is not sufficient to learn a generalizable SVM with only 19 examples (one real and 17 synthetic). As we steadily increase the number of real training images, the feature fusion methods start gaining over the use of just `FC7`. The improvement of 5-shot AGA with feature fusion is about $1.5-2.0\%$ over `FC7` AGA classifier. In the 10 shot case the accuracy increases by $2-3\%$. The most important observation is that the late fusion method performs convincingly better than `FC7` AGA as well as paired training on both spaces. This implies that individual AGA networks could be learned on different feature spaces $\mathcal{X}$ perhaps even originating from different RCNNs and their synthesized points could be concatenated to produce a better representation during final transfer learning. This allows the system to more scalable than with the paired training alternative, which requires complete retraining in order to extend $\mathcal{X}$.

IV.G Discussion

We presented an approach toward attribute-guided augmentation in feature space. Experiments show that object attributes, such as pose / depth, are beneficial in the context of one-shot recognition, i. e., an extreme case of limited training data. Notably, even in case of mediocre performance of the attribute regressor (e. g., on pose), results indicate that synthesized features can still supply useful information to the classification process. While we do use bounding boxes to extract object crops from SUN RGB-D in our object-recognition experiments, this is only done to clearly tease out the effect of augmentation. In principle, as our encoder-decoder is trained in an *object-agnostic* manner, no external knowledge about classes is required.

As SUN RGB-D exhibits high variability in the range of both attributes, augmentation along these dimensions can indeed help classifier training. However, when variability is limited, e. g., under controlled acquisition settings, the gains may be less apparent. In that case, augmentation with respect to other object attributes might be required.

Two aspects are specifically interesting for future work. *First*, replacing the attribute regressor for pose with a specifically tailored component will potentially improve learning of the synthesis function(s) ϕ_i^k and lead to more realistic synthetic samples. *Second*, we conjecture that, as additional data with more annotated object classes and attributes becomes available (e. g., [7]), the encoder-decoder can leverage more diverse samples and thus model feature changes with respect to the attribute values more accurately.

IV.H Acknowledgements

The text of Chapter IV, is based on material as it appears in: M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, "AGA: Attribute-Guided Augmentation", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, 2017 and M. Dixit, R. Kwitt, M. Neithammer and N. Vasconcelos, "Data augmentation with attribute trajectory transfer", [To be submitted to], IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI). The dissertation author is a primary researcher and author of the cited material.

Chapter V

Conclusions

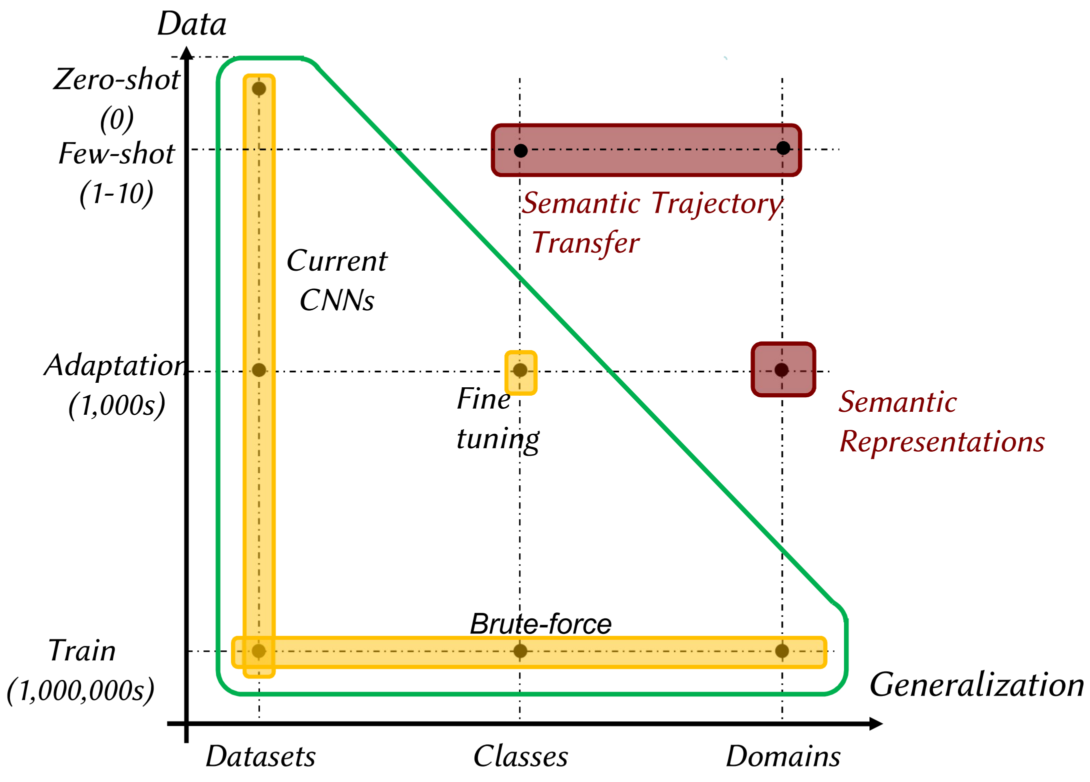


Figure V.1: Map of CNN based transfer learning: across dataset, class and domains. Our contributions are represented as "Semantic Representations" and "Semantic Trajectory Transfer". The former denotes contributions in Chapter II and Chapter III, while latter denotes contributions made in Chapter IV.

It is well known that deep convolutional neural networks trained on large scale vision datasets achieve spectacular results. A more important virtue of these networks, perhaps, is their ability to capture information that is useful and transferable to other vision tasks. The entire spectrum of neural network based transfer learning methods is represented by the figure Figure V.1. Consider, for example, a CNN trained on object recognition. To use this network on the same set of object classes, one can either deploy it in a *zero-shot* manner (in the inference mode) or adapt it to a dataset of 1000s of images if the task is slight different (e.g. object detection or localization). To use the same recognition network, instead, on a set of different, previously unseen, object categories, one can resort to *finetuning* the recognition CNN to a moderate size dataset of new object classes. This method of transfer is often employed in object detection literature [30, 29, 71]. In this work, we propose solutions to two alternate scenarios of transfer learning where the knowledge transfer must either be achieved using very few new data points (few-shot transfer) or across to a completely different domain (from objects to scenes).

As an example of transfer across visual domains, we choose the example of object to scene transfer. We use off-the-shelf object recognition CNNs trained on large scale datasets to generate a Bag-of-semantics (BoS) representation of scene images. Under a BoS, a scene is described as a collection of object probability multinomials obtained from its local regions. The probabilities are referred to as semantics because of their inherent meaning (dog-ness or car-ness of a patch). We propose to embed a scene BoS into an invariant scene representation known as a semantic Fisher vector. The design of a Fisher vector (FV) embedding is not very straight-forward in the space of multinomial probability vectors, due to its non-Euclidean nature. We solve this problem by transforming the multinomials into their natural parameter form, thereby projecting them into a Euclidean space without any loss of their object selectivity. The semantic Fisher vectors derived from the natural parameter space of object CNNs represents a conduit for object-to-scene knowledge transfer. This representation combined with a simple linear classifier, is shown to achieve state-of-the-art scene classification on well known benchmarks. Next we, present a technique to improve the performance of semantic transfer by perfecting the design of the classical Fisher vector embedding [64] itself. These are generally derived as scores or gradients of a Gaussian mixture model that uses a diagonal covariance matrix. While, this was never a problem with low-dimensional feature spaces before, it may not be sufficient for high dimensional CNN features. To cover the manifold of CNN features efficiently we propose the use of mixture of factor analyzers (MFA) [27, 87], a model that locally approximates the distribution with full-covariance Gaussians. We derive the Fisher vector embedding under this model and show that it captures richer descriptor statistics compared to a variance Gaussian. The MFA based Fisher vector improves the performance of object based semantic scene classification as expected. Despite being a transfer learning method with relatively modest data requirements (50 images per class), we show that the MFA FV is comparable to even a scene classification CNN trained from scratch on millions of new labeled images. When combined, the two techniques also result in a surprising $6 - 8\%$ improvement in accuracy. The proposed object-based scene representations are denoted as semantic representations on the transfer learning spectrum in fig. Figure V.1 and generally applicable to a case when the categories in the target domain (e.g. scenes) are loose combinations of those in source domain (e.g. objects).

We next, consider a situation when transfer learning must be achieved with a very small target dataset with not more than 10 examples per class. We refer to this as few-shot transfer. Despite the high quality of representations that can be generated using an off-the-shelf neural network such as the ImageNet CNN [45], such acute scarcity of new data prevents learning a sufficiently generalize-able classifier for the new task. To solve the problem, we propose the idea of *attribue guided* data augmentation. Standard data augmentation involves making flipped, rotated or cropped copies of a training image in order to simulate a sufficient training dataset. These copies however are very trivial and do not add any new information. We propose to generate non-trivial data for few-shot or even one-shot transfer learning with the help of attribute trajectory learning and transfer. Using a small auxiliary dataset labeled with objects and their attributes (properties) such as 3D pose and depth, we learn trajectories of variation in these attributes on the feature space of a pre-trained CNN. For an image of a new, previously unseen object (during transfer), its feature representation is obtained from the network and regressed along the learned pose and depth trajectories to generate additional features. This technique of attribute guided data augmentation helps us generate synthetic examples from original data, which helps improve the performance of one-shot and few-shot object and scene recognition. The process of attribute guided augmentation is alternatively denoted in fig Figure V.1 as semantic trajectory transfer, since the generation of data requires transfer of trajectories of learned variations to the new example.

Bibliography

[1] 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

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, Apr 2002.

[3] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[4] A. Bergamo and L. Torresani, "Meta-class features for large-scale object categorization on a budget," in *Computer Vision and Pattern Recognition (CVPR)*, 2012. [Online]. Available: \url{http://vlg.cs.dartmouth.edu/metaclass}

[5] ——, "Classemes and other classifier-based features for efficient object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, p. 1, 2014.

[6] A. Bergamo, L. Torresani, and A. Fitzgibbon, "Picodes: Learning a compact code for novel-category recognition," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K. Weinberger, Eds., 2011, pp. 2088–2096.

[7] A. Borji, S. Izadi, and L. Itti, "iLab-20M: A large-scale controlled object dataset to investigate deep learning," in *CVPR*, 2016.

[8] Y.-L. Boureau, F. Bach, Y. LeCun, and J. Ponce, "Learning mid-level features for recognition," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, jun. 2010, pp. 2559 –2566.

[9] C. W. C.E. Rasmussen, *Gaussian Processes for Machine Learning.* The MIT Press, 2005.

[10] C. Charalambous and A. Bharath, "A data augmentation methodology for training machine/deep learning gait recognition algorithms," in *BMVC*, 2016.

[11] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.

[12] M. Cimpoi, S. Maji, and A. Vedaldi, "Deep filter banks for texture recognition and segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[13] D.-A. Clevert, T. Unterhiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (ELUs)," in *ICLR*, 2016.

[14] 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.

[15] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893.

[16] J. Delhumeau, P. H. Gosselin, H. Jégou, and P. Pérez, "Revisiting the vlad image representation," in *ACM Multimedia*, 2013, pp. 653–656.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, June 2009, pp. 248–255.

[18] M. Dixit and N. Vasoncelos, "Object based scene representations using Fisher scores of local subspace projections," in *NIPS*, 2016.

[19] M. Dixit, S. Chen, D. Gao, N. Rasiwasia, and N. Vasconcelos, "Scene classification with semantic fisher vectors," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[20] C. Doersch, A. Gupta, and A. A. Efros, "Mid-level visual element discovery as discriminative mode seeking," in *Neural Information Processing Systems (NIPS)*, 2013.

[21] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *International Conference in Machine Learning (ICML)*, 2014.

[22] H. Drucker, C. Burges, L. Kaufman, and A. Smola, "Support vector regression machines," in *NIPS*, 1997.

[23] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C. J. Lin, "LIBLINEAR: A library for large linear classification," *JMLR*, vol. 9, no. 8, pp. 1871–1874, 2008.

[24] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1627–1645, 2010.

[25] M. Fink, "Object classification from a single example utilizing relevance metrics," in *NIPS*, 2004.

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

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

[28] J. Gibbons and S. Chakraborti, *Nonparametric Statistical Inference*, 5th ed. Chapman & Hall/CRC Press, 2011.

[29] R. Girshick, "Fast r-cnn," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[30] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

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

[32] ——, "Multi-scale orderless pooling of deep convolutional activation features," in *Computer Vision ECCV 2014*, ser. Lecture Notes in Computer Science, D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., vol. 8695. Springer International Publishing, 2014, pp. 392–407. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10584-0_26

[33] G. Griffin, A. Holub, and P. Perona, "The caltech-256," caltech technical report, Tech. Rep., 2006.

[34] S. Hauberg, O. Freifeld, A. Boensen, L. Larsen, J. F. III, and L. Hansen, "Dreaming more data: Class-dependent distributions over diffeomorphisms for learned data augmentation," in *AISTATS*, 2016.

[35] K. He, X. Zhang, S.Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *ICML*, 2015.

[37] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," *arxiv*, 2016.

[38] T. S. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Proceedings of the 1998 conference on Advances in neural information processing systems II*, 1999, pp. 487–493.

[39] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *NIPS*, 2015.

[40] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *IEEE Conference on Computer Vision & Pattern Recognition*, jun 2010, pp. 3304–3311. [Online]. Available: http://lear.inrialpes.fr/pubs/2010/JDSP10

[41] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[42] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[43] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR*, 2015.

[44] T. Kobayashi, "Dirichlet-based histogram feature transform for image classification," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[45] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105.

[46] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, ser. CVPR '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 1785–1792. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2011.5995702

[47] R. Kwitt, S. Hegenbart, and M. Niethammer, "One-shot learning of scene locations via feature trajectory transfer," in *CVPR*, 2016.

[48] R. Kwitt, N. Vasconcelos, and N. Rasiwasia, "Scene recognition on the semantic manifold," in *Proceedings of the 12th European conference on Computer Vision - Volume Part IV*, ser. ECCV'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 359–372.

[49] C. H. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 3, pp. 453–465, 2014.

[50] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, vol. 2, 2006, pp. 2169 – 2178.

[51] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, November 1998.

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

[53] Y. Li, L. Liu, C. Shen, and A. van den Hengel, "Mid-level deep pattern mining," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

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

[55] 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.

[56] 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.

[57] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," 2003.

[58] E. Miller, N. Matsakis, and P. Viola, "Learning from one-example through shared density transforms," in *CVPR*, 2000.

[59] T. P. Minka, "Estimating a dirichlet distribution," Tech. Rep., 2000.

[60] V. Nair and G. Hinton, "Rectified linear units improve restricted boltzmann machines," in *ICML*, 2010.

[61] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," 2015.

[62] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning deep object detectors from 3d models," in *ICCV*, 2015.

[63] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, 2007, pp. 1–8.

[64] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proceedings of the 11th European conference on Computer vision: Part IV*, ser. ECCV'10, 2010, pp. 143–156.

[65] A. Quattoni and A. Torralba, "Recognizing indoor scenes," *2012 IEEE Conference on Computer Vision and Pattern Recognition*, vol. 0, pp. 413–420, 2009.

[66] 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.

[67] 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.

[68] N. Rasiwasia and N. Vasconcelos, "Holistic context models for visual recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 5, pp. 902–917, May 2012.

[69] N. Rasiwasia, J. Costa Pereira, E. Coviello, G. Doyle, G. R. Lanckriet, R. Levy, and N. Vasconcelos, "A new approach to cross-modal multimedia retrieval," in *Proceedings of the 18th ACM International Conference on Multimedia*, ser. MM '10. New York, NY, USA: ACM, 2010, pp. 251–260. [Online]. Available: http://doi.acm.org/10.1145/1873951.1873987

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

[71] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Neural Information Processing Systems (NIPS)*, 2015.

[72] G. Rogez and C. Schmid, "MoCap-guided data augmentation for 3D pose estimation in the wild," *CoRR*, vol. abs/1607.02046, 2016. [Online]. Available: http://arxiv.org/abs/1607.02046

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

[74] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *CoRR*, vol. abs/1312.6229, 2013. [Online]. Available: http://arxiv.org/abs/1312.6229

[75] A. Sharif Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, “Cnn features off-the-shelf: An astounding baseline for recognition,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2014.

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

[77] S. Song, S. Lichtenberg, and J. Xiao, “SUN RGB-D: A RGB-D scene understanding benchmark suite,” in *CVPR*, 2015.

[78] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: A simple way to prevent neural networks from overfitting,” *JMLR*, vol. 15, p. 19291958, 2014.

[79] H. Su, C. Qi, Y. Li, and L. Guibas, “Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3d model views,” in *ICCV*, 2015.

[80] Y. Su and F. Jurie, “Improving image classification using semantic attributes,” *International Journal of Computer Vision*, vol. 100, no. 1, pp. 59–77, 2012.

[81] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Computer Vision and Pattern Recognition (CVPR)*, 2015. [Online]. Available: http://arxiv.org/abs/1409.4842

[82] 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.

[83] A. Torralba and A. A. Efros, “Unbiased look at dataset bias,” in *CVPR’11*, June 2011.

[84] L. Torresani, M. Szummer, and A. Fitzgibbon, “Efficient object category recognition using classemes,” in *European Conference on Computer Vision (ECCV)*, Sep. 2010, pp. 776–789. [Online]. Available: \url{http://research.microsoft.com/pubs/136846/TorresaniSzummerFitzgibbon-classemes-eccv10.pdf}

[85] J. Uijlings, K. van de Sande, T. Gevers, and A. Smeulders, “Selective search for object recognition,” *IJCV*, vol. 104, no. 2, pp. 154–171, 2013.

[86] N. Vasconcelos and A. Lippman, “A probabilistic architecture for content-based image retrieval,” in *Proc. Computer vision and pattern recognition*, 2000, pp. 216–221.

[87] J. Verbeek, "Learning nonlinear image manifolds by global alignment of local linear models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 8, pp. 1236–1250, Aug. 2006.

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

[89] R. Wu, B. Wang, W. Wang, and Y. Yu, "Harvesting discriminative meta objects with deep cnn features for scene classification," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[90] J. Xiao, J. Hays, K. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 3485–3492.

[91] J. Yang, K. Yu, Y. Gong, and T. Huang, "Linear spatial pyramid matching using sparse coding for image classification," in *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2009.

[92] S. Yang and D. Ramanan, "Multi-scale recognition with dag-cnns," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[93] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I*, 2014, pp. 818–833. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-10590-1_53

[94] M. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[95] B. Zhou, A. Khosla, L. A., A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization." *CVPR*, 2016.

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