DEEP LEARNING COMPACT AND INVARIANT IMAGE REPRESENTATIONS FOR INSTANCE RETRIEVAL

A DISSERTATION
SUBMITTED TO THE COMPUTER SCIENCE
LABORATORY OF THE PIERRE AND MARIE CURIE
UNIVERSITY
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF
DOCTOR IN PHILOSOPHY

Olivier Morère

April, 2016

Version 1.5

List of Publications

The material reported in this thesis is the subject of the following publications.

Conference Papers

O. Morère, J. Lin, A. Veillard, V. Chandrasekhar, T. Poggio. Nested Invariance Pooling and RBM Hashing for Image Instance Retrieval. *Submitted to European Conference on Computer Vision (ECCV) 2016.*

O. Morère, J. Lin, V. Chandrasekhar, A. Veillard, H. Goh. Co-Sparsity Regularized Deep Hashing for Image Instance Retrieval. *Accepted in International Conference on Image Processing (ICIP) 2016.* [108]

O. Morère, J. Lin, J. Petta, V. Chandrasekhar and A. Veillard Tiny descriptors for image retrieval with unsupervised triplet hashing. *Data Compression Conference (DCC) 2016.* [109]

O. Morère, H. Goh, A. Veillard, V. Chandrasekhar and J. Lin. Co-Regularized deep representations for video summarization. *International Conference on Image Processing (ICIP) 2015.* [106]

C. Dao-Duc, H. Xiaohui and O. Morère. Maritime Vessel Images Classification Using Deep Convolutional Neural Networks. *Symposium on Information and Communication Technology (SOICT) 2015.* [26]

V. Chandrasekhar, J. Lin, O. Morère, A. Veillard and H. Goh. Compact Global Descriptors for Visual Search. *Data Compression Conference (DCC) 2015.* [13]

Technical Reports

O. Morère, A. Veillard, J. Lin, J. Petta, V. Chandrasekhar and T. Poggio. Group Invariant Deep Representations for Image Instance Retrieval. *Center for Brains, Minds*

and Machines (CBMM) 2016. [111]

O. Morère, J. Lin, V. Chandrasekhar, A. Veillard and H. Goh. Deephash: Getting Regularization, Depth and Fine-tuning Right. *arXiv preprint arXiv:1501.04711 2015.* [107]

Contests and Workshops

O. Morère, A. Veillard, H. Goh. Team "LateFusion". *Kaggle National Data Science Bowl Challenge 2015.* [110]

O. Morère, H. Goh, A. Veillard, V. Chandrasekhar. Large Scale Image Classification on a Shoe String. *ImageNet Large Scale Visual Recognition Challenge, European Conference on Computer Vision (ECCV) 2014.* [105]

Journal Articles

O. Morère, V. Chandrasekhar, J. Lin, H. Goh and A. Veillard. A Practical Guide to CNNs and Fisher Vectors for Image Instance Retrieval. *Accepted in Signal Processing (SIGPRO) 2016.* [104]

Contents

List of Tables

List of Figures

Chapter 1

Executive Summary

Image instance retrieval is the problem of finding an object instance present in a query image from a database of images. Also referred to as *particular object retrieval*, this problem typically entails determining with high precision whether the retrieved image contains the same object as the query image. Scale, rotation and orientation changes between query and database objects and background clutter pose significant challenges for this problem.

State-of-the-art image instance retrieval pipelines consist of two major steps: first, a subset of images similar to the query are retrieved from the database, and second, Geometric Consistency Checks (GCC) are applied to select the relevant images from the subset with high precision. The first step is based on comparison of *global image descriptors*: high-dimensional vectors with up to tens of thousands of dimensions representing the image data. The second step is computationally highly complex and can only be applied to hundreds or thousands of images in practical applications. More discriminative global descriptors result in relevant images being more highly ranked, resulting in fewer images that need to be compared pairwise with GCC. As a result, better global descriptors are key to improving retrieval performance and have been the object of much recent interest. Furthermore, fast searches in large databases of millions or even billions of images requires the global descriptors to be compressed into compact representations. This thesis will focus on how to achieve extremely compact global descriptor representations for large-scale image instance retrieval.

After introducing background concepts about supervised neural networks, Restricted Boltzmann Machine (RBM) and deep learning in Chapter 2, Chapter 3 will present the design principles and recent work for the Convolutional Neural Networks (CNN), which recently became the method of choice for large-scale image classification tasks. Next, an original multistage approach for the fusion of the output of multiple CNN is proposed. Submitted as part of the ILSVRC 2014 challenge, results show that this approach can significantly improve classification results. The promising performance of CNN is largely due to their capability to learn appropriate high-level visual

representations from the data. Inspired by a stream of recent works showing that the representations learnt on one particular classification task can transfer well to other classification tasks, subsequent chapters will focus on the transferability of representations learnt by CNN to image instance retrieval.

In Chapter 4, a systematic and in-depth evaluation of Fisher Vectors (FV) and CNN pipelines for image instance retrieval is performed. Here we propose a comprehensive set of practical guidelines believed to be useful to anyone seeking to implement state-of-the-art descriptors for image instance retrieval. Some of the recommendations are general good practices while others are more problem specific. CNN descriptors are shown to offer the best retrieval performance on average, but unlike with image classification, the supremacy of CNN over FV is not always clear in the case of image instance retrieval, so mixing both approaches is sometimes optimal. The evaluation study leads to two issues related to CNN descriptors which need to be addressed: first, the lack of transformation (specifically, scale and rotation) invariance of the descriptors, and second, the high dimensionality and scalar nature of the descriptors making descriptor matching inefficient. These issues are addressed in subsequent chapters.

Chapter 5 tackles the problem of hashing the descriptors to small binary codes for efficient matching with Hamming distances, while retaining the good retrieval performance of the uncompressed descriptors. The very low bitrate range of 32-1024 bits is specifically targeted. The proposed hashing pipeline consists of two parts: (a) An unsupervised dimensionality reduction approach using RBM to produce binary hashes at the target bitrate, and (b) A fine-tuning step to improve the binary embedding functions generated by stacked RBM for which we propose both supervised and semi-supervised variants. The first dimensionality reduction step applies a regularization to RBM specifically designed to optimize the distribution of generated binary hash codes. The proposed approach is a batch-level regularization scheme aiming to improve very low bitrate hashes by encouraging efficient use of the latent subspace both within and across the hashes. The second fine-tuning step is based on metric refinement with Siamese networks. The method is based on the use of a labeled training set of matching and non-matching pairs of instances, and critical improvements in the loss function of the Siamese network leading to improvements in retrieval results are shown. While able to significantly improve retrieval results, Siamese fine-tuning has the drawback of requiring an external labeled dataset of matching and non-matching pairs. Therefore, we subsequently propose Unsupervised Triplet Hashing (UTH), a *fully unsupervised* rank learning scheme based on three weight sharing networks. The scheme is based on preserving the good retrieval performance of the uncompressed descriptors and thus does not require any external training labels.

Finally, Chapter 6 proposes simple database-side pooling schemes that can be effective at mitigating issues related to the aforementioned image transformations, practically overcoming the lack of robustness of CNN descriptors compared with FV. Among

the various schemes proposed, some allow the pre-computation of single descriptors while others require more operations at query time. Chapter 6's main contribution is Nested Invariance Pooling (NIP), a method to produce compact global image descriptors from visual representations extracted from CNN, which are robust to multiple types of image transformations. NIP is inspired from *i-theory*, a recently proposed mathematical theory for computing group invariant transformations with feed-forward neural networks. NIP is shown to be able to produce compact (but non-binary) global image descriptors which are robust to rotations, scale changes and translations and are able to outperform other schemes at equivalent descriptor dimensionality on most evaluation datasets. Finally, NIP is shown to be able to effectively combine with the RBM hashing scheme proposed in Chapter 5, leading to hashes that are both compact and robust to multiple types of image transformations. This thorough empirical evaluation with small and large-scale datasets shows that the proposed scheme is able to produce extremely compact hashes that are able to outperform other schemes, especially at very low bitrates (32-256 bits).

Chapter 2

Image Instance Retrieval

2.1 Image Instance Retrieval

Image instance retrieval is the task of finding an object instance present in a query image from a database of images (Figure 2.1). Also, referred to as particular object retrieval, this problem typically entails determining with high precision whether the retrieved image contains the same object as the query image. Scale, rotation, orientation and lighting changes between query and database objects, and background clutter pose significant challenges for this problem.

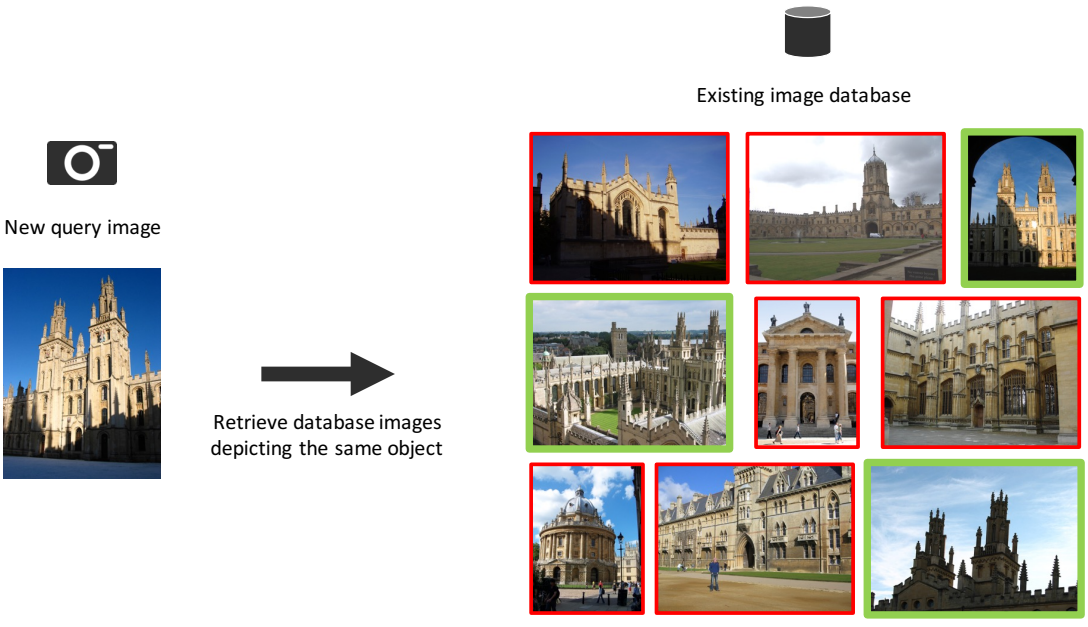


Figure 2.1: In the content-based image instance retrieval problem, the task is to select database images depicting the same object instance as the one depicted in the query image. No external information is used (categories, labels...).

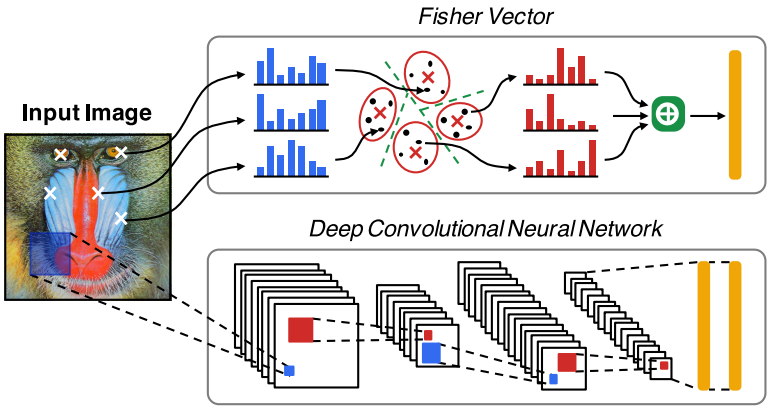


Figure 2.2: Fisher Vector (FV) and Convolutional Neural Network (CNN) based pipelines for the extraction of *global descriptors* from images.

State-of-the-art image instance retrieval pipelines consist of two major steps: first, a subset of images similar to the query are retrieved from the database, and second, Geometric Consistency Checks (GCC) [34, 123, 6] are applied to select the relevant images from the subset with high precision. The first step is based on comparison of *global image descriptors*: high-dimensional vectors with up to tens of thousands of dimensions representing the image data. The second step is computationally highly complex and can only be applied to hundreds or thousands of images in practical applications. As a result, better global descriptors are key to improving retrieval performance and have been the object of much recent interest, with work on specific applications such as digital documents [31], mobile visual search [31, 20], distributed large scale search [76] and compact descriptors for fast real-world applications [32, 78]. More discriminative global descriptors result in relevant images being ranked higher, leading to fewer images needing to be compared pairwise with GCC. Furthermore, fast search in large databases of millions or even billions of images requires the global descriptors to be compressed into compact representations. In this chapter, we review the state-of-the-art in global descriptors and hashing for the image instance retrieval problem.

2.1.1 Global Descriptors

A typical image instance retrieval pipeline starts with the comparison of high-dimensional vectors referred to as *global descriptors*. *Global descriptors* are often aggregated from local descriptors such as SIFT [99, 100] and HoG [25]. State-of-the-art *global descriptors* for image instance retrieval are based on either FV [120, 122]/VLAD [73, 74] or Convolutional Neural Networks (CNN) [9] (Figure 2.2). State-of-the-art global descriptors are reviewed in this section (Table 2.1).

FV & VLAD	Fisher Vector [120], Improved Fisher Vector [122]
	Vector of Locally Aggregated Descriptors [73, 74]
	Residual Enhanced Visual Vector [19]
	PCA-Whitening [70]
	Scalable Compressed Fisher Vector [91]
	Aggregated Fisher Vector [133]
	Selective Match Kernel [147]
	Democratic Aggregation [75]
	Intra-normalization [5]
CNN	Raw CNN descriptors [127]
	Spatial-search aggregation [127]
	Fine-tuned CNN [9]
	Multi-Scale Orderless Pooling (MOP-CNN) [48]
	Conv max-pooling [7, 136]
	Pooling from intermediate layers (SPoC) [8]
	Regional Maximum Activation of Convolutions (R-MAC) [148]

Table 2.1: FV and CNN based methods for *global descriptors* extraction.

Fisher Vectors. Hand-crafted global image descriptors such as Fisher Vectors (FV) [122] and Vector of Locally Aggregated Descriptors (VLAD) [74] allow for robust image matching. The FV is obtained by quantizing the set of local feature descriptors with a small codebook of 64-512 centroids, and aggregating first and second order residual statistics for features quantized to each centroid. The residual statistics from each centroid are concatenated together to obtain the high-dimensional global descriptor representation, typically 8192 to 65536 dimensions. The performance increases as the dimensionality of the global descriptor increases, as shown in [122].
The VLAD descriptor can be considered a special case of the FV, with hard-quantization of feature descriptors, with concatenation of only first-order residual statistics in the final descriptor representation.

There has been extensive work on the FV since it was first proposed for instance retrieval [120]. FV are aggregated on descriptors extracted densely in the image [133], or around interest points like Difference-of-Gaussian (DoG) interest points [98]. In the former case, dense sampling is performed at a single scale or multiple scales. The dense sampling approach is popular for image classification tasks, while the latter is used in image retrieval as the DoG interest points provide invariance to scale and rotation. State-of-the-art results using FV are based on aggregating statistics around interest points like Difference-of-Gaussian [98] or Hessian-affine interest points [103].

Several improvements have been proposed over the baseline VLAD and FV approaches. In [18], another normalization scheme is proposed for the residuals, where the per-cluster mean of residuals is computed instead of the sum, enhancing the dis-

criminativeness of the VLAD descriptor. In [74], a similar Signed Square Rooting (SSR) normalization scheme is proposed for VLAD. In [70], VLAD is extended by using PCA whitening and multiple clusterings for quantization. Intra-normalization scheme is proposed in [5] to alleviate the adverse effect of bursty visual features [72], where the sum of residuals is L2 normalized within each VLAD block.

FV is improved over the baseline approach of [122] by using non-linear additive kernel and normalization. Other improvements to the baseline FV [122] include the Residual Enhanced Visual Vector [19], the Scalable Compressed Fisher Vector [91], better matching kernels [147], and better aggregation schemes [75]. Some of the best reported instance retrieval results are still based on hand-crafted FV [147]. Next, the matter of how CNN have been applied for the particular instance retrieval problem is discussed.

CNN Descriptors. As opposed to the carefully hand-crafted FV, deep learning has achieved remarkable performance for large scale image classification [79, 137]. CNN are now considered to be the mainstream approach for large-scale image classification. Most ImageNet submissions from 2014 onwards are based on CNN. Deep learning has achieved remarkable success in many other visual tasks such as face recognition [145, 142], pedestrian detection [119] and pose estimation [150]. After the winning submission of Krizhevsky et al. in the ImageNet 2012 challenge [79], CNN began to be applied to the instance retrieval problem as well.

While deep learning has unquestionably become the dominant approach for image classification, raw image descriptors from CNN do not systematically have the upper hand over FV in image instance retrieval. The two types of descriptors being radically different in nature, one can expect them to behave very differently based on specific aspects of the problem. Unlike interest points which provide scale and rotation invariance to the FV pipeline, CNN representations used in image-classification are obtained by densely sampling a resized canonical image. CNN do not have a built-in mechanism to ensure resilience to geometric transformations like scale and rotation, and therefore, do not provide explicit rotation and scale invariance, which are often key to instance retrieval tasks. Moderate levels of scale and rotation invariance for CNN features is nevertheless indirectly achieved from the max-pooling operations in the pipeline, the diversity of the training data which typically contains objects at varying scales and orientations, and data augmentation during the training phase where data can be preprocessed and input to the CNN at different scales and orientations.

There has been a fair bit of work on CNN descriptors for image retrieval in recent literature [127, 9, 48, 136, 8, 148]. Razavian et al. [127] evaluate representations extracted from CNN fully-connected layer on a wide range of tasks, including as a global descriptor for instance retrieval, and show promising initial results. Then, Babenko et al. [9] show that a pre-trained CNN can be fine-tuned with domain specific data

Supervised	Kernel-based Supervised Hashing (KSH) [80] Minimal Loss Hashing (MLH) [118] Ranking-based Supervised Hashing (RSH) [154] Column Generation Hashing (CGH) [89]
Unsupervised	Locality Sensitive Hashing (LSH) [27] Restricted Boltzmann Machines (RBM) [63, 113] Spectral Hashing (SH) [156] Iterative Quantization (ITQ) [46]

Table 2.2: Methods for hashing *global descriptors*.

(objects, scenes, etc.) to improve instance retrieval performance on relevant datasets. In [8], Babenko et al. show how pooled intermediate layers of a CNN can be used as a starting representation for instance retrieval. They show that sum-pooling of intermediate feature maps performs better than max-pooling, when the image representation is whitened. Note that the approach in [8] provides limited invariance to translation, but not to scale or rotation. In MOP-CNN [48], the authors propose extracting CNN activations using a sliding window approach at different scales in the image, followed by computing a high-dimensional VLAD representation on the local CNN descriptors. While this results in highly performant descriptors, the starting representations are often orders of magnitude larger than original descriptors. [7, 136] show that spatial max pooling of intermediate maps is an effective representation and higher performance can be achieved compared to using the fully-connected layers. Another very recent work [148] proposes pooling across regional bounding boxes in the image, similar to the popular R-CNN approach [41] used for object detection.

In summary, a typical image instance retrieval pipeline starts with the computation of high-dimensional vectors referred to as global descriptors. However, the dimensionality of such descriptors is typically very high: 8192 to 65536 floating point numbers for FV[122] and 4096 for CNN [79]. As a result, the global descriptor is typically compressed or hashed to compact representations, typically hundreds of bits, for fast retrieval, which is reviewed next.

2.1.2 Hashing

Extremely compact image representations such as 64-bit hashes are a definite must for fast image instance retrieval because (1) 64 bits provide more than enough capacity for any practical purposes, including internet-scale problems and (2) a 64-bit hash is directly addressable in RAM and enables fast matching using Hamming distances. Bringing such high-dimensional representations down to a 64-bit hash is a considerable challenge: the main focus of this thesis.

While there is plenty of work on learning binary codes [53] for compressing small

descriptors like SIFT, there is relatively little work on compression of high-dimensional global descriptors. Proposed methods for compression of descriptors like SIFT or GIST include [46, 153, 59, 53, 156, 80, 96, 118, 149, 14, 15]. The global descriptor data in consideration in this work are two orders of magnitude higher in dimensionality, making the problem significantly more challenging.

Descriptor compression techniques can be roughly grouped into two categories: hashing and quantization. Hashing compresses raw descriptors into short binary vectors with either data-independent methods like Locality Sensitive Hashing (LSH) [27] or data-dependent methods like Iterative Quantization (ITQ) [46] and Bilinear Projection Binary Codes (BPBC) [45]. For instance, the popular ITQ first performs Principal Component Analysis (PCA) to reduce dimensionality, then applies rotations to distribute variance across dimensions, and finally binarizes each dimension according to its sign. Quantization based methods such as Product Quantization (PQ) [74] divide the raw descriptor into smaller blocks and vector quantization is performed on each block. While this results in highly compact descriptors composed of sub-quantizer indices, the resulting representation is not binary and cannot be compared with ultra-fast Hamming distance computations.

Hashing schemes can be further categorized into unsupervised and supervised (including semi-supervised) schemes. Examples of unsupervised hashing methods are Iterative Quantization (ITQ) [46], Spectral Hashing (SH) [156], Locality Sensitive Hashing (LSH) [27], Restricted Boltzmann Machines (RBM) [63, 113], while some examples of state-of-the-art supervised schemes include Minimal Loss Hashing [118], Kernel-based Supervised Hashing [80], Ranking-based Supervised Hashing [154] and Column Generation Hashing [89].

Next, the most important hashing techniques applied to the global descriptor compression problem are reviewed. Perronnin et al. [122] propose ternary quantization of FV, quantizing each dimension to +1,-1 or 0. The authors show that this representation results in little loss in performance. However, this results in descriptor size of thousands of bits. Perronnin et al. also explore Locality Sensitive Hashing [27] and Spectral Hashing [156]. Spectral Hashing performs poorly at high rates, while LSH and simple ternary quantization need thousands of bits to achieve good performance. Gong et al. propose the popular Iterative Quantization (ITQ) scheme and apply it to the GIST descriptor in [46]. ITQ first performs Principal Component Analysis (PCA) to reduce dimensionality, and subsequently learns a rotation to minimize the quantization error of mapping the transformed data to the vertices of a zero-centered binary hypercube. One drawback of this scheme is that the PCA matrix might require several GBs of memory for high dimensional global descriptors. In subsequent work, Gong et al. in [45] show how bilinear projections can be used to create binary hashes of VLAD [46]. Gong et al. [45] focus on generating very long codes for global descriptors, and the Bilinear Projection-based Binary Codes (BPBC) scheme requires tens of thousands of

bits to match the performance of the uncompressed global descriptor. The MPEG-CDVS standard adopted the Scalable Compressed Fisher Vector [91], which was based on binarization of high-dimensional Fisher Vectors. The size of the compressed descriptor in the MPEG-CDVS standard ranges from 256 bytes to several thousand bytes per image, based on the operating point. Bit selection is performed greedily to maximize pairwise Receiver Operating Characteristic (ROC) matching performance. Stacked restricted Boltzmann machines (RBM), primarily known as powerful dimensionality reduction techniques [132], can also be used for hashing. Next, state-of-the-art deep learning techniques are reviewed.

2.2 Deep Learning

In this section a brief introduction is given of several notions and notations related to neural networks and deep learning which are necessary to the understanding of the work presented in this thesis. Section 2.2.1 introduces neural networks and supervised learning with the Multi-Layer Perceptron (MLP). Then, Restricted Boltzmann Machines (RBM), an unsupervised learning scheme used across this thesis, is presented in Section 2.2.2. Finally, a discussion about depth in neural nets and the methods used to achieve it is given in Section 2.2.3.

This section is intended as a brief introduction of notions and notations related to neural networks and deep learning which are subsequently reused throughout the thesis. For a comprehensive guide on the state-of-the-art deep learning algorithms, the reader may refer to the following book from Goodfellow et al. [49].

2.2.1 Multi-Layer Perceptron

A *neural network* is a multi-input, multi-output function $h : \mathbb{R}^n \mapsto \mathbb{R}^m$ where n is the number of input values and m the number of output values. It can be pictured as a directed graph of connected units. Each *neuron*, also called *unit*, may receive multiple inputs and sends a single output.

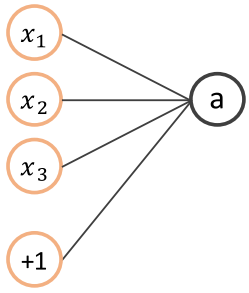


Figure 2.3: A single unit neural network.

Figure 2.3 illustrates the simplest neural network made of a single unit (hence

$m = 1$). The inputs x_i and the output a are related by a function of the following type:

$$a = f(\sum_i W_i x_i + b)$$

where $W_i \in \mathbb{R}$ defining a linear combination of the inputs are the *weight terms*, $b \in \mathbb{R}$ is the *bias term* and $f : \mathbb{R} \mapsto \mathbb{R}$ is the *activation function*. The output value of a unit is often referred to as the *activation* of the unit. Using matrix notations, it becomes:

$$a = f(W'x + b)$$

with $W \in \mathbb{R}^n$.

A common activation function is the *standard logistic* function:

$$f(z) = \frac{1}{1 + \exp(-z)}$$

A perceptron [128] is a binary classifier where classes can be separated based on a threshold on the activation function. Perceptrons can express linear functions only. More complex models can be achieved by stacking multiple layers of units such that the activations of units from one layer become the inputs of the units of the next layer. Such architectures are called Multi-Layers Perceptrons (MLP).

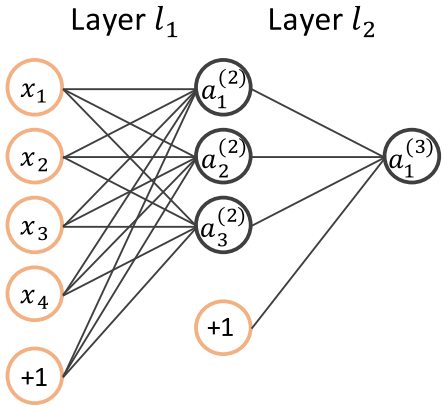


Figure 2.4: A multi-layer neural network with four input units, three hidden units and a single output unit.

Figure 2.4 depicts a network with two layers of units.

We choose to denote by a superscript the layer a variable belongs to. Therefore, the activations of the second layer in Figure 2.4 are given by:

$$a_i^{(2)} = f(\sum_j W_{ij}^{(1)} x_j + b^{(1)})$$

or by, following the previously introduced matrix notation:

$$a_i^{(2)} = f({W_i^{(1)}}' x + b^{(1)})$$

By introducing a vector notation $f(z) = (f(z_i))_i$ where $z = (z_i)_i$, we get:

$$a^{(2)} = f({W^{(1)}}' x + b^{(1)})$$

and thus:

$$\begin{aligned} a^{(3)} &= f({W^{(2)}}' a^{(2)} + b^{(2)}) \\ &= f({W^{(2)}}' f({W^{(1)}}' x + b^{(1)}) + b^{(2)}) \end{aligned}$$

The weight matrices $W^{(l)}$ and the bias terms $b^{(l)}$ can be adjusted by back-propagation of a loss term. The back-propagation algorithm [129] is a supervised learning procedure for networks of units. It repeatedly adjusts the weights of the network connections in order to minimize the difference between the actual network output and the ground truth. For each iteration of the back-propagation algorithm, all the activations are first computed from the bottom (lower intex) to the top (higher index). The top layer is then used to compute the error to be back-propagated from top to bottom. For each unit, the attached error is both an indication of it's responsibility in the overall error and of how much it is going to be affected by the update. Networks are often trained using gradient descent. The cost function for a single training example (x, y) can be defined as follows:

$$J(\mathbf{W}, \mathbf{b}; x, y) = \frac{1}{2} \| h_{\mathbf{W},\mathbf{b}}(x) - y \|^2$$

where $h_{W,b}(x)$ is the output of the MLP parametrized by all weight matrices $W^{(l)}$ and the bias terms $b^{(l)}$ collectively referred to as $\mathbf{W}$ and $\mathbf{b}$.

Given a training set of m training examples, the overall cost function can be defined as follows:

$$J(\mathbf{W}, \mathbf{b}) = \left[\frac{1}{m} \sum_{i=1}^{m} J(\mathbf{W}, \mathbf{b}; x^{(i)}, y^{(i)}) \right] + \frac{\lambda}{2} \sum_l \sum_i \sum_j \left(W_{ji}^{(l)} \right)^2$$

This definition includes the weight decay λ, a regularization parameter used to decrease the importance of the weights and therefore to help prevent overfitting. Gradient descent is used to minimize $J(\mathbf{W}, \mathbf{b})$. The individual parameters are updated as follows:

$$\Delta W_{ij}^{(l)} = -\alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(\mathbf{W}, \mathbf{b})$$

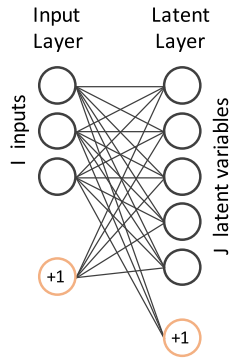


Figure 2.5: A Restricted Boltzmann Machine (RBM).

$$\Delta b_i^{(l)} = -\alpha \frac{\partial}{\partial b_i^{(l)}} J(\mathbf{W}, \mathbf{b})$$

where $\alpha > 0$ is the learning rate. Although $J(\mathbf{W}, \mathbf{b})$ is a non-convex function, gradient descent usually yields acceptable results as local minima are usually not a practical issue for large l [28, 22].

The MLP presented in this section makes use of fully-connected layers modelled by matrix products. For the task of image recognition in particular, restrictions of the fully-connected model making use of strong hypothesis on the signal are preferred, as discussed in Chapter 3.

2.2.2 Restricted Boltzmann Machine

The Restricted Boltzmann machine (RBM) [138] is a variant of the Boltzmann machine [64]. Unlike the MLP, the RBM is a form of unsupervised learning algorithm. RBM is used for a variety of applications including dimensionality reduction [63], classification [83], collaborative filtering [132] and hashing [131].

An RBM is an undirected bipartite graphical model consisting of a layer of visible units x and a layer of hidden or hidden units z, as shown in Figure 2.5. A set of symmetric weights W connects x and z. For an RBM with binary visible and hidden units, the joint set of visible and hidden units has an energy function given by:

$$E(x, z) = -\sum_i c_i x_i - \sum_j b_j z_j - \sum_{i,j} x_i z_j w_{ij} \tag{2.1}$$

where x_i and z_j are the binary states of visible and hidden units i and j respectively, w_{ij} are the weights connecting the units, and c_i and b_j are their respective bias terms. Using the energy function in Equation (2.1), a probability can be assigned to x as

follows:

$$P(x) = \sum_{z} \frac{\exp(-E(x,z))}{Z}$$

where Z is a "partition" term, given by summing over all possible join sets of visible and hidden units:

$$Z = \sum_{x,z} \exp(-E(x,z))$$

The activation probabilities of units in one layer can be sampled by fixing the states of the other layer as follows:

$$P(z_j = 1|x) = f(b_j + \sum_{i} w_{ij} x_i) \tag{2.2}$$

Similarly, with symmetric weights:

$$P(x_i = 1|z) = f(c_i + \sum_{j} w_{ij} z_j) \tag{2.3}$$

where $f(\cdot)$ is the standard logistic function. RBMs can be trained by minimizing the contrastive divergence objective [61], which approximates the maximum likelihood of the input distribution. Alternating Gibbs sampling based on Equations (2.4) and (2.5) is used to obtain the network states to update the parameters w_{ij}, b_i, b_j through gradient descent.

In [63], Hinton and Salakhutdinov proposed to build deep networks by stacking multiple RBM, an architecture subsequently referred to as Stacked RBM (SRBM). By reusing the layer notation of the previous section, activation probabilities between the units $z^{(l-1)}$ of layer $l-1$ and $z^{(l)}$ or layer l become:

$$P(z_j^{(l)} = 1|z^{(l-1)}) = f(b_j^{(l)} + \sum_{i} w_{ij} z_i^{(l-1)}) \tag{2.4}$$

Similarly, with symmetric weights:

$$P(z_i^{(l-1)} = 1|z^{(l)}) = f(c_i^{(l)} + \sum_{j} w_{ij} z_j^{(l)}) \tag{2.5}$$

where $W^{(l)}$, $b^{(l)}$ and $c^{(l)}$ are the corresponding weight and bias terms. Training is done greedily by minimizing contrastive divergence at every successive layer.

Additional regularization of RBM is key to learning high quality representations. The regularization often targets low activity for the latent variables across the set of training instances [88, 113, 60]. In [43], activations are regularized both across training

instances and across representation units by enforcing power law distributions.

SRBM and regularisation are the object of in Chapter 5 where they are discussed in more details.

2.2.3 Deep Neural Networks

A deep neural network can be loosely defined as any neural network that has more than a single hidden layer. Depth is desirable to model complex functions which cannot be efficiently modelled by shallow architectures [10]. Nevertheless, training a deep network end-to-end can be challenging and gradient-based training of deep supervised neural networks had the reputation of providing poor results compared to shallower architectures [10] until recently.

The problem posed by non-convex optimization and local minima is in fact not a practical issue. Recent work [22] suggests that the probability of finding bad local minima decreases with network size, and that finding the global minimum (on the training set) is not useful in practice, and may even lead to overfitting.

The increased training time required for model convergence can also be significantly shortened by relying on the shared-memory parallelization capabilities of modern GPU. When using popular deep networks for image classification along with popular software [77] on modern hardware, parallelized inferences are about twenty times faster on GPU compared to CPU.

One of the most obvious issues is the large number of parameters and non-linearities in deep networks, making models prone to overfitting. The gradient diffusion phenomenon is another issue associated with training large networks: as the error gradient is propagated backwards through several layers, it becomes too diffuse, meaning the responsibility of different units in the classification error is distributed too widely and thinly across units of the same layer. The use of large scale data sets [130] and aggressive data augmentation [159] is one way to reduce overfitting. Another way is through methods such as dropout [65]. In dropout, units from fully-connected layers are stochastically selected and set to zero during training. Dropout is a way of breaking co-adaptations and can be seen as an approximation to geometric mean of predictions of an ensemble of models trained with bagging. By reducing the size of the fully-connected layers at training time, dropout also helps to reduce the gradient diffusion issue. While dropout increases generalization, it also increases training time [24, 140]. Dropconnect was also proposed as a generalization of dropout [152]. In the case of signal data, Convolutional Neural Networks (CNN) [84] are often preferred to fully-connected architectures. Convolutions are also a way to reduce the connectivity which is non-stochastic and makes use of assumptions on the data (more details on CNN in Chapter 3).

A proper weight initialization scheme is also crucial to training deep neural net-

works [143]. With deeper nets, poor initialization often yields negative consequences, because (a) small changes to the network parameters amplify and inappropriate weights initialization often prevents the network from learning, and (b) random initialization of deep networks often leads to poorer local minima [33]. Xavier and Bengio [42] proposed an initialization strategy that helps keeping the weight and gradient values within reasonable range by preserving the variance from one layer to another. This approach allows for training deeper nets from scratch and is later adapted to deep convolutional architectures in [58]. Recently, Google introduced batch normalization [69], which is somehow the general case of the previous initialization concept as it enforces the variance of activations within each layer to be unchanged across iterations. More aggressive learning rates can be used, and training requires about one order of magnitude fewer iterations. However, it requires large enough batch sizes in regards to signal variance or number of classes. Most recently, Microsoft introduced residual learning [57], a method for training hundreds of layers deep networks. Residual nets involve shortcut connections, combining a layer output with its input representation, and removing to the layer the hassle of learning identity. Inspired by dropout and residual learning, stochastic residual learning [67] method stochastically shortcuts or drops layers from the residual network.

Chapter 3

Large Scale Image Classification

3.1 Introduction

Image classification consists in the association of one or several categories with an image based on the analysis of its contents. It is generally considered a complex problem due to the large representation gap between the raw pixels and the high-level concepts. The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [130] is an example of the large-scale image classification benchmark which has been popular recently. The ImageNet dataset [30] is a large scale collection of over 14 million annotated natural images, organized within a hierarchy of over 21 thousand classes. As shown in Figure 3.1, it features both a wide variety of categories (plants, animals, vehicles, places...) and very fine grain sub-categories of the same concept, such as over 100 different breeds of dog. ILSVRC and the ImageNet datasets are discussed in further details in Section 3.3.2.

Figure 3.2 shows the results to the ILSVRC challenges from 2010 to 2015. As evidenced by the results over the years, deep neural networks became the mainstream approach to large-scale image classification. The first deep learning based submission referred to as *AlexNet* by Krizhevsky et al. [79] managed to halve the error rate from above 25% down to 15.3% in 2012. The impact was in fact such that every submission from 2013 onwards is based on deep learning. Results keep steadily improving every subsequent year with 6.67% error in 2014 [144] and 3.6% error in 2015 [57] which for the first time outperforms human accuracy on the task (5.1% error [130]). ImageNet is used throughout this thesis both as a good generic training dataset on natural images and as a good evaluation benchmark.

The deep learning neural networks behind these breakthroughs in visual recognition are known as Convolutional Neural Networks (CNN). Section 3.2 presents the design principles behind the CNN and reviews the evolutions over the years. In standard practice, no single network will usually achieve best performance with complex tasks like ILSVRC. Instead, most competitive submissions are some form of aggregation of

Figure 3.1: Sample of images from five classes of ImageNet-1K which illustrate the difficulties behind the task. The classes are very diverse but can sometimes be very close. For instance, it is hard for the human observer to distinguish breeds of dogs such as "Great Pyrenees" and "Samoyed". High variance is also observed within the classes. In addition, the dataset has inherent labeling ambiguity (e.g. the bottom left image labeled as "cauliflower" while there are also several "turtle" classes in the dataset). We can also see cases of instance overlap (e.g. bottom right image).

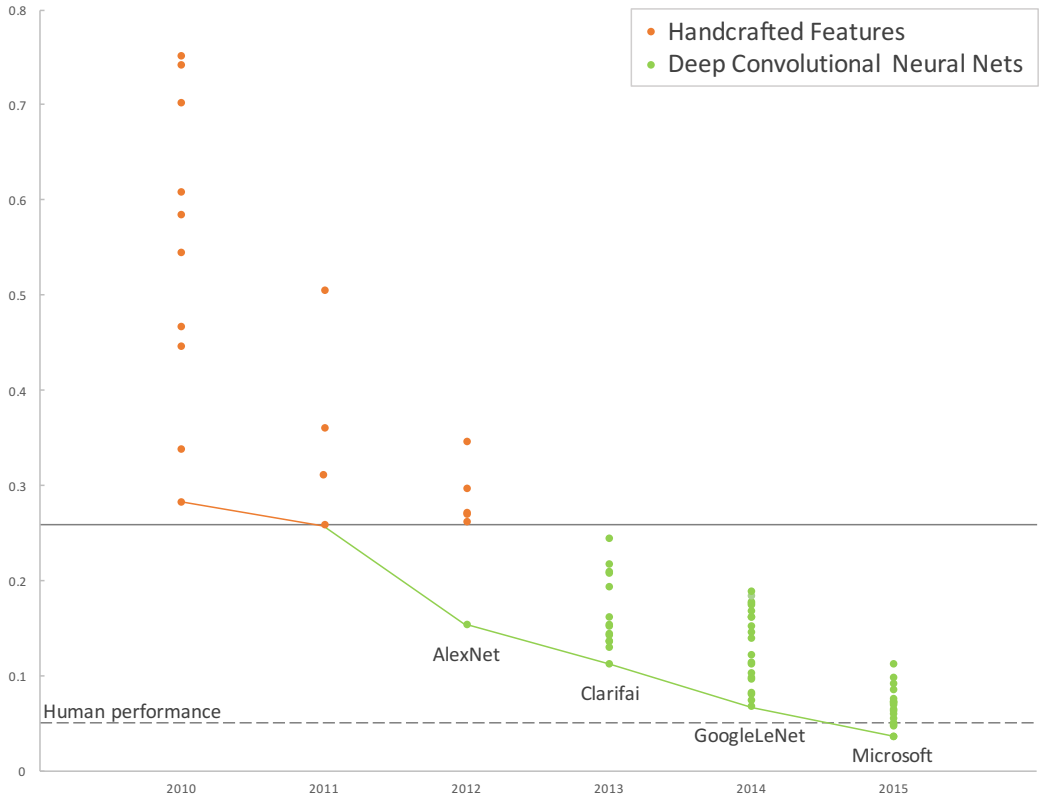


Figure 3.2: ILSVRC challenge results from 2010 to 2015 (top-5 accuracy). A massive shift from the use of handcrafted features to deep learning can be seen after the winning *AlexNet* submission in 2012 [79]. In 2015, the best submissions where able to outperform average human accuracy.

several models. The work presented in Section 3.3 is an original contribution of this thesis on multiple model aggregation. The work was submitted as part of the ILSVRC 2014 challenge.

3.2 Convolutional Neural Networks

Following *AlexNet* in 2012, all the submissions to ILSVRC are a type of CNN. Section 3.2.1 presents the general design principles behind the CNN. The positive performance of the CNN can be largely explained by their capability to learn high-level visual representations from the data as described in Section 3.2.2. Finally, the ongoing trends and more specific design features of CNN are discussed in Section 3.2.3.

3.2.1 Design Principles

CNN designates a particular type of MLP where the usual matrix multiplication as presented in Chapter 2 is replaced by mathematical convolutions. The idea was implemented by LeCun et al. in [84] as early as 1989 and applied to automatic ZIP code recognition for the U.S. Postal Service. Figure 3.3 depicts a CNN following the *LeNet* [86, 85] architecture also proposed by LeCun et al. It has several convolutional layers followed by some fully-connected layers (two and three in this case) which is a typical set up with many CNN. The fully-connected layers can be seen as a "standard" MLP-based classifier. The convolutional part of the CNN is a succession of convolutional operations and downsampling operations which are explained in the following paragraphs.

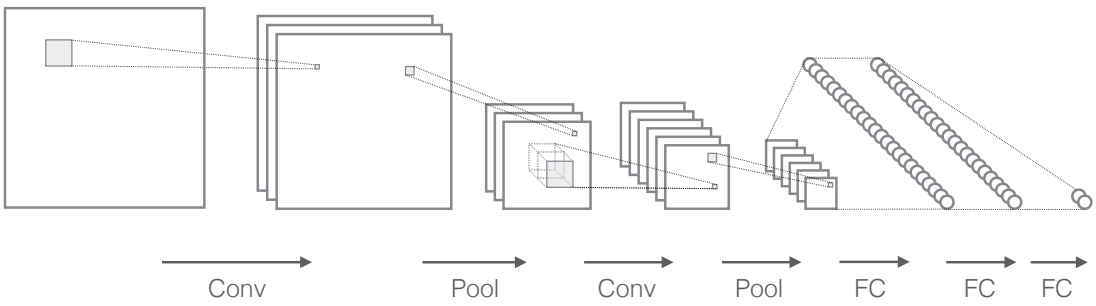


Figure 3.3: Example of *LeNet* CNN architecture with two convolutional layers and three fully-connected layers.

Convolutional Layers. The convolutional layer are strongly regularized versions of the fully-connected layers typical in MLP resulting from two assumptions:

- *Stationarity:* The data is assumed stationary in the two-dimensional space domain, a fairly standard assumption with natural images. This results in the

output layers being organized in several two-dimensional feature maps where all units from the same feature map share the same weights.

- *Local connectivity:* We assume every unit in a layer l is sparsely connected to the units in the previous layer $l-1$ situated in its local neighbourhood (referred to as the *receptive field*). Note that as l becomes greater, every unit is connected to a larger receptive field of the input pixel space.

Both assumptions combined result in the matrix multiplication defining fully connected layers to be replaced by a set of convolution operations. A given feature map h is determined by the weights W of the corresponding convolutional filter, corresponding bias b and unit activations $x = (x^m)$ of all the feature maps of the input layer such that:

$$h_{i,j} = \sigma\left((W * x)_{i,j} + b\right) \tag{3.1}$$

where σ is the activation function and $*$ is the two-dimensional convolution operation defined on *multiple* feature maps as:

$$(W * x)_{i,j} = (\sum_m W^m * x^m)_{i,j} \tag{3.2}$$

$$= \sum_m (W^m * x^m)_{i,j} \tag{3.3}$$

$$= \sum_m \sum_u \sum_v W^m_{u,v} x^m_{i-u,j-v} \tag{3.4}$$

An illustration of the convolutional operation with three input feature maps and a two-by-two convolution kernel is shown on Figure 3.4 (the bias is omitted).

Figure 3.5 shows the result of the convolution of a grayscale image with two different convolution kernels (two edge detectors for different orientations). The basic principle generalizes for higher layers beyond the pixel space to learn increasingly complex image representations as discussed in Section 3.2.2.

In practice, multiple feature maps are learnt by each layer in increasing number as shown on Figure 3.6. It also shows that the size of the input and output feature maps are tied and depend on the size of the convolution kernel. In practice, this is undesirable for flexible network design and the problem is circumvented by padding the input feature maps before applying the convolution.

The activation function originally proposed in *LeNet* is the hyperbolic tangent, a sigmoid function. A more popular function for CNN since *AlexNet* is the Rectified Linear Unit (ReLU) [114] defined as:

$$\text{ReLU}(t) = \begin{cases} t \text{ if } t > 0 \\ 0 \text{ otherwise} \end{cases} \tag{3.5}$$

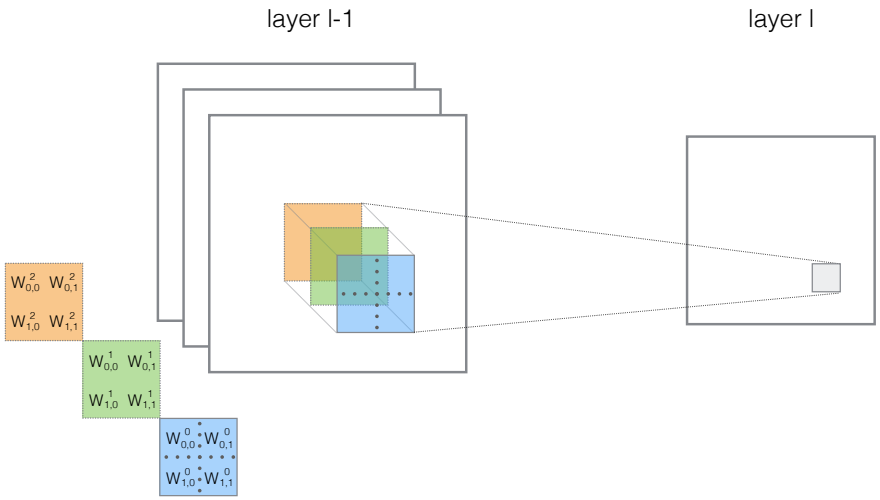
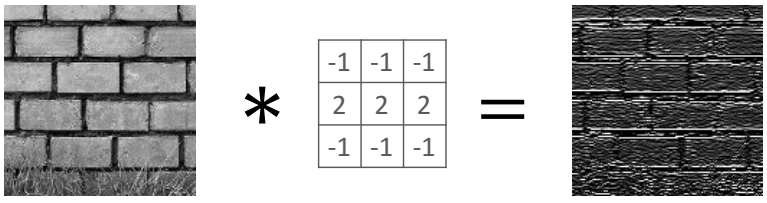
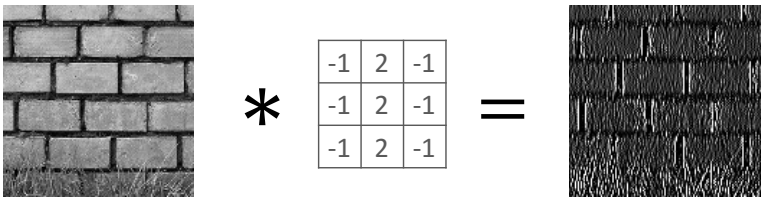


Figure 3.4: Computation of a feature map with two-by-two convolution kernels from 3 input feature maps.



(a) An horizontal edge detector kernel.



(b) A vertical edge detector kernel.

Figure 3.5: Convolution of a grayscale image with two different kernels.

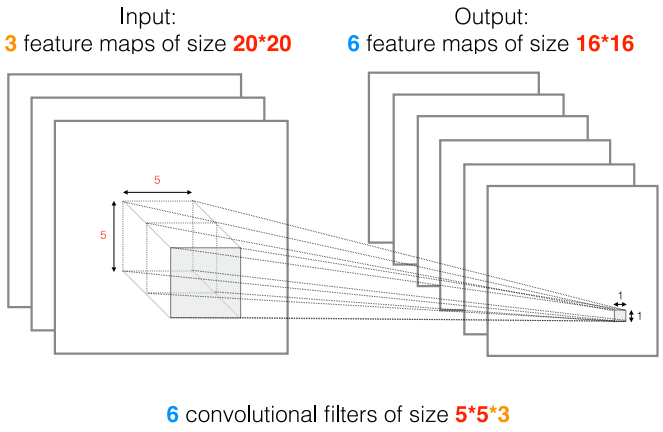


Figure 3.6: Illustration of a convolutional layer.

Unlike a sigmoid function, it has a constant gradient (when non-zero) which contributes to speed up training.

Pooling Layers. As the amount of feature maps is usually increased with layer depth, the spatial dimensionality must be reduced in order to keep layer-wise dimensionality under control. Down-sampling of the feature maps is usually done by "pooling", i.e. grouping values in feature maps by blocks and aggregating the values in each block. In addition to creating more compact representations, the pooling also introduces some level of invariance to distortions and translations to improve robustness to noise and clutter [11].

Originally, *average-pooling* which consists in taking the average value for each block was proposed in *LeNet*. Another popular down-sampling method is *max-pooling* which instead takes the maximum value. Figure 3.7 is the example of a two-by-two spatial max-pooling operation. Following empirical results on image classification shows that non-linear maximum pooling converges faster and improves generalization [134].

Note that aggressive *striding*, skipping units when computing convolutions, is

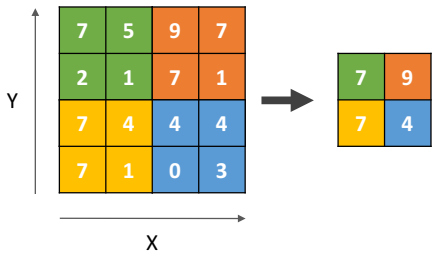


Figure 3.7: A two-by-two spatial max-pooling operation is applied on a single feature map. On the left, the input feature map spatiality is four-by-four. On the right, the resulting feature map spatiality is reduced to two-by-two.

also used as a spatial downsampling method, sometimes in conjunction with max-pooling [79]. Most recent CNN do not do both as discussed later.

3.2.2 Learning Visual Representations

The good image classification results obtained with CNN can for a large part be attributed to their capability to learn high-level visual representations compared with popular handcrafted descriptors such as SIFT and HoG. Deconvolutional networks [167, 167] can be used to visualize representations learnt by the convolution filters at the different depths by approximately mapping the activities of the features back into the image pixel space. Figure 3.8 shows deconvolutional network outputs for the five convolutional layers of an *AlexNet* model trained on ImageNet-1k. We can observe that the first layer models the low-level statistical distribution of the images reminiscent of Gabor wavelets. However, the representations become quickly more elaborate at deeper levels with units specifically activating to high-level concepts (e.g. "keyboard" to "cat").

3.2.3 Trends in CNN Design

As previously discussed, *LeNet* set many of the basic design principles of CNN, many of which are still valid today. The design of CNN was nevertheless refined over the years, as evidenced by the improved results at ILSVRC since *AlexNet* in 2012. Figure 3.9 shows a few popular CNN architectures proposed between 2012 and 2014. It is clear that many diverging choices are being made but several common trends can be identified.

This section is a non-exhaustive review of the recent work related to CNN for image classification. By putting into perspective the different converging and diverging design choices, we attempt to exhibit the different ongoing design trends. In particular, improvements to layer design, increased overall depth, spatial downsampling strategies, activation functions and various tweaks impacting training are highlighted and most of the techniques can be combined with good results.

Streamlined Layer Design. Contrary to *LeNet* or *AlexNet* which add several fully-connected layers (essentially an MLP classifier) on top of the feature maps, the recent trend moves increasingly away from fully-connected layers in favour of a mostly convolution-based architecture. Starting from 2014, a number of networks trained on ImageNet-1k use no more than a single 1000-units fully-connected layer, usually after average pooling of the highest-level feature maps, which directly feeds a softmax layer [92, 58, 57].

While *AlexNet* originally preferred larger convolutional kernels (up to eleven-by-eleven), more recent CNN favour smaller three-by-three convolutional kernels [137,

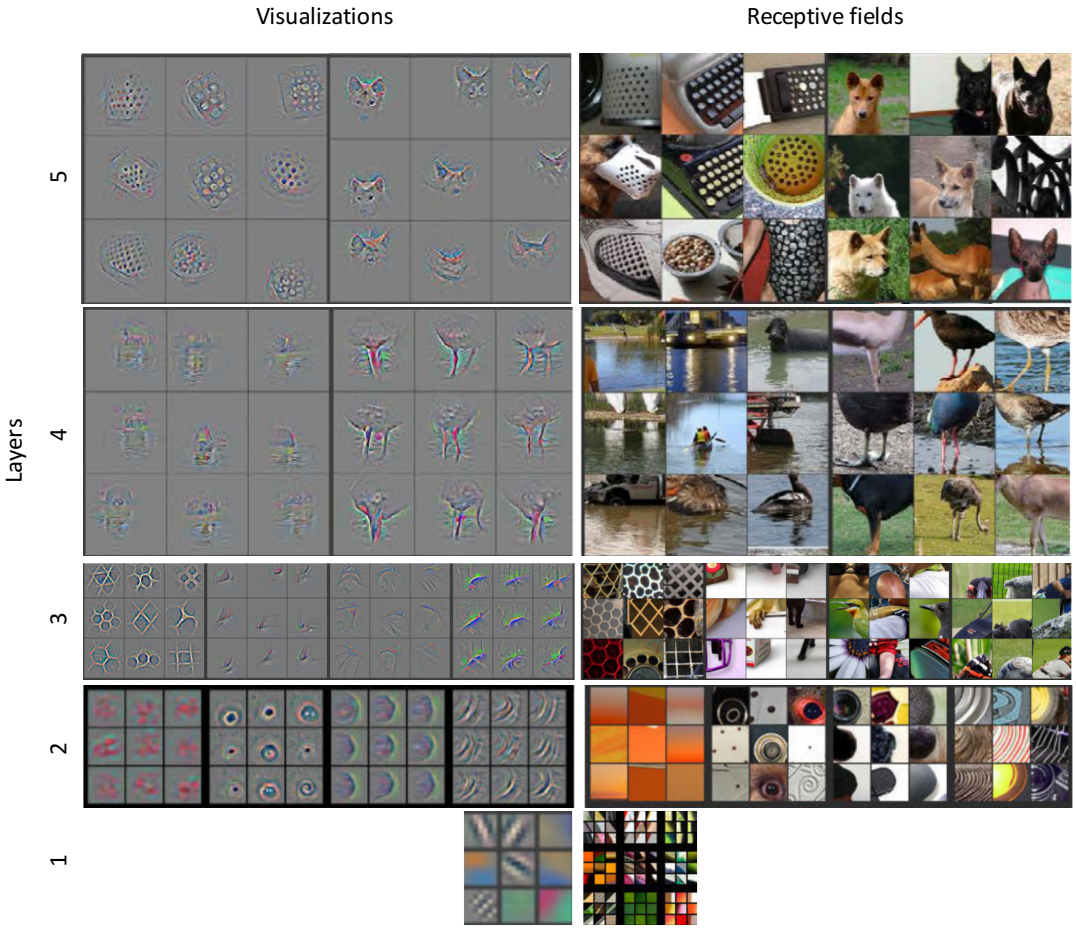


Figure 3.8: Convolutional filters visualized using a deconvolutional network. Filters are randomly sampled from the 5 convolutional layers of *AlexNet* trained on ImageNet-1k. The nine images which maximize the activation of each filter are shown. Note that the receptive field becomes wider when the layer is deeper. The first layer models the low-level features such as edges or gradients. Deeper units activate to more elaborate high-level concepts. Images are from [167].

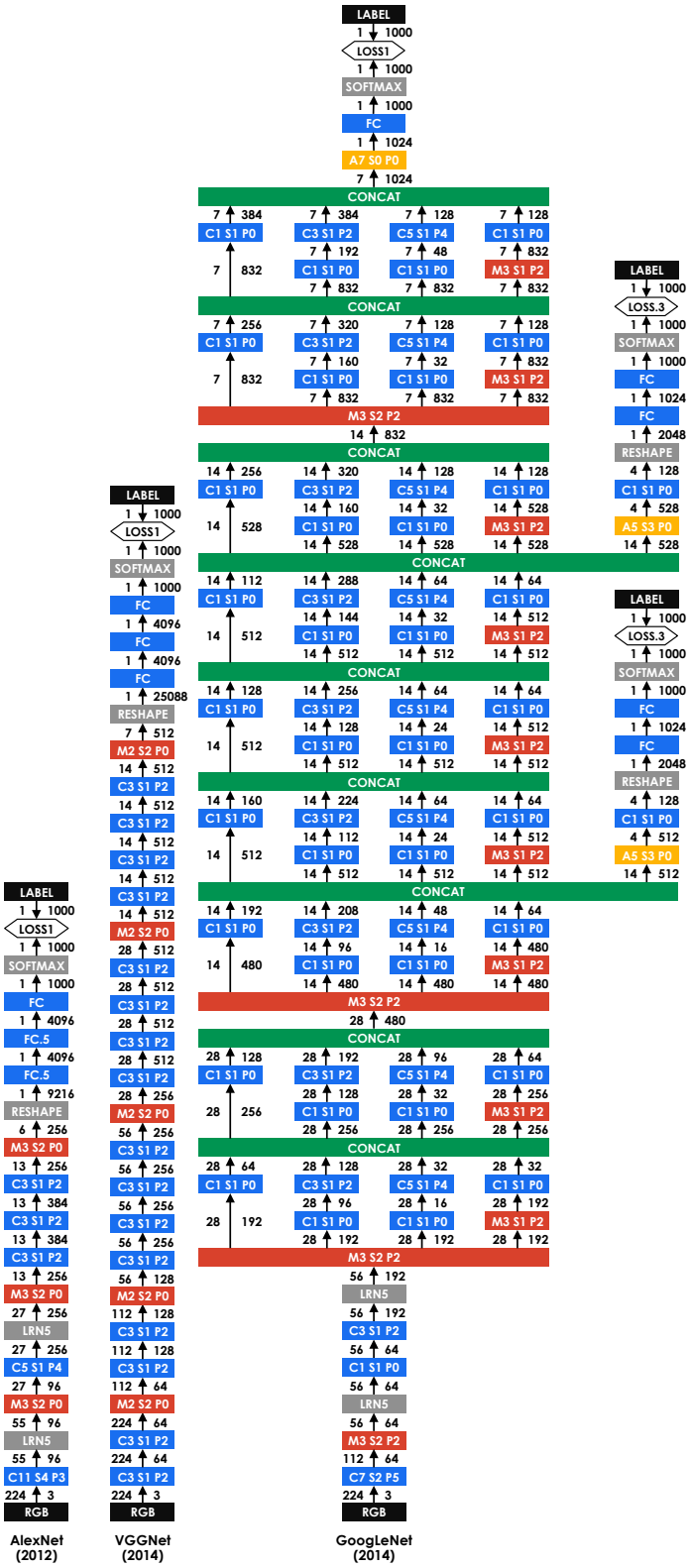


Figure 3.9: Different popular CNN architectures: *Alexnet* [79], *VGG* [137] and *GoogLeNet* [144]. The design of CNN can vary substantially but steadily increasing depth is a common trend amongst others.

58, 57]. Multiple layers of three-by-three convolutions are often stacked without interleaving spatial downsampling layers (such as max-pool). This setup is not strictly equivalent to larger convolutions mainly due to the additional nonlinearities.

one-by-one "convolutional" kernels have also been proposed with the *Network-in-Network* model [92] and are in increasingly common use with good results [144, 57]. They are effectively a combination of the different feature maps and are mostly used to control (reduce [144, 57] or increase [57]) the amount of feature maps.

Ever Deeper Networks. The depth of CNN has also been steadily increasing from the 8 layers of *AlexNet*. The most noteworthy contribution in the field is the Residual Networks (*ResNet*) [57]. They introduce "projection shortcuts" making the learning of identity layers easier and effectively removing a barrier for training deeper models. *ResNet* won the ILSVRC 2015 challenge and their single best-performing model was 152 layers deep.

Recent work on Stochastic *ResNet* [67], an extension of unit dropout to dropping entire layers, showed that very deep networks can be trained effectively (more than 1000 layers) even on smaller datasets.

Improved Spatial Downsampling. Controlling spatial dimensionality can typically be achieved either via pooling or striding. Unlike *AlexNet* which does both (presumably due to hardware memory constraints), most recent CNN do either of the two. While max-pooling seemed to be the favoured approach (e.g. *VGG* or *GoogLeNet*), the recently proposed *ResNet* only uses convolutions and striding. The idea is to let the network learn the pooling strategy rather than enforce it a priori. Empirical results suggest that CNN maximum pooling layers can be replaced by convolutional layers with large strides without accuracy loss [139]. The choice is also possibly influenced by the popularization of generative CNN models [125] for which irreversible pooling operations are undesirable.

We can also cite several works related to introducing stochastic behaviors during pooling [166, 158, 165] and to learning the downsampling operations [87, 50].

Better Activation Units. Several extensions have been proposed to ReLU:

- Leaky ReLU (LReLU) [102] allows for a small, non-zero gradient when the unit is saturated and not active.
- Parametric ReLU (PReLU) [58] where the negative slope is learned during training via back-propagation at the expense of making the network more prone to overfitting.
- Randomized ReLU (RReLU) [161] where the negative slope parameter is randomly sampled during training, is an effective regularizer.

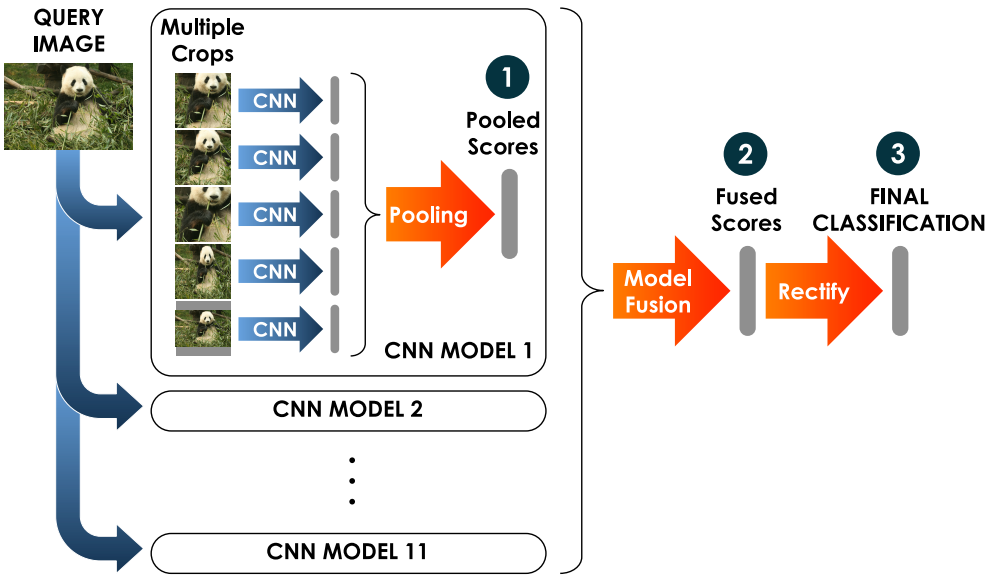


Figure 3.10: Adaptive fusion of multiple CNN. The process has three successive stages: (1) patch aggregation, (2) model fusion and (3) output rectification. Reported numerical results are for the ImageNet-1k validation set (top-5 classification error).

Tweaks Improving Training. A number of techniques such as bach normalization [69] and better weight initialization schemes [58] have been proposed mostly to improve training times and also improve results on deeper nets, and are now commonly used across newer models.

3.3 Adaptive Fusion of CNN

CNN are currently the method of choice for visual recognition tasks including large-scale image classification. As seen in Section 3.2.3, the design choices made play an important role. However, single CNN usually don't achieve best classification performance by themselves as evidenced by submissions to ILSVRC. Best performing models are in fact almost always an aggregation of multiple models.

In this section we propose a method to combine the predictions of multiple CNN. The pipeline is a hierarchical, multi-stage process described in Section 3.3.1. The method was designed with ImageNet-1k benchmark in mind and was submitted at ILSVRC 2014 as "Large Scale Image Classification on a Shoestring" [105] and allowed our team to rank 9th at the classification task.

3.3.1 Method

In this section, we propose a method for the fusion of multiple CNN. The overall process which is represented in Figure 3.10 has three successive stages:

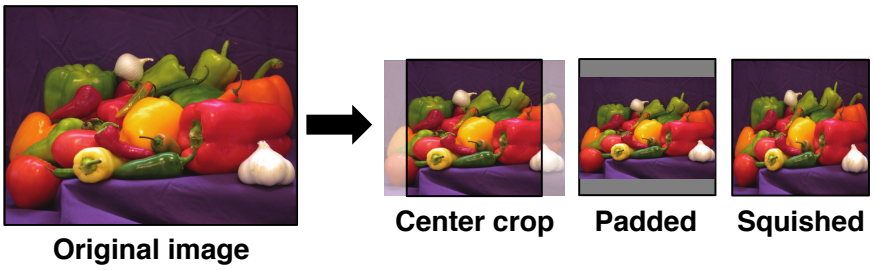


Figure 3.11: Three different strategies for extracting square CNN input from a non-square image. Each has its advantages and downsides.

1. A patch aggregation step during which predictions of multiple patches of the original image are aggregated for each CNN.

2. A model fusion step where the aggregated output for each CNN is fused into a single prediction.

3. An output rectification step to rescale probabilities based on the expected class recall rates.

Image Patch Aggregation. Different images have changing aspect ratios while CNN take inputs of fixed size (usually squared). This fact alone means that the input image must usually be transformed (beyond just isotropic scaling) before being fed into a CNN. Several patch extraction strategies can be devised but each has its advantages and downsides. For instance, as depicted on Figure 3.11, one may want to:

1. preserve all the pixels from the original image;

2. use uniform vertical and horizontal scaling;

3. or avoid padding with artificial data.

While it is possible to do any two of the above ("center crop" complies with 2 and 3, "padded" with 1 and 2, and "squished" with 1 and 3), it is unfortunately impossible to do all three at the same time.

There is in fact no single best patch extraction strategy. Instead, aggregating the predictions obtained from several patches is a common practice. For instance, Krizhevskhy et al. [79] proposes an aggregation scheme for the *AlexNet* model which computes the average of the prediction of 10 different 224×224 crops extracted from the original 256×256 image. This strategy improves the top-5 error score on ImageNet-1k from 18.3% down to 17.0% [79].

Our method refines the approach by extracting 34 patches (Figure 3.12) from every image at different scales and positions. First, we define the parametrized softmax for

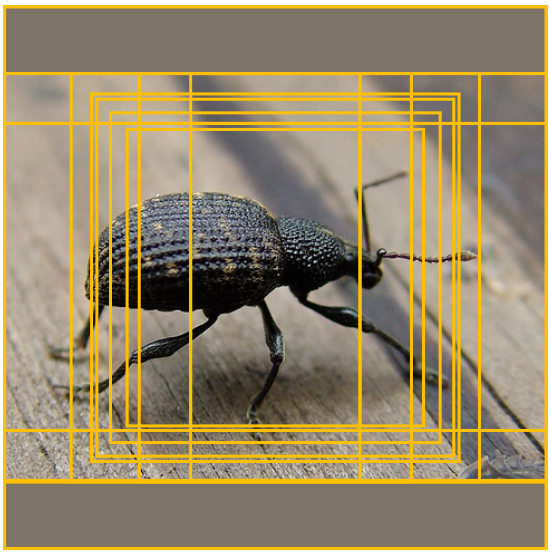


Figure 3.12: The 34 patches extracted from the original image. They are taken at different scales and positions, with or without padding. The dataset mean pixel value is used for padding.

an n-dimensional vector $(v_i)_{i\in[1,n]}$:

$$\text{softmax}_\alpha(v_i) = \frac{\exp(\alpha v_i)}{\sum_{j=1}^{n}\exp(\alpha v_j)} \tag{3.6}$$

and the vector notation:

$$\text{softmax}_\alpha(v) = (\text{softmax}_\alpha(v_i))_{i\in[1,n]} \tag{3.7}$$

This is a variant of the usual softmax normalization where the "hardness" of the normalization can be tuned via the parameter α.

In our method, the activation vector $(f_{i,j})_{j\in[1,1000]}$ for the last full-connected layer prior to softmax normalization (usually referred to as *"fc8"*) is extracted for each patch $i \in [1, 34]$. We subsequently aggregate the vectors in three steps using hyperparameters.

Parametrized softmax is applied across the classes to each individual patch $i \in [1, 34]$:

$$x_i^\alpha = \text{softmax}_\alpha(f_i) \tag{3.8}$$

where α is the softness parameter for class-wise normatization. It is usually parametrized "softer" than the regular softmax ($\alpha < 1$) implying that enhancing the probability of the less-well ranked classes is useful.

Next, we normalize activations across patches for each class $j \in [1, 1000]$ in a similar

fashion:

$$y_j^{\alpha,\beta} = \text{softmax}_\beta(x_j^\alpha) \tag{3.9}$$

where β is the softness parameter for patch-wise normalisation.

Finally, the activation for the different crops are combined with a linear combination with parameters $(\gamma_i)_{i\in[1,34]}$:

$$z_j^{\alpha,\beta,\gamma} = \sum_{i=1}^{34} \gamma_i y_{i,j}^{\alpha,\beta} \tag{3.10}$$

The output vectors $z_j^{\alpha,\beta,\gamma}$ is subsequently L_1 normalized to re-scale the class probabilities for each image.

The parameters α, β and γ are tuned by local search with multiple random initializations for each model.

Model Fusion. Once patch aggregation is performed for each model m, the class probability vectors Z_m for each models are merged with a linear combination with parameters μ_i:

$$Z^\mu = \sum_m \gamma_m Z_m \tag{3.11}$$

The parameters μ of the linear combination for model fusion are adjusted in a similar fashion to the patch aggregation parameters.

Output Rectification. During the final step, we rescale class-wise probabilities so that class recall rates are consistent with the ones encountered in the dataset. Figure 3.13, shows the distribution of the class recall rates in the top-n predictions for various n on the ImageNet-1k validation set for which all classes are equiprobable. We observe that the higher the value of n, the more skewed the distribution becomes which can be indicative of a bias. In the case of ILSVRC which evaluates submissions based on the top-5 error rate (five separate guesses are allowed per instance), we would thus normalize each class predictions with the actual top-5 results.

3.3.2 Evaluation Framework

The ImageNet-1k dataset was used for the evaluation and the model was subsequently submitted to ILSVRC 2014. The ImageNet-1k dataset contains 1000 different classes spread across 1.2 million training set instances, 50,000 validation set instances and 100,000 test set instances. Classes are equiprobable in the validation test and the test set.

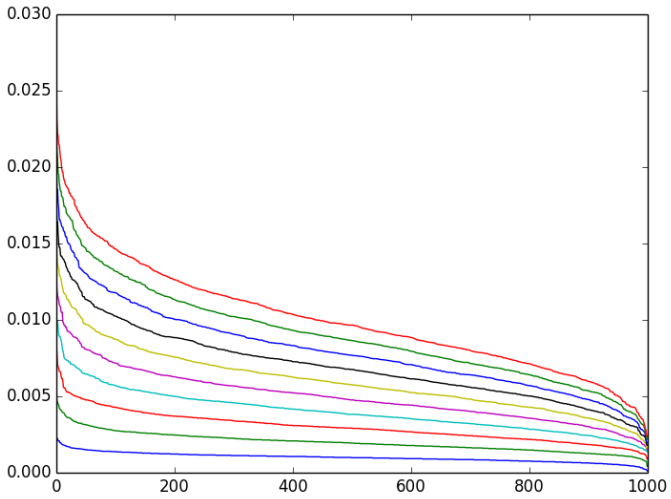


Figure 3.13: Class recall rates in the top-n predictions for $n = 1$ to 10 (bottom to top) on the ImageNet-1k validation set which has equiprobable class occurence. We observe that the higher the value of n, the more skewed the distribution becomes.

The whole adaptive fusion pipeline used the validation set for parameter tuning. All the reported error figures correspond to the top-5 metric used for ILSVRC. The numerical results which are reported are for the validation set. Challenge results (on the test set) are also reported.

The eleven models described below were used for the aggregation.

AlexNet is provided as a Caffe package [77]. It is a close replication of the ILSVRC 2012 contest winning model [79], the main differences being it is a single column architecture (no model parallelism) and it has been trained without RGB altering data augmentation. It is an eight layer network, with five convolutional layers and three fully-connected layers.

CaffeNet model is provided as a Caffe package. Its architecture is very similar to the *AlexNet* one; the main difference being pooling is done before normalization. It performs slightly better than *AlexNet*.

CCV model is part of the eponymous library *libccv* [94]. It has more parameters than *AlexNet*, with smaller filters on the first layer and less aggressive convolution stride.

The next two models come from the New York University Computational Intelligence, Learning, Vision, and Robotics Laboratory (CILVR Lab) [135]. They are both twice as large as *AlexNet*. They were trained with the Torch library [23].

Method	Top-5 error (%)	Improvement (%)
Multiple patches, multiple models, rectification	11.3	26.5
Multiple patches, multiple models	11.4	26.0
Multiple patches, single model	12.1	21.2
Single crop, single model (best) [17]	15.4	0

Table 3.1: ImageNet-1k classification error associated to different stages of the fusion pipeline. The improvement is computed in relation with the best performing single CNN architecture which is VGG-CNN-Slow [17].

Overfeat-Fast model is eight layers deep and has significantly more filters in the last three convolutional layers compared to Zeiler architecture [167].

Overfeat-Accurate is deeper than *Overfeat-Fast*, with a total of nine layers (six convolutional layers and three fully-connected layers). Its first convolutional layer stride is also less aggressive.

The last 6 models are from Oxford University Visual Geometry Group (VGG) [17]. They are provided as a Caffe [77] package on the *Caffe model zoo* platform. They are all based on an eight-layer architecture inspired from *AlexNet*, with significantly larger fully-connected layers. They are based on three architectures: *CNN-Fast*, *CNN-Medium* and *CNN-Slow*.

VGG-CNN-Fast model architecture is similar to *AlexNet* with a comparable number of convolutional filters, and significantly larger fully-connected layers.

VGG-CNN-Medium model architecture is similar to *VGG-CNN-Fast*, with twice as many filters for the last three convolutional layers.

VGG-CNN-Medium-2048 model has the same architecture as *VGG-CNN-M*, with reduced $fc7$ layer size (2048 instead of 4096).

VGG-CNN-Medium-1024 model has the same architecture as *VGG-CNN-M*, with reduced $fc7$ layer size (1024 instead of 4096).

VGG-CNN-Medium-128 model has the same architecture as *VGG-CNN-M*, with reduced $fc7$ layer size (128 instead of 4096).

VGG-CNN-Slow is similar to *VGG-CNN-M*, the differences being a larger pooling kernel size on the first and last convolutional layers and a less aggressive stride on the second convolutional layer.

Table 3.2: Relative weight associated to each model during model fusion.

Model	Aggregation weight (%)
VGG-CNN-M	1.2
VGG-CNN-M-2048	28.1
VGG-CNN-M-1024	0.7
VGG-CNN-M-128	0.6
VGG-CNN-F	0.0
VGG-CNN-S	34.3
CCV	0.0
AlexNet	0.0
CaffeNet	0.0
Overfeat-S	17.5
Overfeat-L	17.5

3.3.3 Empirical Results

Table 3.1 shows that every step of the fusion process leads to improvements. The most significant performance improvement comes with the patch aggregation step (+21.2% over single crop baseline), then with model aggregation (+26.0%). Our adaptive patch aggregation scheme performs notably better than the crop averaging scheme proposed in [17]: 12.1% error against 13.1%.

Table 3.2 shows the relative weight associated to each model during model fusion. The best performing and largest models are given more importance (higher weights) than the smaller ones. Some older models such as *AlexNet* are completely ignored (zero weight). Note the single best performing model (VGG-CNN-S) is given the highest weight, but significant weight is also attributed to other well performing models such as Overfeat and VGG-CNN-M-2048.

Table 3.3 shows the best submission of each team participating to the LSVRC 2014 image classification contest. Our team achieves 11.326% top-5 error earning the 9th place. Note that our submission outperforms the best submission from 2013 (11.743% top-5 error) which is not publicly documented and was not included in the model. The contest results which are computed on the test set are very consistent with our validation set results which shows that our method did not suffer from overfitting during parameter tuning on the validation set.

3.4 Summary and Discussion

This chapter presents the design principles behind the CNN which recently became the method of choice for large-scale image classification tasks, and puts into perspective the related work carried out over recent years. We also propose an original multi-stage approach for the fusion of the output of multiple CNN. Submitted as part of

Table 3.3: LSVRC 2014 top-5 classification error results.

Organization	Team name	Top-5 classification error
Without external training data		
Google	GoogleLeNet	0.06656
Oxford University	VGG	0.07325
Microsoft	MSRA Visual Computing	0.0806
Howard Vision Technologies	Andrew Howard	0.08111
Baidu	DeeperVision	0.09508
National University of Singapore	NUS-BST	0.09794
Toyota Technological Institute	TTIC_ECP-EpitomicVision	0.10222
University of Queensland	XYZ	0.11229
I2R - UPMC	**BDC**	**0.11326**
University of Amsterdam	UvA-Euvision	0.12117
Lunit	Cldi-KAIST	0.13949
Sun Yat-Sen University	SYSU_Vision	0.14446
Orange	ORANGE-BUPT	0.15158
Libccv	Libccv	0.16032
Lenovo	PassBy	0.16705
Fengjun Lv Consulting	Fengjun Lv	0.17352
Ben Graham-University of Warwick	DeepCNet	0.17481
Brno University of Technology	Brno	0.17647
University of Tokyo	MIL	0.18278
South China University of Technology	SCUT_GLH	0.18784
With external training data		
Chinese Academy of Sciences	CASIA_CRIPAC	0.11358
Chinese Ministry of Public Security	Trimps-Soushen	0.1146
Adobe	Adobe-UIUC	0.11578
Orange	ORANGE-BUPT	0.14797

the ILSVRC 2014 challenge, results show that the approach can significantly improve classification results.

As discussed in the previous chapter, the good performance of CNN is largely due to their capability to learn appropriate high-level visual representations from the data. A stream of recent works show that the representations learnt on one particular classification task can transfer well to other classification tasks [1, 164, 167, 127]. In the following chapters, we study the transferability of representations learnt by CNN to image instance retrieval, a visual task of a different nature.

Chapter 4

Global Image Descriptors: CNN vs Fisher Vectors

4.1 Introduction

With deep learning becoming the dominant approach in computer vision, the use of representations extracted from Convolutional Neural Nets (CNN) is quickly gaining ground on Fisher Vectors (FV) as favoured state-of-the-art global image descriptors for image instance retrieval. While the good performance of CNN for image classification is unambiguously recognized, which of the two has the upper hand in the image retrieval context is not entirely clear yet. In this chapter, we propose a comprehensive study, which systematically evaluates FV and CNN for image retrieval.

First, we compare the performances of FV and CNN on multiple publicly available data sets. We investigate a number of details specific to each method. For FV, we compare sparse descriptors based on interest point detectors with dense single-scale and multi-scale variants. For CNN, we focus on understanding the impact of depth, architecture and training data on retrieval results. Our study shows that no descriptor is systematically better than the other and that performance gains can usually be obtained by using both types together. The second part of the study focuses on the impact of geometrical transformations such as rotations and scale changes. FV based on interest point detectors are intrinsically resilient to such transformations while CNN do not have a built-in mechanism to ensure such invariance. We show that performance of CNN can quickly degrade in presence of rotations while they are far less affected by changes in scale. The key findings from our study are summarized in Table 4.1 intended as a quick reference guide for practical guidelines on the use of FV and CNN for image retrieval.

QUESTIONS	OBSERVATIONS AND RECOMMENDATIONS
Best practices for CNN descriptors	
Best single crop strategy?	The largest possible center crop (discarding parts of the image but preserving aspect ratio) or the entire image (preserving the entire image but ignoring aspect ratio) work comparably, both outperforming padding (preserving both).
Best performing layer?	The first fully-connected layer is a good all-round choice on all the tested models.
Do deeper networks help?	Only if the training and test data are similar. Otherwise, extra-depth can hurt performance.
How much does training data matter?	Training data has significant impact on performance. Results also suggest that deeper layers are more domain specific.
Best practices for FV interest points	
Dense or sparse interest points?	It depends on the data set. If scale and rotation invariance are not required, and the data are highly textured, dense sampling outperforms DoG interest points.
Single-scale or multi-scale interest points?	Multi-scale interest points always improve performance.
CNN versus FV	
How do state-of-the-art CNN and FV results compare on standard benchmarks?	It depends on the characteristics of the data set.
Does combining FV and CNN improve performance?	Yes, combining FV with state-of-the-art CNN descriptors can improve retrieval performance often by a significant margin.
Invariance to rotations	
How invariant are CNN features to rotation?	CNN features exhibit very limited invariance to rotation, performance drops rapidly as query rotation angle varies.
Are CNN or FV more invariant to rotation?	FV based on DoG interest points are robust to rotation changes, as would be expected. CNN descriptors are more robust to rotation changes than FV based on dense sampling.
Are deeper CNN layers more invariant to rotation?	The fully-connected layers exhibit similar invariance properties to rotation. Visual features (*pool*5) are slightly more robust to small rotation angles but significantly less robust to larger angles.
Invariance to scale changes	
How scale-invariant are CNN features?	CNN descriptors are robust to scale change and work well even for small query scales.
Are CNN or FV more scale-invariant?	CNN descriptors are more robust to scale changes than any FV. All FV variants experience a much sharper drop in performance as query scale is decreased compared to CNN features.
Are deeper CNN layers more scale-invariant?	Visual features (*pool*5) are more scale-invariant than the deeper fully-connected layers.

Table 4.1: Summary of experimental results and key findings.

4.2 Evaluation Framework

We evaluate the performances of the descriptors on four popular data sets: *Holidays*, *Oxbuild*, *UKBench* and *Graphics*. The four data sets are chosen for the diversity of data they provide: *UKBench* and *Graphics* are object-centric featuring close-up shots of objects in indoor environments. *Holidays* and *Oxbuild* are scene-centric data sets consisting primarily of outdoor buildings and scenes. Retrieval data sets are further described in Section A.

4.2.1 Fisher Vectors

FV are a concatenation of first and second order statistics of a set of feature descriptors quantized with a small codebook. All images are resized (maintaining aspect ratio) so that the larger dimension of the image is equal to 640 pixels prior to FV extraction. We use the implementation of FV from the open source library VLFeat [151]. SIFT detectors and descriptors are also chosen from the same library. The three different types of SIFT descriptors used to generate the FV are Difference of Gaussians (DoG) SIFT, Dense Single-scale SIFT and Dense Multi-scale SIFT.

- **DoG SIFT**. We detect interest points in the DoG scale space, followed by 128-dimensional SIFT descriptors extracted from scaled and oriented patches centered on interest points. Default peak and edge thresholds (0 and 10) are employed to filter out low contrast patches or patches close to the edge of the image. Since the DoG detector extracts scale and rotation invariant interest points, it has been widely applied for the task of instance retrieval. It is important to note that we do not use any feature selection algorithm to select a subset of "good" features - an approach that can result in a significant improvement in performance on the *Graphics* dataset [35].

- **Dense Single-scale SIFT**. We extract SIFT descriptors from densely sampled patches (every four pixels) with fixed scale and upright orientation. The patch size used for the extraction is $m \times s$ where s is the scale parameter and m is the magnification parameter. We choose the default magnification parameter $m = 6$ across all dense SIFT descriptors. $s = 4$ is chosen for single-scale SIFT. Dense SIFT is faster to compute than DoG SIFT as the expensive interest point detection step is avoided - however, this comes at the cost of lower scale and rotation invariance. Note that dense SIFT is mostly popular for image classification tasks.

- **Dense Multi-Scale SIFT**. We apply dense SIFT extraction at multiple resolutions ($s = \{4, 8, 12, 16\}$). This is aimed at gaining some degree of scale invariance.

Closely following [74, 122], we apply dimensionality reduction on SIFT descriptors from 128 to 64 using PCA, and train a Gaussian Mixture Model (GMM) with 256

	ARCHITECTURE			TRAINING			LAYER SIZE			
	parameters	depth (conv+fc)	input size	training set	classes	data size	*pool5*	*fc6*	*fc7*	fc8
OxfordNet	138M	13+3	$224 \times 224 \times 3$	*ImageNet*	1000	1.2M	$7 \times 7 \times 512$	4096	4096	1000
AlexNet	60M	5+3	$227 \times 227 \times 3$	*ImageNet*	1000	1.2M	$6 \times 6 \times 256$	4096	4096	1000
PlacesNet	60M	5+3	$227 \times 227 \times 3$	Places-205	205	2.4M	$6 \times 6 \times 256$	4096	4096	205
HybridNet	60M	5+3	$227 \times 227 \times 3$	Both	1183	3.6M	$6 \times 6 \times 256$	4096	4096	1183

Table 4.2: Details on architecture, training set and layer size of the CNN.

centroids. Both first order (gradients w.r.t. mean) and second order (gradients w.r.t. variance) statistics are encoded to form the FV, resulting in a $64 \times 256 \times 2 = 32768$-dimensional vector representation for each image. Finally, we apply power law normalization to each component ($\alpha = 0.5$), followed by L_2 normalization to obtain the final normalized FV representation [122]. Each dimension of the FV is stored as a floating point number. No compression is applied. We refer to the three FV as FVDoG (FV computed on DoG points), FVDS (FV computed densely at a single scale) and FVDM (FV computed densely at multiple scales) from here on.

4.2.2 Convolutional Neural Network Descriptors

In this work, four different pre-trained CNN models are considered for the instance retrieval problem:

- *OxfordNet* [137]: the best performing single network from the Oxford VGG team at *ImageNet* 2014.
- *AlexNet* [79]: the model referenced as "BVLC reference caffenet" in the Caffe framework [77]. This model was the winning *ImageNet* submission of 2012. This network closely mimics the original *AlexNet* model of [79].
- *PlacesNet* [172]: a state-of-the-art model for scene image classification providing highest accuracy on the SUN397 dataset [160].
- *HybridNet* [172]: another model for both object and scene images classification, outperforming state-of-the-art methods on the MIT Indoor67 dataset [124].

Details on the architecture, training set and layer sizes of the CNN are summarized in Table 4.2.

These state-of-the-art models are chosen as they allow us to run interesting control experiments, where the CNN architecture or training data are varied. *PlacesNet* and *HybridNet* share the same architecture as *AlexNet* [172], while being trained on different data. *OxfordNet* and *AlexNet* are trained on the same data, but have different architectures: compared to *AlexNet*, *OxfordNet* is deeper, has twice as many layers, twice the number of parameters, and achieves better image classification performance in the *ImageNet* 2014 contest [137].

The four models are trained differently, using the ImageNet [30] and Places-205 [172] data sets. With categories like "Amphitheater", "Jail cell" or "Roof garden", Places-205 is a scene-centric data set, while *ImageNet*, featuring categories such as "Vending machine", "Barn spider" or "Chocolate syrup", is more object-centric. Places-205 is twice as large as *ImageNet*, but has five times fewer classes. *OxfordNet* and *AlexNet* are trained on *ImageNet*. *HybridNet* is trained on a combination of *ImageNet* and Places-205 data: the resulting data set being three times larger than *ImageNet* alone, and having a larger variety of classes.

Given an input image, we first resize it to a canonical resolution, compute the feed-forward neural network activations, and extract the last four layers for each CNN model. We refer as $pool5$, $fc6$, $fc7$ and $fc8$ outputs of the last four layers of each network (as denoted in Caffe). $pool5$ is the output of the last convolutional layer after pooling, and $fc6$, $fc7$, $fc8$ are outputs of the fully-connected layers. Contrary to fully-connected layers, $pool5$ still contains explicit spatial information from the input image. The size of the last layer $fc8$ is equal to the number of classes. All descriptors are extracted after applying the rectified linear transform, and L_2 normalized.

4.3 Experimental Results

4.3.1 Best Practices for CNN Descriptors

Image cropping strategy. CNN pipelines take input images at a fixed resolution (see Table 4.2). We wish to determine which single-crop strategy works best for the purpose of instance retrieval where images may vary in size and aspect ratio. We consider the following three different cropping strategies illustrated in Figure 4.1. Numerical values are given to fit *OxfordNet*.

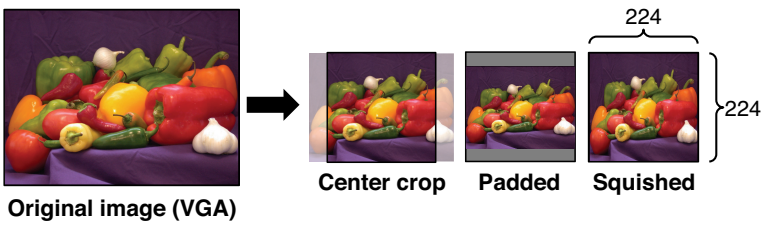


Figure 4.1: Different single-crop strategies used for input into CNN pipelines.

- *Center*: the largest 224 × 224 center crop after rescaling the image to 224 pixels for the smaller dimension while maintaining aspect ratio.
- *Padded*: the original image is resized to 224 pixels for the larger dimension maintaining aspect ratio and any unfilled pixels are padded with a constant value equal to the training set mean.

- *Squished*: the original image is resized to 224 × 224.

Each strategy has its drawbacks: *Center crop* discards information at the periphery, *Squished* ignores aspect ratio, and while preserving overall image content and aspect ratio, *Padded* introduces border artifacts.

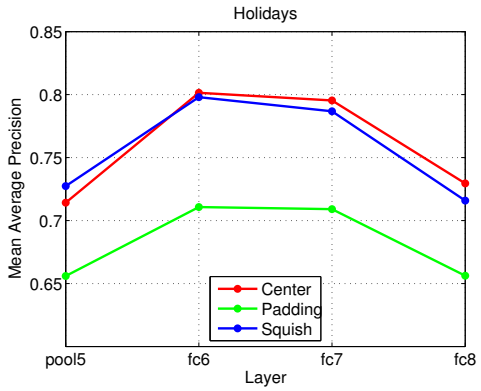


Figure 4.2: mAP for different layers of *OxfordNet*, for different single-crop strategies on the *Holidays* data set. We observe that *Center* crop and *Squished* perform comparably.

In Figure 4.2, we plot mAP for different layers of *OxfordNet*, for the *Holidays* data set. We note that *Center* and *Squished* perform comparably, outperforming *Padded*. The trend is consistent across the different network layers. We observe similar results for other data sets and CNN models. Most data sets in this study have a center bias for the object of interest, explaining the best performances of the *Center* cropping strategy. *Center crop* is selected as the single-crop model input strategy for the following experiments.

Influence of CNN layer. In Figure 4.3, CNN descriptors extracted from different layers are compared for image instance retrieval. The descriptors from layers *pool*5, $fc6$, $fc7$ and $fc8$ are extracted from four models *OxfordNet*, *AlexNet*, *HybridNet* and *PlacesNet* on different data sets.

We note that for each network, intermediate layers perform best for instance retrieval. Such a sweet spot is intuitive as the final layer represents high level semantic concepts, while intermediate convolutional and fully-connected layers provide rich representations of lower level image information. In most cases, intermediate layer $fc6$ performs significantly better than *pool*5 and $fc8$ and marginally better than $fc7$. For *Graphics*, performance drops with increase in depth, as all four CNN models are learnt on natural image statistics, while the *Graphics* data set is biased towards data like CD covers, DVD covers, business cards, and dense text in newspaper articles.

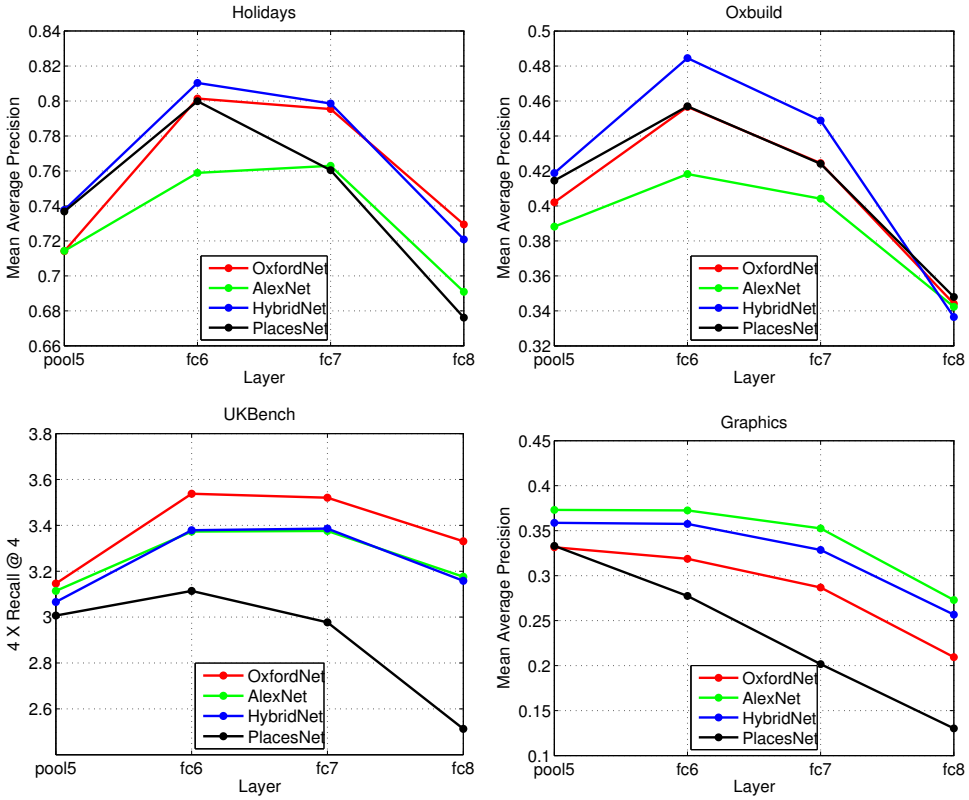


Figure 4.3: mAP for the last four layers of state-of-the-art publicly available CNN. *OxfordNet* and *AlexNet* are trained on the same data, while *PlacesNet*, *HybridNet* and *AlexNet* have the same network architecture, but are trained on different data. We note that performance improves by using deeper networks, and by training on domain specific data, but only if training and testing data have similar characteristics.

Influence of depth. We compare *OxfordNet* and *AlexNet* results in Figure 4.3. *OxfordNet* and *AlexNet* are both trained on the same 1.2 million images from the *ImageNet* data set, but with a variation in the number of layers: sixteen and eight layers respectively. We note that *OxfordNet* outperforms *AlexNet* on all data sets, except *Graphics*. On *Graphics*, the performance of *OxfordNet* is worse, strongly suggesting that deeper models have the potential of achieving higher discriminativeness on domain specific data at the expense of less generalisability on non-specific data.

Influence of training data. For this experiment, we compare *AlexNet* with *PlacesNet* in Figures 4.3. *AlexNet*, and *PlacesNet* use the same 8-layer CNN architecture, but are trained on different data. We observe that *PlacesNet* outperforms *AlexNet* on *Holidays* and *Oxbuild* data sets. This shows that using training data more representative of the test data can improve performance significantly, as *Holidays* and *Oxbuild* are scene-centric. On the object-centric *UKBench* and *Graphics* data sets, *PlacesNet* performs worse than *AlexNet* due to the mismatch between training and testing data.

Further, in Table 4.3, we compare our results to the CNN retrieval results presented in [9]. In [9], Babenko et al. fine-tune a pre-trained *AlexNet* model based on *ImageNet* training data with domain specific images, e.g., landmarks and objects. As shown in Table 4.3, the authors are able to improve retrieval performance over the *AlexNet* baseline model, on *Holidays* and *UKBench* by fine-tuning with landmark and object data respectively. However, the resulting trade-off is a loss in performance on *Holidays* when fine-tuning with object data and vice-versa, suggesting improvements coming from domain specific fine-tuning are obtained at the expense of lower performance on other data sets.

The comparison of *AlexNet*, *PlacesNet* and *HybridNet* shows that *AlexNet* and *HybridNet* perform comparably on the two object related data sets, and *HybridNet* outperforms *PlacesNet* on the two scene related data sets. *HybridNet* significantly and systemically outperforms average combination of *AlexNet* and *PlacesNet* scores. In three out of four data sets, *HybridNet* outperforms the deeper model *OxfordNet*, which was trained on *ImageNet* only. From these observations, it can be concluded that (1) retrieval accuracy improves with the quantity and variety in the training data set, potentially even if additional training data are less related to target retrieval task (2) larger quantity and variety of training data can be more beneficial than deeper architecture. *ImageNet* and *Places* data sets alone being considered as large data sets (1M+ images), this experiment underlines the importance of large scale training data for learning high quality deep representations.

4.3.2 Best practices for FV Interest Points

CNN features are obtained by dense sampling over the image. Contrary to CNN, FV allows for different sampling strategies, including dense sampling. In Table 4.3,

Descriptor	Dim	*Holidays*	UKBench	*Oxbuild*	Graphics
OxfordNet	4096	0.80	3.54	0.46	0.33
AlexNet	4096	0.76	3.38	0.42	0.37
HybridNet	4096	0.81	3.39	0.48	0.36
PlacesNet	4096	0.80	3.11	0.46	0.33
CNN (Fine-tuned on Landmarks) [9]	4096	0.793	3.29	0.545	
CNN (Fine-tuned on Objects) [9]	4096	0.754	3.56	0.393	
FVDoG	32768	0.63	2.8	0.42	0.66
FVDS	32768	0.73	2.38	0.51	0.20
FVDM	32768	0.75	2.45	0.55	0.32

Table 4.3: CNN and FV results for instance retrieval. 4 × Recall @ 4 for *UKBench*, and mAP for other data sets.

we study if such an approach is also effective for FV. The performances of different FV interest point sampling strategies (FVDoG, FVDS and FVDM) are compared in Table 4.3.

Dense sampling (FVDS and FVDM) improves performance over FVDoG on *Holidays* and *Oxbuild* data sets, while hurting performance on *Graphics* and *UKBench*. Single scale dense sampling (FVDS) and multi-scales dense sampling (FVDM) perform comparably, except on *Graphics* with a significant difference in favor of multi-scale sampling.

The gradually and significantly increasing performances of FVDS, FVDM and FVDoG on *Graphics* are explained by the fact the data set involves frequent rotations of objects of interest, and a strong bias toward the center. The better performance of FVDS over FVDoG on both *Holidays* and *Oxbuild* suggests that objects of interest occur roughly at the same scale. This is credible as the data sets consist mainly in landmarks.

Dense sampling is effective for data sets like *Holidays* which consist primarily of outdoor scenes, and are mainly composed of highly textured patches. The improvement in performance of dense sampling approaches can also be attributed to the discriminativeness-invariance trade-off. Where retrieval does not require scale and rotation invariance, and data are highly textured over the entire image, performance can be improved by dense sampling. Sampling at multiple scales also seems to consistently improve results over single scale sampling for dense descriptors.

4.3.3 Comparisons to State-of-the-Art

Fusion of FV and CNN. In Figure 4.4, we present retrieval results obtained by combining FVDoG, FVDS, and FVDM individually with *OxfordNet fc6* features. We

employ a simple early fusion approach where the FV and CNN features are concatenated after weighting by α and $(1-\alpha)$ respectively. $\alpha = 0$ corresponds to using just FVDoG, FVDS or FVDM features individually, while $\alpha = 1$ corresponds to just the *OxfordNet* feature. This early fusion scheme is also equivalent to weighting the squared L_2 distance measure for matching by α and $1-\alpha$ for FV and CNN features respectively.

The goal of this experiment is to show that performance can be significantly improved by combining FV and CNN features, and not necessarily to achieve highest performance on these retrieval benchmarks. Peak performance presented in Figure 4.4 can be improved by

- database-side rotation and scale pooling which is significantly helpful (see Chapter 6),
- better CNN models than *OxfordNet* on individual data sets (see Table 4.3),
- better FV based on Hessian Affine interest points [103] instead of DoG points used in this study [151],
- better FV with more sophisticated aggregation techniques [75],
- combining all FV and CNN descriptors together,
- using more sophisticated fusion and ranking techniques for combining results, like the one proposed in the recent paper [162].

All four data sets show an improvement in peak performance by combining FV and CNN features. The maximum performance is achieved for $\alpha = 0.4$ for the *Holidays*, *UKBench* and *Oxbuild* data sets, and $\alpha = 0.3$ for the *Graphics* data set, using different FV. There is a significant improvement in performance by combining FV and CNN features on all data sets except *Graphics*. The results suggest that a simple hyper-parameter can be used to combine FV and CNN across data sets with similar characteristics. Also, $\alpha = 0.4$ suggests that both FV and CNN contribute significantly to high retrieval performance.

Note that

- FVDoG is the only FV descriptor for which it consistently exists an α value improving performance over individual descriptors when combined with CNN descriptor and
- in average, combinations involving FVDoG yield higher improvement over best individual descriptor compared to combinations involving FVDS or FVDM.

This suggests that CNN descriptors, densely sampled over the input images, lack some of the invariance coming along with Difference of Gaussian FV sampling. Invariance properties of both CNN and FV are further investigated in following sections.

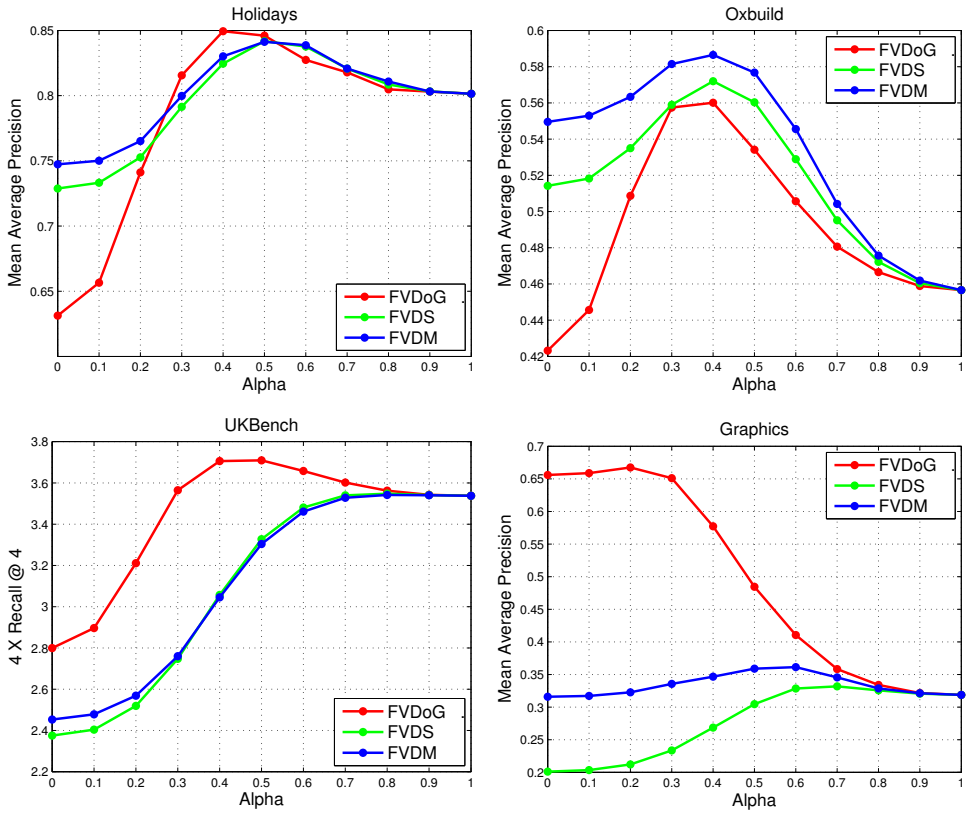


Figure 4.4: Combining different FV with *OxfordNet fc6* with early fusion. FV and *OxfordNet* features are concatenated with weights α and $1 - \alpha$ respectively. $\alpha = 0$ refers to just using FVDoG, FVDS or FVDM above, while $\alpha = 1$ refers to just the *OxfordNet fc6* feature. We observe that retrieval performance improves on all data sets by combining FV and CNN.

Descriptor	Dim	Holidays	UKBench	*Oxbuild*
Bag-of-words 1M [117]	1M		3.19	
VLAD baseline [73]	8192	0.526	3.17	
Fisher baseline [73]	8192	0.495	3.09	
Fisher baseline (ours)	32768	0.63	2.8	0.42
Fisher ADC (320 bytes) [74]	2048	0.634	3.47	
Fisher+color [52]	4096	0.774	3.19	
VLAD++ [5]	32768	0.646		0.555
Sparse-coded features [38]	11024	0.767	**3.76**	
Triangulation Embed [75]	8064	0.77		**0.676**
Triangulation Embed [75]	1920		3.53	
Best CNN results from this study across all CNN	4096	0.81	3.54	0.48
Fusion of *OxfordNet* and Baseline FV	32768+ 4096	**0.85**	3.71	0.59

Table 4.4: State-of-the-art results. mAP for *Oxbuild* and *Holidays*, and 4× Recall @4 for *UKBench*

Comparisons to state-of-the-art. Table 4.4 compares state-of-the-art results reported on *Holidays*, *UKBench* and *Oxbuild*. A wide range of approaches starting from Bag-of-words [117] to latest FV aggregation methods [75] is included. Also included is the best CNN and fusion results reported in this chapter. We note that the best CNN results (based on pre-trained models considered in this work) achieve higher performance than state-of-the-art FV approaches [75] on *Holidays* and *UKBench* data sets. There is a gap in performance between CNN results reported in this work and state-of-the-art FV for *Oxbuild*: however, *Oxbuild* is a much smaller data set with only 55 queries. Finally, we note that the simple fusion technique in Figure 4.4 results in highest or one of the highest performance numbers reported on each data set. Peak performance numbers for the fusion approach can be improved using approaches described above.

4.3.4 Invariance to Rotation

CNN descriptors invariance to rotation. CNN descriptors, unlike FVDoG, have limited levels of rotation invariance. The invariance arises from the max-pooling steps in the CNN pipeline, and rotated versions of objects present in the training data.

The rotation invariance of descriptors extracted from different layers of *OxfordNet* is benchmarked in Figure 4.5, by mimicking the method used for rotation invariant features evaluation in [146]: First, all database and query images are cropped circularly at the center and padded with *ImageNet* mean pixel RGB value to avoid edge artifacts. Descriptors are then extracted from upright database images and from 36 different

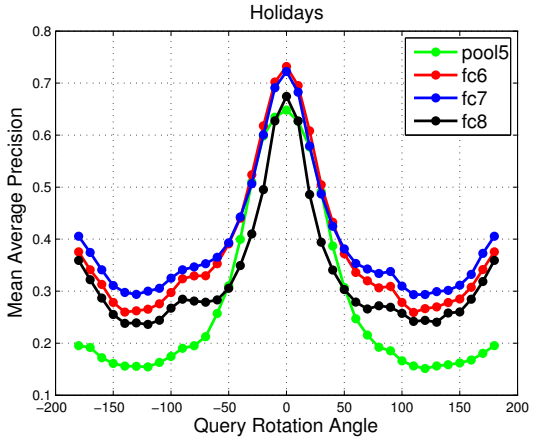


Figure 4.5: MAP on *Holidays* data set as query images are rotated for different layers of *OxfordNet*. We note that CNN features have very limited rotation invariance, with performance dropping steeply for all layers of the network beyond 10°.

rotations of each query image, evenly distributed from 0° to 360°. Finally, a retrieval experiment is done for each rotation, using query descriptors of images of same rotation only.

Overall, CNN descriptors have very limited rotation invariance with performance dropping steeply beyond 10°. The different layers of the network exhibit similar characteristics suggesting that rotation invariance does not increase with depth in the CNN. Fully-connected layers $fc6$, $fc7$ and $fc8$ descriptors behave comparably as the query image is rotated, suggesting rotation invariance doesn't increase with depth. Convolutional layer *pool*5 descriptor is slightly more rotation invariant at small angles, but performs worse at larger angles.

Comparison of FV and CNN rotation invariance. Rotation invariance properties of CNN and FV descriptors are compared following the procedure used in Section 4.3.4, i.e. using rotated query images. In Figure 4.6, *OxfordNet* first fully-connected layer $fc6$ descriptor is compared against FVDS, FVDM and FVDoG descriptors. A scene-centric and an object-centric data set are chosen for the sake of evaluation: *Holidays* and *Graphics*.

As expected, FVDoG descriptor is robust to rotation; the minor modulation in performance is due to filtering artifacts in the DoG interest point detector. However, the *OxfordNet* features, FVDS and FVDM have a steep drop in performance as queries are rotated. The *OxfordNet* features are more rotation invariant than FVDS and FVDM.

On *Graphics* data set, a large gap of performance is observed between FVDoG and other schemes. The performance gap between FVDoG and densely sampled FV strongly suggest the issue is interest point detection related. The underperformance

of all methods except FVDoG stresses the importance of the interest point detector. The poor performance of densely sampled descriptors (FVDS, FVDM, *OxfordNet*) on *Graphics* is probably a consequence of the geometric transformations found in a significant number of query images.

The difference at 0° between FVDS and FVDM on *Graphics* shows query images transformations involve scale changes. The important performance gap remaining at 0° between FVDM and FVDoG suggests that a significant number of query image transformations involve non-scale transformations, possibly rotations. The visualization of *Graphics* confirms this suggestion.

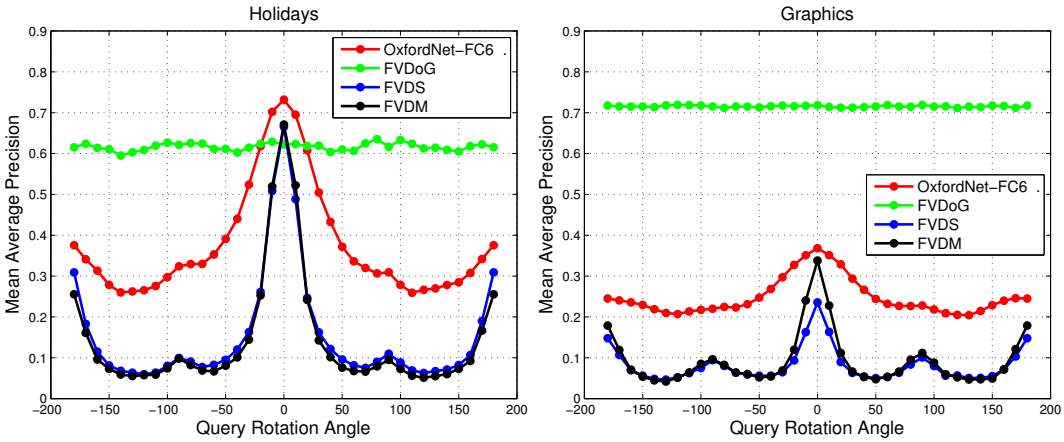


Figure 4.6: Comparison of *OxfordNet* $fc6$ and FV for rotated queries on the *Holidays* and *Graphics* data sets. The FVDoG is robust to rotation, while *OxfordNet*, FVDS, FVDM suffer a sharp drop in performance.

4.3.5 Invariance to Scale

In this section, we study scale invariance properties of CNN and FV. Similar to rotation experiments in Section 4.3.4, control experiments are carried out on the *Holidays* data where the scale of query images is reduced and retrieval performance measured. The starting resolution of all images (database and queries) is VGA, i.e the larger dimension is set to 640 pixels, maintaining aspect ratio. Query images are gradually scaled down with ratios of 0.75, 0.5, 0.375, 0.25, 0.2 and 0.125; the smallest queries are $\left(\frac{1}{8}\right)^{th}$ the size of the VGA resolution image. An anti-aliasing Gaussian filter is applied followed by bicubic interpolation in the downsampling operation. For each query scale, retrieval performance is evaluated against VGA database images.

Both CNN and FV pipelines take in input images at fixed resolution. For FVDoG, FVDS and FVDM pipelines, images are resized to VGA resolution (preserving aspect ratio) before feature extraction, even when input resolution is smaller, as up-sampling images before feature extraction is shown to improve matching performance [98]. For

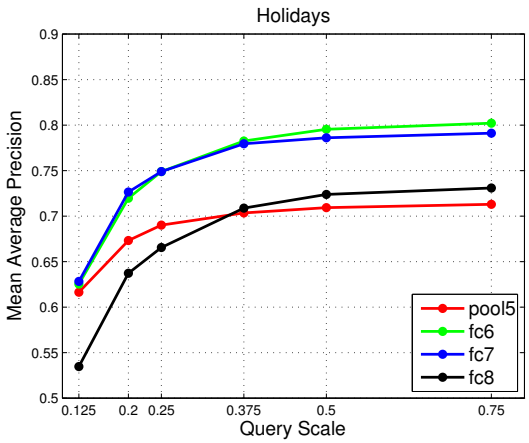


Figure 4.7: mAP as query images are scaled to 0.125 of original resolution, for different layers of *OxfordNet* on the *Holidays* data set. We note that *OxfordNet* features are robust to scale change up to 0.25, with performance dropping steeply after.

CNN, input images are center cropped and resized to fit the model input size specified in Table 4.2.

CNN descriptors invariance to scale. Scale invariance of descriptors extracted from different *OxfordNet* layers is evaluated in Figure 4.7. All four layers display some degree of scale invariance, with no significant performance deterioration above half of the original scale. Fully-connected layers $fc6$, $fc7$ and $fc8$ have similar scale invariance, performance dropping by about 10% at a quarter of original scale. Therefore, deeper fully-connected layers are not more scale invariant. Convolutional layer *pool*5 is more invariant to scale, performance dropping only by about 4% at a quarter of original scale. However, it is less discriminative, with a significant performance gap for smaller scale changes. The *OxfordNet* features are learnt on input images of a fairly low resolution, which explains the overall robustness to large changes in scale.

Comparison of FV and CNN scale invariance. Scale invariance of *OxfordNet* $fc6$ and FV descriptors are compared in Figure 4.8 for the *Holidays* and *Graphics* data sets, following the protocol of previous experiment. None of the descriptors are fully scale invariant. This is expected as significant information is lost when downscaling the query images. With decreasing scale, a steeper drop in performance is observed for FVDM and FVDS compared to *OxfordNet*. Somewhat surprisingly, FVDoG also experiences a sharper drop in performance compared to CNN. The trends are consistent across data sets: the only difference is that the peak performance of FVDoG is higher than CNN on *Graphics*. The sharp drop in performance of FVDoG can be attributed to the failure of the interest point detector at small scales. CNN learnt on

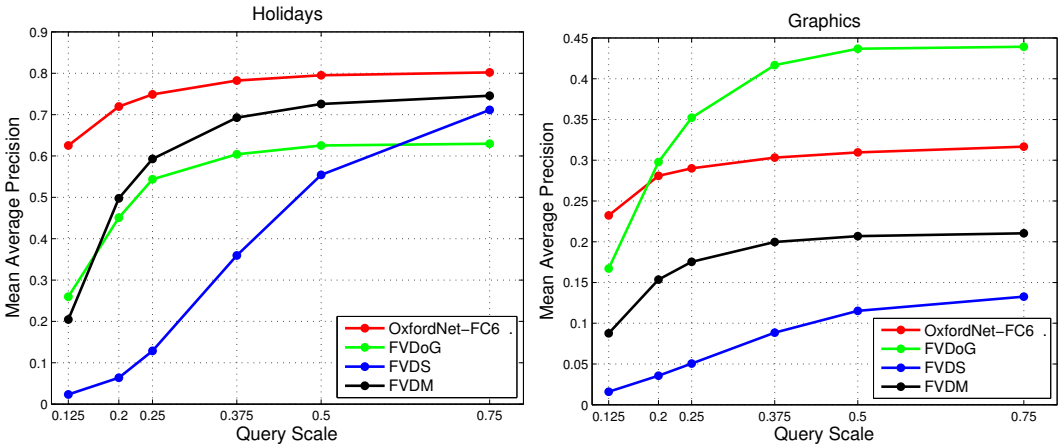


Figure 4.8: Comparison of *OxfordNet* $fc6$ and FV for scaled queries on the *Holidays* and *Graphics* data sets. We observe that *OxfordNet* features are more robust to scale changes compared to FVDoG, FVDS and FVDM, all of which experience a steeper drop in performance as query scale is decreased.

smaller images to begin with, and objects shown at different scales at training time, are sufficient for achieving more scale invariance than FVDoG. In comparison to the rotation experiments, it is interesting to note that FVDoG are more robust to rotation changes, while *OxfordNet* features are more robust to scale changes.

4.4 Summary and Discussion

In this chapter, we proposed a systematic and in-depth evaluation of FV and CNN pipelines for image retrieval. Our study has led to a comprehensive set of practical guidelines we believe can be useful to anyone seeking to implement state-of-the-art descriptors for image retrieval. Some of the recommendations are general good practices while others are more problem specific. Our study shows that CNN descriptors offer the best retrieval performance on average, but we also showed that unlike image classification, the supremacy of CNN over FV is not always clear in the case of image retrieval and strategies mixing both approaches are sometimes optimal.

Despite their good performances, we can point out two issues related to the CNN descriptors that need to be addressed:

- First, the lack of transformation invariance of the descriptors as shown in this study. This problem will be specifically addressed in Chapter 6.
- Second, the high dimensionality and scalar nature of the descriptors making descriptor matching inefficient. Accordingly, dimensionality reduction and binarization is addressed in Chapter 5.

Chapter 5

Hashing Global Descriptors

5.1 Introduction

While we showed in Chapter 4 that representations directly extracted from CNN are good global descriptors for image instance retrieval, we also pointed out that the high-dimensionality and the scalar nature of the descriptors is a barrier to fast descriptor matching. In this chapter, we therefore tackle the problem of hashing the descriptors to small binary codes for efficient matching with Hamming distances while retaining the good retrieval performance of the uncompressed descriptors. The very low bitrate range of 32-1024 bits is specifically targeted.

Our hashing pipeline consists of two parts:

1. An unsupervised dimensionality reduction approach using RBM to produce binary hashes at the target bitrate.

2. A fine-tuning step to improve the binary embedding functions generated by stacked RBM for which we propose both supervised and semi-supervised variants.

The first dimensionality reduction step applies a regularization to RBM specifically designed to optimize the distribution of generated binary hash codes. We refer to this particular form of regularization as RBM for Hashing or RBMH. RBMH is a batch-level regularization scheme aiming to improve very low bitrate hashes by encouraging efficient use of the latent subspace both within and across the hashes.

The second fine-tuning step is based on metric refinement with Siamese networks, an idea originally proposed by Bromley et al. in [12]. The method is based on the use of a labeled training set of matching and non-matching pairs of instances. Contrary to the pairwise contrastive loss function usually used at lower dimensionality such as in [21, 56], we show that critical improvements in the loss function of the Siamese network lead to improvements in retrieval results.

While able to significantly improve retrieval results, Siamese fine-tuning has the drawback of requiring an external labeled dataset of matching and non-matching pairs.

Therefore, we subsequently propose Unsupervised Triplet Hashing (UTH), a *fully unsupervised*, a rank learning scheme based on three weight sharing nets. The scheme is based on preserving the good retrieval performance of the uncompressed descriptors and thus does not require any external training labels. A thorough empirical evaluation conducted on multiple publicly available data sets using CNN descriptors shows that our proposed methods are able to significantly outperform state-of-the-art unsupervised schemes in the target bit range, particularly for extremely compact binary hashes in the 32-256 bits range.

5.2 Restricted Boltzmann Machine for Hashing

RBM can be used to produce compact binary codes from higher dimensionality descriptors. In [131], RBM are trained to hash bag-of-word features from text corpus, and Hamming distances in the latent space are subsequently used for text documents similarity search. In this section, we proposed to use stacked RBM for hashing the high dimensional global descriptors to small binary codes.

5.2.1 Method

Proper regularization is key during the training of RBM. For classification, discriminative performance is improved by constraining binary latent units to be rarely activated, or sparse [113]. This is desirable for classification tasks as it improves separability of the data but this is not necessarily ideal for hashing where efficient use of the limited latent subspace is key. The proposed RBMH method is a batch-level regularization scheme (unlike sparsity schemes which are usually instance level). It achieves efficient space use by controlling sparsity in a way to maximize the entropy not only within every hash but also between the same bit of different hashes. This effectively encourages:

1. half the hash bits to be active for a given image;

2. each hash bit to be equiprobable across images.

We first discuss how high batch-level entropy is encouraged in the RBMH framework. Next, we present how deep RBMH are constructed by stacking multiple RBMH. An overview diagram of the method is available in Figure 5.1

Batch-level Regularization Preserving the notations for RBM introduced in Chapter 2, for two successive layers $l-1$ and l, and a batch B of input instances z_α^{l-1} with corresponding latent representations z_α^l, we define the regularization term adapted

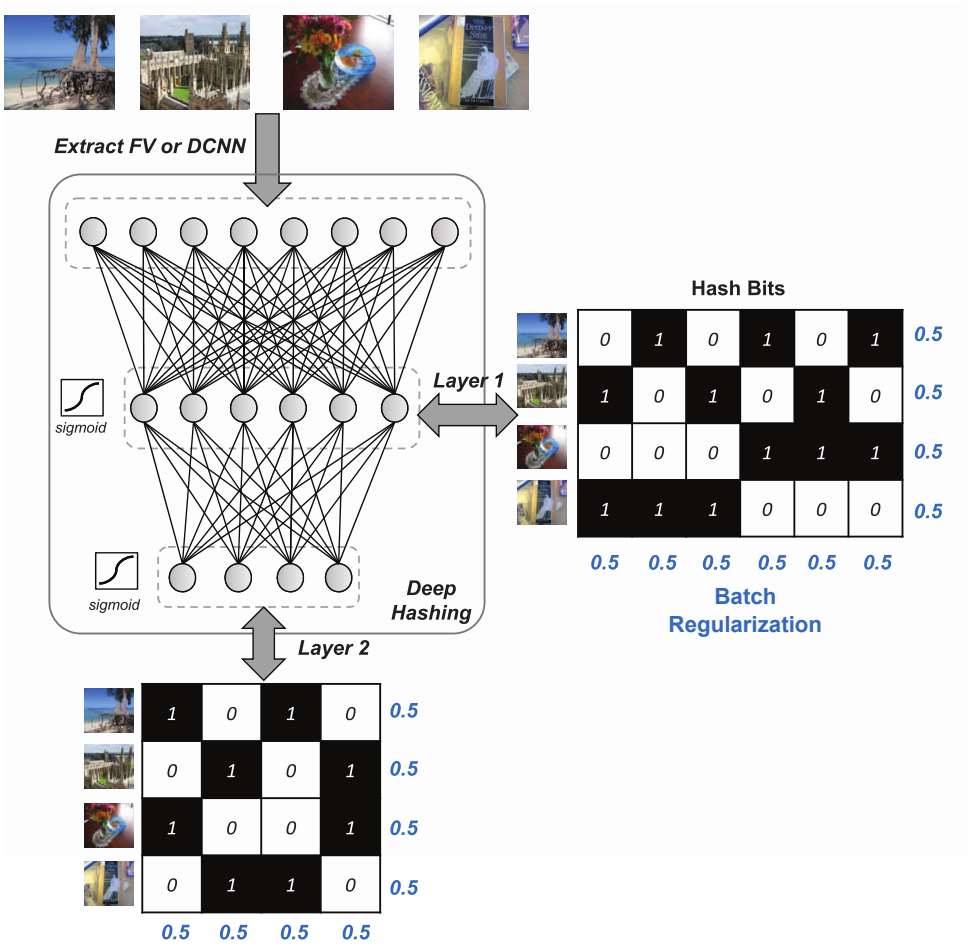
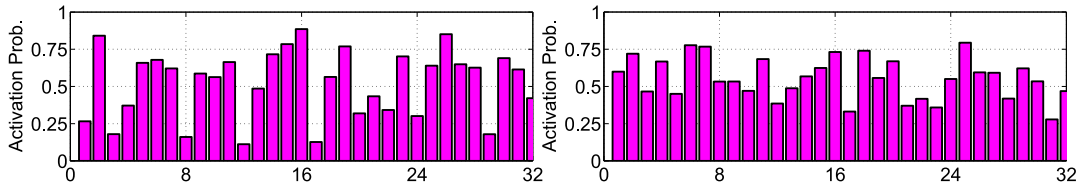


Figure 5.1: In the RBM for Hashing framework, Stacked RBM are trained to hash global descriptors. The latent representations are regularized for the bits activation probability P to be equal to 0.5 both across bits of individual hashes and across images for the same latent unit.



(a) Hinton RBM (b) RBMH

Figure 5.2: Activation probabilities of hash bits with RBMH and the RBM proposed by Nair&Hinton [113]. Statistics of 32 bits binary hashes are computed on *Holidays* data set. The mean values of both activation histograms are close to 0.5, while the RBMH histogram is distributed more evenly across units compared to the standard RBM one (i.e. standard deviation is smaller).

from the fine-grained regularization proposed in [44]:

$$h(B) = -\sum_{z_\alpha^l \in B} \sum_j t_{j\alpha}^l \log z_{j\alpha}^l + (1 - t_{j\alpha}^l) \log(1 - z_{j\alpha}^l). \tag{5.1}$$

where $t_{j\alpha}^l$ are the target activations $z_{j\alpha}^l$ for each sample $z_{j\alpha}^{l-1}$. Unlike in [44], we choose the $t_{j\alpha}^l$ such that each $\{t_{j\alpha}^l\}_j$ for fixed α and each $\{t_{j\alpha}^l\}_\alpha$ for fixed j is distributed according to $U(0,1)$ effectively maximizing entropy. The uniform distribution is suitable for hashing high-dimensional vectors because the regularizer encourages each latent unit to be active with a mean of 0.5, while avoiding activation saturation.

The overall objective function for the RBMH becomes:

$$\underset{\{W^l, b^l, c^{l-1}\}}{\arg\min} -\sum_\alpha \log\Big(\sum_{z_\alpha^l \in E_\alpha} P(z_\alpha^{l-1}, z_\alpha^l) + \lambda h(B) \Big), \tag{5.2}$$

where λ is a regularization constant. It is optimized through batch gradient descent using the contrastive divergence algorithm as described in Chapter 2.

Figure 5.2 shows the activation probabilities for 32-bit hashes provided by RBMH and the RBM proposed in [113]. The mean probability of activation is nearly 0.5 in both cases, but much more evenly distributed across bits with RBMH.

Stacked RBMH The set of raw image representations lie in a complex manifold in a very high-dimensional feature space. Deeper networks have the potential to discover more complex nonlinear hash functions and improve image instance retrieval performance. Following [62], we stack multiple RBMH, greedily training one layer at a time to create a deep network with several layers.

Each layer models the activation distribution of the previous layer and captures higher order correlations between those units. For the hashing problem, we are interested in low-rate points of 64, 256 and 1024 bits. We progressively decrease the dimensionality of latent layers by a factor of 2^n per layer, where n is a tuneable pa-

rameter. For our final models, n is empirically selected for each layer resulting in variable network depth.

Output Binarization Binary hashes are desirable for fast matching with Hamming distances. Therefore, sigmoid activation functions are used for the RBMH. In addition, the output of the topmost RBMH (layer L) is binarized around 0.5:

$$z_j^L = \begin{cases} 1, & \text{if } \frac{1}{1+\exp(-w_j^L z^{l-1} - b_j)} > 0.5 \\ 0, & \text{otherwise.} \end{cases} \tag{5.3}$$

In the next section, we evaluate the performance of the proposed SRBMH scheme.

5.2.2 Evaluation Framework

Global Descriptors. SIFT [98] features obtained from Difference-of-Gaussian (DoG) interest points are used for FV. PCA is used to reduce dimensionality of the SIFT descriptor from 128 to 64 dimensions, which has shown to improve performance [73]. We use a Gaussian Mixture Model (GMM) with 128 centroids, resulting in 8192 dimensions each for first and second order statistics. Only the first-order statistics are retained in the global descriptor representation, as second-order statistics only results in a small improvement in performance [91]. The FV is L_2-normalized to unit-norm, after signed power normalization (referred to as FV from here-on). DCNN features are extracted using the open-source software Caffe [77] with *AlexNet* reference model proposed for 2012 ImageNet classification task [79]. DCNN descriptors are extracted from the first fully-connected layer $fc6$ which has been shown in Chapter 4 to yield performant descriptors for instance retrieval. We refer to this 4096-dimensional descriptor as the CNN descriptor from here-on.

SRBMH Training. For the hashing problem, we are interested in low-rate points of 64, 256 and 1024 bits. SRBMH are trained greedily in a layer by layer fashion, i.e. each new RBMH is trained on the top of the previous one without modifying parameters of previous RBMH. A 150,000 images random subset of ImageNet is used as training data. The set is chosen for its variety and genericness, and because it has no intersection with images used in the retrieval experiments. The batch size is 100 for all experiments. Learning rate is set to 0.001 for the weight and bias parameters, momentum to 0.9. Training is run for a maximum of 30 epochs. For each rate point, different models are considered. For our final models, n is empirically selected for each layer resulting in variable network depth. Each target setting requires several hours to train.

Baselines. Four state-of-the-art hashing schemes are considered as baselines: Locality Sensitive Hashing (LSH), Iterative Quantization (ITQ), Bilinear Projection Binary Codes (BPBC) and Product Quantization (PQ). LSH is based on random unit-norm projections of the raw descriptors, followed by signed binarization [163]. ITQ applies signed binarization after two transforms of raw descriptors: first the PCA, followed by a rotation [47]. Unlike ITQ, BPBC applies bilinear random projections, which require far less memory to transform the data [45].

For FV, we consider blocks of dimensions $D = 64, 256$ and 1024, and train $K = 256$ centroids for each block, resulting in $b = 128, 256$ and 64 bit descriptors respectively. For CNN, we consider blocks of dimensions $D = 32, 128$ and 512, with $K = 256$ centroids, resulting in the same bitrates. L_2 norm is used for PQ and uncompressed descriptors, while Hamming distances are used for all binary hashing schemes.

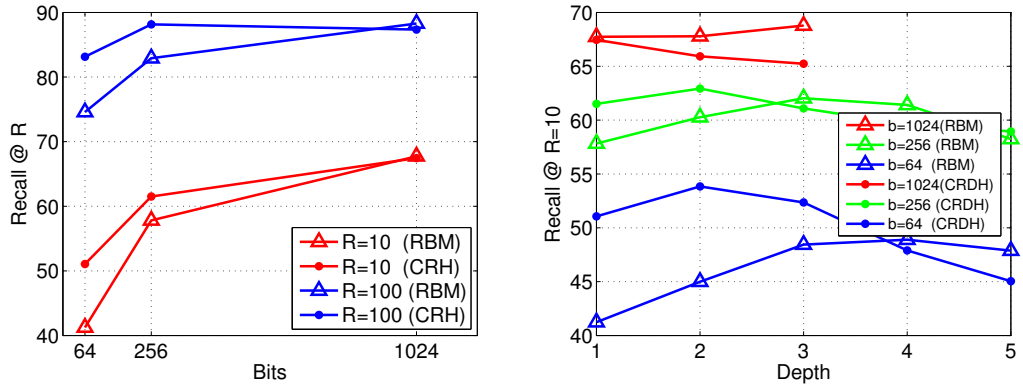
Data Sets. The *Holidays* and *UKBench* data sets are used for small scale image instance retrieval experiments. For large scale experiments, the two data sets are combined with the one million *MIRFlickr* distractor images.

Evaluation Metrics. In most image retrieval use cases, it is important for the relevant image to be present in the first step of the pipeline, matching global descriptors, so that a Geometric Consistency Check (GCC) [34] step can subsequently find it. However, the GCC step is computationally complex and can only be performed on a small number of images. As a result, it is important for the relevant image to be present in a short list, so that the GCC step can find it. Hence, recall is presented at typical operating points of R = {10,100} and R = 1000 for small and large experiments respectively. For *UKBench* small scale experiments, 4× Recall @ $R = 4$ is provided, to be consistent with the literature.

5.2.3 Experimental Results

Impact of Batch-Level Entropy Objective. In Figure 5.3(a), we show the effect of applying the proposed regularization on a single layer RBM 8192-b, for $b = 64, 256, 1024$. Hashing regularization significantly improves performance, ∼10% absolute recall @ $R = 10$ at low-rate point $b = 64$. The performance gap increases as rate decreases. This is intuitive as the regularization encourages a more efficient use of the latent space.

Impact of Depth. In Figure 5.3(b), we plot recall @ $R = 10$ on *Holidays* as depth is increased for a given rate point b. For $b = 1024$, we consider architectures $8192 - 1024$, $8192 - 4096 - 1024$ and $8192 - 4096 - 2048 - 1024$ corresponding to depth $1, 2$ and 3 respectively. For rate points $b = 64$ and 256, similar configurations of varying depth are chosen. We observe that, with no regularization, recall improves as depth is increased



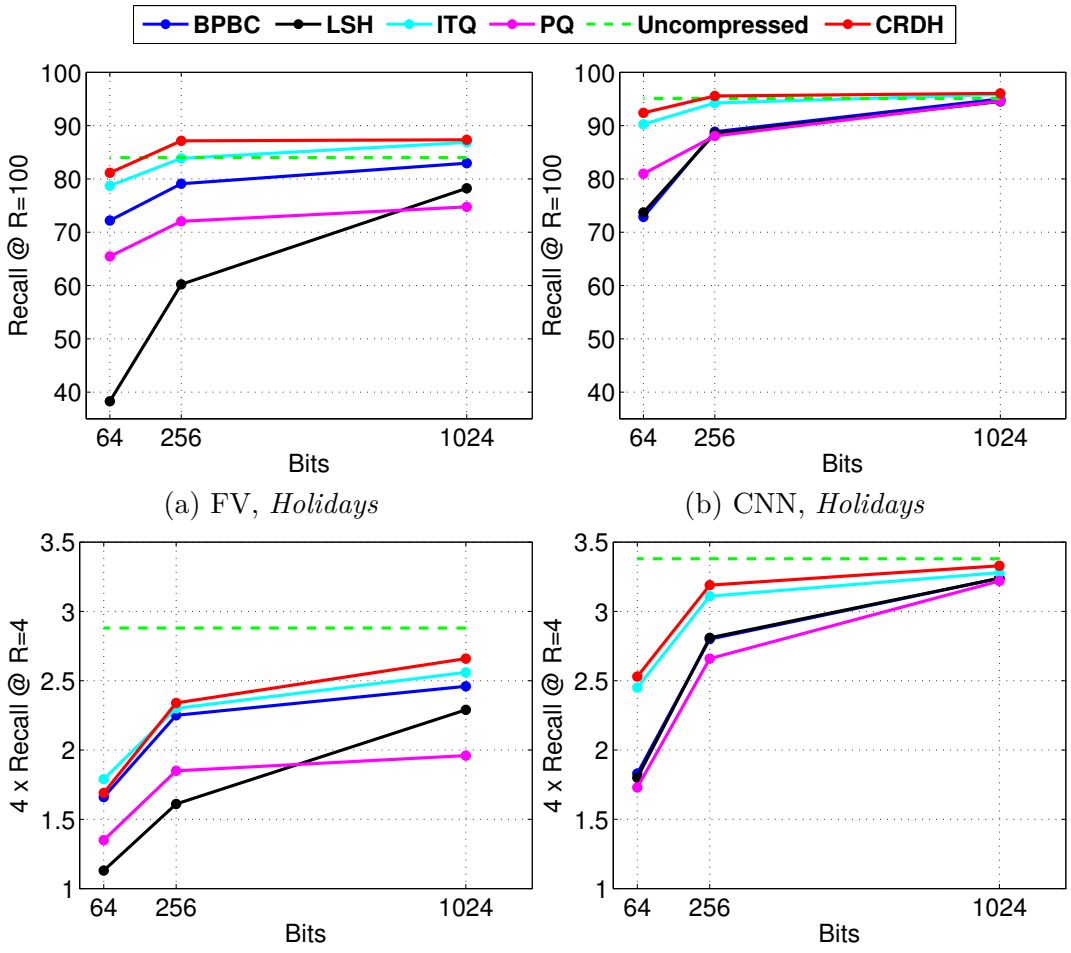
(a) Batch entropy objective (b) Depth

Figure 5.3: Hashing *Holidays* FV. (a) RBMH regularization significantly improves performance for single layer models 8192-b as b is decreased. (b) Recall improves as depth is increased for lower rate points $b = 64$ and $b = 256$. With RBMH regularization, same or better recall can be achieved at lower depth.

for $b = 256$ and $b = 64$, with optimal depth of three and four respectively, beyond which performance drops. At higher rates of $b = 1024$ and beyond, increasing depth does not improve performance. For hashing, a sweet spot in performance for the depth parameter is observed for each rate point, as deeper networks can cause performance to drop due to loss of information over the layers. Similar trends are obtained for recall @ $R = 100$. Importantly, we observe that with the proposed regularization, we can achieve the same performance with lower depth at each rate point. This is critical, as the lower the depth, the faster the hash generation, and the lower the memory requirements.

Comparison of FV-RBMH and CNN-RBMH. At a given rate point, CNN-RBMH outperforms FV-RBMH for all data sets, as shown in Figure 5.4. At low rates, CNN-RBMH improves performance by more than 10% on the small data sets, possibly because CNN features are able to capture more complex low level features and are a lower starting dimensionality compared to FV.

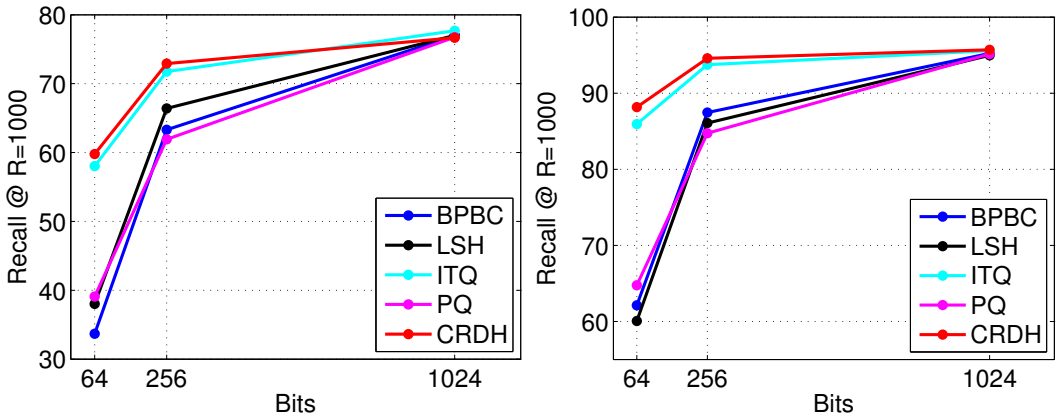
Comparison to Uncompressed Descriptors. The performance of RBMH is compared to the uncompressed descriptor in Figure 5.4. At 256 bits for CNN hashes, we only observe a marginal drop (a few percent) compared to the uncompressed descriptor for retrieval on all data sets. For FV, uncompressed descriptor performance is matched at 1024 bits. The instance retrieval hashing problem becomes increasingly difficult as we move towards a 64-bit hash, with performance dropping steeply.



(a) FV, *Holidays* (b) CNN, *Holidays*

(c) FV, *UKBench* (d) CNN, *UKBench*

Figure 5.4: Small-scale retrieval results for different compression schemes. Proposed RBMH outperforms other schemes by a significant margin.



(a) CNN, *Holidays*+1M (b) CNN, *UKBench*+1M

Figure 5.5: Large-scale retrieval results (with 1M distractor images, at Recall@R=1000) for different compression schemes. *RBMH* outperforms other schemes at most rate points and data sets.

Comparison to State-of-the-Art. Small scale retrieval results are shown in Figure 5.4. One can see that the proposed RBMH outperforms state-of-the-art at most rates on all data sets, for both CNN and FV features. There is 2.4% improvement in absolute Recall @ $R = 100$ at $b = 64$ bits compared to the second performing scheme ITQ on *Holidays* for FV.

The performance ordering of other schemes depends on the bitrate and type of feature, while RBMH is consistent across data sets. Compared to ITQ scheme which applies a single PCA transform, each output bit for RBMH is generated by a series of projections. The PQ scheme performs poorly at the low rates in consideration, as large blocks of the global descriptor are quantized with a small number of centroids, as previously observed in [45]. LSH performs poorly at low rates, but catches up given enough bits. Consistent trends are observed for the large-scale retrieval in Figure 5.5.

Section Summary. In this section, we propose RBM for Hashing (RBMH) to maximize the discriminability of hash codes. Specifically, RBMH is formulated by adding a batch-level entropy objective into the Restricted Boltzmann Machines (RBM) based model, where each hash bit corresponds to a neural unit in latent layer. Unlike existing hashing approaches [13], RBMH batch-level entropy constraint balances both the variance of hash codes per image (i.e., with uniform sparsity 0.5) and the variance of each hash bit across images. This results in effective coding especially at extremely low rates (e.g., 64 bits). For aggressive dimensionality reduction, RBMH is extended into deep network structure (i.e., SRBMH) by stacking multiple RBMH. Through a thorough empirical evaluation on popular benchmark data sets with different image representations (FV and CNN), we show that RBMH outperforms state-of-the-art unsupervised

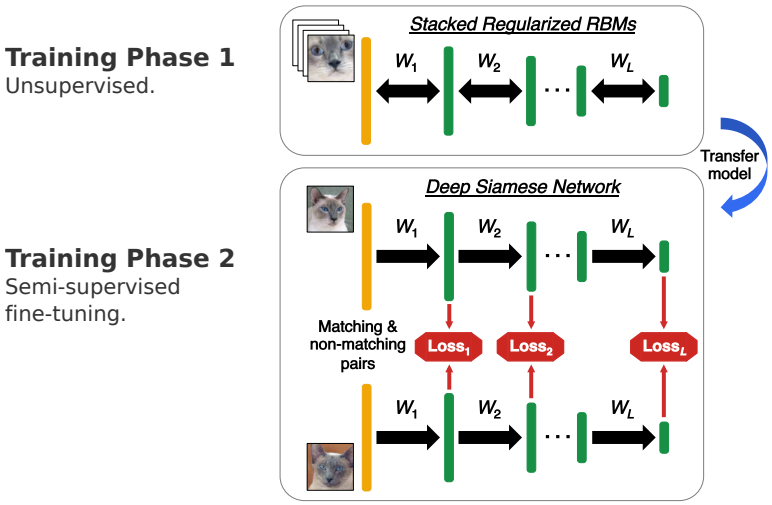


Figure 5.6: The proposed method for learning binary embedding functions involves an unsupervised pre-training stage followed by a weakly-supervised fine-tuning stage. In the first stage, RBM are trained in a layer-wise manner and stacked into a deep network (Section 5.2). In the second stage, first stage parameters are loaded into a Siamese network and subsequently fine-tuned using matching and non-matching pairs data.

descriptor compression methods at low bit-rates (256 down to 64 bits).

The method proposed in this section is unsupervised, i.e., no external labels are used in the training step. Next, we discuss how a semi-supervised fine-tuning approach can be used for further improving hashing performance.

5.3 Dual-margin Siamese Fine-tuning

In the previous section, we showed that global descriptors can be hashed into small binary descriptors using RBMH while retaining good properties for retrieval. However, RBMH is specifically optimized for compression and there is no built-in mechanism to ensure that the good metric properties of the original descriptors are preserved.

In this section, we propose a weakly-supervised method for improving the local structure of binary embedding functions using weight-sharing networks and an additional labeled dataset of matching and non-matching pairs.

5.3.1 Method

The fine-tuning is performed with a learning architecture known as Siamese networks first introduced in [56]. The principle was later successfully applied to deep architectures for face identification [21] and shown to produce representations robust to various transformations in the input space [56]. The use of Siamese architectures in

the context of image retrieval from CNN features was recently suggested as a possible improvement over the state-of-the-art [9].

A Siamese network is a weakly-supervised scheme for learning a similarity measure from pairs of data instances labeled as matching or non-matching. In our adaptation of the concept, the weights of the trained RBM network are fine-tuned by learning a similarity measure at every intermediate layer in addition to the target space. Given a pair of data $(\mathbf{z}_\alpha^0, \mathbf{z}_\beta^0)$, a contrastive loss $\mathcal{D}_l$ is defined for every layer l and the error is back propagated though gradient descent. Back propagation for the losses of individual layers ($l = 1..L$) is performed at the same time. Applying the loss function proposed by Handsell et al. in [56] yields:

$$\mathcal{D}_l(\mathbf{z}_\alpha^0, \mathbf{z}_\beta^0) = y\|\mathbf{z}_\alpha^l - \mathbf{z}_\beta^l\|_2^2 + (1-y)\max(m - \|\mathbf{z}_\alpha^l - \mathbf{z}_\beta^l\|_2^2, 0) \tag{5.4}$$

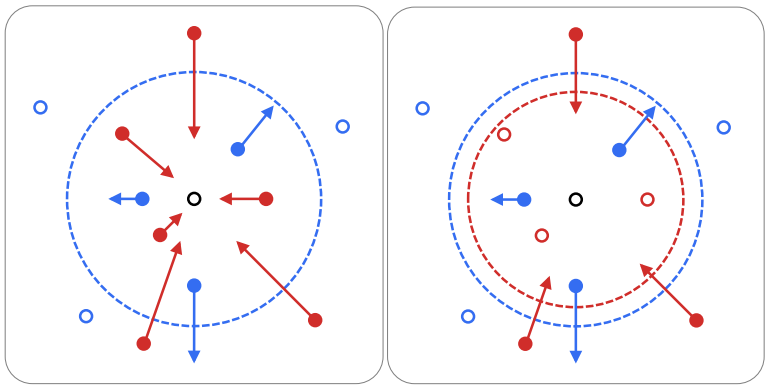
where $y = 1$ if $(\mathbf{z}_\alpha^0, \mathbf{z}_\beta^0)$ is a matching pair or $y = 0$ otherwise, and $m > 0$ is a margin parameter affecting non-matching pairs. As shown in Figure 5.7(a), the effect is to apply a contractive force between elements of any matching pairs and a repulsive force between elements of non-matching pairs which element-wise distance is shorter than $\sqrt{m}$.

However, experiment results in Figure 5.9 show that the loss function (5.4) causes a quick drop in retrieval results. Results with non-matching pairs alone suggest that the handling of matching pairs is responsible for the drop. The indefinite contraction of matching pairs well beyond what is necessary to distinguish them from non-matching elements is a damaging behaviour, especially in a fine-tuning context since the network is first globally optimized with a different objective. Figure 5.8 shows that any two elements, even matching, are always far apart in high dimension. Note that this phenomenon which occurs at the target bitrate of the hashes (e.g. 64 bits and higher) was not originally an issue at the much lower-dimensionality latent spaces considered in [56]

As a solution, we propose a double-margin loss with an additional parameter affecting matching pairs:

$$\begin{aligned}\mathcal{D}_l(\mathbf{z}_\alpha^0, \mathbf{z}_\beta^0) =& y\max(\|\mathbf{z}_\alpha^l - \mathbf{z}_\beta^l\|_2^2 - m_1, 0) \\ &+ (1-y)\max(m_2 - \|\mathbf{z}_\alpha^l - \mathbf{z}_\beta^l\|_2^2, 0)\end{aligned} \tag{5.5}$$

As shown in Figure 5.7(b), the new loss can thus be interpreted as learning "local large-margin classifiers" (if $m_1 \leq m_2$) to distinguish between matching and non-matching elements. In practice, we found that the two margin parameters can be set equal ($m_1 = m_2 = m$) and tuned automatically from the statistical distribution of the sampled matching and non-matching pairs (Figure 5.8).



(a) (b)

Figure 5.7: A sample point (black dot) with corresponding matching (red dots) and non-matching (blue dots) samples. The contrastive divergence loss used for fine-tuning can be interpreted as applying attractive forces between matching elements (red arrows) and repulsive forces between non-matching elements (blue arrows). (a) The loss function (Equation 5.4) proposed in [56] with a single margin parameter for non-matching pairs (blue circle). Matching elements are subject to attractive forces regardless of whether they are already close enough from each other which adversely affects fine-tuning. (b) The proposed loss function (Equation 5.5) with an additional margin parameter affecting matching pairs reciprocally (red circle).

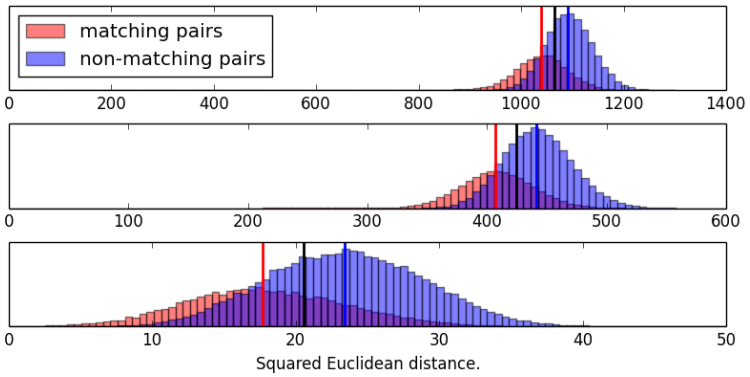


Figure 5.8: Histograms of squared Euclidean distances for 20,000 matching pairs and corresponding 40,000 non-matching pairs for an $8192 - 4096(\text{top}) - 2048(\text{middle}) - 64(\text{bottom})$ stacked RBM network. Image pairs are sampled from *Yandex* data set. The red and blue vertical lines indicate the median values for the matching and non-matching pairs respectively. The Siamese loss shared margin value m is systematically set to be the mean of the two values (black vertical lines).

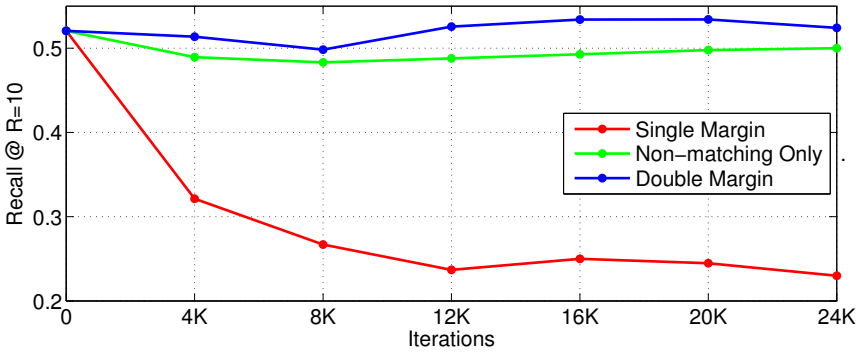


Figure 5.9: Recall @ R=10 on the *Holidays* data set (Refer to Appendix A for detailed description of data sets used for retrieval experiments). over several iterations of Siamese fine-tuning. The recall rate quickly collapses when using the single margin loss function suggested in [56] while performance is better retained when only non-matching pairs are passed. The double-margin loss solves the problem. The network is a stacked RBM ($8192 - 4096 - 2048 - 64$) trained with FV descriptors computed from a 150K random subset of ImageNet data set. Image pairs are sampled from the *Yandex* data set. For every matching pair, a random non-matching element is chosen from the data set to form two non-matching pairs. There are 33 matching pairs and 66 corresponding non-matching pair with every iteration.

5.3.2 Evaluation framework

First, SRBM are trained as in Section 5.2, using the same FV descriptors extraction strategy. Then, a data set of matching and non-matching image pairs is used for fine-tuning. The 200K matching pairs data set is provided by Yandex in their recent work [9]. It consists in images of famous landmarks collected by querying the names of the most viewed Wikipedia landmark pages in Yandex search engine. Data set visualization reveals that most images depict buildings. For each matching pair, a random image is picked to generate two non-matching pairs.

Holidays, *UKBench* and *Oxbuild* data sets are used for small scale retrieval experiments. For large scale experiments, *Holidays* and *UKBench* databases are combined with the one million *MIRFlickr* distractor images. Note the *Yandex* based training set is independent from the data used for evaluation as the authors removed Oxford-related queries, and Holidays near-duplicate images.

5.3.3 Experimental Results

Detailed retrieval results of a three layer model before and after Siamese fine-tuning are provided in Table 5.1. The results show consistent improvements on all data sets and bit-rates, with an average improvement of 2.78% (up to 6.24%). The difference is more significant at higher recall rates with an average of 2.43% @ R=10 compared to 3.13% @ R=100. They are however quite comparable when relative improvement rate

Data set	Layer	Recall @ R=10			Recall @ R=100		
		bef.	aft.	diff.	bef.	aft.	diff.
Holidays	4096	70.83	73.67	**2.84**	89.92	91.40	**1.48**
	2048	67.74	71.12	**3.38**	88.77	92.04	**3.27**
	64	52.06	53.04	**0.98**	80.38	83.91	**3.53**
UKBench	4096	79.22	82.22	**3.00**	92.04	93.73	**1.69**
	2048	75.62	79.37	**3.75**	90.79	92.82	**2.03**
	64	47.94	49.25	**1.31**	73.02	73.94	**0.92**
Oxbuild	4096	19.38	21.73	**2.35**	41.09	45.19	**4.10**
	2048	14.32	17.23	**2.91**	36.03	41.03	**5.00**
	64	10.69	12.01	**1.32**	23.75	29.99	**6.24**

Table 5.1: Small scale retrieval results before and after Siamese fine-tuning, with corresponding improvement. A SRBM model ($8192 - 4096 - 2048 - 64$) is trained on ImageNet FV descriptors, then fine-tuned on *Yandex*. For each model, the three layers retrieval performance is evaluated on *Holidays*, *UKBench* and *Oxbuild* data sets, before and after fine-tuning. Siamese fine-tuning consistently improves retrieval performance at all three layers.

is considered: 7.46% @ R=10 and 7.24% @ R=100 relatively.

We notice differences across test sets with improvements on the *Oxford* set being more pronounced. The fine-tuning data set *Yandex*, like retrieval data set *Oxbuild*, mostly depicts landmark structures. The proximity between the two data sets may explain the higher performance improvement on *Oxbuild*. The systematic improvements on all data sets are nevertheless evidence of the high transferability of both unsupervised training and semi-supervised fine-tuning.

Data set	Bitrate	Recall @ R=1000		
		bef.	aft.	diff.
Holidays + 1M	1024	52.49	56.48	**3.99**
	256	46.07	49.60	**3.53**
	64	30.63	31.87	**1.24**
UKBench + 1M	1024	85.66	86.41	**0.75**
	256	78.70	80.74	**2.04**
	64	61.13	62.74	**1.61**

Table 5.2: Large scale retrieval results before and after Siamese fine-tuning, with corresponding differences. First, three SRBM are trained to respectively hash high-dimentional FV descriptors to bitrates $b = 1024$, 256 and 64. They are then fine-tuned with Siamese networks. Fine-tuning improves retrieval performance at all bitrates on both data sets.

Similar trends are observed with large scale retrieval experiments in Table 5.2, where Siamese fine-tuning improves performance at all bitrates and on all data sets.

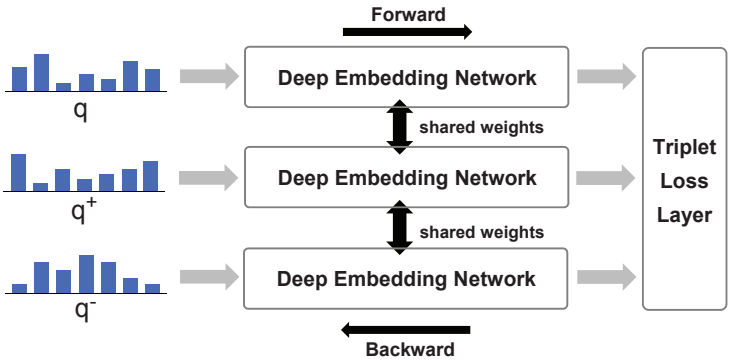


Figure 5.10: Illustration of the proposed unsupervised triplet hashing scheme.

5.3.4 Conclusion

Results show that a Siamese network can be used to further improve hashing performance at low rates. Deep Siamese fine tuning, to be effective, requires critical changes to the loss function in the fine-tuning step. One disadvantage of this approach is the additional data set of matching image pairs required for fine-tuning performance. In the next section, we propose a fine-tuning scheme, which is completely unsupervised, while achieving significant gains in performance.

5.4 Unsupervised Triplet Fine-Tuning

In this section, we propose Unsupervised Triplet Hashing (UTH), an extension of hashing with SRBM which, unlike Siamese fine-tuning, uses a *fully unsupervised* approach for refining the binary embedding functions. The method uses triplet networks, a rank learning scheme based on weight sharing nets (Figure 5.10). Instead of relying on external datasets of matching and non-matching pairs, the idea is to retain the already good metric properties of the original high-dimensional descriptor space during hashing.

Unlike other approaches using triplet learning networks [66, 155, 82], the proposed approach is *fully-unsupervised* and does not require extra labeled data for the triplets. Through a thorough empirical evaluation, we show that UTH is able to generate extremely compact descriptors with good retrieval performances.

5.4.1 Method

Triplet Network. Triplet networks can learn the ranking information provided through data triplets. A triplet (q, q^+, q^-) contains a query image descriptor q, a pos-

itive image descriptor q^+ and a negative image descriptor q^-, query q is more similar (closer) to positive image q^+ than to negative image q^-. We learn a binary embedding function $p : \mathbb{R}^n \mapsto 2^m$, typically $n >> m$, such that $d(p(q), p(q^+)) < d(p(q), p(q^-))$. Accordingly, we define a triplet ranking loss, $l(q, q^+, q^-) = max\{0, g + d(p(q), p(q^+)) - d(p(q), p(q^-))\}$, where g is a positive margin parameter. By normalising the two distances with softmax (denoted d') and setting $g = 1$, we can rewrite the triplet loss function as,

$$l(q, q^+, q^-) = max\{0, 1 + d'(p(q), p(q^+)) - d'(p(q), p(q^-))\} \tag{5.6}$$

The idea of using weight sharing networks for model fine-tuning is not novel and previous work can be found for image classification and semantic retrieval [66, 155, 82].

Unsupervised Triplet Sampling. Unlike previously proposed approaches, proposed triplet learning is fully unsupervised thus does not require labelled data. Based on the observation that the original uncompressed descriptors already provide good retrieval performance (Figure 5.11), we construct triplets according to the Euclidean distance in the original space, i.e. such that $\|q - q^+\|_2 < \|q - q^-\|_2$. Accordingly, our global objective function is,

$$min \quad l(q, q^+, q^-) \quad s.t. \ \|q - q^+\|_2 < \|q - q^-\|_2 \tag{5.7}$$

A possible triplet sampling strategy is to randomly sample three images, arbitrarily choose q and label the two others as q^+ and q^- according to their relative distance to q. The space of all possible such triplets is very vast. For instance retrieval, we are mostly interested in correctly discriminating ranks for the top end of the retrieval list, suggesting a more targeted sampling strategy may improve recall.

We propose a threshold sampling method to generate informative triplets. Given a training set containing M images, we build a look-up table with M buckets offline. For the m-th bucket, we compute distances between the m-th image descriptor and the rest, rank the distances in descending order and store the distances and their associated image IDs in that bucket. To sample a triplet, we first randomly draw an image q from the training set, then we sample a positive image q^+ from the bucket associated with q, with the constraint that $\|q - q^+\|_2$ is close to a pre-defined threshold T_p. Similarly, we sample a negative image q^- from the same bucket such that $\|q - q^-\|_2 \to T_n$. We note that $T_p < T_n$.

5.4.2 Evaluation Framework

Training. As illustrated in Figure 5.12(a), pre-training is crucial to triplet network performance. Best performance is achieved with SRBM model based weight initial-

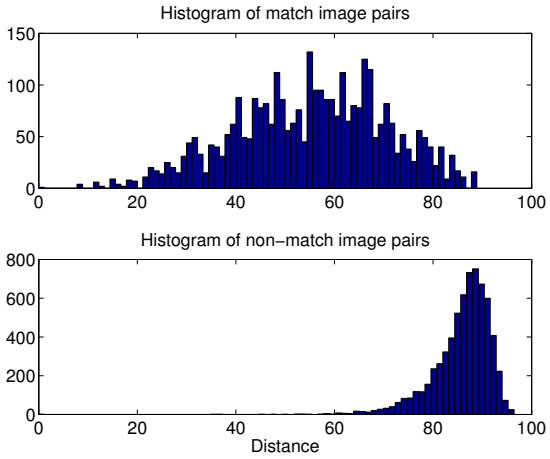


Figure 5.11: The distribution of squared Euclidean distances for CNN descriptors from 3,300 matching image pairs and 6,600 non-matching image pairs reveals that the original uncompressed descriptors already contain good ranking information.

ization (UTH_SRBM), as random weights initialization (UTH_UniW) converges to significantly worse local minima (17.4% vs. 51.6% mAP). In following experiments, triplet networks are initialized using parameters from pre-trained SRBM models.

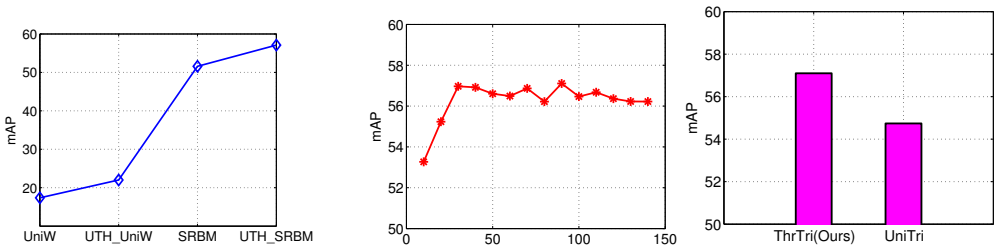
SRBM used for UTH initialization are trained on descriptors of a 150K image random subset of ImageNet, excluding category labels. Image descriptors are extracted from *OxfordNet* first fully-connected layer $fc6$. The following SRBM architectures are used: $4K-2K-256$, $4K-2K-128$, $4K-1K-64$ and $4K-2K-32$ for producing $256, 128, 64$, and 32 bits hashes respectively.

Figure 5.12(b) shows that the triplet loss optimization converges quickly after a few epochs. The models are fine-tuned for a maximum of 150 epochs, with learning rate set in the range $[0.0005, 0.01]$ and momentum to 0.9.

The impact of triplet sampling strategies is illustrated in Figure 5.12(c). Proposed threshold triplet sampling (ThrTri) significantly outperforms uniform triplet sampling (UniTri), with mAP of respectively 57.1% and 54.7% on *Holidays* data set. Shown capable of sampling more informative triplets, the threshold triplet sampling strategy is used in the following experiments.

Baselines. The proposed UTH scheme is compared against the following unsupervised schemes: (1) Shift-invariant Kernel LSH (SKLSH) [126], (2) PCAHash [46], (3) Spectral Hashing (SH) [156], (4) LSH, (5) ITQ, (6) BPBC, and (7) SRBM. The uncompressed *OxfordNet* descriptors (4096-dimensional original floating point representation) are also compared against. L_2 norm is used for uncompressed descriptors, while Hamming distance is used for all binary hashing schemes.

For image instance retrieval experiments, recall@10 and recall@100 results are pre-



(a) Network weights initilization (b) Training epochs (c) Triplet sampling

Figure 5.12: (a) Performance comparisons of pre-trained network weights with SRBM vs. random generated unit-norm network weights, with or without fine-tuning (UTH); (b) Impact of the number of fine-tuning epochs on retrieval performance; (c) Performance comparisons of the proposed threshold triplet sampling (ThiTri) against uniformly sampled triplets (UniTri). All results are reported in terms of mAP on *Holidays* data set at 64 bits.

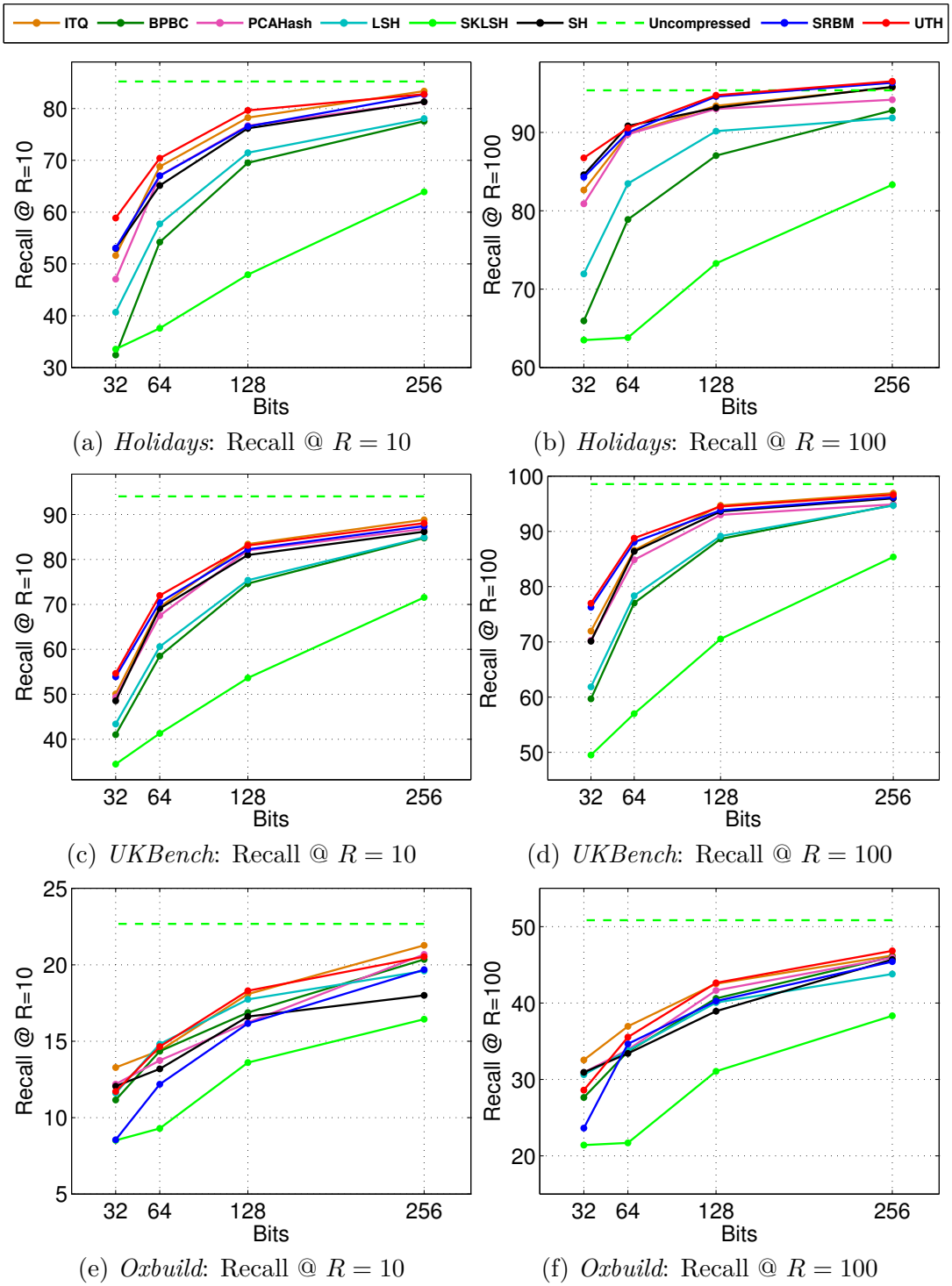
sented at varying bitrates. Data sets *Holidays*, *UKBench* and *Oxbuild* are used in small scale experiments. Large-scale experiments results are presented for *Holidays* and *UKBench* data sets, combined with the one million *MIRFlickr* distractor data set. Note retrieval data sets are independent from the training set described above.

5.4.3 Experimental Results

Small-scale Retrieval Experiments. Figure 5.13 shows that the proposed UTH significantly outperforms SRBM, especially on *Holidays* and *Oxbuild* at low bitrates. Besides, ITQ, PCAHash and SH obtain higher accuracy than LSH and BPBC. SKLSH performs the worst. The ordering of schemes is largely consistent across recall@10 and recall@100 results.

UTH outperforms most schemes across various bitrates on different data sets. On both *Holidays* and *UKBench*, UTH significantly outperforms the baseline schemes at extremely low bitrates (e.g., 32), for example, +5% than ITQ in terms of recall@100 on *UKBench* at 32 bits. The baseline schemes catch up in performance as bitrate increases, i.e., the improvements of UTH over the other schemes become smaller at 256 bits (expect SKLSH). We note that UTH performs slightly worse than ITQ, PCAHash and SH on *Oxbuild*.

Note there is a significant gap between the uncompressed descriptor and all the compression schemes at extremely low bitrates on all data sets, while the performance of compression schemes approach to uncompressed descriptors as bitrate increases to 256 bits. For instance, compared to uncompressed descriptors, there is a 26% drop in recall@10 at 32 bits on *Holidays* for UTH. The drop is largely reduced to 2% at 256 bits.



(a) *Holidays*: Recall @ $R = 10$

(b) *Holidays*: Recall @ $R = 100$

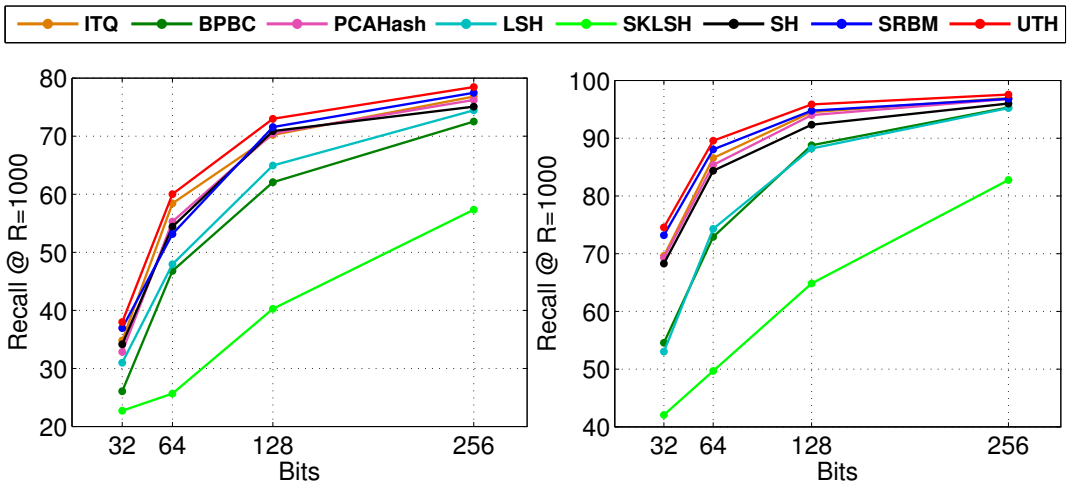
(c) *UKBench*: Recall @ $R = 10$

(d) *UKBench*: Recall @ $R = 100$

(e) *Oxbuild*: Recall @ $R = 10$

(f) *Oxbuild*: Recall @ $R = 100$

Figure 5.13: Retrieval results in terms of Recall @ $R = 10$, Recall @ $R = 100$ for different compression schemes on *Holidays*, *UKBench* and *Oxbuild* data sets. The proposed UTH outperforms most schemes across various bitrates on different data sets.



(a) *Holidays*+1M: Recall @ $R = 1000$ (b) *UKBench*+1M: Recall @ $R = 1000$

Figure 5.14: Large scale retrieval results (with one million distractor images) for different compression schemes. The proposed UTH outperforms other schemes at all bitrates on both Holidays+1M and UKBench+1M.

Large-scale Retrieval Experiments. Figure 5.14 shows that UTH outperforms all other schemes at all bitrates. Large scale experiments results are consistent with the trends observed on small scale experiments.

5.4.4 Conclusion

In this section, we proposed UTH, a method for learning binary embedding functions to produce very compact and performant hashes for image instance retrieval. UTH expands SRBM model training with a fine-tuning step based on triplet networks, aiming to preserve ranking information of high-dimensional global descriptors. Unlike other approaches, the rank based fine-tuning pipeline is fully unsupervised and does not require any labelled data. Through a thorough empirical evaluation on small and large-scale data sets, it is showed that UTH can reduce the data size of uncompressed descriptors by 512× (256 bits hashes) without considerable retrieval performance loss. UTH is also shown to outperform other unsupervised schemes in the 32-256 bits range.

5.5 Summary and Discussion

This chapter first showed how high-dimensional descriptors can be compressed to very compact binary representations. Key to achieving excellent performance at low rates is the regularization of SRBM. Next is shown how Siamese fine tuning can be used to improve performance. This technique is applicable where labelled external data of matching instance pairs is available. In the absence of such labelled data, we propose a

triplet hashing scheme (UTH) which preserves rank ordering, without requiring labels or external training data.

A perfect image hashing scheme would convert a high-dimensional descriptor into a low-dimensional bit representation without losing retrieval performance. We believe that the techniques proposed in this chapter are a significant step in this direction. Through a rigorous evaluation process, we show that our models perform well across various data sets, regardless of the type of image descriptors used, CNN or Fisher Vectors. The improvement over other hashing schemes is particularly marked for very low bitrates (e.g. 256 bits and lower). This work therefore represents a strong step towards the *Holy Grail* of high-performing tiny hashes.

Chapter 6

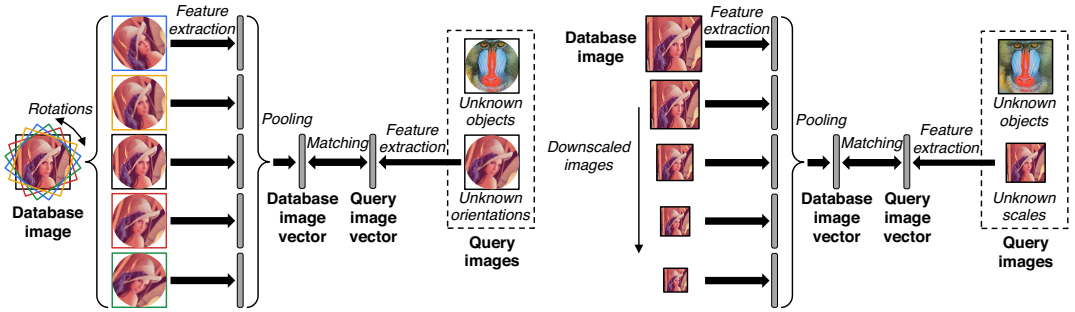
Invariant Deep Representations

In Chapter 4, we conducted a comprehensive and systematic evaluation of FV and CNN descriptors for instance retrieval. One of the main conclusions was that despite their good general performance, the CNN descriptors suffer from a lack of robustness to image transformations such as rotations and scale changes. Indeed, unlike FV based on interest point detectors, CNN do not have a built-in mechanism to ensure robustness to transformations such as rotation or scale changes. In this section, we propose techniques for making these descriptors invariant to those transformations.

First Section 6.1 shows that simple database-side pooling schemes can be effective at mitigating issues related to the aforementioned image transformations, practically overcoming the lack of robustness of CNN descriptors compared with FV. Amongst the various schemes proposed, some allow the pre-computation of single descriptors while others require more operations at query time.

In Section 6.2, we then propose Nested Invariance Pooling (NIP), a method to produce compact global image descriptors from visual representations extracted from CNN which are robust to multiple types of image transformations. NIP is inspired from *i-theory* [3, 4, 2], a mathematical theory for computing group invariant transformations with feed-forward neural networks. We show that NIP is able to produce compact (but non-binary) global image descriptors which are robust to rotation, scale changes and translations and are able to outperform other schemes at equivalent descriptor dimensionality on most evaluation datasets.

Finally, we show in Section 6.3 that NIP can be effectively combined with the RBMH scheme presented in Chapter 5, leading to hashes that are both compact and robust to multiple types of image transformations. We show through a thorough empirical evaluation on small and large-scale datasets that NIP+RBMH is able to produce extremely compact hashes that are able to outperform the other proposed schemes specially at very low bitrates (32-256 bits).



(a) Database-side rotation pooling (b) Database-side scale pooling

Figure 6.1: Database-side pooling strategy with (a) rotation transformation and (b) scale transformation. Descriptors are extracted from database images at different (a) rotations or (b) scales, then pooled together to obtain a single representation per image. Descriptors are also extracted for different rotations (a) or scales (b) of query images, allowing to benchmark the transformation invariance of corresponding database-side pooling strategy. For rotations experiments, all database and query images are cropped circularly at the center to avoid edge artefacts, and padded with a default mean RBM value (*ImageNet* mean pixel value).

6.1 Database-Side Pooling

In this section, we propose several simple schemes based on database-side pooling, where transformations are applied to database images in order to improve the robustness of CNN descriptors to image transformations. As illustrated in Figure 6.1, the two transformations used to benchmark database-side pooling are rotation and scale. Evaluation of both are done independently. In both cases, transformation invariance of the database pooled descriptors is evaluated by gradually applying the same type of transformation on the queries.

The different pooling strategies are evaluated using CNN and FV descriptors. For CNN, image descriptors are extracted from *OxfordNet* first fully-connected layer, $fc6$, shown to perform well in Chapter 4. For FV, FVDoG based on interest point detectors is used. Image instance retrieval performance is evaluated on the data sets *Holidays* and *Graphics*.

6.1.1 Gaining Invariance to Rotation

The database rotation pooling scheme illustrated in Figure 6.1(a) is proposed for gaining rotation invariance. Each database image is rotated within a range of $-p^\circ$ to p°, in steps of 10°. The CNN features for each rotated database image are pooled together into one common global descriptor representation. In Figure 6.2, we present results for max-pooling, where we store the component-wise maximum value across all rotated representations in an angular range. $P = 0$ refers to no pooling, while $P = p$ refers to

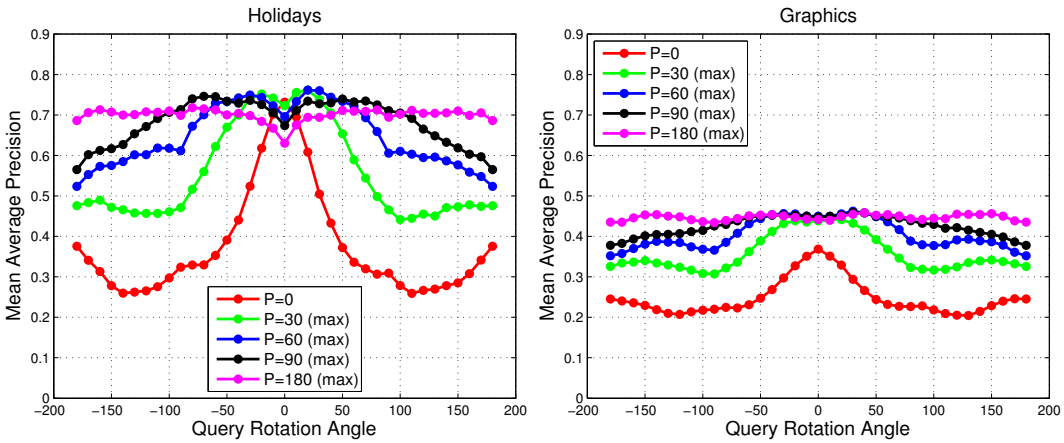


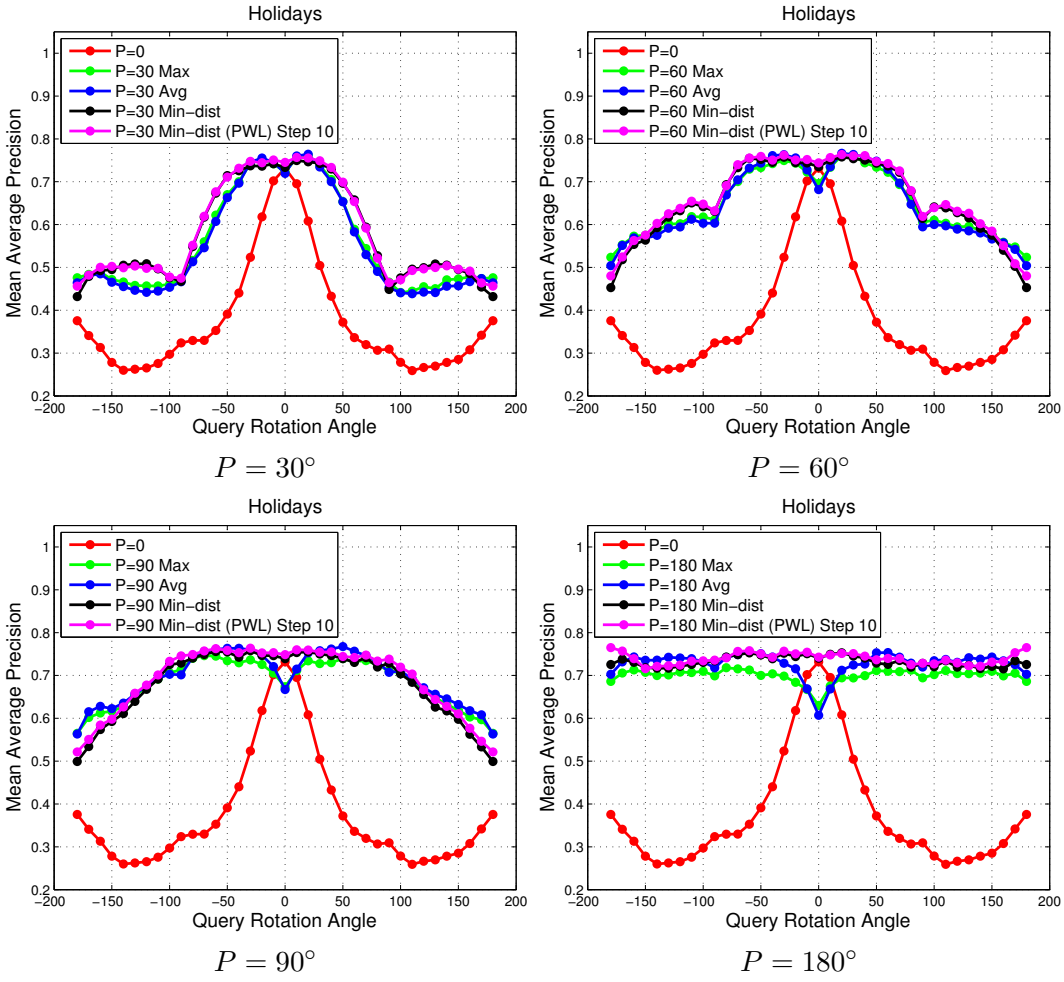
Figure 6.2: mAP vs query rotation angle for different pooling parameters. Results are presented on the *OxfordNet* $fc6$ layer on the *Holidays* data set. $P = 0$ refers to no pooling. $P = p$ refers to max-pooling over individual feature dimensions, for rotations between $-p^\circ$ and p° in steps of $s = 10^\circ$. Invariance to rotation increases with increasing p, at the expense of lower performance at angle 0°.

pooling in the range of $-p^\circ$ to p° in steps of $s = 10^\circ$. The parameter s indicates the quantization step size of angular rotation of database images.

Performance is plotted as the query rotation angle is varied for varying pooling parameter P, on the *OxfordNet fc6* layer for the *Holidays* and *Graphics* data sets. The invariance-discriminativeness trade-off is shown in Figure 6.2. We observe that the max pooling scheme performs surprisingly well for gaining rotation invariance. As P is increased, the performance curve flattens in the range of $-P^\circ$ to P°, at the expense of lower performance for upright queries, i.e. at angle 0. For the *Holidays* data set, most database and query images share similar "upright" orientations. For the *Graphics* data set, note the gap in performance of different schemes at angle 0, between no pooling and different pooling schemes. This gap in performance can be attributed to rotated objects in the query data set.

To further understand the effectiveness of database-side pooling, we evaluate different types of pooling methods and database augmentation techniques in Figure 6.3. We show results for component-wise max pooling and average pooling over rotated database images for different pooling parameters P. We note that max pooling and average pooling perform comparably for small P. For $P = 180^\circ$, we note that average pooling outperforms max pooling.

We compare the two pooling strategies to a simple database augmentation technique labeled *Min-dist*, which stores descriptors for each rotated version of the database image. For *Min-dist*, at query time, we compute the minimum distance to all the rotated versions for each database image. The *Min-dist* increases the size of the database by $\frac{2P}{s} + 1$, where $s = 10$ is the step size in degrees, and P is the pooling parameter.



$P = 30°$ $P = 60°$

$P = 90°$ $P = 180°$

Figure 6.3: Comparison of different types of database pooling and augmentation techniques, for varying pooling parameter P. *OxfordNet* layer $fc6$ is used on the *Holidays* data set. We notice that max and average pooling come close to the performance of *Min-dist* and *Min-dist(PWL)*, which require storage of multiple feature descriptors $(\frac{2P}{s}+1)$ for each database image, where $s = 10$ is the quantization step size in degrees.

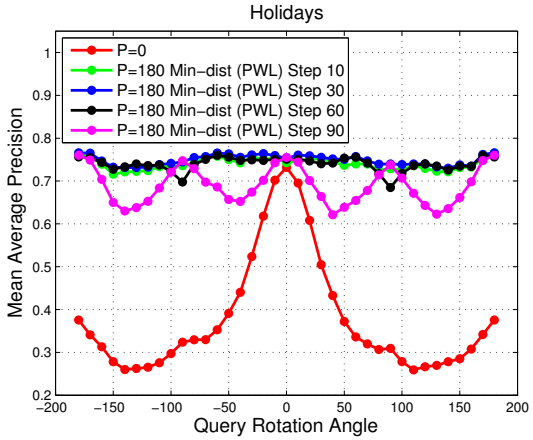


Figure 6.4: Results of the *Min-dist (PWL)* scheme as step size parameter s is varied. The *OxfordNet* layer $fc6$ is used on the *Holidays* data set. A piece-wise linear approximation of the manifold, on which rotated descriptors of each image lie, is used to trade-off performance and matching complexity. The performance of step size $s = 60°$ is close to that of $s = 10°$, while reducing memory requirements by $6\times$.

For small $s \to 0$, the *Min-dist* scheme provides an approximate upper performance boundary that the max and average pooling schemes can achieve, as a descriptor for each rotated version is explicitly stored in the database. We observe that both max and average pooling are surprisingly effective, as their performance comes close to that of the *Min-dist* scheme while storing only one descriptor per database image. Note that the *Min-dist* scheme performs the best, as there is no drop in performance at 0°, compared to the pooling methods.

Next, we also propose a scheme illustrated in Figure 6.5 for reducing memory requirements of the *Min-dist* scheme at the expense of increased matching complexity. The scheme labeled *Min-dist (PWL)* assumes a piece-wise linear approximation of the manifold on which the descriptors of each rotated image lie, and computes the closest distance to the manifold. The results for the *Min-dist (PWL)* with step size 10° are shown in Figure 6.3, and it performs comparably to the *Min-dist* scheme. Instead of maintaining database descriptors at finely quantized angular rotations of $s = 10°$, we increase s to $30°, 60°, 90°$ and present results in Figure 6.4. We note that the performance of step size $s = 60°$ is close to that of $s = 10°$ for *Min-dist (PWL)*, while reducing memory requirements by $6\times$. The drop in performance for the *Min-dist (PWL)* scheme at -135°,-45°,45°,145° for $s = 90°$ shows inherent data set bias at these angles. In conclusion, the proposed simple but elegant *Min-dist (PWL)* scheme helps gain rotation invariance, while requiring storage of fewer descriptors compared to the *Min-dist* approach.

The surprising effectiveness of max and average pooling to gain rotation invariance for CNN features led us to run the same set of experiments on FVDM. We present the

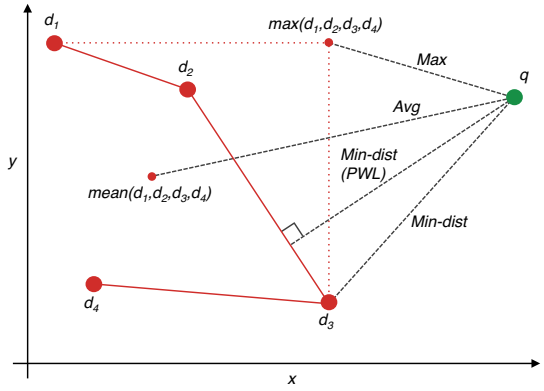


Figure 6.5: Illustration of the different pooling schemes. The scheme labeled *Min-dist (PWL)* assumes a piece-wise linear approximation of the manifold on which the descriptors of each rotated image lie, and computes the closest distance to the manifold.

results for max pooling, average pooling, *Min-dist*, and *Min-dist (PWL)* with step size $s = 10°$ in Figure 6.6 for $P = 180°$. Average pooling on FVDM helps gain invariance to rotation while lowering peak performance achieved without pooling ($P = 0$). However, note the large difference in performance between max and average pooling for FVDM. CNN features are sparse with a small number of dimensions with high values: spikes resulting from the activation of units in the network. FVDM data are comparatively denser. As a result, max pooling on *OxfordNet* features is far more effective than for FVDM. Finally, in Figure 6.6, *Min-dist* and *Min-dist (PWL)* perform the best, as also observed for *OxfordNet* features.

6.1.2 Gaining Invariance to Scale

Next, we discuss how performance at small scales can also be improved by pooling descriptors on the database side. As illustrated in Figure 6.1(b), the component-wise pooling operation across scales is similar to the database-pooling performed on rotated images. The parameter SP refers to the number of scales over which *OxfordNet* features are pooled. $SP = n$ refer to pooling across the first $n + 1$ scales of the set of six scale-ratios (seven including one) $(1, 0.75, 0.5, 0.375, 0.25, 0.2, 0.125)$). $SP = 1$, hence, refers to no database pooling.

In Figure 6.7(a), we first study mAP vs query scale for different types of pooling on the *Holidays* data set for $SP = 6$ (pooling over all scales in consideration). *OxfordNet fc6* features are used in this experiment. We note that max-pooling outperforms average pooling by a small margin, and comes close to the performance of the *Min-dist* scheme, which stores the descriptors of all the scaled versions of the database image and computes the minimum distance. Similar to the rotation experiment, the *Min-dist (PWL)* scheme, which computes the minimum distance to a piece-wise linear manifold of the CNN descriptors for the six scaled images, is also effective for the scale experi-

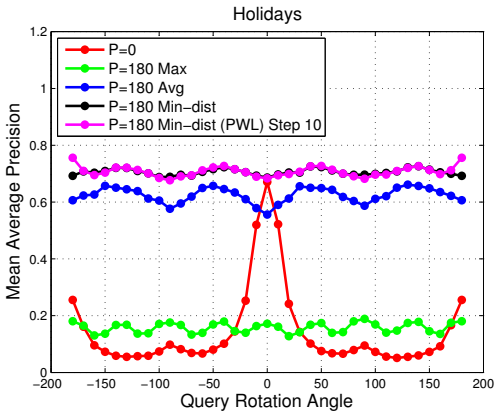


Figure 6.6: Comparison of different types of database pooling and augmentation techniques for pooling parameter $P = 180$ for FVDM on the *Holidays* data set. Note the difference in performance of max and average pooling for FVDM, compared to max and average pooling on *OxfordNet* features in Figure 6.3(d).

ment. *Min-dist (PWL)* outperforms *Min-dist* by a small margin, as it is more robust to matching query data which lie at intermediate quantized scales. For $SP = 6$, there is a significant improvement in performance at small scales for the pooling schemes, with only a marginal drop in performance for points close to the original scale (seen from the right most points on the curve in Figure 6.7(a)).

In Figure 6.7(b), we study varying pooling parameter SP for max-pooling. Performance at small scales increases as SP is increased, with only a marginal drop at query scale 0.75. A significant gain in performance of 10% is achieved for the smallest query scale 0.25, showing the effectiveness of the max-pooling approach.

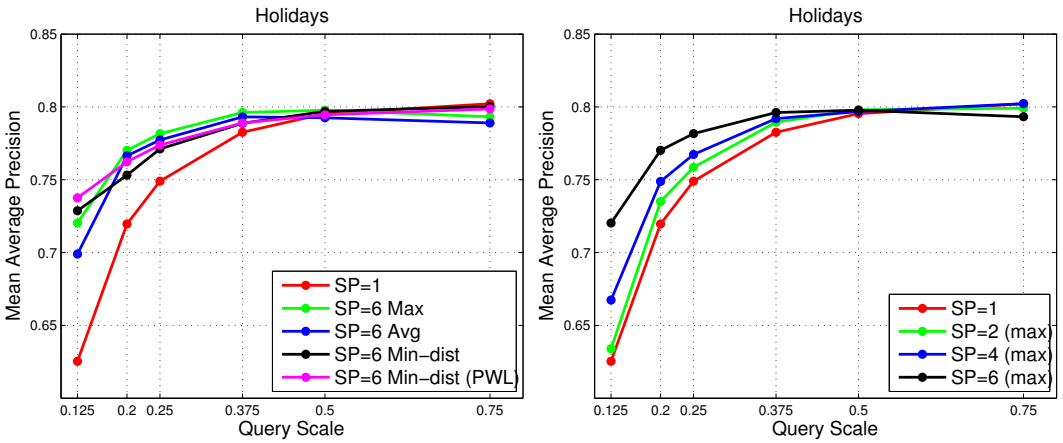
6.1.3 Conclusion

In this section, we showed that simple database-side pooling methods can be effective at improving the robustness of CNN descriptors. Nevertheless, the simple aggregation strategies were shown to sometimes be a matter of trade-off between robustness and retrieval performance.

In the next section, we propose a more comprehensive approach to compute global descriptors which are both significantly outperforming the raw CNN descriptors and robust to multiple types of image transformations applied simultaneously.

6.2 Nested Invariance Pooling

In this section, we propose Nested Invariance Pooling (NIP), a method to produce compact global image descriptors from visual representations extracted from CNNs.



(a) (b)

Figure 6.7: Performance of different database-side pooling schemes, as query scale is changed. Results reported on *OxfordNet* $fc6$ on the *Holidays* data set. $SP = 1$ refers to no database-pooling. $SP = n$ refer to pooling across the first $n + 1$ scales of the set $(1, 0.75, 0.5, 0.375, 0.25, 0.2, 0.125)$). In Figure (a), we notice that max pooling comes close to the performance of *Min-dist* which requires storing descriptors at all scales. In Figure (b), we observe that performance improves at small scales with database side pooling as parameter SP is increased.

The proposed method draws its inspiration from the *i-theory* [3, 4, 2], a mathematical theory for computing group invariant transformations with feed-forward neural networks. The theory is an information processing model explaining how feedforward information processing can be made robust to various types of signal distortions.

After showing that CNNs are compatible with the *i-theory*, we propose a simple and practical way to apply the theory to the construction of global image descriptors which are robust to various types of transformations of the input image at the same time. Through a thorough empirical evaluation based on multiple publicly available datasets, we show that proposed method is able to significantly consistently improve retrieval results while keeping dimensionality low. Rotations, translations and scale changes are studied in the scope of this section but the proposed approach is extensible to other types of transformations. We show that using moments of increasing order for incorporating invariance to multiple transformation groups throughout nesting is important. Resulting NIP descriptors are invariant to various types of image transformations and we show that the process significantly improves retrieval results while keeping dimensionality low (512 dimensions).

6.2.1 I-theory in a Nutshell

Many common classes of image transformations such as rotations, translations and scale changes can be modeled by the action of a transformation group. Let an image

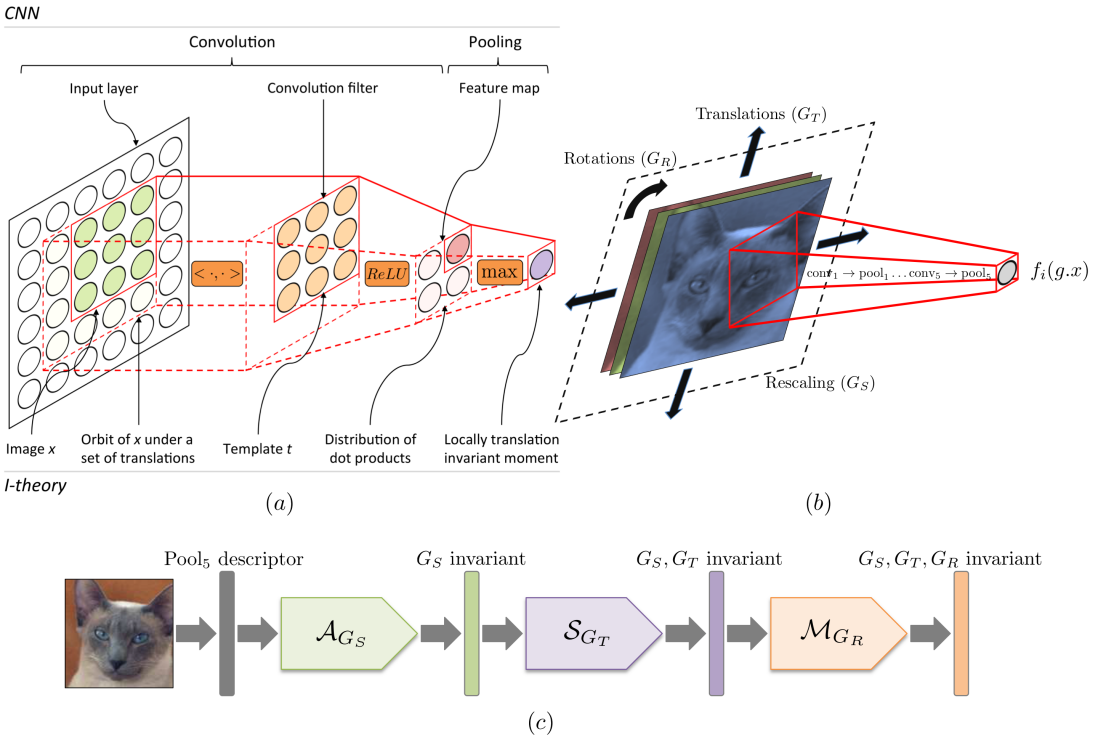


Figure 6.8: (a) A single convolution-pooling operation from a CNN schematized for a single input layer and single output unit. The parallel with *i-theory* shows that the universal building block of CNN is compatible with the incorporation of invariance to local translations of the input according to the theory. The network architecture is responsible for the invariance properties while back-propagation provides a practical way to learn the templates from data. (b) A specific succession of convolution and pooling operations learnt by the CNN (depicted in red) computes the *pool*5 feature f_i for each feature map i from the RGB image data. A number of transformations g can be applied to the input x in order to vary the response $f_i(g.x)$. (c) The proposed method takes inspiration from the *i-theory* to create compact and robust global image descriptors from CNN. Starting with raw *pool*5 descriptors, it can be used to stack-up an arbitrary number of transformation group invariances while keeping the dimensionality under control. The particular sequence of transformation groups and statistical moments represented on the diagram was shown to produce the best performing hashes on average, but other arbitrary combinations are also able to improve retrieval results.

$x \in E$ and a group G of transformations acting over E with group action $G \times E \rightarrow E$ denoted with a dot (.). The orbit of x by G is the subset of E defined as $O_x = \{g.x \in E | g \in G\}$. The orbit corresponds to the set of transformations of x under groups such as rotations, translations and scale changes. It can be easily shown that O_x is globally invariant to the action of any element of G and thus any descriptor computed directly from O_x would be globally invariant to G.

The *i-theory* builds invariant representations for a given object $x \in E$ in relation with a predefined template $t \in E$ from the distribution of the dot products $D_{x,t} = \{< g.x, t > \in \mathbb{R} | g \in G\} = \{< x, g.t > \in \mathbb{R} | g \in G\}$ over the orbit. The following representation (for any $n \in \mathbb{N}^*$) is proven to have proper invariance and selectivity properties provided that the group is compact or locally compact:

$$\mu_{G,t,n}(x) = \frac{1}{\int_G dg} \left(\int_G | < g.x, t > |^n dg \right)^{\frac{1}{n}} \tag{6.1}$$

One may note that the transformation can be applied either on the image or the template indifferently. Note that the sequence $(\mu_{G,t,n}(x))_{n \in \mathbb{N}^*}$ is analogous to a histogram. Such a representation is mathematically proven to have proper invariance and selectivity properties provided that the group is compact or at least locally compact [3].

In practice, while a compact group (e.g. rotations) or locally-compact group (e.g. translations, scale changes) is required for the theory to be mathematically provable, the authors of [3] suggest that the theory extends well (with approximate invariance) to non-locally compact groups and even to continuous non-group transformations (e.g. out-of-plane rotations, elastic deformations) provided that proper class-specific templates can be provided. Recent work on face verification [90] and music classification [168] apply the theory to non-compact groups with good results.

6.2.2 CNNs are I-theory Compliant Networks

Popular CNN architectures designed for image classification such as *AlexNet* [79] and *OxfordNet* [137] share a common building block: a succession of convolution-pooling operations designed to model increasingly high-level visual representations of the data as presented in Chapter 3. The highest level visual features may then be fed into fully-connected layers acting as classifiers.

As shown in detail on Figure 6.8 (a), the succession of convolution and pooling operations in a typical CNN is in fact a way to incorporate local translation invariance strictly compliant with the framework proposed by the *i-theory*. The network architecture provides the robustness such as predicted by the invariance theory, while training via back propagation ensures a proper choice of templates. Multiple convolution-pooling steps are applied (five times in both *AlexNet* and *OxfordNet*) resulting in

increased robustness and higher level templates. Note that the iterative composition of local translation invariance approximately translates into robustness to local elastic distortions for the features at the *pool*5 layer.

In this study, instead of the popular first fully-connected layer ($fc6$) which is on average the best single CNN layer to use as a global out-of-the-box descriptor for image retrieval (see Chapter 4), we decide to use the locally invariant *pool*5 as a starting representation for the proposed global descriptors and further enhance their robustness to selected transformation groups in a way inspired from *i-theory*.

6.2.3 Multi-Group Invariant CNN Descriptors

We build the NIP descriptors starting from the already locally robust *pool5* feature maps of *OxfordNet*. Global invariance to several transformation groups are then sequentially incorporated following the *i-theory* framework. The specific transformation groups considered in this study are translations G_T, rotations G_R and scale changes G_S. For every feature map i of the *pool5* layer ($0 \leq i < 512$), we denote $f_i(x)$ the corresponding unit's output. As shown on Figure 6.8 (b), transformations g are applied on the input image x varying the output of the *pool5* feature $f_i(g.x)$. Note that the transformation f_i is non-linear due to multiple convolution-pooling operations thus it is not strictly a mathematical dot product but can still be viewed as an inner product. Accordingly, the pooling scheme used by NIP with $G \in \{G_T, G_R, G_S\}$ is:

$$\mathcal{X}_{G,i,n}(x) = \frac{1}{\int_G dg} \left(\int_G f_i(g.x)^n dg \right)^{\frac{1}{n}} \tag{6.2}$$

$$= \frac{1}{m} \left(\sum_{j=0}^{m-1} f_i(g_j.x)^n \right)^{\frac{1}{n}} \tag{6.3}$$

when O_x is discretized into m samples. The corresponding global image descriptors are obtained after each pooling step by concatenating the moments for the individual features:

$$\mathcal{X}_{G,n}(x) = (\mathcal{X}_{G,i,n}(x))_{0 \leq i < 512} \tag{6.4}$$

As shown in Equation 6.3, the pooling operation has an order parameter n defining the "hardness" of the pooling. $n = 1$ is average pooling while $n \to +\infty$ on the other extreme is max-pooling. $n = 2$ is analogous to standard deviation. Subsequently, we refer to the moments for $n = 1, 2, +\infty$ as $\mathcal{A}_G$, $\mathcal{S}_G$ and $\mathcal{M}_G$.

Work on *i-theory* [168] has shown that it is possible to chain multiple types of group invariances one after the other [168]. We apply this principle on NIP descriptors by making them invariant to several transformations. For instance, following scale invariance with average ($n = 1$) by translation invariance with hard max-pooling ($n \to$

$+\infty$) is done by:

$$\max_{g_t \in G_T} \left(\frac{1}{\int_{g_s \in G_S} dg_s} \int_{g_s \in G_S} f_i(g_t g_s.x) dg_s \right) \tag{6.5}$$

$$= \max_{j \in [0, m_t - 1]} \left(\frac{1}{m_s} \sum_{i=0}^{m_s - 1} f_i(g_{t,j} g_{s,i} t.x) \right) \tag{6.6}$$

Operations are sometimes commutable (e.g. $\mathcal{A}_G$ and $\mathcal{A}_{G'}$) and sometimes not (e.g. $\mathcal{A}_G$ and $\mathcal{M}_{G'}$) depending on the specific combination of moments so the sequence of transformations does matter for NIP. The hardness parameter n must also be chosen carefully. Empirically, we found pooling progressively with increasing moments (e.g. $\mathcal{A}_G$, then $\mathcal{S}_G$, then $\mathcal{M}_G$) to work well, as presented in the experiments section.

Pairwise Matching Distance. Image instance retrieval starts with the construction of a list of database images ordered according to their pairwise matching distance with the query image. With CNN descriptors, the matching distance is strongly affected by commonly encountered image transformations. As shown in Chapter 4, a rotation of the query image by more than ten degrees causes a sharp drop in results. This particular issue is much less pronounced with the popular Fisher vectors, largely due to the use of interest point detectors.

Figure 6.9 provides an insight on how adding different types of invariance with the proposed method will affect the matching distance on different image pairs of matching objects. With the incorporation of each new transformation group, we notice that the relative reduction in matching distance is the most significant with the image pair, which is the most affected by the transformation group.

6.2.4 Evaluation Framework

The *pool*5 layer from the sixteen layers *OxfordNet* [137] is chosen as starting representation, with a total dimensionality of 25088 organized in 512 feature maps of size 7×7.

For rotation invariance, rotated input images are padded with the mean pixel value computed from the *ImageNet* data set. The step size for rotations is ten degrees yielding 36 rotated images per orbit. For scale changes, ten different center crops geometrically spanning from 100% to 50% of the total image have been taken. For translations, the entire feature map is used for every feature, resulting in an orbit size of $7 \times 7 = 49$.

We evaluate the instance retrieval performance of the descriptors against four popular data sets: *Holidays*, *UKBench*, *Oxbuild*, and *Graphics*. The four data sets are chosen for the diversity of data they provide: *UKBench* and *Graphics* are object-centric featuring close-up shots of objects in indoor environments. *Holidays* and *Oxbuild* are

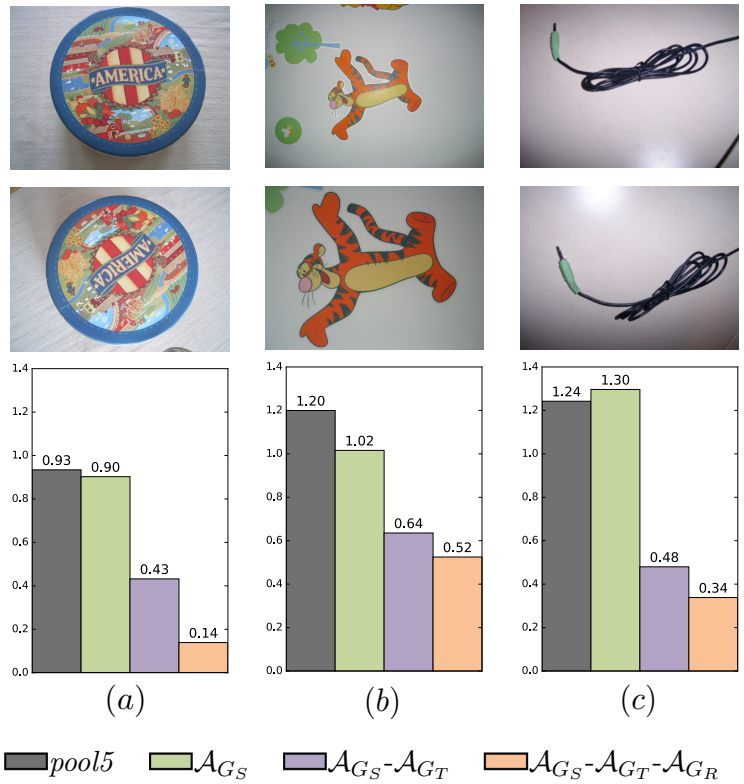


Figure 6.9: Distances for three matching pairs from *UKBench*. For each pair, four pairwise distances (L_2-normalized) are computed corresponding to the following descriptors: *pool*5, $\mathcal{A}_{G_S}$, $\mathcal{A}_{G_S}$-$\mathcal{A}_{G_T}$ and $\mathcal{A}_{G_S}$-$\mathcal{A}_{G_T}$-$\mathcal{A}_{G_R}$. Adding scale invariance makes the most difference on (b), translation invariance on (c), and rotation on (a) which is consistent with the scenarios suggested by respective images pairs.

SEQUENCE	DIMS	DATASET			
		Oxbuild	*Holidays*	UKB	Graphics
pool5	25088	0.427	0.707	0.823(3.11)	0.315
fc6	4096	0.461	0.782	0.910(3.50)	0.312
$\mathcal{A}_{G_T}$	512	0.477	0.800	0.924(3.56)	0.322
$\mathcal{A}_{G_R}$	25088	0.462	0.779	0.954(3.72)	0.500
$\mathcal{A}_{G_S}$	25088	0.430	0.716	0.828(3.12)	0.394
$\mathcal{A}_{G_T}$-$\mathcal{A}_{G_R}$	512	0.418	0.796	0.955(3.73)	0.417
$\mathcal{A}_{G_T}$-$\mathcal{A}_{G_S}$	512	0.537	0.811	0.931(3.61)	0.430
$\mathcal{A}_{G_R}$-$\mathcal{A}_{G_S}$	25088	0.494	0.815	0.959(3.75)	0.552
$\mathcal{A}_{G_T}$-$\mathcal{A}_{G_R}$-$\mathcal{A}_{G_S}$	512	0.484	0.833	0.971(3.82)	0.509
$\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$	512	**0.592**	**0.838**	**0.975(3.84)**	**0.589**

Table 6.1: Retrieval results (mAP) for different sequences of transformation groups and moments. Results are computed with the mean average precision (mAP) metric. For reference, 4×Recall@4 results are also provided for *UKBench* (between parentheses). G_T, G_R, G_S denote the groups of translations, rotations and scale changes respectively. Note that averages commute with other averages so the sequence order of the composition does not matter when only averages are involved. Best results are achieved by choosing specific moments. $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ corresponds to the best average performer. *fc6* and *pool*5 are provided as a baseline.

scene-centric data sets consisting primarily of outdoor buildings and scenes. Results are evaluated using Mean Average Precision (mAP) and 4× Recall @ $R = 4$ for *UKB*, to be consistent with the literature.

6.2.5 Results

Transformations, Order and Moments. As shown in Table 6.1, we first study the effects of incorporating various transformation groups and using different moments on descriptors. *Pool5* which is the starting point of NIP descriptors and $fc6$ which is considered the best off-the-shelf descriptor (Chapter 4, [127]) are provided as baselines. We present results for all possible combinations of transformation groups for average pooling (order does not matter as averages commute) and for the single best performer which is $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ (order matters).

First, we can immediately point out the high potential of *pool*5. Although it performs notably worse than $fc6$ as-is, a simple average pooling over the space of translations $\mathcal{A}_{G_T}$ makes it both better and eight times more compact than $fc6$. Similar observations have also been reported by [8, 7].

Second, as shown in Figure 6.10, accuracy increases with the number of transformation groups involved. On average, single transformation schemes perform 21%

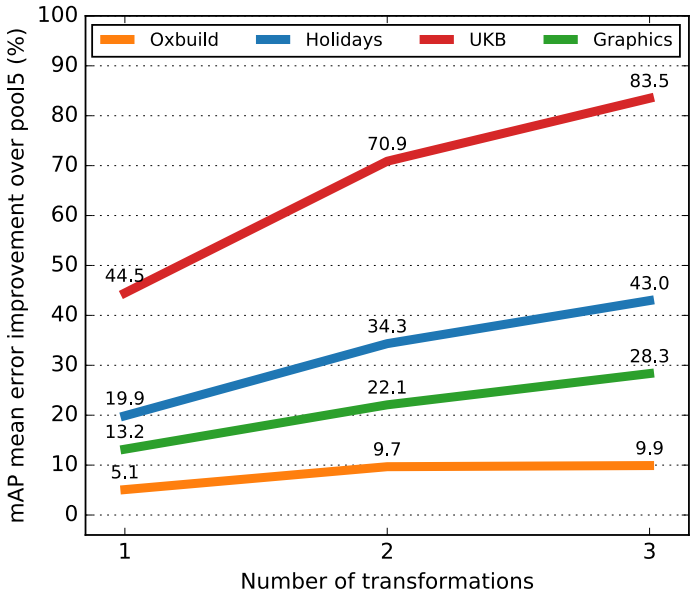


Figure 6.10: Results from Table 6.1 for the seven strategies using averages only (rows three to nine) expressed in terms of improvement in mAP over *pool*5, and aggregated by number of invariance groups. Improvements range from +5% on *Oxbuild* using one transformation to +83.5% on *UKBench* using three transformations. On all four data sets, results clearly improve with the amount of groups considered.

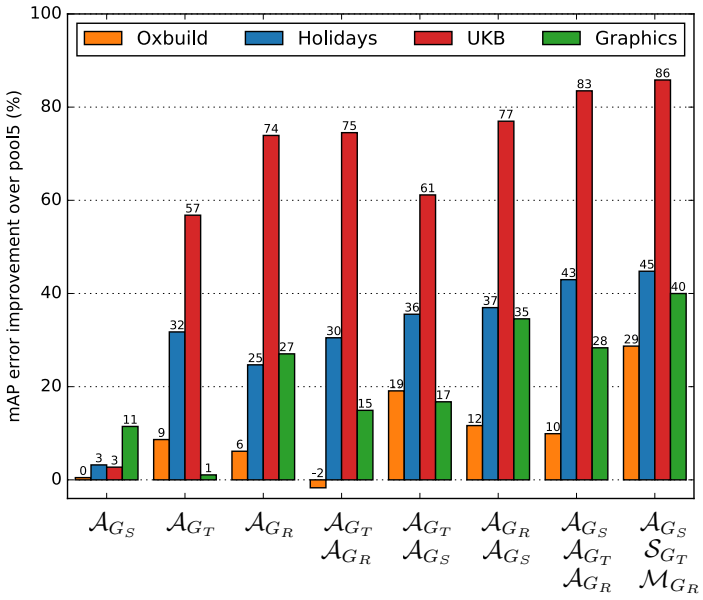


Figure 6.11: Results from Table 6.1 expressed in terms of improvement in mAP over *pool*5. Most strategies yield significant improvements over *pool*5 on most data sets. The average improvement is 68% for the best strategy.

METHOD	DIMS	DATASET		
		Oxbuild	*Holidays*	UKB
T-embedding [75]	1024	0.560	0.720	3.51
T-embedding [75]	512	0.528	0.700	3.49
FV+Proj [52]	512	-	0.789	3.36
FC+PCAWhitening [127]	500	0.322	0.642	-
FC+VLAD+PCA [48]	512	-	0.784	-
FC+Finetune+PCAWhitening [9]	512	0.557	0.789	3.30
Conv+MaxPooling [136]	256	0.533	0.716	-
FV+FC+PCAWhitening [121]	512	-	0.827	3.37
Conv+SPoC+PCAWhitening [8]	256	0.589	0.802	3.65
R-MAC+PCAWhitening [148]	512	**0.668**	-	-
R-MAC+PCAWhitening [148]	256	0.561	-	-
NIP	512	0.592	**0.838**	**3.84**
NIP+PCAWhitening	256	**0.609**	**0.836**	**3.83**

Table 6.2: Retrieval performance comparing NIP to other state-of-the-art methods. We include results in recent papers with comparable dimensionality of descriptors reported in those papers. L2 distance is used for all methods.

better compared to *pool*5, 2-transformations schemes perform 34% better, and the 3-transformations scheme performs 41% better.

Third, choosing statistical moments different than averages further improves the retrieval results. In Figure 6.11, we observe that $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ performs roughly 17% better (average results over all data sets) than $\mathcal{A}_{G_S}$-$\mathcal{A}_{G_T}$-$\mathcal{A}_{G_R}$. Notably, the best combination corresponds to an increase in the orders of the moments: $\mathcal{A}$ being a first-order moment, $\mathcal{S}$ second order and $\mathcal{M}$ of infinite order. A different way of stating this fact is that a more invariant representation requires a higher order of pooling. Overall, $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ improves results over starting representation *pool*5 by 39% (*Oxbuild*) to 87% (*Graphics*) depending on the data set. Better improvements with *Graphics* can be explained with the presence of many rotations in the data set (smaller objects taken under different angles) while *Oxbuild* consisting mainly of upright buildings is less significantly helped by incorporating rotation invariance.

Comparison with State-of-the-Art. State-of-the-art descriptors include variants of VLAD/FV [75, 52], deep descriptors [9, 136, 8, 148] and descriptors combining deep CNN and VLAD/FV [48, 121]. As shown in Table 6.2, we observe that 512-D NIP descriptors largely outperform most state-of-the-art methods with 512 or higher dimensions, on all data sets. Following [136, 8, 148], we also perform PCA whitening to reduce the dimensionality of NIP to 256. One can see that the 256-D NIP descriptors yield superior performance to [136, 8, 148] on all data sets.

First, we compare NIP to the most related papers [136, 8, 148] which propose 256-D deep descriptors by aggregating convolutional features with various pooling operations. With only one layer of pooling, [136, 8, 7] can be considered a special case of NIP, providing only limited levels of translation invariance. The recently proposed Regional Maximum Activation of Convolutions (R-MAC) [148] reports outstanding results on building data set *Oxbuild* with very small dimensionality (e.g. 0.668 mAP for 512-D R-MAC and 0.561 mAP for 256-D R-MAC). The authors propose a fast R-CNN type pooling [40], which is effective when the object of interest is in a small portion of the image. Such an approach will be less effective when the object of interest is affected by groups of distortions like rotation and perspective, and located at the centre of the image. Here, we observe that nested pooling over many types of distortions with progressively increasing moments is essential to achieving geometric invariance and high retrieval performance with low dimensional descriptors. Besides, the technique proposed in [148] can be incorporated with NIP to further improve performance.

Next, we note that [136] reports better results on *Holidays* (0.881 mAP) and *Oxbuild* (0.844 mAP), with very high-dimensional descriptors (from 10K to 100K). These very high dimensional descriptors are obtained by combining CNN descriptors with spatial max pooling [7]. In contrast, NIP results are generated using only 256 to 512 dimensional descriptors.

6.2.6 Conclusion

In this section, we proposed Nested Invariance Pooling (NIP), a novel method based on *i-theory* for creating robust and compact global image descriptors from CNN for image instance retrieval. Through a thorough empirical study, we show that the incorporation of every new group invariance property following the method leads to consistent and significant improvements in retrieval results. NIP has a few parameters (sequencing of transformations and choice of statistical moments are important), but experiments show that many default and reasonable settings produce results which can generalize well across all data sets, meaning that the risk of overfitting is low. This study also confirms the high potential of the feature pyramid ($pool5$) as a starting representation for high-performance compact hashes instead of the more commonly used first fully-connected layer ($fc6$).

6.3 Hashing with Group Invariant Features

In this section, we combine NIP with RBMH (Figure 6.12), the descriptor hashing scheme presented in Chapter 5, with the aim to achieve the best performing tiny 32-256 bits hashes. Multi-group invariant NIP representations, shown to be outstanding in previous section, are used as starting representations for hashing. RBMH are used

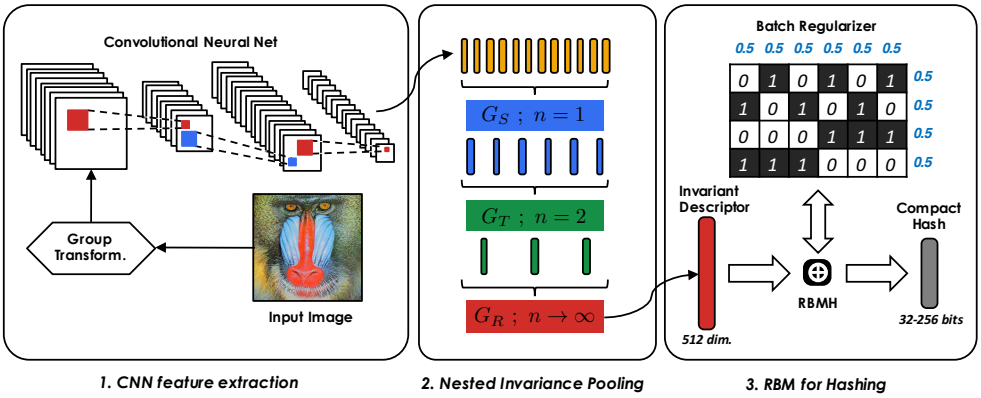


Figure 6.12: Nested Invariance Pooling (NIP) to produce robust descriptors from CNNs can be followed by RBMH for compact and invariant hashes.

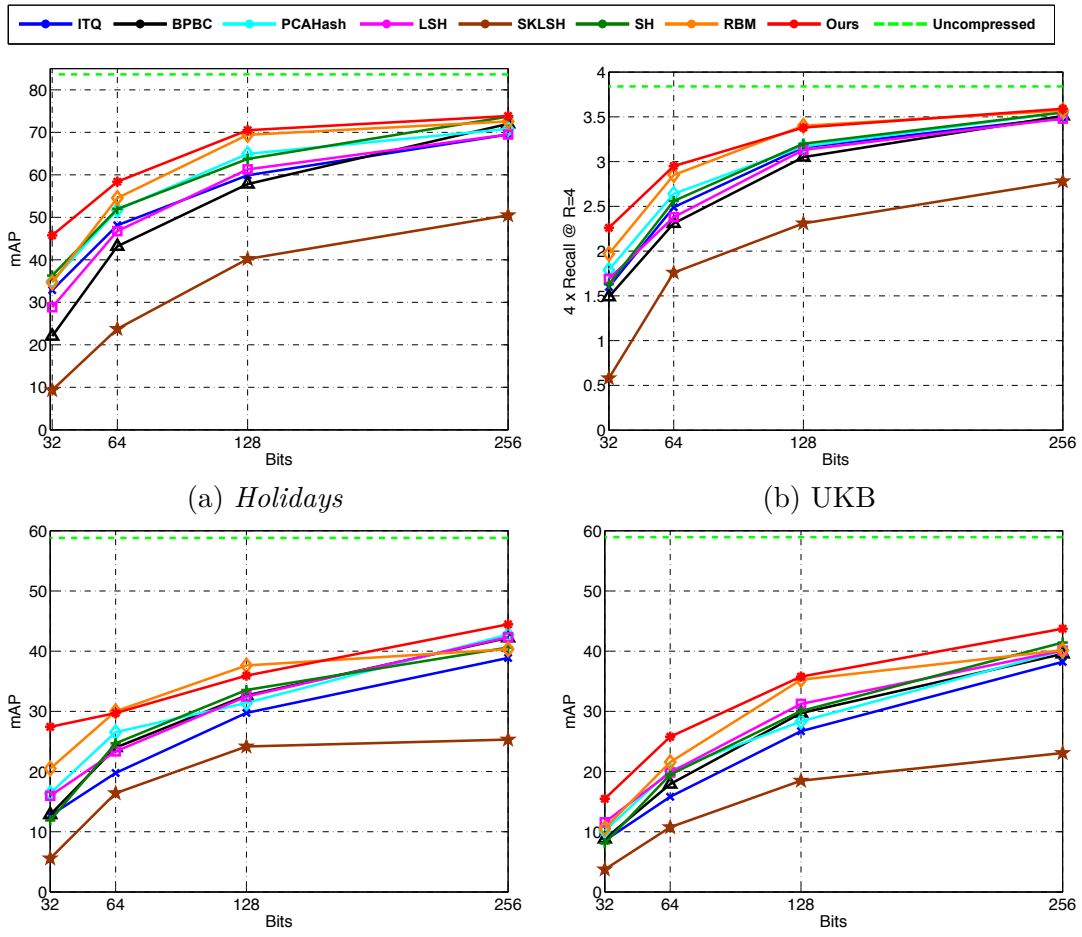
then to produce compact binary hashes from the $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ 512-D floating point values descriptors.

Resulting hashes are compared to hashes obtained, using the same starting representation, with popular unsupervised hashing methods including ITQ [46], Bilinear Projection Binary Codes (BPBC) [45], PCAHash [46], LSH [27], SKLSH [126], SH [156] and RBM. Evaluated bitrates range from 32 to 256 bits. Original $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ 512-D floating point descriptors are also introduced as the baseline uncompressed scheme. Experiments are conducted both with small-scale (Section 6.3.1) and large-scale (Section 6.3.2) retrieval datasets. For small scale image instance retrieval experiments, four popular data sets are used: *Holidays*, *UKBench*, *Oxbuild* and *Graphics*. For large-scale experiments, results are presented using the four data sets combined with the one million MIR-FLICKR distractor images [68].

Finally, results of NIP+RBMH are also compared with other state-of-the-art methods in Section 6.3.3. NIP+RBMH are showed to be amongst the most compact and efficient binary codes for image retrieval reported in the literature.

6.3.1 Small Scale Experiments

Small scale retrieval results of multi-group invariant hashes are shown in Figure 6.13, compared to other popular unsupervised hashing methods. RBMH codes outperforms other methods at most code sizes on all data sets. First, there is a significant improvement at smaller code sizes like 32 bits, due to the proposed batch-level regularization: 0.457 vs. 0.369 in terms of mAP, compared to the second performing method RBM on *Holidays* at 32 bits. Second, the improvements of NIP+RBMH over other methods becomes smaller as code size increases (except SKLSH). For code size larger than 256 bits, the performances of all methods approach the upper bound, i.e., uncompressed descriptors. Finally, compared to uncompressed descriptors, there is a marginal drop



(a) *Holidays* (b) UKB

(c) *Oxbuild* (d) *Graphics*

Figure 6.13: Comparison of RBMH with other hashing methods on four benchmark data sets. All methods are built upon the best NIP descriptors. To examine the effect of compression, retrieval results using uncompressed NIP descriptors are also presented.

for all methods on *UKBench* at 256 bits, while performance gap is larger for other data sets.

6.3.2 Large Scale Experiments

In Figure 6.14, large scale retrieval results are presented, combining the one million MIR FLICKR distractor images with each data set respectively. Trends consistent with small scale retrieval results in Figure 6.13 are observed.

6.3.3 Comparison with State-of-the-Art Image Hashing Pipelines

Invariant binary hashes are compared against different state-of-the-art pipelines, including methods compressing VLAD/FV with direct binarization [122], hashing [51], PQ [74, 171], and methods based on compact deep descriptors [9, 136].

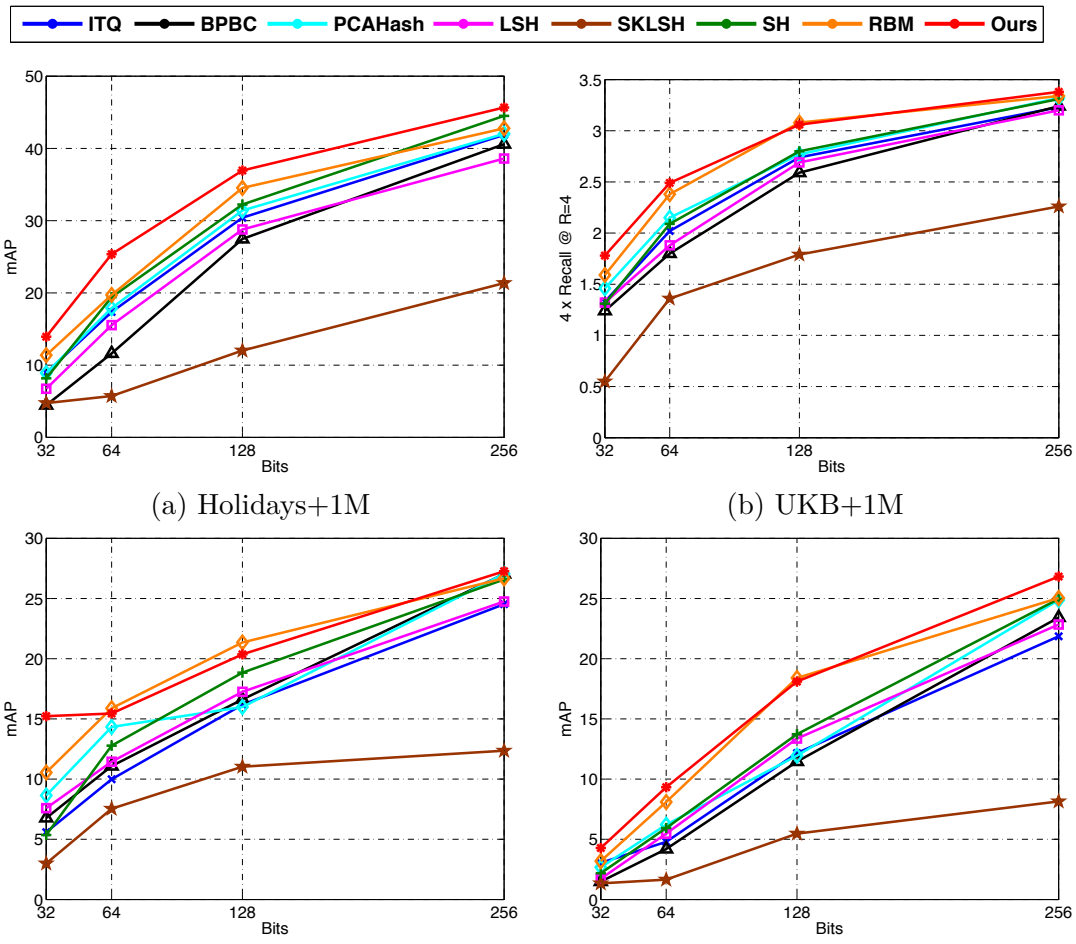
As shown in Table 6.3, first, a simple binarization strategy (thresholding at data set mean) applied to $\mathcal{A}_{G_S}$-$\mathcal{S}_{G_T}$-$\mathcal{M}_{G_R}$ descriptor degrades retrieval performance only very marginally and is sufficient to obtain significantly better accuracy than [122, 9] at comparable code size (512 bits), e.g., 3.7 vs. 2.79 in [122] for 4× Recall @ $R = 4$ on *UKBench*.

Second, NIP+RBMH outperforms state-of-the-art by a significant margin at comparable code sizes (from 32 to 256 bits). NIP+RBMH achieves the best performance on *Holidays* at small code size (128 bits), 0.705 vs. 0.644 mAP reported in the state-of-the-art [170].

Note that Hamming distance is used for the multi-group invariant binary descriptors, while other methods like PQ variants employ Euclidean distances (L2 or ADC), which typically result in higher accuracy than Hamming distance, at the expense of higher computational cost.

6.4 Summary and Discussion

Nested Invariance Pooling (NIP), a method to produce global image descriptors from CNN which are both compact and robust to typical geometric transformations, is proposed in this chapter. The method provides a practical and mathematically proven way for computing invariant object representations with feed-forward neural networks. The NIP descriptors are compact, robust to multiple classes of image transformations, and able to substantially outperform the higher-bitrate original CNN descriptors. They also compare very favourably to other existing schemes at similar bitrates (512 bits). NIP is showed to be compatible with the RBMH hashing scheme proposed in Chapter 5. Combined, the NIP+RBMH pipeline produces some of the best performing hashes available in the literature, especially at very low bitrates (32-256 bits).



(a) Holidays+1M

(b) UKB+1M

(c) Oxbuild+1M

(d) Graphics+1M

Figure 6.14: Comparison of RBMH with other hashing methods on large scale retrieval experiments. All methods are based on the best NIP descriptors.

METHOD	DIMS (size in bits)	DIST.	DATASET		
			Oxbuild	*Holidays*	UKB
Binarized FV [122]	520(520)	Cosine	-	0.460	2.79
FV+SSH [51]	256(256)	ADC	-	0.544	3.08
FV+SSH [51]	128(128)	ADC	-	0.499	2.91
FV+SSH [51]	32(32)	ADC	-	0.334	2.18
FV+PQ [74]	128(128)	ADC	-	0.506	3.10
VLAD+PQ [170]	128(128)	L2	-	0.586	2.88
VLAD+CQ [170]	128(128)	L2	-	0.644	3.19
VLAD+SQ [171]	128(128)	L2	-	0.639	3.06
FC+Finetune+PCAWhitening [9]	16(512)	L2	0.418	0.609	2.41
Conv+MaxPooling [136]	256(256)	Cosine	0.436	0.578	-
Binarized NIP	512(512)	Hamming	**0.477**	**0.781**	**3.70**
NIP+RBMH	256(256)	Hamming	**0.445**	**0.739**	**3.59**
NIP+RBMH	128(128)	Hamming	**0.359**	**0.705**	**3.38**

Table 6.3: Retrieval performance comparing NIP+RBMH to other state-of-the-art methods at small codesizes (from 32 to 512 bits). ADC denotes asymmetric distance computation [74, 51].

Chapter 7

Conclusions

We started this dissertation in Chapter 2 by introducing a set of methods relevant to the image instance retrieval task, namely hand crafted global descriptors with Fisher Vectors (FV), supervised learning with multi-layer neural networks, unsupervised learning with Restricted Boltzmann Machine (RBM), and subsequently a discussion about deep learning. While Fisher Vectors have long been the state-of-the-art for instance retrieval task, deep learning has recently emerged as a promising alternative.

In Chapter 3, we presented the design principles behind the Deep Convolutional Neural Network (CNN), which recently became the method of choice for large-scale image classification tasks, and proposed an original multistage approach for the fusion of the output of multiple CNN. This work was submitted as part of the ILSVRC 2014 challenge and results show that the approach can significantly improve classification results. Previously leveraging on hand-crafted descriptors, submissions to recent editions of ILSVRC are now unanimously using CNN, which are more accurate largely due to their capability to learn good high-level visual representations from the data. Enlightened by the abundant literature describing how CNN representations learnt on one particular classification task can transfer well to other classification tasks, we study the transferability of representations learnt by CNN to image instance retrieval in following chapters.

In Chapter 4, we proposed a systematic and in-depth evaluation of FV and CNN pipelines for image retrieval, leading to a comprehensive set of practical guidelines for the implementation of state-of-the-art descriptors. CNN descriptors offer the best retrieval performance on average, and further accuracy improvements are obtained by mixing CNN and FV at the cost of increased descriptor size. The study also pointed out on one hand the lack of transformation invariance of the CNN descriptors and on the other hand the high dimensionality and scalar nature of the descriptors, making descriptor matching inefficient. In subsequent chapters, we investigated methods for making CNN descriptors more compact and more invariant to transformations.

In Chapter 5, we addressed dimensionality reduction and binarization of CNN de-

scriptors. We first show how high-dimensional descriptors can be compressed to very compact binary representations in an unsupervised fashion using RBM, and proposed a novel RBM regularization scheme for Hashing (RBMH), key to achieving excellent performance at low rates. Next, we show how Siamese fine tuning can be used to further improve hashes performance, and proposed a novel objective function suited for training Siamese networks in high dimensional latent spaces settings. However, the technique is applicable where labelled external data of matching instance pairs is available. In the absence of such labelled data, we proposed a fully Unsupervised Triplet Hashing scheme (UTH), which preserves the original descriptors rank ordering. Regardless of the type of image descriptors used, proposed schemes are shown to outperform state-of-the-art hashing schemes for image instance retrieval, the improvement being particularly marked at very low bitrates (e.g. 256 bits and lower).

In Chapter 6, we addressed the problem of CNN descriptors limited robustness to geometric transformations, as initially observed in Chapter 4. We first show that simple database-side pooling methods can be effective at improving the robustness of CNN descriptors, and subsequently proposed Nested Invariance Pooling (NIP), a novel method to produce global image descriptors from CNN. The NIP method provides a practical and mathematically proven way for computing invariant object representations with feed-forward neural networks. NIP descriptors are compact, robust to multiple classes of image transformations and able to substantially outperform the higher-bitrate original CNN descriptors on image instance retrieval tasks. Finally, we show that NIP is compatible with the RBMH hashing scheme proposed in Chapter 5. Combined, the NIP+RBMH pipeline produces some of the best performing hashes available in the literature, especially at very low bitrates (32-256 bits).

Appendices

Appendix A

Retrieval Data Sets

In this thesis, several popular image instance retrieval data sets are selected to allow significant observations and to benchmark against the state-of-the-art methods. The data sets involved in retrieval experiments are described in this section, ordered by chronological date of publication.

A.1 University of Kentucky Benchmark

Published in 2006, the University of Kentucky data set [117] consists of 2550 categories of common objects for a total of 10200 manually collected images (4 images per category). The data set mainly features close up shot of objects found inside the house. Most often, only the object of interest is present in each image, resulting in no or little foreground or background clutter. All 10200 images are 640x480 pixels and used as queries. The data set is referred as *UKBench* in this document.

A.2 Oxford Buildings

Published in 2007, the Oxford Buildings data set [123] consists of 5062 images collected from Flickr. The manually annotated data set depicts landmarks of 11 different buildings of Oxford city. Each category is represented by 5 queries, for a total of 55 queries. Because of the limited number of queries, one should be cautious when interpreting retrieval results based on this data set. Sometimes also referred as "Oxford5k" in the literature, this data set is referred as *Oxbuild* in this document.

A.3 INRIA Holidays

Published in 2008, the INRIA Holidays data set [71] contains 1491 large resolution images, mostly personal holiday pictures from INRIA researchers. It includes a large variety of outdoor scene types: natural, man-made, water and fire effects. It features

a total of 500 categories. One query is used for each category to perform retrieval experiments on the 991 remaining database images. Variations in lighting conditions are rare in this data set as the pictures from the same location are taken at the same time. The data set is referred as *Holidays* in this document.

A.4 MIR-FLICKR

Published in 2010, MIR-FLICKR data set [68] is made of one million images and associated tags collected from Flickr. It depicts a large variety of objects and sceneries. This data set is referred as *MIRFlickr*.

In this document, *MIRFlickr* tags are not used. Combined with small scale retrieval data sets, *MIRFlickr* images are used as distractors for large scale retrieval experiments.

A.5 Stanford Mobile Visual Search

Published in 2011, the Stanford Mobile Visual Search (*SMVS*) data set [16] contains a list of 16,319 manually collected matching image pairs, comprising a wide range of object categories, including CDs, DVDs, books, software products, landmarks, business cards, text documents, museum paintings and video clips. *SMVS* has 1200 database images and 3300 queries.

The *Graphics* data set is a subset of SMVS, which notably was used in the MPEG standard: Compact Descriptors for Visual Search (CDVS) [141]. It contains five categories of objects: CDs, DVDs, books, business cards and newspaper prints, with at least one of the references being a clean version of the product obtained from the product website. There are 500 unique objects, 1500 queries, and 1000 database images.

The query images include foreground and background clutter that would be considered typical in real-world scenarios, e.g., a picture of a CD might contain other CDs in the background. This data set distinguishes from the other ones as it contains images of rigid objects captured under widely varying lighting conditions, perspective distortion, foreground and background clutter. Query images are taken with heterogeneous phone cameras. Each query has two relevant images.

A.6 Retrieval Data Sets Summary

Name	Year	DB	Query	Classes	Topic	Source
UKBench	2006	10200	10200	2550	Mostly house objects	Shot on purpose
Oxbuild	2007	5062	55	17	Buildings of Oxford	Flickr
Holidays	2008	991	500	500	Mostly holidays scenes	Personal photos
MIRFlickr	2010	1M	-	-	General	Flickr
Graphics	2011	1000	1500	500	DVD, book, business card	Shot on purpose

Table A.1: A summary of the data sets used in retrieval experiments, ordered by date of publication. On one hand, *UKBench* and *Graphics* are more object-centric, with many close-up shots of objects in indoor environments. On the other hand, *Oxbuild* and *Holidays* are more scene-centric data sets consisting primarily of outdoor buildings and landscapes. The combination of the numerous *MIRFlickr* distractor images with small scale retrieval databases allows to run large scale retrieval experiments.

Appendix B

Video Summarization

B.1 Introduction

Video sharing websites measure user engagement through click rates and viewership. To make a novel video attractive for the audience, its video link is often presented as a thumbnail of either a single representative frame or a slideshow of several keyframes. In this work, we explore the problem of automatically generating diverse, representative and attractive keyframe-based summaries for videos.

Compact keyframe-based video summaries are a popular way of generating viewership on video sharing platforms. Yet, creating relevant and compelling summaries for arbitrarily long videos with a small number of keyframes is a challenging task. We propose a comprehensive keyframe-based summarization framework combining deep convolutional neural networks and restricted Boltzmann machines. An original co-regularization scheme is used to discover meaningful subject-scene associations. The resulting multimodal representations are then used to select highly-relevant keyframes. A comprehensive user study is conducted comparing our proposed method to a variety of schemes, including the summarization currently in use by one of the most popular video sharing websites. The results show that our method consistently outperforms the baseline schemes for any given amount of keyframes, both in terms of attractiveness and informativeness. The lead is even more significant for smaller summaries.

Summarization-based techniques can be broadly divided into three categories:

1. keyframe-based,
2. skimming-based,
3. story-based.

In keyframe-based summarization, the video is summarized using a small number of keyframes selected based on some criterion, such as low-level features like pixel

Figure B.1: Deep co-regularized keyframe summary. Our method extracts diverse, representative and attractive keyframes.

data, motion features, optical flow and frame differences [95, 157, 169], or higher-level information like objects and faces [93, 39]. For this class of algorithms, clustering techniques like k-means are popular: clustering or grouping is performed based on raw RGB pixels, or a combination of low and high level features [112, 55, 29, 36, 37]. The frames closest to the cluster centers are chosen to be part of the summary.

Skimming-based summarization is used to produce longer video summaries. The video is divided into smaller shots using shot boundary detection algorithms and a series of shots are selected to form the summary video. Subshot selection is based on motion activity [116, 115, 81] and other high level features, such as person and landmark descriptors [54].

Finally, in storyboard-based summarization, algorithms take into account relationships between the different subshots [101]. This enables long egocentric videos to be summarized to gain an understanding of the underlying events.

Contributions. This work focuses on generating compact keyframe-based summarization, with the main contributions are as follows:

- A comprehensive keyframe-based summarization framework combining deep convolutional neural networks (CNN) and restricted Boltzmann machines (RBM).
- A co-regularization scheme for restricted RBM able to learn joint high-level subject-scene representations.
- A comprehensive user study comparing our method against various schemes including the algorithm in use by the video sharing website *Dailymotion*.

B.2 Co-Regularized Deep Representations

A good keyframe-based summary should consist of easily recognizable subjects in context-setting scenes. To achieve this, we generate frame-level descriptions by exploiting deep convolutional architectures to recognize subjects and scenes. Compact representations are then computed with a novel co-regularization unsupervised learning

scheme to exhibit the high-level associations between subjects and scenes. Keyframes are subsequently generated from these compact representations.

B.2.1 Deep Convolutional Neural Networks

Deep convolutional neural networks (CNN) have recently been used to obtain astonishing performances in both image classification [79, 127] and image retrieval (Section 6.2) tasks. For every frame sampled from the video at regular intervals, CNN descriptors are extracted using the open source Caffe framework [77] along with two pre-trained networks: VGG-ILSVRC-2014-D [137] and Places-CNN [172].

VGG-ILSVRC-2014-D is the best performing single network from the VGG team during the ILSVRC 2014 image classification and localization challenge using the it ImageNet [30] data set. This 138 million parameters network is made of 16 layers: 13 convolutional layers followed by 3 fully-connected layers. It detects 1000 mostly subject-centric categories (e.g. animals, objects, plants, etc...).

Places-CNN is a 60 million parameters network following the it AlexNet [79] structure: a total of 8 layers: 5 convolutional layers followed by 3 fully-connected layers. It is trained on the Places 205 data set, a scene-centric image data set featuring 205 categories including indoors and outdoors sceneries.

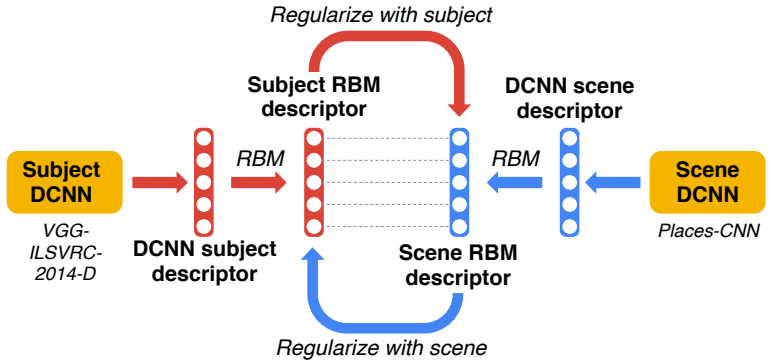
For both CNN, descriptors are extracted from the last layer before the softmax operation, having a dimensionality of 1000 and 205 for VGG-ILSVRC-2014-D and Places-CNN, respectively.

B.2.2 Co-Regularized Restricted Boltzmann Machines

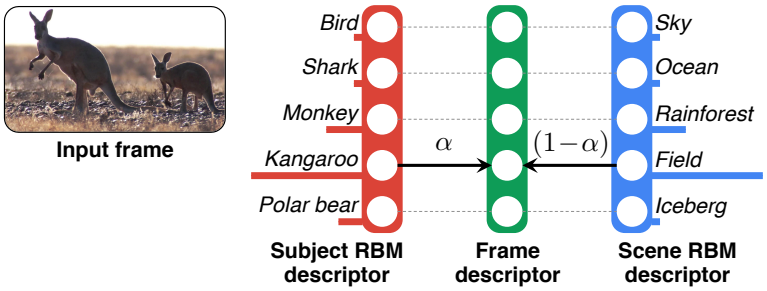
To create the video summaries from the CNN descriptions of subjects $\mathbf{x}_o$ and scenes $\mathbf{x}_p$, we introduce a pair of concurrently trained restricted Boltzmann machines (RBM) to learn their projections ($\mathbf{z}_o$ and $\mathbf{z}_p$) to K units each, where K is the desired number of keyframes. An RBM is a bipartite network with a projection matrix $\mathbf{W}$ which maps between its input and output units. RBM are trained through gradient descent on the approximate maximum likelihood objective, based on network states drawn from Gibbs sampling [61, 62].

In this work, we introduce co-regularization for RBM. The object RBM is regularized by place representations and in turn regularizes the training of the place RBM (Figure B.2(a)). Given randomly sampled minibatches of subject and scene CNN descriptors $\{\mathcal{X}_o^i, \mathcal{X}_p^i\}_i$, we introduce co-regularization cross entropy penalties to the RBM objective functions:

$$\arg\min_{\mathbf{W}_o} -\sum_i \sum_{\mathbf{z}_o^i \in \mathcal{Z}_o^i} \left(\log \mathbb{P}(\mathbf{x}_o^i, \mathbf{z}_o^i) - \lambda_o \sum_k \log \mathbb{P}(\hat{z}_{p,k}^i \mid \hat{z}_{o,k}^i) \right), \tag{B.1}$$



(a) Training co-regularized RBM.



(b) Generating a frame descriptor from co-regularized RBM.

Figure B.2: A pair of co-regularized RBM – one representing subjects and another representing scenes – are learned concurrently.
(a) During training, a subject unit is regularized by its corresponding scene unit and vice versa. (b) The frame descriptor is a linear combination of the two co-regularized RBM descriptors forming relevant subject-scene associations.

$$\arg\min_{\mathbf{W}_p} -\sum_i \sum_{\mathbf{z}_p^i \in \mathcal{Z}_p^i} \left(\log \mathbb{P}(\mathbf{x}_p^i, \mathbf{z}_p^i) - \lambda_p \sum_k \log \mathbb{P}(\hat{z}_{o,k}^i \mid \hat{z}_{p,k}^i) \right), \tag{B.2}$$

where $\{\mathcal{Z}_o^i, \mathcal{Z}_p^i\}_i$ are the RBM projections of $\{\mathcal{X}_o^i, \mathcal{X}_p^i\}_i$, $\{\lambda_o, \lambda_p\}$ are the regularization constants, and $\{\hat{z}_{o,k}^i, \hat{z}_{p,k}^i\}_i$ refer to unit k in the distribution-sparsified representations of the minibatch [44]. Sparsity across units helps avoid co-adaptation between the units and improves representational diversity across instances of frames. The co-regularization terms serve the purpose of binding a subject and the scene in which it occurs to the same unit position.

The frame descriptor is a linear combination of the two RBM descriptors (Figure B.2(b)). The final set of keyframe timings $t_k, k \in [1..K]$ is the ordered set of K timings that gives the maximum response for each unit of the frame descriptor:

$$\arg\max_t \quad \alpha\, z_{o,k}^t + (1-\alpha) z_{p,k}^t, \tag{B.3}$$

where $\alpha \in [0, 1]$ is a balance hyperparameter that causes the summary to be more subject-centric or scene-centric.

This proposed co-regularization method is not specific to subjects or scenes, and is generalizable to other concepts or modalities, such as faces or activities.

B.3 Video Summarization

Using our method, we summarized all 11 episodes from the BBC educational TV series *Planet Earth*[1]. Each episode is approximately 50 minutes long. A sample of our results is shown in Figure B.5(a).

B.3.1 Model Visualization

B.3.1.1 Balancing Subject- and Scene-Centricity

As shown in Figure B.3, bias towards subject or scenes can be adjusted by tuning the α parameter from Equation B.3. This flexibility allows for interesting functionalities such as customising content based on user profiling or explicit queries. The choice of α value can also be made independently for each unit in order to generate the most visually attractive keyframe, for example based on vibrancy. In practice, setting the default value to $\alpha = 0.5$ (as used in this empirical study) seems to produce satisfactory results.

[1] http://www.bbc.co.uk/programmes/b006mywy

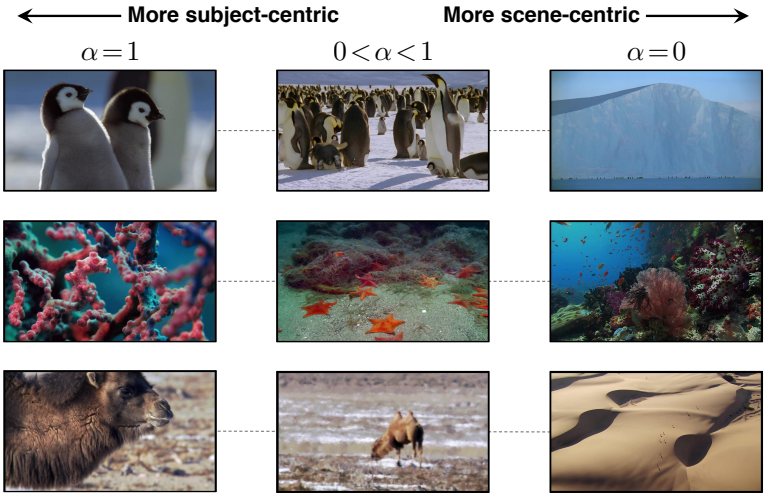


Figure B.3: Actual keyframes selected by varying α. Our model can be tuned to select keyframes that are more subject-centric (left), scene-centric (right) or a balance of both (middle).

B.3.1.2 Visualisation of Co-Regularized RBM Units

Although neural networks tend to be thought of as black boxes, visualization is often useful to decipher what has been learned [167]. To better understand our co-regularized model, we analysed the responses of each unit across the data set. For this analysis, we trained a single $K = 12$ model across all 11 episodes. For each of the 24 RBM units, the top 100 frames that most strongly activate each unit were aggregated via a weighted average. The resulting graphical representation of each unit is shown on Figure B.4. We observe that the visual appearances of frames corresponding to a subject-scene pair of units are consistently similar. There is also diversity across the units within an RBM.

The top 2 categories of each unit identified from the weight matrices are also shown in Figure B.4. We notice that the correlation with the visual representation is strong and the subject-scene association is sensibly learned. We can also observe an interesting effect of co-regularization, where associations can be made between subjects (e.g. *polar bear* and *king penguin*) that occur in the same scene (*iceberg*) but never within the same frame.

B.3.2 User Engagement Study

B.3.2.1 Evaluation Framework

Our method is compared against three other keyframe-based summarisation schemes: naive uniform sampling, k-means clustering and the method currently in use by the

Figure B.4: Visualization of the units for a $K = 12$ model. The visual representations of subject-scene pairs are well correlated. The categories of the two models are associated in a sensible way and correspond well with the visual representations.

video sharing website *Dailymotion*[2]. Each summary is presented as a timeline of keyframes as shown on Figure B.5.

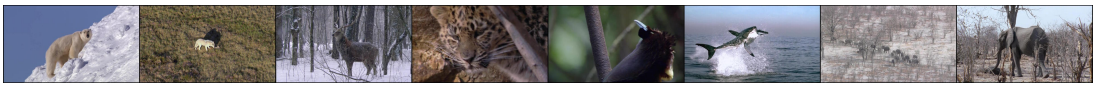
Uniform sampling takes k keyframes with evenly spaced timestamps: $t_i = \frac{d}{k}\left(\frac{1}{2} + i\right), i \in [1..K]$ where d is the total duration of the video. The k-means clustering scheme uses frames sampled at the same frequency as for our method (1 *fps*) and down-sized to 32×32 RGB pixels. Lloyd's algorithm [97] is used to separate the data into K clusters. 100 runs with different centroid seeds are performed to mitigate the effects of local minima. For each cluster, the frame closest to its centroid is selected as keyframe. *Dailymotion* proposes an 8-keyframes video summary (excluding title frame) which was used as a blackbox scheme to compare our method against. The evaluation videos were uploaded on the website and the proposed summary keyframes were then handpicked from the original footages.

The study was performed by showing pairs of summaries – our method against one of the three baseline schemes – to eight different testers who have not previously seen the videos. For each pair, they are asked to answer the two following questions:

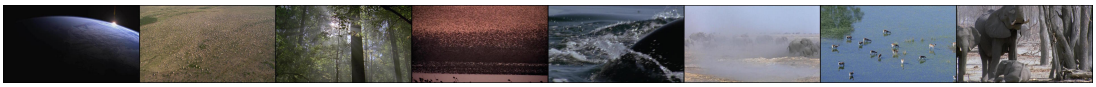
- Q1: *Which video would you rather watch? (attractiveness)*
- Q2: *Which summary was more informative? (informativeness)*

Using all the 11 *Planet Earth* episodes, summaries were generated for different

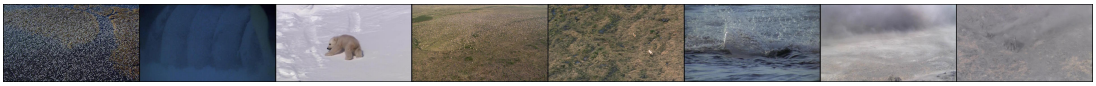
[2] http://www.dailymotion.com/



(a) Our method



(b) Uniform sampling



(c) k-means

(d) Dailymotion

Figure B.5: Eight keyframes summaries for episode 1 from the TV series *Planet Earth*.

amount of keyframes $K = 4, 6, 8$, except for *Dailymotion* which imposes $K = 8$ by default. In total, $8 \times 11 \times 2 \times 3 \times +8 \times 11 = 616$ answers were collected for each question.

Uniform sampling appears as a natural choice for the wildlife documentaries used during this study given the slow pace of the action and high visual appeal of the average frame. K-means is expected to be able to capture the diversity of the scenes well, whereas it may not perform as well with respect to subjects.

B.3.2.2 Results and Discussion

Table B.1 aggregates the answers from the testers. Overall, our method was systematically found more attractive (75% to 97.73% of the time) and more informative (76.14% to 94.32%). Perceived attractiveness and informativeness are strongly correlated. Against *Dailymotion*'s algorithm, our method scores favourably more than three times out of four representing a marked improvement over the scheme currently used by the service.

	uniform			k-means			daily.
K	4	6	8	4	6	8	8
Q1	79.55	82.95	76.14	97.73	82.95	75.00	77.27
Q2	78.41	80.68	81.82	94.32	80.68	76.14	78.41

Table B.1: How often our method is preferred over each of the three schemes (percentage) for different K.

For varying amounts K of keyframes, the improvement is rather consistent against

uniform sampling whereas against k-means, the improvement is more pronounced when K is smaller. This is an indication that our overall sucessful method is particularly well-suited for compact summaries.

B.4 Conclusions

Building upon recent advances in deep learning and image recognition, we proposed a comprehensive keyframe-based summarization framework combining CNN and RBM. Through a comprehensive empirical study, we showed that our method is able to outperform a number of existing schemes. In addition, our novel co-regularization scheme, which discovered meaningful subject-scene associations, is generalizable to other concepts and modalities.

Bibliography

[1] P. Agrawal, R. Girshick, and J. Malik. Analyzing the performance of multilayer neural networks for object recognition. In *European Conference on Computer Vision (ECCV)*, pages 329–344. Springer, 2014. 3.4

[2] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio. Magic materials: a theory of deep hierarchical architectures for learning sensory representations. *CBCL paper*, 2013. 6, 6.2

[3] F. Anselmi, J. Z. Leibo, L. Rosasco, J. Mutch, A. Tacchetti, and T. Poggio. Unsupervised learning of invariant representations in hierarchical architectures. *arXiv preprint arXiv:1311.4158*, 2013. 6, 6.2, 6.2.1

[4] F. Anselmi and T. Poggio. Representation learning in sensory cortex: a theory. Technical report, Center for Brains, Minds and Machines (CBMM), 2014. 6, 6.2

[5] R. Arandjelovic and A. Zisserman. All about vlad. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1578–1585, 2013. 2.1.1, 2.1.1, 4.3.3

[6] Y. Avrithis and G. Tolias. Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval. *International Journal of Computer Vision (IJCV)*, 107(1):1–19, 2014. 2.1

[7] H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson. From generic to specific deep representations for visual recognition. In *Computer Vision and Pattern Recognition Workshops (CVPR)*, pages 36–45, 2015. 2.1.1, 2.1.1, 6.2.5, 6.2.5

[8] A. Babenko and V. Lempitsky. Aggregating local deep features for image retrieval. In *International Conference on Computer Vision (ICCV)*, pages 1269–1277, 2015. 2.1.1, 2.1.1, 6.2.5, 6.2.5

[9] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky. Neural codes for image retrieval. In *European Conference on Computer Vision (ECCV)*, pages 584–599. Springer, 2014. 2.1.1, 2.1.1, 2.1.1, 4.3.1, 5.3.1, 5.3.2, 6.2.5, 6.2.5, 6.3.3

[10] Y. Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009. 2.2.3

[11] Y.-L. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *International Conference on Machine Learning (ICML)*, pages 111–118, 2010. 3.2.1

[12] J. Bromley, I. Guyon, Y. LeCun, E. Sackinger, and R. Shah. Signature verification using a Siamese time delay neural network. In J. Cowan and G. Tesauro, editors, *Neural Information Processing Systems (NIPS)*, volume 6. Morgan Kaufmann, 1993. 5.1

[13] V. Chandrasekhar, J. Lin, O. Morère, A. Veillard, and H. Goh. Compact global descriptors for visual search. In *Data Compression Conference (DCC)*, pages 333–342. IEEE, 2015. (document), 5.2.3

[14] V. Chandrasekhar, M. Makar, G. Takacs, D. Chen, S. S. Tsai, N.-M. Cheung, R. Grzeszczuk, Y. Reznik, and B. Girod. Survey of sift compression schemes. In *Mobile Multimedia Processing Workshop (MMP)*, pages 35–40. Citeseer, 2010. 2.1.2

[15] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, R. Grzeszczuk, and B. Girod. Chog: Compressed histogram of gradients a low bit-rate feature descriptor. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2504–2511. IEEE, 2009. 2.1.2

[16] V. R. Chandrasekhar, D. M. Chen, S. S. Tsai, N.-M. Cheung, H. Chen, G. Takacs, Y. Reznik, R. Vedantham, R. Grzeszczuk, J. Bach, et al. The Stanford mobile visual search data set. In *Multimedia Systems (MMSys)*, pages 117–122. ACM, 2011. A.5

[17] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman. Return of the devil in the details: Delving deep into convolutional nets. *British Machine Vision Conference (BVMC)*, 2014. 3.3.3, 3.1, 3.3.2, 3.3.3

[18] D. Chen, S. Tsai, V. Chandrasekhar, G. Takacs, H. Chen, R. Vedantham, R. Grzeszczuk, and B. Girod. Residual enhanced visual vectors for on-device image matching. *Conference Record of the Forty Fifth Asilomar Conference on Signals, Systems and Computers (ASILOMAR)*, pages 850–854, 2011. 2.1.1

[19] D. Chen, S. Tsai, V. Chandrasekhar, G. Takacs, R. Vedantham, R. Grzeszczuk, and B. Girod. Residual enhanced visual vector as a compact signature for mobile visual search. *Signal Processing (SIGPRO)*, 93(8):2316–2327, 2013. 2.1.1, 2.1.1

[20] D. M. Chen and B. Girod. A hybrid mobile visual search system with compact global signatures. *Transactions on Multimedia (TMM)*, 17(7):1019–1030, 2015. 2.1

[21] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 539–546. IEEE, 2005. 5.1, 5.3.1

[22] A. Choromanska, M. Henaff, M. Mathieu, G. Ben Arous, and Y. LeCun. The loss surfaces of multilayer networks. In *Artificial Intelligence and Statistics (AISTATS)*, pages 192–204, 2015. 2.2.1, 2.2.3

[23] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *Neural Information Processing Systems Workshop (NIPS)*, 2011. 3.3.2

[24] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *Acoustics, Speech and Signal Processing (ICASSP)*, pages 8609–8613. IEEE, 2013. 2.2.3

[25] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. IEEE, 2005. 2.1.1

[26] C. Dao-Duc, H. Xiaohui, and O. Morère. Maritime vessel images classification using deep convolutional neural networks. In *Symposium on Information and Communication Technology (SOICT)*, page 42. ACM, 2015. (document)

[27] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni. Locality-sensitive hashing scheme based on p-stable distributions. In *Annual Symposium on Computational Geometry (SoCG)*, pages 253–262. ACM, 2004. 2.1.1, 2.1.2, 6.3

[28] Y. N. Dauphin, R. Pascanu, C. Gulcehre, K. Cho, S. Ganguli, and Y. Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Neural Information Processing Systems (NIPS)*, pages 2933–2941, 2014. 2.2.1

[29] S. E. F. De Avila, A. P. B. Lopes, A. da Luz, and A. de Albuquerque Araújo. VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method. *Pattern Recognition Letters*, 32(1):56–68, 2011. B.1

[30] 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 (CVPR)*, pages 248–255. IEEE, 2009. 3.1, 4.2.2, B.2.1

[31] L.-Y. Duan, R. Ji, Z. Chen, T. Huang, and W. Gao. Towards mobile document image retrieval for digital library. *Transactions on Multimedia (TMM)*, 16(2):346–359, 2014. 2.1

[32] L.-Y. Duan, J. Lin, Z. Wang, T. Huang, and W. Gao. Weighted component hashing of binary aggregated descriptors for fast visual search. *Transactions on Multimedia (TMM)*, 17(6):828–842, 2015. 2.1

[33] D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research (JMLR)*, 11:625–660, 2010. 2.2.3

[34] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2.1, 5.2.2

[35] G. Francini, S. Lepsøy, and M. Balestri. Selection of local features for visual search. *Signal Processing: Image Communication (SPIC)*, 28(4):311–322, 2013. 4.2.1

[36] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini. VISTO: visual storyboard for web video browsing. In *Conference on Image and Video Retrieval (CIVR)*, pages 635–642. ACM, 2007. B.1

[37] M. Furini, F. Geraci, M. Montangero, and M. Pellegrini. Stimo: Still and moving video storyboard for the web scenario. *Multimedia Tools and Applications (MTA)*, 46(1):47–69, 2010. B.1

[38] T. Ge, Q. Ke, and J. Sun. Sparse-coded features for image retrieval. In *British Machine Vision Conference (BMVC)*, 2013. 4.3.3

[39] J. Ghosh, Y. J. Lee, and K. Grauman. Discovering important people and objects for egocentric video summarization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1346–1353. IEEE, 2012. B.1

[40] R. Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015. 6.2.5

[41] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 580–587. IEEE, 2014. 2.1.1

[42] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Artificial Intelligence and Statistics (AISTATS)*, pages 249–256, 2010. 2.2.3

[43] H. Goh, N. Thome, and M. Cord. Biasing restricted Boltzmann machines to manipulate latent selectivity and sparsity. In *Neural Information Processing Systems Workshop (NIPS)*, 2010. 2.2.2

[44] H. Goh, N. Thome, M. Cord, and J.-H. Lim. Unsupervised and supervised visual codes with restricted Boltzmann machines. In *European Conference on Computer Vision (ECCV)*, pages 298–311. Springer, 2012. 5.2.1, 5.2.1, B.2.2

[45] Y. Gong, S. Kumar, H. Rowley, and S. Lazebnik. Learning binary codes for high-dimensional data using bilinear projections. In *Computer Vision and Pattern Recognition (CVPR)*, pages 484–491, 2013. 2.1.2, 5.2.2, 5.2.3, 6.3

[46] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. In *Computer Vision and Pattern Recognition (CVPR)*, pages 817–824. IEEE, 2011. 2.1.1, 2.1.2, 5.4.2, 6.3

[47] Y. Gong and S. Lazebnik. Iterative quantization: A procrustean approach to learning binary codes. *Computer Vision and Pattern Recognition (CVPR)*, pages 817–824, 2011. 5.2.2

[48] Y. Gong, L. Wang, R. Guo, and S. Lazebnik. Multi-scale orderless pooling of deep convolutional activation features. In *European Conference on Computer Vision (ECCV)*, pages 392–407. Springer, 2014. 2.1.1, 2.1.1, 6.2.5, 6.2.5

[49] I. Goodfellow, Y. Bengio, and A. Courville. Deep learning. Book in preparation for MIT Press, 2016. 2.2

[50] I. Goodfellow, D. Warde-farley, M. Mirza, A. Courville, and Y. Bengio. Maxout networks. In *International Conference on Machine Learning (ICML)*, pages 1319–1327, 2013. 3.2.3

[51] A. Gordo, F. Perronnin, Y. Gong, and S. Lazebnik. Asymmetric distances for binary embeddings. *Pattern Analysis and Machine Intelligence (PAMI)*, 36(1):33–47, 2014. 6.3.3, 6.3

[52] A. Gordo, J. A. Rodríguez-Serrano, F. Perronnin, and E. Valveny. Leveraging category-level labels for instance-level image retrieval. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3045–3052. IEEE, 2012. 4.3.3, 6.2.5, 6.2.5

[53] K. Grauman and R. Fergus. Learning binary hash codes for large-scale image search. In *Machine Learning for Computer Vision*, pages 49–87. Springer, 2013. 2.1.2

[54] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool. Creating summaries from user videos. In *European Conference on Computer Vision (ECCV)*, pages 505–520. Springer, 2014. B.1

[55] Y. Hadi, F. Essannouni, and R. O. H. Thami. Video summarization by k-medoid clustering. In *Symposium on Applied Computing (SAC)*, pages 1400–1401. ACM, 2006. B.1

[56] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 1735–1742. IEEE, 2006. 5.1, 5.3.1, 5.3.1, 5.7, 5.9

[57] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *ArXiv e-prints*, Dec. 2015. 2.2.3, 3.1, 3.2.3, 3.2.3

[58] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *International Conference on Computer Vision (CVPR)*, pages 1026–1034, 2015. 2.2.3, 3.2.3, 3.2.3, 3.2.3

[59] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon. Spherical hashing. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2957–2964. IEEE, 2012. 2.1.2

[60] G. Hinton. A practical guide to training restricted Boltzmann machines. *Momentum*, 9(1):926, 2010. 2.2.2

[61] G. E. Hinton. Training products of experts by minimizing contrastive divergence. *Neural Computation (NC)*, 14(8):1771–1800, 2002. 2.2.2, B.2.2

[62] G. E. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural Computation (NC)*, 18(7):1527–1554, 2006. 5.2.1, B.2.2

[63] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006. 2.1.1, 2.1.2, 2.2.2, 2.2.2

[64] G. E. Hinton and T. J. Sejnowski. Learning and relearning in Boltzmann machines. *Parallel distributed processing: Explorations in the microstructure of cognition*, 1:282–317, 1986. 2.2.2

[65] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012. 2.2.3

[66] E. Hoffer and N. Ailon. Deep metric learning using triplet network. In *International Conference on Learning Representations (ICLR)*, 2015. 5.4, 5.4.1

[67] G. Huang, Y. Sun, Z. Liu, D. Sedra, and K. Weinberger. Deep networks with stochastic depth. *arXiv preprint arXiv:1603.09382*, 2016. 2.2.3, 3.2.3

[68] M. J. Huiskes, B. Thomee, and M. S. Lew. New trends and ideas in visual concept detection: the mir flickr retrieval evaluation initiative. In *Multimedia Information Retrieval (MIR)*, pages 527–536. ACM, 2010. 6.3, A.4

[69] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015. 2.2.3, 3.2.3

[70] H. Jégou and O. Chum. Negative evidences and co-occurences in image retrieval: The benefit of pca and whitening. In *European Conference on Computer Vision (ECCV)*, pages 774–787. Springer, 2012. 2.1.1, 2.1.1

[71] H. Jégou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. In A. Z. David Forsyth, Philip Torr, editor, *European Conference on Computer Vision (ECCV)*, volume I of *LNCS*, pages 304–317. Springer, oct 2008. A.3

[72] H. Jégou, M. Douze, and C. Schmid. On the burstiness of visual elements. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1169–1176. IEEE, 2009. 2.1.1

[73] H. Jégou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3304–3311. IEEE, 2010. 2.1.1, 2.1.1, 4.3.3, 5.2.2

[74] H. Jégou, F. Perronnin, M. Douze, J. Sanchez, P. Perez, and C. Schmid. Aggregating local image descriptors into compact codes. *Pattern Analysis and Machine Intelligence (PAMI)*, 34(9):1704–1716, 2012. 2.1.1, 2.1.1, 2.1.1, 2.1.2, 4.2.1, 4.3.3, 6.3.3, 6.3

[75] H. Jégou and A. Zisserman. Triangulation embedding and democratic aggregation for image search. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3310–3317. IEEE, 2014. 2.1.1, 2.1.1, 4.3.3, 4.3.3, 4.3.3, 6.2.5, 6.2.5

[76] R. Ji, L.-Y. Duan, J. Chen, L. Xie, H. Yao, and W. Gao. Learning to distribute vocabulary indexing for scalable visual search. *Transactions on Multimedia (TMM)*, 15(1):153–166, 2013. 2.1

[77] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Conference on Multimedia (ACMMM)*, pages 675–678. ACM, 2014. 2.2.3, 3.3.2, 4.2.2, 5.2.2, B.2.1

[78] Y.-G. Jiang, J. Wang, X. Xue, and S.-F. Chang. Query-adaptive image search with hash codes. *Transactions on Multimedia (TMM)*, 15(2):442–453, 2013. 2.1

[79] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems (NIPS)*, pages 1–9, 2012. 2.1.1, 3.1, 3.2, 3.2.1, 3.9, 3.3.1, 3.3.2, 4.2.2, 5.2.2, 6.2.2, B.2.1

[80] B. Kulis and K. Grauman. Kernelized locality-sensitive hashing for scalable image search. In *International Conference on Computer Vision (ICCV)*, pages 2130–2137. IEEE, 2009. 2.1.1, 2.1.2

[81] R. Laganière, R. Bacco, A. Hocevar, P. Lambert, G. Païs, and B. E. Ionescu. Video summarization from spatio-temporal features. In *TRECVid Video Summarization Workshop*, pages 144–148. ACM, 2008. B.1

[82] H. Lai, Y. Pan, Y. Liu, and S. Yan. Simultaneous feature learning and hash coding with deep neural networks. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3270–3278, 2015. 5.4, 5.4.1

[83] H. Larochelle and Y. Bengio. Classification using discriminative restricted Boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 536–543. ACM, 2008. 2.2.2

[84] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation (NC)*, 1(4):541–551, 1989. 2.2.3, 3.2.1

[85] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 3.2.1

[86] Y. LeCun, L. Jackel, L. Bottou, A. Brunot, C. Cortes, J. Denker, H. Drucker, I. Guyon, U. Muller, E. Sackinger, et al. Comparison of learning algorithms for handwritten digit recognition. In *International Conference on Artificial Neural Networks (ICANN)*, volume 60, pages 53–60, 1995. 3.2.1

[87] C.-Y. Lee, P. W. Gallagher, and Z. Tu. Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree. *Artificial Intelligence and Statistics (AISTATS)*, 2016. 3.2.3

[88] H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area v2. In *Neural Information Processing Systems (NIPS)*, pages 873–880, 2008. 2.2.2

[89] X. Li, G. Lin, C. Shen, A. Van den Hengel, and A. Dick. Learning hash functions using column generation. In *International Conference on Machine Learning (ICML)*, pages 142–150, 2013. 2.1.1, 2.1.2

[90] Q. Liao, J. Z. Leibo, and T. Poggio. Learning invariant representations and applications to face verification. In *Neural Information Processing Systems (NIPS)*, pages 3057–3065, 2013. 6.2.1

[91] J. Lin, L.-Y. Duan, T. Huang, and W. Gao. Robust Fisher codes for large scale image retrieval. In *International Conference on Acoustics and Signal Processing (ICASSP)*, 2013. 2.1.1, 2.1.1, 2.1.2, 5.2.2

[92] M. Lin, Q. Chen, and S. Yan. Network in network. *arXiv preprint arXiv:1312.4400v3*, pages 1–10, 2014. 3.2.3

[93] D. Liu, G. Hua, and T. Chen. A hierarchical visual model for video object summarization. *Pattern Analysis and Machine Intelligence (PAMI)*, 32(12):2178–2190, 2010. B.1

[94] L. Liu. CCV-a modern computer vision library. Accessed: 2015-12-31. 3.3.2

[95] T. Liu, H.-J. Zhang, and F. Qi. A novel video key-frame-extraction algorithm based on perceived motion energy model. *Circuits and Systems for Video Technology (CSVT)*, 13(10):1006–1013, 2003. B.1

[96] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang. Supervised hashing with kernels. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2074–2081. IEEE, 2012. 2.1.2

[97] S. P. Lloyd. Least squares quantization in PCM. *Transactions on Information Theory (TIT)*, 28(2):129–137, 1982. B.3.2.1

[98] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, November 2004. 2.1.1, 4.3.5, 5.2.2

[99] D. G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157. IEEE, 1999. 2.1.1

[100] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision (IJCV)*, 60(2):91–110, Nov. 2004. 2.1.1

[101] Z. Lu and K. Grauman. Story-driven summarization for egocentric video. In *Computer Vision and Pattern Recognition (CVPR)*, pages 2714–2721, 2013. B.1

[102] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning (ICML)*, volume 30, 2013. 3.2.3

[103] K. Mikolajczyk. *Software for Computing Hessian-affine Interest Points and SIFT Descriptor*, 2010. 2.1.1, 4.3.3

[104] O. Morère, V. Chandrasekhar, J. Lin, H. Goh, and A. Veillard. A practical guide to CNNs and Fisher vectors for image instance retrieval. *Signal Processing (SIGPRO)*, 2016. (document)

[105] O. Morère, H. Goh, and A. Veillard. Large scale image classification on a shoe string. In *European Conference on Computer Vision Workshop (ECCV)*, 2014. (document), 3.3

[106] O. Morère, H. Goh, A. Veillard, V. Chandrasekhar, and J. Lin. Co-regularized deep representations for video summarization. In *International Conference on Image Processing (ICIP)*, pages 3165–3169. IEEE, 2015. (document)

[107] O. Morère, J. Lin, V. Chandrasekhar, A. Veillard, and H. Goh. Deephash: Getting regularization, depth and fine-tuning right. *arXiv preprint arXiv:1501.04711*, 2015. (document)

[108] O. Morère, J. Lin, V. Chandrasekhar, A. Veillard, and H. Goh. Co-sparsity regularized deep hashing for image instance retrieval. In *International Conference on Image Processing (ICIP)*. IEEE, 2016. (document)

[109] O. Morère, J. Lin, J. Petta, V. Chandrasekhar, and A. Veillard. Tiny descriptors for image retrieval with unsupervised triplet hashing. *Data Compression Conference (DCC)*, 2016. (document)

[110] O. Morère, A. Veillard, and H. Goh. Kaggle national data science bowl challenge, team LateFusion. https://www.kaggle.com/t/142299/latefusion, 2015. (document)

[111] O. Morère, A. Veillard, J. Lin, J. Petta, V. Chandrasekhar, and T. Poggio. Group invariant deep representations for image instance retrieval. Technical report, Center for Brains, Minds and Machines (CBMM), 2016. (document)

[112] P. Mundur, Y. Rao, and Y. Yesha. Keyframe-based video summarization using delaunay clustering. *International Journal on Digital Libraries (IJDL)*, 6(2):219–232, 2006. B.1

[113] V. Nair and G. E. Hinton. 3D object recognition with deep belief nets. In *Neural Information Processing Systems (NIPS)*, pages 1339–1347, 2009. 2.1.1, 2.1.2, 2.2.2, 5.2.1, 5.2, 5.2.1

[114] V. Nair and G. E. Hinton. Rectified linear units improve restricted Boltzmann machines. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2010. 3.2.1

[115] J. Nam and A. H. Tewfik. Event-driven video abstraction and visualization. *Multimedia Tools and Applications (MTA)*, 16(1-2):55–77, 2002. B.1

[116] C.-W. Ngo, Y.-F. Ma, and H.-J. Zhang. Video summarization and scene detection by graph modeling. *Circuits and Systems for Video Technology (CSVT)*, 15(2):296–305, 2005. B.1

[117] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 2161–2168. IEEE, 2006. 4.3.3, 4.3.3, A.1

[118] M. Norouzi and D. M. Blei. Minimal loss hashing for compact binary codes. In *International Conference on Machine Learning (ICML)*, pages 353–360, 2011. 2.1.1, 2.1.2

[119] W. Ouyang and X. Wang. Joint deep learning for pedestrian detection. In *International Conference on Computer Vision (ICCV)*, pages 2056–2063, 2013. 2.1.1

[120] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007. 2.1.1, 2.1.1, 2.1.1

[121] F. Perronnin and D. Larlus. Fisher vectors meet neural networks: A hybrid classification architecture. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3743–3752, 2015. 6.2.5, 6.2.5

[122] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3384–3391. IEEE, 2010. 2.1.1, 2.1.1, 2.1.1, 2.1.1, 2.1.2, 4.2.1, 6.3.3

[123] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2007. 2.1, A.2

[124] A. Quattoni and A. Torralba. Recognizing indoor scenes. In *Computer Vision and Pattern Recognition (CVPR)*, pages 413–420. IEEE, 2009. 4.2.2

[125] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *arXiv preprint arXiv:1511.06434*, 2015. 3.2.3

[126] M. Raginsky and S. Lazebnik. Locality-sensitive binary codes from shift-invariant kernels. In *Neural Information Processing Systems (NIPS)*, pages 1509–1517, 2009. 5.4.2, 6.3

[127] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *Computer Vision and Pattern Recognition Workshop (CVPR)*, pages 806–813, 2014. 2.1.1, 2.1.1, 3.4, 6.2.5, B.2.1

[128] F. Rosenblatt. *The perceptron, a perceiving and recognizing automaton Project Para*. Cornell Aeronautical Laboratory, 1957. 2.2.1

[129] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:9, 1986. 2.2.1

[130] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision (IJCV)*, pages 1–42, 2014. 2.2.3, 3.1, 3.1

[131] R. Salakhutdinov and G. Hinton. Semantic hashing. *International Journal of Approximate Reasoning*, 50(7):969–978, 2009. 2.2.2, 5.2

[132] R. Salakhutdinov, A. Mnih, and G. Hinton. Restricted Boltzmann machines for collaborative filtering. In *International Conference on Machine Learning (ICML)*, pages 791–798. ACM, 2007. 2.1.2, 2.2.2

[133] J. Sánchez, F. Perronnin, T. Mensink, and J. Verbeek. Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision (IJCV)*, 105(3):222–245, 2013. 2.1.1, 2.1.1

[134] D. Scherer, A. Müller, and S. Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In *International Conference on Artificial Neural Networks (ICANN)*, pages 92–101. Springer, 2010. 3.2.1

[135] P. Sermanet, D. Eigen, and X. Zhang. OverFeat: Integrated recognition, localization and detection using convolutional networks. *International Conference on Learning Representations (ICLR)*, 2014. 3.3.2

[136] A. Sharif Razavian, J. Sullivan, A. Maki, and S. Carlsson. A baseline for visual instance retrieval with deep convolutional networks. In *International Conference on Learning Representations (ICLR)*. ICLR, 2015. 2.1.1, 2.1.1, 6.2.5, 6.2.5, 6.3.3

[137] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015. 2.1.1, 3.9, 3.2.3, 4.2.2, 4.2.2, 6.2.2, 6.2.4, B.2.1

[138] P. Smolensky. Information processing in dynamical systems: foundations of harmony theory. In *Parallel distributed processing: explorations in the microstructure of cognition*, pages 194–281. MIT Press, 1986. 2.2.2

[139] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for simplicity: The all convolutional net. *International Conference on Learning Representations Workshop (ICLR)*, 2015. 3.2.3

[140] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research (JMLR)*, 15(1):1929–1958, 2014. 2.2.3

[141] Stanford University. *MPEG CDVS (Compact Descriptors for Visual Search) Benchmark. Stanford Digital Repository.* A.5

[142] Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1891–1898, 2014. 2.1.1

[143] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In *International Conference on Machine Learning (ICML)*, pages 1139–1147, 2013. 2.2.3

[144] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015. 3.1, 3.9, 3.2.3

[145] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1701–1708, 2014. 2.1.1

[146] G. Takacs, V. Chandrasekhar, S. Tsai, D. Chen, R. Grzeszczuk, and B. Girod. Unified real-time tracking and recognition with rotation-invariant fast features. In *Computer Vision and Pattern Recognition (CVPR)*, pages 934–941. IEEE, 2010. 4.3.4

[147] G. Tolias, Y. Avrithis, and H. Jégou. To aggregate or not to aggregate: Selective match kernels for image search. In *International Conference on Computer Vision (ICCV)*, pages 1401–1408, 2013. 2.1.1, 2.1.1

[148] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of CNN activations. In *International Conference on Learning Representations (ICLR)*, 2016. 2.1.1, 2.1.1, 6.2.5, 6.2.5

[149] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1–8. IEEE, 2008. 2.1.2

[150] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1653–1660, 2014. 2.1.1

[151] A. Vedaldi and B. Fulkerson. Vlfeat: An open and portable library of computer vision algorithms. In *Conference on Multimedia (ACMMM)*, pages 1469–1472. ACM, 2010. 4.2.1, 4.3.3

[152] L. Wan, M. D. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *International Conference on Machine Learning (ICML)*, pages 1058–1066, 2013. 2.2.3

[153] J. Wang, S. Kumar, and S.-F. Chang. Semi-supervised hashing for scalable image retrieval. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3424–3431. IEEE, 2010. 2.1.2

[154] J. Wang, W. Liu, A. Sun, and Y.-G. Jiang. Learning hash codes with listwise supervision. In *International Conference on Computer Vision (ICCV)*, pages 3032–3039, 2013. 2.1.1, 2.1.2

[155] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu. Learning fine-grained image similarity with deep ranking. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1386–1393, 2014. 5.4, 5.4.1

[156] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Neural Information Processing Systems (NIPS)*, pages 1753–1760, 2009. 2.1.1, 2.1.2, 5.4.2, 6.3

[157] W. Wolf. Key frame selection by motion analysis. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 2, pages 1228–1231. IEEE, 1996. B.1

[158] H. Wu and X. Gu. Max-pooling dropout for regularization of convolutional neural networks. In *International Conference on Neural Information Processing (ICONIP)*, pages 46–54. Springer, 2015. 3.2.3

[159] R. Wu, S. Yan, Y. Shan, Q. Dang, and G. Sun. Deep image: Scaling up image recognition. *arXiv preprint arXiv:1501.02876*, 2015. 2.2.3

[160] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3485–3492. IEEE, 2010. 4.2.2

[161] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *International Conference on Machine Learning Workshop (ICML)*, 2015. 3.2.3

[162] F. Yang, Z. Jiang, and L. S. Davis. Submodular reranking with multiple feature modalities for image retrieval. In *Asian Conference Computer Vision (ACCV)*, pages 19–34. Springer, 2014. 4.3.3

[163] C. Yeo, P. Ahammad, and K. Ramchandran. Rate-efficient visual correspondences using random projections. In *International Conference on Image Processing (ICIP)*, pages 217–220. IEEE, 2008. 5.2.2

[164] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Neural Information Processing Systems (NIPS)*, pages 3320–3328, 2014. 3.4

[165] D. Yu, H. Wang, P. Chen, and Z. Wei. Mixed pooling for convolutional neural networks. In *Rough Sets and Knowledge Technology (RSKT)*, pages 364–375. Springer, 2014. 3.2.3

[166] M. D. Zeiler and R. Fergus. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*, pages 1–9, 2013. 3.2.3

[167] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision (ECCV)*, pages 818–833. Springer, 2014. 3.2.2, 3.8, 3.3.2, 3.4, B.3.1.2

[168] C. Zhang, G. Evangelopoulos, S. Voinea, L. Rosasco, and T. Poggio. A deep representation for invariance and music classification. In *Acoustics, Speech and Signal Processing (ICASSP)*, pages 6984–6988. IEEE, 2014. 6.2.1, 6.2.3

[169] H. J. Zhang, J. Wu, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, 30(4):643–658, 1997. B.1

[170] T. Zhang, C. Du, and J. Wang. Composite quantization for approximate nearest neighbor search. In *International Conference on Machine Learning (ICML)*, pages 838–846, 2014. 6.3.3

[171] T. Zhang, G.-J. Qi, J. Tang, and J. Wang. Sparse composite quantization. In *Computer Vision and Pattern Recognition (CVPR)*, pages 4548–4556, 2015. 6.3.3

[172] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Neural Information Processing Systems (NIPS)*, pages 487–495. Curran Associates, Inc., 2014. 4.2.2, 4.2.2, B.2.1