Handcrafted vs Non-Handcrafted Features

for computer vision classification

Loris Nanni[1], Stefano Ghidoni[1], Sheryl Brahnam[2]

[1]DEI, University of Padua, viale Gradenigo 6, Padua, Italy; E-Mail: {loris.nanni, stefano.ghidoni}@unipd.it;
[2]Computer Information Systems, Missouri State University, 901 S. National, Springfield, MO 65804, USA; E-Mail: sbrahnam@missouristate.edu;

Abstract**—** This work presents a generic computer vision system designed for exploiting trained deep Convolutional Neural Networks (CNN) as a generic feature extractor and mixing these features with more traditional hand-crafted features. Such a system is a single structure that can be used for synthesizing a large number of different image classification tasks. Three substructures are proposed for creating the generic computer vision system starting from handcrafted and non-handcrafter features: i) one that remaps the output layer of a trained CNN to classify a different problem using an SVM; ii) a second for exploiting the output of the penultimate layer of a trained CNN as a feature vector to feed an SVM; and iii) a third for merging the output of some deep layers, applying a dimensionality reduction method, and using these features as the input to an SVM. The application of feature transform techniques to reduce the dimensionality of feature sets coming from the deep layers represents one of the main contributions of this paper. Three approaches are used for the non-handcrafted features: deep transfer learning features based on convolutional neural networks (CNN), principal component analysis network (PCAN), and the compact binary descriptor (CBD). For the handcrafted features, a wide variety of state-of-the-art algorithms are considered: Local Ternary Patterns, Local Phase Quantization, Rotation Invariant Co-occurrence Local Binary Patterns, Completed Local Binary Patterns, Rotated local binary pattern image, Globally Rotation Invariant Multi-scale Co-occurrence Local Binary Pattern, and several others. The computer vision system based on the proposed approach was tested on many different datasets, demonstrating the generalizability of the proposed approach thanks to the strong performance recorded. The Wilcoxon signed rank test is used to compare the different methods; moreover, the independence of the different methods is studied using the Q-statistic. To facilitate replication of our experiments, the MATLAB source code will be available at (https://www.dropbox.com/s/bguw035yrqz0pwp/ElencoCode.docx?dl=0).

Index Terms**—**Deep learning; transfer learning, non-handcrafted features, texture descriptors; texture classification; ensemble of descriptors.

I. INTRODUCTION

Many computer vision algorithms for image classification rely on the detection and extraction of local characteristics in images; as a result, much of the computer vision literature is focused on discovering, understanding, characterizing, and improving features that can be extracted from images. A large number of features reported in the literature have been manually designed, or “handcrafted,” with an eye for overcoming specific issues like occlusions and variations in scale and illumination. The design of handcrafted features often involves finding the right trade-off between accuracy and computational efficiency. The powerful Scale Invariant Feature Transform (SIFT) [1], for example, is well-known for its robustness to object rotation and scale variations, but this robustness comes at a high computational cost. The most widely used features often generate an ensemble of variants to tackle the inherent shortcomings in the speed and/or accuracy of the original versions. For example, some fast variants of SIFT [2] have been proposed to be used in real-time applications and on devices with low computational power.

The computation of handcrafted features, such as SIFT, is normally a two-step process [3]. First, a keypoint detector locates characteristic regions of an image (e.g., corners), which are then characterized by calculating a descriptor that is capable of distinguishing each particular keypoint from the others. This process is outlined in detail in [4]. The set of cues considered for building the descriptor depends on the specific feature being used. At a lower level, a descriptor is a vector of measurements that can be used to train a classifier, such as a Support Vector Machine (SVM) [5].

The deep learning paradigm [6] enables the creation of complex networks for solving (among the others) the problem of image classification, usually tackled by means of CNNs, where deep layers in these complex networks act as a set of feature extractors that are often quite generic and, to some extent, independent of any specific classification task [7]. This means that deep learning obtains a set of features learned directly from observations of the input images [8], possibly preprocessed using a pyramidal approach [9]. The idea behind this approach is to discover multiple levels of representation so that higher level features can represent the semantics of the data, which in turn can provide greater robustness to intra-class variability [10]. Of interest here are the characteristics of the different layers in many deep neural networks that have been trained on images: first layer features resemble either Gabor filters or color blobs and tend to be generalizable, i.e. transferable to many different datasets and tasks [11]. Therefore, it is possible to consider the deep layers of a CNN as a feature extractor, much like SIFT, the major difference being that the features extracted by a CNN are learned using the data in contrast to hand-crafted features that are designed beforehand by human experts to extract a given set of chosen characteristics.

Because the features extracted by the lower levels of a CNN strongly depend on the training set, special care must be taken in the selection of the dataset. The choice of a representative training dataset is usually performed by generating or selecting huge training sets, in the order of millions of images [12]. This solution, however, is far from ideal, as it requires considerable human effort in the selection of images suitable for building the training set and substantial computational power during the training phase. Both these drawbacks can partially be overcome by exploiting semi-supervised and unsupervised techniques [13] and parallel computing to reduce the human labor and the computational costs involved in the training.

In table 1, we provide a brief descriptive summary of several recent deep transfer learning methods, including the approach proposed here for comparison purposes.

Table 1. Descriptive Summary of other deep transfer learning approaches

Paper	**Features extracted using CNN**
here	Using deep layers, shallow layers, and the scores obtained by CNN
[14]	Representation of images as strings of the top layers of pretrained CNNs
[15]	The convolutional layers of a CNN are used as a filter bank
[16]	Representation of images using the seventh fully connected layer of a CNN on ImageNet
[17]	Representation of images using the last convolutional layer of a CNN
[18]	Representation of images using five convolutional layers and two fully connected layers
[19]	Lower-layer features are used
[20]	Features are extracted from the first few layers of a pre-trained CNN model
[21]	Top-layer activations are used as features

In this paper, we are interested in exploiting features extracted by the deep layers of a network to build a general, or General-Purpose (GP), ensemble capable of handling a broad range of image classification problems. An ideal GP system is capable of providing results comparable with state-of-the-art systems by working out-of-the-box with little or no fine-tuning. Some recent papers have investigated how it is possible to build heterogeneous GP ensembles [22, 23]. For example, in [22], a GP ensemble performed competitively compared with other state-of-the-art methods across sixteen benchmark datasets representing very different problems (for instance, numerous medical problems, many image classification problems, a vowel dataset, and a credit card dataset). However, no single ensemble investigated in [22] worked consistently well across all sixteen datasets. Nonetheless, one GP ensemble worked well across all the image datasets—on some datasets performing even better than an SVM whose parameters had been fine-tuned for that specific dataset.

Several papers have also investigated GP ensembles that exploit the different types of information available using different feature extraction methods and representations of the data. In [23], for example, the goal was to discover a GP ensemble for protein classification that combined different types of protein representations and descriptors and that was capable of performing well across fourteen protein classification datasets representing several different classification problems. Again, no single ensemble was discovered that obtained top performance across all fourteen datasets; however, it was shown that it is always possible to find a more limited GP ensemble that performs consistently well across each type of dataset. Finally, in [24], a GP heterogeneous ensemble combining many handcrafted and non-handcrafted features, including features taken from a specific layer of a deep neural network trained on a specific dataset, was able to perform competitively well across twenty-five datasets (fourteen image datasets and eleven UCI data mining datasets). In some cases, it outperformed an SVM classifier for which both the kernel and parameters selection were carefully fine-tuned for the datasets.

Our goal in this paper is to build a better performing GP ensemble by comparing and combining state-of-the-art handcrafted features with non-handcrafted features. The handcrafted approaches considered in this paper include Local Ternary Patterns, Local Phase Quantization, Rotation Invariant Co-occurrence Local Binary Patterns, Completed Local Binary Patterns, Rotated Local Binary Pattern Image, Globally Rotation Invariant Multi-scale Co-occurrence Local Binary Pattern, as well as several others. The non-handcrafted features examined here include two recently proposed methods, the Principal Component Analysis Network (PCAN) [10] and the Compact Binary Descriptor (CBD) [25]), as well as different methods of deep transfer learning. The deep transfer learning methods are based on Convolutional Neural Networks (CNN) trained to perform classification on the ImageNet ILSVRC challenge dataset. The approach chosen for the deep learners exploits the features provided by the last layer of the CNN along with the feature set obtained from an internal layer. Two feature dimensionality reduction methods (discrete cosine transform and principal component analysis) are used for reducing these feature sets since they often have high dimensionality. The features extracted from the CNN are then used to train a Support Vector Machine (SVM), which implements the deep transfer learning approach.

The contributions of this paper are twofold. The first is our proposal of combining features extracted from the deep layers of CNNs with powerful handcrafted approaches to solve a classification problem: in this way we are able to verify that each of the two feature extraction paradigms is capable of extracting information that is neglected by the other paradigm. In order to reduce the extremely large feature vectors that are extracted from deep layers (which could result in the curse of dimensionality [26, 27]), two reduction methods are employed: Discrete Cosine Transform (DCT) and Principal Component Analysis (PCA). The second contribution is our wide and thorough evaluation of the possible combinations of handcrafted and non-handcrafted features that are needed to build a generic computer vision classification system. This can be seen as a detailed guideline for synthesizing the GP computer vision system concept outlined above. Such a guideline is of general interest because the system performance of our approach was evaluated over a mix of very different computer vision classification tasks in order to avoid any polarization of the proposed architecture.

The different handcrafted and non-handcrafted features were compared and combined in this work, with their independence examined using the Q-statistic [28]. The proposed GP ensemble obtains state-of-the-art results on a large set of different datasets. Performance differences among the methods are evaluated using the Wilcoxon signed rank test. The MATLAB source code to replicate our experiments will be available at (https://www.dei.unipd.it/node/2357 +Pattern Recognition and Ensemble Classifiers).

II. DEEP CONVOLUTIONAL NEURAL NETWORKS

Deep learning has revolutionized the field of machine learning. First proposed in 2006 [29], the primary feature of deep learning is its layered structure. Deep learning can be implemented by means of several different algorithms, all of which are marked by a cascade of many processing layers organized in a hierarchical structure. Each of these layers add a level of abstraction to the overall representation. In the image interpretation task, layers close to the input deal with low-level features like edges and texture. These low-level features can be combined together to build a more complex representation. For example, at the second level, features like image patches and contours might be considered. Layer by layer, complexity increases.

The approach to deep learning considered in this paper is based on Convolutional Neural Networks (CNNs) [30]. Several CNN software packages and MATLAB toolboxes are available to researchers. In this work, we use the MatConvNet toolbox for MATLAB,[1] where CNNs are built by choosing a set of *computational blocks* (CBs) (a naming convention employed in the

[1] The toolbox is freely available at http://www.vlfeat.org/matconvnet/

MatConvNet toolbox) that implements the data processing functions. These blocks are connected following a Directed Acyclic Graph (DAG). Each CB takes an input vector $\boldsymbol{x}$ and provides an output vector $\boldsymbol{y}$ that depends on $\boldsymbol{x}$ and on a set of parameters $\boldsymbol{w}$ that are learned during the training phase. The MatConvNet toolbox provides a set of CBs that can be rearranged to form many different networks, thereby exploiting the modularity of the CNN structure.

Below, the deep learning-based approaches considered in this work will be detailed. It should be pointed out that these approaches are used as feature extraction methods where the output of each algorithm is considered as a feature vector and exploited in a fashion similar to how handcrafted features are commonly employed.

A. **Principal component analysis network (PCANet)**

PCANet [10] is a simple deep learning network that is based on cascading PCA to learn multistage filter banks and on binary hashing and blockwise histograms for indexing and pooling. The idea behind PCANet is to produce a basic yet competitive deep learner for image classification that could be used to justify more complex deep learning strategies. There are no nonlinear operations in PCANet until the last output layer, which utilizes the binary hashing and blockwise histograms.

PCANet implements a three stage process. Given a training set of N images $\{I_i\}_{i=1}^{N}$of size $m \times n$ and assuming a patch size or 2D filter of size $k_1 \times k_2$ at all stages, then the three stages can be described as follows:

Stage 1 - PCA: a patch is taken around each pixel, and all overlapping patches of the i-th image are collected: $x_{i,1}, x_{i,2}, \ldots, x_{i,\tilde{m}\tilde{n}} \in \mathcal{R}^{k_1 k_2}$, where each $x_{i,j}$ denotes the j-th vectorized patch in I_i, $\tilde{m} = m - \lceil {}^{k_1}/_{2} \rceil$, $\tilde{n} = n - \lceil {}^{k_2}/_{2} \rceil$. The mean patch is then subtracted from each patch, obtaining $\bar{X}_i = [\bar{x}_{i,1}, \bar{x}_{i,2}, \ldots, \bar{x}_{i,\tilde{m}\tilde{n}}]$, where $\bar{x}_{i,j}$ is a mean-removed patch. Assuming the number of filters in layer i is L_i, PCA minimizes the reconstruction error within a family of orthonormal filters, with the leading principal eigenvectors capturing the main variation of all the mean-removed training patches. The PCA filters are expressed as $W_l^1 \doteq mat_{k_1,k_2}\left(\boldsymbol{q_1}(\boldsymbol{XX^T})\right) \in \mathcal{R}^{k_1 k_2}$, where $\boldsymbol{XX^T}$are the L_1 principal eigenvectors, $mat_{k_1,k_2}(\boldsymbol{v})$ is a function mapping $\boldsymbol{v} \in \mathcal{R}^{k_1 k_2}$ to a matrix $\boldsymbol{W} \in \mathcal{R}^{k_1 k_2}$, and $\boldsymbol{q_1}(\boldsymbol{XX^T})$ is the l-th principal eigenvector of $\boldsymbol{XX^T}$.

Stage 2 - PCA: this is a process similar to stage 1. Let the l-th filter output of stage 1 be I_i^l. As in stage 1, all the overlapping patches of I_i^l can be collected and the patch mean subtracted from each patch. The PCA filters W_l^2 of stage 2 can then be obtained. The number of output images is $L_1 L_2$. Stage 2 can be repeated if need be to build a deeper architecture.

Stage 3 - Output (Hashing and Histograms): each of the L_1input images in I_i^l for stage 2 has L_2 real value outputs $\{I_i^l * W_l^2\}_{l=1}^{L_2}$. These outputs are binarized using a Heaviside step-like function $\{H(I_i^l * W_l^2)\}_{l=1}^{L_2}$, whose value is 1 for positive entries and 0 otherwise. Around each pixel, the binary bits of vector L_2 are considered as a decimal number, thereby converting the L_2 outputs into a single integer-valued "image" T_i^l with every pixel integer in the range $[0, 2^{L_2} - 1]$. Each of the L_1 images are then partitioned into B blocks. A histogram with 2^{L_2} bins of the decimal values in each block is computed, and all B histograms are concatenated into one vector $\text{Bhist}(T_i^l)$. The feature f_i of the input image I_i is defined as the set of block-wise histograms:

$$f_i \doteq [\text{Bhist}(T_i^1), \ldots, \text{Bhist}(T_i^{L_1})]^T \in\in \mathcal{R}^{(2^{L_2})L_1 B}. \qquad (1)$$

B. Compact binary descriptor (CBD)

The Compact Binary Descriptor (CBD) [25], unlike most other binary descriptors such as

LBP, is not handcrafted but is based on a learning method where Pixel Difference Vectors (PDVs) are extracted from local patches by computing the difference between each pixel and its neighbors. Such PDVs are then projected in an unsupervised manner onto low-dimensional binary vectors such that the variance of the binary codes in the training set is maximized and the loss of information between the original real-value codes and the binary codes is minimized.

Let $\boldsymbol{X} = [x_1, x_2, \ldots, x_n]$ be the training set containing *N* samples, where $x_n \in \mathbb{R}^d$ is the *n*-th pixel difference vector and $1 \leq n \leq N,$ the aim of CBD is to learn *K* hash functions that map and quantize each x_n into a binary vector $b_n = [b_{n1}, b_{n2}, \ldots, b_{nK}] \in \{0,1\}^{1\times K}$ that more compactly encodes the discriminative information.

Letting $\boldsymbol{W}_k \in \mathbb{R}^d$ be the projection vector for the *k*-th function, the *k*-th binary code b_{nK} of x_n can be computed as $b_{nK} = 0.5 \times (sng(w_k^T x_n) + 1)$, where $sng(v)$ is 1 if $v \geq 0$ and -1 otherwise.

The following optimization objective function for *W* achieves the goals of making the learned binary codes discriminative and compact:

$$\min_{W} J(\boldsymbol{W}) = \mathrm{tr}(\boldsymbol{W^T Q W}) + \lambda_1 \mathrm{tr}(\boldsymbol{W^T X X^T W}) \qquad (2)$$

$$-2 \times \mathrm{tr}(\boldsymbol{B} - 0.5) \times \boldsymbol{X}^T\boldsymbol{W}))$$

$$-\lambda_2 \times N \times \mathrm{tr}(\boldsymbol{1}^{1\times k}\boldsymbol{W}^T\boldsymbol{X}\boldsymbol{1}^{N\times 1}),$$

$$\text{subject to } \boldsymbol{W}^T\mathbf{W} = \mathrm{I},$$

where λ_1 and λ_2 are two parameters that balance the effects of the different terms, and $\boldsymbol{Q} \triangleq -{}^{1}/_{N} \times (\boldsymbol{XX^T} - 2\boldsymbol{XM^T} + \boldsymbol{MM^T}) + \lambda_2 \boldsymbol{X}\boldsymbol{1}^{N\times 1}\boldsymbol{1}^{1\times N}\boldsymbol{X}^T,$ where $\boldsymbol{M} \in \mathbb{R}^{Nxd}$ is the mean matrix which is a repeated column vector of the mean of all the PDVs in the training set, and $\boldsymbol{B}$ is the binary code matrix:

$$\boldsymbol{B} = 0.5 \times (sgn(\boldsymbol{W^T X}) + 1) \in \mathbb{R}^{N\times K} \qquad (3)$$

When $\boldsymbol{W}$ is fixed, the optimization objective function for $\boldsymbol{B}$ takes the relaxed form of (3) (see [25]).

$\boldsymbol{B}$ and $\boldsymbol{W}$ are iteratively optimized following a two-stage method that can be described algorithmically as follows:

```
Step 1 (initialization): initialize $W$ to be the top $K$ largest eigenvalues of $XX^T$.
Step 2 (Optimization): for $t = 1, 2, \dots, T$:
        2.1 Fix $\boldsymbol{W}$ and update $\boldsymbol{B}$ using (3);
        2.2 Fix $\mathbf{B}$ and update $\mathbf{W}$ using (2);
        2.3. if $|\boldsymbol{W}^T - \boldsymbol{W}^{t-1}| < \varepsilon$ and $t > 2$, goto step 3.
Step 3 (Output): Output matrix $\boldsymbol{W}$.
```

C. **Class scores used as features**

The classification system proposed in this paper remaps a given deep CNN trained to solve a problem that is different from the one for which it was originally trained. Suppose $\boldsymbol{A} = \{a_0, a_1, \dots, a_N\}$ is a set of labels representing some classification problem into which a given input $\boldsymbol{x}$ should be classified, and let $\boldsymbol{y} = f_A(\boldsymbol{x})$ be the classification function implemented by the deep CNN [30]. Such a function takes the input vector $\boldsymbol{x}$ and provides an output vector $\boldsymbol{y}$ of length *N* that contains the scores assigned to all possible outcomes considered in the classification problem. For example, the classification problem *Animal* might have the following outcome: *Animal* = {*pig*, *dog*, *horse*}. A deep CNN trained on this problem would assign to each input image three values (under the implicit assumption that an input image will contain a pig, a dog, or a horse, or none of the three animal classes). The approach taken in this paper is to solve a different classification problem, say *Tool* = {*Hammer, drill, screwdriver*}, by considering the CNN scores assigned to some other problem, e.g. the *Animal* problem.

Consider the scheme in Figure 1. Given a problem Q, the input $\boldsymbol{x}$ is sent to several different deep CNNs that have been trained to solve several different classification problems (e.g. A, B, C), each characterized by a different number of labels (M, N, and P, respectively). The three deep networks provide three output vectors: $\{a_0, \ldots, a_M\}$, $\{b_0, \ldots, b_N\}$, and $\{c_0, \ldots, c_P\}$. These outputs then become the inputs to three SVMs that are trained to provide a result that solves problem Q. In other words, the SVMs learn the correlations between A, B, C, and Q. The outputs of the three SVMs are then combined using the sum rule.

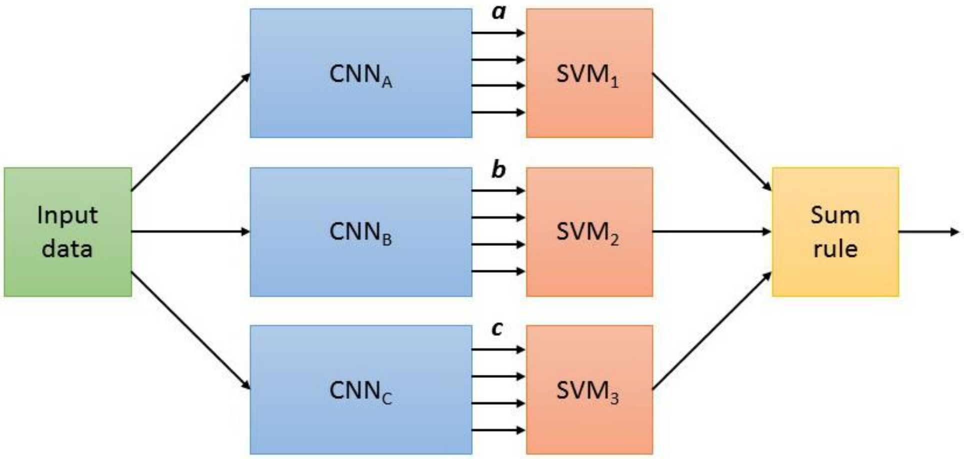

Figure 1: Scheme for deep transfer learning. The same input is sent to three different CNNs, whose outputs are processed by three SVMs. The outputs of the SVMs are finally combined by sum rule.

It is important to note that only the three SVMs need be trained using this double-stage classification pipeline since the deep CNNs implement models that have already been trained for different problems—a training phase, it should be noted, that was computationally very expensive to accomplish.

Input images are preprocessed before being sent to the CNNs. A first preprocessing step required by the CNNs should guarantee a constant size for the input data. This means that images must be resized in order to have the same number of rows and columns. Furthermore, the average training image is subtracted from each image before processing, minimizing the outlier effect (i.e.

uncommon pixel values) as suggested in [30]. The pretrained models used in this work are those available with the MatConvNet[2] toolbox and are those used in [31], [32], and [33]. These models are able to classify 1000 categories.

The rationale behind our approach is based on the supposition that there is a certain degree of similarity between different classification problems. In other words, it is assumed that a class a_i belonging to classification problem A will have some degree of similarity with another class b_j belonging to classification problem B. Taken alone the degree of similarity will most likely be limited and lacking in discriminant power. To build a robust GP system it is necessary to consider multiple classification problems simultaneously, allowing each one to bring to the table its own degree of similarity to the problem at hand. A system for assigning the similarities to this larger set of classes then needs to be trained.

A large number of output classes is a key aspect of the CNNs employed in our system. However, the large number of features (partially correlated among them) provided by each CNN (which are the inputs of each SVM), coupled with the relatively small number of training samples that are commonly available in the datasets of the new problems (in the order of 500-2000 images), can be a source of performance degradation due to the curse of dimensionality. This problem can be limited using a random subspace (RS) ensemble [34] in place of a stand-alone classifier since RS only considers randomly drawn subsets of the group of input features, resulting in a reduction of the number of input values. In this work, each random subspace ensemble is composed of 50 SVMs combined by sum rule and trained using a random subset of 50% of the given input values.

D. **Inner layers**

Unlike other approaches proposed in the literature, our system is not limited to the analysis

[2] The models are available at: http://www.vlfeat.org/matconvnet/pretrained/

of the output layer of the considered CNN; rather, it extends the analysis to the features provided by deeper layers, as shown in Figure 2. CNNs for classification tasks are often designed in order to reduce the dimensionality of feature vectors while approaching the shallow layers: deep layers, therefore, provide high dimensional vectors. Since these deep layer feature vectors serve as inputs to the individual SVMs [5] that are used for synthesizing the transfer learning, the high dimensionality of these vectors needs to be counterbalanced in order to avoid the curse of dimensionality.

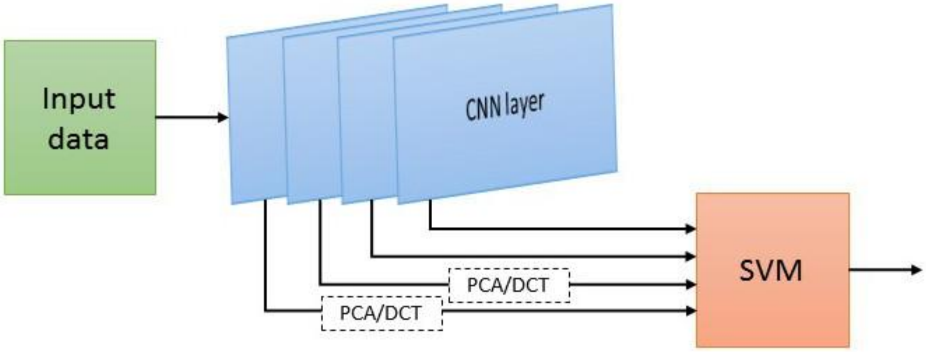

Figure 2: Working principle of the feature extraction from inner layers. The output of each layer is considered as a feature vector; a dimensionality reduction method (PCA and DCT) is possibly applied, depending on the vector size. All vectors are then processed by an SVM to provide the final output.

The dimensionality reduction methods employed in this work are Principal Component Analysis (PCA) and discrete cosine transform (DCT). PCA [35] is a popular method for unsupervised dimensionality reduction. It maps feature vectors into a smaller number of uncorrelated directions calculated to preserve the global Euclidean structure. PCA also extracts an orthogonal projection matrix so that the variance of the projected vectors is maximized. The DCT transform [36] is a good compromise between information packing and computational complexity. Moreover, most DCT components are very small in magnitude, because most of the salient information exists in the coefficients with low frequencies. For this reason, removing small coefficients from the representation introduces only small errors in image reconstruction.

Both PCA and DCT are used here to limit the dimension of the feature vector of each layer to 4,000 elements. What this means is that an orthogonal basis of 4,000 dimensions is provided by PCA, and the first 4,000 frequencies are selected by DCT (the lower frequencies provide the highest information content). The dimensionality reduction turned out to be quite strong, as the original feature length for the layers considered in this work reached dimensions of up to 100,000 as detailed below in Section IV.B.

III. HANDCRAFTED FEATURES

Handcrafted features have been used for more than a decade in a number of computer vision applications, including object detection and image classification. Over time, the number of features have increased to better adapt to the various tasks being tackled by researchers. Feature-based approaches can be considered state-of-the-art for the *generic computer vision* problem we are building in this paper. For this reason, we consider different feature-based approaches as a reference point against which to compare the performance of feature-based systems exploiting a combination (ensemble) of different features. We believe this to be the best choice given that our recognition task is not specific but rather generic.

The handcrafted features used in our comparisons are variations of three main approaches: Local Binary Pattern (LBP) [37], Local Ternary Pattern (LTP) [38] and Local Phase Quantization (LPQ) [39]. Many state-of-the-art descriptors are based on LBP [40]. These approaches extract features that are general-purpose and are, therefore, most suitable for building a generic computer vision systems.

A. **Local Binary Pattern (LBP)**

Proposed by Ojala et al. [37], LBP has rapidly become a popular descriptor mainly because of its low computational complexity and ability to code fine details. The canonical LBP operator

[37] is computed at each pixel location by considering the values of a small circular neighborhood (with radius *R* pixels) around the value of a central pixel I_c, as follows:

$$\mathrm{LBP}(N, R) = \sum_{n=0}^{N-1} s(I_n - I_c)2^n \qquad (4)$$

where *N* is the number of pixels in the neighborhood, *R* is the radius, and s(*x*) = 1 if $x \geq 0$, otherwise s(*x*) = 0. The descriptor is the histogram of these binary numbers.

B. **Local Ternary Pattern (LTP)**

One variant that has inspired many others is the Local Ternary Patterns (LTP), proposed by Tan and Triggs [38], which utilizes a 3-valued encoding scheme that includes a threshold around zero for the evaluation of the local gray-scale difference. The ternary coding is accomplished by introducing the threshold τ in the canonical LBP *s(x)* function:

$$s(x) = \begin{cases} 1, x \geq \tau \\ 0, |x| < \tau \\ -1, x \leq -\tau \end{cases} \qquad (5)$$

Because of the length of the ternary coding, the original LTP code is split into a positive and a negative LBP code.

C. **Local Phase Quantization (LPQ)**

The LPQ operator, first proposed by Ojansivu and Heikkila [39], is based on the blur invariance property of the Fourier phase spectrum and uses the local phase information extracted from the 2-D short-term Fourier transform (STFT) computed over a rectangular neighborhood at each pixel position of the image.

In the Fourier domain, the model for spatially invariant blurring of an image, g(**x**), is:

$$\mathrm{G}(\mathbf{u}) = \mathrm{F}(\mathbf{u}) \cdot \mathrm{H}(\mathbf{u}), \qquad (6)$$

where G(**u**), F(u), and H(**u**) are the discrete Fourier transforms (DFT) of the blurred image $g(\mathbf{x})$, the original image $f(\mathbf{x})$, and the point spread function (PSF) $h(\mathbf{x})$, respectively, and **u** is a vector of coordinates $[u, v]^T$.

The magnitude and phase aspects in (6) can be separated. Thus,

$$|G(\mathbf{u})| = |F(\mathbf{u})| \cdot |H(\mathbf{u})| \quad \text{and} \tag{7}$$
$$\angle G(\mathbf{u}) = \angle F(\mathbf{u}) + \angle H(\mathbf{u}).$$

In the case where the blur $h(\mathbf{x})$ is centrally symmetric, the Fourier transform is always real-valued, and its phase is a two-valued function given by $\angle H(\mathbf{u}) = 0$ if $H(\mathbf{u}) \geq 0$, π otherwise.

The LPQ method uses the local phase information extracted using STFT computed over a rectangular neighborhood N_x of size M by M at each pixel position **x** of the image $f(\mathbf{x})$:

$$F(\mathbf{u}, \mathbf{x}) = \sum_{y \in Nx} f(x-y) e^{-j2\pi \mathrm{u}^T y} = w_u^T f_x \tag{8}$$

where $\mathbf{w}_\mathrm{u}$ is the basis vector of the 2-D DFT at frequency **u**, and f_x is a vector containing all M^2 image samples from N_x.

Only four complex coefficients are considered, however, corresponding to the 2-D frequencies $\mathbf{u}_1 = [a,0]^T$, $\mathbf{u}_2 = [0, a]^T$, $\mathbf{u}_3 = [a, a]^T$, and $\mathbf{u}_1 = [a,- a]^{T,}$, where a is the first frequency below the first zero crossing of H(**u**) that satisfies $\angle G(\mathbf{u}) = \angle F(\mathbf{u})$ for all $\angle H(\mathbf{u}) \geq 0$.

If we let

$$\mathbf{F}_x^c = [F(\mathbf{u}_1,\mathbf{x}), F(\mathbf{u}_2,\mathbf{x}), F(\mathbf{u}_3,\mathbf{x}), F(\mathbf{u}_4,\mathbf{x})], \quad \text{and} \tag{9}$$
$$\mathbf{F}_\mathbf{x} = [\mathrm{Re}\{ \mathbf{F}_x^c \}, \mathrm{Im}\{ \mathbf{F}_x^c \}]^T,$$

where Re{·} and Im{·} return the real and the imaginary parts of a complex number, respectively, then the corresponding transform matrix is

$$\mathbf{W} = [\mathrm{Re}\{\mathbf{w}_{\mathbf{u}1}, \mathbf{w}_{\mathbf{u}2}, \mathbf{w}_{\mathbf{u}3}, \mathbf{w}_{\mathbf{u}4},\}, \mathrm{Im}\{\mathbf{w}_{\mathbf{u}1}, \mathbf{w}_{\mathbf{u}2}, \mathbf{w}_{\mathbf{u}3}, \mathbf{w}_{\mathbf{u}4},\}]^T \tag{10}$$

Thus,

$$\mathbf{F_x} = \mathbf{W} f_x. \qquad (11)$$

Assuming that for f_x the correlation coefficient between adjacent pixel values is ρ and the variance of each sample is $\sigma^2 = 1$, the covariance between positions $\mathbf{x}_i$ and $\mathbf{x}_j$ becomes $\sigma_{ij} = \rho^{\|\mathbf{x}_i - \mathbf{x}_j\|}$, where $\| \cdot \|$ denotes the L_2 norm.

The covariance matrix of the transform coefficient vector $\mathbf{F_x}$ can be obtained from $\mathbf{D} = \mathbf{WCW}^T$. Where $\mathbf{C}$ is the covariance matrix of all M samples in N_x.

The coefficients need to be decorrelated before quantization. Assuming a Gaussian distribution, a whitening transform is applied:

$$\mathbf{G_x} = \mathbf{V}^T\mathbf{F_x}, \qquad (12)$$

where $\mathbf{V}$ is an orthonormal matrix derived from the singular value decomposition (SVD) of the matrix $\mathbf{D}$, and $\mathbf{G_x}$ is computed for all image positions.

The resulting vectors are quantized using a scalar quantizer, g_j is the jth component of $\mathbf{G_x}$ and $q_j = 1$ if $g_j \geq 0$, otherwise 0. These quantized coefficients are represented as integers between 0-255 using the binary coding $b = \sum_{j=1}^{8} q_j 2^{j-1}$. Finally, a histogram of these integer values is composed and used as a feature vector.

D. Other LBP Variants

Variants explored in this paper that are based on the above handcrafted methods are detailed in Table 2, which provides a brief description of each variant.

Acronym	Description	Ref
RICLBP	Rotation Invariant Co-occurrence LBP: a variant that takes the concept of co-occurrence among LBPs and incorporates rotation equivalence classes.	[41]
CLBP	Complete Local Binary Pattern: an LBP variant that codes a given local region using a given central pixel.	[42]
disCLBP	Discriminative Completed Local Binary Pattern.	[43]
RLBP	Rotated Local Binary Pattern: an LBP variant that computes the descriptor with respect to a local reference.	[44]
MrElbp	Median Robust Extended Local Binary Pattern: an LBP variant that compares regional image medians rather than raw image intensities.	[45]
Msjlbp	Rotation Invariant Multi-scale Co-occurrence Local Binary Pattern.	[46]
HASC	Heterogeneous Auto-Similarities of Characteristics: an LBP variant that models linear and non-linear feature dependencies.	[47]
GOLD	Gaussians of Local Descriptors: a flexible local feature representation leveraging parametric probability density functions.	[43]

Table 2. Brief description of some other state-of-the-art LBP variants

IV. EXPERIMENTAL RESULTS

To test our approach, we selected several datasets that include very different image types. A large variety of image types is crucial for demonstrating the generalizability of our approach and for providing stable results and high accuracy irrespective of the specific problem being addressed. The datasets selected include, among many others, medical and subcellular images, pictures of smoke, and images of ancient books, flora, and paintings. All the following datasets used in our experiments are publicly available:

- **PS** (Pap Smear dataset) [48]: includes images of cells used in the diagnosis of cervical cancer.

- **VI** (Virus dataset) [49]: contains images of viruses extracted using negative stain transmission electron microscopy. Although the dataset provides masks for removing the background of each image, these were not used in our tests.
- **CH** (Chinese Hamster Ovary cells) [50]: contains fluorescent microscopy images representing five different classes taken from Chinese Hamster Ovary cells.
- **SM** (smoke dataset) [51]: contains images of smoke used for assessing intelligent video surveillance systems. The same division into training and testing set proposed by the authors of [51] is used in this paper.
- **HI** (Histopathology dataset) [52]: contains images of four fundamental types of tissues.
- **BR** (breast cancer dataset) [53]: contains images of malignant and benign breast cancer.
- **PR** (DNA-binding and non-binding proteins) [54]: contains images where texture descriptors were used for protein classification.
- **HE (**2D HeLa dataset) [50]: contains images acquired using a fluorescence microscope. The images represent single cells divided into ten staining classes.
- **LO** (Locate endogenous mouse sub-cellular dataset) [55]: contains images unevenly distributed among ten classes of endogenous proteins or features of specific organelles.
- **TR** (Locate transfected mouse sub-cellular organelles dataset) [55]: contains 553 images unevenly distributed in eleven classes of fluorescence-tagged or epitope-tagged proteins transiently expressed in specific organelles.
- **PI** (holy bible dataset) [56]: contains images extracted from digitalized pages of ancient editions of the Holy Bible dating from 1450 A.D. to 1471 A.D. The dataset is organized into thirteen classes characterized by clear semantic meanings and significant search relevance.

- **RN** (fly cell dataset): contains 200 fluorescence microscopy images of fly cells, evenly distributed among ten classes.
- **PA** (painting dataset) [57]: contains 2338 paintings by 50 artists, belonging to thirteen different painting styles. The division of training/testing sets was provided by the authors and used in our tests.
- **LE (**Brazilian flora dataset) [34]: contains 400 samples evenly distributed among twenty classes. For each image, a subdivision into three windows of size 128×128 pixels was manually performed that produced a total of 1,200 texture samples.
- **IS**: (ISMIR 2004 Genre Classification dataset): contains 1,458 music pieces assigned to six different genres: classical, electronic, jazz/blues, metal/punk, rock/pop, and world. This dataset is one of the most widely used datasets in music information retrieval research. In our tests the audio signal is converted into a spectrogram image showing the spectrum of frequencies (vertical axis) varying according to time (horizontal axis). Texture descriptors are extracted from the whole spectrogram (for details, see [58])
- **KU** (butterfly dataset) [59]: contains 140 butterfly images representing fourteen different butterfly species of the Styridae family. Each of the butterfly images was cropped to a 256 × 256 pixel image before processing and divided in two equal regions (an upper and a lower region) without overlap. For each region a different SVM was trained and combined by sum rule.
- **FM**: Flickr Material Database: contains images of ten material categories in the database: fabric, foliage, glass, leather, metal, paper, plastic, stone, water and wood. Each category has 100 images, 50 of which are close-up views while the remaining 50

offer views at object-scale. There is a binary, human-labeled mask associated with each image describing the location of the object. We only consider pixels inside this binary mask for material recognition and disregard all the background pixels. We use the same split suggested by the authors of the dataset. State-of-the-art computer vision systems still achieve a performance level that is lower than that of humans (human beings achieve an accuracy of 84.9%).

- **SR** (scene dataset) [60]: contains images of scenes divided into fifteen classes. This dataset is widely used in the literature. The testing protocol established by other papers, which we follow, requires five experiments, each using 100 randomly selected images per category for training and the remaining images for testing. Each image is divided into four equal regions without overlap and a central region of 1/2 the size of the original image. For each region, a different SVM is trained, and the five SVMs are combined by sum rule.

The datasets detailed above are summarized in Table 3, which shows the number of classes, the number of samples, the image size, and the URL for downloading the data. Unless otherwise specified in the description of the dataset, the protocol used in our experiments is a five-fold cross-validation method.

Dataset	#Classes	#Samples	Sample Size	URL for Download
PS	2	917	Various	http://labs.fme.aegean.gr/decision/downloads
VI	15	1500	41×41	http://www.cb.uu.se/~gustaf/virustexture
CH	5	327	512×382	http://ome.grc.nia.nih.gov/iicbu2008/hela/index.html#cho
SM	2	2868	100×100	http://staff.ustc.edu.cn/~yfn/vsd.html
HI	4	2828	various	http://www.informed.unal.edu.co/histologyDS
BR	2	584	various	upon request to Geraldo Braz Junior [ge.braz@gmail.com]
PR	2	349	various	upon request to Loris Nanni [nanni@dei.unipd.it]
HE	10	862	512×382	http://ome.grc.nia.nih.gov/iicbu2008/hela/index.html
LO	10	502	768×512	http://locate.imb.uq.edu.au/downloads.shtml
TR	11	553	768×512	http://locate.imb.uq.edu.au/downloads.shtml
PI	13	903	various	http://imagelab.ing.unimo.it/files/bible_dataset.zip
RN	10	200	1024×1024	http://ome.grc.nia.nih.gov/iicbu2008/rnai/index.html
PA	13	2338	various	http://www.cat.uab.cat/~joost/painting91.html
LE	20	1200	128×128	Upon request to bruno@ifsc.usp.br
IS	6	1424	513×800	upon request to Loris Nanni [nanni@dei.unipd.it]
KU	14	140	128×256	upon request to Yılmaz Kaya [yilmazkaya1977@gmail.com]
FM	10	1000	192×256	http://people.csail.mit.edu/celiu/CVPR2010/FMD/
SR	15	4485	various	http://www-cvr.ai.uiuc.edu/ponce_grp/data/

Table 3. Descriptive Summary of the Dataset.

The area under the ROC curve (AUC) [61] is used to assess performance. AUC is calculated using the one-versus-all approach [62] for multiclass problems. Experiments were statistically validated using the Wilcoxon signed rank test [63].

A. Class scores as features

The aim of the first experiment, reported in Table 4, is to establish the usefulness of extracting features using the proposed approach. If not specified, the RS-SVM is used as the classifier. During the testing phase, all the pretrained models provided by the MatConvNet toolbox were considered. The performance level achieved is reported in Table 4 for the following combinations of deep CNN and SVM:

S1: the imagenet-vgg-verydeep-19 pretrained model and a stand-alone SVM is used as the classifier;

C1: the imagenet-vgg-verydeep-19 pretrained model;

C2: the imagenet-vgg-verydeep-16 pretrained model;

C3: the imagenet-vgg-f pretrained model;

C4: the imagenet-vgg-m pretrained model;

C5: the imagenet-vgg-s pretrained model;

C6: the imagenet-vgg-m-2048 pretrained model;

C7: the imagenet-vgg-m-1024 pretrained model;

C8: the imagenet-vgg-m-128 pretrained model;

C9: the imagenet-caffe-ref pretrained model;

C10: the imagenet-caffe-alex pretrained model.

Table 4 also reports the performance of the following ensembles:

E1: the sum rule of C1 and C2;

E2: the sum rule of C1, C3, and C9;

E3: fusion, by sum rule, of all ten RS-SVMs trained with C1-C10.

	PS	**VI**	**CH**	**SM**	**HI**	**BR**	**PR**	**HE**	**LO**	**TR**	**PI**	**RN**	**PA**	**LE**	**IS**	**KU**	**FM**	**SR**
S1	83.2	94.7	97.1	96.8	80.9	74.5	73.5	96.3	98.0	95.3	83.4	86.2	72.4	94.3	83.5	87.9	61.8	90.4
C1	84.6	95.0	98.2	97.0	84.0	77.6	74.0	96.7	98.2	95.4	83.4	87.7	74.5	94.6	85.7	88.1	61.7	95.2
C2	82.7	93.0	98.4	99.0	85.8	78.7	73.5	97.1	98.4	97.5	84.0	87.8	76.2	96.0	86.7	89.3	62.6	94.5
C3	83.4	95.6	99.6	97.1	84.5	79.8	83.1	96.4	79.0	97.2	86.9	88.2	76.3	93.5	87.9	88.6	64.3	93.6
C4	83.1	95.4	99.5	98.3	85.1	79.2	84.6	97.3	97.8	96.8	84.5	87.4	74.7	91.0	85.9	88.4	63.3	95.6
C5	84.3	93.8	99.6	97.7	82.7	78.5	80.9	97.5	98.6	96.4	84.2	84.9	75.1	94.3	85.1	90.5	62.7	94.7
C6	83.3	94.5	99.4	97.3	84.3	77.9	83.8	96.7	96.2	94.6	84.8	82.1	73.0	89.3	85.4	88.1	64.1	95.5
C7	80.4	93.6	99.7	97.2	85.4	77.2	81.9	96.2	95.8	94.0	82.8	80.0	72.8	91.7	84.5	89.0	61.8	95.5
C8	81.6	92.1	99.0	95.9	83.2	80.1	81.8	94.5	95.3	93.3	84.1	83.2	74.7	89.7	80.9	88.1	62.4	96.4
C9	85.0	94.3	99.3	97.1	79.9	76.4	76.8	95.4	95.2	94.8	85.5	86.2	73.0	91.8	86.6	88.5	64.9	92.9
C10	82.8	93.9	99.3	97.0	82.5	74.9	86.6	95.7	97.4	96.4	87.1	89.1	72.9	91.2	86.9	**90.7**	64.2	93.6
E1	86.9	96.1	99.0	98.8	87.4	82.4	79.0	98.1	99.0	97.8	85.8	90.9	76.5	**96.7**	89.3	90.5	64.6	95.9
E2	90.0	97.1	**99.9**	98.9	86.6	82.7	85.2	98.4	98.4	98.2	89.4	91.2	77.8	96.0	91.1	89.9	67.7	96.9
E3	**91.4**	**97.9**	**99.9**	**99.2**	**90.5**	**85.7**	**90.5**	**98.8**	**99.1**	**98.7**	**91.4**	**92.1**	**79.7**	96.0	**91.8**	90.4	**69.9**	**97.6**

Table 4. Performance of the proposed approaches.

Examining Table 4, it is clear that the ensembles improve the performance of the stand-alone CNN. The best performance is obtained by combining all the CNNs, E3 outperforming all the other methods with a p-value of 0.05. In addition, C1 (based on RS-SVM) greatly outperforms S1 (based on the stand-alone SVM) with a p-value of 0.01. We tested intersection, linear, and radial basis function kernels for SVM for all the tested descriptors. The optimal parameters were found using an internal 5-fold cross-validation protocol inside the training set.

It is interesting to note that RS boosts performance with the non-handcrafted descriptors, a performance boost that is probably due to the correlation among the features. In contrast, the RS ensemble using handcrafted features results in only a very marginal gain in performance [24].

B. **Transfer learning**

To reduce the computation time, only stand-alone SVMs are trained in the following tests. Preliminary tests showed that an RS ensemble produced a very small performance improvement. We use an Intersection kernel when the features are the output of a layer of CNN or when the output is reduced by DCT. When the output is reduced by PCA, a radial basis function (RBF) kernel is used before feeding into an SVM. Experimentally DCT works better with Intersection kernel and PCA with RBF kernel. The dimensionality reduction methods were applied to features larger than 5000 elements, which were shown by the imagenet-vgg-verydeep-19 model in its layers #30 (100,352 elements) and #38 (25,088 elements). Other layers considered in this study (#39, #40, #41 and #42) produce feature vectors of 4,096 elements that were not reduced. Both DCT and PCA were set to output a feature vector dimension of 4,000 elements: such a number is not a ratio of the input dimension but rather was chosen to sidestep the curse of dimensionality while training the SVM. For the sake of space, we have reported only the most interesting results.

In Table 5 we compare different approaches for deep transfer learning:

- Base: the CNN (the imagenet-vgg-verydeep-19 model), where output of layer #42 is used for describing the images;
- DCT: the CNN (the imagenet-vgg-verydeep-19 model), where output of layer #38 is reduced by DCT, and the DCT coefficients are used for describing the images;
- PCA: the CNN (the imagenet-vgg-verydeep-19 model), where output of layer #38 is reduced by PCA, and the PCA coefficients are used for describing the images;
- B+D: fusion by sum rule between Base and DCT;
- B+P: fusion by sum rule between Base and PCA;
- B+P+D: fusion by sum rule among Base, DCT, and PCA;
- 38+40+42: fusion by sum rule among Base (layers #42 and #40), DCT (layer #38), and PCA (layer #38). For each feature set, a different SVM is trained, then the four SVMs are combined by sum rule;
- 30+38+42: fusion by sum rule among Base (layer #42), DCT (layers #30 and #38), and PCA (layers #30 and #38). For each feature set a different SVM is trained, then the five SVMs are combined by sum rule;
- 38:42: all the approaches trained with the layers used between #38 and #42, i.e. Base (layers #42, #41, #40, #39), DCT (layer #38), and PCA (layer #38) combined by sum rule;

- MM: fusion of the different models (available at http://www.vlfeat.org/matconvnet/pretrained/). The layers used in each model are reported in Table 5.
- MM+E3: fusion by sum rule between MM and the methods labelled E3 in Table 4. Before fusion, the scores of MM and E3 are normalized to mean 0 and standard deviation 1.

	PS	VI	CH	SM	HI	BR	PR	HE	LO	TR	PI	RN	PA	LE	IS	KU	FM	SR
base	91.8	97.5	99.7	99.8	90.0	85.1	88.5	98.3	99.1	98.8	89.8	79.5	93.1	96.7	91.5	92.1	65.0	99.4
DCT	92.6	97.0	99.6	99.6	89.8	84.6	89.6	98.2	98.7	98.1	92.8	66.3	93.2	93.6	89.4	92.0	65.8	99.2
PCA	89.6	91.1	98.8	98.9	78.8	84.0	81.6	96.6	97.3	95.0	89.3	84.1	90.5	84.6	89.3	89.1	51.8	97.7
B+D	92.9	97.6	99.7	99.9	90.1	86.2	89.7	98.4	99.1	98.9	91.5	79.5	94.1	95.8	91.7	92.3	66.4	99.4
B+P	92.3	97.6	99.7	99.7	89.9	86.4	89.5	98.5	99.1	98.7	92.7	82.8	93.8	96.8	92.4	91.2	64.9	99.4
B+P+D	92.8	97.6	99.7	99.7	90.8	86.5	90.1	98.5	99.1	98.7	94.1	82.4	94.2	96.5	92.1	91.7	66.4	99.4
38+40+42	92.7	97.7	99.7	99.9	91.0	86.7	89.7	98.5	99.2	98.8	94.9	81.8	94.3	97.0	92.5	91.5	65.0	99.4
30+38+42	94.0	98.0	99.8	99.9	91.7	87.0	91.4	99.2	99.5	99.1	97.5	82.5	95.3	95.6	92.8	90.8	66.2	99.4
38:42	92.6	97.7	99.7	99.9	91.2	87.1	89.7	98.5	99.2	98.9	92.4	81.5	94.4	97.2	92.5	91.6	64.9	99.4
MM	95.3	98.2	99.9	99.9	92.8	89.3	93.4	99.4	99.6	99.5	96.9	85.6	95.9	96.1	94.1	90.8	71.8	99.3
MM+0.33×E3	95.1	98.3	99.9	99.9	92.8	89.3	93.3	99.4	99.7	99.6	97.3	91.3	95.3	96.9	94.1	90.8	72.7	99.3

Table 5. Performance of the transfer learning.

As it can be seen from the results reported in Table 5, a set of SVMs using different layers improves performance: (*B*+*P*+*D*) outperforms *base* with a p-value of 0.05; (*30*+*38*+*42*) outperforms (*B*+*P*+*D*) with a p-value of 0.05; and (MM) outperforms (*30*+*38*+*42*) with a p-value of 0.01.

In Table 6 we report the layers used for each model of the ensemble labelled MM. It should be noted that if the feature set extracted from a layer has a dimension higher than 5000 then it is reduced by PCA and DCT.

MODEL	LAYERS
imagenet-vgg-verydeep-19	30 38 42
imagenet-vgg-verydeep-16	28 35 37
imagenet-vgg-f	6 16 21
imagenet-vgg-m	8 18 21
imagenet-vgg-s	10 18 20
imagenet-vgg-f	6 20 21
imagenet-vgg-m	7 16 21
imagenet-vgg-s	10 18 20
imagenet-vgg-m-2048	8 18 21
imagenet-vgg-m-1024	8 18 21
imagenet-vgg-m-128	8 18 21

Table 6. Layers used in each model.

In Figures 1-4 we provide a set of plots showing performance vs layers for two CNN models, viz. the imagenet-vgg-verydeep-19(labelled model 1) and the imagenet-vgg-f (labelled model 3) on PS (labelled dataset 1) and VR (labelled dataset 2). It is interesting to observe that deep layers produce a good set of features for describing the images.

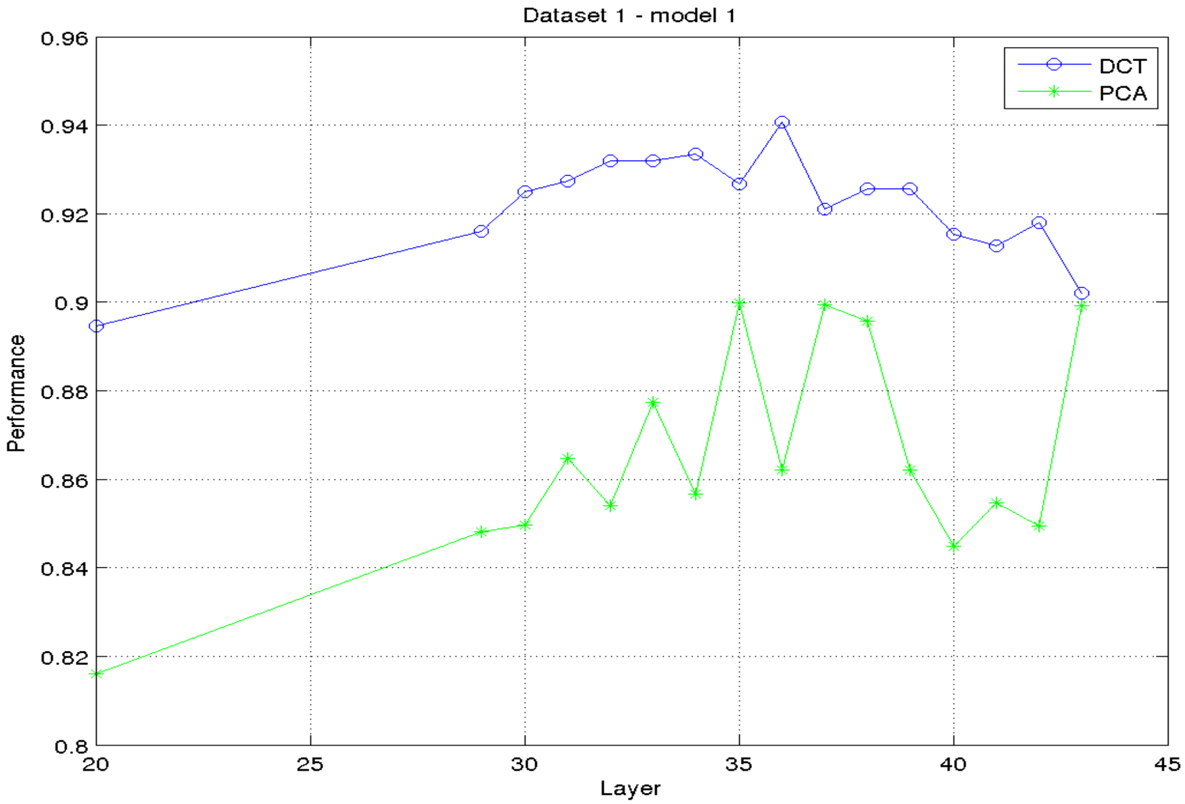

Figure 1. Performance of imagenet-vgg-verydeep-19 on the PS dataset

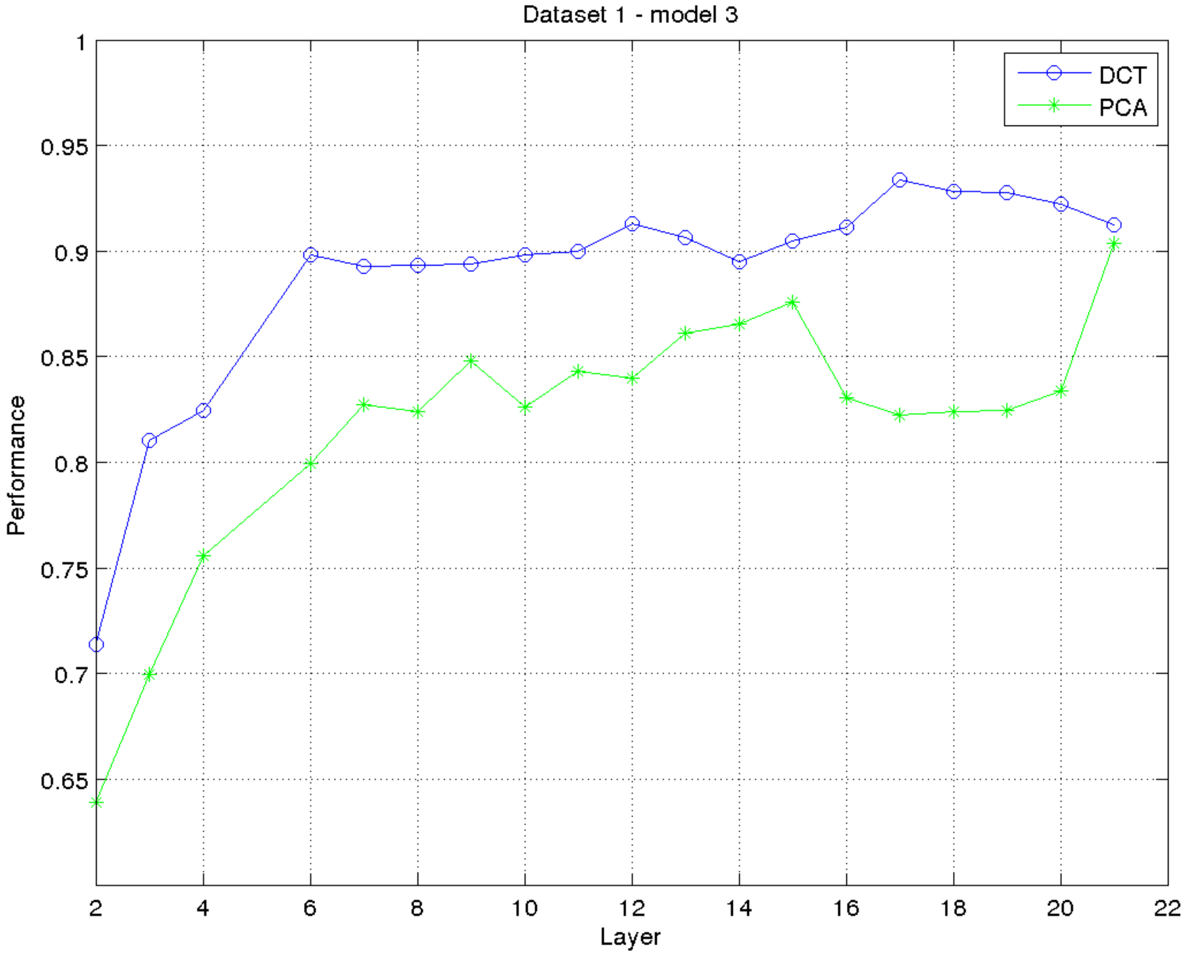

Figure 2. Performance of imagenet-vgg-f on the PS dataset

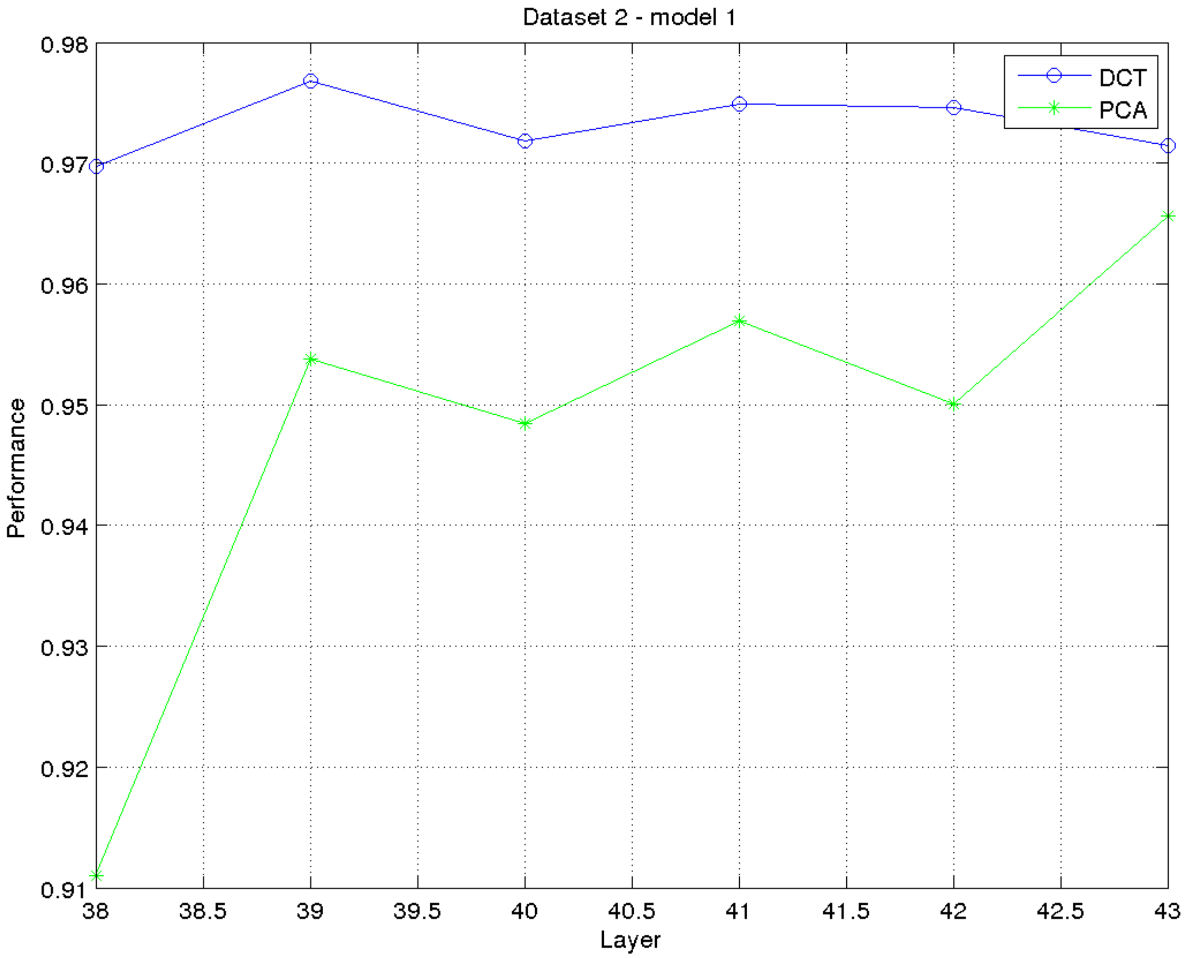

Figure 3. Performance of imagenet-vgg-varydeep 19 on VI dataset

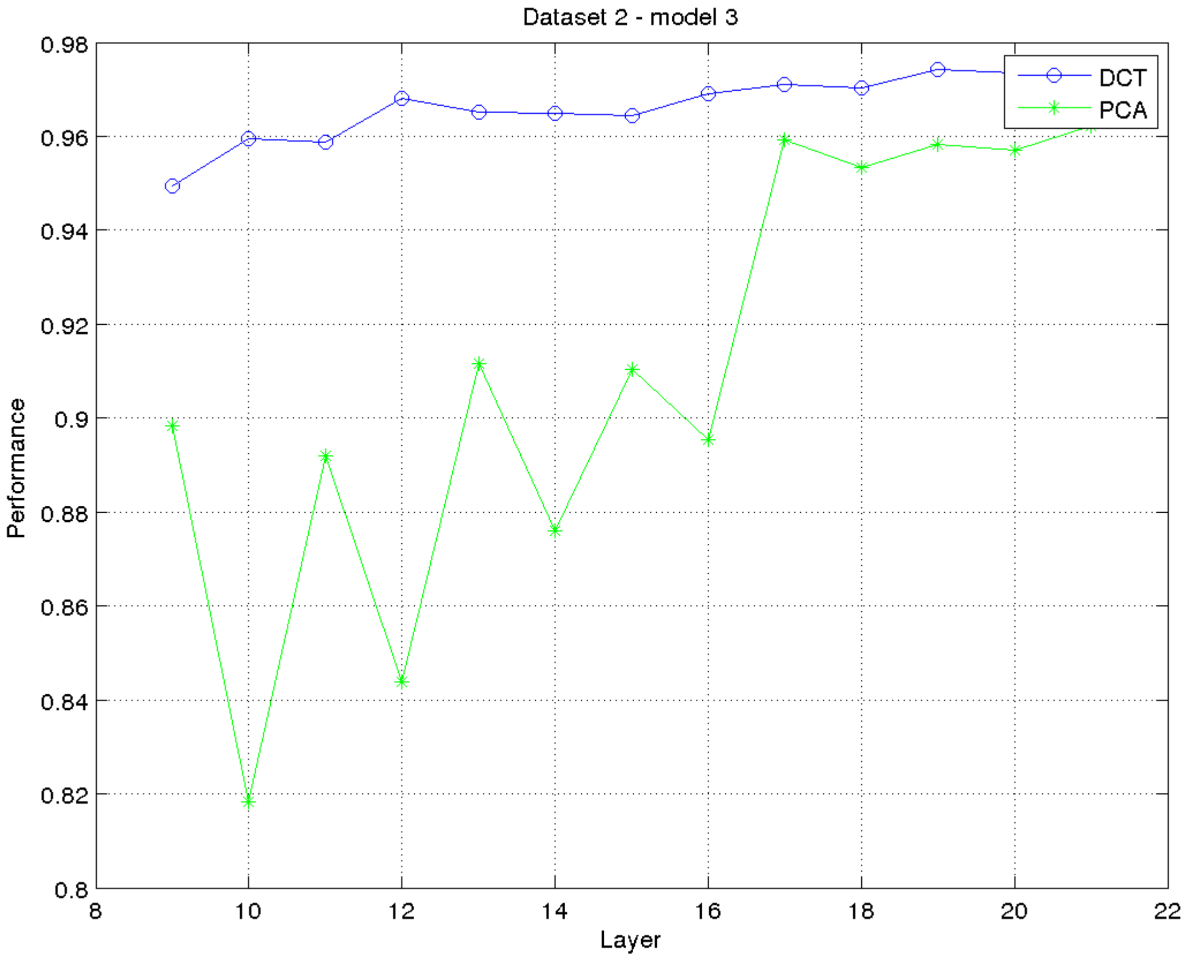

Figure 4. Performance of imagenet-vgg-f on the VI dataset

C. Performance of other non-handcrafted features

In the following test, the results of which are summarized in Table 7, we report the performance of two other non-handcrafted feature approaches tested in this work, as well as the fusion of SVM trained from features provided by CNN layers.

The following approaches are reported in Table 7:

- PC: performance obtained by PCANet used to train a SVM with Intersection Kernel (we report it because PCANet works poorly with a RBF Kernel);

- CB: performance obtained by CBD used to train a SVM with an RBF Kernel (we report it because PCANet works poorly with Intersection Kernel);
- (A)+W×(B): the weighted sum rule between A with weight 1 and B with weigh W. Before fusion, the scores of A and B are normalized to mean 0 and standard deviation 1.

	PS	VI	CH	SM	HI	BR	PR	HE	LO	TR	PI	RN	PA	LE	IS	KU	FM	SR
PC	86.1	93.1	93.1	52.0	82.0	77.3	87.4	89.2	92.2	90.3	68.9	76.4	51.1	90.5	86.5	92.0	69.0	94.4
CB	83.4	88.7	88.9	80.8	75.6	78.9	89.5	82.9	93.3	92.0	67.5	86.3	79.1	85.8	88.5	92.3	80.9	92.5
B+P+D	92.8	97.6	99.7	99.7	90.8	86.5	90.1	98.5	99.1	98.7	94.1	82.4	94.2	96.5	92.1	91.7	66.4	99.4
(PC+CB)+(B+P+D)	93.5	97.8	99.6	99.9	89.2	85.7	91.6	98.6	99.3	98.9	94.3	86.7	93.6	97.3	93.0	92.1	72.5	99.4
(PC+CB)/2+(B+P+D)	93.4	97.8	99.7	99.9	90.4	86.3	91.1	98.6	99.3	99.1	94.3	85.0	94.0	96.9	92.9	91.9	71.4	99.4
(PC+CB)+5×(30+38+42)	96.0	98.1	98.8	98.8	94.7	92.4	94.9	98.5	98.7	98.5	97.4	92.4	94.9	96.2	93.3	91.5	69.9	99.4
0.2×(PC+CB)+(30+38+42)+(E3)	96.1	98.4	99.1	99.1	95.2	93.1	95.7	98.8	98.9	98.9	98.1	95.6	92.8	97.4	93.5	91.2	71.7	99.2
0.1×(PC+CB)+MM+0.33×(E3)	95.1	98.3	99.9	99.9	92.7	89.0	93.3	99.4	99.7	99.6	97.2	92.5	95.2	97.2	94.1	90.9	73.6	99.3

Table 7. Performance obtained combining different non-handcrafted features.

Notice that when the performance of PC and CB is worse than (30+38+42) their fusion improves the performance of (30+38+42). Nonetheless, the performance of 0.1×(PC+CB)+MM+0.33×(E3) is similar to that obtained by MM or MM+0.33×(E3).

D. **Performance of handcrafted features and their fusions with non-handcrafted features**

The aim of the experiment reported in Table 8 is to compare the proposed approach with the state-of-the-art texture descriptors detailed in section III. Below we provide details of the parameters used in this experiment:

- Local Ternary Patterns (LTP) [38]: features were extracted using a threshold of 3, and the concatenation of the feature sets was obtained with (radius=1; number of neighborhood=8) and (radius=2; number of neighborhood=16); only uniform bins were extracted;

- Local Phase Quantization (LPQ) [39]: concatenation of the feature sets was obtained with local window sizes of 3 and 5;
- Rotation Invariant Co-occurrence LBP (RICLBP) [41]: features were extracted by concatenating the feature sets obtained with the scale of LBP radius (s) and the interval of LBP pair (r) of (s=1, r=2) (s=2, r=4) (s=4, r=8);
- Discriminative Completed Local Binary Pattern [43] (disCLBP_v2): the best approach was used and labeled as in the original paper dis(S+M), with the number of neighborhoods 8 and with the radius=1, 3, and 5;
- disCLBP: as with disCLBP_v2, but with the following parameters: (radius=1; number of neighborhood=8), (radius=2; number of neighborhood=16), and (radius=3; number of neighborhood=24);
- Complete Local Binary Pattern (CLBP) [42]: features were extracted by concatenating the feature set obtained with (radius=1; number of neighborhood=8) and (radius=2; number of neighborhood=16).
- Rotated Local Binary Pattern image (RLBP) [44]: features were extracted by concatenating the feature set obtained with (radius=1; number of neighborhood=8) and (radius=2; number of neighborhood=16). Uniform bins are extracted;
- RLBPselec: all bins are extracted with (radius=1; number of neighborhood=8) and (radius=2; number of neighborhood=16); then a feature selection process is applied based on the frequency of the patterns (exactly as in the original paper [44]) using a threshold of 0.9;

- MrElbp [45]: default parameter settings in the original journal paper were used: radius taking the value of {2, 4, 6, 8} and a point set of 8. Images are resized to a fixed dimension (height of 150 pixel) before feature extraction;
- Msjlbp [46]: using the default parameters for the code;
- HASC [47]: Heterogeneous Auto-Similarities of Characteristics;
- GAB [47]: the mean-squared energy and the mean amplitude were calculated from 5 different scale levels and 14 different orientations. In this way, a feature vector of size 5×14×2 is obtained;
- GOLD Gaussians of Local Descriptors [64]: here we train a different SVM from each region of the spatial pyramid and combine them by sum rule. We also use one level spatial pyramid decomposition. The decomposition consists of the entire image, followed by level one, where the image is subdivided into four quadrants.

In Table 8 the fusion between handcrafted and non-handcrafted features is reported in the rows labelled *Fus*. As it can be observed, handcrafted features capture information that is different from the system proposed in this paper. We checked the error independence with Yule's Q-statistic [28] (Q-stat). The values of Q are bounded between [-1,1]. Classifiers that recognize the same patterns correctly have Q > 0; those which commit errors on different patterns have Q < 0. In Table 8, the Q-statistic is reported between handcrafted and non-handcrafted features in the row labelled *Q-stat*. Results show that different descriptors train a set of partially uncorrelated classifiers. For this reason, the fusion between handcrafted and non-handcrafted features outperforms the handcrafted features (p-value 0.01).

To avoid presenting a large table, we report the fusion and the q-statistic between the handcrafted and non-handcrafted features for the three best performing handcrafted features only.

	PS	VI	CH	SM	HI	BR	PR	HE	LO	TR	PI	RN	PA	LE	IS	KU	FM	SR
LTP	91.4	93.4	99.9	99.7	91.5	**96.9**	89.6	98.1	99.4	99.2	92.8	96.9	89.0	97.9	92.3	91.5	79.0	97.6
0.1×(PC+CB)+MM+0.33×(E3)	95.1	**98.3**	99.9	**99.9**	92.7	89.0	93.3	99.4	99.7	99.6	97.2	92.5	**95.2**	97.2	94.1	90.9	73.6	**99.3**
Fus	**95.9**	97.5	**100**	**99.9**	**93.3**	96.2	**94.1**	**99.5**	**99.9**	**99.9**	**98.3**	**97.5**	95.1	**98.6**	**95.0**	**91.6**	**80.7**	99.2
Q-stat	0.19	0.07	-0.2	0.87	0.18	0.17	0.16	0.15	0.17	-0.2	0.15	0.13	0.63	0.71	0.80	0.09	0.08	0.81
LPQ	90.3	94.9	**99.9**	99.8	91.3	**95.6**	86.1	97.5	97.6	97.6	90.7	95.3	88.3	98.9	91.9	91.1	78.2	97.8
0.1×(PC+CB)+MM+0.33×(E3)	95.1	**98.3**	**99.9**	**99.9**	92.7	89.0	93.3	99.4	**99.7**	99.6	97.2	92.5	**95.2**	97.2	94.1	90.9	73.6	**99.3**
Fus	**95.4**	97.8	**99.9**	**99.9**	**93.5**	95.4	**92.7**	**99.5**	**99.7**	**99.7**	**97.3**	**96.5**	95.1	**99.2**	**94.9**	**91.7**	**80.7**	**99.3**
Q-stat	0.19	0.08	0.15	0.93	0.19	0.17	0.16	0.10	0.09	-0.2	0.15	0.08	0.61	0.71	0.71	0.09	0.08	0.78
RIC	91.8	97.6	99.2	99.8	90.2	92.9	88.6	97.3	99.0	98.8	90.8	96.6	86.7	97.4	88.6	89.3	80.3	97.0
0.1×(PC+CB)+MM+0.33×(E3)	95.1	98.3	**99.9**	**99.9**	92.7	89.0	93.3	99.4	99.7	99.6	97.2	92.5	**95.2**	97.2	**94.1**	**90.9**	73.6	**99.3**
Fus	**95.6**	**98.5**	**99.9**	**99.9**	**93.3**	**94.1**	**93.6**	**99.5**	**99.8**	**99.7**	**98.3**	**97.7**	95.1	**98.3**	93.5	90.3	**82.3**	99.2
Q-stat	0.19	0.08	-0.2	0.87	0.17	0.16	0.18	0.17	-0.2	0.02	0.14	0.10	0.58	0.66	0.81	0.09	0.03	0.67
disCLBP_v2	87.9	97.0	99.2	99.8	90.3	79.3	86.5	97.1	97.7	51.6	88.9	94.3	87.9	98.0	89.5	88.4	78.2	96.5
disCLBP	88.2	96.2	99.2	99.7	88.7	78.4	77.6	98.0	98.2	52.2	77.1	79.0	88.1	96.8	91.8	88.9	**81.3**	95.8
CLBP	88.0	95.6	99.3	**99.9**	**92.4**	**95.3**	83.9	**98.7**	98.4	98.6	**91.7**	94.9	**89.2**	**98.1**	92.7	89.9	80.1	98.0
RLBP	87.4	94.2	99.0	99.8	89.4	90.3	81.3	97.6	**99.1**	**99.4**	85.5	95.2	85.9	97.4	89.2	88.4	80.4	95.9
RLBPselec	86.5	91.4	**99.7**	99.3	84.2	89.9	86.3	98.4	97.6	51.4	81.9	91.6	82.2	96.5	91.7	84.9	74.3	90.5
MrElbp	87.5	**98.0**	98.6	99.8	91.9	78.0	86.4	98.2	98.4	97.5	**91.7**	90.2	85.9	97.3	88.5	89.3	76.2	97.1
msjlbp	85.7	91.0	98.9	99.1	85.0	91.0	84.8	96.0	98.9	98.2	85.4	**96.8**	84.6	95.4	91.8	81.4	78.5	92.5
HASC	**90.1**	94.2	**99.7**	99.8	87.2	90.6	88.9	97.2	99.0	99.3	88.0	96.7	85.6	**98.1**	93.2	89.8	74.0	96.0
GAB	90.0	91.5	99.3	98.1	83.0	91.0	83.4	95.0	98.0	98.4	89.4	96.7	79.4	95.4	91.6	91.6	66.2	94.5
GOLD	89.1	95.3	99.1	99.8	87.4	80.1	**91.2**	98.2	98.4	96.1	98.7	90.2	88.4	97.0	**93.3**	**92.6**	75.4	**98.2**

Table 8. Comparison of the proposed method with state-of-the-art approaches and further fusions.

For the state-of-the-art handcrafted feature approaches, we used methods where the original code is available, and we used the parameters settings suggested by the original authors. Some performance differences, compared to the original papers, are obtained on some datasets (e.g. disCLBP in the PS dataset). This is due to the different parameter testing protocols. We used the same split for the training/testing sets for all the methods tested in this paper, as well as the same approach for finding the parameters of SVM.

Of course, a higher performance could be obtained if different feature descriptors were combined with different classifiers. The aim of this work, however, is not to propose the best ensemble of descriptors that optimizes the classification step; rather, it is to show how to use different pre-trained CNNs and the different layers of these CNNs for obtaining a very high performing set of features. Since the proposed method is tested on several different image datasets, we are fairly confident about the generalizability of the proposed approach.

E. **Validation against ensembles proposed in the literature**

In this subsection, we have reported the performance of a system based on handcrafted and non-handcrafted features in a set of bioimage datasets used in literature for validating ensembles of descriptors for medical images (cell phenotype recognition, subcellular localization, and histopathological classification). Our method obtains a performance that is comparable with the best ensemble proposed in the literature [65]. Moreover, as Table 9 shows, our method outperforms several other approaches that have been proposed in the last few years. An advantage of our method is that it is not based on complex, computationally expensive image-level statistics of a dense set of local descriptors, as is the case with [65]. Furthermore, the full MATLAB source code for reproducing our experiments is freely available, while the source code used in [65] is not. Thus, our method could be very useful for practitioners.

For the non-handcrafted features, we use the best approach proposed in this work (0.1×(PC+CB)+MM+0.33×(E3)) named NonH in Table 9. For the handcrafted features, we have used the set of handcrafted features proposed in the best ensemble reported in [66], adding the local binary patterns variant proposed in [67]. In Table 9, the set of handcrafted features is named Hand. Our proposed ensemble, named HERE, is given by the weighted sum rule 2×Hand + NonH. Before fusion the scores of Hand and NonH are normalized to mean 0 and standard deviation 1.

The set of biological images is labelled IICBU 2008 Benchmark [68]; some datasets in that suite (CH, HE and RN) are already detailed in Table 3. The other datasets referenced in Table 9 are the following:

- Muscle aging (MA): the classes are images of C. elegans muscles at 4 ages;

- Terminal bulb aging (TB): images of C. elegans terminal bulb at 7 ages (hence, 7 classes);
- Lymphoma (LY): malignant lymphoma of three subtypes;
- Liver gender (LG): liver tissue sections from 6-month male and female mice on a caloric restriction diet (the classes are the 2 genders);
- Liver aging (LA): liver tissue sections from female mice on ad-libitum diet of 4 ages.

As was the case in all the experiments reported above, for our descriptors we always use the same parameters across all the tested datasets to avoid any overfitting. It will also be noted that in Table 9 we use accuracy as the performance indicator as this is the performance indicator that is used in the literature when accessing an ensemble's performance on the IICBU 2008 benchmark dataset.

	CH	**HE**	**RN**	**TB**	**LY**	**MA**	**LG**	**LA**
Hand	97.85	94.42	90.50	70.62	92.00	91.67	**100**	99.62
NonH	99.69	93.14	56.00	71.03	74.67	90.42	99.67	97.90
HERE	**100**	95.93	**91.50**	**72.58**	90.93	92.92	**100**	**100**
[65]	99.90	**98.30**	86.50	64.80	**96.80**	**97.90**	99.60	**100**
[69]	98.50	94.4	67.5	44.6	---	---	---	---
[70]	93	84	82	49	85.0	53	99	51
[71]	93.1	68.3	55.0	51.1	70.9	89.6	91.7	73.8
[72]	99.0	84.0	73.0	55.0	66.0	---	99	89.0
[73]	98.4	90.7	90.1	---	---	---	---	---
[74]	---	---	---	---	92.7	---	99.2	96.4

TABLE 9. Comparison with other set of features proposed in the literature.

Except for [65], our method HERE outperforms all the other approaches reported in Table 9 with a p-value of 0.1. HERE also outperforms the ensemble of handcrafted features named Hand. It is clear that our set of features extracted using CNN builds a very high performing ensemble of descriptors.

V. Conclusions

In this work, the exploitation of deep learning-based features for synthesizing a generic image classification system was analyzed. This was done by describing an input image using a feature vector built with non-handcrafted features: deep transfer learning features based on convolutional neural networks, principal component analysis network, and the compact binary descriptor. The image representation method developed in this paper is general, geared to work efficiently for any image classification problem. This is verified by the high number of different datasets and classification tasks employed while developing the system. Non-handcrafted features extracted from the input images were used for training an SVM on a wide range of image classification problems. Several CNNs were used for extracting a set of feature vectors from an image, which were then used to feed an ensemble of SVMs that were finally combined by sum rule.

The proposed method was compared with an analogous system based on handcrafted features, an image description method widely used in the literature. Several state-of-the-art handcrafted descriptors (LTP and LPQ, and some other powerful LBP variants) were used to compare the efficiency of the image description method proposed in this paper against the state of the art on a number of different datasets.

The large dimensionality of feature sets extracted from the deep layers cannot be handled using the best performing feature selection techniques like [75]: for this reason, DCT and PCA were used in this work. However, this approach can be developed further, using, for example, high performing feature selection, such as kernel partial least square [76] or Lagrange multipliers [77];

these could be applied to the output of DCT and PCA for a further dimensionality reduction, thereby limiting the curse of dimensionality. Alternatively, faster feature selection methods, like Fisher score, Gini index or T-test (Duda, Hart, Stork, and pattern classification), can be directly applied to the original feature sets extracted from the deep layers.

The experimental results demonstrated that the performance achieved using our transfer learning scheme is higher than what is achievable by other standard approaches. The experimental results comparing handcrafted features against non-handcrafted features show that the two systems extract different information from the input images. As a result, the fusion of handcrafted features with the novel features proposed in this paper greatly outperforms the standard approaches with a p-value of 0.01.

The MATLAB source code to replicate our experiments will be made available at (https://www.dropbox.com/s/bguw035yrqz0pwp/ElencoCode.docx?dl=0).

REFERENCES

[1] D.G. Lowe, Distinctive image features from scale-invariant keypoints, International Journal of Computer Vision, 60 (2004) 91-110.

[2] H. Bay, T. Tuytelaars, L.V. Gool, SURF: Speeded up robust features, European Conference on Computer Vision, 1 (2006) 404-417.

[3] A.I. Awad, M. Hassaballah, Image feature detectors and descriptors: Foundations and applications, Studies in Computational Intelligence, Springer1, Berlin, 2016.

[4] J. Shi, C. Tomasi, Good features to track, IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94)1994, pp. 593-600.

[5] N. Cristianini, J. Shawe-Taylor, An introduction to support vector machines and other kernel-based learning methods, Cambridge University Press, Cambridge, UK, 2000.

[6] J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks, 61 (2015) 85-117.

[7] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature, 521.7553 (2015) 436-444.

[8] K. Bora, M. Chowdhury, L.B. Mahanta, M. K. Kundu, A. K. Das, Pap smear image classification using convolutional neural network, Tenth Indian Conference on Computer Vision, Graphics and Image Processing 2016, pp. No. 55.

[9] X.-H. Han, J. Lei, Y.-W. Chen, HEp-2 cell classification using k-support spatial pooling in deep CNNs, Deep learning and data labeling for medical applications, Springer, Berlin, 2016, pp. 3-11.

[10] T.-H. Chan, K. Jia, S. Gao, J. Lu, Z. Zeng, Y. Ma, Pcanet: a simple deep learning baseline for image classification?, IEEE Transactions on Image Processing, 24 (2015) 5017-5032.

[11] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks?, Cornell University, arXiv:1411.1792, 2014.

[12] Q.V. Le, Building high-level features using large scale unsupervised learning, IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)2013, pp. 8595-8598.
[13] D. Erhan, Y. Bengio, A. Courville, P.A. Manzagol, P. Vincent, S. Bengio, Why does unsupervised pre-training help deep learning?, The Journal of Machine Learning Research, 11 (2010) 625-660.
[14] C. Barat, C. Ducottet, String representations and distances in deep Convolutional Neural Networks for image classification, Pattern Recognition in Bioinformatics, 54 (2016) 104-115.
[15] M. Cimpoi, S. Maji, A. Vedaldi, Deep convolutional filter banks for texture recognition and segmentation, arXiv.org, arXiv:1411.6836 (2014).
[16] Y. Gong, L. Wang, R. Guo, S. Lazebnik, Multi-scale orderless pooling of deep convolutional activation features, arXiv.org, arXiv:1403.1840 (2014).
[17] K. He, X. Zhang, S. Ren, S. Sun, Spatial pyramid pooling in deep convolutional networks for visual recognition, Computer Vision – ECCV 20142014, pp. 346-361.
[18] R. Girshick, J. Donahue, T. Darrell, J. Malik, Rich feature hierarchies for accurate object detection and semantic segmentation, CVPR, IEEE2014, pp. . . In CVPR. IEEE, 2014.
[19] B. Athiwaratkun, K. Kang, Feature representation in convolutional neural networks, arXiv.org, arXiv:1507.02313 (2015).
[20] B. Yang, B. Yan, B. Lei, S.Z. Li, Convolutional channel features, IEEE International Conference on Computer Vision (ICCV)2015.
[21] A.S. Razavian, H. Azizpour, J. Sullivan, S. Carlsson, CNN features off-the-shelf: an astounding baseline for recognitio, CoRR, 1403.6382 (2014).
[22] L. Nanni, S. Brahnam, A. Lumini, Double committee adaBoost, Journal of King Saud University, 25 (2013) 29-37.
[23] L. Nanni, A. Lumini, S. Brahnam, An empirical study of different approaches for protein classification, The Scientific World Journal, Article ID 236717 (2014) 1-17.
[24] L. Nanni, S. Brahnam, S. Ghidoni, A. Lumini, Toward a General-Purpose Heterogeneous Ensemble for Pattern Classification, Computational Intelligence and Neuroscience, Article ID 909123 (2015) 1-10.
[25] J. Lu, E.L. Venice, Z. Xiuzhuang, J. Zhou, Learning compact binary face descriptor for face recognition, IEEE Trans Pattern Analysis Mach Intell, (2015).
[26] Z. Pan, Z.T. Deng, Dimensionality reduction via kernel sparse representation, Frontiers of Computer Science, 8 (2014) 807-815.
[27] Y.Y. Zhang, J.C. Zhang, Z.C. Pan, Z.D. Q., Multi-view dimensionality reduction via canonical random correlation analysis, Frontiers of Computer Science, 10 (2016) 856-869.
[28] L.I. Kuncheva, C.J. Whitaker, Measures of Diversity in Classifier Ensembles and their Relationship with the ensemble accuracy, Mach Learn, 51 (2003) 181-207.
[29] G. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets., Neural computation, 18 (2006) 1527-1554.
[30] A. Vedaldi, K. Lenc, MatConvNet-convolutional neural networks for MATLAB, Cornell University, arXiv:1412.4564, 2014.
[31] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, Cornell University, arXiv:1409.1556v6 2014.
[32] K. Chatfield, K. Simonyan, A. Vedaldi, A. Zisserman, Return of the devil in the details: delving deep into convolutional networks, BMVC 20142014.
[33] A. Vedaldi, Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, T. Darrell, Convolutional architecture for fast feature embedding, Cornell University, arXiv:1408.5093v1, 2014.
[34] D. Casanova, J. Joaci de Mesquita, O.M. Bruno, Plant leaf identification using gabor wavelets,, International Journal of Imaging Systems and Technology, 19 (2009) 236 – 243.
[35] R.O. Duda, P.E. Hart, Pattern classification and scene analysis, Academic Press, London, 1973.
[36] E. Feig, S. Winograd, Fast algorithms for the discrete cosine transform, IEEE Transactions on Signal Processing, 49 (1992) 2174-2193.
[37] T. Ojala, M. Pietikainen, T. Maeenpaa, Multiresolution gray-scale and rotation invariant texture classification with local binary patterns, IEEE Trans Pattern Analysis Mach Intell, 24 (2002) 971-987.
[38] X. Tan, B. Triggs, Enhanced local texture feature sets for face recognition under difficult lighting conditions, Analysis and Modelling of Faces and Gestures, LNCS 4778 (2007) 168-182.

[39] V. Ojansivu, J. Heikkila, Blur insensitive texture classification using local phase quantization, ICISP2008, pp. 236–243.
[40] G.-H. Liua, L. Zhang, Y.-K. Hou, Z.-y. Li, J.-Y. Yang, Image retrieval based on multi-texton histogram, Pattern Recognit, 43 (2010) 2380-2389.
[41] R. Nosaka, C.H. Suryanto, K. Fukui, Rotation invariant co-occurrence among adjacent LBPs, ACCV Workshops2012, pp. 15-25.
[42] Z. Guo, L. Zhang, D. Zhang, A completed modeling of local binary pattern operator for texture classification, IEEE Transactions on Image Processing, 19 (2010) 1657-1663
[43] Y. Guo, G. Zhao, M. Pietikainen, Discriminative features for texture description, Pattern Recognit Lett, 45 (2012) 3834-3843.
[44] R. Mehta, K. Egiazarian, Dominant rotated local binary patterns (drlbp) for texture classification, Pattern Recognit Lett, 71 (2015) 16-22.
[45] L. Liu, S. Lao, P. Fieguth, Y. Guo, X. Wang, M. Pietikainen, Median robust extended local binary pattern for texture classification, IEEE Trans Image Process, (In press).
[46] X. Qi, S. Linlin, G. Zhao, Q. Li, M. Pietikainen, Globally rotation invariant multi-scale co-occurrence local binary pattern, Image and Vision Computing, (2015).
[47] M. San Biagio, M. Crocco, M. Cristani, S. Martelli, V. Murino, Heterogeneous auto-similarities of characteristics (hasc): Exploiting relational information for classification, IEEE Computer Vision (ICCV13)2013, pp. 809-816.
[48] J. Jantzen, J. Norup, G. Dounias, B. Bjerregaard, Pap-smear benchmark data for pattern classification, Nature inspired Smart Information Systems (NiSIS), Albufeira, Portugal, 2005, pp. 1–9.
[49] G. Kylberg, M. Uppström, I.-M. Sintorn, Virus texture analysis using local binary patterns and radial density profiles, in: S. Martin, S.-W. Kim (Eds.) 18th Iberoamerican Congress on Pattern Recognition (CIARP)2011, pp. 573-580.
[50] M.V. Boland, R.F. Murphy, A neural network classifier capable of recognizing the patterns of all major subcellular structures in fluorescence microscope images of HeLa cells, BioInformatics, 17 (2001) 1213-1223.
[51] F. Yuan, Video-based smoke detection with histogram sequence of LBP and LBPV pyramids, Fire Safety Journal, 46 (2011) 132-139.
[52] A. Cruz-Roa, J.C. Caicedo, F.A. González, Visual pattern mining in histology image collections using bag of features, Artificial Intelligence in Medicine, (2011) 91-106.
[53] G.B. Junior, A. Cardoso de Paiva, A.C. Silva, A.C. Muniz de Oliveira, Classification of breast tissues using Moran's index and Geary's coefficient as texture signatures and SVM, Computers in Biology and Medicine, 39 (2009) 1063-1072.
[54] L. Nanni, J.-Y. Shi, S. Brahnam, A. Lumini, Protein classification using texture descriptors extracted from the protein backbone image, Journal of Theoretical Biology, 3 (2010) 1024-1032.
[55] N. Hamilton, R. Pantelic, K. Hanson, R.D. Teasdale, Fast automated cell phenotype classification, BMC Bioinformatics, (2007) 8-110.
[56] D. Borghesani, C. Grana, R. Cucchiara, Miniature illustrations retrieval and innovative interaction for digital illuminated manuscripts in multimedia systems, Multimedia Systems, 20 (2014) 65-79.
[57] F. Khan, S. Beigpour, J. van de Weijer, M. Felsberg, Painting-91: a large scale database for computational painting categorization, Machine Vision and Applications, 25 (2014) 1385-1397.
[58] L. Nanni, Y.M.G. Costa, A. Lumini, M.Y. Kim , S.R. Baek, Combining visual and acoustic features for music genre classification, Expert Systems with Applications, 45 (2015) 108-117.
[59] Y. Kaya, L. Kayci, Application of artificial neural network for automatic detection of butterfly species using color and texture features, The Visual Computer, 30 (2013).
[60] A. Oliva, A. Torralba, Modeling the shape of the scene: a holistic representation of the spatial envelope, International Journal of Computer Vision, 42 (2001) 145-175.
[61] T. Fawcett, ROC graphs: Notes and practical considerations for researchers, HP Laboratories, Palo Alto, USA, 2004.
[62] T. Landgrebe, R. Duin, A simplified extension of the area under the ROC to the multiclass domain, Seventeenth annual symposium of the pattern recognition association of South Africa2006, pp. 241-245.
[63] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research, 7 (2006) 1-30.

[64] G. Serra, C. Grana, M. Manfredi, R. Cucchiara, Gold: Gaussians of local descriptors for image representation, Computer Vision and Image Understanding, 134 (2015) 22–32.
[65] Y. Song, W. Cai, H. Huang, D. Feng, Y. Wang, M. Chen, Bioimage classification with subcategory discriminant transform of high dimensional visual descriptors, BMC Bioinformatics, 17 (2016) 465.
[66] L. Nanni, M. Paci, F.L.C.D. Santos, S. Brahnam, J. Hyttinen, Review on texture descriptors for image classification, in: S. Alexander (Ed.) Computer Vision and Simulation: Methods, Applications and Technology, Nova Publications, Hauppauge, NY, 2016.
[67] Z. Zhu, X. You, C.L.P. Chen, D. Too, W. Ou, X. Jiang, J. Zoe, An adaptive hybrid pattern for noise-robust texture analysis, Pattern Recognit, 48 (2015) 2592-2608.
[68] L. Shamir, N.V. Orlov, D.M. Eckley, I. Goldberg, IICBU 2008: a proposed benchmark suite for biological image analysis, Medical & Biological Engineering & Computing, 46 (2008) 943–947.
[69] L.P. Coelho, J.D. Kangas, A.W. Naik, E. Osuna-Highley, E. Glory-Afshar, M. Fuhrman, R. Simha, P.B. Berget, J.W. Jarvik, R.F. Murphy, Determining the subcellular location of new proteins from microscope images using local features, Bioinformatics, 29 (2013) 2343-2352.
[70] L. Shamir, N. Orlov, E.D. M., T.J. Macura, J. Johnston, I.G. Goldberg, Wndchrm - an open source utility for biological image analysis, Source Code Biol Med, 3 (2008) 13.
[71] J. Zhou, S. Lamichhane, G. Sterne, B. Ye, P. H., BIOCAT: a pattern recognition platform for customizable biological image classification and annotation, BMC Bioinformatics., 14 (2013) 291.
[72] V. Uhlmann, S. Singh, A.E. Carpenter, CP-CHARM: segmentation-free image classification made accessible, BMC Bioinformatics, 17 (2016) 51.
[73] B. Zhang, T.D. Pham, Phenotype recognition with combined features and random subspace classifier ensemble, BMC Bioinformatics, 12 (2011) 128.
[74] T. Meng, L. Lin, M. Shyu, S. Chen, Histology image classification using supervised classification and multimodal fusion, IEEE International Symposium on Multimedia2010, pp. 145–152.
[75] W. Shao, Y. Ding, H.-B. Shen, D. Zhang, Deep model-based feature extraction for predicting protein subcellular localizations from bio-images, Frontiers of Computer Science, 11 (2017) 243–252.
[76] M. Gutkin, R. Shamir, G. Dror, SlimPLS: a method for feature selection in gene expression-based disease classification, PLoS ONE, 4 (2009) e6416.
[77] S. Sun, Q. Peng, X. Zhang, Global feature selection from microarray data using Lagrange multipliers, Knowl-Based Syst, (2016).