Elsevier Editorial System(tm) for Neurocomputing
Manuscript Draft

Manuscript Number:

Title: Image Receptive Fields for Artificial Neural Networks

Article Type: Full Length Article (FLA)

Keywords: Computer vision; Visual object recognition; Artificial neural networks model; Supervised learning algorithm; Visual receptive fields; Extreme Learning Machine

Corresponding Author: Prof. Jean-Philippe Urban,

Corresponding Author's Institution: Université de Haute Alsace

First Author: Jean-Luc Buessler

Order of Authors: Jean-Luc Buessler; Philippe Smagghe; Jean-Philippe Urban

Abstract: This article describes the structure of the Image Receptive Field Neural Network (IRF-NN), introduced recently by our team. This structure extends simplified learning introduced by Extreme Learning Machine and Reservoir Computing techniques to the field of images.

Neurons are organized in a single hidden layer feedforward network architecture with an original organization of the network's input weights. To represent color images efficiently, without prior feature extraction, the weight values linked to a neuron are determined by a 2-D Gaussian function. The activation of a neuron by an image presents the properties of a nonlinear localized receptive field, parameterized with a small number of degrees of freedom.

This article shows that an efficient representation of the images is provided by a large number of neurons, each associated to a randomly initialized and constant receptive field. The training step determines only the output weights of the network. It is therefore extremely fast, without retropropagation or iterations, and remains efficient with large sets of images.

The network is easy to implement, presents excellent generalization performances for classification applications, and allows the detection of unknown inputs. The efficiency of this technique is illustrated with several benchmarks, photo and video datasets.

Image Receptive Fields for Artificial Neural Networks

Jean-Luc Buessler, Philippe Smagghe, Jean-Philippe Urban[*]

Faculté des Sciences et Techniques, Université de Haute-Alsace, 4 rue des Frères Lumière, 68093 Mulhouse, France

Abstract

This article describes the structure of the *Image Receptive Fields Neural Network* (IRF-NN), introduced recently by our team. This structure extends simplified learning introduced by *Extreme Learning Machine* and *Reservoir Computing* techniques to the field of images.

Neurons are organized in a single hidden layer feedforward network architecture with an original organization of the network's input weights. To represent color images efficiently, without prior feature extraction, the weight values linked to a neuron are determined by a 2-D Gaussian function. The activation of a neuron by an image presents the properties of a nonlinear localized receptive field, parameterized with a small number of degrees of freedom.

This article shows that an efficient representation of the images is provided by a large number of neurons, each associated to a randomly initialized and constant receptive field. The training step determines only the output weights of the network. It is therefore extremely fast, without retropropagation or iterations, and remains efficient with large sets of images.

The network is easy to implement, presents excellent generalization performances for classification applications, and allows the detection of unknown inputs. The efficiency of this technique is illustrated with several benchmarks, photo and video datasets.

Keywords: Computer vision, Visual object recognition, Artificial neural networks model, Supervised learning algorithm, Visual receptive fields, Extreme Learning Machine.

1. Introduction

The IRF-NN has been designed by our research team with the purpose of easing the use of images in supervised learning applications. Its algorithm, although very simple, allows to work directly on photographs, or images, in color or gray-level, without any preprocessing or prior feature extraction. Training of a multi-class recognition task, for example, is simply achieved by presenting a collection of views and their labels.

The network uses the most elementary neural network architecture: a feedforward neural network with a single hidden layer. The novelty of the approach lies in the initialization of the weights. The weights connected to a neuron are not considered independent, but their values are determined by a 2-D Gaussian function discretized into an input image size bitmap. The activation of a neuron is a function of the spatial integration of the input pixels weighted by the Gaussian function of their position. The gain factor and the sigmoid function of the neuron modulate this response strongly: some visual stimuli cause a response in the quasi linear zone of the sigmoid while others provoke a response close to negative or positive saturation.

Albeit the model is elementary, a neuron of the internal layer of the IRF-NN presents the properties of a receptive field: its response is mostly sensitive to a local region in the image and to specific stimuli; similar stimuli trigger activations of similar magnitude. Considering the global network activation, we will show that the neighborhoods induced in the stimuli space are well adapted to the processing of images, since it can take lighting or color changes into account, as well as position, rotation, or shape variations, thanks to the spatial organization of the weights. It can be observed, for example, that moving an object in the image causes a gradual modification of the response.

The initialization of the weights, and therefore of the receptive field connected to a neuron, depends on a few parameters that are considered as *degrees of freedom* (DOF). A few configuration choices specify the range of these DOF as well as the receptive field type. For example the weights can be either of same sign or centered with zero mean. This latter case favors a response to contrast between the central and peripheral region of the receptive field.

A complementary novelty in the IRF-NN approach is the random initialization of the free parameters of the receptive fields. In practice, all free parameters of the receptive fields are initialized randomly: center and radius, but also magnitude and color sensitivity. Thus, each neuron has a specific and unique sensitivity. Unlike other approaches, no notion of convolution product or weight sharing is involved here. The receptive fields are not duplicated or iterated through the image. Our empirical studies establish that the activation vector of the internal layer forms an image encoding with interesting properties for learning algorithms.

Supervised learning using a set of examples can be implemented in an extremely simple way. It follows a technique suc-

[*] Corresponding author
Email address: jp.urban@uha.fr (Jean-Philippe Urban)

cessfully introduced in the field of dynamic systems as *Reservoir Computing* (RC) [1, 2], and more recently for classification or nonlinear function approximation with the *Extreme learning machine* (ELM) [3]. They are based on the following observation: endowing a network with a randomly initialized internal layer of large size allows to avoid the strenuous adaptation phase of the input weights to the examples. Only the output layer is then adapted through supervised learning and the determination of the output weights becomes a linear problem that can be solved using a simple algorithm, without iterations or local minima problems.

The ELM network uses randomly initialized and independent input weights. A theoretical approach shows that this network, like a Multi-Layer Perceptron, has universal approximation capability [4]. It can approximate any continuous target function and classify any disjoint regions. Why would it not work with images? There is no limitation in size or nature of the input vector. Experiments reveal easily the problem. The ELM network can be configured to achieve classification of photographs, whatever their sizes. Its recognition score can reach 100% on the training set. However, no generalization is observed even for photos that differ only slightly.

Generalization is based on the proximity of the vectors in an appropriate space. It is necessary to design the mapping of the inputs into the internal space for a better representation of images. In particular, an image should not be considered as a table of independent components. Pixel values should be interpreted in correlation with their position and their neighborhood.

The notion of receptive fields in the IRF-NN takes this neighborhood implicitly into account. The main difference with an ELM network resides in the initialization of the input layer, not in a supplementary algorithmic step. The weights of the input layer are not independent random variables since their initialization is based only on a dozen of DOF by neuron. Empirical results confirm that generalization is effective and appropriate for various image characteristics.

The IRF-NN approach allows to process images with an efficiency and simplicity similar to the one that made the success of the RC and ELM approaches. Their fundamentals are identical. An ordinary linear regression is sufficient for an efficient adaptation of the output weights, even with a large number of images. The architecture implemented follows very closely the one of a *single layer feedforward network* (SLFN). The weights of the input layer are determined in a generic way, independently of the images to be processed, using a random draw for the free parameters of the neurons. There is only one global coefficient (noted q) that takes the dynamics of the image set into account to optimize the nonlinear response globally. Once initialized, the input layer remains constant. It can be stored either as a weight matrix or as a table of free parameters of the neurons. The weight vectors can be generated at any time to take images of various sizes into account, without requiring any modifications of the network or its training.

The properties of the IRF-NN appear to be remarkable. The examples presented in this article verify that the algorithm can process a large number of images (several tens of thousands), to distinguish and recognize 1,000 objects, and learning time takes only a few minutes. The network is able to generalize efficiently from only a few views, as well as identify a particular view within a very similar set of images. Photograph processing for object recognition is fast, compatible with real-time video applications.

Several recent conference papers [5, 6, 7] have presented some of the properties of the IRF-NN. The purpose of this paper is to describe further the principle of the neural network with the basic algorithms and the setting of the parameters, to discuss a number of its characteristics and to establish the main network properties.

Section 2 gives a short state of the art and presents the main artificial neural network techniques for image recognition. Section 3 details the IRF-NN architecture, weight initialization for gray-level and color images and the algorithms of the network. Section 4 discusses our concept of *image receptive fields* (IRFs) induced by the organization of the weights. It first gives a general interpretation of the network in terms of random and sparse sampling of a continuous scale-space representation of the input image. Then it provides some detailed information about the networks functioning and its configuration. Section 5 illustrates some IRF-NN properties with various image datasets. It shows that the internal representation forms an efficient encoding of the images. A single linear classifier can then be used to perform classification or recognition of large sets of images.

2. Related works

The neural network developed in this article presents two characteristics that are atypical in the field of image processing. The IRF-NN uses the image directly without prior feature extraction and yet is based on a very simple neural feedforward architecture with only one internal layer. Over the years, the state of the art tends to associate pixel-array inputs to very large multilayer networks. Increasing in size is however not the only path to improvement. Our work is consistent with a few recent publications showing that simple networks can produce stunning and competitive performances.

2.1. Learning techniques applied to images

Object recognition in computer vision has been intensively investigated in the last four decades. When it comes to identify objects from previously selected examples, the task can be assimilated to supervised classification, a field in which nonlinear approaches like neural networks or SVMs have proved themselves very successful (e.g.[8, 9]). An image however differs quite from the signals for which artificial neural networks have proven efficient. It is a large 2-D pixel array, subject to many fluctuations like lighting or angle of view, and in which the object to be recognized represents only a part of the data vector in variable environments. The region representing the object is itself variable in size, position, and even angle. And it gets even more complex when the context of the scene is taken into account, where several objects can be present and induce shadows, reflections, occultations, etc.

An obvious approach is to reduce the number of variables of the problem. The classifier is not presented with an array of pixels but with a vector of descriptors resulting from various processing steps. Invariant feature extraction eases the classification task; if necessary the dimension of the vector is further reduced, *e.g.* by principal axis projection with *principal component analysis* (PCA). The feature-based approach is very general and uses several decision algorithms like nearest neighbors, classification trees, bag-of-works, etc. The advantages of using neural networks in this context are notably discussed in the interesting review of Egmont-Petersen *et al.* [10].

The success of these approaches depends chiefly on appropriate feature selection, which is task specific and subject to the designer skills. There are no generic features available and no theory for feature selection [11]. The improvements of detectors and the evaluation of their performances are an active research area (e.g. [12]). A recent paper by Andreopoulos and Tsotsos [13] presents a very detailed overview of the object recognition literature. It emphasizes that the role of learning algorithms has become much more important in recent years, with more advanced techniques.

In some works, the feature selection is integral part of learning. The cascade of classifiers approach of Viola and Jones [14] uses a very large number of elementary features, initially far larger than the number of pixels. The training process progressively excludes inappropriate features and retains only a little part of them that ensure fast recognition processing. The initial features, i.e. the Haar wavelet family, are not task specific. The Viola-Jones detector requires no prior expertise and is an effective tool for recognition. The training, however, remains long and difficult. This seminal work has motivated many of the recent advances, particularly in face detection [15].

2.2. *Direct use of images*

Less well known in the community of image processing, the pixel-based neural approaches work directly with images. They require more computing power but are compatible with today's processors. The Neocognitron, proposed by K. Fukushima in 1980 [16], is the oldest of these architectures and has been steadily refined over the years (e.g. [17, 18]). It organizes the network in layers working as convolutional filters. The successive transformations of the image create a representation that invariant to some deformations like translation, 2D rotation, scaling, etc. This representation is used by the recognition layer with simple decision rules: linear combination and winner-take-all selection. The convolutional scheme of connections allows the neural units to be connected to a small neighborhood of the previous layer and to share their weights. The same operations are then repeated on different parts of the image and the number of free parameters to adapt is reduced.

More recent variants like LeCun's Convolutional Neural Networks [19], or Hinton's Deep Learning architecture [20] obtained remarkable results for some applications like automatic classification of manuscript numbers or characters. Several research teams develop these networks with success on challenging benchmarks, e.g. Cardoso & Wichert [21], Cireşan *et al.* [22], Krizhevsky *et al.* [23].

These networks are relatively large. LeCuns convolutional network is composed of 6 layers, about 10,000 units and 60,000 free trainable parameters. Krizhevskys structure has 8 layers, 650,000 neurons, and 60 million parameters. This complexity is certainly justified by the task, as this network is trained on 1.2 million high-resolution images of the ImageNet LSVRC-2010 contest and recognizes 1,000 classes. But learning cycles took five to six days, which suggests to consider alternatives.

A few recent works point to possible ways for simpler architectures and easier learning. Cox *et al.* [24] evaluate random layers without structured filter kernels, nor any learning. They can obtain networks which achieve state-of-art performance on face recognition tasks. Coates *et al.* [25] obtain high performance for unsupervised classification with single layer networks. Their analysis highlights that a large number of nodes in the hidden layer and dense feature extraction are critical to performance. Saxe *et al.* [26] evaluate networks with a single convolutional layer, and particularly compare performances between random and tuned weights. Their work demonstrates that training of weigths improves performances, but for object recognition, good architecture selection is more important than weight tuning.

Our work extends these last approaches by using a very simple structure. The hidden layer is single, but not convolutionnal. The random initialization of the weights is associated to the idea of receptive fields. Working with only a single hidden layer is a first step to better explore its operation and its potential before adding more layers.

3. Image Receptive Field Neural Network

The IRF-NN retains the structure and functioning of a simple feedforward neural network with one hidden layer. The purpose of the minor algorithmic modifications, compared with the standard algorithm, is to directly accept images as inputs, in the form of an array of gray-level or color pixels.

The input vector becomes therefore very large, with components that are strongly correlated locally. The complexity found in image classification problems necessitates the use of a large number of neurons. This scaling-up of the network size implies some particular adaptations, which remain computationally inexpensive and offer a real efficiency in the learning of images.

This section gives a detailed presentation of the implemented technique. It starts with a brief restatement of the SLFN equations and the specifics of ELM training. The novelty of the proposed network is described with the organization of the weights in terms of receptive fields. In the second part, the configuration parameters and the user settings are discussed, and the section concludes with a synthetic overview of the algorithms.

3.1. *IRF-NN Architecture*

IRF-NN features a classical feedforward architecture, known as *multi-layer perceptron* (MLP) [27], in its simplest form with a single hidden layer (SLFN). Each unit or neuron of this internal layer performs the weighted sum of the inputs using an adaptive weight vector. Its activation is determined by a nonlinear transfer function, for example the hyperbolic tangent function. The network's output is a linear combination of the activated internal neurons.

More formally, the response of a SLFN can be expressed by two equations. Noting $\mathbf{x} \in \mathbf{\Omega} \subset \mathbb{R}^{\mathbf{d}}$ the input vector, M the number of neurons, i the neuron index, $\mathbf{w}_i^{in}$ the input weight vector of a neuron, and w_{qi}^{out} component q of the output weight, the output vector $\hat{\mathbf{s}} \in \mathbb{R}^{\mathbf{m}}$ is defined by elements

$$\hat{s}_q(\mathbf{x}) = \sum_{i=1}^{M} \mathrm{w}_{qi}^{out} \mathrm{h}_i(\mathbf{x}) + \mathrm{w}_{q0}^{out} \tag{1}$$

or in matrix form $\hat{\mathbf{s}}(\mathbf{x}) = \mathbf{W}^{\mathbf{out}}\mathbf{h}$, with $\mathbf{W}^{out} = \left(W_i^{out}\right)_{0 \le i \le M} \in \mathbb{R}^{m\,\mathrm{x}(M+1)}$ and $\mathbf{h}$ the activation vector defined as $h_0 = 1$ and

$$\mathrm{h}_i(\mathbf{x}) = \tanh\left(\sum_{j=1}^{d} w_{ij}^{in} x_j + w_{i0}^{in}\right) = \tanh\left(\mathbf{x}^{\mathrm{T}}\mathbf{w}_i^{in} + \mathrm{w}_{i0}^{in}\right). \tag{2}$$

When the neural network is used for multiclass classification, a standard technique [8] is to encode the desired responses in a binary vector of type *1-of-n*. Let $c = c(k)$ be the class of input k, and m the number of classes, the vector $\mathbf{s}_k \in \mathbb{B}^m$ is defined as $s_{k,c} = 1$ and $s_{k,i} = 0 \;\; \forall i \neq c$. The network response is then estimated as the index of the largest output value

$$\hat{c}(k) = \arg\max_i \left(\hat{s}_i(k)\right) \tag{3}$$

The standard approach of SLFN uses examples $(\mathbf{x}_k, \mathbf{s}_k)$ to adapt both input and output weight matrices in a training phase. The IRF-NN is created with a large hidden layer and keeps the $\mathbf{W}^{in}$ weights constant. These uncomplicated supervised learning techniques have been popularized by the reservoir computing and ELM approaches. A large number of neurons in the hidden layer and a random initialization of the input weights make the adaptation of the input weights needless. Only the output weights $\mathbf{W}^{out}$ have to be adapted, therefore the computation rule is very simple and can be expressed as a linear regression

$$\mathbf{W}^{out} = \mathbf{H}^{\dagger}\,\mathbf{S} \tag{4}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose generalized inverse [28] of the hidden layer activation matrix for N examples, and $\mathbf{S}$ the is the matrix of the associated target vectors. No epochs or iterations are necessary to calculate the weights.

Many recent works confirm the efficiency of this approach in numerous applications. In the following we show that this efficiency is confirmed for inputs as complex as images, by modifying the function of the input weights.

3.2. Receptive fields induced by weight organization

The IRF-NN differs from a SLFN or an ELM mainly in the way its inputs are handled. The input weights are no more considered as independent variables, but are structured to play the role of receptive fields. They are initialized randomly but only with a few DOF per neuron.

The IRF-NN architecture is represented in Figure 1, highlighting the adaptation of the input stream to images. For the sake of clarity and to avoid confusions, the network components are renamed as follows. Input vector $\mathbf{x}$ represents an image and is therefore noted $\mathbf{\Phi}$, index j of the input component represents a pixel and is noted p, weights $\mathbf{W}^{in}$, organized as receptive fields, are noted $\mathbf{G}$, and the adaptive weights $\mathbf{W}^{out}$ are simply noted $\mathbf{W}$.

First we introduce the receptive fields for gray-level images. The input of the IRF-NN becomes a digital image $\mathbf{I} = \mathbf{I}_k$, where index k references the image in a set. Image $\mathbf{I}$, represented by a matrix of dimension $\mathrm{n}_x{\times}\mathrm{n}_y$ needs to be reorganized as a column vector $\mathbf{\Phi}$ of size $d = n_x.n_y$ to be treated as an $\mathbb{R}^d$ input vector in the scalar product of equation (2). The bi-dimensionality of the image and the pixel positions are however not lost in this operation, as they are taken into account by the weight coefficients.

Input weights $\mathbf{g}_i$ associated to neuron i constitute a vector of same dimension as the input image. Its values are generated as a function of the spatial location of the pixel they connect so that weights $\mathbf{g}_i$ form a smooth and localized function in the image coordinates. In this article it is considered that all neurons implement the same type of functions, 2-D elliptic Gaussian functions. These functions have a few parameters that can be adjusted to determine a specific receptive field. In our notation, all parameters associated to a neuron i are represented in vector $\mathbf{\Omega}_i$. The weights are then defined as

$$\begin{aligned} g_{ip} &= g(x_p, y_p, \mathbf{\Omega}_i) \\ &= \gamma_i + \frac{1}{\pi n_x n_y \sigma_x \sigma_y} \exp\left[-\frac{(x_p/n_x - \mu_{x,i})^2}{\sigma_{x,i}^2} - \frac{(y_p/n_y - \mu_{y,i})^2}{\sigma_{y,i}^2}\right] \end{aligned} \tag{5}$$

where x_p, y_p are the coordinates of pixel p, n_x and n_y are the width and height of the image, $\mu_{x,i}, \mu_{y,i}, \sigma_{x,i}, \sigma_{y,i}$ define the center and the width of the function along the x and y axes.

To ease the use of images of various sizes, the pixel coordinates are presented in the form $x_p\ /n_x$ and $y_p\ /n_y$ in the expression of g_{ip}. Parameters μ and σ are thus expressed as reals in the interval $[0, 1]$ of these rescaled coordinates axes.

Two additional coefficients are introduced to scale the amplitude of the dot product: α_i adjusts the slope of the sigmoid function and thus the sensitivity domain of the neuron; q is the network's global gain control parameter that takes the dynamics of the image into account. Its tuning is discussed in section 4.2. Neuron activation (2) is reformulated as

$$h_i(k) = h_i(\mathbf{\Phi}_k) = tanh\left(\frac{1}{q}\left(\alpha_i \mathbf{g}_i^T \mathbf{\Phi}_k + \beta_i\right)\right), \tag{6}$$

and comparison with (2) identifies ${\mathbf{W}_i}^{\mathrm{in}} = [w_{0i}^{in}, \mathbf{w}_i^{in}] = \left[\frac{\beta_i}{q}, \frac{\alpha_i\,\mathbf{g}_i}{q}\right]$.

By setting the parameters of its Gaussian function, each neuron can be associated to a specific receptive field. In gray-level, the set of parameters is

$$\mathbf{\Omega}_i = \left\{\mu_{x,i}, \mu_{y,i}, \sigma_{x,i}, \sigma_{y,i}, \alpha_i, \beta_i, \gamma_i\right\} \tag{7}$$

which determines the center and radius of the Gaussian receptive field, the amplitude of the response and two bias constants. For color images α_i becomes a 3-D vector that will be discussed later. In our approach, these sets $\mathbf{\Omega}_i$ are considered as the DOF of the weight vectors. They are determined randomly during initialization, using a uniform distribution over a bounded interval.

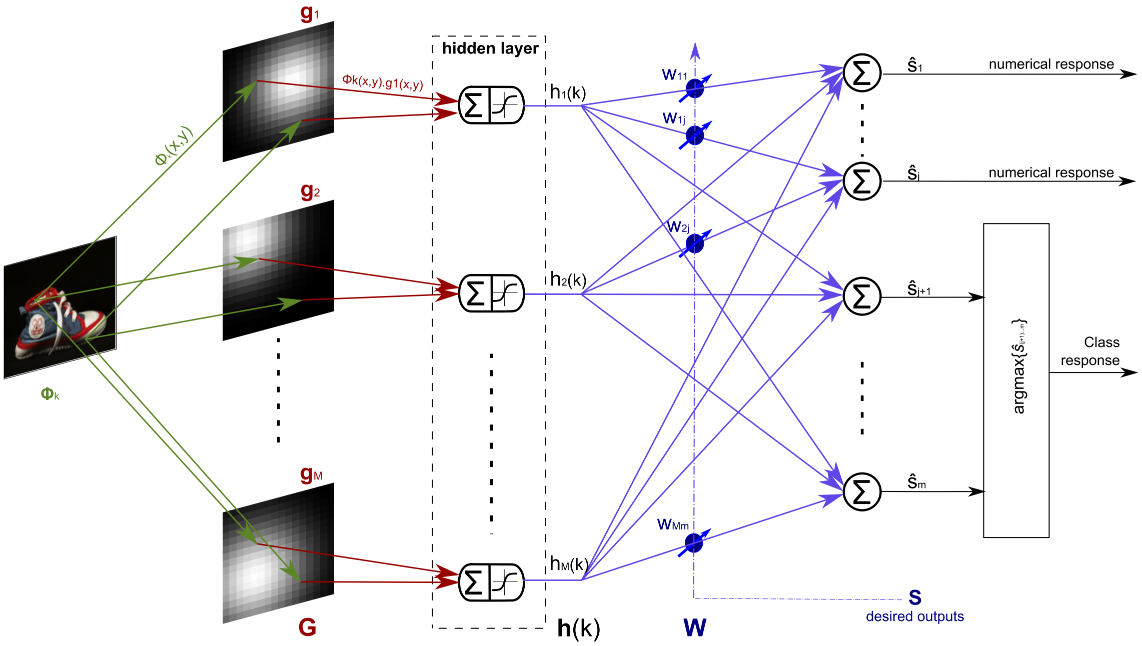

Figure 1: IRF-NN architecture. The raw images are directly presented to the network in vector form. The weights of the hidden layer **G** are organized as random Gaussian receptive fields $\mathbf{g}_i$ that are randomly parameterized at initialization but not modified during learning. Only the output weights **W** are adapted.

Only the bounds of distribution coefficient σ need to be considered as a network configuration parameter. Radius σ can be adjusted according to the nature of the images to be processed, but very generally $\sigma_{.,i} \in [0.01, 0.5]$ leads to good results. Let's consider the other coefficients: $\alpha_i \in [-1,\ 1]$, since the practical amplitude range is adjusted by coefficient q (discussed in section 4.2); $\mu_{.,\mathrm{i}} \in [0, 1]$ to fix the center of the Gaussian inside the image; β is not used in this work, so $\beta_\mathrm{i} = 0$; parameter γ_i will be used to set a zero mean to a proportion of the receptive fields $\mathbf{g}_i$ as presented in section 4.3.

Note that the initialization of the IRF-NN only involves a random selection of parameters $\mathbf{\Omega}_i$. Weight matrix **G** can be computed at initialization time if the images to be processed have all the same size. Otherwise, they can be easily generated as required for each image format.

3.3. Color Receptive Fields

The principle of *image receptive fields* (IRFs) can easily be extended to color images. Figure 2 gives a view of the weights computed for an image of size 200x200 pixels after random initialization of an IRF-NN. The general idea is both to keep the spatial structure of the IRFs and to endow the neuron with a specific sensitivity to color.

This extension can be achieved by only adapting parameter α_i. In a gray-level image this coefficient can be interpreted as the tuning of the neuron's sensitivity to an intensity range. In a color image, for example in RGB space, a coefficient $\alpha_{i,c}$ can be associated to each color plane. These coefficients form vector $\boldsymbol{\alpha}_i$ that determines thus the maximum response axis of neuron i in color space.

The neuron's response is computed by applying its receptive field to each of the three RGB planes and then weighting their values with coefficients α_i. Color is therefore taken into account by introducing the color planes into equation (6),

$$h_i(\mathbf{\Phi}_k) = tanh\left(\sum_{c=1}^{3}\left(\frac{\alpha_{i,c}}{q}\mathbf{g}_i^T\mathbf{\Phi}_{k,c}\right)\right), \tag{8}$$

in which the RGB image $\mathbf{I}_k$ is represented by $\mathbf{\Phi}_k = (\mathbf{\Phi}_{k,c})_{c=1,2,3}$. Each component $\mathbf{\Phi}_{k,c}$ is a vector of size $d = n_x.n_y$ pixels and the color index is noted $c = 1, 2, 3$ for respectively R,G,B. To simplify the notation, coefficient β, unused in this version, is omitted.

For color IRFs, the set of parameters is extended:

$$\mathbf{\Omega}_i = \left\{\mu_{x,i}, \mu_{y,i}, \sigma_{x,i}, \sigma_{y,i}, (\alpha_{i,c})_{c=R,G,B}, \beta_i, \gamma_i, \rho_i\right\}, \tag{9}$$

in which parameter ρ_i is introduced to balance luminance and chrominance sensitivities. As previously, all these values are drawn at initialization time.

The last parameter is used to fix the $\alpha_{i,c}$ distribution. Drawing random and independent values ensures the orientation diversity of the neurons in color space, but their responses remain more sensitive to luminance than to chrominance, as the RGB channels of images are strongly correlated. Many authors use prior transformations that separate the chromatic and achromatic components to increase the efficiency of segmentation operations or image comparisons (e.g. [29, 30]). As linear transformations, noted **T**, are generally preferred, they can be directly incorporated by computing coefficients $\alpha_{i,c} = \alpha'_{i,c}\sum_j \mathrm{T}_{cj}$, after drawing of $\alpha'_{i,c}$.

In the IRF approach with random weights, selection of a particular color transformation matrix **T** is unnecessary. To favor diversity in neuron sensitivity, the idea is to mix luminance and chrominance components in variable proportion. The mixture

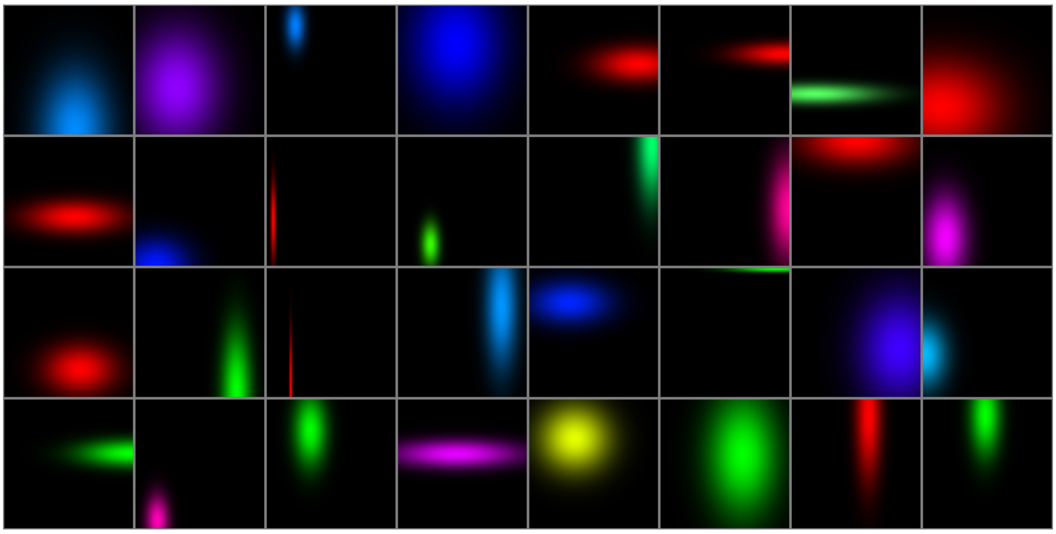

Figure 2: Examples of color Gaussian receptive fields $\mathbf{g}_i$ represented as images.

is obtained without explicit response processing, only by coefficient adjustment. Noting $\alpha_{i,c}$ the values to determine, $\alpha'_{i,c}$ random positive values of mean $\bar{\alpha}'_i$, the response for a color pixel can be decomposed into luminance and chrominance estimations, respectively $r_i^{lu} = \sum_c \alpha'_{i,c}\phi_c$ and $r_i^{ch} = \sum_c (\alpha'_{i,c} - \bar{\alpha}'_i)\phi_c$. Since the desired response verifies $r_i = \sum_c \alpha_{i,c}\phi_c = \rho_i r_i^{lu} + (1 - \rho) r_i^{ch}$, we identify the color coefficients $\alpha_{i,c}$ with

$$\alpha_{i,c} = \alpha'_{i,c} - (1 - \rho_i)\bar{\alpha}'_i = \alpha'_{i,c} - (1 - \rho_i)\frac{1}{3}\sum_{c=1}^{3} \alpha'_{i,c} \tag{10}$$

In practice, random coefficients $\alpha'_{i,c}$ for a color neuron are drawn uniformly in interval $[-1, 1]$, and $\rho_i \in \left[0, C_p\right]$ with boundary C_p, a network configuration parameter set in interval [0, 1].

3.4. Supervised Learning

Learning is considerably simplified when compared with the training of an MLP network. Only the output layer of the network needs to be tuned to a specific set of examples.

The input weights initialized with Gaussian functions are kept constant. Activation vector $\mathbf{h}$ of equation (8) is then a function of the image without any adaptive parameters. This layer is regarded as generic and independent of image sets. It is therefore easy to learn simultaneously several functions of the training set and each output component is a linear function of parameters $\mathbf{W}$ to be determined.

The size of the output vector is not limited, no more than the kind of functions to be learned. According to the needs, outputs can be real values, a classification result for two or multiple classes, a group of several functions, or a multipurpose classification. The use of the network is extremely flexible. Only a set of examples of responses is needed.

These remarks apply in theory to all SLFN approaches and ELM networks. Practically, the linear regression algorithms are limited by the numerical implementation, especially for huge data matrices. It is interesting to observe that the IRF-NN works efficiently with both large internal vectors and a great amount of training images.

Using the previous notation, and considering a color or gray-level image $\mathbf{I}_k$, the network output $\hat{\mathbf{s}} \in \mathbb{B}^m$ is determined by

$$\hat{\mathbf{s}}(\mathbf{I}_k) = \hat{\mathbf{s}}(\mathbf{\Phi}_k) = \mathbf{W}.\mathbf{h}(\mathbf{\Phi}_k) = \sum_{i=1}^{M} \mathbf{w}_i h_i(\mathbf{\Phi}_k) \tag{11}$$

The only adaptive parameters are those of matrix $\mathbf{W} \in \mathbb{R}^{M \times m}$ and training with examples forms linear equations. In this work only *batch training* is considered, although an incremental technique is a practical alternative that we have tested.

Let $\mathbb{D}_A = \{(\mathrm{I}_1, \mathbf{s}_1), ..., (\mathrm{I}_k, \mathbf{s}_k), ...(\mathrm{I}_N, \mathbf{s}_N)\}$ be the set of labeled examples where $\mathbf{s}_k$ is the response vector associated to image $\mathbf{I}_k$. In matrix notation, $\mathbf{H} = \left[\mathbf{h}(\mathbf{x_1})^T \cdots \mathbf{h}(\mathbf{x}_N)^T\right]^T \in \mathbb{R}^{N \times M}$ represents the activation of the M neurons for the N images according to (8), and $\mathbf{S} = [\mathbf{s_1} \cdots \mathbf{s}_N]^T \in \mathbb{R}^{N \times m}$ represents the desired responses for the N images. Matrix $\mathbf{W}$ satisfies output equation $\mathbf{S} = \mathbf{H}.\mathbf{W}$, therefore

$$\mathbf{W} = \mathbf{H}^{\dagger}.\mathbf{S} \tag{12}$$

where $\mathbf{H}^{\dagger}$ is the Moore-Penrose pseudo-inverse of matrix $\mathbf{H}$.

In this context, the $\mathbf{h}_k$ vectors are linearly independent and the network configuration tends to enhance their orthogonality, as discussed in the next sections. Matrix $\mathbf{H}$ is then very large, never singular, but generally close to singularity.

The computationally better way to obtain the pseudo-inverse is then by using the *singular value decomposition* (SVD) [28]. Faster algorithms have been considered, but they have proven to be not stable enough for this application. Most of them use, explicitly or not, quadratic forms $\mathbf{H}^T\mathbf{H}$ or $\mathbf{H}\mathbf{H}^T$ that induce a more limited numerical precision of algorithms. Divergences of the solution are then frequently observed. The SVD implementation of the pseudo-inverse gives a good precision, and can automatically avoid instable responses by discarding the singular values smaller than a tolerance [31].

When the network is used with a relatively large number of neurons compared to the number of examples, i.e. $N < 2M$, it is necessary to introduce a regularization to avoid a decline in generalization (overlearning effect). In this case, the *truncated SVD* (TSVD) variant provides an efficient technique that is coherent with the proposed approach: only the r strongest singular values are kept, the others are replaced with zeros [Hansen 1987]. The pseudo-inverse is therefore computed as

$$\mathbf{H}^{\dagger} = \mathbf{V} \Sigma_k^{+} \mathbf{U}^T \tag{13}$$

Where the $\mathbf{U}$ and $\mathbf{V}$ matrices are determined by the decomposition of $\mathbf{H}$ into singular values $\mathbf{H} = \mathbf{U} \Sigma \mathbf{V}^{\mathrm{T}}$ and $\Sigma_k^{+} = diag(\sigma_1^{-1}, \sigma_2^{-1}, ..., \sigma_k^{-1}, 0, ..., 0) \in \mathbb{R}^{N \times M}$ corresponds to the inverse of Σ truncated at rank $k(rank-k)$. Practically the rank is kept as large as possible, typically $k = M/2$.

4. Interpretation and configuration of Receptive Fields

The originality of our approach lies mainly in the organization of the weights as receptive fields. This section provides some detailed information about their principle, functioning, and discusses some user choices. The learning and response algorithms are simple and configuration free. The guideline, intuitive but largely verified, is to favor the greatest diversity of IRFs, and a good distribution of their variants.

4.1. Gaussian functions

In the work presented in this article, the IRFs are defined as 2-D Gaussian functions. Other functions are potentially possible as long as they present a localized and smooth response. A non-smooth local selection provides usable but not as good results, as can be verified by using a binary rectangular mask where center, size, and amplitude are random parameters specific to each neuron. Comparisons and extensions towards new IRF functions, *e.g.* Gaussian derivatives, are beyond the scope of this introduction article.

To justify choosing the Gaussian function, and to present an interpretation of the network's operation, the strong relation between this approach and the scale-space theory developed by Lindeberg [32] is emphasized. The scale-space representation embeds the original image into a set of gradually smoothed signals, in which the fine scale details are successively suppressed. Starting with image $f(x, y)$, a family L of derived images is defined by the convolution of $f(x, y)$ with Gaussian kernel $g(x, y; t) = \frac{1}{2\pi t} e^{-(x^2+y^2)/2t}$,

$$L(x, y; t) = (g(.,.;t) * f(.,.))(x, y) . \tag{14}$$

The convolution is performed on variables x and y, while t after the semicolon indicates the scale level. This definition works for a continuum of scales t ⩾ 0, but practical implementations use some sampled scale levels.

Lindeberg has studied the properties of this representation for continues signal and for discrete images. The only possible smoothing kernels that satisfy adequate conditions are Gaussian functions. They notably ensure that the views at coarse scales correspond to simplifications of structures at fine scales without artifact elements induced by the smoothing method.

The comparison of IRF with scale-space techniques considers the linear part of the neural response. For the sake of clarity, only gray-level images are considered and intermediate variables are introduced. The neural response vector (6) can be reformulated as $h_i(k) = tanh(z_{i,k}) = tanh\left(\frac{\alpha_i}{q} \ell_{i,k}\right)$ to emphasize the linear computation of argument

$$\ell_{i,k} = \ell_i(\mathrm{I}_k) = \mathbf{g}_i^T(\Omega_i)\,\mathbf{\Phi}_k \tag{15}$$

Unlike Lindeberg's L representation, the IRF-NN approach performs no convolutions and therefore does not create the images at different scales. But it can be easily verified that $\ell_i(I_k)$ is a point $\mathrm{L}(\mathrm{x}, \mathrm{y}; \mathrm{t})$ of the continuous scale-space $\mathrm{L}(.,.;.)$ formed from image I_k. Indeed, convolution at point $(x, y; t)$ is expressed as

$$\begin{aligned} L(x, y; t) &= \sum_{i=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} g(i, j; t) f(x-i, y-j) \\ &= \sum_{i=-\infty}^{+\infty} \sum_{j=-\infty}^{+\infty} g(x-i, y-j; t) f(i, j) \end{aligned} \tag{16}$$

which corresponds to the defined scalar product when the numerical integration covers the complete image, with imposed constraints $\sigma_{\mathrm{x}} = \sigma_{\mathrm{y}}$ and $\mathrm{n_x} = \mathrm{n_y}$ and identifying parameters as $\mathrm{u} = \mathrm{n_x}\mu_{\mathrm{x,i}}$, $\mathrm{v} = \mathrm{n_y}\mu_{\mathrm{y,i}}$, $\mathrm{t} = \mathrm{n_x}{\sigma_{\mathrm{x}}}^2/2$. Therefore $\ell_i(\mathbf{I}_k) = L(n_{\mathrm{x}}\mu_{\mathrm{x,i}}, n_y\mu_{y,\mathrm{i}}; n_x^2\sigma_{x,i}^2/2)$.

The set of scalar products performed by the neurons of the network form vector $\ell(\mathbf{I}_k) = (\ell_i(\mathbf{I}_k))_{i=1}^{M}$ that can be considered as a random and sparse sampling of continuous L, since parameters Ω_i are randomly selected at network initialization. One can note that the L space provides a largely redundant representation since all the scale-space levels are constructed from the initial image.

The IRF representation can therefore be seen as the combination of three virtual stages: representation of the input image in a continuous scale-space, a random sampling of this representation, and a nonlinear transformation. This framework allows analyzing certain properties of the IRF-NN, or considering the choice of functions for receptive fields.

4.2. Sensitivity tuning

Coefficients α_i introduced in (6) are part of the free parameters of the gray-level neuron. They are randomly initialized in the interval [- 1, 1] and control the sign and the gain of the linear computation of each IRF. The role played by these coefficients requires further consideration to get a better grasp of the functioning of the network and to optimize its configuration.

In equation (6), note that gain parameter α_i is associated to a neuron i while q is global for all neurons. Let temporarily $q = 1$ and consider, to illustrate the discussion with a simple case, some weight vector $\mathbf{g}_i$ with $\sum_p g_{i,p} = 1$. Its empirical variance is then $s_{gi}^2 = \frac{1}{n_x n_y}\left(\sum_p g_{i,p}^2 - 1\right)$ and the linear response $l_{i,k}$ in (15) can be expressed as $l_{i,k} = n_x n_y\left(r_{gi,k} s_k s_{gi} + \bar{\mathrm{I}}_k\right)$, bringing up the linear correlation coefficient $r_{gi,k}$ between the two vectors. The response clearly depends on the correlation, but also on the mean value and standard deviation of image $\mathbf{I}_k$. Illumination or contrast changes modify $l_{i,k}$ as well as a change of pattern in the image.

Before applying the nonlinear transformation, the range of these values needs to be analyzed. The purpose of the IRF-NN is to treat any kind of image. The activation vector **h** must characterize all parts of the picture: very small dark objects as well as very large bright ones in a same image. It must also remain sensitive to small or large variations all over the image space. A difference in two images can only be taken into account if it induces a difference in the internal representation **h**, thus a sufficient modification in the response of at least one neuron.

Coefficient α_i controls the amplitude of the argument of the hyperbolic tangent function and therefore the value ranges for which the neuron presents a quasi-linear response, a non-linear response, or an almost constant response that will be assimilated to a saturation zone. Consider a particular image and one IRF, the sensitivity to image variations is best in the quasi-linear response part, its slope is proportional to α_i and decreases rapidly when $\left|z_{i,k}\right| > 0.9$. If we have virtually a lot of IRFs with the same weight vector **g**, but different α_i, all variations can be represented in the activation vector **h** at least by one component. The neurons with high gains can respond to some stimulus in the dark values but will be saturated when the intensity increases. Neurons with smaller gains will take over. Similarly, the global network must be able to detect changes of just a few pixels as well as major image changes. In summary, the role of α_i is to randomly distribute the neurons along some gray-level axis. A wide distribution of neuron gain factors is necessary in the same way that a large distribution of scale factor improves detection of various forms and sizes of objects.

A global network parameter q is used to facilitate the range selection of α_i which are arbitrarily fixed in the interval $[-1, 1]$. Work in progress studies the automatic determination of the optimal value of q. It uses the distribution of the responses for an image set, and thus takes the image representation (e.g., gray-level ranges in $[0, 1]$ or $[0, 255]$) and the effective dynamics of the images into account.

A first step of the study determines boundaries of acceptable values for q. This computation is briefly presented. For a given set of N images with $\mathbf{H} \in \mathbb{R}^{N\times M}$, let $\mathbf{Z}'$ be the linear part computed with $q = 1$:

$$\mathrm{Z}' = \left(\alpha_i \ell_i\left(\mathrm{I}_k\right)\right)_{k=1..N}^{i=1..M}.$$

q will be determined to optimize $\mathbf{H} = tanh(\mathbf{Z}) = tanh(\mathbf{Z}'/q)$. An obvious constraint is to ensure that each image is well represented by vector $\mathbf{h}_k = tanh(\mathbf{Z}_k)$. Any vector whose values are either almot null, or totally saturated with values almost equal to 1, is out of order. To formalize first the non-null rule, let $h_{\min} = 0 + \varepsilon_m$ be the arbitrary fixed minimal boundary. The internal vector $h_{i,k}$ is considered as not-null if $\max_i \left|h_{k,i}\right| > h_{\min}$. All vectors $\mathbf{h}_k$ respect this constraint if $\min_k \left(\max_i \left|h_{k,i}\right|\right) > h_{\min}$. Since *tanh* is an increasing function, the development shows a maximum value for q since

$$\min_k \left(\max_i \tanh\left(\left|\frac{\alpha_i \ell_{i,k}}{q}\right|\right)\right) > h_{\min} \Rightarrow q < \frac{\min_k \left(\max_i |\mathbf{Z}'|\right)}{\operatorname{atanh}(h_{\min})}.$$

The same reasoning holds to determine the minimum using the rule of non-saturation with limit $h_{\max} = 1 - \varepsilon_M$. Then

$$\frac{\max_k \left(\min_i |\mathbf{Z}'|\right)}{\operatorname{atanh}(h_{\max})} < q < \frac{\min_k \left(\max_i |\mathbf{Z}'|\right)}{\operatorname{atanh}(h_{\min})} \tag{17}$$

This relation imposes limits for parameter q. It also provides a quick test of pertinence of internal representation for the image set, and for detecting inconsistent images, if any. The complementary study in progress aims to determine the optimal choice q_{opt} which ensures not only extrema, but also the best distribution of the h_{ij}. Good results for a learning set can be obtained simply by imposing a predeterminated proportion of **H** in the unsaturated range defined by $[h_{min}, h_{max}]$, for example a proportion of 15 or 20%. This method remains empirical and needs futher study. A stronger objective is the determination of q independently of the image set, allowing the use of any image. Two preliminary observations support this approach: performance is weakly sensitive to the value of q, and optimal values empirically identified on various datasets are not too different. The bar graph in figure 3 illustrates the proportion of neural activation components for a few images selected in various datasets. Color is coded in $[0, 255]$, $q = 0.5$, and the experimental protocol is developed in section 5.2. Despite the diversity of the images used, 10 to 30% of the components of vector **h** fall in the unsaturated range $[h_{min}, h_{max}]$. The proportion in this range is heterogeneous in some dataset (COIL or Flowers) but the differences are not really more important between datasets. VIDEO has a high and more homogeneous rate that is close to a few CBS views. MNIST is very particular with its gray-level and almost binary views, resulting in a very high near-null activation rate.

4.3. Receptive Field Support and Normalization

The efficiency of the IRFs can be increased by introducing certain variants or algorithmic precisions in weight initialization step (5). Therefore, the Gaussian functions can be centered to endow the network with sensitivity to local contrasts in the image. The weight vector can be normalized to improve the comparison of signals generated by the various neurons. These variants affect exclusively the determination of weight matrix **G** and do not introduce any supplementary complexity in the training or operation of the IRF-NN.

The notion of receptive field support is now introduced. In image processing applications, Gaussian functions are usually used as filters implemented by a convolution product on the whole image. Therefore their support size is limited to reduce the computational cost. In this work, the Gaussian functions are not used as convolution products. Each weight vector is involved in only one inner product computation and the limitation of the support size affects much less the performances. The main purpose here is not to reduce processing time, but to define a variant of the IRF with zero mean weights on a localized region.

This support size can be defined by the distance of the pixel to the center of the Gaussian, or equivalently by the argument of the Gaussian (5). Let RFS_i be the set of support pixels defined

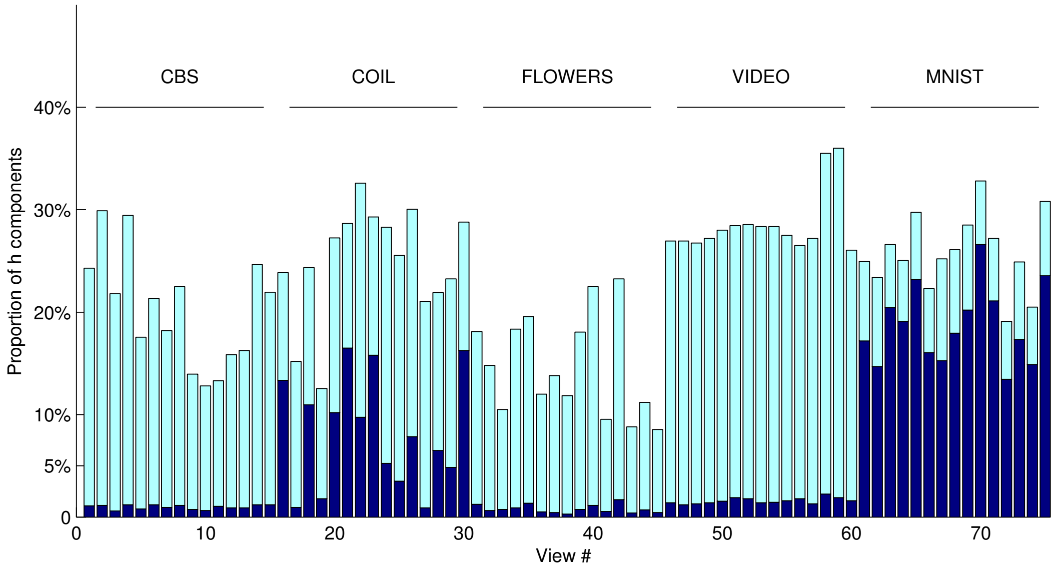

Figure 3: Distribution of the components of internal representation **h**. For each view, sampled from various datasets, the proportion of **h** (in absolute value) is shown for near-null and unsaturated ranges, respectively dark bars: $[0, h_{min}]$ and light bars: $[h_{min}, h_{max}]$.

by

$$RFS_i = \left\{ p \mid \frac{(x_p/n_x - \mu_{x,i})^2}{\sigma_{x,i}^2} + \frac{(y_p/n_y - \mu_{y,i})^2}{\sigma_{y,i}^2} > C_s \right\}, \quad (18)$$

where C_s is a threshold to be fixed ($C_s = 1.6$ in this study). The weights are initialized by (5) with condition

$$\left| \begin{array}{ll} g_{ip} = g(x_p, y_p, \mathbf{\Omega}_i) & if\ p \in RFS_i \\ g_{ip} = 0 & else \end{array} \right. .$$

A variant to these positive IRFs is a *zero mean receptive field* (ZM-RF). Weight vectors with zero mean on all pixels are not very effective. More interesting are the vectors centered only on a local support region, as such IRFs that respond to local contrast $I - \bar{I}_S$, where $\bar{I}_S$ is the mean image intensity of support region RFS_i. Their weights are then positive around the center, negative for a farther ring and zero beyond (Figure 4). The ZM-RFs have therefore ON and OFF zones like the commonly used Laplacian of Gaussian (*Mexican Hat*) or the Difference of 2-D Gaussian functions. A comparison of these functions for IRFs is interesting and will be developed in a forthcoming paper. Constant γ_i in equation (5) is used to handle this centering operation. It is either set to 0 for positive IRFs, or else initialized to

$$\gamma_i = -\bar{g}_{ip} = -\frac{1}{\#RFS_i} \sum\nolimits_{p \in RFS_i} g_{ip}. \quad (19)$$

Experimental results show that this kind of IRFs increase the recognition performances for certain objects or in certain contexts, but not necessary for all images. A general observation is confirmed here: the best results are obtained using the greatest diversity of IRFs. For a given neural network, a mixture of zero null and positive IRFs ensures the sensitivity of the network to both local contrast and intensity of the signal. Therefore we include a proportion C_{ZM} of ZM-RF neurons ($C_{ZM} = 0.9$ in this study).

A last optimization, applied to all IRFs, is weight normalization to adjust the amplitude of the Gaussians. This may appear unnecessary because definition (5) theoretically imposes unity-sum of the weights, i.e. $\sum_p g_{ip} = 1$. Moreover, the neural response is also always multiplied with stochastic gain α_i. However, observations confirm the advantage of this normalization. The rationale of this step is to favor a similar dynamic range of response for all IRFs. The random gain provides then a similar probability distribution for each of them. Note that implicit unity sum assumes infinite support, whereas supports are always limited to image pixels and are often unbalanced, especially when the center of the Gaussian is in the vicinity of the image edge. The normalization of the dynamics of both localized and zero means IRFs is therefore advised.

The amplitude normalization must be consistent for all IRFs, and should take the ones with zero mean into account. Lindeberg [33] shows that the normalization of a Gaussian or its derivatives is equivalent to setting the sum of the positive values always equal to one. This technique is commonly used when the sum of the weights is zero (e.g. [34]), and gives good results for our approach. We retain therefore $g'_{ip} = \frac{1}{\sum_p \max(0, g_{ip})} g_{ip}$ where g_{ip} is determined by (5) or by (18).

5. Network properties and applications

This section presents and illustrates some IRF-NN properties. A few of these properties and perspectives of application have been presented in recent conference papers [5, 6, 7]. We also carried out an in-depth study to determine the parameters of the network, its encoding performances on large image datasets, and its generalization capabilities. This study, that is the subject of a companion article [35], has been conducted on public image databases COIL and ALOI [36] for 3-D objects in rotation and compared to other classification techniques. This

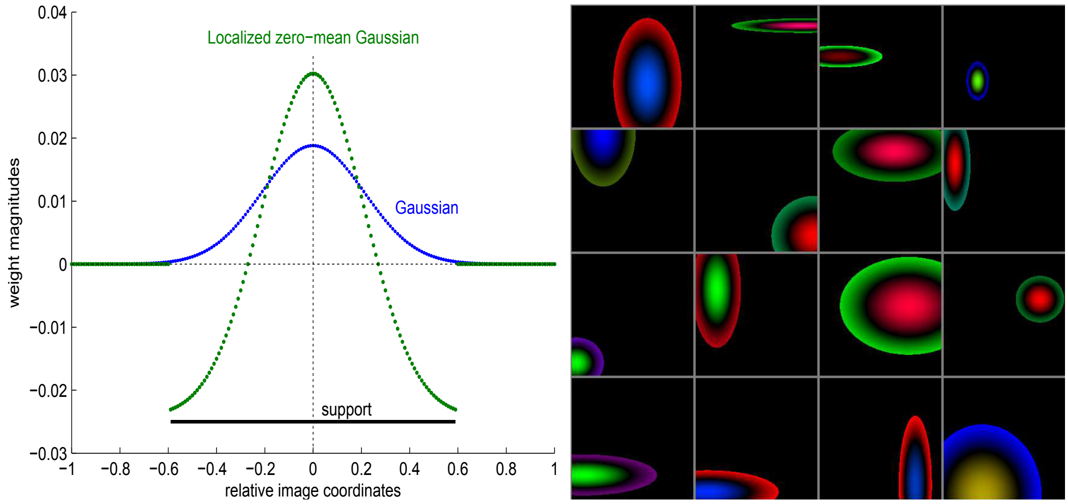

Figure 4: Zero Mean Receptive Fields. Left: 1-D illustration of principle, with null value out of the support range and normalization of amplitude (zero-mean and unity sum of positive values) on support. Right: representation of some random ZM-RF (positive values of color vectors).

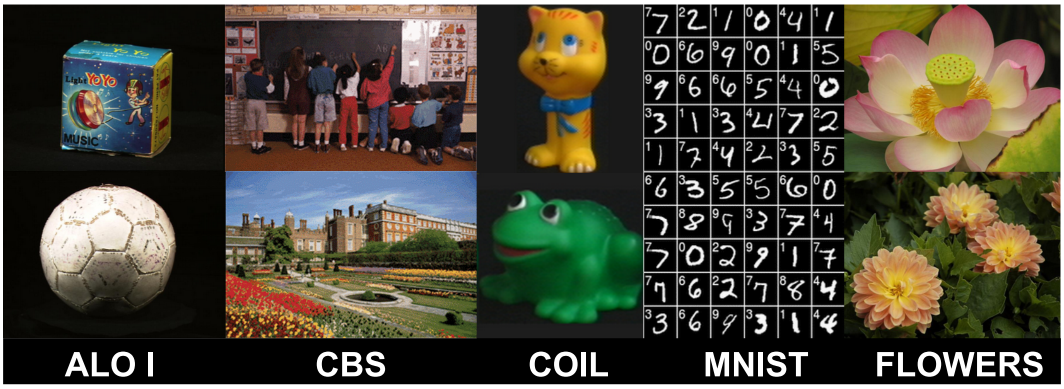

Figure 5: The datasets used in the experiments.

section summarizes the main results and develops new experiments.

5.1. Image datasets

The datasets used for the experiments, all freely available, are presented briefly. The diversity of the tested images is illustrated in figure 5 with a couple of views extracted from each set.

The widely used COIL[1] [37] and the more recent ALOI[2] [36] datasets consist of color views of small 3-D objects in rotation on a turntable, taken at a five degree interval. COIL and ALOI consist respectively of 100 and 1,000 objects, amounting to 7,200 and 72,000 image views.

The *Change Blindness Scenes* (CBS)[3] dataset has been devised to conduct psychological experiments to compare how color, object position, and object presence are encoded in visual memory [38]. CBS has been used to measure the time it takes for observers to detect small scene changes. This set contains 66 color pictures of real scenes, each of which is supplemented with 2-9 variants (300 views in total) where a small element in the scene has been altered in color, position, and presence/absence. The changed versions would look as natural as the original versions (Figure 6).

MNIST[4] is a well-known database of handwritten digits composed of a training set of 60,000 examples and a test set of 10,000 examples. The digits have been size-normalized and centered in a fixed-size image [19].

Flowers collection[5] contains more than 8,000 photos of flowers belonging to 102 different categories [39].

Experiments have also been conducted on video sequences from various sources, including commercials, a movie, and personal videos. For this experiment, MOVIE is defined with the first 100,000 frames (66mn) of the *The Return of the King*[6] movie.

[1] http://www.cs.columbia.edu/CAVE/software/softlib/coil-100.php

[2] http://staff.science.uva.nl/ aloi/

[3] http://wiki.cnbc.cmu.edu/Objects

[4] http://yann.lecun.com/exdb/mnist/

[5] http://www.robots.ox.ac.uk/ vgg/data/flowers/index.html

[6] The Lord of the Rings: The Return of the King. Dir. Peter Jackson. New Line Cinema, 2003.DVD.

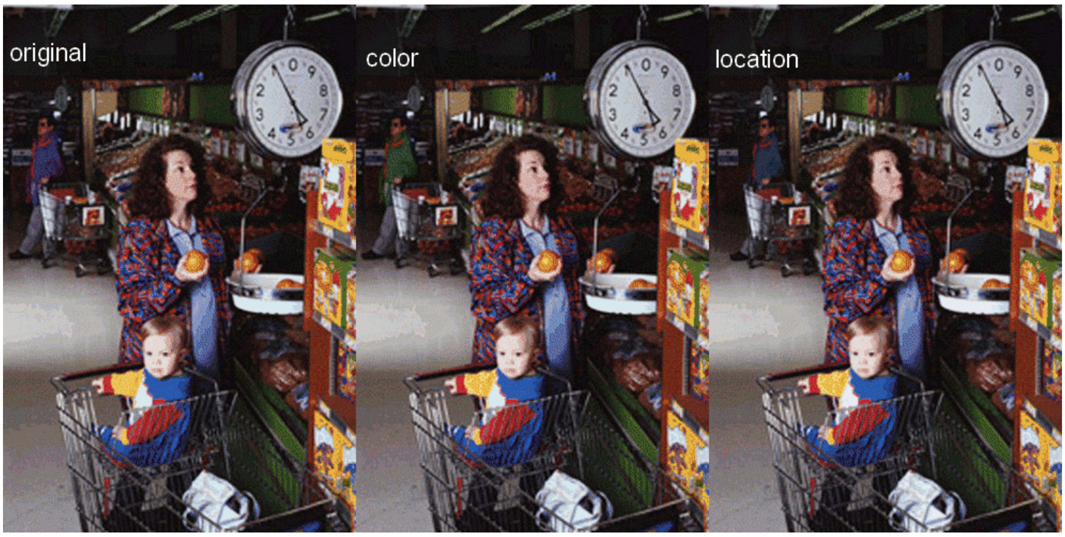

Figure 6: CBS scene (m21055). Variants are based on color or location changes of a minor image element. See the man in the back.

5.2. *Internal representation of images*

Since the input layer is constant, without adaptation to image sets, an important verification is the capacity of the network to represent any image. Internal vector **h** is an image encoding which must be significant for any image, and discriminant and sensitive to small changes. The training and response steps only use this internal data.

Numerous experiments have confirmed that the internal representation is rich, sensitive and discriminant. Properties of this image mapping are easier to point out with supervised training tests, but a direct examination gives an interesting insight. This analysis doesn't require labeled datasets. After the initialization of an IRF-NN, any picture I can be presented. Vector $\mathbf{h}(I)$ is computed by (6) for gray-level images or (8) for color images.

Lets consider an initialized IRF-NN of 2,000 neurons and input some views. The bar graph in figure 3 is an indicator of the neurons' activation ranges for samples of 15 images selected in various datasets (CBS, COIL, FLOWERS, VIDEO, MNIST). For this activation analysis, absolute values are arbitrarily classified in 3 ranges: near-null $[0, \mathrm{h}_{\mathrm{min}}]$, unsaturated $[\mathrm{h}_{\mathrm{min}}, \mathrm{h}_{\mathrm{max}}[$, and quasi-saturated $[\mathrm{h}_{\mathrm{max}}, 1]$, where thresholds used are $h_{min} = .1$, $h_{max} = .98$. The graph verifies that for various kinds of pictures, the representation consists of a sufficient number of neurons in the unsaturated range. Best application results are generally obtained for a proportion from 5 to 20%. The exhaustive test verifies that each image has several components in the unsaturated range (see section 4.2).

To what extent is vector **h** sensitive to image details? Let $\Delta_{\mathbf{h}} = |\mathbf{h}_{\mathrm{k}} - \mathbf{h}_{\mathrm{k+1}}|$ be the difference between the representation of two images k and $k+1$. Figure 7 presents two histograms of $\Delta_{\mathbf{h}}$ for images taken from the CBS dataset. One is for two almost identical views (original and minor color variant of scene m21055 in Figure 6), and the other for views of very different scenes (m21055 and h17146). As expected, the differences are small but significant for similar views, and large for different scenes.

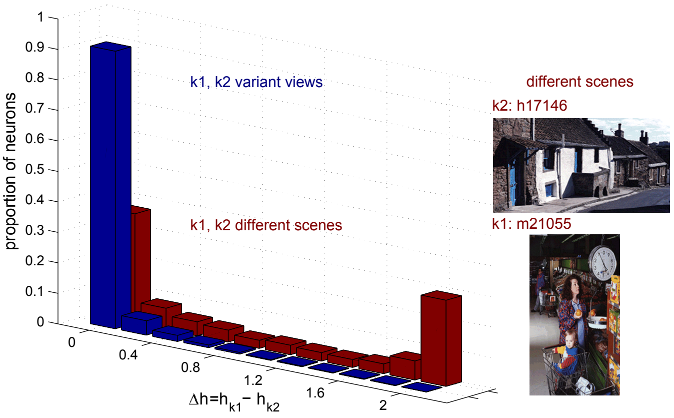

Figure 7: Histograms of the difference between two h vectors.

In similar views, about 90% of the component values of vector h are identical. The amplitudes of the other values are widely distributed and reflect the range of sensitivity of the random IRFs, as shown in Figure 8. The figure plots the evolution of the components for the three variants of scene m21055 shown in Figure 6. Intermediary images are created by linear combination of these views to outline the changes. To the sake of readability, only 150 neurons are drawn; among those presenting large amplitude variations, eight are arbitrarily selected and colored to emphasize the diversity of changes.

The graphs presented here are illustrative. The internal representation depends on network parameters, but also on random initialization and is therefore not strictly identical for two networks. Systematic tests are therefore easier to perform on applications with the whole network, as presented in the next section.

5.3. *Generalization and supervised classification*

The most notable property of the IRF representation is the possibility to realize image classification tasks with a simple linear operator. This property stems from non-linear mapping

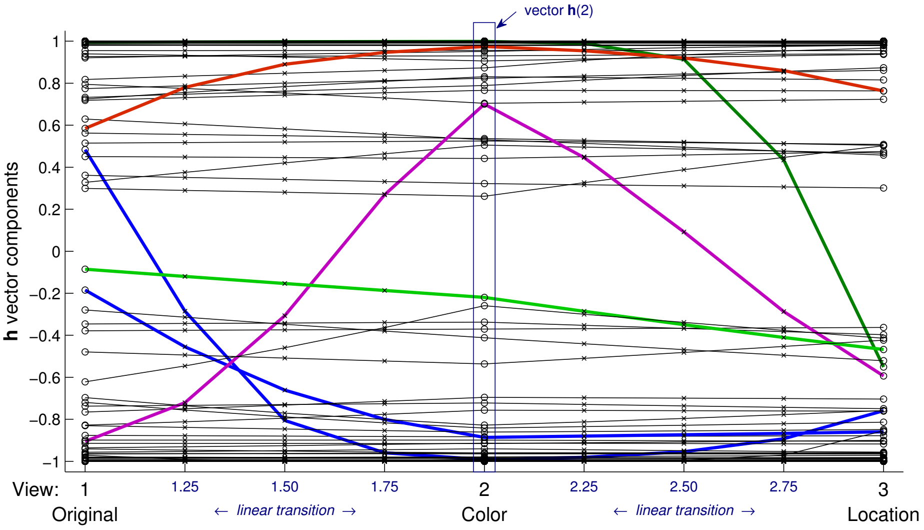

Figure 8: Evolution of vector **h** for variant views.

of inputs in a high-dimensional space and is shared by most neural networks. But it is well known that its effectiveness is related to the kernel functions and parameters of the mapping. The suitability of IRF for images must be experimentally studied. This section presents some results to illustrate classification performances and generalization capacity.

We first need to verify that the network can handle large image sets and can discriminate possibly resembling pictures. The objective is as well to validate the IRF principle and to verify the parameters used for an image set. A quick and easy test is to measure the recognition rate after training and verify that $\tau_{learning}$ = 100%, repeatable for any network initialization. A first test associates a different class to each view to verify that the network is able to distinguish them. A preliminary trial on the 366 CBS views has shown that 44 images cannot be differentiated. It allowed us to detect duplicates and renamed image files in the dataset. Therefore only 322 views are retained for this test and are all correctly recognized after training.

Table 1 shows exhaustive tests on all datasets. In this paper most experiments are realized with 2,000 neuron networks. When the number of views increases, more neurons are necessary to ensure total discrimination. Discussions on the number of neurons will be detailed in next papers. We just report that with fewer neurons, some similar views are more frequently confused. Such confusions appear with MOVIE: about 230 views are not correctly identified with 5,000 neurons. Using 2,000 neurons the misidentifications are about 400; with 30,000 neurons about 20. These confusions are not real errors, they are all labeled with a previous or next frame (maximum offset = 3 frames for 30,000 and 6 frames pour 5,000 neurons). A sufficient diversity of the representation is needed to distinguish very similar images. The global result is surprising: although the movie contains numerous static shots almost all the frames are discriminated; only totally identical dark images can evidently not be distinguished and are not counted in the test.

The generalization capacity can also be statistically evaluated with the recognition rate on image sets. Before discussing exhaustive result, consider again the previous CBS example and observe now response vector $\hat{\mathbf{s}}$. Each component is interpreted as the probability that the tested image belongs to the corresponding class. Figure 9 plots $\hat{\mathbf{s}}$ values for 5 variants of scene m21055. The first network (left) has been trained with one view per scene. Its response for every variant gives a very high probability (near 1) for label m21055 and near 0 for any other class. As expected, the figure confirms a suitable generalization. This example is confirmed for the complete dataset: every view is correctly associated with the original scene, and the classification margins are near 0.9. When the network is trained with 2 independently labeled views of the scene (C2 and P1), score $\hat{\mathbf{s}}$ shows a perfect recognition of the learned images, and can approach 0.5 for intermediate views (figure 9 right). This result is an additional confirmation of the IRF-NN sensitivity and generalization properties.

Numerous experiments have been systematically realized on the datasets and give excellent results for recognition of objects using different views. In the reported tests, the COIL and ALOI datasets are split in two sets, a learning set composed of 18 photos per object with a 20 degree rotation (25% views of the dataset), and a testing set composed by all remaining views, and a testing set composed by all remaining views. Table 2 summarizes the statistical results with recognition rate on the testing set. It shows very good performance for these tasks. The network can work with a very large number of classes (or objects), and a large learning set, as will be illustrated in more detail with ALOI in a forthcoming paper [35]. The results are near the state of the art, and largely better for ALOI, which is remarkable for such a simple architecture.

The computational cost is very competitive. The total duration of the COIL test[7] is about 1 mn and can be detailed: 12s for

[7] The IRF-NN is implemented with Matlab version 7.12 on an Intel Core i7-2600 workstation at 3.40Ghz with 16Gb of memory in the Windows 7 64 bits environment.

Datasets	views	IRF neurons	confusions	Discrimination
CBS	344	2,000	0	100%
COIL	7,200	2,000	0	100%
FLOWERS	8,000	2,000	0	100%
MNIST	60,000	30,000	260±40	99.6%
ALOI	72,000	30,000	0	100%
MOVIE	100,000	5,000	230±40	99.7%
		30,000	22±4	99.98%

Table 1: Discrimination of views by IRF-NNs. All experiments are repeated with 20 randomly initialized networks.

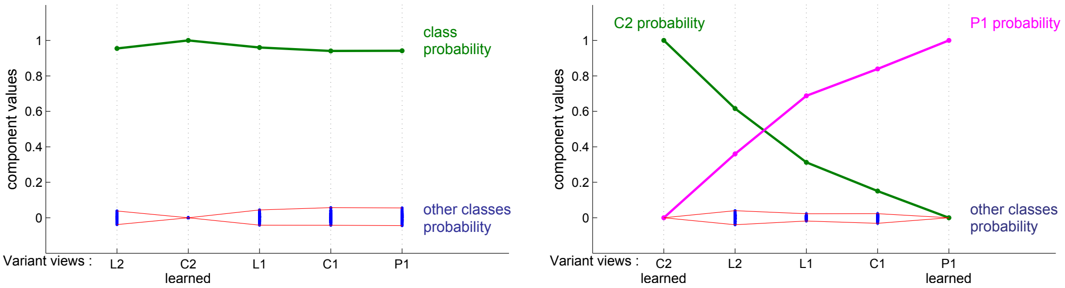

Figure 9: Neural response (blue points) for variant views of m21055. Left: Network trained with 1 view of each the 66 scenes. Right: training with 2 views. Red line : min and max response for other classes.

the loading of the images, 1.8 s for the initialization of the 1,800 neurons; 15 s for the training of the 1,800 images; 34 s for the test of 5,400 images. With 9,000 neurons for the ALOI dataset, the total duration is 25mn. As a comparison, the same test with SalBayes take several hours and a SIFT approach hundreds of hours [40].

5.4. *Novelty detection*

The IRF-NN has an interesting supplementary property: it can distinguish a known image from an unknown one [7]. A novel image induces a network response that differs significantly from the one observed for views similar to the learning set. Analysis and combination of some simple criteria like classification margins and empirical standard deviation of vector $\hat{\mathbf{s}}$ can be used to score novelty or *déjà vu* for any image. This property is efficient when the number of classes is sufficiently large, but the training can be realized with just some examples of each class, without need of extension or negative examples.

This property widens the potential applications of the network considerably. Classification of views can be completed with an unknown response for inputs that do not correspond to any image of the training set. The technique can also be used to score a sliding window which scans a large picture to detect and localize objects. Localization of known objects in a complex scene is very efficient and precise. The IRF-NN provides the possibility to find in a single scan a lot of very different objects and label them.

6. Conclusions and perspectives

The IRF-NN is a variant of feedforward neural network designed to learn images. The current version is composed of a single internal layer where each neuron is connected to every pixel of the image through weighted links. The original idea is to compute the weights of each neuron as a 2-D Gaussian function of pixel positions. One weight vector is defined by a dozen of degrees of freedom that set position, radius, magnitude, and color sensitivity of the neuron. These parameters are randomly initialized and remain constant. Each neuron has therefore a localized sensitivity to image components and its response depends on a non-linear (sigmoid) function, bearing likeness to receptive fields observed in biology.

A very large number of neurons and random initialization of their receptive fields map the images in a high-dimensional space. The internal activation vector is sufficiently rich and sensitive to represent any image. The presented results show that this representation is specific, discriminant for similar views but induces also neighborhood properties when the images present small differences. Interestingly, the neighborhood is effective for various variations in the image: change of position, size or deformation of elements, modification of colors, etc. A large enough IRF-NN can recognize a thousand objects in rotation after training with some example views of each. It can discriminate more of 100,000 photographs or video frames.

The algorithms are simple and fast. The large internal layer obviates adaptation of receptive fields to the image set. The IRF-NN supervised training only adapts the output weights, as

Dataset	# of classes (or objets)	Learning set	Test set	# of Neurons	Rate of class recognition	State of art results
CBS	66	66	278	2,000	100%	-
COIL	100	1,800	5,400	1,000	99.5%	99.9% [41]
MNIST	10	60,000	10,000	7,500	98.6%	99.8% [42]
ALOI	1,000	18,000	54,000	9,000	99.8%	89.6% [40]

Table 2: Recognition of classes (or objects) after training with a labeled training set

ELM networks. The learning algorithm can use a single linear multivariable regression that is very fast, simple, without iterations or local minima problems. The paper details the parameters and options, and shows that implementation and configuration is very easy.

The approach introduces a flexible use of learning in the field of image analysis both for recognition of views and for localization in larger scenes. The internal representation can be used with other classification algorithms; current evaluations show that a few are faster but the described linear technique give always better results and never diverges. Various improvements are under study to accelerate the computations, to work with more pictures, and to extent applications.

References

[1] H. Jaeger, The echo state approach to analysing and training recurrent neural networks-with an erratum note, Bonn, Germany: German National Research Center for Information Technology GMD Technical Report 148 (2001).

[2] D. Verstraeten, B. Schrauwen, M. D'Haene, D. Stroobandt, An experimental unification of reservoir computing methods, Neural Networks 20 (2007) 391–403.

[3] G.-B. Huang, Q.-Y. Zhu, C.-K. Siew, Extreme learning machine: Theory and applications, Neurocomputing 70 (2006) 489–501.

[4] G.-B. Huang, L. Cheng, C.-K. Siew, Universal approximation using incremental constructive feedforward networks with random hidden nodes, IEEE Trans. on Neural Networks 17 (2006) 879.

[5] P. Daum, J.-L. Buessler, J.-P. Urban, Image receptive fields neural networks for object recognition, in: 21st international conference on Artificial neural networks, ICANN'11, volume II, Springer-Verlag, 2011, pp. 95–102.

[6] P. Smagghe, J.-L. Buessler, J.-P. Urban, Novelty detection in image recognition using irf neural networks properties, in: ESANN.

[7] P. Smagghe, J.-L. Buessler, J.-P. Urban, Deja-vu object localization using irf neural networks properties, in: IJCNN, volume to appear.

[8] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford University Press, New York, 1995.

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

[10] M. Egmont-Petersen, D. d. Ridder, H. Handels, Image processing with neural networks - a review, Pattern Recognition 35 (2002) 2279–2301.

[11] T. Tuytelaars, K. Mikolajczyk, Local invariant feature detectors: A survey, Foundations and Trends in Computer Graphics and Vision 3 (2007) 177–280.

[12] K. Mikolajczyk, B. Leibe, B. Schiele, Local features for object class recognition, Tenth IEEE International Conference on Computer Vision, Vols 1 and 2, Proceedings (2005) 1792–1799.

[13] A. Andreopoulos, J. K. Tsotsos, 50 years of object recognition: Directions forward, Computer Vision and Image Understanding 117 (2013) 827–891.

[14] P. Viola, M. Jones, Rapid object detection using a boosted cascade of simple features, in: Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on, volume 1, IEEE, 2001, pp. I–511–I–518 vol. 1.

[15] C. Zhang, Z. Zhang, A survey of recent advances in face detection, Technical report, 2010.

[16] K. Fukushima, Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position, Biological cybernetics 36 (1980) 193–202.

[17] K. Fukushima, Neocognitron for handwritten digit recognition, Neurocomputing 51 (2003) 161–180.

[18] K. Fukushima, Training multi-layered neural network neocognitron, Neural Networks 40 (2013) 18–31.

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

[20] G. E. Hinton, S. Osindero, Y.-W. Teh, A fast learning algorithm for deep belief nets, Neural computation 18 (2006) 1527–1554.

[21] A. Cardoso, A. Wichert, Handwritten digit recognition using biologically inspired features, Neurocomputing 99 (2013) 575–580.

[22] D. Ciresan, U. Meier, J. Masci, J. Schmidhuber, Multi-column deep neural network for traffic sign classification, Neural Networks 32 (2012) 333–338.

[23] A. Krizhevsky, I. Sutskever, G. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems 25, pp. 1106–1114.

[24] D. Cox, N. Pinto, Beyond simple features: A large-scale feature search approach to unconstrained face recognition, in: Automatic Face and Gesture Recognition and Workshops (FG 2011), 2011 IEEE International Conference on, pp. 8–15.

[25] A. Coates, H. Lee, A. Y. Ng, An analysis of single-layer networks in unsupervised feature learning, in: G. Gordon, D. Dunson, M. Dudik (Eds.), 14th International Conference on Articial Intelligence and Statistics (AISTATS), volume 15, JMLR W and CP, 2011.

[26] A. M. Saxe, P. W. Koh, Z. Chen, M. Bhand, B. Suresh, A. Y. Ng, On random weights and unsupervised feature learning, in: 28th International Conference on Machine Learning,.

[27] S. Haykin, Neural Networks: A Comprehensive Foundation, Macmillan College Publishing Company, Inc., New York, 1994.

[28] G. H. Golub, C. F. Van Loan, Matrix computations, Johns Hopkins Studies in Mathematical Sciences, 3 edition, 1996.

[29] Y.-I. Ohta, T. Kanade, T. Sakai, Color information for region segmentation, Computer graphics and image processing 13 (1980) 222–241.

[30] T. Gevers, W. Smeulders, Color based object recognition, Pattern recognition 32 (1999) 453–464.

[31] E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, Lapack users guide, release 2.0, SIAM, Philadelphia 326 (1995) 327.

[32] T. Lindeberg, Scale-space for discrete signals, Pattern Analysis and Machine Intelligence, IEEE Transactions on 12 (1990) 234–254.

[33] T. Lindeberg, Discrete derivative approximations with scale-space properties: A basis for low-level feature extraction, Journal of Mathematical Imaging and Vision 3 (1993) 349–376.

[34] J. C. Russ, The Image Processing Handbook, Second Edition,CRC Press,1995., 2nd edition, 1995.

[35] J.-L. Buessler, P. Smagghe, J.-P. Urban, View recognition of a thousand objects with a simple neural network, to be submitted.

[36] J. M. Geusebroek, G. J. Burghouts, A. W. M. Smeulders, The amsterdam library of object images, International Journal of Computer Vision 61 (2005) 103–112.

[37] S. A. Nene, S. K. Nayar, H. Murase, Columbia Ob ject Image Library (COIL-100), Technical Report, Department of Computer Science, Columbia University, 1996.

[38] V. Aginsky, M. J. Tarr, How are different properties of a scene encoded in visual memory?, Visual Cognition Special Issue on Change Detection and Visual Memory, 7 (2000) 147–162.

[39] M.-E. Nilsback, A. Zisserman, A visual vocabulary for flower classification, in: Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on, volume 2, IEEE, 2006, pp. 1447–1454.

[40] L. Elazary, L. Itti, A bayesian model for efficient visual search and recognition, Visual Search and Selective Attention 50 (2010) 1338–1352.

[41] J. Matas, S. Obdrzalek, Object recognition methods based on transformation covariant features, in: 12th European Signal Processing Conference (EUSIPCO 2004).

[42] D. Ciresan, U. Meier, J. Schmidhuber, Multi-column deep neural networks for image classification, in: Computer Vision and Pattern Recognition (CVPR), 2012 IEEE, IEEE, 2012, pp. 3642–3649.