Understanding image representations by measuring their equivariance and equivalence

Karel Lenc
Visual Geometry Group, University of Oxford
{karel;vedaldi}@robots.ox.ac.uk

Andrea Vedaldi

Abstract

Despite the importance of image representations such as histograms of oriented gradients and deep Convolutional Neural Networks (CNN), our theoretical understanding of them remains limited. Aiming at filling this gap, we investigate three key mathematical properties of representations: equivariance, invariance, and equivalence. Equivariance studies how transformations of the input image are encoded by the representation, invariance being a special case where a transformation has no effect. Equivalence studies whether two representations, for example two different parameterisations of a CNN, capture the same visual information or not. A number of methods to establish these properties empirically are proposed, including introducing transformation and stitching layers in CNNs. These methods are then applied to popular representations to reveal insightful aspects of their structure, including clarifying at which layers in a CNN certain geometric invariances are achieved. While the focus of the paper is theoretical, direct applications to structured-output regression are demonstrated too.

1. Introduction

Image representations have been a key focus of the research in computer vision for at least two decades. Notable examples include textons [11], histogram of oriented gradients (SIFT [14] and HOG [4]), bag of visual words [3][24], sparse [30] and local coding [29], super vector coding [33], VLAD [9], Fisher Vectors [17], and the latest generation of deep convolutional networks [10, 21, 31]. However, despite their popularity, our theoretical understanding of representations remains limited. It is generally believed that a good representation should combine invariance and discriminability, but this characterisation is rather vague; for example, it is often unclear what invariances are contained in a representation and how they are obtained.

In this work, we propose a new approach to study image representations. We look at a representation \( \phi \) as an abstract function mapping an image \( x \) to a vector \( \phi(x) \in \mathbb{R}^d \) and we empirically establish key mathematical properties of this function. We focus in particular on three such properties (Sect. 2). The first one is equivariance, which looks at how the representation changes upon transformations of the input image. We demonstrate that most representations, including HOG and most of the layers in deep neural networks, change in a easily predictable manner with the input (Fig. 1). We show that such equivariant transformations can be learned empirically from data (Sect. 2.1) and that, importantly, they amount to simple linear transformations of the representation output (Sect. 3.1 and 3.2). In the case of convolutional networks, we obtain this by introducing and learning a new transformation layer. By analysing the learned equivariant transformations we are also able to find and characterise the invariances of the representation, our second property. This allows us to quantify invariance and show how it builds up with depth in deep models.

The third property, equivalence, looks at whether the information captured by heterogeneous representations is in fact the same. CNN models, in particular, contain millions of redundant parameters [5] that, due to non-convex optimisation in learning, may differ even when retrained on the same data. The question then is whether the resulting differences are genuine or just apparent. To answer this question we learn stitching layers that allow swapping parts of dif-

Figure 1: Equivariant transformation of CNN filters. Top: Conv1 and Conv2 filters of a convolutional neural network visualised with the method of [23]. Other rows: geometrically warped filters reconstructed from an equivariant transformation of the network output learned using the method of Sect. 2.
different networks. Equivalence is then obtained if the resulting “Franken-CNNs” perform as well as the original ones (Sect. 3.3).

The rest of the paper is organised as follows. Sect. 2 discusses methods to learn empirically representation equivariance, invariance, and equivalence. Sect. 3.1 and 3.2 present experiments on shallow and deep representation equivariance respectively, and Sect. 3.3 on representation equivalence. Sect. 3.4 demonstrates a practical application of equivariant representations to structured-output regression. Finally, Sect. 4 summaries our findings.

Related work. The problem of designing invariant or equivariant features has been widely explored in computer vision. For example, a popular strategy is to extract invariant local descriptors [13] on top of equivariant (also called co-variant) detectors [12, 13, 15]. Various authors have also looked at incorporating equivariance explicitly in the representations [20, 25]. Deep CNNs, including the one of Krizhevsky et al. [10] and related state-of-the-art architectures, are deemed to build an increasing amount of invariance layer after layer. This is even more explicit in the scattering transform of Sifre and Mallat [22].

In all these examples, invariance is a design aim that may or may not be achieved by a given architecture. By contrast, our aim is not to propose yet another mechanism that could make learning of invariances possible, but rather a method to systematically tease out invariance, equivariance, and other properties that a given representation may have. To the best of our knowledge, there is very limited work in conducting this type of analysis. Perhaps the contributions that come closer study only invariances of neural networks to specific image transformations [8, 31]. However, we believe to be the first to functionally characterise and quantify these properties in a systematic manner, as well as being the first to investigate the equivalence of different representations.

2. Notable properties of representations

Image representations such as HOG, SIFT, or CNNs can be thought of as functions $\phi$ mapping an image $x \in X$ to a vector $\phi(x) \in \mathbb{R}^d$. This section describes three notable properties of representations — equivariance, invariance, and equivalence — and gives algorithms to establish them empirically.

Equivariance. A representation $\phi$ is equivariant with a transformation $g$ of the input image if the transformation can be transferred to the representation output. Formally, equivariance with $g$ is obtained when there exists a map $M_g : \mathbb{R}^d \rightarrow \mathbb{R}^d$ such that:

$$\forall x \in X : \; \phi(gx) \approx M_g \phi(x).$$

A sufficient condition for the existence of $M_g$ is that the representation $\phi$ is invertible, because in this case $M_g = \phi \circ g \circ \phi^{-1}$. It is known that representations such as HOG are at least approximately invertible [28]. Hence it is not just the existence, but also the structure of the mapping $M_g$ that is of interest. In particular, $M_g$ should be simple, for example a linear function. This is important because the representation is ultimately used by comparatively simple predictors, such as linear classifiers, or is, in the case of CNNs, an intermediate step in a sequence of quasi-linear filters.

The nature of the transformation $g$ is in principle arbitrary; in practice, in this paper we will focus on geometric transformations such as affine warps and flips of the image.

Invariance. Invariance is a special case of equivariance obtained when $M_g$ is the simplest possible transformation, i.e. the identity function. It is also possible to look for invariance at the level of individual components of the representation. Invariance is often regarded as a key property of representations since one of the goals of computer vision is to establish invariant properties of images. For example, the category of the objects contained in an image is invariant to viewpoint changes. By studying invariance systematically, it is possible to clarify if and where the representation achieves it.

Equivalence. While equi/invariance look at how a representation is affected by transformations of the image, equivalence studies the relationship between different representations. Two heterogeneous representations $\phi$ and $\phi'$ are equivalent if there exist a map $E_{\phi \rightarrow \phi'}$ such that

$$\forall x : \; \phi'(x) \approx E_{\phi \rightarrow \phi'} \phi(x).$$

If $\phi$ is invertible, then $E_{\phi \rightarrow \phi'} = \phi' \circ \phi^{-1}$ satisfies this condition; hence, as for the mapping $M_g$ before, the interest is not just in the existence but also in the structure of the mapping $E_{\phi \rightarrow \phi'}$.

Example: equivariant HOG transformations. Let $\phi$ denote the HOG [4] feature extractor. In this case $\phi(x)$ can be interpreted as a $H \times W$ field of $D$-dimensional feature vectors or cells. If $g$ denotes image flipping around the vertical axis, then $\phi(x)$ and $\phi(gx)$ are related by a well defined permutation of the feature components. This permutation swaps the HOG cells in the horizontal direction and, within each HOG cell, swaps the components corresponding to symmetric orientations of the gradient. Hence the mapping $M_g$ is a permutation and one has exactly $\phi(gx) = M_g \phi(x)$. The same is true for horizontal flips and $180^\circ$ rotations, and, approximately, for $90^\circ$ rotations. HOG implementations [27] do in fact explicitly provide such permutations.

Example: translation equivariance in convolutional representations. HOG, densely-computed SIFT (DSIFT), and

\[\text{Most HOG implementations use 9 orientation bins, breaking rotational symmetry.}\]
convolutional networks are examples of convolutional representations in the sense that they are obtained from local and translation invariant operators. Barring boundary and sampling effects, any convolutional representation is equivariant to translations of the input image as this result in a translation of the feature field.

### 2.1. Learning properties with structured sparsity

When studying equivariance and equivalence, the transformation $M_g$ and $E_{\phi\rightarrow\phi'}$ are usually not available in closed form and must be estimated from data. This section discusses a number of algorithms to do so. The discussion focuses on equivariant transformations $M_g$, but dealing with equivalence transformations $E_{\phi\rightarrow\phi'}$ is similar.

Given a representation $\phi$ and a transformation $g$, the goal is to find a mapping $M_g$ satisfying (1). In the simplest case $M_g = (A_g, b_g)$, $A_g \in \mathbb{R}^{d \times d}$, $b_g \in \mathbb{R}^d$ is an affine transformation $\phi(gx) \approx A_g \phi(x) + b_g$. This choice is not as restrictive as it may initially seem: in the examples above $M_g$ is a permutation, and hence can be implemented by a corresponding permutation matrix $A_g$.

Estimating $(A_g, b_g)$ is naturally formulated as an empirical risk minimisation problem. Given data $x$ sampled from a set of natural images, learning amounts to optimising the regularised reconstruction error

$$E(A_g, b_g) = \lambda R(A_g) + \frac{1}{n} \sum_{i=1}^{n} \ell(\phi(gx_i), A_g \phi(x_i) + b_g),$$

where $R$ is a regulariser and $\ell$ a regression loss whose choices are discussed below. The objective (2) can be adapted to the equivalence problem by replacing $\phi(gx)$ by $\phi'(x)$.

**Regularisation.** The choice of regulariser is particularly important as $A_g \in \mathbb{R}^{d \times d}$ has a $O(d^2)$ parameters. Since $d$ can be quite large (for example, in HOG one has $d = DWH$), regularisation is essential. The standard $l^2$ regulariser $\|A_g\|_F^2$ was found to be inadequate; instead, sparsity-inducing priors work much better for this problem as they encourage $A_g$ to be similar to a permutation matrix.

We consider two such sparsity-inducing regularisers. The first regulariser allows $A_g$ to contain a fixed number $k$ of non-zero entries for each row:

$$R_k(A) = \begin{cases} +\infty, & \exists i : \|A_{g,\cdot,\cdot}\|_0 > k, \\ \|A\|_F^2, & \text{otherwise.} \end{cases}$$

Constraining the rows of $A_g$ rather than the matrix as a whole makes sense because each row can be interpreted as a predictor of a particular component of $\phi(gx)$.

The second sparsity-inducing regulariser is similar, but exploits the convolutional structure of many representations. Convolutional features are obtained from translation invariant and local operators (non-linear filters), such that the representation $\phi(x)$ can be interpreted as a feature field with spatial indexes $(u,v)$ and channel index $t$. Due to the locality of the representation, the component $(u,v,t)$ of $\phi(gx)$ should be predictable from a corresponding neighbourhood $\Omega_{g,m}(u,v)$ of features in the feature field $\phi(x)$ (Fig. 2). This results in a particular sparsity structure for $A_g$ that can be imposed by the regulariser

$$R_{g,m}(A) = \begin{cases} +\infty, & \exists t, t', (u,v), (u',v') \notin \Omega_{g,m}(u,v) : A_{uvt, u'v't'} \neq 0 \\ \|A\|_F^2, & \text{otherwise,} \end{cases}$$

where $m$ denotes the neighbour size and indexes of $A$ have been identified with triplets $(u,v,t)$. The neighbourhood itself is defined as the $m \times m$ input feature sites closer to the back-projection of the output feature $(u,v)$.

**Loss.** As empirically shown in Sect. 3.2, the choice of loss $\ell$ is important. For HOG and similar histogram-like features, the $l^2$, Hellinger’s, or $\chi^2$ distances work well. However, for more sophisticated features such as deep layers in CNNs, it was found that target-oriented losses can perform substantially better in certain cases. To understand the concept of target-oriented loss, consider a CNN $\phi$ trained end-to-end on a categorisation problem such as the ImageNet ILSVRC challenge. A common approach [1, 6, 18] is to use the first several layers $\phi_1$ of $\phi = \phi_2 \circ \phi_1$ as a general-purpose feature extractor. This suggests an alternative objective that preserves the quality of the equivariant features $\phi_1$ in the

\[ \phi_2(x) = \psi(\phi_1(x)) \]

\[ \psi : \mathbb{R}^{d \times d} \rightarrow \mathbb{R}^{d \times d} \]

\[ \psi = \begin{cases} \psi_1, & \text{if } \psi_1 \text{ is good} \\ \psi_2, & \text{otherwise} \end{cases} \]

\[ \psi_1 = \begin{cases} \sqrt{l_2}, & \text{if } l_2 \text{ is good} \\ \sqrt{l_1}, & \text{otherwise} \end{cases} \]

\[ \psi_2 = \begin{cases} \sqrt{l_2}, & \text{if } l_2 \text{ is good} \\ \sqrt{l_1}, & \text{otherwise} \end{cases} \]

\[ \psi_1 = \begin{cases} \sqrt{l_2}, & \text{if } l_2 \text{ is good} \\ \sqrt{l_1}, & \text{otherwise} \end{cases} \]

\[ \psi_2 = \begin{cases} \sqrt{l_2}, & \text{if } l_2 \text{ is good} \\ \sqrt{l_1}, & \text{otherwise} \end{cases} \]
original problem:
\[
E(A_g, b_g) = \lambda R(A_g) + \frac{1}{n} \sum_{i=1}^{n} \ell(y_i, \phi_2 \circ (A_g, b_g) \circ \phi_1(g^{-1}x_i)).
\]

Here \(y_i\) denotes the ground truth label of image \(x_i\) and \(\ell\) is the same classification loss used to train \(\phi\). Note that in this case \((A_g, b_g)\) is learned to compensate for the image transformation, which therefore is set to \(g^{-1}\). This formulation is not restricted to CNNs, but applies to any representation \(\phi_1\) given a target classification or regression task and a corresponding pre-trained predictor \(\phi_2\) for it.

### 2.2. Equivariance in CNNs: transformation layers

The method of Sect. 2.1 can be substantially refined for the case of convolutional representations and certain transformation classes. The structured sparsity regulariser (4) encourages \(A_g\) to match the convolutional structure of the representation. If \(g\) is an affine transformation more can be said: up to sampling artefacts, the equivariant transformation \(M_g\) is local and translation invariant, \(i.e.\) convolutional. The reason is that an affine \(g\) acts uniformly on the image domain \(^3\) so that the same is true for \(M_g\). This has two key advantages: it reduces dramatically the number of parameters to learn and it can be implemented efficiently as an additional layer of a CNN. Such a transformation layer consists of a permutation layer that maps input feature sites \((u, v, t)\) to output feature sites \((g(u, v), t)\) followed by a bank of \(D\) filters, each of dimension \(m \times m \times D\). Here \(m\) corresponds to the size of the neighbourhood \(\Omega_{g,m}(u, v)\) in Sect. 2.1. Intuitively, the main purpose of these filters is to permute and interpolate feature channels.

Note that \(g(u, v)\) does not, in general, fall at integer coordinates. In our case, the permutation layer assigns \(g(u, v)\) to the closest lattice site by rounding but it can be also distributed to the nearest \(2 \times 2\) sites by using bilinear interpolation.\(^4\)

### 2.3. Equivalence in CNNs: stitching layers

The previous section looked at how equivariance can be studied more efficiently in CNNs; this section does the same for equivalence. Following the task-oriented loss formulation of Sect. 2.1, consider two representations \(\phi_1\) and \(\phi_1'\) and a predictor \(\phi_2\) learned to solve a reference task using the representation \(\phi_1\). For example, these could be obtained by decomposing two CNNs \(\phi = \phi_2 \circ \phi_1\) and \(\phi' = \phi'_2 \circ \phi'_1\) trained on the ImageNet ILSVRC data (but \(\phi_1\) could also be

\(^3\)This means that \(g(x + u, y + v) = g(x, y) + (u', v')\).

\(^4\)Better accuracy could be obtained by using image warping techniques. For example, sub-pixel accuracy can be obtained by upsampling in the permutation layer and then allowing the transformation filter to be translation variant (or, equivalently, by introducing a suitable non-linear mapping between the permutation layer and transformation filters).

### 3. Experiments

The experiments begin in Sect. 3.1 by studying the problem of learning equivariant mappings for shallow representations. Sect. 3.2 and 3.3 move on to deep convolutional representations, examining equivariance and equivalence respectively. In Sect. 3.4 equivariant mappings are applied to structure-output regression.

#### 3.1. Equivariance in shallow representations

This section applies the methods of Sect. 2.1 to learn equivariant maps for shallow representations, and HOG features in particular. The first method to be evaluated is sparse regression, followed by structured sparsity. Finally, the learned equivariant maps are validated in example recognition tasks.

---

**Figure 4: Equivariant classification using HOG features.** Classification performance of a HOG-based classifier trained to discriminate dog and cat heads as the test images are gradually rotated and scaled and the effect compensated by equivariant maps learned using LS, RR, and FS.

| HOG size | Iso-Scale \(2\times[\_]\) |
|----------|-------------------------------|
| \(k\) \(m\) | \(3 \times 3\) | \(5 \times 5\) | \(7 \times 7\) | \(9 \times 9\) |
| 5 \(\infty\) | 1.67 | 12.21 | 82.49 | 281.18 |
| 5 1 | 0.97 | 2.06 | 3.47 | 5.91 |
| 5 3 | 1.23 | 3.90 | 7.81 | 13.04 |
| 5 5 | 1.83 | 7.46 | 17.96 | 30.93 |

**Table 1: Regression cost.** Cost (in seconds) of learning the equivariant regressors of Fig. 4. As the size of the HOG arrays becomes larger, the optimisation cost increases significantly unless structured sparsity is considered by setting \(m\) to a small number.
Structured sparsity. The conclusion of the previous experiments is that sparsity is essential to achieve good generalisation. However, learning $M_g$ directly, e.g. by forward-selection or by $l^1$ regularisation, can be quite expensive even if the solution is ultimately sparse. Next, we evaluate using the structured sparsity regulariser (4), where each output feature is predicted from a prespecified neighbourhood of input features dependent on the image transformation $g$. Fig. 3c repeats the experiment of Fig. 3a for a $45^\circ$ rotation, but this time limited to neighbourhoods of $m \times m$ input HOG cells. To be able to span larger intervals of $m$, an array of $15 \times 15$ HOG cells is used. Since spatial sparsity is now imposed a-priori, LS, RR, and FS perform nearly equally well for $m \leq 3$, with the best result achieved by FS with $k = 5$ and a small neighbourhood of $m = 3$ cells. There is also a significant computational advantage in structured sparsity (Tab. 1) as it limits the effective size of the regression problems to be solved. We conclude that structured sparsity is highly preferable over generic sparsity.

Regression quality. So far results have been given in term of the reconstruction error of the features; this paragraph relates this measure to the practical performance of the learned mappings. The first experiment is qualitative and uses the HOGggle technique [28] to visualise the transformed features. As shown in Fig. 5, the visualisations of $\phi(gx)$ and $M_g\phi(x)$ are indeed nearly identical, validating mapping $M_g$. The second experiment (Fig. 4) evaluates instead the performance of transformed HOG features quantitatively, in a classification problem. To this end, an SVM

---

Figure 3: Regression methods. The figure reports the HOG feature reconstruction error (average per-cell Hellinger distance) achieved by the learned equivariant mapping $M_g$ by setting $g$ to different image rotations (3a) and scalings (3b) for different learning strategies (see text). No other constraint is imposed on $A_g$. In the right panel (3c) the experiment is repeated for the $45^\circ$ rotation, but this time imposing structured sparsity on $A_g$ for different values of the neighbourhood size $m$.

Figure 5: Qualitative evaluation of equivariant HOG. Visualisation of the features $\Phi(x)$, $\Phi(gx)$ and $M_g\Phi(x)$ using the $\phi^{-1}$ HOG inverse. $M_g$ is learned using FS with $k = 5$ and $m = 3$ and $g$ is set to a rotation by $45^\circ$ and up/down-scaling by $\sqrt{2}$ respectively. The dashed boxes show the support of the reconstructed features.

Sparse regression. The first experiment (Fig. 3) explores variants of the sparse regression formulation (2). The goal is to learn a mapping $M_g = (A_g, b_g)$ that predicts the effect of selected image transformations $g$ on the HOG features of an image. For each transformation, the mapping $M_g$ is learned from 1,000 training images by minimising the regularised empirical risk (5). The performance is measured as the average Hellinger’s distance $\|\phi(gx) - M_g\phi(x)\|_{\text{Hell.}}$. on a test set of further 1,000 images. Images are randomly sampled from the ImageNet ILSVRC train and validation datasets respectively.

This experiment focuses on predicting a small array of $5 \times 5$ of HOG cells, which allows to train full regression matrices even with naive baseline regression algorithms. Furthermore, the $5 \times 5$ array is predicted from a larger $9 \times 9$ input array to avoid boundary issues when images are rotated or rescaled. Both these restrictions will be relaxed later. Fig. 3 compares the following methods to learn $M_g$: choosing the identity transformation $M_g = I$, learning $M_g$ by optimising the objective (2) without regularisation (Least Square – LS), with the Frobenius norm regulariser for different values of $\lambda$ (Ridge Regression – RR), and with the sparsity-inducing regulariser (3) (Forward-Selection – FS) for a different number $k$ of regression coefficients per output dimension.

As can be seen in Fig. 3a, 3b, LS overfits badly, which is not surprising given that $M_g$ contains 1M parameters even for these small HOG arrays. RR performs significantly better, but it is easily outperformed by FS, confirming the very sparse nature of the solution (e.g. for $k = 5$ just 0.2% of the 1M coefficients are non-zero). The best result is obtained by FS with $k = 5$. As expected, the prediction error of FS is zero for a $180^\circ$ rotation as this transformation is exact (Sect. 2), but note that LS and RR fail to recover it. As one might expect, errors are smaller for transformations close to identity, although in the case of FS the error remains small throughout the range.

---

5The Hellinger’s distance $\left(\sum_i(\sqrt{g_i} - \sqrt{f_i})^2\right)^{1/2}$ is preferred to the Euclidean distance as the HOG features are histograms.
functions, grouped into five convolutional layers (comprising filtering, max-pooling, normalization and ReLU) and three fully-connected layers (filtering and ReLU). The experiments look at the convolutional layers Conv1 to Conv5 right after the linear filters.

Regression methods. The first experiment (Fig. 6) compares different methods to learn equivariant mappings $M_g$ in a CNN. The first method is FS, computed for different neighbourhood sizes $m$ and sparsity $k$. The second is the task-oriented formulation of Sect. 2.1 using a transformation layer. Both the Hellinger reconstruction error of the features and the classification error (task-oriented loss) are reported. As in Sect. 2.2, the latter is the classification error of the compensated network $\phi_M \circ M_g \circ \phi_0 (g^{-1}x)$ in the ImageNet ILSVRC data (the reported error is measured on the validation data, but optimised on the training data). The figure reports the evolution of the loss as more training samples are used. For the purpose of this experiment, $g$ is set to vertical image flip. Fig. 7 repeats the experiments for the task-oriented objective and a complete span of rotations $g$ where performance can slightly deteriorate showing that the regression methods for finding $M_g$ can be improved.\footnote{E.g. by implementing better interpolation techniques in the permutation layer and solving the boundary effect.}

Several observations can be made. First, all methods perform substantially better than doing nothing ($\sim 75\%$ top-1 error), recovering most if not all the performance of the original classifier (43%). This demonstrates that linear equivariant mappings $M_g$ can be learned successfully for CNNs too. Second, for the shallower features up to Conv2, FS is better: it requires less training samples and it has a smaller reconstruction error and comparable classification error than the task-oriented loss. Compared to Sect. 3.1, however, the best setting $m = 3, k = 25$ is substantially less
3.3. Equivalence of deep representations

While the previous two sections studied the equivariance of representations, this section looks at their equivalence. As expected, further validating the approach.

Table 2: CNN equivariance. Performance on the ILSVRC 2012 validation set. Several "Franken-CNNs" obtained by stitching the models of Sect. 2 and Sect. 2.3. The top-1 performance of hybrid models is shown.

| Layer | Horiz. Flip | Vert. Flip | Sc. | Rot. 90° | Top1 | Top5 |
|-------|-------------|------------|-----|----------|------|------|
| Conv1 | 0.46        | 0.46       | 0.46| 0.46     | 0.46 | 0.46 |
| Conv2 | 0.46        | 0.46       | 0.46| 0.46     | 0.46 | 0.46 |
| Conv3 | 0.46        | 0.46       | 0.46| 0.46     | 0.46 | 0.46 |
| Conv4 | 0.46        | 0.46       | 0.46| 0.46     | 0.46 | 0.46 |
| Conv5 | 0.46        | 0.46       | 0.46| 0.46     | 0.46 | 0.46 |

3.4. Testing transformations. Next, we investigate which geometric transformations a CNN (Table 2) can be expected to transform and how similar the corresponding filter predictors are to the identity. The top-1 performance of the unmodified ALEXN-φ and the learned equivariant map φ′ is reported. The resulting error is almost the same as that obtained from the unmodified ALEXN-φ with a mapping substantially more than 95%

Table 3: CNN invariance. Number and percentage of invariant feature channels in the ALEXN network, identified by analysing the corresponding equivariant transformations.

| Layer | Horiz. Flip | Vert. Flip | Rot. 90° |
|-------|-------------|------------|----------|
| Conv1 | 0.46        | 0.46       | 0.46     |
| Conv2 | 0.46        | 0.46       | 0.46     |
| Conv3 | 0.46        | 0.46       | 0.46     |
| Conv4 | 0.46        | 0.46       | 0.46     |
| Conv5 | 0.46        | 0.46       | 0.46     |

Through the identification of invariant features in the representation, we can implicitly learn invariances to such factors. For vertical flips and rotations, the learned equivalent mapping may in fact capture the same visual information by replacing the first few layers with a similar mapping. However, for transformations not experienced during training (such as vertical flips), the learned mapping may not be possible.
changeable in all cases, whereas Conv5 is not fully interchangeable, particularly for Pl.C5. This corroborates the intuition that Conv1 and Conv2 are generic image codes, whereas Conv5 is task-specific. Note however that, even in the worst case, performance is dramatically better than chance, demonstrating that all such features are compatible to an extent.

3.4. Application to structured-output regression

As a complement of the theoretical investigation so far, this section shows a direct practical application of the learned equivariant mappings of Sect. 2 to structured-output regression [26]. In structured regression an input image \( x \) is mapped to a label \( y \) by the function \( \hat{y}(x) = \text{argmax}_{y,z} \Phi(x, y, z, w) \) (direct regression) where \( z \) is an optional latent variable and \( \Phi \) a joint feature map. If \( y \) and/or \( z \) include geometric parameters, the joint feature can be partially of fully rewritten as \( \Phi(x, y, z) = M_y, \phi(x) \), reducing inference to the maximisation of \( (M_y)^T w, \phi(x) \) (equivariant regression). There are two computational advantages: (i) the representation \( \phi(x) \) needs to be computed just once and (ii) the vectors \( M_y, w \) can be precomputed.

This idea is demonstrated on the task of pose estimation, where \( y = g^{-1} \) is a geometric transformation in a class \( g \in G \) of possible poses of an object. As an example, consider estimating the pose of cat faces in the PASCAL VOC [7] data using for \( G \) either (i) rotations or (ii) affine transformations (Fig. 9). The rotations in \( G \) are sampled uniformly every 10 degrees and the ground-truth rotation of a face is defined by the line connecting the nose to the midpoints between the eyes. These keypoints are obtained as the center of gravity of the corresponding regions in the PASCAL part annotations [2]. The affine transformations in \( G \) are obtained instead by clustering the vectors \( [\bar{x}_1^T, \bar{x}_r^T, \bar{x}_n^T]^T \) containing the location of eyes and nose of 300 example faces in the PASCAL VOC data. The clusters are obtained using GMM-EM on the training data and used to map the test data to the same pose classes for evaluation. \( G \) then contains the set of affine transformations mapping the keypoints \( [\bar{x}_1^T, \bar{x}_r^T, \bar{x}_n^T]^T \) in a canonical frame to each cluster center.

The matrices \( M_g \) are pre-learned (from generic images, not containing cats) using FS with \( k = 5 \) and \( m = 3 \) as in Sect. 2. Since cat faces in PASCAL VOC data are usually upright, a second more challenging version of the data (denoted by the symbol \( \circ \)) augmented with random image rotations is considered as well. The direct \( (w, \phi(g^{-1}x)) \) and equivariant \( (w, M_{g^{-1}}\phi(x)) \) scoring functions are learned using 300 training samples and evaluated on 300 test ones.

Table 5 reports the accuracy and speed obtained for HOG and CNN Conv3, Conv4, and Conv5 features for direct and equivariant regression. The latter is generally as good or nearly as good as direct regression, but up to 22 times faster validating once more the mappings \( M_g \). Fig. 8 shows the cumulative error curves for the different regressors.

| \( \phi(x) \) | Bsn | HOG | Conv3 | Conv4 | Conv5 |
|---|---|---|---|---|---|
| Rot \( [\circ] \) [\text{deg}] | 25.8 | 14.9 | 17.0 | 13.3 | 11.0 | 10.5 | 11.1 | 10.1 | 13.4 |
| Rot \( \circ \) [\text{deg}] | 86.9 | 18.9 | 19.1 | 13.2 | 15.0 | 12.8 | 15.3 | 12.9 | 17.4 |
| Aff [-] | 0.35 | 0.25 | 0.25 | 0.25 | 0.28 | 0.24 | 0.26 | 0.24 | 0.26 |
| Speedup [-] | - | 1.219 | 1.86 | 1.93 | 1.93 | 1.23 |

Table 5: Equivariant regression. The table reports the prediction errors for the cat head rotation/affine pose with direct/equivariant structured SVM regressors. The error is measured in expected degrees of residual rotation or as the average keypoint distance in the normalised face frame, respectively. The baseline method predicts a constant transformation.

Figure 8: Equivariant regression errors. Cumulative error curves for the rotation and affine pose regressors of Table 5.

Figure 9: Equivariant regression examples. Rotation (top) and affine pose (bottom) prediction for cat faces in the PASCAL parts data. The estimated affine pose is represented by eyes and nose location. The first four columns contain examples of successful regressions as the last a failure case. Regression uses the CNN Conv5 features computed within the green dashed box.

4. Summary

This paper introduced the idea of studying representations by learning their equivariant and equivalence properties. Using these methods it was shown that shallow representations and the first several layers of deep state-of-the-art CNNs transform in an easily predictable manner with image warps and that they are interchangeable, and hence equivalent, in different architectures. Deeper layers share some of these properties but to a lesser degree, being more task-specific. Analysing these results, it was also possible to quantify the number of invariant feature channels at different layers in CNNs. Besides the theoretical interest of these results, it was suggested that this technology can be fruitfully applied to significantly accelerate structured-output regression classifier in a simple and elegant manner.
References

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

[2] X. Chen, R. Mottaghi, X. Liu, S. Fidler, R. Urtasun, and A. Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014.

[3] G. Couriau, C. R. Dance, L. Dan, J. Williamowski, and C. Bray. Visual categorization with bags of keypoints. In Proc. ECCV Workshop on Stat. Learn. in Comp. Vision, 2004.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proc. CVPR, 2005.

[5] M. Denil, B. Shakibi, L. Dinh, M. Ranzato, and N. de Freitas. Predicting parameters in deep learning. In Proc. NIPS, 2013.

[6] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. Decaf: A deep convolutional activation feature for generic visual recognition. CoRR, abs/1310.1531, 2013.

[7] M. Everingham, A. Zisserman, C. Williams, and L. V. Gool. The PASCAL visual object classes challenge 2007 (VOC2007) results. Technical report, Pascal Challenge, 2007.

[8] I. Goodfellow, H. Lee, Q. V. Le, A. Saxe, and A. Y. Ng. Measuring invariances in deep networks. In Advances in neural information processing systems, pages 646–654, 2009.

[9] H. Jegou, M. Douze, C. Schmid, and P. Pérez. Aggregating local descriptors into a compact image representation. In Proc. CVPR, 2010.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Proc. NIPS, 2012.

[11] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. IJCV, 43(1), 2001.

[12] T. Lindeberg. Principles for automatic scale selection. Technical Report ISRN KTH/NA/P 98/14 SE, Royal Institute of Technology, 1998.

[13] D. G. Lowe. Object recognition from local scale-invariant features. In Proc. ICCV, 1999.

[14] D. G. Lowe. Distinctive image features from scale-invariant keypoints. IJCV, 2(60):91–110, 2004.

[15] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In Proc. CVPR, 2003.

[16] O. Parkhi, A. Vedaldi, C. V. Jawahar, and A. Zisserman. The truth about cats and dogs. In Proc. ICCV, 2011.

[17] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In Proc. CVPR, 2006.

[18] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In CVPR DeepVision Workshop, 2014.

[19] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge, 2014.

[20] U. Schmid and S. Roth. Learning rotation-aware features: From invariant priors to equivariant descriptors. In Proc. CVPR, 2012.

[21] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. volume abs/1312.6229, 2014.

[22] L. Sifre and S. Mallat. Rotation, scaling and deformation invariant scattering for texture discrimination. In Proc. CVPR, 2013.

[23] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. In ICLR Workshop, 2013.

[24] J. Stivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In Proc. ICCV, 2003.

[25] K. Sohn and H. Lee. Learning invariant representations with local transformations. CoRR, abs/1206.6418, 2012.

[26] B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In Proc. NIPS, 2003.

[27] A. Vedaldi and B. Fulkerson. VLFeat – An open and portable library of computer vision algorithms. In Proc. ACM Int. Conf. on Multimedia, 2010.

[28] C. Vondrick, A. Khosla, T. Malisiewicz, and A. Torralba. HOGgles: Visualizing object detection features. In Proc. ICCV, 2013.

[29] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. Proc. CVPR, 2010.

[30] J. Yang, K. Yu, and T. Huang. Supervised translation-invariant sparse coding. In Proc. CVPR, 2010.

[31] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013.

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

[33] X. Zhou, K. Yu, T. Zhang, and T. S. Huang. Image classification using super-vector coding of local image descriptors. In Proc. ECCV, 2010.