Deep Fishing: Gradient Features from Deep Nets

Albert Gordo[1]
albert.gordo@xrce.xerox.com
Adrien Gaidon[1]
adrien.gaidon@xrce.xerox.com
Florent Perronnin[2]
perronnin@fb.com

[1] Computer Vision Group, Xerox Research Centre Europe
[2] Facebook AI Research*

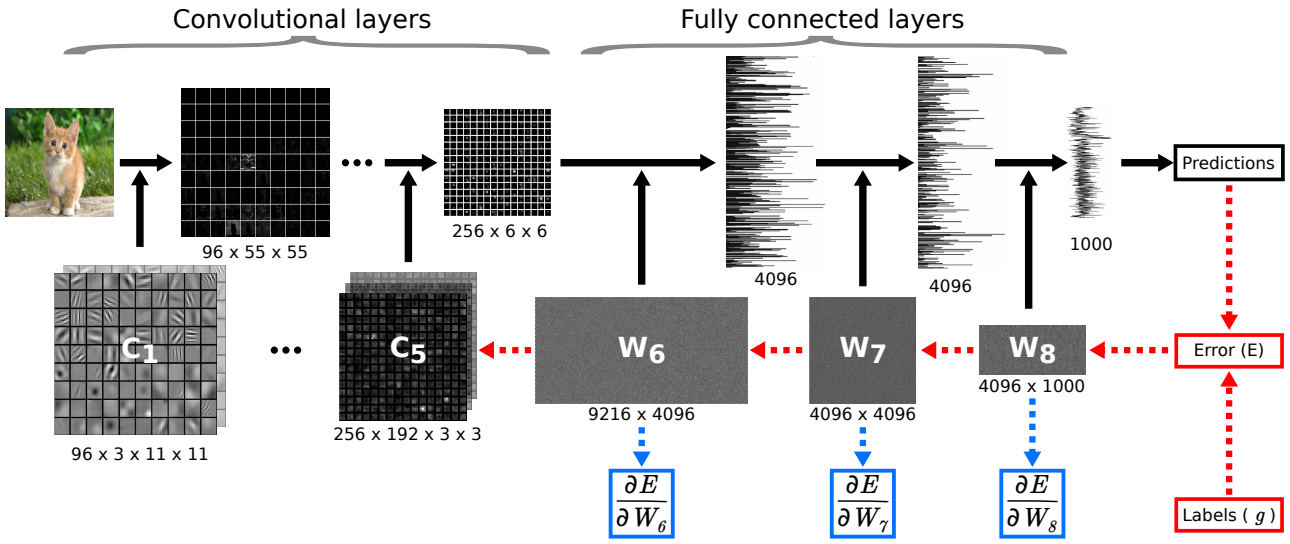

Figure 1: AlexNet architecture. C_k are the parameters (4D tensors) of the convolutional layers. W_k are the parameters (matrices) of the fully connected layers. Black (resp. red) arrows represent the information flow during the forward (resp. backward) pass. **Inspired by the Fisher Kernel, we study the use of gradient-related information (the blue matrices) as transferable representations.**

Convolutional Networks (ConvNets) have recently improved image recognition performance thanks to end-to-end learning of deep feed-forward models from raw pixels. Deep learning is a marked departure from the previous state of the art, the Fisher Vector (FV), which relied on gradient-based encoding of local hand-crafted features. In this paper, we discuss a novel connection between these two approaches. First, we show that one can derive gradient representations from ConvNets in a similar fashion to the FV. Second, we show that this gradient representation actually corresponds to a structured matrix that allows for efficient similarity computation. In particular, for the fully-connected layers, the gradient of the error w.r.t. the weights can be computed as an outer product,

$$\frac{\partial E}{\partial W_k} = x_{k-1}\left[\frac{\partial E}{\partial y_k}\right]^T, \tag{1}$$

where x_{k-1} is the input of layer k, y_k is the output of layer k *before* the non-linearity, and the gradient w.r.t. y_k is easily computed during the back-propagation stage. To compute the back-propagation at test time, where the true label vector g is unknown, we use a uniform vector with values $1/P$, where P is the number of classes.

We show that the trace kernel between gradient representations of two images A and B can be efficiently decomposed in two dot-product subkernels, one on forward features, and one on backward features, that are combined multiplicatively:

$$\begin{aligned} K_{tr}\left(\frac{\partial E}{\partial W_k}(A), \frac{\partial E}{\partial W_k}(B)\right) &= (x_{k-1}^A)^T x_{k-1}^B \cdot \left[\frac{\partial E}{\partial y_k}(A)\right]^T \frac{\partial E}{\partial y_k}(B) \\ &= K_{fw}(A,B) \cdot K_{bw}(A,B). \end{aligned} \tag{2}$$

We experimentally study the benefits of transferring this representation and kernel over the outputs of ConvNet layers, and find consistent improvements on the Pascal VOC 2007 and 2012 datasets with respect to using only the representations based on the forward activations on two popular network architectures (AlexNet and VGG16) pretrained on ImageNet ILSVRC2012 (*cf*. Tables 1 and 2 for results).

Table 1: Left: Results on Pascal VOC 2007 and VOC 2012 with AlexNet (A) and VGG16 (V). Results on VOC 2012 are on the validation set (mAP in %)

Features	VOC2007 (A)	VOC2007 (V)	VOC2012 (A)	VOC2012 (V)
x_5 (Pool5)	71.0	86.7	66.1	81.4
x_6 (FC6)	77.1	89.3	72.6	84.4
x_7 (FC7)	79.4	89.4	74.9	84.6
y_8 (FC8)	79.1	88.3	74.3	84.1
x_8 (Prob)	76.2	86.0	71.9	81.3
$x_5; x_6$	76.4	89.2	71.6	84.0
$\frac{\partial E}{\partial W_6} = x_5\left[\frac{\partial E}{\partial y_6}\right]^T$	80.2	89.3	75.1	84.6
$x_6; x_7$	79.1	89.5	74.3	84.6
$\frac{\partial E}{\partial W_7} = x_6\left[\frac{\partial E}{\partial y_7}\right]^T$	**80.9**	**90.0**	**76.3**	**85.2**
$x_7; y_8$	79.7	89.2	75.3	84.6
$\frac{\partial E}{\partial W_8} = x_7\left[\frac{\partial E}{\partial y_8}\right]^T$	79.7	88.2	75.0	83.4

Table 2: Comparison with other ConvNet results (mAP in %).

	VOC'07	VOC'12
Proposed - AlexNet	80.9	76.5
Proposed - VGG16	**90.0**	85.3
DeCAF	73.4	-
Razavian *et al.*	77.2	-
Oquab *et al.*	77.7	78.7
Zeiler *et al.*	-	79.0
Chatfield *et al.*	82.4	83.2
He *et al.*	80.1	-
Wei *et al.*	81.5	81.7
Simonyan *et al.*	89.7	**89.3**

* Work done while FP was at the Computer Vision Group of the Xerox Research Centre Europe.