# LR-CNN FOR FINE-GRAINED CLASSIFICATION WITH VARYING RESOLUTION

Marion Chevalier, Nicolas Thome, Matthieu Cord, Jérôme Fournier, Gilles Henaff, Elodie Dusch

# LR-CNN FOR FINE-GRAINED CLASSIFICATION WITH VARYING RESOLUTION

*M. Chevalier*[1,2], *N. Thome*[1], *M. Cord*[1], *J. Fournier*[2], *G. Henaff*[2], *E. Dusch*[2]

(1) Sorbonne Universités, UPMC Univ Paris 06, LIP6, 4 place Jussieu 75005 Paris, France
(2) Thales Optronique S.A.S., 2 avenue Gay-Lussac, 78990 Elancourt, France

## ABSTRACT

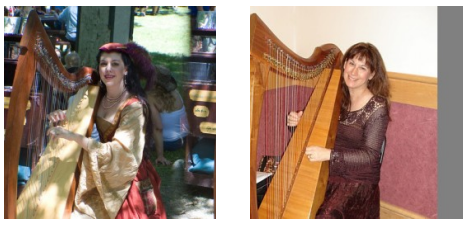
In this work, we present an extended study of image representations for fine-grained classification with respect to image resolution. Understudied in literature, this parameter yet presents many practical and theoretical interests, e.g. in embedded systems where restricted computational resources prevent treating high-resolution images. It is thus interesting to figure out which representation provides the best results in this particular context. On this purpose, we evaluate Fisher Vectors and deep representations on two significant fine-grained oriented datasets: FGVC Aircraft [1] and PPMI [2]. We also introduce LR-CNN, a deep structure designed for classification of low-resolution images with strong semantic content. This net provides rich compact features and outperforms both pre-trained deep features and Fisher Vectors.

***Index Terms***— Fisher vectors, Convolutional neural networks, Fine-grained classification, Image recognition

## 1. INTRODUCTION

Image classification received a lot of interest from Computer Vision and Machine Learning communities in the recent years [3]. Fine-grained classification is a challenging application, consisting in distinguishing very similar classes. Many fine-grained oriented datasets have been proposed to treat this task, e.g. FGVC Aircraft [1], Caltech-UCSD Birds-200-2011 [4], 102 Category Flower Dataset [5], PPMI [2]. In [2], one must determine whether persons are playing or merely holding an instrument. As shown in figure 1, this task requires spotting and recognizing very small discriminating details.

Many methods have recently emerged to address the fine-grained classification problem. Fisher Vectors [6], extending Bag of Visual Words [7, 8, 9, 10, 11], have achieved particularly good results on multiple image classification tasks [12]. Recently, deep networks [13, 14, 15] and more specifically Convolutional Neural Networks have also obtained outstanding performances on several large scale image classification datasets [16, 17, 18]. Due to their complex structure, these nets usually have several dozen million weights: such complex architectures cannot be properly trained on mid-scale datasets. However, recently, they have successfully been used in a transfer fashion: the weights are learned on a very large external dataset (*e.g.* ImageNet [19]), then the net is used unchanged as a deep feature extractor on the target dataset. Such pre-trained deep features have proven very efficient as off-the-shelf features on a few mid-scale datasets [20]. Smaller nets have also been designed so as to be learned on lesser datasets [21], yet they have not been adapted to fine-grained tasks.



(a) Playing the harp (b) Holding the harp

**Fig. 1**. Examples of PPMI images. Discriminating between people playing the instrument and people holding the instrument is a difficult task, even for a human eye.

Our first contribution is to analyze the impact of a specific parameter on Fisher Vector and deep features: input image resolution. In a context of embedded systems, image acquisition process and computational resources can be very limited, thus only low-resolution images are available. Due to the system's movements, objects can moreover be seen at different sizes, which leads to consider feature robustness regarding a wide range of image resolutions. To address these practical issues, our second contribution is to propose a deep structure designed for classifying small images in a fine-grained context, based on the deep network recently designed for ImageNet classification [16, 17, 22]. For this comparative study and the evaluation of our method, we provide very detailed experiments on two major fine-grained oriented datasets, FGVCAircraft [1] and PPMI [2], comprising images of quite high resolutions. To highlight the impact of the resolution parameter, we generate a batch of datasets in which all images have the same resolution.

## 2. IMAGE REPRESENTATIONS

In this section, we first present the feature types introduced above, then we discuss the issue of the image resolution.

**FV features.** Introduced by Perronnin et al. [6], Fisher Vector (FV) is a major Computer Vision representation which has recently proven its efficiency in fine-grained contexts by winning the Fine-Grained classification challenge 2013 [12]. This representation is based on three main steps. First, local features are extracted on all training images. In this work, we consider the well-known SIFT descriptor introduced by [23]. The distribution of these features is then estimated through a Gaussian Mixture Model (GMM). Finally, for each image, local features are encoded with respect to the first and second order statistics of this GMM and aggregated. This resulting aggregated vector is the image representation. In [24], Perronnin et al. proposed two major steps to boost FV performances. Following their work, FV $z$ are power normalized by applying the function $sign(z_i) \times |z_i|^\alpha$ to each element $z_i$, then the whole FV is further L2-normalized.

**CNN features.** A convolutional neural network (CNN) is a succession of convolutional layers followed by fully-connected layers. Each layer takes as input the output values of the previous layer. Convolutional layers can be thought of as a batch of filters applied on the image at different scales, whereas fully-connected layers compute linear combinations of all output values of the previous layer. For our study, we use CNN-M network [17], which contains five convolutional blocks and three fully-connected layers. This network was trained on ILSVRC12 training set, which contains 1.2 million images belonging to 1,000 classes [19]. In our experiments, we use a part of this pre-trained network as a deep feature extractor, similarly to what has been tested on Caltech-UCSD Birds [20] or PASCALVOC [17]. More precisely, we extract deep features at two specific levels of the network: *(a)* after the first fully-connected layer and *(b)* after the last fully-connected layer (both features have size 4096).

**Resolution adaptation for feature computation.** To show the influence of resolution on the performances of these image representations, we focus on seven specific resolutions ranging from $200 \times 200$ to $10 \times 10$ pixels. On this purpose, we generate, for each dataset [1, 2], 7 derived datasets, each of them containing all images of the original dataset downsized to a $s \times s$ pixel image. These datasets are then considered as independent classification problems: for each resolution $s$, classifiers are trained on images of size $s \times s$ and tested on test images of size $s \times s$. Both feature types presented above are however very sensitive to these resolution changes. Due to its fixed structure, CNN-M requires a fixed $224 \times 224$ input image size. Besides, small images do not enable computing enough SIFT, which highly affects FV computation. To address both these issues, small $s \times s$ images are thus magnified using a nearest-neighbour interpolation. This process provides images with a large enough number of pixels without altering the image quality obtained for small resolution.

## 3. LR-CNN

Standard deep networks used for large-scale image datasets such as Krizhevsky-like structures [16] have achieved outstanding performances on fine-grained classification of high resolution images, yet they contain at least 60 million weights; such a complex system cannot be learned properly on a mid-scale dataset containing a few thousand training images. To address this issue, we present LR-CNN, a deep structure adapted to low-resolution image classification in a fine-grained context.

**Architecture.** Based on CNN-M structure, this architecture is a version of structures adapted to fine-grained classification of high resolution images scaled to suit a low resolution context. On this purpose, we design a deep structure computing very rich image representations through three convolutional layers and two fully-connected layers. All convolutional blocks are composed of 64 filters, providing a large number of image descriptors. Filters of the first convolutional layer have size $5 \times 5$, computing convolution at a global scale in the input image. The responses to these filters are then cross-map normalized. A max pooling step summarizes local information in each response map. We introduce overlap in these layers, providing a richer pooling step. The second convolutional layer contains 64 filters of size $5 \times 5$. Max-pooling is performed the same way as in conv1. The last convolutional layer comprises filters of size $3 \times 3$: this filter size appears to be significant since it determines which proportion of the input image is addressed. The number and size of the filters are determined so that the network covers a proportion of the input image which enables distinguishing small discriminating details. These convolutional layers are followed by a fully-connected layer, which computes a linear combination of all outputs of third convolutional layer. A final fully-connected layer computes the score of the input image for each class. Non-linearities are introduced by applying function Rectified Linear Units (ReLU) $f(x) = max(0, x)$ to the responses of filters from all layers. The outputs of this network are finally normalized through a softmax step, computed as follows:

$$y_k = \frac{e^{x_k}}{\sum_{k'} e^{x'_k}} \tag{1}$$

where $x_k$ is the score of filter $k$ from previous layer and $y_k$ is the corresponding output. This structure is summarized in table 1.

**Training.** Although backpropagation is a standard training framework, this algorithm contains many details that need to be controlled. The weights of the different layers in the CNN are learned by iteratively correcting each weight with respect to its contribution to the output loss. More specifically, at each iteration $t$ of this backpropagation algorithm, the weight $w_{ijk}^{(t)}$

| Block | Conv1 | Conv2 | Conv3 | Fc4 | Fc5 |
|---|---|---|---|---|---|
| Details | 64 filters $5 \times 5$, st.1<br>ReLU, contrast norm<br>max. pool [3 3], st. 2 | 64 filters $5 \times 5$, st.1<br>ReLU<br>max. pool [3 3], st. 2 | 64 filters $3 \times 3$, st.1<br>ReLU<br>max. pool [3 3], st. 2 | output size 128<br>ReLU | output size 100<br>softmax |

**Table 1**. LR-CNN structure, designed for small image classification in a fine-grained context.

is updated of

$$\Delta w_{ijk}^{(t)} = m\Delta w_{ijk}^{(t-1)} - \lambda\varepsilon w_{ijk}^{(t-1)} - \varepsilon g_{ijk}^{(t)} \qquad (2)$$

where $m$ is the momentum, $\varepsilon$ is the learning rate, $\lambda$ is the weight decay and $g_{ijk}^{(t)}$ is the gradient of the loss function with respect to the weight $w_{ijk}$ at time $t$. Through equation 2, it can easily be seen that momentum and weight decay regulate some inertia of the weight at previous time, and learning rate regulates how much the weight is updated with respect to the loss gradient. In our case, we use a log-loss cost function.

Some layers (usually the first fully-connected layer) may have an important number of weights. To prevent the system from overfitting, dropout [16] is used; this technique freezes some of the weights at time $t$ with probability $p$, which lightens the network and thus prevents overfitting. Other techniques like Dropconnect [25] have emerged, proposing to drop connections instead of weights, however dropout strategies lead to better results on the studied benchmarks.

Data augmentation further helps preventing overfitting. This consists in multiplying the number of training images by applying to them some transformations which leave the label unchanged. In our case, images are reduced to $37 \times 37$ pixels in which five $32 \times 32$ crops are sampled from the corners and the center; each crop is flipped around the vertical axis.

## 4. EXPERIMENTS

We test FV and CNN features on the very challenging task of fine-grained classification, using two significant datasets. FGVC Aircraft [1] comprises 100 classes of airplane variants, each containing 100 aircraft images. PPMI [2] is a 4800 image dataset of persons with musical instruments. Each image contains a person and an instrument out of a list of 12. The aim is to determine whether the person is playing the instrument or merely holding it. For our experiments, we use the on-line available $258 \times 258$ normalized images.

**Setups.** For FV features, on both datasets, we extract multi-scale dense gray-level SIFT [23]. We use a 64 gaussians GMM and $1 \times 1 + 2 \times 2 + 4 \times 4$ spatial pyramids. The resulting FV feature of an image has a size 344064. Following [24], FV are L2-normalized and power normalized with $\alpha = 0.5$. Fisher Vectors are implemented using VLFeat library [26].

For this study, we use the existing deep network CNN-M from MatConvNet library [27] trained on ImageNet dataset [19]. To show the importance of the choice of the deep feature, we extract features by using the output of different layers of the network: the output of the second fully-connected layer (CNNM19), for this is a very commonly used baseline in deep feature analysis, and the output of the first fully-connected layer (CNNM16). These features are then L2-normalized.

To avoid any misinterpretation due to scaling issues, we choose to use linear classifiers. We use Crammer and Singer optimization on the multi-class FGVC Aircraft classification task, and a standard L2-regularized L1-loss linear SVM on the binary PPMI classification task. Both classifiers are implemented using Liblinear library [28].

**FV vs CNN on varying resolution.** In figures 2 and 3, we present classification performances vs input image size for several resolutions sampled from $200 \times 200$ to $10 \times 10$ pixels.[1] For FGVCAircraft, the performances range between 53.3% and 19.2% for FV, 46.4% to 15.8% for CNNM19, 54.6% and 17.3% for CNNM16. The performances on PPMI vary from 86.0% to 70.9% for FV, from 87.4% to 69.0% for CNNM19, from 89.7% to 70.9% for CNNM16.

We now focus on the impact of resolution on classification performances. It is noticeable that each baseline has the same bimodal behaviour on both datasets regarding resolution diminution. As long as the resolution remains quite high, performances plateau: for FV, the performance loss on resolutions $200 \times 200$ to $100 \times 100$ is 0.7% on FGVC Aircraft and 0.4% on PPMI. Then, for all resolutions lower than these critical values, the methods seem to get much more sensitive to the image resolution: the FV performance gap between resolutions $50 \times 50$ to $20 \times 20$ is 14.2% on FGVC Aircraft and 4.2% on PPMI. MatConvNet deep features also show the same behaviour: on FGVC Aircraft, CNNM19 features performances drop of 3.3% from $200 \times 200$ to $100 \times 100$ and of 20.2% from $100 \times 100$ to $20 \times 20$. To illustrate this behaviour, we show in figure 4 images of two Airbus variants at different resolutions. At resolution $200 \times 200$, the variant (A380) can easily be read on the fine and the fuselage of the first airplane. One can even see that the A380 has two floors, while the A330-200 only has one. When decreasing image resolution, at $50 \times 50$ pixels, the variant is no longer readable. However, it can still be guessed that the A380 has two floors. Finally, critical resolution ($20 \times 20$ pixels) makes it impossible to distinguish either of these discriminating details.

We also show that FV reaches the best performances on low-resolution images, although relative performances of FV and CNN features strongly depend on the CNN layer

[1] Performances can be improved using external data and additional information, but these processes are out of the scope of this paper.

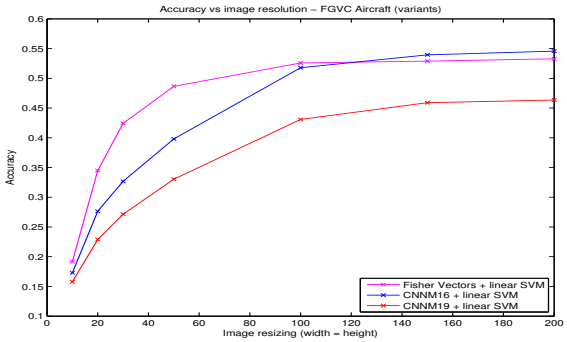



**Fig. 2**. Performances using FV and CNN-M features vs image resolution on FGVC Aircraft

**Fig. 4**. Example of two Airbus airplane images (top A380, bottom A330-200) at resolutions $200 \times 200$ (left), $50 \times 50$ (middle) and $20 \times 20$ (right). Resolution degradation occludes the discriminating details that enable distinguishing classes.

we choose as the image representation. Indeed, for features extracted on top of the fully-connected part (CNNM16), performances are competitive with those of FV at several resolutions, or even better (e.g. on PPMI for resolutions $\geq 100 \times 100$). However, when extracting features at a deepest level of the network (such as CNNM19), classification performances drop. This behaviour, observed on both datasets, tends to support the conclusion that the last layers of a network are more specific to the learning dataset, whereas the shallowest layers offer a more generic description of an input image, as shown in [29]. It could however be interesting to adapt the pre-trained weights of CNNM by fine-tuning the network on the targeted tasks.

**LR-CNN results.** For initialization, the weights are drawn from a gaussian distribution of mean 0 and standard deviation 0.01, and a 50% dropout is performed after the first fully-connected layer. The weight decay is set at 0.01 and momentum value is 0.9. The learning rate is set at $10^{-2}$, then gradually decreased down to $10^{-5}$ when the training error plateaus. To fairly compare all features quality, we use this network as a feature extractor: the output of conv3 is the image feature (size 2304), and a linear SVM is used for classification.

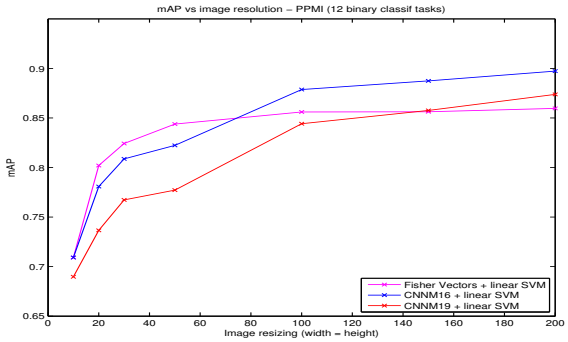



**Fig. 3**. Performances using FV and CNN-M features vs image resolution on PPMI

In this work, we propose an architecture optimized for low resolution image classification in a fine-grained context.[2] This model obtains a 44.8% accuracy on FGVCAircraft, outperforming FV (42.4%) as well as pre-trained deep features CNNM16 (32.7%) and CNNM19 (27.2%). We show that LR-CNN provides features capturing small details, thus bringing a significant gain of performance over pre-trained networks on fine-grained classification of low-resolution images. Our network benefits from learning "best practices" such as dropout [16] as well as an advanced parametrization. Filter size appears to be an important parameter, since performances drop of 1.1% when upsizing filters of conv3 to $5 \times 5$. Having two fully-connected layers is another significant improvement: accuracy drops of 4.3% when no hidden fully-connected layer exists. LR-CNN obtains thus better results than FV while providing a much smaller representation: while FV requires over 300k-dimensional features to describe a 1k pixel image, LR-CNN only requires 2k dimensions. Adapted from the deep structures designed for high resolution images, this network has the capacity to compute rich and compact image representations.

## 5. CONCLUSION

We evaluate two very competitive features (Fisher-based and CNN-based) on a fine-grained classification task with respect to the dataset resolution. We highlight a bimodal behaviour: for mid- to high-resolutions ($\geq 100 \times 100$ pixels) performances remain quite stable, then drop when reaching a critical resolution ($\leq 50 \times 50$ pixels). We also show the predominance of FV over CNN features on low resolution image datasets in a fine-grained context. Finally, we introduce LR-CNN, a CNN structure optimized for classification of fine-grained complex low-resolution images, which outperforms pretrained CNN and FV features while proposing much more compact features.

[2]It should be noted that due to the fixed structure of CNNs, a particular strategy is required to test LR-CNN on other resolutions, and the resulting performances would not be comparable.

## 6. REFERENCES


[1] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," Tech. Rep., 2013.

[2] B. Yao and L. Fei-Fei, "Grouplet: A structured image representation for recognizing human and object interactions," in *CVPR*, 2010.

[3] M. Cord and P. Cunningham, *Machine learning techniques for multimedia*, Springer, 2008.

[4] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," Tech. Rep. CNS-TR-2010-001, California Institute of Technology, 2010.

[5] M-E. Nilsback and A. Zisserman, "Automated flower classification over a large number of classes," in *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, 2008.

[6] F. Perronnin and C. Dance, "Fisher kernels on visual vocabularies for image categorization," *CVPR*, 2007.

[7] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *ICCV*, 2003.

[8] C. Theriault, N. Thome, and M. Cord, "Extended coding and pooling in the hmax model," *IEEE Transactions on Image Processing*, 2013.

[9] C. Theriault, N. Thome, and M. Cord, "Dynamic scene classification: Learning motion descriptors with slow features analysis," *CVPR*, 2013.

[10] S. Avila, N. Thome, M. Cord, E. Valle, and A. de A. Araújo, "Bossa: extended bow formalism for image classification," *ICIP*, 2011.

[11] S. Avila, N. Thome, M. Cord, E. Valle, and A. de A. Araújo, "Pooling in image representation: the visual codeword point of view," *Computer Vision and Image Understanding (CVIU)*, vol. 117, pp. 453–465, 2013.

[12] "Fine-grained challenge 2013," https://sites.google.com/site/fgcomp2013/.

[13] G. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief networks," *Neural Computation*, 2006.

[14] H. Goh, N. Thome, M. Cord, and J-H. Lim, "Unsupervised and supervised visual codes with restricted boltzmann machines," *ECCV*, 2012.

[15] H. Goh, N. Thome, M. Cord, and J.H. Lim, "Top-down regularization of deep belief networks," *NIPS*, 2013.

[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *NIPS*, 2012.

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

[18] M. Oquab, L. Bottou, I. Laptev, and J. Sivic, "Learning and transferring mid-level image representations using convolutional neural networks," in *CVPR*, 2014.

[19] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," *CVPR*, 2009.

[20] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *CVPR Workshops 2014*, 2014.

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

[22] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," *ICLR*, 2014.

[23] D. Lowe, "Object recognition from local scale-invariant features," *ICCV*, 1999.

[24] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *ECCV*, 2010.

[25] L. Wan, M. Zeiler, S. Zhang, Y. Le Cun, and R. Fergus, "Regularization of neural networks using dropconnect," *ICML*, 2013.

[26] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008.

[27] A. Vedaldi and K. Lenc, "Matconvnet – convolutional neural networks for matlab," *CoRR*, vol. abs/1412.4564, 2014.

[28] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *JMLR*, 2008.

[29] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?," *NIPS*, 2014.