Classemes and Other Classifier-based Features for Efficient Object Categorization - *Supplementary material* -

Alessandro Bergamo, and Lorenzo Torresani, *Member, IEEE*

1 Low-level features

We extract the *SIFT* [1] features for our descriptor according to the following pipeline. We first convert each image to gray-scale, then we normalize the contrast by forcing the 0.01% of lightest and darkest pixels to be mapped to white and black respectively, and linearly rescaling the values in between. All images exceeding 786,432 pixels of resolution are downsized to this maximum value while keeping the aspect ratio. The 128-dimensional SIFT descriptors are computed from the interest points returned by a DoG detector [2]. We finally compute a Bag-Of-Word histogram of these descriptors, using a K-means vocabulary of 500 words.

2 Classemes

The LSCOM categories were developed specifically for multimedia annotation and retrieval, and have been used in the TRECVID video retrieval series. We took the LSCOM CYC ontology dated 2006-06-30, which contains 2832 unique categories. We removed 97 categories denoting abstract groups of other categories (marked in angle brackets in [3]), and then removed plural categories that also occurred as singulars, and some people-related categories which were effectively near-duplicates, and arrived at $C = 2659$ categories. For a general application these examples would not need to be manually filtered in any way, but in order to perform fair comparisons against the Caltech image database, near duplicates of images in that database were removed by a human-supervised process.

3 PiCoDes

3.1 Choice of the low-level feature dimensionality

As mentioned in the paper, for practical purposes, for the PiCoDes training we reduce the dimensionality of the image descriptor Psi using PCA. The dimensionality of 6415 was chosen based on a preliminary experiment of multiclass classification on Caltech 256, which is shown in figure 1. We used a linear SVM trained on image descriptors $\Psi(\boldsymbol{x})$ of varying dimensionality. This study showed that the vector of 6415 dimensions causes only a small drop in accuracy ($\sim 1\%$) compared to the full 68K-dimensional feature vector.

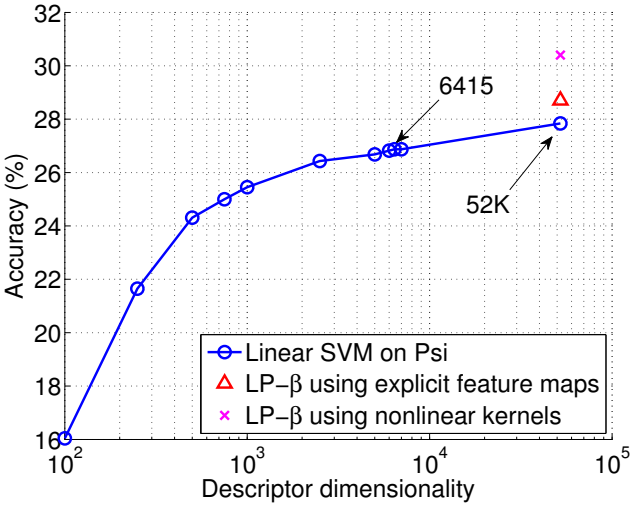

Fig. 1. The accuracy versus compactness trade off. The benchmark is Caltech256, using 10 examples per class. The pink cross shows the multiclass categorization accuracy achieved by an LP-β classifier using exact kernel distances; the red triangle is the accuracy of an LP-β classifier that uses "lifted-up" features to approximate kernel distances; the blue line shows accuracy of a linear SVM trained on PCA projections of the lifted-up features, as a function of the number of PCA dimensions.

3.2 Derivation of eq. 7 in the paper

We present below the derivation of eq. 7 of the paper. First, we rewrite our objective function, i.e., eq. 6, in

A. Bergamo and L. Torresani are with the Department of Computer Science, Dartmouth College, Hanover, NH 03755, U.S.A.
E-mail: {aleb, lorenzo}@cs.dartmouth.edu

expanded form:

$$E(\boldsymbol{A}, \boldsymbol{w}_{1..K}, b_{1..K}) = \sum_{k=1}^{K} \left\{ \frac{1}{2}\|\boldsymbol{w}_k\|^2 + \frac{\lambda}{N} \sum_{i=1}^{N} \ell\left[y_{ik}(b_k + \sum_{c=1}^{C} w_{kc} \mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0]) \right] \right\}$$

Fixing the parameters $\boldsymbol{w}_{1..K}, \boldsymbol{b}, \boldsymbol{a}_1, \ldots, \boldsymbol{a}_{c-1}, \boldsymbol{a}_{c+1}, \ldots, \boldsymbol{a}_C$ and minimizing the function above with respect to $\boldsymbol{a}_c$, is equivalent to minimizing the following objective:

$$E'(\boldsymbol{a}_c) = \sum_{k=1}^{K} \sum_{i=1}^{N} \ell\left[y_{ik} w_{kc} \mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0] + y_{ik} b_k + \sum_{c' \neq c} y_{ik} w_{kc'} \mathbf{1}[\boldsymbol{a}_{c'}^T \boldsymbol{x}_i > 0] \right].$$

Let us define $\alpha_{ikc} \equiv y_{ik} w_{kc}$, and $\beta_{ikc} \equiv (y_{ik} b_k + \sum_{c' \neq c} y_{ik} w_{kc'} \mathbf{1}[\boldsymbol{a}_{c'}^T \boldsymbol{x}_i > 0])$. Then, we can rewrite the objective as follows:

$$\begin{aligned} E'(\boldsymbol{a}_c) &= \sum_{k=1}^{K} \sum_{i=1}^{N} \ell\left[\alpha_{ikc} \mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0] + \beta_{ikc} \right] \\ &= \sum_{i=1}^{N} \Bigg[\mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0] \sum_{k=1}^{K} \ell(\alpha_{ikc} + \beta_{ikc}) + \\ &\quad (1 - \mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0]) \sum_{k=1}^{K} \ell(\beta_{ikc}) \Bigg] \\ &= \sum_{i=1}^{N} \Bigg[\mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x}_i > 0] \\ &\quad \sum_{k=1}^{K} \ell(\alpha_{ikc} + \beta_{ikc}) - \ell(\beta_{ikc}) \Bigg] + const \ . \end{aligned}$$

Finally, it can be seen that optimizing this objective is equivalent to minimizing

$$E(\boldsymbol{a}_c) = \sum_{i=1}^{N} v_i \mathbf{1}[z_i \boldsymbol{a}_c^T \boldsymbol{x}_i > 0]$$

where $v_i = \left| \sum_{k=1}^{K} \ell(\alpha_{ikc} + \beta_{ikc}) - \ell(\beta_{ikc}) \right|$ and $z_i = \text{sign}\left(\sum_{k=1}^{K} \ell(\alpha_{ikc} + \beta_{ikc}) - \ell(\beta_{ikc}) \right)$. This yields eq. 7.

3.3 Optimization strategy for eq. 7

We optimize equation 7 using the block-coordinate optimization procedure described in section 3.3.2 of the paper. Note that we have also experimented with several other optimization methods, including stochastic gradient descent applied to a modified version of our objective, where we replaced the binarization function $h(\boldsymbol{x}; \boldsymbol{a}_c) = \mathbf{1}[\boldsymbol{a}_c^T \boldsymbol{x} > 0]$ with the sigmoid function $\sigma(\boldsymbol{x}; \boldsymbol{a}_c) = 1/(1 + \exp(-\frac{2}{T} \boldsymbol{a}_c^T \boldsymbol{x}))$ to relax the problem. This type of minimization is similar to that traditionally used in neural networks, with the difference that here we are optimizing a large-margin multiclass objective with respect to our PICODES representation. However, the issue with this minimization strategy is that it no longer optimizes for pure *binary* features, which is the way we intend to use them at application time: after learning, we want to replace back $\sigma(\boldsymbol{x}; \boldsymbol{a}_c)$ with $h(\boldsymbol{x}; \boldsymbol{a}_c)$ to obtain binary descriptors. In practice, we found that for this reason codes optimized in this fashion performed much worse than those directly learned via the coordinate descent procedure described in section 3.3.2 of the paper.

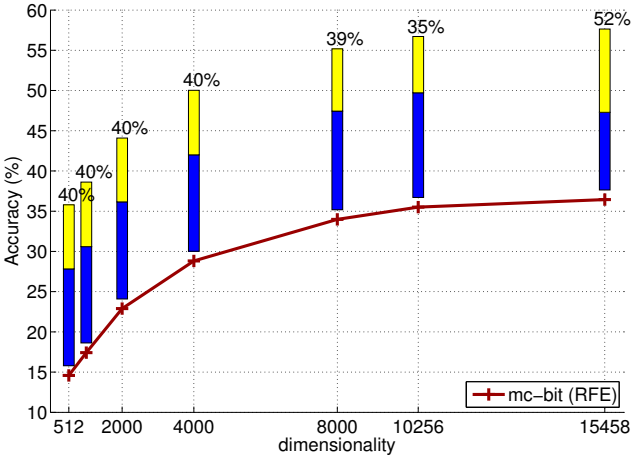

Fig. 2. Multiclass recognition accuracy as a function of MC-BIT dimensionality on ILSVRC2010. We use Recursive Feature Elimination [4] to reduce the dimensionality of our MC-BIT descriptor. The percentage at each dimensionality indicates the proportion of classeme features retained in the descriptor. Although initially the full descriptor contains more classemes than meta-classes, the majority of features selected at each step are meta-classes.

4 META-CLASSES

Figure 2 shows the accuracy obtained by performing feature selection on the descriptor MC-BIT. The figure shows also that the feature selection chooses more features corresponding to abstract meta-classes (inner nodes) than real object classes (leaves) of the learned tree.

Figure 3 shows the results achieved with the individual subcomponents of MC-BIT: "MC-BIT-TREE" and "MC-BIT-1VSALL", showing that the grouping of classes performed by the label tree produces features that lead to better generalization on novel classes.

Figure 4 shows a comparison between two label trees: our proposed tree used for the MC descriptor, and the ImageNet semantic tree, which was created according to the WordNet hierarchy.

Figure 5 provides an experimental justification for the number of random projections used to generate the MC-LSH descriptor. The plot shows that 200K dimensions are sufficient to maintain the discriminative power of the original descriptor (the compression

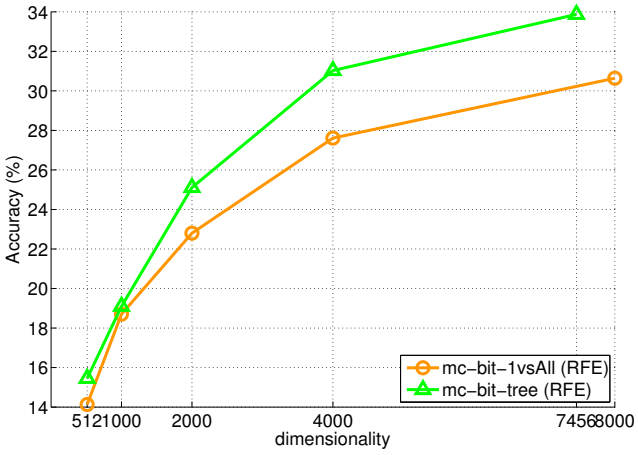

Fig. 3. Multiclass recognition accuracy on ILSVRC2010 as a function of the dimensionality of the image descriptor. We use Recursive Feature Elimination to reduce the dimensionality of all descriptors. The two curves correspond to the results achieved with the individual subcomponents of MC-BIT: "MC-BIT-TREE", which consists of the 7232 meta-class classifiers learned for the inner nodes of the label tree, and "MC-BIT-1VSALL", which contains the outputs of the 8000 one-vs-the-rest classeme classifiers (the leaves of the tree). Note that the accuracy obtained using only the inner node features is clearly superior to the one generated by only the classeme features.

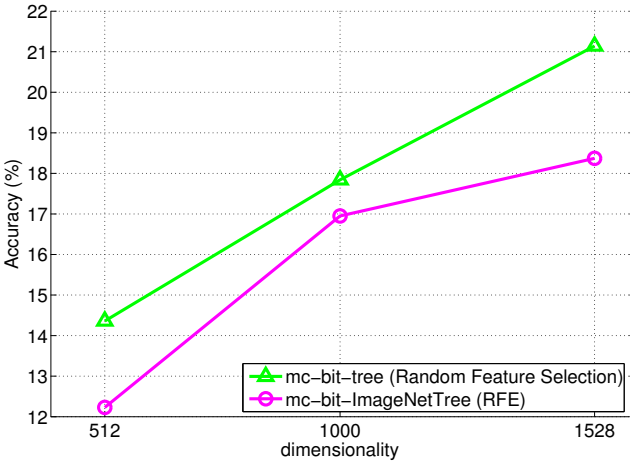

Fig. 4. Comparison between label trees on ILSVRC2010. The curve for MC-BIT-TREE shows the accuracy achieved with the inner-node features of MC-BIT (the 7232 abstract meta-classes in our learned tree). The MC-BIT-IMAGENETTREE curve corresponds to the inner nodes of the ImageNet semantic tree (we use a pruned version of this tree, containing only the 8,000 synsets that we used to train our MC descriptor). We used Random Feature Selection for the MC-BIT-TREE descriptor, and Recursive Feature Elimination for MC-BIT-IMAGENETTREE. Note that the accuracy obtained with our learned label tree is clearly superior to the one generated using the semantic ImageNet tree (despite we use for this last one a more powerful feature selection algorithm, Recursive Feature Elimination instead of Random Feature Selection).

causes a negligible drop in accuracy, $< 1\%$ according to our evaluation using the original real-valued MC descriptors on the same ILSVRC 2010 benchmark). The advantage of using LSH is that the storage is reduced from 60 KB to 25KB per image (see Table 1). We apply LSH as follows: we first reduce the dimensionality of the data via PCA, producing a 9000-dimensional descriptor. We center the vector by subtracting the mean (which has been calculated using a validation set), and we apply 200,000 random projections drawn from a Gaussian distribution with 0 mean and unit variance.

5 NOTE ABOUT TABLE 2 OF THE PAPER

For the estimation of the recognition time (Table 2 of the paper), we used a computer with a CPU Intel Xeon X5675@3.07GHz and 20 GB of memory. All methods are implemented in C; for real-valued descriptors SSE instructions were used to speed-up the evaluation of the model; binary descriptors are stored in bitmap form and a look-up table is used to decompress the data to be evaluated with the model; the decompression of the data stored with Product Quantization is implemented as a look-up table as well.

6 OBJECT-RECOGNITION ON CALTECH 256

In this section we present additional experiments on Caltech 256 for the task of object categorization. Figure 6 shows the accuracy obtained with our descriptors and many other methods found in the literature. Note that our descriptors outperform many of them, while being much more compact.

Figure 7 studies the accuracy of our descriptors and other binary codes as a function of the number of bits. We use 10 training examples per class and 25 for testing. The classification model is a 1-vs-all linear SVM for all methods, with the exception of LP-β. Note that PICODES of 2048 bits match the accuracy of the state-of-the-art LP$-\beta$ classifier while enabling orders of magnitude faster training and testing.

7 OBJECT-CLASS RECOGNITION ON ILSVRC2010

Figure 8 shows the trade-off between accuracy and storage for different image signatures. Note that our descriptors are the most compact ones while producing near state-of-the-art classification accuracy.

Figure 9 shows the storage-speed envelope of modern image descriptors. The plot shows how our descriptors are among the most compact and fast ones. Note that the times reported here do not include the feature extraction time, since in our motivating

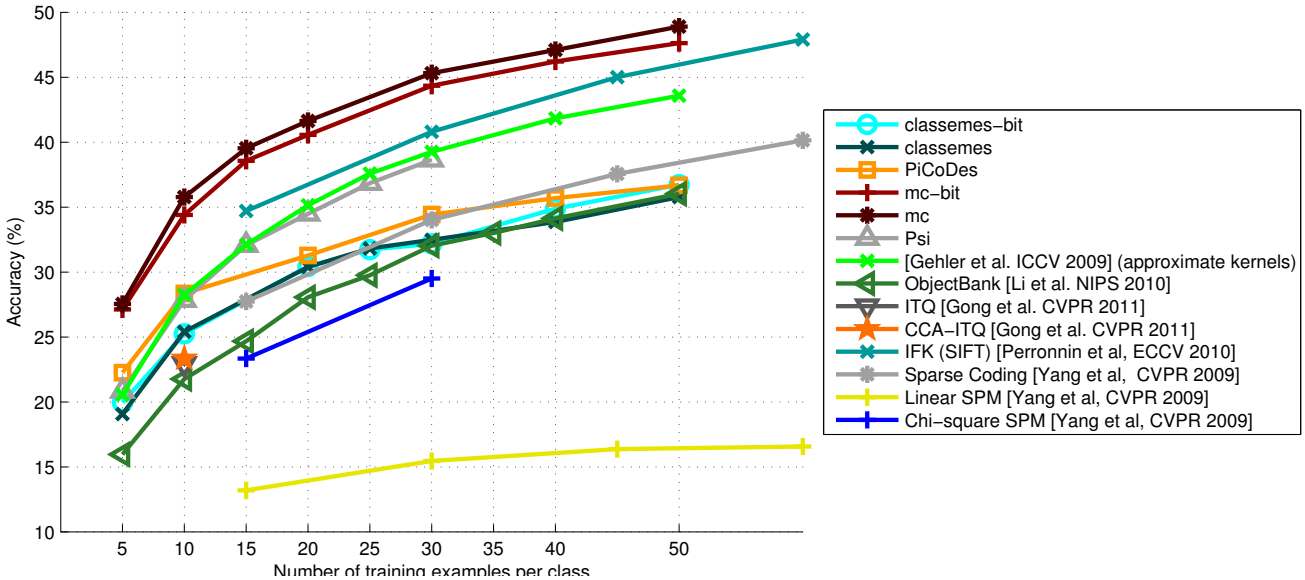

Fig. 6. Multi-class recognition on Caltech 256 using different image representations. The classification model is a linear SVM (except for LP-β). The accuracy is plotted as a function of the training set size.

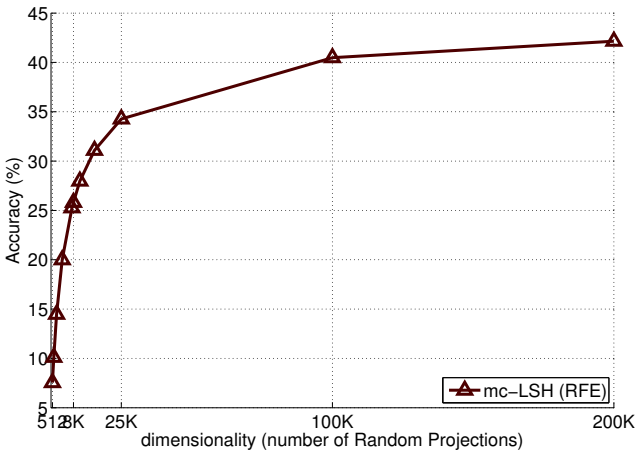

Fig. 5. Multiclass recognition accuracy on ILSVRC2010 as a function of the number of random projections used to generate the MC-LSH descriptor. We use Recursive Feature Elimination to reduce the dimensionality of the descriptor. The plot shows that 200K dimensions provide a significant compression of the original descriptor, at a negligible drop in accuracy (note that the axes use the log scale). See the text for more details.

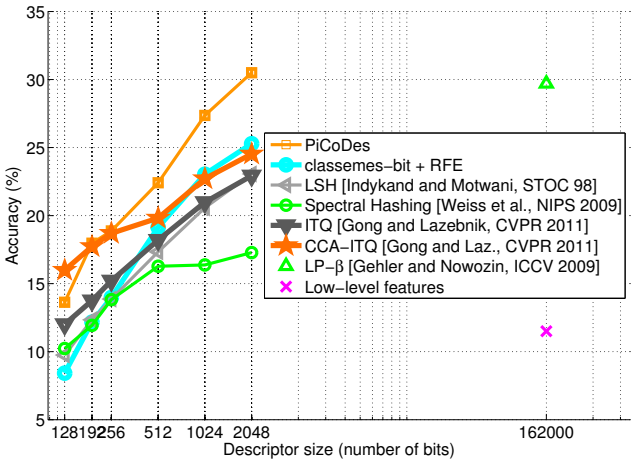

Fig. 7. Multiclass categorization accuracy on Caltech256 using different binary codes, as a function of the number of bits. PICODES outperform all the other compact codes. PICODES of 2048 bits match the accuracy of the state-of-the-art LP-β classifier.

application of object-class search feature extraction is performed during an offline stage, when the search index is created. The creation of the index is typically done incrementally and does not have to obey strict time requirements. However, we also point out that the time needed to extract our meta-class descriptor is actually in the same order of magnitude as those of other commonly used features for categorization, including Fisher Vectors [5]. For example, according to our experiments on a few full-scale images taken from PASCAL VOC 2007, the extraction of Fisher Vectors takes roughly 1 second per image if sparse SIFT features are used, but it is about 8 seconds if dense SIFT (single scale at every pixel) are used (as in Perronnin, CVPR 2012 [5]). By comparison, extraction of our complete mc descriptor (i.e., including computation of the low-level features used by mc) takes on average about 5 seconds. All timing experiments were done on the same machine using a single core.

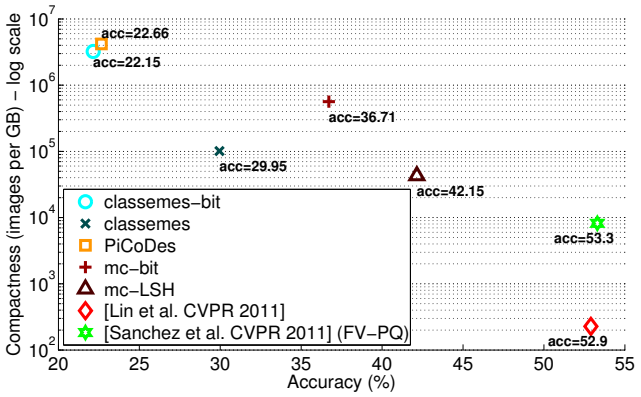

Fig. 8. Accuracy versus compactness of different image descriptors. On both axes, higher is better (note the logarithmic axes). The number next to each point indicates the multi-class accuracy obtained on the benchmark ILSVRC 2010.

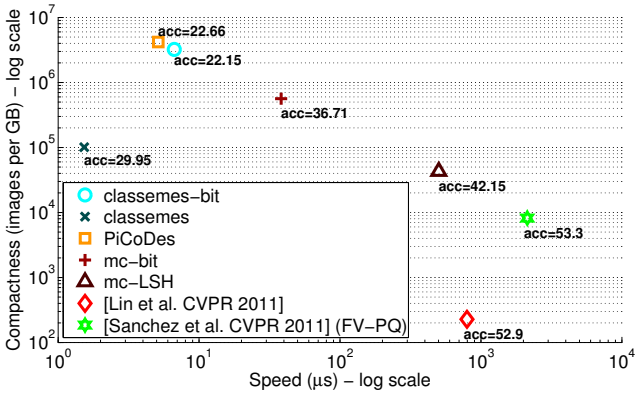

Fig. 9. Compactness versus recognition time for different image descriptors. Note that the recognition time does not include the feature extraction time (see text). The number next to each point indicates the multi-class accuracy obtained on the benchmark ILSVRC 2010.

REFERENCES

[1] D. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCV*, vol. 60, no. 2, pp. 91–110, 2004.
[2] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," http://www.vlfeat.org/, 2008.
[3] "Lscom: Cyc ontology dated (2006-06-30)," http://lastlaugh.inf.cs.cmu.edu/lscom/ontology/LSCOM-20060630.txt, http://www.lscom.org/ontology/index.html.
[4] O. Chapelle and S. S. Keerthi, "Multi-class feature selection with support vector machines," *Proc. Am. Stat. Ass.*, 2008.
[5] J. Sánchez and F. Perronnin, "High-dimensional signature compression for large-scale image classification," in *CVPR*, 2011, pp. 1665–1672.