Modeling Image Patches with a Generic Dictionary of Mini-Epitomes – CVPR 2014 paper supplementary material –

George Papandreou
TTI Chicago
gpapan@ttic.edu

Liang-Chieh Chen
UC Los Angeles
lcchen@cs.ucla.edu

Alan L. Yuille
UC Los Angeles
yuille@stat.ucla.edu

Abstract

The supplementary material contains: (1) Image classification experimental results on the Caltech-101 dataset. (2) Analysis and experimental evaluation of the efficient DCT-based algorithm for epitomic patch matching with approximate nearest neighbor (ANN) techniques. (3) A discussion about the whitening effect of gradient-domain processing and its impact on classification performance. (4) More details about our epitomic footprint encoding.

Epitome	Position	Dictionary Size K	
/Patch	Encod.	512	1024
16/8	1x1	71.67	72.25
	2x2	72.58	71.95
12/8	1x1	72.09	72.88
	2x2	**72.93**	71.91
10/8	1x1	72.07	**72.95**
8/8	1x1	71.41	71.57

Table 1. Image classification results (average accuracy) of our epitomic dictionary-based method on the Caltech-101 dataset.

1. Image Classification Experiments on Caltech-101

Here we report additional image classification experiments with the proposed epitomic dictionary method on the Caltech-101 dataset. The PASCAL VOC-2007 dataset is richer and more challenging and allows drawing safer conclusions when comparing alternative classification methods. However, performing experiments on the Caltech-101 allows us to directly compare the proposed method with other published techniques which only report results on the Caltech-101 dataset.

We used exactly the same encoding method as in the case of the PASCAL VOS-2007 benchmark described in Sections 4.1 and 4.2 of the main paper, i.e., histograms with hard assignments encoding the label, sign, and, optionally, the position of the best match of each patch in the epitome. In all cases we employ standard SPM histograms [7] into 4×4, 2×2, 1×1 bins. We use the shorthand "*Epitome-1024-16/8-Pos-2x2*" for a dictionary with $K = 1024$ 16×16 mini-epitomes containing 8×8 patches and match position encoding in the epitome into 2×2=4 bins. To emphasize that the proposed method learns generic dictionaries, we use the same epitomic dictionaries learned on the PASCAL VOC-2007 training set for our Caltech-101 experiments.

We follow the standard established protocol for our Caltech-101 evaluation. Specifically, we train 102 1-vs-all SVM classifiers (using χ^2 kernels approximated by explicit feature maps [13]) for classifying each image into one out of the 102 candidate classes (101 objects plus background). We draw four random splits of the dataset into 30 train images and up to 30 test images. We use the first validation split to set the SVM regularization parameter. We measure the average classification accuracy across the 102 classes and report its mean. The deviation (difference between max and min performance) over the three test splits has been up to 1% in all our experiments.

Our results are shown in Table 1. Since each SVM classifier is trained with only 30 positive instances and the feature length is larger due to the finer SPM sampling (21 bins in the Caltech-101 vs. 8 bins in the VOC-2007 case), we only tried dictionaries with $K = 512$ and 1024 mini-epitomes. Our best result is 72.95% average accuracy.

For comparison, we show in Table 2 results reported by other authors on the same task. Our results are better than previously published techniques in the literature of feature learning methods (first column of Table 2), including some multi-layer systems trained by unsupervised [14] or supervised [6] criteria. SIFT coupled with VQ encoding (second column of Table 2) does somewhat better when big dictionaries are used. State-of-the-art performance is achieved by the very recent HMP method of [2] (third column of Table 2), which learns features in a multilevel fashion. We believe that embedding epitomic patch models in similar deep architectures is a very promising avenue which we plan to follow in future work.

Method	Acc	Method	Acc	Method	Acc
SIFT-ST [5]	67.70	SIFT-VQ-600	72.65	SIFT-FV-256	77.78
AdDeconvNets [14]	71.00	SIFT-VQ-2K	73.93	HMP [1]	76.80
Jarrett et al. [6]	65.50	SIFT-VQ-4K	74.41	M-HMP [2]	82.50

Table 2. Image classification results (average accuracy) of recently published methods on the Caltech-101 dataset. The SIFT results refer to the modern implementation of [3], not the original SPM implementation of [7] which reported performance 64.6% for SIFT-VQ-200.

2. Efficient Search with Approximate Nearest Neighbor Techniques

We describe here in more detail the implementation of the approximate nearest neighbor (ANN) method for approximate epitomic patch matching and compare its performance with the exact matching algorithm. Epitomic patch matching can be cast as nearest neighbor search between each $h \times w$ image patch and all $h \times w$ patches contained in all K $H \times W$ mini-epitomes in the dictionary. The patches need to be first whitened by computing their gradient and then contrast normalized, i.e., $\hat{\mathbf{x}}_i = \frac{\mathbf{D}\mathbf{x}_i}{\sqrt{\|\mathbf{D}\mathbf{x}_i\|_2^2+\lambda}}$ and $\hat{\boldsymbol{\nu}}_j = \frac{\mathbf{D}\boldsymbol{\nu}_j}{\sqrt{\|\mathbf{D}\boldsymbol{\nu}_j\|_2^2+\lambda}}$ for the image and epitome patches, respectively. Here $\mathbf{D}$ is the $(h\cdot(w-1)+(h-1)\cdot w)\times(h\cdot w)$ 2-D gradient matrix, i.e., $\mathbf{D}\mathbf{x}_i$ is the vector of $x-$ and $y-$ derivatives of the $h \times w$ patch $\mathbf{x}_i$.

Instead of the Euclidean distance $\|\hat{\mathbf{x}}_i - \hat{\boldsymbol{\nu}}_j\|$ between the normalized patch gradient vectors which have length $(h\cdot(w-1)+(h-1)\cdot w)$, we can equivalently use the Euclidean distance between the whitened and normalized 2-D discrete cosine transforms (DCT) of the patches, $\|\hat{\mathbf{x}}_i^* - \hat{\boldsymbol{\nu}}_j^*\|$, which only have length $h \cdot w$. Here $\hat{\mathbf{x}}_i^* = \frac{\boldsymbol{\Lambda}\mathbf{x}_i^*}{\sqrt{\|\boldsymbol{\Lambda}\mathbf{x}_i^*\|_2^2+\lambda}}$, where $\mathbf{x}_i^* \triangleq \mathbf{C}\mathbf{x}_i$ is the vector of 2-D DCT coefficients and $\mathbf{C}$ is the 2-D DCT transform operator. This holds because the 2-D DCT transform diagonalizes the gradient operator, i.e., $\mathbf{D}^T\mathbf{D} = \mathbf{C}^T\boldsymbol{\Lambda}^2\mathbf{C}$, with $\boldsymbol{\Lambda}$ a diagonal matrix with non-negative entries, and therefore $\|\mathbf{D}\mathbf{x}_i - \mathbf{D}\boldsymbol{\nu}_j\| = \|\boldsymbol{\Lambda}\mathbf{x}_i^* - \boldsymbol{\Lambda}\boldsymbol{\nu}_j^*\|$ [11].

We visualize in Figure 1 the entries of the diagonal weighting matrix $\boldsymbol{\Lambda}$ corresponding to 2-D DCT implementation of gradient-domain processing of 16×16 patches. We note that weighting with $\boldsymbol{\Lambda}$ enhances the higher frequencies in the DCT spectrum.

The resulting epitomic matching algorithm consists of the following steps: (1) Extract patches $\mathbf{x}_i$ from the image. (2) Compute the 2-D DCT transform of the patches $\mathbf{x}_i^* = \mathbf{C}\mathbf{x}_i$. (3) Whiten by point-wise multiplication of the 2-D DCT transform coefficients with the emphasis mask $\boldsymbol{\Lambda}$, followed by contrast normalization, $\hat{\mathbf{x}}_i^* = \frac{\boldsymbol{\Lambda}\mathbf{x}_i^*}{\sqrt{\|\boldsymbol{\Lambda}\mathbf{x}_i^*\|_2^2+\lambda}}$. (4) Search for the nearest neighbor in the database of similarly pre-processed epitome dictionary patches $\hat{\boldsymbol{\nu}}_j^*$. One

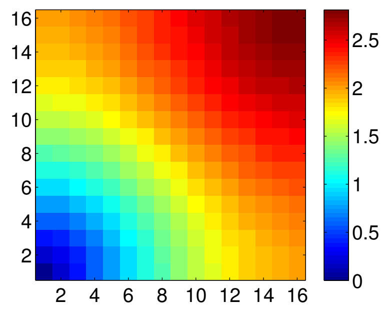


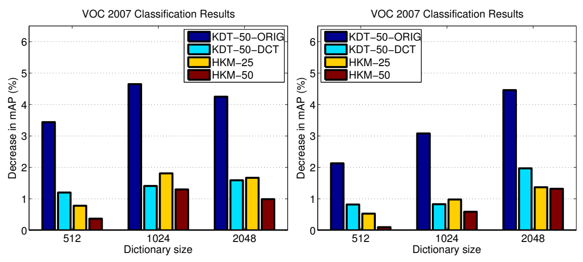
Figure 1. The entries of the diagonal weighting matrix $\boldsymbol{\Lambda}$ corresponding to 2-D DCT implementation of gradient-domain processing of 16×16 patches.

can readily plug-in the proposed algorithm in an existing SIFT-based image classification system by replacing SIFT computation with the pre-processing steps (1)-(3).

Our simple Matlab-based implementation of the above pre-processing steps (1)-(3) has runtime 0.1 sec for a typical PASCAL VOC-2007 image, which can be significantly reduced by employing widely available optimized routines for DCT computation. Interestingly, many image and video coding standards use the DCT for image or video frame compression. This presents the opportunity to deploy our methods directly in the compressed domain [10], which could be appealing for resource-limited devices.

We have experimented with ANN algorithms based on both kd-tree and hierarchical K-means data structures, as implemented in the VLFeat [12] and FLANN [8] libraries, respectively. These algorithms allow finding approximate nearest neighbors with a number of comparisons that grows sub-linearly with the dictionary size. A key property of kd-trees is that they split the search space with hyperplanes perpendicular to the coordinate system axes, making them very fast. However, this also implies that the coordinate system choice matters. In our case, we have found that epitomic kd-tree search is much more efficient in the DCT domain compared to the original gradient domain. Intuitively, the low-pass part of the DCT spectrum allows much faster rejection of putative matches between patches which grossly differ. The hierarchical K-means algorithm is invariant to isometries (linear maps that preserve distance, as is the case with the proposed DCT-based transformation) and hence gives exactly the same results in both the original gradient and the DCT domains. However, the DCT representation is preferable here as well due to its smaller dimensionality, which reduces the search runtime.

In Figure 2 we summarize experiments measuring the performance loss (mAP) in PASCAL VOC-2007 image classification experiments due to approximate nearest neighbor search compared to exact epitomic search. We have evaluated kd-tree search allowing at most 50 comparisons per search query (KDT-50) and hierarchical K-means



(a) (b)

Figure 2. Performance loss in PASCAL VOC-2007 image classification experiments due to approximate nearest neighbor search compared to exact epitomic search (decrease in mAP, lower is better). (a) *Epitome-K-16/8-Pos-2x2* dictionary with 16×16 epitomes. (b) *Epitome-K-12/8-Pos-2x2* dictionary with 12×12 epitomes.

search allowing at most either 25 or 50 comparisons per search query (HKM-25, HKM-50). For kd-trees we have experimented with both the original gradient and DCT representations. We see that using DCT is crucial for kd-tree ANN search to be effective. Both HKM and the DCT-based algorithm can be used with small (about 1%) performance loss over exact search with moderate search times comparable to those of SIFT-based VQ encoding algorithms.

3. Whitening by Gradient-Domain Processing and its Effect in Image Classification

Many authors have emphasized the importance of whitening image patches, which involves (a) removing second-order correlations and (b) enhancing the high-frequency edge contrast. See, e.g., the discussion in [4] for a setting similar to ours.

In all our experiments we perform whitening by gradient-domain processing, i.e., we choose in Equation (1) of the main paper the $d \times d$ covariance matrix $\mathbf{\Sigma}_0^{-1} = \mathbf{D}^T\mathbf{D} + \epsilon\mathbf{I}$, where $\mathbf{D}$ is the gradient operator computing the $x-$ and $y-$ derivatives of the $h \times w$ patch and ϵ is a small constant.

The analysis in the previous Section allows us to better understand the whitening effect of gradient-domain processing: It corresponds to approximately decorrelating the patch $\mathbf{x}_i$ via the 2-D discrete cosine transform (DCT) $\mathbf{x}_i^* = \mathbf{C}\mathbf{x}_i$, followed by high-frequency enhancement by element-wise multiplication of DCT coefficients with the mask $\mathbf{\Lambda}$ shown in Figure 1.

Similar to [4], we have found that gradient-domain processing performs significantly better in terms of classification accuracy than image-domain processing, i.e., using $\mathbf{\Sigma}_0^{-1} = \mathbf{I}$ and finding matches in the dictionary by minimizing the image-domain reconstruction error $R^2(\mathbf{x}_i; k, p) = \frac{1}{c_i^2}\left(\|\mathbf{x}_i - \alpha_i \mathbf{T}_p \boldsymbol{\mu}_k\|^2 + \lambda(|\alpha_i| - 1)^2\right)$ in place of Equation (2) in the main paper. For example, using the *Epitome-1024-16/8-Pos-2x2* and performing epitomic search in the image domain yields classification performance 49.50% mAP on the PASCAL VOC-2007 benchmark, which is significantly lower than the 56.12% mAP we obtain with our proposed gradient-domain processing.

4. Epitomic footprint encoding

We have performed preliminary experiments on an epitomic footprint encoding, which is inspired by the Fisher Vector encoding in [9]. The main idea is to encode the difference $\Delta\boldsymbol{\mu}_k$ between an image-specific and the generic epitome, which captures how much the epitome needs to adapt to best approximate a novel image. Specifically, we use the formula (compare with Equation 4 of the main paper)

$$\left(\sum_{i,p} \gamma_i(k,p) \frac{\alpha_i^2}{c_i^2} \mathbf{T}_p^T \mathbf{\Sigma}_0^{-1} \mathbf{T}_p\right) \Delta\boldsymbol{\mu}_k = \sum_{i,p} \gamma_i(k,p) \frac{\alpha_i}{c_i^2} \mathbf{T}_p^T \mathbf{\Sigma}_0^{-1} (\mathbf{x}_i - \alpha_i \mathbf{T}_p \boldsymbol{\mu}_k)\,, \quad (1)$$

where i is an index running over all patches extracted from a single image. In our current implementation we use hard assignments (found by ANN search) to set the weights $\gamma_i(k,p)$.

We can use the set of $\{\Delta\boldsymbol{\mu}_k\}_{k=1}^K$ directly as a feature vector for image classification, similarly to the VLAD encoding method for SIFT features [9]. We have performed preliminary experiments exploring this idea on the PASCAL VOC-2007 dataset and obtained performance 51.61% mAP with the *Epitome-256-12/8* dictionary.

The epitomic footprint descriptor can be visualized, as shown in Figure 6 of the main paper. An intriguing practical benefit of this is that we can compress and store the footprint descriptor as a JPEG image. For the *Epitome-256-12/8* dictionary the descriptor has length $256 \cdot 12 \cdot 12 \cdot 2 = 73,728$ (the two factor is for the sign encoding) and thus occupies 288 KB of memory if stored as single precision array. Preliminary experiments indicate that the epitomic descriptor can be stored as JPEG image with file size less than 10 KB with negligible distortion for a 30× compression rate. Thanks to the generative nature of our model we can thus use optimized off-the-shelf tools from image processing and coding. This is in sharp contrast to SIFT-based representations, where the feature compression has required the development of specialized algorithms and software [9].

References

[1] L. Bo, X. Ren, and D. Fox. Hierarchical matching pursuit for image classification. In *NIPS*, 2011.

[2] L. Bo, X. Ren, and D. Fox. Multipath sparse coding using hierarchical matching pursuit. 2013.

[3] K. Chatfield, V. Lempitsky, A. Vedaldi, and A. Zisserman. The devil is in the details: an evaluation of recent feature encoding methods. In *BMVC*, 2011.

[4] A. Coates, H. Lee, and A. Ng. An analysis of single-layer networks in unsupervised feature learning. In *AISTATS*, 2011.

[5] A. Coates, H. Lee, and A. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proc. ICML*, 2011.

[6] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.

[7] Z. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *CVPR*, 2006.

[8] M. Muja and D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP*, 2009.

[9] J. Sánchez, F. Perronnin, T. Mensink, and J. J. Verbeek. Image classification with the Fisher vector: Theory and practice. *IJCV*, 105(3):222–245, 2013.

[10] M. Shneier and M. Abdel-Mottaleb. Exploiting the JPEG compression scheme for image retrieval. *PAMI*, 18(8), 1996.

[11] G. Strang. The discrete cosine transform. *SIAM Review*, 41(1):135–147, 1999.

[12] A. Vedaldi and B. Fulkerson. VLFeat: An open and portable library of computer vision algorithms, 2008.

[13] A. Vedaldi and A. Zisserman. Efficient additive kernels via explicit feature maps. *PAMI*, 34(3):480–492, 2012.

[14] M. Zeiler, G. Taylor, and R. Fergus. Adaptive deconvolutional networks for mid and high level feature learning. In *ICCV*, 2011.