Dictionary Learning in Texture Classification

Mehrdad J. Gangeh[1], Ali Ghodsi[2], and Mohamed S. Kamel[1]

[1] Pattern Analysis and Machine Intelligence (PAMI) Lab,
Department of Electrical and Computer Engineering, University of Waterloo,
200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1
{mgangeh,mkamel}@pami.uwaterloo.ca
[2] Department of Statistics and Actuarial Science, University of Waterloo,
200 University Avenue West, Waterloo, Ontario, Canada N2L 3G1
aghodsib@uwaterloo.ca

Abstract. Texture analysis is used in numerous applications in various fields. There have been many different approaches/techniques in the literature for texture analysis among which the texton-based approach that computes the primitive elements representing textures using *k*-means algorithm has shown great success. Recently, dictionary learning and sparse coding has provided state-of-the-art results in various applications. With recent advances in computing the dictionary and sparse coefficients using fast algorithms, it is possible to use these techniques to learn the primitive elements and histogram of them to represent textures. In this paper, online learning is used as fast implementation of sparse coding for texture classification. The results show similar to or better performance than texton based approach on CUReT database despite of computation of dictionary without taking into account the class labels.

Keywords: Dictionary learning, matrix factorization, sparse coding, texture classification.

1 Introduction

Texture provides important information in various fields of image analysis and computer vision. It has been used in many different problems including texture classification, texture segmentation, texture synthesis, material recognition, 3D shape reconstruction, color-texture analysis, appearance modeling, and indexing [1-4].

As texture is a complicated phenomenon, there is no definition that is agreed upon by the researchers in the field [2, 3]. This is one of the reasons that there are various analysis techniques in the literature, each of which tries to model one or several properties of texture depending on the application in hand.

Among these techniques, the approaches based on representing textures using some primitive elements, either predefined or learned, has recently shown great success in texture analysis. These approaches have roots in influential paper by Julesz [5]. He introduced textons as fundamental primitive elements that can describe

M. Kamel and A. Campilho (Eds.): ICIAR 2011, Part I, LNCS 6753, pp. 335–343, 2011.

textures. However, he did not propose any method how to compute these primitive elements in [5].

Based on Julesz proposal, two techniques have recently obtained prevalence in texture analysis. First, techniques based on local binary patterns (LBPs) [6], in which *fixed* operators, i.e., LBPs and their histogram are used to represent a texture. Second, texton-based approach where *learned* textons (composed in a dictionary) are used as primitive elements to represent a texture. Our focus in this paper is on this latter approach, i.e., *learned* dictionary of textons.

Leung and Malik were the first to develop a complete texture classification system using texton-based approach [7]. They defined 2D textons as the cluster centers in filter bank responses, which made it possible to generate textons from the images automatically as the prototypes representing the source textures. These textons formed a dictionary from which a texton histogram could be constructed for each image using a similarity measure. Their work was further improved by Schmid [8], Cula and Dana [9], and Varma and Zisserman [10, 11].

In texton-based approach, the textons in the dictionary are learned using a clustering algorithm such as *k*-means. However, as explained in [10], one main shortcoming of *k*-means is that it can be only applied to points within a texture class. It cannot be applied across classes as it merges data points (by taking mean of points) and thus the resultant cluster centers cannot be identified uniquely with individual textures. This means that the cluster centers computed using *k*-means across classes are not representing textures in a class anymore.

A solution to this problem is computing the dictionary using dictionary learning approaches based on sparse coding or using matrix factorization[1]. Previously, these approaches for dictionary learning were too slow to be utilized in these applications. However, with recent advances in this field and by introducing fast algorithms such as online learning [12], rank-one downdate (R1D) [13], and coordinate descent [14], it is now computationally feasible to compute the dictionary on millions of patches (data samples in general) in reasonable time. This means that the dictionary can be learned on whole training set (not per class) using these approaches. The main advantage is that we do not use the class labels at this stage, i.e., learning dictionary is fully unsupervised.

Here, we propose using online learning [12] for learning a dictionary on the whole training set and computation of sparse coefficients over the whole dictionary and show that despite of fully unsupervised learning of dictionary, on standard databases such as Columbia Utrecht Reflectance and Texture (CUReT) database [15], it performs similar to or better than texton-based approaches using *k*-means, where dictionary is learned per class.

The rest of the paper is organized as follows: Section 2 presents the theory of dictionary learning and sparse coding (DLSC) related to our work. Experimental setup is described in Section 3 followed by results in Section 4. The paper is concluded in Section 5.

[1] The connection between matrix factorization and dictionary learning using sparse coding is explained in [12].

2 Dictionary Learning and Sparse Coding

In this section, we first provide an overview of dictionary learning and sparse coding (DLSC) and its connection to texton-based approach for texture classification. Then we provide the formulation for dictionary learning with sparse representation for texture classification.

2.1 Background

Dictionary learning and sparse representation/coding are two closely related topics in the literature. The initial work on these two topics was originated from two communities and problems under two different names, i.e., sparse coding (SC), which was originated by neurologists as a model for simple cells in mammalian primary visual cortex [16, 17]; and, independent component analysis (ICA), which was originated by researchers in signal processing to estimate the underlying hidden components of multivariate statistical data (refer to [18] for a review of ICA). These two problems merged, eventually, into similar techniques, but somewhat different description (the connection between SC and ICA is also explained in [18]).

The main result of these two research works was that a class of signals with sparse nature, such as the images of natural scenes, can be represented using some primitive elements that form a dictionary, and that each signal in this class, can be represented by using only few elements in the dictionary (sparse representation).

In fact, there are, at least, two ways in the literature to exploit sparsity [19]: first, using a linear/nonlinear combination of some *predefined* bases, e.g., wavelets [20]. Second, by using primitive elements in a *learned* dictionary, such as techniques employed in SC or ICA. This latter approach is our focus in this paper.

As mentioned in the introduction, dictionary learning was introduced to the field of texture analysis by Julesz theory that stated textures can be represented using a few primitive elements [5] and following the work done in [7, 8, 9, 10, 11] that initiated the texton-based approach in texture classification. Texton-based approach mainly consists of two steps, dictionary learning and computation of models (features) for each texture image. In the first step, extracted patches from each texture image in a class are submitted to a clustering algorithm such as *k*-means and obtained cluster centers are used as primitive elements (called textons) that form the dictionary. In the second step, for each texture image, a histogram of textons is computed. To compute this histogram, patches are extracted from each texture image and each patch is compared with the textons in the dictionary. The closest match based on a similarity measure such as Euclidean distance is used to update the corresponding bin in the histogram of textons. Thus, each patch in a texture image is represented by only one single texton in the dictionary (the closest match). This is a kind of sparse representation, in which only one atom in the dictionary is active per patch. These two steps can be performed using DLSC, which is described next.

2.2 Mathematical Formulation

Considering a finite training set of signals $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m] \in \mathbb{R}^d$, they can be represented by a dictionary $\mathbf{D}$ and a set of sparse coefficients $\boldsymbol{\alpha}$ using

$$\min_{\mathbf{D},\boldsymbol{\alpha}} \sum_{i=1}^{m} \left(\frac{1}{2} \|\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda \varphi(\boldsymbol{\alpha}_i) \right), \tag{1}$$

where λ is a regularization parameter and $\varphi(.)$ is a sparsity inducing function. The most common sparsity inducing function is norm and the corresponding problem is known as the *Lasso* [21]

$$\min_{\mathbf{D},\boldsymbol{\alpha}} \sum_{i=1}^{m} \left(\frac{1}{2} \|\mathbf{x}_i - \mathbf{D}\boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1 \right), \tag{2}$$

To prevent obtaining very large values of **D**, which consequently leads to very small values of , a constraint is imposed on the columns of **D** such that they have unit norm [12].

Solving (2) using one of the approaches in the literature such as online learning [12] yields the dictionary **D** and the sparse coefficients $\boldsymbol{\alpha}$. If the dictionary has been already computed (using all $\mathbf{x}_i, i = 1, \ldots,$ in the training set), (2) can be used to find the sparse coefficients for a signal **x** in test set (**D** is fixed in this case).

2.3 Texture Classification

Texture classification using dictionary learning and sparse representation based on (2) can be done in two steps. In first step, the dictionary $\mathbf{D} \in \mathbb{R}^{d}$ (k is the number of primitive elements in the dictionary) is learned using $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m] \in \mathbb{R}^{d}$, where $\mathbf{x}_i, i = 1, \ldots,$ are patches extracted with size $\sqrt{d} \times$ from texture images in training set in all classes. With fast algorithms such as R1D or online learning, this can be performed in few minutes over millions of patches.

After learning the dictionary, we need to find the model (feature set) for each texture image in training and test sets. To this end, patches of the same size as what is used in dictionary learning step are extracted from each texture image, i.e., $\mathbf{X} = [\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d}$, where n is the number of patches extracted, which is not necessarily the same as m. Then using (2), the corresponding coefficients $\boldsymbol{\alpha}_i \in \mathbb{R}^{k \times n}, i = 1, \ldots$ are computed. For each patch , most of the elements in the corresponding coefficient are zero. The nonzero elements in determine the primitive elements in the dictionary **D** that contribute towards the representation of the patch . If we sum up all these coefficients for all patches extracted from a texture image, we effectively find the histogram of primitive elements contributing towards the representation of this particular texture, i.e.,

. (3)

We impose a positive constraint on in (2) such that we eventually obtain a histogram H with positive values in all bins. This also prevents cancelling the effect of different patches when they are summed up in (3). Hence, we rewrite (2) as follows to consider this constraint as well as the constraint we considered on **D** columns in previous subsection

$$\min_{\mathbf{D},\alpha} \sum_{i=1}^{m} \left(\frac{1}{2} \|\mathbf{x}_i - \mathbf{D}\alpha_i\|_2^2 + \lambda \|\alpha_i\|_1 \right) \quad \text{s.t.} \quad \forall j = 1, \ldots, k, \quad , \tag{4}$$

where is the j^{th} column of **D**. In this way, while in texton-based approach each patch is represented using only the closest texton in the dictionary, here each patch is represented by using several primitive elements in the dictionary and hence it can potentially provide richer representation than texton-based approach. The number of nonzero elements in can be controlled using λ in (4), i.e., larger values of λ yield sparser coefficients [12].

The distance between two normalized histograms is measured using statistic, i.e., using $\chi^2(\mathbf{H}_1, \mathbf{H}_2) = 1/2 \sum_i (h_{1i} - h_{2i})^2 / (h_{1i} + h$. One nearest neighbor is used as the classifier as suggested in [11].

Although, dictionary learning is also used in [22] for texture classification, our work is different in following three aspects. Firstly, in [22] one dictionary is learned per class and then these dictionaries are composed (concatenated) to form the overall dictionary (this is the same as what is reported in the literature for finding dictionary using *k*-means). We find the dictionary on the whole training set (not per class) and this means that we do not use class labels at this stage at all. Secondly, to find the sparse coefficients, in [22] part of dictionary which is most similar to the current patch is considered (it is not explained what kind of similarity is used) and the reason mentioned is that using the whole dictionary is computationally very expensive. We find the sparse coefficients on whole dictionary (this is possible with recent advances in computation of the *Lasso* in the literature as mentioned before). Thirdly, we have placed positive constraint on the coefficients as we eventually sum them up to find the histogram of primitive elements (in the dictionary) as the feature set for an image to be classified. In [22] this positive constraint on the coefficients is not considered and this might not be needed as the coefficients are not found on the whole dictionary but just on part of dictionary most similar to the current patch. In fact, our experiments show that without this positive constraint on the coefficients, the performance of the classification system is very poor.

3 Experimental Setup

The performance of the proposed classification system is evaluated on CUReT database. The database is used the same as what is reported in [11]. That is, there are 92 images per class and 61 classes. Each image is 200 × 200 pixels with the intensity resolution of 8 bit/pixel. The comparison is made with texton-based approach using raw pixel representation. This means that no filter banks are used.

Data Preparation and Preprocessing. To make the images indiscriminable to the average intensity level and contrast, the mean of texture images is removed and they are also normalized to have unit standard deviation.

Computation of Dictionary. To compute the dictionary, 500 random patches are extracted from each texture image in the training set. Patch sizes of 5 × 5, 7 × 7, and 9 × 9 are used in the experiments. No filter banks are applied and raw pixel representation is used. The mean of patches are removed to make the images locally invariant to the average intensity. In texton-based approach, Weber's law normalization is used as reported in [10, 11]. In DLSC, each patch is normalized to have unit norm. This is done based on the constraint on primitive elements in the dictionary as stated in (4). In texton-based approach, all patches belonging to one class are submitted to the *k*-means algorithm to find the cluster centers. These cluster centers over all classes are then composed into a single dictionary. In DLSC approach, all patches from all classes are used at once for learning the dictionary. Hence, no class labels are used at this stage. Online learning [12] is used for the implementation of (4) in DLSC. As suggested in [12], the regularization parameter λ in (4) is chosen as $1.2/\sqrt{d}$, where d = (patch size). This yields about 10 nonzero coefficients in average for the patches of 9 × 9.

Learning Models (Histograms). After computation of the dictionary, we need to find the model. To this end, small overlapping patches with the same size as what was used in the previous step are extracted from the top left to the bottom right of each ROI. As in the dictionary learning, no filter bank is used and raw pixel representation is considered. The mean of each patch is removed and they are normalized according to Weber's Law in texton-based approach and to unit ℓ_2 norm in DLSC. In texton-based approach, Euclidean distance is used as the similarity measure to find the closest texton in the dictionary to each patch. In DSLC, online learning implementation of (4) is used with the same λ value as previous step with positive constraint on the coefficients α. Each coefficient is normalized to sum to one and all of them are then summed up to yield the overall frequency histogram of primitive elements for each texture image, which is used as the signature (model) of the particular texture image after normalization.

4 Results

In this section, we present the results of texture classification on CUReT database using both texton-based and DLSC approaches.

Fig. 1 compares the dictionary learned using these two techniques. In texton-based approach, 10 textons are learned in each class using *k*-means and eventually all textons are composed into a dictionary (610 textons for 61 classes). As can be seen in Fig. 1, every 10 adjacent textons are similar as they are taken from the same class. In DLSC approach, all patches extracted from all classes are used for the learning of dictionary using (4). Hence class labels are not used at this stage. Different from texton-based dictionary, the primitive elements from all classes are spread over entire dictionary in DLSC.

Table 1 shows the performance of one nearest neighbor classifier using texton-based and DLSC approaches. The experiments are repeated 100 times over random sets of training and test sets. The performance is compared for three different patch and four different training set sizes. As can be seen from this table, the performance

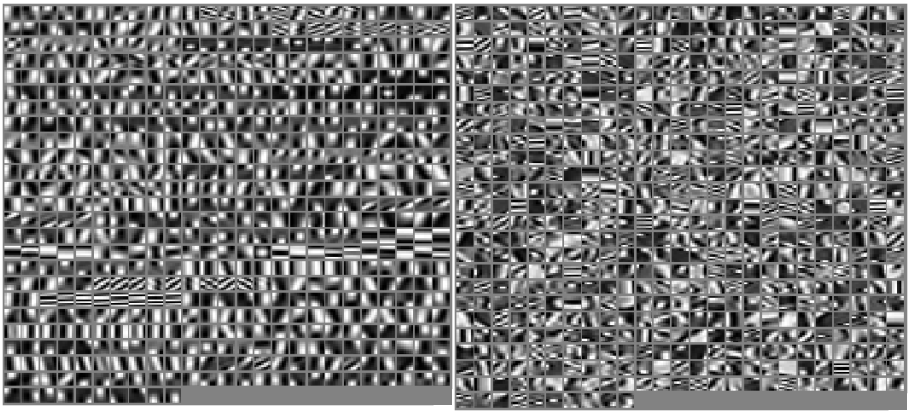

Fig. 1. Dictionary of 610 primitive elements learned using patches of size 7 × 7 extracted from 23 training texture images per class using: (*left*) *k*-means algorithm where 10 textons per class are learned and all these textons are composed into a dictionary and (*right*) DLSC as described in this paper where all primitive elements are learned at once by submitting all extracted patches from all classes to (4).

Table 1. Comparison between the classification accuracy of texton-based and DLSC approaches. The experiments are repeated 100 times on various random split of training and test sets. The dictionary is consisting of 610 primitive elements and results are reported for different train and patch sizes.

Patch Size / Train Size	5 × 5		7 × 7		9 × 9	
	Texton	DLSC	Texton	DLSC	Texton	DLSC
6	73.76 ± 4.25	74.22 ± 4.37	74.73 ± 4.15	75.77 ± 4.27	75.65 ± 3.92	76.32 ± 4.14
12	83.46 ± 2.60	84.02 ± 2.55	84.25 ± 2.66	85.03 ± 2.55	85.20 ± 2.54	85.32 ± 2.49
23	90.09 ± 1.56	90.52 ± 1.55	90.81 ± 1.62	91.33 ± 1.57	91.42 ± 1.61	91.62 ± 1.59
46	94.83 ± 0.95	95.26 ± 0.93	95.49 ± 0.93	95.85 ± 0.87	95.94 ± 0.85	96.14 ± 0.87

of DLSC is similar to or better than texton-based approach in all cases. This is while the dictionary of DLSC is learned over whole training set at once whereas dictionary of texton-based approach is learned per class, i.e., class labels are taken into account in this learning.

5 Discussion and Conclusion

Sparse representation using few primitive elements learned from data has recently shown great success in different fields such as face recognition and denoising. One of main obstacles for widespread application of this approach was rather slow algorithms

for the computation of dictionary and sparse coefficients over millions of data samples, which is usually the case in image processing and computer vision tasks. The initial algorithm proposed in [16], for example, took hours to compute the dictionary over patches extracted from only ten natural scenes.

With recent fast algorithms proposed for dictionary learning and sparse coding such as online learning and R1D, it is now feasible to perform the computation over millions patches in few minutes. In this paper, we proposed using one of these algorithms, i.e., online learning, for the purpose of texture classification over large databases such as CUReT. In contrast to *k*-means algorithm used in texton-based approach that has to learn the dictionary per class, the proposed approach can learn the dictionary over all classes and hence class labels are not used at all in this step. Yet, the results of classification are similar to or better than texton-based approach. The positive constraint imposed on sparse coefficients enables learning the coefficients over whole dictionary and, consequently, finding the model histogram for each texture image is as simple as summing up the sparse coefficients learned for all patches extracted from the particular texture image.

In future work, we would also like to impose positive constraint on the dictionary and utilize nonnegative matrix factorization using fast implementations such as R1D [13]. We would also like to extend this work to supervised dictionary learning [19] and compare it to our current results for possible further improvements.

Acknowledgments. The first author gratefully acknowledges the funding from the Natural Sciences and Engineering Research Council (NSERC) of Canada under Canada Graduate Scholarship (CGS D3-378361-2009).

References

1. Petrou, M., Sevilla, P.G.: Image Processing Dealing with Texture. John Wiley and Sons, West Sussex (2006)
2. Ahonen, T., Pietikainen, M.: Image Description Using Joint Distribution of Filter Bank Responses. Pattern Recognition Letters 30(4), 368–376 (2009)
3. Mirmehdi, M., Xie, X., Suri, J.: Handbook of Texture Analysis. Imperial Collage Press, London (2008)
4. Hadjidemetriou, E., Grossberg, M.D., Nayar, S.K.: Multiresolution Histograms and Their Use for Recognition. IEEE Trans. on PAMI 26(7), 831–847 (2004)
5. Julesz, B.: Textons, the Elements of Texture Perception, and Their Interactions. Nature 290(5802), 91–97 (1981)
6. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns. IEEE Trans. on PAMI 24(7), 971–987 (2002)
7. Leung, T., Malik, J.: Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons. Int'l J. Computer Vision 43(1), 29–44 (2001)
8. Schmid, C.: Weakly Supervised Learning of Visual Models and Its Application to Content-Based Retrieval. International Journal of Computer Vision 56(1/2), 7–16 (2004)
9. Cula, O.G., Dana, K.J.: 3D Texture Recognition Using Bidirectional Feature Histograms. International Journal of Computer Vision 59(1), 33–60 (2004)

10. Varma, M., Zisserman, A.: A Statistical Approach to Texture Classification from Single Images. International Journal of Computer Vision: Special Issue on Texture Analysis and Synthesis 62(1-2), 61–81 (2005)
11. Varma, M., Zisserman, A.: A Statistical Approach to Material Classification Using Image Patch Exemplars. IEEE Trans. on PAMI 31(11), 2032–2047 (2009)
12. Marial, J., Bach, F., Ponce, J., Sapiro, G.: Online Learning for Matrix Factorization and Sparse Coding. Journal of Machine Learning Research 11, 19–60 (2010)
13. Biggs, M., Ghodsi, A., Vavasis, S.: Nonnagative Matrix Factorization via Rank-One Downdate. In: Int'l Conf. on Machine Learning (ICML), Helsinki, Finland, pp. 64–71 (2008)
14. Friedman, J., Hastie, T., Tibshirani, R.: Regularized Paths for Generalized Linear Models via Coordinate Descent. Journal of Statistical Software 33(1), 1–22 (2010)
15. Dana, K.J., van Ginneken, B., Nayar, S.K., Koenderink, J.J.: Reflectance and Texture of Real-World Surfaces. ACM Transactions on Graphics 18(1), 1–34 (1999)
16. Olshausen, B.A., Field, D.J.: Emergence of Simple-Cell Receptive Field Properties by Learning a Sparse Code for Natural Images. Nature 381, 607–609 (1996)
17. Olshausen, B.A., Field, D.J.: Sparse Coding with an Overcomplete Basis Set: A Strategy Employed by V1? Vision Research 37(23), 3311–3325 (1997)
18. Hyvärinen, A., Karhunen, J., Oja, E.: Independent Component Analysis. John Wiley and Sons, New York (2001)
19. Marial, J., Bach, F., Ponce, J., Sapiro, G., Zisserman, A.: Supervised Dictionary Learning. In: 22nd Conference on Neural Information Processing Systems (NIPS), Vancouver, Canada, pp. 1033–1040 (2008)
20. Mallat, S.: Wavelet Tour of Signal Processing: The Sparse Way, 3rd edn. Academic Press, Burlington (2009)
21. Tibshirani, R.: Regression Shrinkage and Selection via the Lasso. Journal of the Royal Statistical Society, Series B 58(1), 267–288 (1996)
22. Xie, J., Zhang, L., You, J., Zhang, D.: Texture Classification via Patch-Based Sparse Texton Learning. In: Int'l Conf. on Image Processing (ICIP), Hong Kong, pp. 2737–2740 (2010)