Texture Classification Using Compressed Sensing

Li Liu
National University of Defense Technology
School of Electronic Science and Engineering
Changsha, Hunan, China
feiyunly@engmail.uwaterloo.ca

Paul Fieguth
University of Waterloo
Department of System Design Engineering
Waterloo, Ontario, Canada
pfieguth@uwaterloo.ca

Abstract

This paper presents a simple, novel, yet very powerful approach for texture classification based on compressed sensing and bag of words model, suitable for large texture database applications with images obtained under unknown viewpoint and illumination. At the feature extraction stage, a small set of random features are extracted from local image patches. The random features are embedded into the bag of words model to perform texture classification. Random feature extraction surpasses many conventional feature extraction methods, despite their careful design and complexity. We conduct extensive experiments on the CUReT database to evaluate the performance of the proposed approach. It is demonstrated that excellent performance can be achieved by the proposed approach using a small number of random features, as long as the dimension of the feature space is above certain threshold. Our approach is compared with recent state-of-the-art methods: the Patch method (Varma and Zisserman, TPAMI 09), the MR8 filter bank method (Varma and Zisserman, IJCV 05) and the LBP method (Ojala et al., TPAMI 02). It is shown that the proposed method significantly outperforms MR8 and LBP and is at least as good as the Patch method with drastic reduction in storage and computational complexity.

1 Introduction

Texture is ubiquitous in natural images and constitutes a important visual cue for a variety of image analysis and computer vision applications like image segmentation, image retrieval and shape from texture. Texture classification is a fundamental issue in computer vision and image processing, playing a significant role in a wide range of applications that include medical image analysis, remote sensing, object recognition, content-based image retrieval and many more.

A Recent "Bag of Words" (BoW) approach, borrowed from the text literature, opens up a new prospect for texture classification. The BoW encodes both the local texture information by using feature extractors to extract texture information from local patches to form textons, and the global texture appearance by statistically computing a orderless histogram for each image representing the frequency of the repetition of the textons. However, the local feature extractors from which texton dictionary is built still play a crucial role.

There are two main ways to construct the texton dictionary: 1) detecting a sparse set of points in a given image using Local Interest Point (LIP) detectors and then using local descriptors to extract features from a local patch centered at the LIPs [1] [2], 2) extracting local features pixel by pixel over the input image densely. The dense approach is more common and widely studied. Among the most popular dense descriptors are the use of large support filter banks to extract texture features at multiple scales and orientations [3] [4] [5]. However, more recently, in [6] the authors challenge the dominant role that filter banks have been playing in texture classification area, and claim that classification based on textons directly learned from the raw image pixels outperforms that based on textons based on filter bank responses.

The key parameter in patch-based classification is the size of the patch. Small patch sizes cannot capture large-scale structures that may be the dominant features of some textures, are not very robust against local changes in texture, and are highly sensitive to noise and missing pixel values caused by illumination variations. However, the disadvantage of the patch representation is the quadratic increase in the dimension of the patch space with the size of patch. The high dimensionality poses two challenges to the clustering algorithms used to learn textons: First the present of irrelevant and noisy features can mislead the clustering algorithm; Second, in high dimensions data may be very sparse (the curse of dimensionality), making it difficult for an algorithm to find any structure.

It is therefore natural to ask whether high dimensional

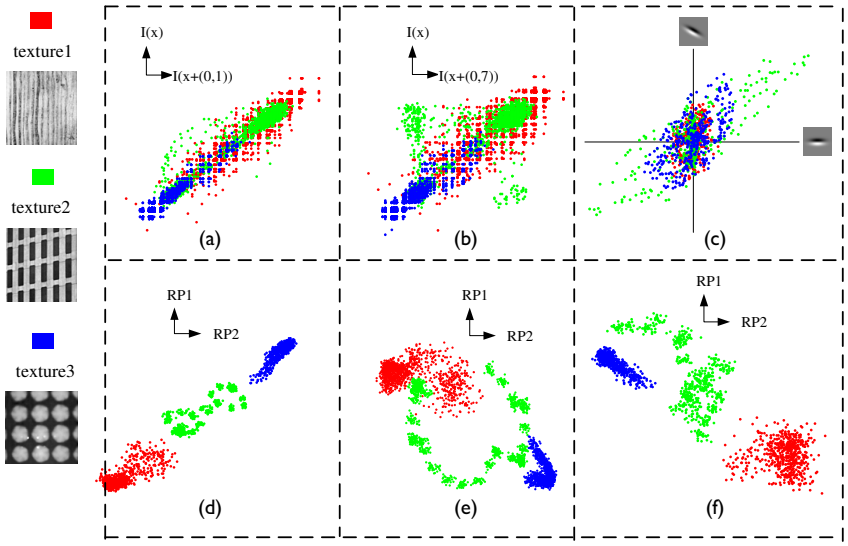

Figure 1. Compressed sensing measurements of local patches form good shape clusters and distinguish texture classes. Three textures, leftmost, are shown from the Brodatz database. Compare the spatial distribution and separability of: (a) (b) raw pixel values, (c) two linear filter responses (computed with support region 49×49), and random (CS) features extracted from patches of size (d) 9×9, (e) 15×15, (f) 25×25.

patch vectors can be projected into a lower dimensional subspace without suffering great information loss. There are many potential benefits of a low dimensional space: reduced storage requirements, reduced computational complexity and possibly improved classification performance. A small salient feature set simplifies both the pattern representation and the subsequent classifiers used. This brings us into the realm of recent theory of compressive sensing.

The compressed sensing (CS) approach [7] [8] [9], the motivation for this research, is appealing because its surprising results that high-dimensional sparse data can be accurately reconstructed from just a few nonadaptive linear random projections. When applying CS to our texture classification problem, the key question is therefore how much information about the local texture patches is preserved by these random projections.

The abilities of CS for perfect signal reconstruction have been proved [7] [8]. A natural question emerges, can the power of CS be leveraged in the texture classification problem? The application of CS for texture classification problem we investigate here has received only a minimal treatment to date. Limited work has been reported [10] [11], exploiting the specific structure of sparse coding for texture patches, depending on the recovery process and careful design of the sparsifying dictionary [12]. In contrast, our work performs classification in the compressed space, not relying on any reconstruction process. We present a comprehensive series of experiments intended to precisely illustrate the benefits of this novel theory for texture classification.

The proposed method is computationally simple, yet very powerful. Instead of performing texture classification in the original high dimensional patch space or making efforts to figure out which feature extraction method is suitable for all types of textures, we just use random projections and perform texture classification in a much lower dimensional space. The theory of compressed sensing helps to remove these difficulties and indicates that the precise choice of feature space is no longer critical: random features contain enough information to preserve the underlying local texture structure and hence correctly classify any test image. Figure 1 explores this claim, contrasting the distribution of raw pixels, filter responses and random CS features. Clearly, Figure 1 is anecdotal evidence and in no way comprehensive.

The rest of this paper is organized as follows. Section 2 reviews the CS background. In Section 3, we present the details of the proposed features and the texture classification framework and discuss the benefits and advantages of the proposed method in details. In Section 4, we verify the proposed method with extensive experiments on

benchmark texture database CUReT and provide comparisons with three state-of-the-art methods: the patch method, MR8 filter bank and LBP method. Section 5 concludes the paper.

2 Background

The theory of compressed sensing has recently been brought to the forefront by the work of Candès and Tao [7] and Donoho [8], who have shown the advantages of random projections for capturing information about sparse or compressible signals. CS is based on the premise that a small number of random linear measurements of a compressible signal or image contains enough information for reconstruction and processing. This emerging theory has generated enormous amounts of research with applications such as high-dimensional geometry, image reconstruction, image compression, machine learning and data-streaming algorithms [11] [13] [14] [15]. The beauty of the CS theory is that if a signal may be sparsely represented in some basis, it may be perfectly recovered based on a relatively small set of random projection measurements. CS relies on two fundamental principles. Compressive sensing measurement process is illustrated in Figure 2.

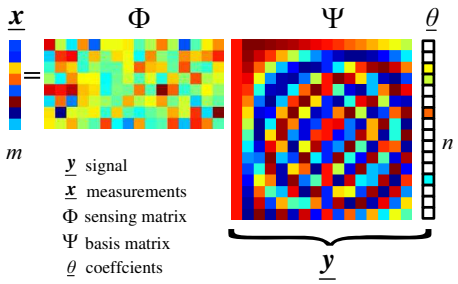

Figure 2. Compressed Sensing measurement process

1. *Sparsity*: Let $\underline{\boldsymbol{y}} \in \mathbb{R}^{n\times 1}$ be an n-pixel image and $\boldsymbol{\Psi} = [\underline{\boldsymbol{\psi}}_1 \ \ldots \ \underline{\boldsymbol{\psi}}_n]$ an orthnormal basis (dictionary), where $\underline{\boldsymbol{\psi}}_i \in \mathbb{R}^{n\times 1}$, such that

$$\underline{\boldsymbol{y}} = \sum_{i=1}^{n} \theta_i \underline{\boldsymbol{\psi}}_i = \boldsymbol{\Psi}\underline{\boldsymbol{\theta}} \tag{1}$$

   where $\underline{\boldsymbol{\theta}} = [\theta_1 \ \ldots \ \theta_n]^T$ denotes the vector of coefficients that represents $\underline{\boldsymbol{y}}$ in the basis $\boldsymbol{\Psi}$. A signal or image is said to be sparse if most of the coefficients in $\underline{\boldsymbol{\theta}}$ are zero or they can be discarded without much loss of "information". If the coefficients sorted in decreasing order decay rapidly, then $\underline{\boldsymbol{y}}$ is said to be (approximately) compressible.

2. *Incoherent Sampling*: Let $\boldsymbol{\Phi} = [\underline{\boldsymbol{\phi}}_1^T \ \ldots \ \underline{\boldsymbol{\phi}}_m^T]$ be an $m\times n$ sampling matrix, with $m \ll n$, such that $\underline{\boldsymbol{x}} = \boldsymbol{\Phi}\underline{\boldsymbol{y}}$ is an $m \times 1$ vector of linear measurements. While the matrix $\boldsymbol{\Phi}\boldsymbol{\Psi}$ is rank deficient, and hence loses information in general, it can be shown to preserve the information in sparse and compressible signals if it satisfies the so-called *restricted isometry property* [9].

In the CS problem, we are interested in economically recording information about a signal $\underline{\boldsymbol{y}}$. We allocate a budget of m nonadaptive questions to ask about $\underline{\boldsymbol{y}}$. Each question takes the form of a linear functional applied to $\underline{\boldsymbol{y}}$. Thus, the information we extract from $\underline{\boldsymbol{y}}$ is given by $\underline{\boldsymbol{x}} = \boldsymbol{\Phi}\underline{\boldsymbol{y}}$.

3 Texture Classification Using Compressed Sensing

To present the idea, assume that there are *C* distinct texture classes, with each class having *S* samples. We start the description of our approach by defining texture pattern in a local patch of size $\sqrt{n} \times \sqrt{n}$ of a texture image as the joint distribution of the gray levels of n image pixels by row-recording the pixels in the patch into a vector $\underline{\boldsymbol{p}}$. That is, we treat an local image patch by vectorizing it into a long one-dimensional vector.

The patch method [6] is based on the VZ algorithm [4] (following the terminology of Varma and Zisserman) by replacing the filter responses with the source patch vector $\underline{\boldsymbol{p}}$. Our proposed classifier is identical to the patch method except that, at the feature extracting stage, instead of using $\underline{\boldsymbol{p}}$, the compressed sensing measurements $\underline{\boldsymbol{x}} = \boldsymbol{\Phi}\underline{\boldsymbol{p}}$ are used as features, where $\boldsymbol{\Phi} \in \mathbb{R}^{m\times n}$ denotes the random compressed sensing measurement matrix, defined in Section 2. In this paper, we choose $\boldsymbol{\Phi}$ to be a Gaussian random matrix, i.e., the entries of $\boldsymbol{\Phi}$ are independently sampled from a zero mean normal distribution. Throughout this paper, the patch space is defined as

$$\mathcal{P} = \{\underline{\boldsymbol{p}} : \underline{\boldsymbol{p}} \in \mathbb{R}^{n\times 1}\} \tag{2}$$

which we call the *patch space*. We will work with the *compressed space* $\mathcal{X}$ defined as

$$\mathcal{X} = \{\underline{\boldsymbol{x}} : \underline{\boldsymbol{x}} = \boldsymbol{\Phi}\underline{\boldsymbol{p}},\ \underline{\boldsymbol{x}} \in \mathbb{R}^{m\times 1},\ \underline{\boldsymbol{p}} \in \mathcal{P}\} \tag{3}$$

In other words, $\mathcal{X}$ is a compressed representation of $\mathcal{P}$.

Our texture classification system consists of the following stages:

1. Compressed texton dictionary learning stage. In this stage, we learn a universal compressed texton dictionary directly in the compressed domain $\mathcal{X}$ which is different from the patch method where the texton dictionary is built from the patch domain $\mathcal{P}$. The global

compressed texton dictionary learnt at this stage is denoted as $\mathcal{W} = \{\{\underline{\boldsymbol{w}}_{c,j}\}_{j=1}^{K}\}_{c=1}^{C}$, where K is the number of textons learnt from each class.

2. Histogram of textons learning stage. Given the compressed texton dictionary obtained in the stage 1, a histogram of compressed textons is learnt for each particular training sample by labeling each of the image pixels with the compressed texton that lies closest to it in the compressed domain. Each texture class c then is represented by a set of models $\mathcal{H}_c = \{\underline{\boldsymbol{h}}_{c,s}\}_s, \underline{\boldsymbol{h}}_{c,s} \in \mathbb{R}^{CK\times 1}$ corresponding to the training samples of that class. Each histogram of textons needs to be normalized, $\underline{\boldsymbol{h}}_{c,s}(k) = \underline{\boldsymbol{h}}_{c,s}(k)/\sum_{k=1}^{CK}\underline{\boldsymbol{h}}_{c,s}(k), \quad k = 1, ..., CK$.

3. The classification stage. The set of learnt models from stage 2 is used to classify a novel sample into one of the C texture classes.The process to compute the normalized histogram of compressed textons $\underline{\boldsymbol{h}}_{new}$ for a novel image is the same as computing the model for each training sample. Then, the calculated model $\underline{\boldsymbol{h}}_{new}$ is used to classify it into one of the known classes by a classifier. We have chosen to use the simple nearest neighbor classifier (NNC), and the distance between two histograms is measured using the χ^2 statistic which is defined as

$$\chi^2(\underline{\boldsymbol{h}}_1, \underline{\boldsymbol{h}}_2) = \frac{1}{2}\sum_{k=1}^{CK}\frac{[\underline{\boldsymbol{h}}_1(k) - \underline{\boldsymbol{h}}_2(k)]^2}{\underline{\boldsymbol{h}}_1(k) + \underline{\boldsymbol{h}}_2(k)} \qquad (4)$$

4 Experimental Evaluation

4.1 Methods in Comparison Study

One of our main goals is compare the proposed approach with the state-of-the-art Patch method [6]. In addition, the following three popular methods will be used for comparison to further demonstrate the excellent performance of the proposed method:

Patch-MRF [6]: Patch-MRF is a variant of the Patch method. For the Patch-MRF, an image is represented using a two-dimensional histogram: one dimension is the quantized bins of the center pixel's intensities of local patches and the other dimension is the learnt textons from the patch vector with the center pixel excluded. The number of bins for the center pixel used in [6] is as large as 200 and the size of texton dictionary is $61 \times 40 = 2440$.

MR8 [6]: The MR8 consists of 8 filter responses derived from the original responses of 38 filters [4] [6]. A complicated anisotropic Gaussian filtering method [16] was used to calculate the MR8 responses [4].

LBP [17] [18]: The rotationally invariant, uniform LBP texton dictionary at different scales, $\text{LBP}^{\text{riu2}}_{8,1}$, $\text{LBP}^{\text{riu2}}_{8,1+16,2}$, $\text{LBP}^{\text{riu2}}_{8,1+16,2+24,3}$, $\text{LBP}^{\text{riu2}}_{8,1+16,2+24,3+24,4}$, $\text{LBP}^{\text{riu2}}_{8,1+16,2+24,3+24,4+24,5}$ advocated in [17] and [18], will be used for comparison with the proposed approach. For simplicity, in the remainder of this paper, these LBP textons are denoted as 1-scale, ... , 5-scale respectively. The histogram of textons model for each texture sample will be obtained by concatenating histograms produced by operators at three resolutions into a single histogram as used in [17].

4.2 Image Data and Experimental setup

For our experimental evaluation we have used two texture datasets, derived from the CUReT database which has now become a benchmark and is widely used to assess classification performance.

For the **CUReT large dataset $\mathcal{D}^C$ (61 classes)**, we use the same subset of images as Varma and Zisserman [4] [6] [20], containing 61 texture classes shown in Figure 3 with 92 images for each class, resulting a total of $61 \times 92 = 5612$ images. These images are captured under different illuminations with seven different viewing directions. In the experiments on this dataset, half (46 samples per class) of the samples are chosen for training and the remaining half are chosen for testing.

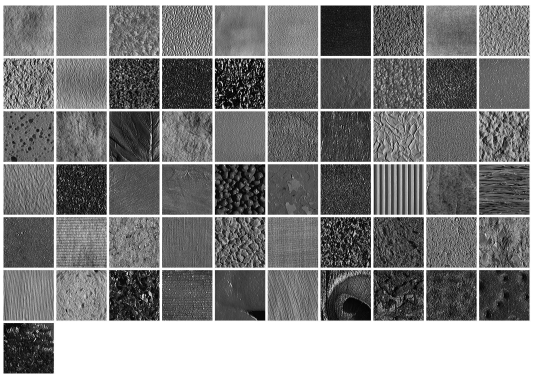

Figure 3. CUReT: The 61 textures in the CUReT database

The **CUReT small dataset $\mathcal{D}^c$ (61 classes)** preserves all texture classes of $\mathcal{D}^C$, however, each kind of textures is represented by only a single texture image taken from the original CUReT database [21], where all of the textures have the same illumination and imaging conditions. We partitioned each 320×320 texture image into nine 106×106 nonoverlapping sub-images, consistent with [19]. Out of the nine samples in each class, five samples are used as the training samples, and the other four samples are utilized as the testing data.

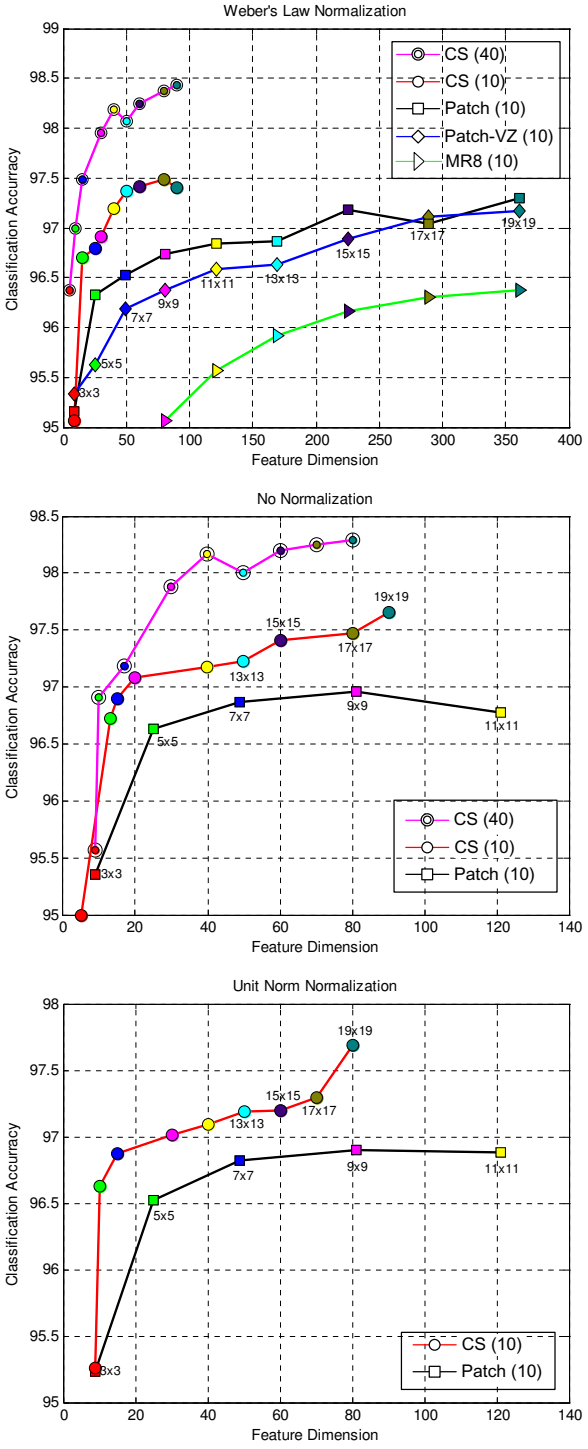

Figure 4. Classification results for the proposed method and the patch method on dataset $\mathcal{D}^C$ as a function of feature dimensionality. The number of the textons per class used is shown in the bracket of the legend. The bracketed values denote the number of the textons per class used. Classification rates obtained based on the same patch size are shown in the same color.

All our experimental results are reported using random partition of the training and testing set and random selection of the samples from the training set unless explicitly stated otherwise. In terms of the extracted CS vector, we consider three kinds of normalization:

1. Weber's law normalization:

$$\underline{\boldsymbol{x}} \leftarrow \underline{\boldsymbol{x}} \left[\frac{\log(1 + \|\underline{\boldsymbol{x}}\|_2/0.03)}{\|\underline{\boldsymbol{x}}\|_2} \right] \tag{5}$$

2. Unit norm normalization:

$$\underline{\boldsymbol{x}} \leftarrow \frac{\underline{\boldsymbol{x}}}{\|\underline{\boldsymbol{x}}\|_2} \tag{6}$$

3. No normalization.

4.3 Experimental Results

In this subsection, we compare the proposed approach specifically to the current state-of-the-art Patch method [6] [4] on the CUReT database. To make the comparison as meaningful as possible, we use the same experimental settings as Varma and Zisserman [6].

In their comprehensive study, Varma and Zisserman [4] presented six filter banks for texton-based texture classification on $\mathcal{D}^C$. They concluded that the rotationally invariant, multi-scale, Maximum Response MR8 filter bank yields better results than any other filter bank. The filter support they used there is 49×49. However, in their more recent study [6], they challenged the dominant role that filter banks have come to play in the texture classification field and claim that their Patch method outperforms even the MR8 filter bank and the Patch-MRF performs the best.

The topmost in Figure 4 presents a comparison of the performance of the CS classifier, the Patch classifier and the MR8 filter bank method. All the results are shown as a function of the actual feature dimension used, except the MR8 results, which are plotted against patch size. Our CS results are statistically significantly better than those of any other method. In particular, the variation in implementation and test between the Patch results in [6] (Patch-VZ) and our own (Patch) is small relative to the improvement offend by the two CS results (CS -10 and CS-40).

For any given patch of size larger than 3×3, the CS classifier performs better than the Patch method and much better than the MR8 filter bank. This is strong evidence that CS classifier possesses both of the advantages of these two methods, i.e., the Patch method's being able to achieve good performance than MR8 and MR8's lower dimensional feature space. It would thus appear that low dimensional CS features with all the salient information present in an image patch preserved is more beneficial for classification than using the original high dimensional patch features which

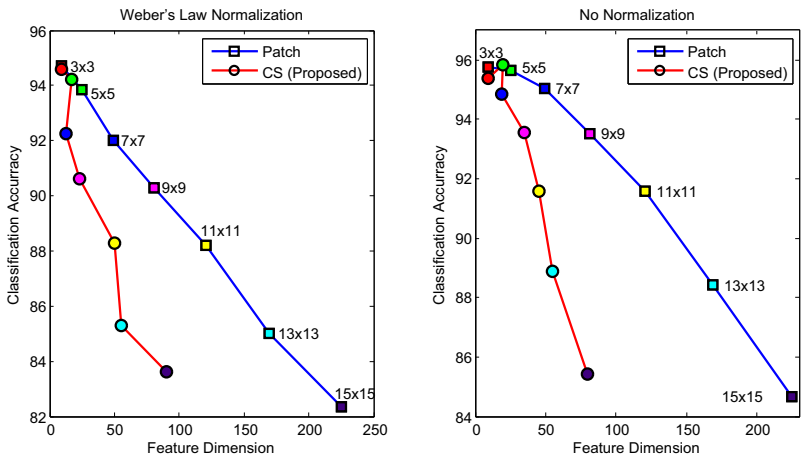

Figure 5. Comparison classification results on $\mathcal{D}^c$ as a function of patch size for the proposed approach and the Patch method.

cause difficulties in the clustering algorithm followed, or relying on pre-selected filter banks which usually result in loss of information when projecting downwards. A classifier which is able to preserve all the information from all the pixel values in a small number of measurements is superior. The LBP method performs significantly worse than any other method here (see Table 1). To assess statistical significance, all the results are averaged over tens of random partitions of the training and testing set.

To sum up, three points are notable in these results. First, the CS method outperforms the Patch classifier, the MR8 filter bank and the LBP method. This is a clear indicator that CS matrix preserves the salient information contained in the local patch (as predicted by the CS theory) and performing classification in the compressed domain is not a disadvantage. Second, the CS method does better than the Patch method, but at a much lower-dimensional feature space. That means not only does the use of low dimensional features reduce storage requirements and computation times, an improvement in classification rate is also achieved. Third, the CS method significantly outperforming the MR8 filters further shows the power of the CS method. Because both of them use linear projections to extract information from the local patch, but MR8 can't extract such good features as random features.

Furthermore, the results for the proposed method using very small patch size 3×3 are comparable with the Patch method. These results are slightly lower than the classification rate obtained by using the patch method, most likely because for the small patches, its small dimensionality in theory leaves little space for CS to improve.

Table 1. Classification results for comparison of the proposed method and LBP on dataset $\mathcal{D}^C$.

	3×3 (%)	5×5 (%)	7×7 (%)	9×9 (%)	11×11 (%)
CS (10)	95.07	96.70	96.80	96.91	97.19
Scale	1-scale	2-scale	3-scale	4-scale	5-scale
LBP	81.46	91.65	94.06	94.61	95.72

Now we turn to discuss classification results using different normalizations (defined in equation (5) and (6) in Section 4.2) for feature vectors. The reason for testing different types of normalization for feature vectors is that they lead to different classification performance. For the No-Normalization case and the Unit-Norm normalization case, the maximum patch size is restricted to 11×11 for the Patch method because of computational expense. These results indicates that firstly, the proposed approach outperforms the Patch method in all three normalization cases; secondly, for the proposed method, classification difference caused by different normalization is insignificant; thirdly, if computation time is a factor in consideration, no normalization is a good choice.

Finally, Table 2 presents the best performance achieved by each method in comparison. The proposed method gives the highest classification accuracy of 98.43% here, a result even higher than the best of Patch-MRF in [6], which is highly storage and computational time demanding.

Figure 5 shows the classification accuracy of the proposed method and the patch method on dataset $\mathcal{D}^c$ for

Table 2. Comparison of the best classification performance on dataset $\mathcal{D}^C$: Results are reported with the same experimental setup. The "Best" results for MR8 and Patch-MRF are reported in [6].

Method	LBP	MR8	Patch-MRF	CS
Best (%)	95.72	97.43	98.03	98.43

a varying patch sizes.We can observe that the proposed method performs very similarly to the patch method, but at a much lower dimension feature space. What should also be noticed is that the classification performance goes down as the patch size is increased in this small and simple dataset. This is different from the CUReT large dataset. This may be due there are not so many complex texture patterns due to the small intra-class variation. Thus we can not take advantage of a wide range of exemplars to learn intra-class variation. Furthermore, the results of the proposed approach is much better than those of LBP reported in [19] on the same dataset.

5 Conclusion and Future Work

In this paper, we have described a classification method based on representing textures as a small set of compressed sensing measurements of local texture patches. We have shown that CS measurements of the local patches can be effectively used in texture classification problem. The proposed method has been shown to achieve the state-of-art classification performance and to be at least as good as the one based on the original local patch, but with significant reductions of computational complexity and storage. We have shown that when the number of CS features is sufficiently large (in our experiments, approximately 1/3 of the dimension of the original patch) to preserve all the information contained in local patch, the difference in classification performance due to the further increase of the number of features is negligible.

There are significant distinctions between our approach and previous studies in texture classification with four main contributions:

1. *Random features*: We demonstrated the effectiveness of random features for texture classification;

2. *Low dimensional feature space*: The proposed approach facilitates a very straightforward and efficient tradeoff between the patch method, which is high dimensional, and the common filter-bank based method, which also requires large support regions and further complex post processing to get better features such as MR8.

3. *A flexible approach to trade off between computational complexity and performance.*

4. *Universality*: We collected the features for texture classification without assuming any prior information about the texture images.

The promising results of this paper motivates a further examining of CS-based texture classification, such as the use of a more sophisticated classifier like SVM, which may, in some cases, provide enhanced classification performance than the nearest neighbor classifier used in the current study, as in [22]. Furthermore, the proposed approach can also be embedded into the *signature/EMD* framework as currently being investigated in the texture analysis community, which is considered to offer some advantages over the *histograms*/χ^2 distance framework [1] [2]. Another issue requiring further study is to extend the proposed framework to the pixel-level classification or texture segmentation problem which is a little different from the image-level classification problem considered in this paper.

References

[1] S. Lazebnik, C. Schmid, and J. Ponce, A Sparse Texture Representation Using Local Affine Regions, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(8):1265-1278, August, 2005.

[2] J. Zhang, M. Marszalek, S. Lazebnik and C. Schmid, Local Features and Kernels for Classification of Texture and Object Categories: A Comprehensive Study, *International Journal of Computer Vision*, 73(2):213-238, 2007.

[3] T. Leung and J. Malik, Representing and Recognizing the Visual Appearance of Materials Using Three-Dimensional Textons, *International Journal of Computer Vision*, 43(1):29-44, 2001.

[4] M. Varma and A. Zisserman, A Statistical Approach to Texture Classification from Single Images, *International Journal of Computer Vision*, 62(1-2):61-81, 2005.

[5] O. G. Cula and K. J. Dana, 3D Texture Recognition Using Bidirectional Feature Histograms, *International Journal of Computer Vision*, vol. 59, no. 1, pp. 33-60, 2004.

[6] M. Varma and A. Zisserman, A Statistical Approach to Material Classification Using Image Patches, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(11):2032-2047, November, 2009.

[7] E. J. Candès and T. Tao, Near-Optimal Signal Recovery From Random Projections: Universal Encoding Stratigies? *IEEE Trans. Information Theory*, 52(12):5406-5425, December, 2006.

[8] D. L. Donoho, Compressed sensing, IEEE Trans. Information Theory, 52(4):1289-1306, April, 2006.

[9] E. J. Candès and T. Tao, Decoding by Linear Programming, *IEEE Trans. Information Theory*, 51(12):4203-4215, December, 2005.

[10] G. Peyré, Sparse Modeling of Textures, *Journal of Mathematical Imaging and Vision*, 34(1):17-31, 2009.

[11] J. Mairal, F. Bach, J. Ponce, Guillermo Sapiro, and A. Zisserman. Discriminative Learned Dictionaries for Local Image Analysis, *In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[12] J. M. Duarte-Carvajalino and G. Sapiro, Learning to Sense Sparse Signals: Simultaneous Sensing Matrix and Sparsifying Dictionary Optimization, *IEEE Trans. Image Processing*, 18(7):1935-1408, July 2009.

[13] J. Romberg. Imaging via Compressive Sampling, *IEEE Signal Processing Magazine*, 25(2):14-20, 2008.

[14] J. Wright, A. Yang, A. Ganesh, S. S. Sastry and Y. Ma Robust Face Recognition via Sparse Representation, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 31(2):210-217, February, 2009.

[15] J. Haupt, R. Castro, R. Nowak, G. Fudge, A. Yeh, Compressive Sampling for Signal Classification, *in Fortieth Asilomar Conference on Signals, Systems and Computers*, (Pacific Grove, CA), October, 2006.

[16] J. M. Geusebroek, A. W. M. Smeulders, and J. V. Weijer, Fast Anisotropic Gauss Filtering, *IEEE Trans. Image Processing*, 12(8):938-943, August, 2003.

[17] T. Ojala, M. Pietikainen, and T. Maenpaa, Multiresolution Gray-Scale and Rotation Invariant Texture Classification with Local Binary Patterns, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(7):971-987, 2002.

[18] M. Pietikäinen, T. Nurmela, T. Mäenpää, and M. Turtinen, View-based Recognition of Real-World Textures, *Pattern Recognition*, 37(2):313-323, 2004.

[19] S. Liao, Max W. K. Law, and Albert C. S. Chung, Dominant Local Binary Patterns for Texture Classification, *IEEE Trans. on Image Processing*, 18(5):1107-1118, May 2009.

[20] `http://www.robots.ox.ac.uk/~vgg/research/texclass/data/curetcol.zip`.

[21] `http://www1.cs.columbia.edu/CAVE/software/curet/`

[22] K. I. Kim, K. Jung, S. H. Park, and H. J. Kim, Support Vector Machines for Texture Classification, *IEEE Trans. Pattern Analysis and Machine Intelligence*, 24(11):1542-1550, November 2002.