TEXTURE CLASSIFICATION USING SPARSE K-SVD TEXTON DICTIONARIES

Muhammad Rushdi and Jeffrey Ho
Computer and Information Science and Engineering, University of Florida, Gainesville, U.S.A.

Keywords: Texture classification, Texton, Sparse representation, Image dictionary.

Abstract: This paper addresses the problem of texture classification under unknown viewpoint and illumination variations. We propose an approach that combines sparse K-SVD and texton-based representations. Starting from an analytic or data-driven base dictionary, a sparse dictionary is iteratively estimated from the texture data using the doubly-sparse K-SVD algorithm. Then, for each texture image, K-SVD representations of pixel neighbourhoods are computed and used to assign the pixels to textons. Hence, the texture image is represented by the histogram of its texton map. Finally, a test image is classified by finding the closest texton histogram using the chi-squared distance. Initial experiments on the CUReT database show high classification rates that compare well with Varma-Zisserman MRF results.

1 INTRODUCTION

The problem of classifying texture images under unknown viewpoint and illumination variations is a challenging task. The difficulty of the problem stems from several facts. Firstly, images of the same material with large variations of the pose, illumination, or scale may appear so different even for a human observer. Secondly, techniques that are accurate, adaptable and fast need to be developed to predict correctly the classes of new texture images in a reasonable amount of time.

Indeed, a lot of work has been done in the area of texture classification (Davies, 2008), (Varma and Zisserman, 2009), (Liu et al., 2009), (Zhao and Pietikainen, 2006), (Leung and Malik, 2001), (Cula and Dana, 2001). Leung and Peterson (Leung and Peterson, 1992) used moment-invariant and log-polar features to classify texture. Kang (Kang and Nagahashi, 2005) developed a framework for scale-invariant texture analysis using multi-scale local autocorrelation features. Dana *et al* created the CUReT database (Dana et al., 1999) which contains texture images for 61 categories where each category is represented by 205 images of different viewing and illumination conditions. Varma and Zisserman developed several texture classifiers based on filter banks (Varma and Zisserman, 2002) and image patch exemplars (Varma and Zisserman, 2009). Hayman *et al* (Hayman and Eklundh, 2004) created the KTH-TIPS texture database (Fritz and Eklundh, 2004) that samples texture at multiple illumination, poses and scales. As well, they devised an SVM-based approach to classify texture.

Our contribution in this paper is to introduce a novel texture classification algorithm that combines the sparse K-SVD representation with texton-based systems. We give some background materials, explain our method, then report the results of our experiments.

2 BACKGROUND

2.1 Sparse K-SVD Algorithm

Elad *et al* (Rubinstein et al., 2010) define a model for sparse signal representation, the *sparse dictionary model*, where the signal dictionary $\mathbf{D}$ is decomposed into a pre-specified base dictionary Φ and a sparse dictionary $\mathbf{A}$

$$\mathbf{D} = \Phi \mathbf{A}. \tag{1}$$

In this model, let the sparse signal representation γ has a maximum of t non-zero elements. As well, each column of $\mathbf{A}$ is normalized and has a maximum of p non-zero elements. To train a sparse dictionary, we need to approximately solve the optimization problem

Rushdi M. and Ho J..
TEXTURE CLASSIFICATION USING SPARSE K-SVD TEXTON DICTIONARIES.
DOI: 10.5220/0003376101870193
In *Proceedings of the International Conference on Computer Vision Theory and Applications* (VISAPP-2011), pages 187-193
ISBN: 978-989-8425-47-8

$$\begin{aligned} &\underset{\mathbf{A},\Gamma}{\text{minimize}} \quad \|\mathbf{X}-\Phi\mathbf{A}\Gamma\|_F^2 \\ &\text{subject to} \quad \begin{cases} \forall i \quad \|\Gamma_i\|_0^0 \le t \\ \forall j \quad \|\mathbf{a}_j\|_0^0 \le p, \quad \|\Phi\mathbf{a}_j\|_2 = 1. \end{cases} \end{aligned} \tag{2}$$

In this expression, the columns of Γ are the sparse K-SVD representations of the corresponding columns of the dataset $\mathbf{X}$, and the function $\|.\|_0^0$ counts the non-zero entries of a vector. Solving this problem is carried out by alternating sparse-coding and dictionary update steps for a fixed number of iterations. The sparse-coding step can be efficiently implemented using orthogonal matching pursuits (OMP) (Davis et al., 1997). The sparse dictionary model strikes a balance between complexity (via the choice of the base dictionary Φ) and adaptability (via the training of the sparse dictionary $\mathbf{A}$). As well, training the sparse dictionary is less time-consuming, less prone to noise and instability, and more computationally efficient than that of explicit dictionaries. This what makes this model appealing for pattern recognition tasks.

2.2 Varma-Zisserman Texture Classifier

Varma and Zisserman introduced a texture classifier based on image patch exemplars (Varma and Zisserman, 2009). This approach can be summarized as follows. Firstly, all images are made zero-mean and unit-variance. Secondly, image patches of $N \times N$ window size are taken and reordered in N^2-dimensional feature space. Thirdly, image patches are contrast normalized using Weber's law

$$\mathbf{F}(\mathbf{x}) \leftarrow \mathbf{F}(\mathbf{x})[log(1+L(\mathbf{x})/0.03)]/L(\mathbf{x}) \tag{3}$$

where $L(\mathbf{x}) = \|\mathbf{F}(\mathbf{x})\|_2$ is the magnitude of the patch vector at that pixel $\mathbf{x}$. Fourthly, all of the image patches from the selected training images in a texture class are aggregated and clustered using the k-means algorithm. The set of the cluster centres from all classes form the *texton dictionary*. Fifthly, training (and testing) images are modelled by the histogram of texton frequencies. Finally, novel image classification is achieved by nearest-neighbour matching using the χ^2 statistic. This classifier is known as the *joint classifier*. One important variant of the joint classifier is the *MRF classifier* which explicitly models the joint distribution of the central pixels and their neighbours. Refer to (Varma and Zisserman, 2009) for further details.

3 SPARSE K-SVD TEXTON-BASED TEXTURE CLASSIFIER

Our goal is to build a texture classification system using a combination of sparse-coding and texton techniques. A cross-functional flowchart of the system is shown in Figure 1. The flowchart consists of the following stages:

3.1 Parameter Setup

Several parameters need to be specified for the sparse K-SVD algorithm. These parameters include the block (or window) size, the base dictionary size and model, the sparse dictionary size, the sparsity of the sparse dictionary with respect to the base dictionary, the sparsity of the image patch K-SVD representation with respect to the sparse dictionary, the sparse-coding criteria (sparsity-based or error-based), the number of iterations for estimating the sparse dictionary. Refer to (Rubinstein et al., 2010) for further information on these parameters. As well, for classification with textons, we need to specify the window size (which must be the same as the K-SVD block size), the size of the texton dictionary, and the number bins for the MRF variant of the Varma-Zisserman classifier.

3.2 Building the Base Dictionary

In this work, we focus on separable base dictionaries since they have compact representation, sub-quadratic implementation, and memory-efficient computation. A separable dictionary is the Kronecker product of several one-dimensional dictionaries. We consider here two types of these dictionaries: analytic and data-driven dictionaries. Analytic dictionaries are defined using standard signal bases like the DCT, overcomplete DCT, and wavelet dictionaries. Data-driven dictionaries are generated from the given image data using some clustering schemes such as k-means clustering, or median-based clustering. Specifically, for each category $c, 1 \le c \le C$ where C is the number of texture classes, we sample image blocks from all of training images, convert the blocks into columns, apply Weber's law to contrast-normalize the columns (Varma and Zisserman, 2009), and then we apply the clustering scheme to find the class-specific base dictionary Φ_c. Then, we concatenate the class-specific base dictionaries to get the overall base dictionary Φ

$$\Phi = [\Phi_1|\Phi_2|...|\Phi_C]. \tag{4}$$

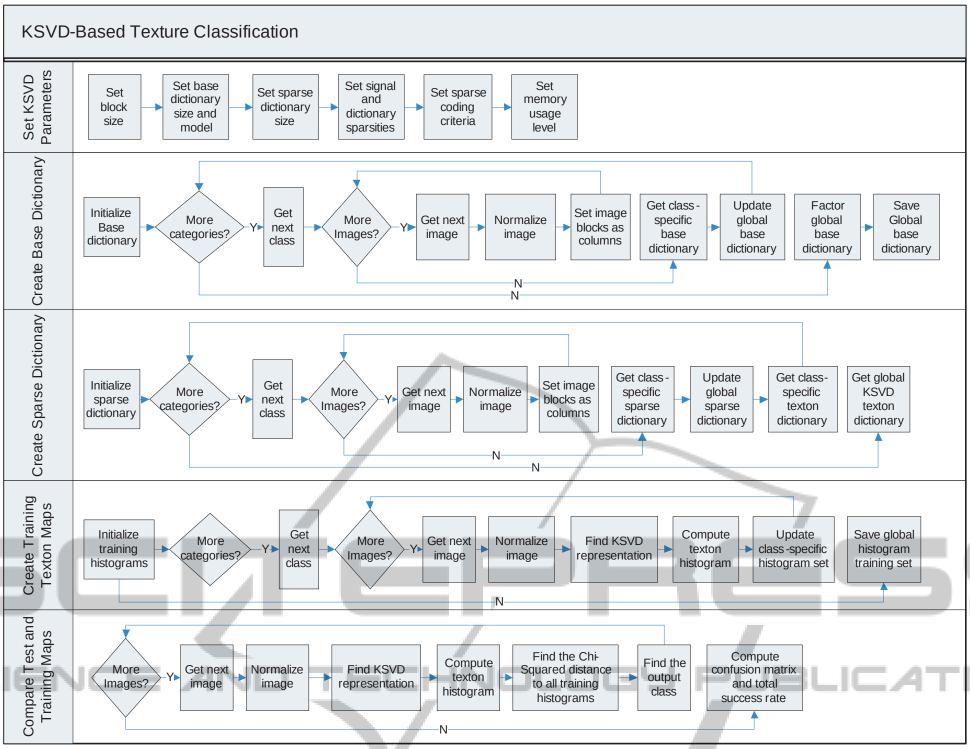

Figure 1: Cross-functional block diagram of the Sparse K-SVD Texton-Based Texture Classifier.

In Section 4, we give more details on creating data-driven dictionaries and subsequently converting them into separable forms.

3.3 Building the Sparse Dictionary

Once the separable base dictionary is generated (analytically or from data), we move on to estimating a sparse dictionary of the texture data with some sparsity level p. To guarantee that each texture category is fairly represented in the sparse dictionary, we estimate a sub-dictionary for each category then join the sub-dictionaries to from the whole sparse dictionary. Specifically, for each category $c, 1 \leq c \leq C$ where C is the number of texture classes, we estimate a solution of the optimization problem

$$\begin{aligned} &\underset{\mathbf{A}_c,\Gamma_c}{\text{minimize}} && \|\mathbf{X}_c - \Phi\mathbf{A}_c\Gamma_c\|_F^2 \\ &\text{subject to} && \begin{cases} \forall i & \|\Gamma_{c_i}\|_0^0 \leq t \\ \forall j & \|\mathbf{a}_{c_j}\|_0^0 \leq p, \quad \|\Phi\mathbf{a}_{c_j}\|_2 = 1. \end{cases} \end{aligned} \tag{5}$$

then we concatenate the class-specific dictionaries to get the overall sparse dictionary $\mathbf{A}$

$$\mathbf{A} = [\mathbf{A}_1|\mathbf{A}_2|\ldots|\mathbf{A}_C]. \tag{6}$$

3.4 Building the Texton Dictionary

We generate a texton dictionary of the texture data as follows. Firstly, for each category $c, 1 \leq c \leq C$ where C is the number of texture classes, we apply a clustering algorithm to find the centroids of the texture class. Secondly, we join all of the class-specific centroids to form the texton dictionary $\mathbf{X}_{textons}$. Thirdly, we apply sparse-coding techniques (e.g. orthogonal matching pursuit) to $\mathbf{X}_{textons}$ to find the corresponding K-SVD representation $\Gamma_{textons}$

$$\begin{aligned} &\Gamma_{textons} = \underset{\Gamma}{\text{argmin}} && \|\mathbf{X}_{textons} - \Phi\mathbf{A}\Gamma\|_F^2 \\ &\text{subject to} && \forall i \|\Gamma_i\|_0^0 \leq t. \end{aligned} \tag{7}$$

Note that the size of the texton dictionary should be small to get reasonable processing times for test texture images.

3.5 Computing the Image Texton Histograms

For each training or test image, we follow a texton-based approach to find the texton histogram of the image. Firstly, we convert each pixel and its neighbourhood into a column. Secondly, we sparse-code the columnized neighbourhood $\mathbf{x}_{mn}$ by solving the minimization problem

$$\begin{aligned} \gamma_{mn} = \underset{\gamma}{\operatorname{argmin}} \quad & \|\mathbf{x}_{mn} - \mathbf{\Phi A}\gamma\|_2^2 \\ \text{subject to} \quad & \|\gamma\|_0^0 \leq t. \end{aligned} \tag{8}$$

Thirdly, we assign each pixel to the closest texton in the texton dictionary. Indeed, the Euclidean distance is measured between the K-SVD representation γ_{mn} of the pixel neighbourhood and each element of the K-SVD representation of the texton dictionary $\Gamma_{textons}$. This step produces the *texton map* of the image. Finally, we compute the frequency histogram of the texton map with respect to the textons. This produces a vector with a length equal to the size of texton dictionary. Alternatively, a texture image may be represented by an MRF model (Varma and Zisserman, 2009).

3.6 Classifying Test Images

Given a test texture image, its texton frequency histogram is computed as described above. Then the distances between this histogram and all of the training image texton histograms are measured using the χ^2 statistic where

$$\chi^2(\mathbf{x},\mathbf{y}) = \frac{1}{2}\sum_i \frac{(x_i - y_i)^2}{x_i + y_i}. \tag{9}$$

4 CONSTRUCTION OF DATA-DRIVEN SEPARABLE IMAGE DICTIONARIES

As we mentioned in Section 3.2, a separable image dictionary may be found from data by applying clustering methods then approximating the resulting dictionary by a Kronecker product of 1D dictionary components. One advantage of this base dictionary construction approach is that the resulting dictionary closely represents the population of texture classes. This is in contrast to an analytic dictionary (e.g. DCT or wavelets) that might be too generic to represent the texture data set. A second advantage can be realized if we use clustering schemes that represent each cluster by one of its members (e.g. k-Medoids algorithms). In this case, representations of image patches and sparse dictionary atoms will have a direct physical interpretation in terms of actual points in the texture data. Now, we give further implementation details of this base dictionary construction approach.

4.1 Clustering Approaches for Image Dictionaries

There are numerous clustering approaches in the literature (Duda et al., 2001), (Theodoridis and Koutroumbas, 2009). We will focus here on two of them: k-means, and k-medoids algorithms. The *k-means* algorithm is a popular and well-known algorithm. It aims to move the cluster representatives $\theta_j, j = 1,\ldots,M$ (where M is the number of clusters) into regions that are dense in points of the dataset $\mathbf{X}$ by minimizing the sum of squared Euclidean distances between vectors $\mathbf{x}_i, i = 1,\ldots,N$ (where N is the number of data points) and cluster means θ_j. The k-means algorithm is computationally simple and works well for large datasets. However, the k-means algorithm is sensitive to outliers and noise. As well, k-means is not suitable for data with nominal or finite discrete-valued domains.

Another family of clustering algorithms is the *k-medoids algorithms*. In the k-medoids methods, each cluster is represented by a vector selected among the elements of $\mathbf{X}$, which is usually referred to as the *medoid*. These methods overcome the problems of k-means. First, k-medoids methods are applicable to data sets from both continuous and discrete domains. Second, k-medoids methods are less sensitive to outliers and noise. However, these methods are more computationally demanding than k-means. In particular, a basic version of the k-medoids methods, the *Partitioning Around Medoids (PAM) algorithm*, is prohibitively slow and memory-intensive for clustering of large datasets. To avoid this problem, another variant of the k-medoids methods, the *Clustering LARge Applications (CLARA)* algorithm, exploits the idea of randomized sampling in the following way. CLARA draws randomly a sample $\mathbf{X}^{'}$ of size $N^{'}$ from the entire data set $\mathbf{X}$ and then determines the set $\Theta^{'}$ that best represents $\mathbf{X}^{'}$ using the PAM algorithm. The rationale behind CLARA is that if the sample $\mathbf{X}^{'}$ is statistically representative of $\mathbf{X}$, then the set $\Theta^{'}$ will be a satisfactory approximation of the set Θ of the medoids that would result if the PAM algorithm was run on the whole dataset $\mathbf{X}$ (Theodoridis and Koutroumbas, 2009).

4.2 Approximation of Separable Dictionaries with Kronecker Products

Once a data-driven dictionary is constructed by clustering, we need to convert it into a separable form. This can be achieved by approximating the dictionary as the Kronecker product of 1D dictionaries. This problem can be stated as follows. Let the base dictionary $\Phi \in \mathbb{R}^{m\times n}$ be an m-by-n matrix with $m = m_1 m_2$ and $n = n_1 n_2$. We need to solve the minimization problem

$$\begin{aligned} &\underset{\Phi_0,\Phi_1}{\text{minimize}} \quad \|\Phi - \Phi_0 \otimes \Phi_1\|_F^2 \\ &\text{subject to} \quad \begin{cases} \Phi_0 \in \mathbb{R}^{m_1\times n_1} \\ \Phi_1 \in \mathbb{R}^{m_2\times n_2} \end{cases} . \end{aligned} \tag{10}$$

Loan and Pitsianis (Loan and Pitsianis, 1993) approximately solve this problem as follows. Firstly, it is shown that this problem is equivalent to

$$\begin{aligned} &\underset{\Phi_0,\Phi_1}{\text{minimize}} \quad \|\mathcal{R}(\Phi) - vec(\Phi_0)vec(\Phi_1)^T\|_F^2 \\ &\text{subject to} \quad \begin{cases} \Phi_0 \in \mathbb{R}^{m_1\times n_1} \\ \Phi_1 \in \mathbb{R}^{m_2\times n_2} \end{cases} \end{aligned} \tag{11}$$

where $\mathcal{R}(\Phi)$ is the column rearrangement of Φ (relative to the blocking parameters m_1, m_2, n_1, and n_2) and $vec(\Phi_0), vec(\Phi_1)$ are columnized arrangements of Φ_0 and Φ_1, respectively.
Secondly, if $\tilde{\Phi} = \mathcal{R}(\Phi)$ has singular value decomposition

$$U^T \tilde{\Phi} V = \Sigma = diag(\sigma_i) \tag{12}$$

where σ_1 is the largest singular value, and $U(:,1), V(:,1)$ are the corresponding singular vectors, then the matrices $\Phi_0 \in \mathbb{R}^{m_1\times n_1}, \Phi_1 \in \mathbb{R}^{m_2\times n_2}$ defined by

$$\begin{aligned} vec(\Phi_0) &= \sigma_1 U(:,1) \\ vec(\Phi_1) &= V(:,1) \end{aligned} \tag{13}$$

minimize $\|\Phi - \Phi_0 \otimes \Phi_1\|_F^2$ (Golub and Loan, 1989).

5 EXPERIMENTS AND RESULTS

5.1 Experimental Texture Data

We performed our texture classification experiments on the Columbia-Utrecht Reflectance and Transmission (CUReT) texture image database (Dana et al., 1999). The database contains images of 61 texture materials. Each material has 205 images taken under different viewing and illumination conditions. In (Varma and Zisserman, 2009), Varma and Zisserman picked a subset of 92 images for each class. For this subset, a sufficiently large portion of the texture is visible across all materials. A central 200×200 region was cropped from each of the selected images and the remaining background was discarded. The selected regions where converted to gray scale, then normalized to zero-mean and unit-variance. This cropped CUReT database (Varma and Zisserman, 2009) has a total of 61×92 images. We use this cropped CUReT database in our experiments. For each class, 46 images are randomly chosen for training and the remaining 46 are used for testing.

5.2 Implementation Details

As we mentioned in Section 3, several parameters needs to be experimentally set in our system. Based on experiments, we found reasonable operating ranges for some parameters as follows. The size of the base dictionary was set between 80 and 400 depending on the sparse dictionary size. Much larger base dictionary adversely affected the performance. The sparsity of the sparse dictionary was set to values between 7 and 10 depending on the sizes of the base and sparse dictionaries. Sparsity of the K-SVD representation vectors was set between 11 and 15 depending on the size of the sparse dictionary. The number of training iterations was fixed at 30. A block size of 8×8 was chosen as it gave the best classification rate. A sparsity-based sparse-coding (SC) was found to give better results than error-based criteria. For the texton dictionary, 10 textons per class were computed giving a total of $61 \times 10 = 610$ textons. Increasing the texton beyond this increased the complexity with negligible improvement in performance. We tested both the joint and the MRF classifier variants of the Varma-Zisserman classifiers. We eventually chose to focus on the MRF variant as it consistently exhibited better performance. The number of MRF bins was set to 90. For the CLARA clustering method, 10 random samples were drawn from the data to find the cluster medoids. The size of the drawn sample was set to $N' = 40 + 2M$ where M is the number of base dictionary clusters (set between 4 and 7 per class).

5.3 Base Dictionary Model Selection

For the K-SVD base dictionary model, we tried standard DCT dictionaries , and data-driven dictionaries generated by the CLARA clustering method (as described in Section 4). As we can see from Table 1, using a DCT base dictionary gives better results than the data-driven CLARA base dictionary. Our interpretation is that the DCT base dictionary covers a wider range of image structures than the clustering-based dictionary that we used. This can be verified by visualizing both dictionaries and their respective sparse dictionaries. Figures 2, 3 show sample atoms of the base dictionaries and the corresponding sparse dictionaries for best DCT and CLARA results of Table 1. It is clearly noticeable that the DCT has more image structures than the CLARA dictionary. This

Table 1: Comparison of the classification performance of the sparse K-SVD texture classifier for different choices of base dictionary construction method (DCT versus CLARA) and different values of the per-class sparse dictionary size. Given the 61 CUReT classes, the total sparse dictionary sizes are $2 \times 61 = 122$ and $9 \times 61 = 549$, respectively.

Per-class **A** size / Φ Method	DCT	CLARA
2	92.19%	91.80%
9	94.62%	92.48%

in turn is reflected on the richness of the associated sparse dictionaries. We still think that data-driven base dictionaries constitute a good choice if the underlying data is well-chosen to cover the texture data variability. We plan to examine this in future work.

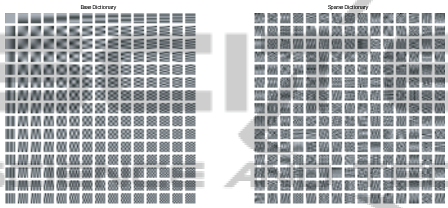

Figure 2: Sample atoms from the DCT base dictionary and its corresponding sparse dictionary for the best DCT-based result of Table 1.

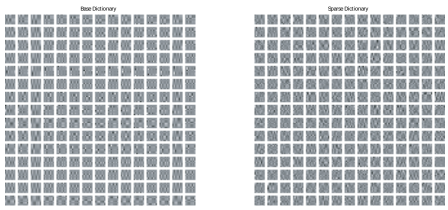

Figure 3: Sample atoms from the CLARA base dictionary and its corresponding sparse dictionary for the best CLARA-based result of Table 1.

5.4 Spare Dictionary Sizing

The sparse dictionary **A** size is particularly important since it widely affects the running time during dictionary learning, training, and testing. As well, this dictionary size clearly affects the classification rate (as shown in Figure 4). The best classification rate of 95.12% is achieved at a dictionary size of $15 \times 61 = 915$. Increasing the sparse dictionary size beyond that would give better results but at a high computational cost.

5.5 Comparison to other Work

Table 2 compares our method with the Varma-Zisserman patch-based classifiers. Despite that our method is slightly sub-optimal, it has some potential advantages that are mainly inherited from the sparse K-SVD structure. First, complexity can be well modelled through the choice of an analytic or a data-driven base dictionary. Second, adaptability to data is achieved through training of the sparse dictionary component. Third, the sparse representation reduces the space and time computational complexities. Finally, sparse signals and dictionary atoms can be ascribed direct interpretations in terms of few base dictionary elements. We explore these ideas further in future work.

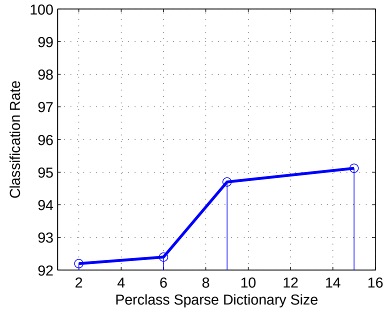

Figure 4: Effect of the per-class sparse dictionary size on the overall classification rate. The best classification rate of 95.12% is achieved at a dictionary size of $15 \times 61 = 915$.

Table 2: Comparison of the classification performance of our method with the Varma-Zisserman patch-based classifiers. Despite that our method is slightly sub-optimal it has other advantages such as design flexibility, immunity to outliers, sparse representation, and physical interpretation.

Ours	VZ Joint	VZ Neighbour	VZ MRF
95.12%	96.16%	96.08 %	97.47%

6 CONCLUSIONS AND FUTURE WORK

We introduced a novel texture classification algorithm that combines the sparse K-SVD representation with texton-based systems. Our system carries the benefits of the sparse K-SVD algorithm into the texture classification problem: sparse representation, adaptability to data, and efficient computation. In particular, we explored the idea of using data-driven base dictionaries instead of standard analytical ones to represent more specialized texture data. Results of the simulations and experiments show that our system closely matches the state-of-the-art algorithms. In the future, we will focus on performing better design of the base and sparse dictionaries. As well, we pursue methods for optimal parameter selection.

REFERENCES

Cula, O. and Dana, K. (2001). Compact representation of bidirectional texture functions. volume 1, pages I–1041 – I–1047 vol.1.

Dana, K., Van-Ginneken, B., Nayar, S., and Koenderink, J. (1999). Reflectance and Texture of Real World Surfaces. *ACM Transactions on Graphics (TOG)*, 18(1):1–34.

Davies, E. (2008). *Handbook of Texture Analysis*, chapter Introduction to Texture Analysis. Imperial College Press.

Davis, G., Mallat, S., and Avellaneda, M. (1997). Greedy adaptive approximation. *J. Constr. Approx.*, 13:57–98.

Duda, R., Hart, P., and Stork, D. (2001). *Pattern Classification*. Wiley.

Fritz, M., H. L. C. B. and Eklundh, J.-O. (2004). The kth-tips database. available at www.nada.kth.se/cvap/databases/kth-tips.

Golub, G. and Loan, C. V. (1989). *Matrix Computations*. John Hopkins Press.

Hayman, L., C. B. and Eklundh, J.-O. (2004). On the significance of real-world conditions for material classification. In *8th European Conference on Computer Vision*, pages 253–266.

Kang, Y., M. K. and Nagahashi, H. (2005). *Scale Space and PDE Methods in Computer Vision*, volume 3459, chapter Scale Invariant Texture Analysis Using Multi-scale Local Autocorrelation Features, pages 363–373. Springer.

Leung, M. and Peterson, A. (1992). Scale and rotation invariant texture classification. In *The Twenty-Sixth Asilomar Conference on Signals, Systems and Computers*, volume 1, pages 461–465.

Leung, T. and Malik, J. (2001). Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43.

Liu, G., Lin, Z., and Yu, Y. (2009). Radon representation-based feature descriptor for texture classification. *Image Processing, IEEE Transactions on*, 18(5):921 – 928.

Loan, C. V. and Pitsianis, N. (1993). Approximation with kronecker products. In *Linear Algebra for Large Scale and Real Time Applications*, pages 293–314. Kluwer Publications.

Rubinstein, R., Zibulevsky, M., and Elad, M. (2010). Double sparsity: Learning sparse dictionaries for sparse signal approximation. *Signal Processing, IEEE Transactions on*, 58(3):1553 –1564.

Theodoridis, S. and Koutroumbas, K. (2009). *Pattern Recognition, 4th Edition*. Academic Press.

Varma, M. and Zisserman, A. (2002). Classifying images of materials: Achieving viewpoint and illumination independence.

Varma, M. and Zisserman, A. (2009). A statistical approach to material classification using image patch exemplars. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(11):2032–2047.

Zhao, G. and Pietikainen, M. (2006). Local binary pattern descriptors for dynamic texture recognition. volume 2, pages 211 –214.