Learning Structural Element Patch Models With Hierarchical Palettes

Jeroen Chua,
University of Toronto
jeroen@psi.utoronto.ca

Inmar Givoni
University of Toronto
inmar@psi.utoronto.ca

Ryan Adams
Harvard University
rpa@seas.harvard.edu

Brendan Frey
University of Toronto
frey@psi.utoronto.ca

Abstract

Image patches can be factorized into 'shapelets' that describe segmentation patterns called structural elements (stels), and palettes that describe how to paint the shapelets. We introduce local palettes for patches, global palettes for entire images and universal palettes for image collections. Using a learned shapelet library, patches from a test image can be analyzed using a variational technique to produce an image descriptor that represents local shapes and colors separately. We show that the shapelet model performs better than SIFT, Gist and the standard stel method on Caltech28 and is very competitive with other methods on Caltech101.

1. Introduction

Separating the shapes of parts comprising objects from image-specific appearance details, such as color and low-level texture, has long since been recognized as an important problem [12]. Here, we describe a solution that combines two previously described approaches: stel models [8] and patch-based descriptors [6, 11, 21].

Stel models describe images using a probabilistic image segmentation that is shared across images, along with an image-specific palette of colors that is used to paint each region in an image. These are learned using an EM algorithm [7]. If objects in the training images are well-aligned up to affine transformations, the learned regions tend to correspond to object parts. However, when objects undergo extreme variations in deformation and articulation, the representation used for the standard stel model and its extensions (*cf* [16, 17]) requires a prohibitively large number of appearance configurations. Additionally, most prior work involves learning a separate model for each class, which may be impractical for a large number of classes. To address these issues, we combine stel models with a quite different technique in the vision community, whereby each image is broken down into a collection of small patches that are analyzed to produce an image descriptor that is robust to deformations [6, 11, 21]. In contrast to the standard stel model, the descriptors we learn are not class-specific.

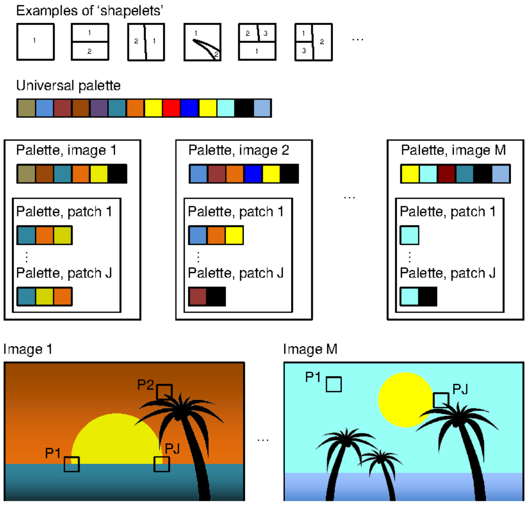

Figure 1. Shapes comprising image patches can be described using 'shapelets' (tiny stel models), with varying numbers of regions. To describe an image as a collection of shapelets, each shapelet region needs to be painted, or colored. Each image selects a subset of global colors from a universal palette, and the shapelet describing a patch within an image selects a subset of that image's global colors to paint the shapelet. In this way, the same shapelet can use different colors within an image (patches P1 and PJ in image 1) and across images (patches P2 in image 1 and PJ in image m).

Fig. 1 illustrates our method. Patches from an image are described by tiny stel models called "shapelets", along with palettes specifying how to paint the shapelet regions. In contrast to previous stel models [7, 16, 17], our model uses a hierarchical palette, whereby the palette for a patch is a subset of an intra-image global palette, which is a subset of an inter-image universal palette. Inference and learning of the model are performed via a variational method. Unlike methods that learn whole-image parts (*cf* [19, 5]), our method extracts local features. Like the features found using sparse coding [15], SIFT [11], convolutional networks [6, 21] and Boltzmann machines [18], shapelets describe local shape boundaries. However, shapelets also provide a factorized representation of local shape and color, which we show to be useful for object recognition.

2. The shapelet model

Given an unlabeled set of training images, the shapelet model attempts to explain all image patches using a library of patch shapes, or shapelets, and a hierarchical set of palettes used to color each patch in each image. The shapelets capture information concerning local shape in a patch regardless of the specific colors used, whereas the palettes capture information concerning the coloring of patches and images regardless of the patch or image structure. A universal palette accounts for the colors of all training images. Each image selects a subset of global image colors from that palette to form the image palette, and, in turn, each patch in the image chooses a subset of colors from the image palette to form the patch palette. The hierarchy of palettes enables the model to capture inter and intra-image coloring consistencies.

Using a generative modeling approach, a shapelet is a probabilistic grouping of pixels into a set of regions that reflect co-occurrence of color. Shapelets capture local image structure by modeling this co-occurrence without regard for the specific color identities. In order to explain patches of different complexity, each shapelet is allowed to contain a different number of regions. A shapelet with a single region describes a uniformly colored patch, while a shapelet with several regions can capture more complex patterns. An image patch can be described by a shapelet and a palette reflecting the color means and variances of each shapelet region. This gives rise to a flexible patch-based image representation: multiple patches can be compactly explained by a single shapelet with different palettes (Fig. 1).

Regions within a shapelet serve a similar role to stels [16, 17] in that both represent probabilistic pixel groupings based on color co-occurrence. However, stel models are often learned on whole images, and so stels often correspond to object parts (*eg*, the shirt of a person), while shapelets are learned on the patch level and correspond to shape primitives (*eg*, a quarter circle).

Given a test image and a learned library of shapelets, we infer distributions describing which shapelets best model each patch in the image, along with the palettes used to color the patches and the image. This factorized representation can be used to perform higher level vision tasks. In the sequel, we provide details of the model and the learning and inference algorithms, and demonstrate the application of the obtained shape and color factorization to the tasks of image reconstruction and object recognition.

2.1. Probabilistic graphical model for shapelets

Given M images, each containing J patches of size $N_y \times N_x = N$ pixels, our goal is to learn a library of shapelets and to infer the universal, global (per-image), and patch palettes. We denote the number of shapelets in the library by S, where each shapelet s contains R_s regions. We let the universal palette contain U colors, the global palettes contain G colors, and the patch palettes contain up to $\hat{R}$ colors, where $\hat{R} = \max_s R_s$.

A shapelet s, containing R_s regions, is represented as a collection of N discrete distributions over the R_s region indices. For each shapelet, these distributions indicate the region preference of every pixel in a patch.

To generate a set of images from our model, one first generates a set of universal colors comprised of U means in color space. Next, to generate an image, a global palette is generated by randomly selecting G color means from the universal palette and specifying their variances (note that color variances are image-specific). Then, each patch in the image picks one shapelet from the library of S shapelets. Pixels in the patch are assigned to one of the R_s regions by independently sampling from each pixel's discrete distribution, as given by the selected shapelet. Finally, each of the R_s regions is assigned to one of the G global colors by sampling from their respective distributions defined by the patch palette, and use their global color's mean and variance to color their assigned pixels. Note that multiple regions may pick the same global color; all that is required is that pixels belonging to the same region are explained by the same global color. Note that the generation of local shape (shapelets) is explicitly separated from the generation of local appearance (palettes); it is this separation that allows the shapelet model to efficiently factorize shape and appearance.

The graphical model is given in Fig. 2. $u_{mg} \in \{1...U\}$ is the universal color index being used in image m by image color g; the shapelet index being used in image m and patch j is $s_{mj} \in \{1...S\}$; the region index to which pixel n in image m and patch j, using shapelet s, is assigned is $r_{mjn} \in \{1...R_s\}$; the global color index to which pixel n in image m and patch j is assigned is $g_{mjn} \in \{1...G\}$; the observed value of pixel n in image m and patch j is denoted by $\vec{x}_{mjn} \in \mathbb{R}^H$, where H is the number of color channels (*eg*, $H = 3$ in the case of RGB-space).

We parameterize the above hidden variable distributions as follows:

$$P(u_{mg} \,|\, \vec{\beta}) = \text{Discrete}(\vec{\beta}) \tag{1}$$

$$P(s_{mj} \,|\, \vec{\theta}) = \text{Discrete}(\vec{\theta}) \tag{2}$$

$$P(r_{mjn} \,|\, s_{mj}{=}s, \vec{\pi}_{ns}) = \text{Discrete}(\vec{\pi}_{ns}) \tag{3}$$

$$P(g_{mjn} \,|\, r_{mjn}{=}r, \vec{\alpha}_{mjr}) = \text{Discrete}(\vec{\alpha}_{mjr}) \tag{4}$$

The parameter $\vec{\beta}$ controls the selection of colors from the

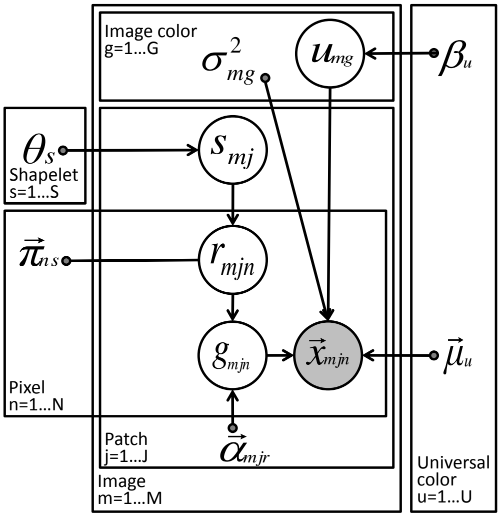

Figure 2. Graphical model for the shapelet model. An image m is generated by first selecting G image colors from a library of U universal colors. Then, each patch j in the image selects a shapelet, s_{mj}, to model its local shape, and $R_{s_{mj}}$ colors from the G image colors to model its local color. Each patch $\vec{x}_{mj}$ is then colored according to the choice of shapelet and palettes.

universal palette that form global colors. The parameter $\vec{\theta}$ controls the shapelet choice for modeling a particular patch, and is further discussed below. $\vec{\pi}_{ns}$ parameterizes, for a pixel n in shapelet s, the distribution over the R_s regions and is analogous to a distribution over stel assignments [16, 17] for patches. Similarly, $\vec{\alpha}_{mjr}$ parameterizes, for a patch j in an image m for the region r, the distribution over the G image colors, and forms the lowest level of our color hierarchy (patch palette), as shown in Fig. 1.

Recall that our framework is capable of handling shapelets with a differing number of regions through an appropriate choice of the $R_s \forall s$. The choice of shapelet, and thus of shapelet complexity, is controlled by the parameter $\vec{\theta} = \theta_1, \ldots, \theta_S$. We wish to bias our framework to prefer to use shapelets with fewer regions ('simpler' shapelets). We achieve this by setting:

$$\theta_s = \frac{\exp(-\lambda(R_s - 1)^2)}{\sum_{s=1}^{S} \exp(-\lambda(R_s - 1)^2)}, \tag{5}$$

where λ is a regularization parameter. This prior is fixed since R_s is fixed $\forall s$. Alternative forms of regularization are also possible, such as using a Dirichlet prior with S parameters. However, such a prior may require careful parameter setting to balance the use of simpler and more complex shapelets. In contrast, the regularization choice in Eq. 5 has only one tunable parameter, and in practice we find it has the desired effect of encouraging the use of shapelets with fewer regions.

For the observation model, we assume each H-dimensional pixel value is distributed according to an axis-aligned H-dimensional Gaussian:

$$P(\vec{x}_{mjn} \,|\, g_{mjn}{=}g, u_{mt}{=}u) = \prod_{h=1}^{H} \mathcal{N}(x^h_{mjn} \,|\, \mu^h_u, (\sigma^2)^h_{mg}) \tag{6}$$

where $\mathcal{N}()$ is the normal distribution, μ^h_u is the mean of the hth color channel of uth universal color, and $(\sigma^2)^h_{mg}$ is the variance of the hth color channel in the set $(\sigma^2)_{mg}$ of variances.

3. Variational inference and learning

Letting $\Phi = \{\pi, \alpha, \beta, \mu, (\sigma^2)\}$, and $\mathbf{h} = \{\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u}\}$ (the collection of all hidden variables), we perform maximum-likelihood parameter estimation of the likelihood function $P(\mathbf{x} \,|\, \vec{\theta}, \Phi) = \sum_{\mathbf{h}} P(\mathbf{x}, \mathbf{h} \,|\, \vec{\theta}, \Phi)$ with respect to Φ, keeping $\vec{\theta}$ fixed according to Eq. 5. The complete data log-likelihood is given by:

$$\begin{aligned}
&\log P(\mathbf{x}, \mathbf{h} \,|\, \vec{\theta}, \Phi) = \\
&\sum_{mg} \log P(u_{mg} \,|\, \vec{\beta}) + \sum_{mj} \log P(s_{mj} \,|\, \theta_s) \\
&+ \sum_{mjn} \log P(r_{mjn} \,|\, s_{mj}, \vec{\pi}_{ns}) + \sum_{mjn} \log P(g_{mjn} \,|\, r_{mjn}, \vec{\alpha}_{mjr}) \\
&+ \sum_{mjn} \log P(x^h_{mjn} \,|\, g_{mjn}{=}g, u_{mt}{=}u, \vec{\mu}_u, (\vec{\sigma^2})_{mg}).
\end{aligned} \tag{7}$$

To keep the notation uncluttered, we omit conditioning on the parameters below. Exact computation of the posterior distribution over hidden variables is not tractable due to explaining away, so we perform inference and learning using a variational algorithm. We use a factorized approximation to the posterior:

$$Q(\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u}) = Q(\mathbf{s}, \mathbf{r}, \mathbf{g}) Q(\mathbf{u}). \tag{8}$$

We further assume that $Q(\mathbf{s}, \mathbf{r}, \mathbf{g})$ factorizes as follows:

$$Q(\mathbf{s}, \mathbf{r}, \mathbf{g}) {=} \prod_{mj} Q(s_{mj}) \prod_{n} Q(r_{mjn} \,|\, s_{mj}) Q(g_{mjn} \,|\, r_{mjn}). \tag{9}$$

Note that this is an approximation since the true posterior does not factorize over patches, j, nor pixels, n, due to the sharing of the universal colors $\mathbf{u}$.

We are free to specify the functional forms of the Q-distributions $Q(s_{mj})$, $Q(r_{mjn} \,|\, s_{mj})$, and $Q(g_{mjn} \,|\, r_{mjn})$. A natural choice is the one that minimizes the free-energy F between $Q(\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})$ and $P(\mathbf{x}, \mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})$:

$$F {=} \sum_{\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u}} Q(\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u}) \log \frac{Q(\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})}{P(\mathbf{x}, \mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})}, \tag{10}$$

where $Q(\mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})$ decouples according to Eqs. 8 and 9. This is a natural objective to minimize as F is an upper-bound on the negative likelihood, $-P(\mathbf{x} \,|\, \vec{\theta}, \Phi)$ and so minimizing Eq. 10 is equivalent to maximizing a lower bound on the marginal likelihood of the data [13].

Consider the factor $Q(s_{mj})$. We note that keeping all other Q-distributions fixed, the distribution $Q^*(s_{mj})$ that minimizes Eq. 10 is given by $Q^*(s_{mj}) \propto \exp(\mathbb{E}_{\mathbf{h}_{\setminus s_{mj}}}[\ln P(\mathbf{x}, \mathbf{s}, \mathbf{r}, \mathbf{g}, \mathbf{u})])$, where \ indicates a set of elements except for a specific element (*ie*, $\mathbf{h}_{\setminus s_{mj}}$ is the set $\mathbf{h}$ without s_{mj}). This results in a tractable expression for $Q^*(s_{mj})$. Similarly, we can obtain tractable expressions for $Q^*(r_{mjn} \,|\, s_{mj})$ and $Q^*(g_{mjn} \,|\, r_{mjn})$, as well as $Q^*(u_{mg})$, where we have further approximated $Q(\mathbf{u})$ with $\prod_{mg} Q(u_{mg})$. More details are provided in the supplementary materials.

These variational inference updates can be used as an E-step in a variational EM framework that is guaranteed to increase a lower bound on the data likelihood. In order to perform the M-step, we take the derivatives of Eq. 10, with respect to the model parameters. For the M-step, it is useful to compute the quantity $Q(g_{mjn}) \;\forall\; mjn$:

$$Q(g_{mjn}) = \sum_{s_{mj}} \Big(Q(s_{mj}) \sum_{r_{mjn}} Q(r_{mjn} \,|\, s_{mj}) Q(g_{mjn} \,|\, r_{mjn}) \Big). \tag{11}$$

With the Q-distributions computed as above, the M-Step palette updates are:

$$\vec{\mu}_u = \frac{\sum_{mgjn} \Big(Q(u_{mg} = u) Q(g_{mjn} = g) \vec{x}_{mjn} \Big)}{\sum_{mgjn} \Big(Q(u_{mg} = u) Q(g_{mjn} = g) \Big)} \tag{12}$$

$$(\sigma^2)^h_{mg} = \frac{\sum_u \Big(Q(u_{mg} = u) \sum_{jn} Q(g_{mjn} = g)(x^h_{mjn} - \mu^h_u)^2 \Big)}{\sum_u \Big(Q(u_{mg} = u) \sum_{jn} Q(g_{mjn} = g) \Big)} \tag{13}$$

where h indexes a color channel. The palette updates resemble the mean and variance updates for learning a mixture of Gaussians, where the responsibilities are given by $Q(u_{mg} = u) Q(g_{mjn} = g)$, the datapoints are the pixels, and we also sum over all G image-level colors to remove the color hierarchy going from the U universal colors to the G image colors. The remaining updates are given in the supplementary materials.

4. Experimental results

4.1. Shapelet library

A set of shapelets learned from one of the Caltech28 trials described in the next section is shown in Fig. 3. The shapelets learned by our model vary in complexity from simple horizontal lines to more complex patterns such as pie-wedge shapes and quarter circles. Some of the shapelets also resemble Gabor-like filters, capturing patterns such as lines in different orientations and positions. Note that whereas standard coding methods would need to have separate filters to account for different combinations of intensity patterns, the shapelet model can make do with fewer filters. For example, the shapelet in the second column and first row can account for a patch with a bright strip above a dark region, but can equally well account for a dark strip above a bright region.

4.2. Image reconstruction

Since we have a generative model, our shapelet model can perform image reconstruction in addition to object classification. To perform reconstruction of a given image m, we first take a learned shapelet library, which defines the $\vec{\pi}_{ns} \forall n, s$, and using the method outlined in Sect. 2, we infer the posterior distribution over shapelets $P(s_{mj} \,|\, \vec{\theta}, \mathbf{X}_{mj}) \forall j$ ($\mathbf{X}_{mj} = \{\vec{x}_{mjn}\}_{n=1}^N$ refers to all pixels in a particular patch), the image palette parameters $\vec{\mu}_{mg}$, $(\vec{\sigma}^2)_{mg} \forall g$, and the patch palette parameters $\vec{\alpha}_{mjr} \forall j, r$. We then follow our generative model described in Sect. 2.1, except using the posterior distribution over shapelets instead of the prior over shapelets given in Eq. 5. In addition, instead of sampling from our multinomial distributions, we use the index with the highest probability mass, and instead of sampling from our Gaussian observation models, we use their means. Reconstructions are shown in Figure 4 using a shapelet library learned on Caltech28. The original images were resized to be 100×100, and we performed inference using a patch size of 8×8, a stride of 2, and $G = 6$. Since we use overlapping patches, we also average the appearance of a pixel over all patches that overlap with that pixel. As shown in the image reconstructions, our model is capable of reconstructing images with a variety of properties, such as having very many or very few colors. Additionally, while our model does not smooth over object boundaries, it tends to smooth over areas of fine detail, such as textured regions. This is to be expected, as modeling co-occurrence of colors in a patch does not lend itself well to explaining texture. We note that our reconstructions may benefit from using smaller patches and shorter strides.

4.3. Generating images

Since the shapelet model is a generative model of consistently colored image patches, we can use it to sample image patches subject to the constraint that patches should have similar intensity patterns on their boundaries. Fig. 5 shows images that were generated using the shapelet model described above. The method produces an image of global color indices. Each 8×8 patch of indices was generated by comparing the boundary region indices with the boundary

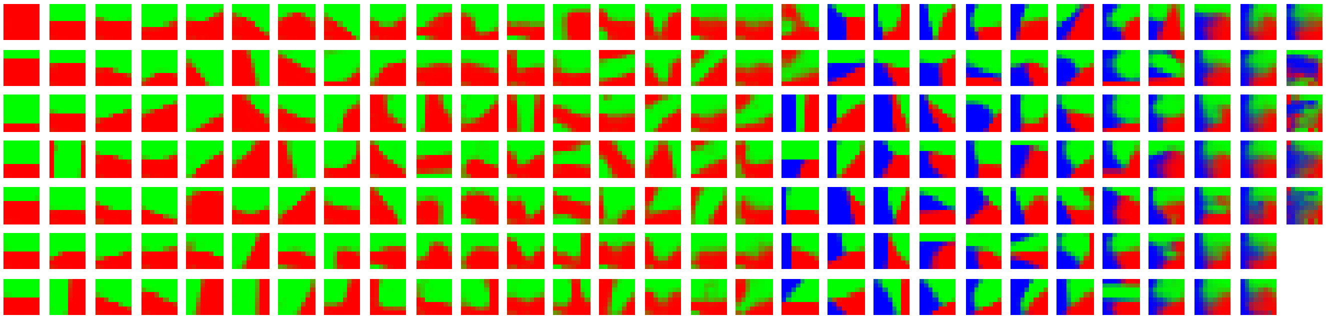

Figure 3. A library of 201 shapelets learned from Caltech28 using our shapelet model. For each shapelet, we show the multinomial parameters of the (up to) three regions for each pixel as a linear combination of red, green and blue, where each color represents a multinomial parameter. For visualization purposes, the shapelets are ordered by increasing entropy, and the region colors are arbitrary.

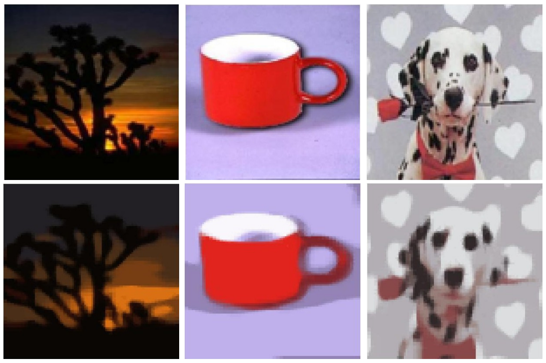

Figure 4. Image reconstructions (bottom row) and the original resized 100 × 100 images (top row). For pictures with a multitude of colors, such as the sunset picture (left), our model finds G colors that captures the dominant colors in the image. Our model can also reconstruct images where there is a relatively small range of colors, such as in the cup picture (middle). Finally, we note that our model tends to smooth regions with fine details, such as the spots on the dog's nose and ears (right).

regions indices of every shapelet. One of the 10 shapelets with lowest boundary conditional entropy was randomly selected. Then, its indices were copied into the patch, after mapping the shapelet's region indices to the corresponding global indices and generating new global indices if not all region indices could be mapped.

As shown in Fig. 5, our shapelet model is able to generate contiguous regions over a large spatial range despite operating on the patch level. In addition, our model is able to reuse image colors in different areas of the image. These properties are a result of the edge constraints, and having a color hierarchy, namely the patch palettes and image palette.

5. Object recognition

To explore the usefulness of our method, we compare it with other competitive methods that use similar size codebooks on the task of object recognition, using the Caltech28 [2] and Caltech101 [10] datasets.

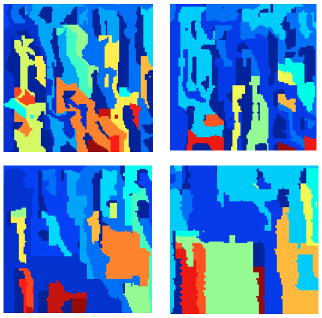

Figure 5. 100 × 100 pixel images were generated by sampling patches using the shapelet model with the constraint that the boundaries of neighboring patches must have similar intensity patterns. Boundaries of 1, 2, 3 and 4 pixels were used. Through the patch palettes and edge constraints, our model is able to generate long range structures, despite operating on the patch level. The image generated using a boundary of 4 pixels (lower right) has large contiguous regions, while the image generated using a boundary of 1 pixel (top left) has smaller contiguous regions. Note that the model is able to reuse image colors in different areas of the image.

5.1. Shapelet-based image descriptor

Given a library of shapelets, we infer the shapelet labels and palettes for a given test image and use these to construct a feature vector for object recognition. Our feature vector consists of separate shape and color descriptors.

After running inference on a test image, we describe each patch using the S-dimensional posterior distribution of shapelet labels, $P(s_{mj} \,|\, \mathbf{X}_{mj})$. Additionally, we can describe each patch according to its distribution of palette entries. Although we could use information from the patch-level and image-level palettes to quantify the distribution of colors in a patch, we have found that using the universal colors provides a richer patch description. For each patch, we construct a histogram with U bins and assign each pixel to a single bin according to its nearest universal color.

Since the shapelet model factorizes shape and color, the posterior distribution over shapelets can be viewed as an

S-dimensional descriptor of local shape and the histogram over colors can be viewed as a U-dimensional descriptor of local color. With this setup, each image can be described by $J \times S$ descriptors of local shape, and a $J \times U$ descriptors of local color, where J is the number of patches in the image.

To reduce dimensionality and gain robustness to local deformations, we follow [9, 20] and spatially pool features using three levels, consisting of 1×1, 2×2 and 4×4 grids, leading to a total of $1+4+16 = 21$ spatial regions. To make features gathered over all levels of the pyramid comparable, we normalize each type of descriptor (shapelet and palette) for each of the 21 regions to have unit length. The complete image representation consists of $21S$ descriptors of local shape, and $21U$ descriptors of local color, each of which are normalized to have unit length.

5.2. Shapelet-based classifier

In all experiments, we used RGB color images when possible, resized images to be 100×100, used 8×8 patches with a stride of 2, set $G = 6$ and $U = 125$, used a shapelet library having one single-region shapelet, 120 two-region shapelets, and 80 three-region shapelets, and set $\lambda = 2$. 8×8 patches are just large enough to capture interesting local structure in 100×100 images, and following [4], we set our stride to be as small as computationally feasible. Our results are not sensitive to the settings of G and U: setting $G = [3...8]$ and $U = [64, 125, 216]$ yields no statistical difference. For more details on how we set the shapelet library parameters, see the supplementary materials.

We use a one-versus-all SVM classifier [3] and set the soft-margin penalty to 1. Using the shapelet descriptors and palette descriptors, we compute similarity between two images A and B by:

$$K(\vec{x}^A, \vec{x}^B) = wK_s(\vec{x}_s^A, \vec{x}_s^B) + (1-w)K_c(\vec{x}_c^A, \vec{x}_c^B) \quad (14)$$

where $\vec{x}_s^A$ is the shapelet descriptor and $\vec{x}_c^A$ is the palette (color) descriptor for image A, K_s measures similarity between shapelet descriptors, K_c measures similarity between palette descriptors, and $w \in [0..1]$ is the weighting between the two. We use an intersection kernel for both descriptors.

5.3. Other methods

We compare our method with a patch-based version of the original stel model [7], the multi-level stel model [16], SIFT with the spatial pyramid match kernel (SIFT+SPM) [9], and Gist descriptors with the intersection kernel (Gist+IK) [14].

To test whether our method benefits from flexibility in the number of regions per shapelet and a hierarchical palette, we use the original stel learning algorithm to learn a stel mixture model for patches by treating each patch as a separate image. For classification, we use the posterior distribution over stel models for each patch as the local shape descriptor, and proceed with spatial pooling and SVM classification. For the multi-level stel model, we restate the results as published in [7][1].

To examine the usefulness of allowing the lowest level of our color hierarchy, namely allowing each patch to select up to R colors from the G image colors, we remove this layer from our model. Here, each image selects G global colors, and each shapelet then uses image color g to model region with index g. This is similar to our original shapelet model, but now $G = R$, and $\alpha_{mjrg} = 0$ if $r \neq g$.

We use our own implementation of the SIFT+SPM method as described in [9] using a patch size of 16×16 and a stride of 4 to extract SIFT descriptors and a codebook size of 200. To learn the codebook, we select a random set of $100,000$ SIFT descriptors from the training set. For Gist, we use the software available online with default settings [14] and our own implementation of the intersection kernel.

5.4. Caltech 28: Recognition results

For each class, we use 30 training examples and 30 testing examples. We use 10 different randomly chosen train/test splits to obtain confidence intervals, and the splits are identical across the different methods.

To perform classification, we must set the weighting, w, between the shapelet and color similarities in Eq. 14. We report results for the settings $w = \{0.5, 1\}$.

In order to make a fair comparison to SIFT and Gist descriptors, we also collected color histograms over the image, and used the similarity metric defined by Eq. 14, where we replaced the shapelet feature vectors by either SIFT descriptors collected over a spatial pyramid, or Gist descriptors, as appropriate. We form the color histograms by tiling the color space and dividing each of the H color channels into $\sqrt[H]{U}$ bins. The color histograms are collected over image patches of size 8×8 with a stride of 2, just as with the shapelet model.

We report classification results in Table 1 as the mean of the diagonal of the confusion matrix. Our method outperforms the other methods whenever color information is available, but when it is not available, the SIFT-based approach does better.

One question to ask is how useful color alone is for classification. Setting $w = 0$ so that only color is used as the descriptor yields a classification rate of $51.4\%(0.4\%)$. The multi-resolution stel model as reported in [16] achieves a recognition rate of 78.1% on this dataset, but their approach may benefit from using a color histogram[2]. Additionally, their performance numbers may change if tested on our train/test splits. Since our performance is better by a

[1] We were unable to run their code on our partitions as the code is unavailable.

[2] We note that we could not use color histograms to boost the performance of [16] as the source code is unavailable

Method	Descriptor only ($w = 1$)	Descriptor with color ($w = 0.5$)
Shapelets	74.6%(0.4%)	**83.1%(0.5%)**
Shapelets w/o indirection R=3	44.6%(0.4%)	60.9%(0.4%)
Shapelets w/o indirection R=6	33.9%(0.6%)	43.6%(0.4%)
Patch-stel model $R = 3$	74.2%(0.3%)	79.9%(0.3%)
Patch-stel model $R = 6$	72.4%(0.3%)	78.5%(0.5%)
Gist+IK	73.0%(0.3%)	77.5%(0.4%)
SIFT+SPM	75.4%(0.4%)	79.7%(0.4%)

Table 1. Caltech28 classification rates for baseline measures. "Shapelets w/o indirection" refers to our model without the mapping of patch palette colors to image colors, which performs much worse than all other methods. This demonstrates the usefulness of the finest level of our color hierarchy. Note the performance gain for all methods from adding a color histogram. The standard deviation of the estimated mean classification rate is shown in brackets.

relatively small margin over other approaches, we evaluate if this improvement is statistically significant by performing paired t-tests using the same 10 train/test splits for all methods. Our performance is superior to the other benchmarks with statistical significance (p-value ≤ 0.05) for the setting of $w = 0.5$. It is interesting to note that when ignoring color information and classifying based on shapelet similarity alone ($w = 1$), we achieve comparable recognition rates to the SIFT+SPM and Gist approaches, but we outperform them when color information is included. This difference in performance is due to our model's explicit factorization of local shape information from local color information. Because of this factorization, the shapelet model can make greater use of this source of information. On the other hand, SIFT descriptors, which are a function of image gradients within a patch, already incorporate a form of color information and so do not gain as much when explicit color information is provided. We also note the poor performance of the shapelet model when the last layer of color hierarchy is removed, which indicates the importance of the last layer.

	Faces_easy	Leopards	Motorbikes	bonsai	brain	butterfly	cougar_face	crab	cup	dalmatian	dolphin	elephant	euphonium	ewer	ferry	flamingo	grand_piano	joshua_tree	kangaroo	laptop	lotus	schooner	soccer_ball	starfish	stop_sign	sunflower	watch	yin_yang
Faces_easy	**98**									1													2					
Leopards		**70**		1		1	1	1	1	3	3	2		1	1			1	7		1		1	7	1	1		
Motorbikes			**96**		1	1			1				1						1		2						1	1
bonsai		1		**73**	4		1	1	1		1	1		1	3	2		1	4		2	1			3	3		
brain				1	**86**	1	3	1		1		1	2					2					5					
butterfly		1	1		1	**71**	1	4			2	1	1	1		1				1	3	1	1	5		2		
cougar_face		2	1			2	**68**	3	1	10	1	2	2	2	1			1	5		4		1	1			1	
crab		2				4	4	**70**		3	4		2	1		2		2	1		2			6	1	1	1	
cup					1	1			**72**	2	2	1	2	6	1	1	2	1		4	1		1	1	1		2	1
dalmatian		5		1	1	1	7	4	2	**63**	1	3				2		3	2		3			2	2	2	2	2
dolphin		1				3	1	4	1		**44**	1	1	1	2	5	1	1	3	1	2	1	1	4	1	1	1	1
elephant	1	2		2	1	1	2	1	3	4	4	**72**	1		2	3			3		1		1	2	2	4	1	
euphonium								1		1	1		**78**		1	1		1			2						1	2
ewer		2		2			1	1	4		1			**80**		1	1	1				1	1	1	1	1	1	
ferry		4		6					3		4		5	1	**76**	6	1	3	1	1	1	6			1		2	
flamingo		1		1		1	1	1	1	2	8	4	1	1	5	**63**		2	1		3	3	1	3	2	2	3	1
grand_piano								1	1		3		1	1	2	1	**91**									2		
joshua_tree		2		2			1			3			1		1	2		**74**	4		3	1		1	1	1		
kangaroo		5		2			2		1	1	5	6		1		1		2	**66**		3		1	3		1	1	1
laptop					1				1		3		1		1		1			**92**								
lotus		1		1		5	3	1	1	1	2	1		1		2		2	1		**47**		2	2		3	1	1
schooner		1							1		1	1		2	1	2						**83**						
soccer_ball				1	1	1		2	1		2			1							3	1	**73**	2			1	3
starfish		1		2		1	1	1		3	6	1				3		2	1		3	1	3	**57**	1	3		
stop_sign				1		1	1		1							1	1	1							**83**		1	
sunflower				1		1	1	1			1				1	1			1		13			3		**72**		
watch					1	4	1	1	1	1				1				1			1	1	1				**81**	1
yin_yang					1																	1	3					**86**

(a) Confusion matrix using only shapelet information

	Faces_easy	Leopards	Motorbikes	bonsai	brain	butterfly	cougar_face	crab	cup	dalmatian	dolphin	elephant	euphonium	ewer	ferry	flamingo	grand_piano	joshua_tree	kangaroo	laptop	lotus	schooner	soccer_ball	starfish	stop_sign	sunflower	watch	yin_yang
Faces_easy	**99**																											
Leopards		**87**					1	1				2				1			4					2				
Motorbikes			**95**			1			1										1			1					1	1
bonsai		2		**82**	3		1	1	1			2			1	3		2	1					1	1		1	
brain					**86**	1	1	2										1					2	1			1	
butterfly						**76**	1	3		2	1	1		2							4			2		3		
cougar_face				1		1	**81**	3	2	4		2		1					6					1				
crab			2		2	2	1	**78**	1	2	4		6			1	1							5	1			
cup					1	2			**70**	1	1		2	10	1		2	1		2			3	2			3	
dalmatian				1	1	3	4	2	4	**84**	1	4				1		1	2		2		3	1	1			
dolphin		1			1			1	1		**77**	1		2	2	2		1	1		2		2	5	1	1		1
elephant			1	3			2	1	2	2	2	**78**	1		1	1			1				1	1	2	2	1	1
euphonium				1									**85**		1		1			1	1						1	1
ewer				1					3		2			**75**		1	1	1	1	1		2						
ferry				2					4		1		3	1	**90**	3		1				5					2	
flamingo		2		1	1	2	1	1	1	1	4	1	1	2		**75**		1	3		4	1		7	3	1	1	
grand_piano									1			1		1		1	**95**									1		
joshua_tree		1		3								1	1					**84**	2		4	1	1			1		
kangaroo		4					2	2				4		1		2		1	**73**		1			4				
laptop					1						1			1						**96**							1	
lotus				2		4	1	1						1		2					**71**		3	2	1	2	2	
schooner									3		1	1		1	1							**90**	2					
soccer_ball									1	1	2			2					1				**78**	2			1	1
starfish			1				1	1		1	1	1				4		1	2		5		3	**65**		1		
stop_sign								1	3												1				**90**			1
sunflower						2						1									2					**89**		
watch				1	1	3	2		1	2								1									**84**	
yin_yang						1								1									2				1	**94**

(b) Confusion matrix using shapelet and color information with $w = 0.5$.

Figure 6. Confusion matrices for Caltech28. Entry (i, j) is the percentage of the time class i was classified as class j, averaged over 10 trials. For pairs of classes not disambiguated by local shape alone, such as the lotus and sunflower classes, and classes where color is highly informative, such as the dolphin class (abundance of blue), adding color information significantly improves performance. However, adding color information hurts performance for a few classes, such as the ewer class. Performance increases are in green circles, and decreases are in red rectangles.

Fig. 6 shows the confusion matrix of the shapelet model both with and without color information. Note that some class pairs that are not well disambiguated by shape alone, like sunflowers and lotuses, are well distinguished when color information is also used. We note again that for such classes, we receive such significant boost in classification rate by adding color information since the shapelet model factorizes description of local shape from description of local color. However, this gain in classification rate is not true for all classes. For example the ewer class has a lower classification rate when color is added. This occurs since in the ewer images, the background, which is often a uniform color that is different for each image, takes up a large portion of the image. Because of this, the color histograms capture color information primarily about the uninformative background. To compound this difficulty, the cup class images appears on similar kinds of backgrounds as the ewer images, and Fig. 6 shows that with color information, ewers and cups are confused more often.

Method	Descriptor only ($w = 1$)	Descriptor with color ($w = 0.5$)
Shapelets	$52.1\%(0.7\%)$	$\mathbf{56.7\%(0.7\%)}$
Patch-stel model $R = 3$	$52.8\%(0.5\%)$	$56.1\%(0.7\%)$
Gist+IK	$48.6\%(0.4\%)$	$52.4\%(0.6\%)$
SIFT+SPM[3]	$53.0\%(0.4\%)$	$56.4\%(1.1\%)$

Table 2. Caltech101 classification rates. The standard deviation of the estimated mean classification rate is shown in brackets.

5.5. Caltech 101: Recognition results

We used the experimental setup as described in Sect. 5.4 for all methods, except with only 15 training examples, and 5 random train/test splits for obtaining confidence intervals. Classification results are reported in Table 2. For the setting of $w = 0$, the classification rate is $27.6\%(0.4\%)$.

We note that our results are competitive with both the SIFT+SPM and Gist+IK approach, as well as the standard stel model applied to patches for $R = 3$. We note again that the boost in performance we get from adding color information is significantly greater than the boost seen by the SIFT+SPM and Gist methods, since we factorize local shape and local color in our model. Due to space restrictions we omit reporting the confusion matrix for this dataset, but note again that including color descriptors as well as shapelet descriptors generally improves the classification rate of all classes and helps to disambiguate similarly shaped classes, just as in Caltech28. Finally, we ran our method on the same three train/test partitions the authors of the multi-resolution stel model [16] used to evaluate their method. We achieved a recognition rate of $57.9\%(0.6\%)$ with a setting of $w = 0.5$ on this partition, while the method of [16] achieved 58.92%. These additional trials are not included in the set of trials reported in Table 2.

6. Conclusion

In this work, we present extensions of the stel model to a patch-based framework, and introduce the notion of hierarchical palettes for describing the coloring of an image set, images, and patches in an image. We demonstrate that our framework factorizes local shape from local color in the form of shapelets and palettes, respectively. With such a factorization, object classification can be performed using descriptors encoding shape and color information separately. We have showed that our model is competitive on Caltech28 and Caltech101 against several baselines.

We note that we may achieve stronger classification results using the recent work in [1], which illustrates how to construct more powerful codebooks when the number of codewords is fairly small. We may also benefit from different pooling arrangements, an approach which has been recently shown to yield significant improvements [1, 20].

7. Acknowledgements

The authors would like to acknowledge funding to Brendan Frey from the Canadian Institute for Advanced Research (CIFAR) and the Natural Sciences and Engineering Research Council of Canada (NSERC), and funding to Jeroen Chua from NSERC.

References

[1] Y.-L. Boureau, N. L. Roux, F. Bach, J. Ponce, and Y. LeCun. Ask the locals: multi-way local pooling for image recognition. In *ICCV*, 2011.

[2] L. Cao and L. Fei-Fei. Spatially coherent latent topic model for concurrent object segmentation and classification. In *ICCV*, 2007.

[3] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

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

[5] S. M. Eslami and C. Williams. Factored shapes and appearances for parts-based object understanding. In *BMVC*, 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] N. Jojic and Y. Caspi. Capturing image structure with probabilistic index maps. In *CVPR*, 2004.

[8] N. Jojic, A. Perina, M. Cristani, V. Murino, and B. J. Frey. Stel component analysis: Modeling spatial correlations in image class structure. In *CVPR*, pages 2044–2051, 2009.

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

[10] F.-F. Li, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1), 2007.

[11] D. Lowe. Object recognition from local scale-invariant features. In *Proceedings of ICCV*, 1999.

[12] D. Marr. *Vision: A computational investigation into human representation and processing of visual information*. W. H. Freeman and Company, San Franciso, 1982.

[13] R. Neal and G. E. Hinton. A view of the EM algorithm that justifies incremental, sparse, and other variants. In *Learning in Graphical Models*, pages 355–368. Kluwer Academic Publishers, 1998.

[14] A. Oliva and A. Torralba. Building the gist of a scene: the role of global image features in recognition. In *Progress in Brain Research*, page 2006, 2006.

[15] B. A. Olshausen and D. J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, June 1996.

[16] A. Perina, N. Jojic, U. Castellani, M. Cristani, and V. Murino. Object recognition with hierarchical stel models. In *ECCV (6)*, 2010.

[17] A. Perina, N. Jojic, and V. Murino. Structural epitome: a way to summarize one's visual input. In *NIPS*, 2010.

[18] M. Ranzato and G. E. Hinton. Modeling pixel means and covariances using factorized third-order Boltzmann machines. *JMLR*, 7:2369–2397, 2003.

[19] D. Ross and R. S. Zemel. Learning parts-based representations of data. *JMLR*, 7:2369–2397, 2003.

[20] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

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

[3] We do not exactly reproduce the results reported by [9] of 56.4% using 15 training examples due to our resizing of images