Supplementary document: Transient Attributes for High-Level Understanding and Editing of Outdoor Scenes

Pierre-Yves Laffont Zhile Ren Xiaofeng Tao Chao Qian James Hays
Brown University

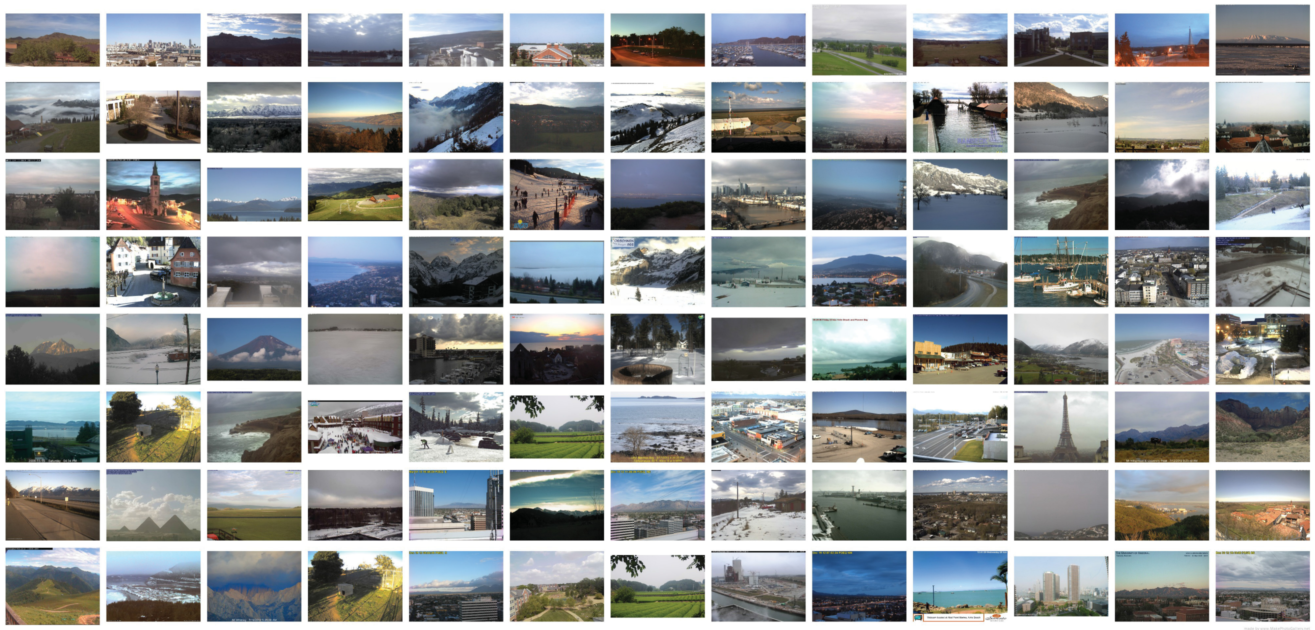


Figure 1: *Overview of the 101 webcams annotated in our **Transient Attribute Database**.*

This document contains additional details and figures on:

Our Transient Attribute Database: selection of representative images (Section 1.1), screen captures of our crowdsourced annotation tasks (Section 1.2), comparison of our aggregated labels to meteorological data (Section 1.3), and correlation between attributes (Section 1.4).

Our attribute recognition method: description of the image features and encoding methods used (Section 2).

In addition, our project website[1] contains:

- the list of our 40 transient attributes along with their definition and positive/negative examples
- all 8571 images from our database with their attribute labels
- our trained attribute regressors (with Fisher Vector encoding and spatial pyramids), along with train/test splits
- results of our appearance transfer method on 60 test cases for different attribute manipulation queries, along with the match/target images used, and comparison to three other methods

[1]`http://transattr.cs.brown.edu`

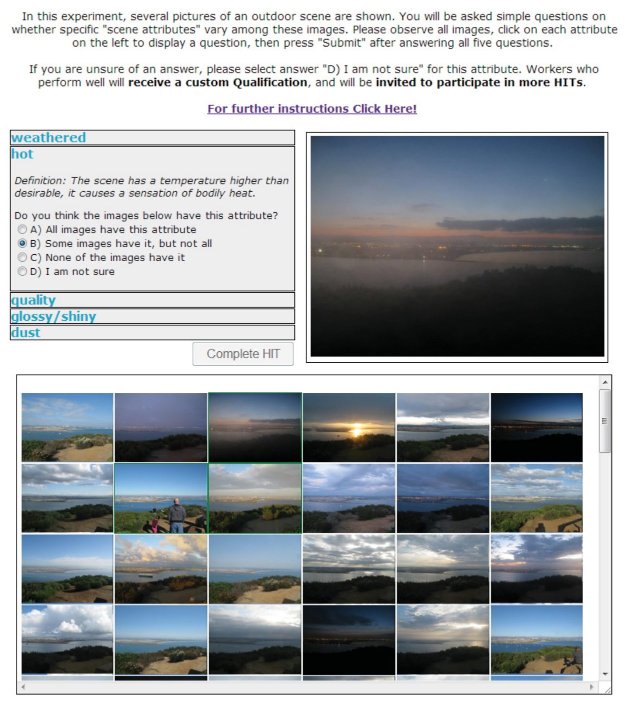
1 The transient attribute database

1.1 Selecting representative images

For each webcam in our dataset we select 60-120 high quality frames which are representative of the appearance variations of the scene. For the first 35 webcams in our dataset, we used the iterative approach of Abrams et al. [2011] to find "interesting" frames, then manually filtered the results using a web-based interface. The annotations gathered for these webcams allowed us to train an initial classifier for each attribute. For each new webcam to add in our dataset, we run these classifiers on a thousand randomly selected to estimate their attribute labels, then select around 90 images that exhibit the most attribute variations with a greedy algorithm. Note that this process was used to ensure the selected images exhibit lots of appearance variations; we then discard the output of the initial classifiers, and rely on the annotations collected for those new webcams in order to improve attribute recognition performance.

1.2 Crowsourcing experiments

Figures 2-3 show the user interfaces that we developed for our crowdsourcing experiments. Both tasks were run on Amazon Mechanical Turk. We displayed the original webcam images in these tasks, i.e. before they are manually aligned for the image manipulation.


In this experiment, several pictures of an outdoor scene are shown. You will be asked simple questions on whether specific "scene attributes" vary among these images. Please observe all images, click on each attribute on the left to display a question, then press "Submit" after answering all five questions.

If you are unsure of an answer, please select answer "D) I am not sure" for this attribute. Workers who perform well will **receive a custom Qualification**, and will be **invited to participate in more HITs**.

For further instructions Click Here!

weathered

hot

Definition: The scene has a temperature higher than desirable, it causes a sensation of bodily heat.

Do you think the images below have this attribute?

A) All images have this attribute

[✓ B) Some images have it, but not all]

C) None of the images have it

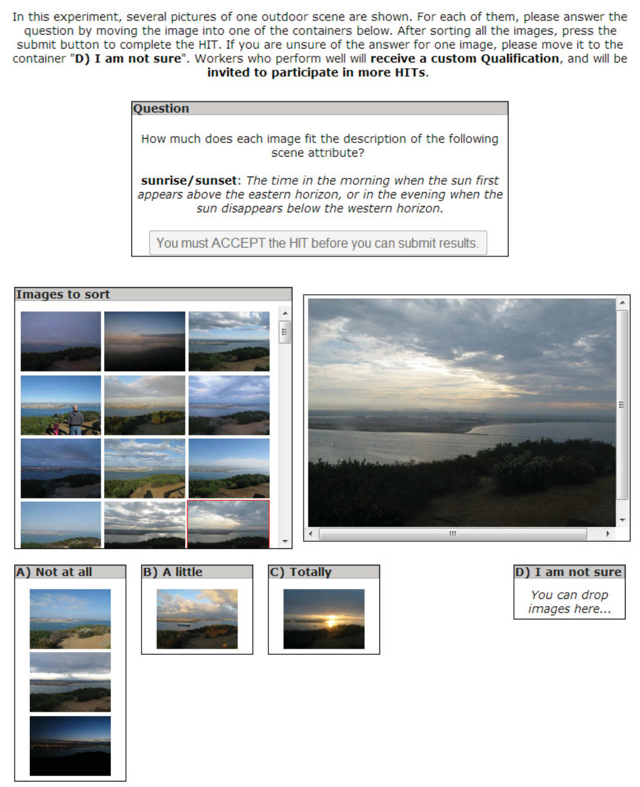
D) I am not sure

quality

glossy/shiny

dust

Figure 2: *User interface for the first crowdsourcing experiment (**discovering transient scene attributes**).*


In this experiment, several pictures of one outdoor scene are shown. For each of them, please answer the question by moving the image into one of the containers below. After sorting all the images, press the submit button to complete the HIT. If you are unsure of the answer for one image, please move it to the container "**D) I am not sure**". Workers who perform well will **receive a custom Qualification**, and will be **invited to participate in more HITs**.

Question

How much does each image fit the description of the following scene attribute?

sunrise/sunset: *The time in the morning when the sun first appears above the eastern horizon, or in the evening when the sun disappears below the western horizon.*

You must ACCEPT the HIT before you can submit results.
Images to sort
A) Not at all
B) A little
C) Totally
D) I am not sure
You can drop images here...

Figure 3: *User interface for the second crowdsourcing experiment (**annotating images with their attributes**).*

1.3 Comparison to weather measurements

We compare the aggregated labels of weather-related attributes to the weather data recorded by Weather Underground[2], for a subset of webcams with known GPS position.

Figure 4 shows the aggregated label for the attribute "cloud", for each group of images corresponding to different reported "sky conditions". As expected, the mean attribute label is higher for images reported as more cloudy by weather underground. The curve does not fully extend to the $[0-1]$ range however; one reason being that the reported weather data is often be inaccurate [Islam et al. 2013].

Unsurprisingly, other attributes can be much harder to recognize. For example, we did not find significant difference when comparing the labels of attribute "dry" (resp. "windy") on the images with lowest 10% and highest 10% measured humidity (resp. wind speed) in each scene. This is not a problem for our approach however – we focus on *perceived* attributes rather than ground truth properties of the scene.

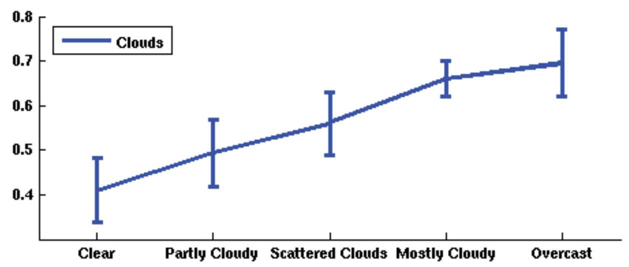


Figure 4: *Mean value of the aggregated labels for attribute "clouds", with respect to the "sky condition" label reported by Weather Underground. We use images from 6 geolocated webcams. Bars represent the 95% confidence intervals.*

[2]http://www.wunderground.com/weather/api/

1.4 Attribute correlation

In order to discover semantic relationships between attributes, we compute the Pearson linear correlation coefficient between each pair of attributes. The resulting correlation matrix is shown in Figure 5. Unsurprisingly, related attributes such as dark/night have a highly positive correlation, while inversely related attributes such as warm/snow have a strongly negative correlation.

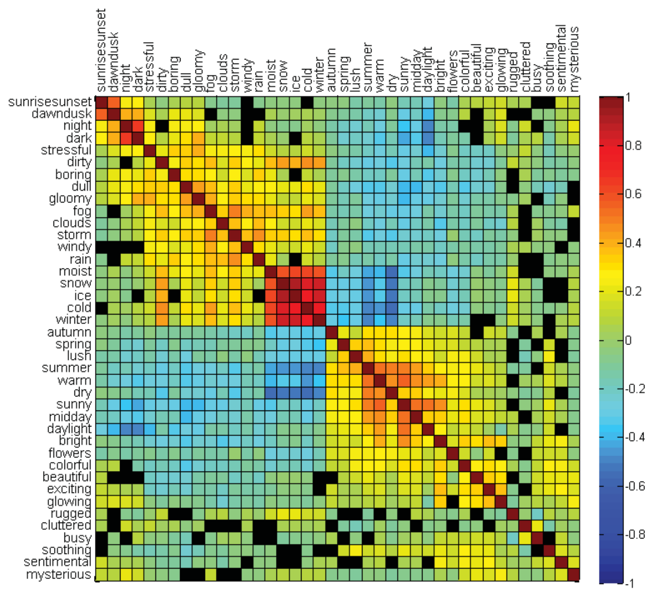


Figure 5: *Attribute correlation matrix after aggregating annotations, color-coded according to the scale on the right. Black cells correspond to unreliable comparisons where p-value* > 0.05.

2 Attribute recognition

2.1 Image features for attribute recognition

The features we use for recognition are global image representations and local descriptors that are known to be effective for scene classification (please refer to the full description by Xiao et al. [2010]):

- **HOG 2x2**: At every 8 pixels, a 31-dimensional histogram of oriented gradients (HOG) features is computed. Neighboring HOG descriptors are concatenated to form 124-dimensional 2x2 HOG patches.

- **SSIM**: At every 5 pixels, 30-dimensional self-similarity (SSIM) descriptors are calculated by quantizing the local patch correlation map into 30 radial bins (3 radii at 10 angles).

- **GIST**: The output magnitudes of 24 multiscale oriented Gabor like filters are linearly combined to form this GIST image representation.

- **Geometric context color histograms**: A 784-dimensional LAB color histogram is extracted from the whole image. Similar histograms are also extracted for each geometric context class (ground, sky, vertical, and porous), with each color sample's influence being weighted by the probability of that geometric class at the sample's location.

Patterson and Hays [2012] use Bag of Visual Words encoding for both HOG and SSIM, after the features are computed. HOG and SSIM features are translated to visual words based on visual word vocabularies (of size 300) generated by k-means clustering. For both features, this visual word representation is computed at three different spatial pyramid levels: 1x1, 2x2, and 4x4. Individual kernels are then generated using L_1 distance for the HOG features and χ^2 distance for the SSIM, GIST, and geometric context color histogram features. Finally, individual kernels for all features are normalized and linearly combined to form a combined feature kernel.

We can use this combination of features and kernels for training predictors to recognize transient attributes; we report the performance of three predicting methods (SVM, logistic regression, and SVR, as in the main paper) in Table 2. We further extend this previous work by improving on the feature encoding methods for HOG and SSIM.

2.2 Fisher Vector encoding for HOG and SSIM

The features and kernels described in Section 2.1 are useful for recognizing attributes, as shown by Patterson and Hays [2012]. However, the Bag of Visual Words approach for encoding HOG and SSIM can be lossy, even when soft binning is employed.

We replace Bag of Visual Words with Fisher Vector encoding to better address this weakness. Fisher Vector encoding uses a generative model to represent the visual vocabulary of image features. For a particular image, a Fisher Vector encodes how the parameters of that model should change to better represent that image's features. As Perronnin et al. [2010], we use a Gaussian Mixture Model (GMM) as the generative model on the image features. This allows for Fisher Vector encoding to capture 1st and 2nd order statistics (mean and covariance of the GMM) from the features, instead of just a histogram count as with Bag of Visual Words. However, a Fisher Vector image representation inherently has no spatial context. Therefore, we augment the spatial context with a conventional spatial pyramid approach.

	Random split		Holdout split	
	MSE	AP	MSE	AP
SVM (bag of words)	0.048	0.95	0.077	0.75
log reg (bag of words)	0.063	0.90	0.094	0.72
SVR (bag of words)	**0.020**	**0.96**	**0.046**	**0.77**

Table 2: *Recognition performance when using Bag of Words encoding for HOG and SSIM. Performance on both splits is worse than with Fisher Vector encoding (shown in Table 1 of the main paper). We use mean squared error (MSE) and average precision (AP) to evaluate their performance.*

Implementation. In our implementation, we use the VLFeat library implementations of GMM and Fisher Vector encoding [Vedaldi and Fulkerson 2010]. First, prior to any training or encoding, we reduce the 124-dimensional HOG 2x2 descriptor down to $D_{\text{HOG}} = 64$ dimensions using principal component analysis. This intuition comes from [Perronnin et al. 2010], where the authors showed that reducing high dimensionality features could be beneficial, especially in reducing the sparseness and size of the resulting Fisher Vector encoding. However, we do not find it necessary to apply dimensionality reduction for SSIM, since SSIM already has only $D_{\text{SSIM}} = 30$ dimensions.

In practice, for the HOG (resp. SSIM) local features, we use on the order of 10^6 descriptors from randomly sampled images to train a GMM with $K = 256$ Gaussians; Perronnin et al. [2010] used a similar number of randomly sampled SIFT descriptors for training. This results in a 32768-dimensional Fisher Vector encoding for HOG and a 15360-dimensional Fisher vector encoding for SSIM. We further apply a power normalization with an exponent of $a = 0.3$ and perform L2 normalization. Although tuning these three parameters (the number of Gaussians K, the input HOG feature dimensionality D_{HOG}, and the power normalization factor a) can improve performance, we found that the recognition performance on the Transient Attribute Database is fairly robust to these parameter changes.

Next, we incorporate spatial pyramids to add spatial context to the otherwise global Fisher Vector representation. As shown by Perronnin et al. [2010], this is a simple but effective way to improve performance. With spatial pyramids, the image is subdivided into multiple levels of image regions, and a Fisher Vector is extracted from the low-level features in each region. We experimented with different basic configurations of spatial pyramids, and empirically determined that the best basic spatial pyramid configuration is to divide the image into thirds (top third, middle third, bottom third). This makes sense because this matches the configuration of many outdoor photographs where the sky is in the top third and the ground is in the bottom third of the image. Finally, the Fisher Vectors encodings from all four regions (i.e., whole image + the three thirds) are concatenated to form the final encoding. Each Fisher Vector encoding is normalized separately before concatenation.

Encoding all N images in the database yields a matrix of N stacked Fisher encodings; the matrix is of size $N \times 131072$ for HOG and $N \times 61440$ for SSIM. We multiply this matrix of encodings with its transpose to create a feature kernel. We use those feature kernels in replacement of the kernels generated for HOG and SSIM in Section 2.1: we linearly combine them with our existing normalized kernels for GIST and geometric context color histograms, with equal weights, to form the combined feature kernel.

Recognition rate. Replacing the Bag of Visual Words encoding with Fisher Vector encoding for the HOG and SSIM features improves performance on both training-test splits, as seen by comparing Table 1 (in the main paper) with Table 2. In particular, average Precision increases from 0.77 to 0.80 when using SVR for predicting attribute values on the holdout test split.

Performance. The Fisher Vector encoding run time is comparable to using Bag of Visual Words. For a new image, recognizing all attributes takes less than 20 seconds on a machine with 32 cores at 2GHz. This includes generating the combined feature image representation, including Fisher Vectors, and then predicting labels with trained SVR models. However, the memory requirements for Fisher Vector encoding are greater due to its relatively higher dimensionality. Uncompressed, the HOG and SSIM Fisher Vector encodings for our entire dataset occupy about 6.1 GB in memory. Despite this slight drawback, Fisher Vector encoding is a valuable tool for building a more effective combined feature kernel. We release our trained predictors on the project website.

References

ABRAMS, A., FEDER, E., AND PLESS, R. 2011. Exploratory analysis of time-lapse imagery with fast subset pca. In *WACV'11*.

ISLAM, M., JACOBS, N., WU, H., AND SOUVENIR, R. 2013. Images+weather: Collection, validation, and refinement. In *CVPR Workshop on Ground Truth (CVPRW)*.

PATTERSON, G., AND HAYS, J. 2012. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *CVPR*.

PERRONNIN, F., SÁNCHEZ, J., AND MENSINK, T. 2010. Improving the fisher kernel for large-scale image classification. In *ECCV*.

VEDALDI, A., AND FULKERSON, B. 2010. Vlfeat: An open and portable library of computer vision algorithms. In *International Conference on Multimedia*.

XIAO, J., HAYS, J., EHINGER, K. A., OLIVA, A., AND TORRALBA, A. 2010. Sun database: Large-scale scene recognition from abbey to zoo. In *CVPR*.