Sunset Detector

CSSE463 – Image Recognition

Ander A Solorzano
Rose-Hulman Institute of Technology
Electrical Engineering and Robotics Engineering
Terre Haute, IN
solorzaa@rose-hulman.edu

Ruffin White
Rose-Hulman Institute of Technology
Electrical Engineering and Robotics Engineering
Terre Haute, IN
whitemrj@rose-hulman.edu

ABSTRACT

The purpose of this project is to determine whether a data set of images contain sunsets. Using a nonlinear Support Vector Machine (SVM), we trained a classifier based on a sample of training images. Our set of images consisted of sunsets and non-sunsets, ranging from traditional coastline during late afternoon to images of domestic farmlands and urban terrains. Our feature extraction method calculated the first two moments of the LTS color space for every region of the image that is a geometrically divided grid structure. These features are extracted from both the training and test sets and are later used to normalize the numerical values of each feature using the range the entire collection. The training set is then used to train SVM utilizing a RBF kernel. Using a grid search approach, we searched for an optimal γ parameter value that yields the highest TPR and lowest FPR value. Once we settled on a γ parameter of the RBF kernel, we varied the threshold value that discerns the detected class to generate a traditional ROC curve. Our best selected results yielded a TPR of 86.61% and a FPR of 9.80% with 18.4 as a *gamma* parameter of the RBF function. Furthermore, our accuracy of correct images consisted of 88.83% with a precision of 90.16% for the entire set of test images greater than the decision threshold of -0.2735. However, after using alternative ways of normalizing our images, using different grids sizes, and overlapping grid regions, we were able to achieve an accuracy of 91.6%, a precision of 89.0%, TPR of 95.3%, and FPR of 12.2% with a decision threshold of -0.479.

Keywords—RGB, LST, feature extraction, SVM, kernel, RBF, hyperplane, color moments, ROC, True Positive Rate, False Positive Rate, mean, variance, overlapping grids, grid search

I. INTRODUCTION

The purpose of this project is to train a classifier that can be able to distinguish between sunset images and non-sunset images. One of the reasons we might care about a sunset detection algorithm is to become aware of the limitations and trade backs when manually configuring the feature identification process, as well as the finer details in configuring an automated training algorithm. For instance, if we would like to achieve a low false positive rate (FPR), we would have to create a very discriminative algorithm that only matches ideal cases for sunsets. However, this would also cause a drop in the true positive rate (TPR) and thus miss some of the sunset images. The other route we can take is to decrease our threshold parameters and thus make our algorithm more lenient when selecting sunset images. As expected this would increase our TPR but also our FPR. This methodology of using abstract photo identification to classified sunsets and non-sunset can then be expanded to other fields and application. For instance, this algorithm can be used in the medical technologies for detecting brain tumors of a set of thousands of scanned images. We can also apply our algorithm to industrial applications like detecting faults in the assembly line of machined parts or defects in mass-produced silicon fabrication. Thus the end-user application will dictate whether we would like a very discriminative algorithm (ideal for industrial applications) or a more lenient algorithm that allows for higher detection rates (ideal for medical applications).

To be able to achieve a solution to this problem, we would have to extract some features of the set of images that we would expect to find in a typical sunset image. For instance, a typical sunset image would have bright and warm colors located in the upper half of the image and dark colors in the foreground of the image. Furthermore, we would expect to see a representation of a sunset sky for at least half of the picture.

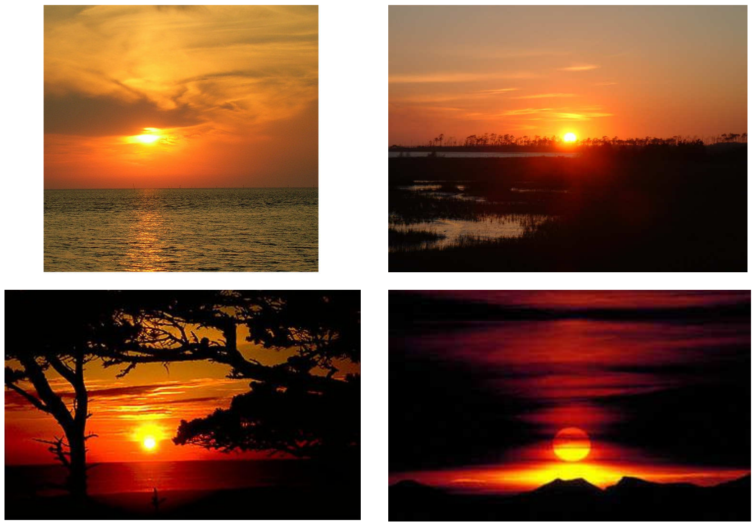


Figure 1: Ideal examples of sunset images. These images were used to train our classifier due to the rich amount of expected features from a sunset image.

Nevertheless, not all sunset images we encountered consisted of an ideal type of sunsets. Some images contained very little warm colors but were clearly sunset images. Others were homogenous throughout (i.e. mostly monochromatic with pale cool colors) and small bright regions. Other sunset images were also tricky to detect due to the composition and overall brightness of the picture. These include sunset images with a large portion of the foreground occupied by non-sunset type objects and little sunset type features left in the negative space of the image. Other types of hard-to-detect sunsets contained non-ideal weather conditions and different seasonal times. For example, a winter sunset with a snow storm present can be hard to detect due to the large amount of vertical shadows and white snow and frost. Other types of hard-to-detect sunsets may include black and white sunset images or sunset images that were orientated differently in order to throw off the classifier.

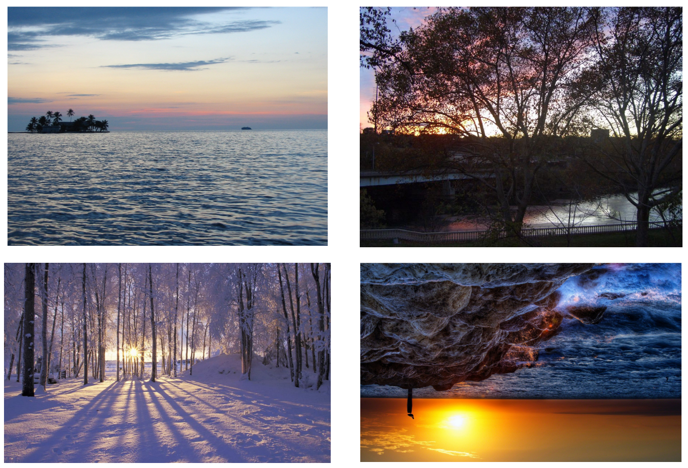


Figure 2: Types of hard-to-detect sunset images. The top left image resembles a monochromic sunset with almost no rich sunset features. The top right image contains a lot of non-sunset type foliage that's silhouettes the entire foreground thus leaving almost no space for sunset features. The bottom left sunset image is taken at a different time of the year (winter) that can throw off the detector. The bottom right image is a sunset image that was purposefully inverted in orientation since the classifier would expect to find sunset features in the upper half of images.

Aside from the set of sunset images we were give, we were also supplied with a set of non-sunset images that were sometimes misclassified as a sunset image due to their close resemblance of sunset features.

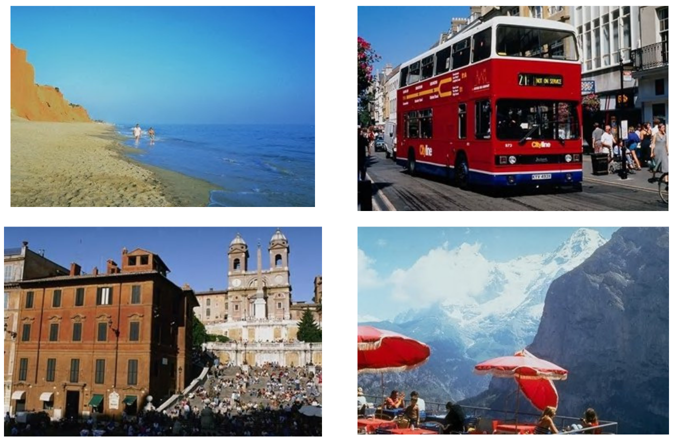


Figure 3: Types of non-sunset images. Although these are clearly non-sunset images, these images contain some sunset-like features such as bright blue skyline, contrasting upper and lower half dynamics as well as a few elements of warm colors that might through off the classifier.

II. PROCEDURE

A. Importing the images and converting color spaces

Within the algorithm written in MATLAB, the first step of our procedure is to acquire the entire set of training and test images and extract their RGB values. We do this by specifying each set's specific directory and iteratively importing every image file.

One troublesome issue blindly importing all the files within a 'image only' directory is that not all files are in fact images. The Windows operating system specifically has a troublesome habit of catching the preview thumbnail of all the images within the directory for quick access. When importing images from a directory, it helps to first filter out any of these hidden thumbnail caches. Once the image is opened, it is first imported into a three-dimensional matrix consisting of its red green and blue components. We then proceeded to convert the data into a more useful color space for our feature extraction algorithms, in this case, LST. Our formulas for LST conversion are simplified from the actual specifications. Bhey serve the same basic premise by representing the image, not just its intensities, of three individual frequency spectrums with their intensity and contrast.

B. Extracting features

In order to maximize efficiency and reduce resource consumption, we utilized the time within every iteration of our file acquisition loop to additionally process and store the results from our feature extraction. We geometrically defined the entire picture into a 7x7 gridded structure and then looped through every section calculating the first two moments for each band of the LTS color space. This results in calculating both the mean and variance of three separate values for each of the 49 section, resulting in exactly 294 features per image (2*3*7*7 = 294). This feature extraction is done and saved for every image within each set (this includes training sunsets, training non-sunset, test sunsets, and test non-sunset). This is done with care in order to properly separate and segment each data set from each other. We then carefully collected the entire assortment of features into a single N by 294 matrix, where N is the total number of assorted images. We then normalized the range of values for each given feature in order to provide the SVM a manageable finite range of feature values to be used for training and test classification. It is important to know that both the training and tests set contribute to the total normalization process of each feature.

C. Using grid searching to optimize the γ parameter for RBF kernel

When training a SVM using a nonlinear RBF kernel, optimizing the additional parameters such as C and γ is useful, if not essential, in training a reasonable classifier. One of the few methods as suggested by a group of researchers at National Taiwan University (*A Practical Guide to Support Vector Classification*) includes grid search approach. An implementation of grid searching would start by generating a wide logarithmic range between several decades for both C and γ consecutively while testing each possible combination in the gridded assortment of values to train and test the classifier.

Using the results generated, it is possible to plot a meaningful topological graph highlighting combinations with the highest accuracy. However, the effects of configuring the C parameter were minimal and so to lessen computation time, we reduced our search by limiting variable parameters to γ only, leaving C fixed at a suitable value of 10. Thus, by looping through γ parameters ranging from 10^{-2} to 10^{4} with 69 points in between, we then plotted the accuracy and observed TPR and FPR values in a logarithmic plot and selected a RBF kernel γ parameter which we held fixed when generating the ROC curve.

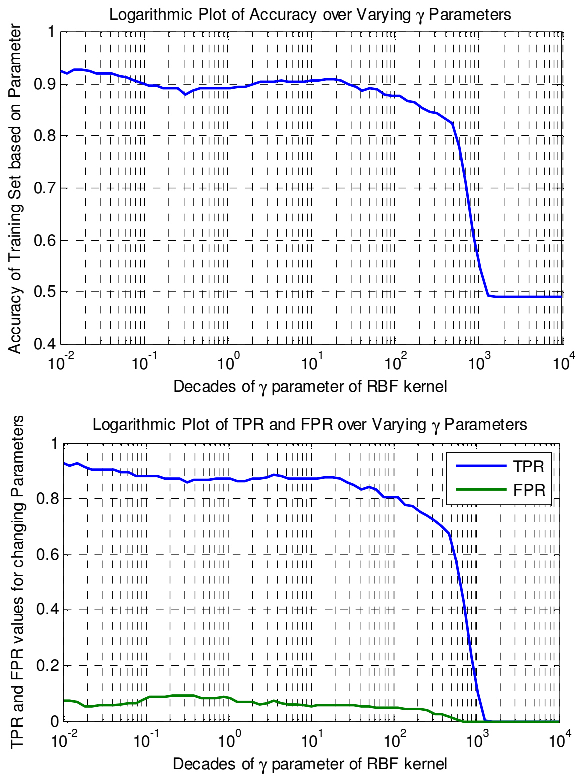


Figure 4: Generated logarithmic plots for the accuracy (top) and TPR versus FPR values (bottom) of the RBF kernel γ parameter.

From these generated logarithmic plots, we selected the γ parameter to be 18.4. Although a very small value like 0.01 does give a higher accuracy and a higher TPR with a relatively similar FPR value, we decided not to use a small value since it would be similar to *memorizing* the classes of the training set rather than *learning* the classes. Furthermore, memorization of classes can present a problem if the training set and testing set of images vary significantly. As a result, we settle on using a value that would not necessarily memorize the classes and yet give an acceptable TPR and FPR value for our type of application.

D. *Generating an ROC for varying distance thresholds*

In order to generate a traditional ROC curve from our finalized classifier, we began by varying our classifier distance threshold. Sweeping through the range of distances outputted by our SVM when given the test set, we can plot the corresponding FPR vs. TPR. In an ideal case, somewhere along our ROC curve adheres closely to the coordinate (1,0). This could then be interpreted that the classifier is capable of achieving 100% TPR while maintaining a 0% FPR. Although this does not entail 100% accuracy, it does ensure that every detected sunset was an actual sunset and that there were no images misclassified as falsely detected as a sunset. Thus possible false negatives and true negatives are still plausible and require further investigation or the construction of a full confusion matrix.

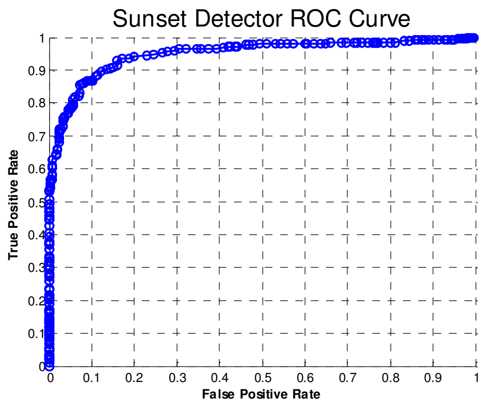


Figure 5: ROC curve of TPR values versus FPR values with a fixed RBF kernel γ parameter and varying threshold distances.

III. RESULTS

A. *Overall results*

From our test set of images, we observed that the minimum and maximum distance values from the output of the *svmfwd()* function are -2.011 and 1.994 respectively. Images with a distance value closer to 1.994 resemble sunsets while images closer to -2.011 resemble non-sunsets. In between this range of distances, hard-to-detect sunsets can be closer to -2.011 than 1.994 while non-sunset images with sunset-like features can be closer to 1.994 than -2.011. The generation of an ROC curve can help us select a decision threshold which will classify images as sunsets and non-sunsets. The type of application at hand (in this case a sunset detector) will dictate whether larger TPR with higher FPR values or lower TPR with minimum FPR values will be needed. Ideally for any application, we would like to have the highest TPR with no FPR but due to the sunset-like features of the non-sunset images and the non-sunset-like features of sunset images generate false readings.

From the ROC curve, we decided to use a decision threshold distance of -0.2735. Images with distances greater than the decision threshold were classified as sunset images while images with distances less than the threshold were classified as non-sunsets. We generated the ROC curve for the set of test images containing 254 sunset images and 245 non-sunset images. *Refer to Table 1 shows the values of accuracy, TPR, FPR, and precision.*

RBF Kernel Distance γ Parameter	**18.4**
Decision Threshold	**-0.2735**
Accuracy	**88.83%**
Precision	**90.16%**
Recall (True Positive Rate)	**86.61%**
False Positive Rate	**9.80%**

Table 1: Overall results for our selected decision threshold value and a fixed RBF kernel γ parameter.

B. *Accurate and correct readings*

Based on our decision threshold, we were able to classify correctly most of the ideal sunset images in the test set. The reason why these images were classified correctly was because it contained enough sunset-line or non-sunset-like features which the classifier accurately detected. *Refer to Appendix A for sample of images that were classified correctly as sunsets and non-sunsets.*

C. *False Positives and False Negative Readings*

Our algorithm did struggle when classifying images with confusing sunset-like features. For example, some of the hard-to-detect sunset images were not classify as sunsets while some non-sunset features were classified as sunsets. These false readings occurred due to the leniency of our classifier and due to the features of the images. Some of the results do not make any sense since there were some images that were taken at different angles yet one of the images was classified correctly as a sunset while another image was classified as a non-sunset. *Refer to Appendix A for the sample of images that were incorrectly classified.*

D. *Extensions for Sunset Detection Improvement*

After the reported results above, we decided to improve the algorithm and add extensions to our classifier that will push the TPR closer to 100% while leaving the FPR relatively the same. For convenience purposes, we always used the same RBF kernel γ parameter, the same *C* kernel parameter, and the same iteration or instance of the decision threshold (i.e. step 109).

1) Using alternate normalization method

By using an alternative means of normalization utilizing the first two moments, mean and standard deviation, we were able to generate a higher percentage of accuracy as well as a higher percentage of precision. We were also able to observe that the minima decreased and the maxima increased, resulting in larger range and less congested set of distances around the median.

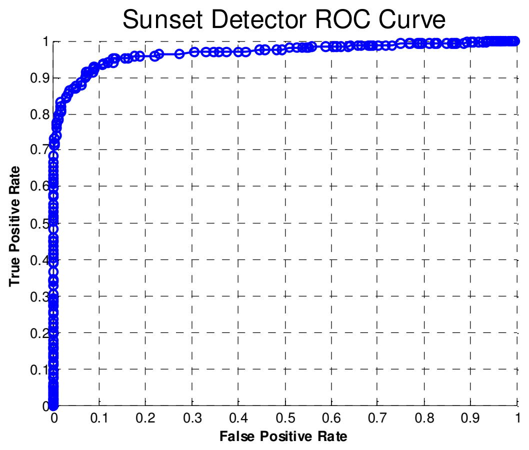


Figure 6: Improved ROC curve by using alternative means of normalization methods.

RBF Kernel Distance γ Parameter	**18.4**
Decision Threshold	**-0.663**
Accuracy	**90.00%**
Precision	**68.40%**
Recall (True Positive Rate)	**95.30%**
False Positive Rate	**15.50%**

Table 2: Overall results after using alternate means of normalization.

2) Using different size grids

We also decided to tweak the sizes of the grids in the images and used a 9x9 grid separations as supposed to 7x7 grid separations. We applied this algorithm to the alternate normalized method in order to achieve better results. Further increasing the grid separations (i.e. going to 11x11, 13x13, and so on) showed little improvements. Thus we decided to use a 9x9 grid separations. The results for using a 9x9 grid separation with the alternate means of normalization are as follows:

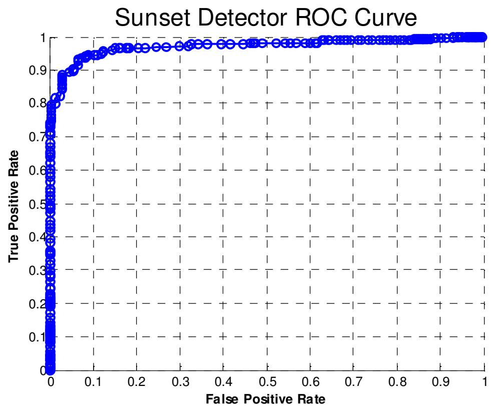


Figure 7: Improved ROC curve by using 9x9 grid separations and alternative means of normalization methods. Compared to the last ROC curve, you can see that the data points in the upper left grid got pushed up thus showing improvements in the algorithm.

RBF Kernel Distance γ Parameter	**18.4**
Decision Threshold	**-0.485**
Accuracy	**91.60%**
Precision	**89.0%**
Recall (True Positive Rate)	**95.3%**
False Positive Rate	**12.2%**

Table 3: Overall results after using alternative means of normalization and a 9x9 grid separation.

3) *Using overlapping grid blocks*

Finally we decided to see if overlapping grid would improve our results. We observed changes to the results after plotting 5%, 10%, and 25% overlapping regions. There were almost no changes to the FPR, accuracy, and precision values. The only values that changed were the range of distances and the TPR rates. Using the extensions described above, the results for including a 10% overlapping of grid regions are as follows:

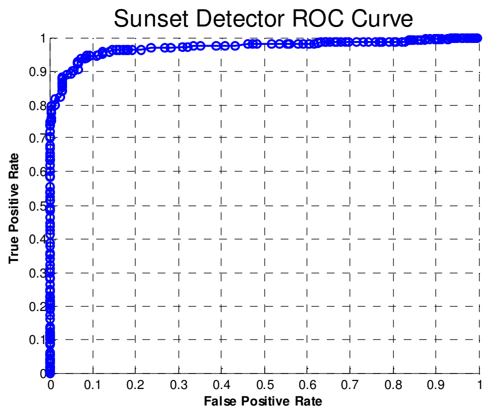


Figure 8: Improved ROC curve with a 10% overlap of grids, using 9x9 grids, and using alternative means of normalization.

RBF Kernel Distance γ Parameter	**18.4**
Decision Threshold	**-0.479**
Accuracy	**91.4%**
Precision	**89.0%**
Recall (True Positive Rate)	**95.3%**
False Positive Rate	**12.2%**

Table 4: Overall results after using alternative means of normalization, a 9x9 grid separation, and a 10% overlap of grids.

4) *Using a more difficult set of images as the test set*

To test the limits of our algorithm, we then used a more difficult set of images that contained black and white sunsets, upside down sunsets, non-sunset images with sufficient sunset features to throw of the classifier (e.g. erupting volcanoes), and sunsets under harsh weather conditions. For consistency's sake, we used the same kernel parameters, the same decision instance value (step 109), and the same extensions mentioned above.

Refer to Appendix B for sample of images correctly and incorrectly classified for a difficult set of images.

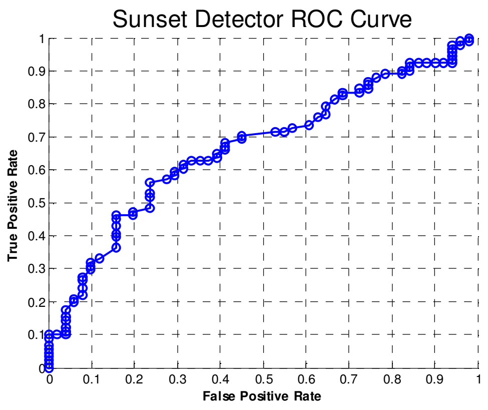


Figure 9: Generated ROC curve for the difficult set of sunset images using the same tuned parameters as mentioned earlier. To improve this ROC curve, we will have to select different RBF kernel γ parameter and generate the ROC curve for that fixed value. A better decision threshold that yields higher and better results can then be selected.

RBF Kernel Distance γ Parameter	**18.4**
Decision Threshold	**-0.167**
Accuracy	**63.4%**
Precision	**67.0%**
Recall (True Positive Rate)	**84.6%**
False Positive Rate	**74.5%**

Table 5: Overall results for the difficult set of images.

IV. FUTURE WORK

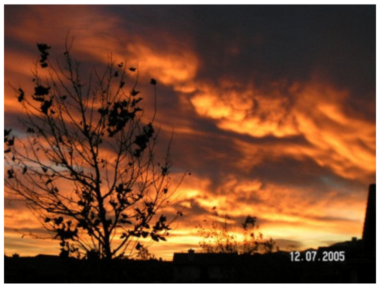
Looking back at our current progress, noting that if we had additional time to work, proposed improvements would include:

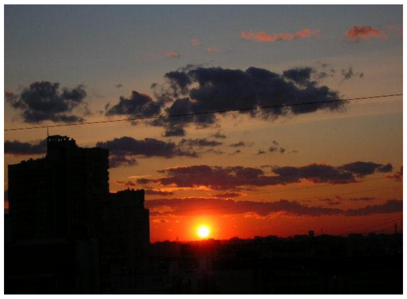
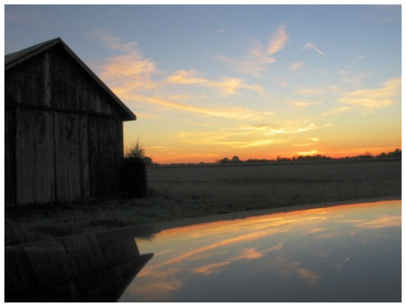
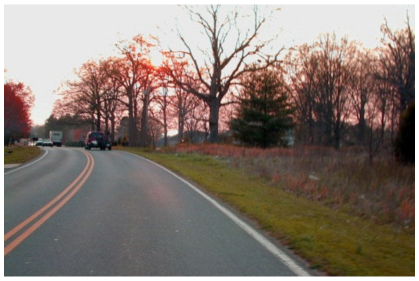
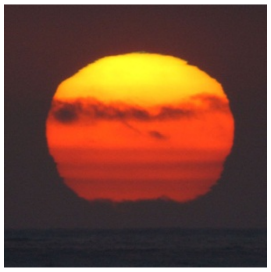
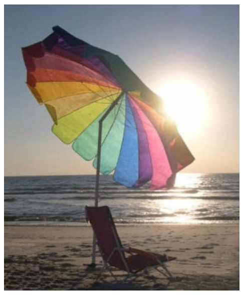
For an additional week we would like to experiment utilizing different methods for feature extraction. This would include parallel operation using separate color spaces and edge detection. One notable way we would like to extrapolate on our feature extraction methods, consists of a way to develop an image recognition pattern. This would locate particular bright spots and singular sources of vectored light or reflections of patterns that would be high candidates for sunset images.

If we had an additional year to work on the project, we would like to add further feature extraction on non-visual data elements, specifically types of metadata recorded during the time of the photo with modern technology and cameras. This would include ISO settings, shutter speeds, flash settings, face detection, key words or tags in the photo's title, and even geographical/timestamp data based on modern cell phones localization sensors. Using this additional data as added features, the SVM would help our algorithm understand the scenario and context of *how* and *where* the photo was taken. Images with geographical locations along the coastal regions and timestamps correlating with sunrise and sunsets along with our preexisting features extraction could significantly help classify candidates as actual sunsets.

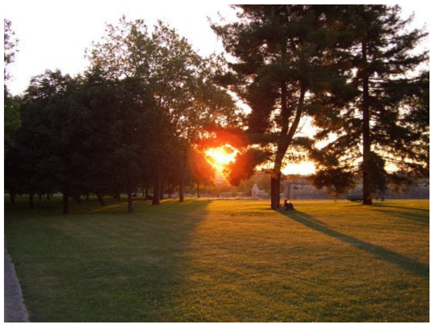
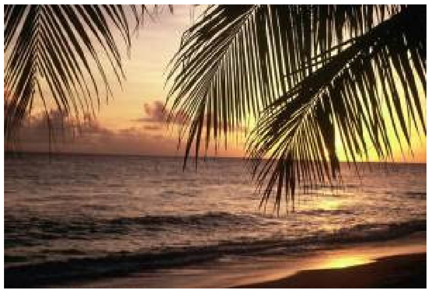
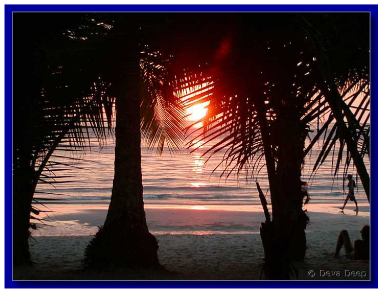
APPENDIX A: *Image results from the original set of test images*

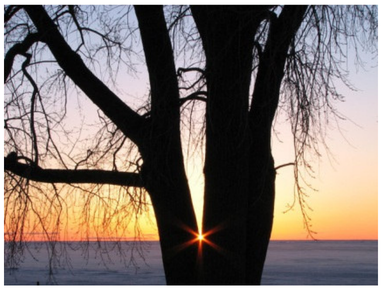
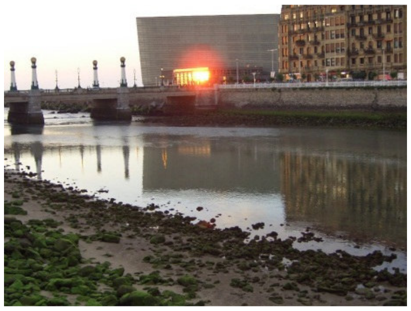
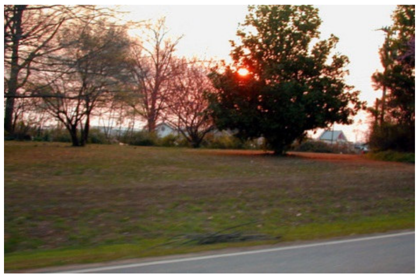
Sample of sunset images detected as sunsets



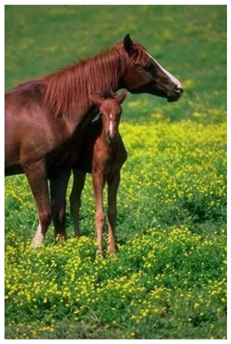






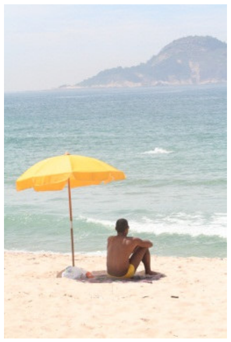
Sample of sunset images detected as non-sunsets

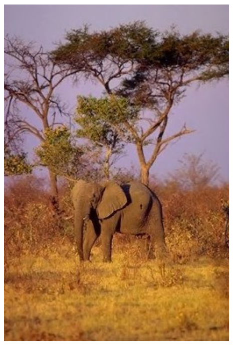




Sample of non-sunset images detected as sunsets



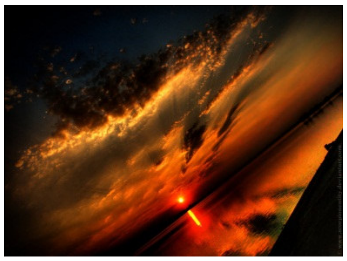


Sample of non-sunset images detected as non-sunsets

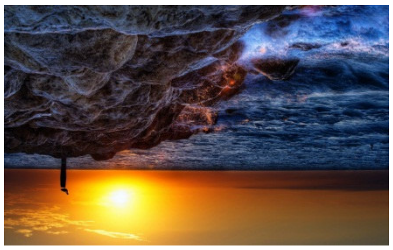
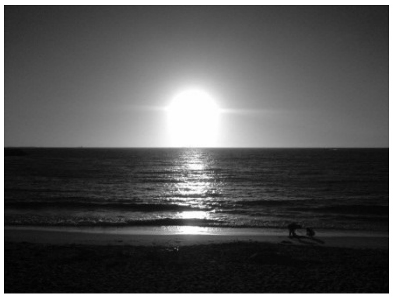
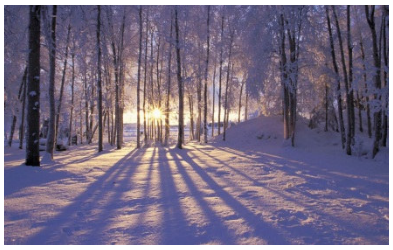
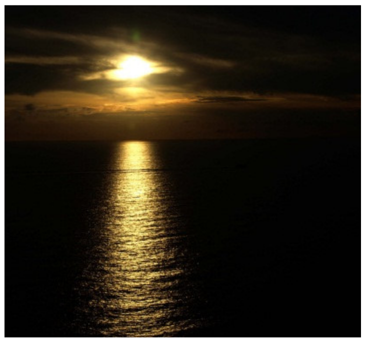
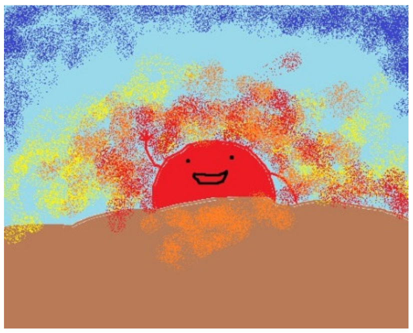
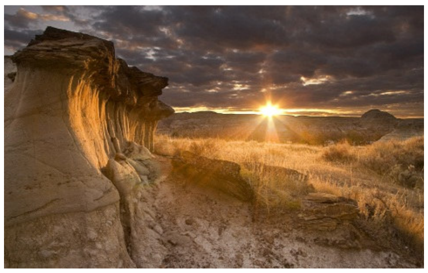


APPENDIX B: *Image results from a difficult set of images*

Sample of sunset images detected as sunsets



Sample of sunset images detected as non-sunsets

Sample of non-sunset images detected as sunsets

Sample of non-sunset images detected as non-sunsets