A SUNSET DETECTOR:
DETECTION OF SUNSET IMAGES IN ARBITARY IMAGES

Justin Fry
Nate Moore

Rose-Hulman Institute of Technology
Email: {fryjc, moorend}@rose-hulman.edu

ABSTRACT

We present a method for automatically classifying arbitrary images as either containing or not containing a sunset. Our method is based on the use of a support vector machine (SVM) to classify images. First, we split each given image into a set of 294 features. These features were defined by splitting each image into a seven by seven grid, and extracting six features from each cell (the average and standard deviation of three color bands). We extracted these features from a set of over 1,000 images pre-classified as either containing or not containing a sunset, then used the features to train our SVM. Next, we used a separate set of testing images to determine the accuracy of our SVM's sunset detection. After tuning the parameters of our SVM (our results are based on an SVM using a radial basis function (RBF) kernel with a σ of 1.6), we used it to classify the set of testing images. Our method appeared to classify the set of testing images relatively accurately; after tuning, our SVM classified images with a true positive rate (TPR) of 91.11% and a false positive rate (FPR) of 16.51%. As an extension to our work, we performed a similar classification of images using a neural network. It displayed a TPR of 81.4% and an FPR of 10.3% (without modified thresholds). Ultimately, the neural network outperformed the SVM. The total area under the ROC curve of the neural network was 0.9262, compared to the area under the SVM's ROC curve, which was 0.8381. This indicates a higher overall level of accuracy in the neural network.

Keywords— image recognition, sunset detection, computer graphics, support vector machines, neural nets

1. INTRODUCTION

Sunset detection is an interesting problem because it requires the detection of a rather poorly defined element of an image. Although a human can (generally) easily determine whether an image does or does not contain a sunset, an algorithm cannot do so simply by checking for certain color values or by examining a certain portion of an image. The definition of a "sunset" is not clearly defined, and the image data representing a sunset can be very inconsistent.

As an example of this difficulty, observe figures Figure 1Figure 2Figure 3,Figure 4, and Figure 5 below. They demonstrate the variability of images that may be considered to contain a sunset. Figures Figure 1Figure 2Figure 3 each contain a sunset, although the sun itself may be occluded and the predominant colors of the image may not be the traditional warm colors normally associated with a sunset. Figures Figure 3 andFigure 4 are both examples of images not containing a sunset. Although figure Figure 3 could easily be disqualified as a potential sunset due to its cool color profile and lack of a visible sun, figure 4 is not so easily discarded. It displays a warm color profile and even contains an artifact that is sun-like (the bright patch of smoke lying just above the hilltops).

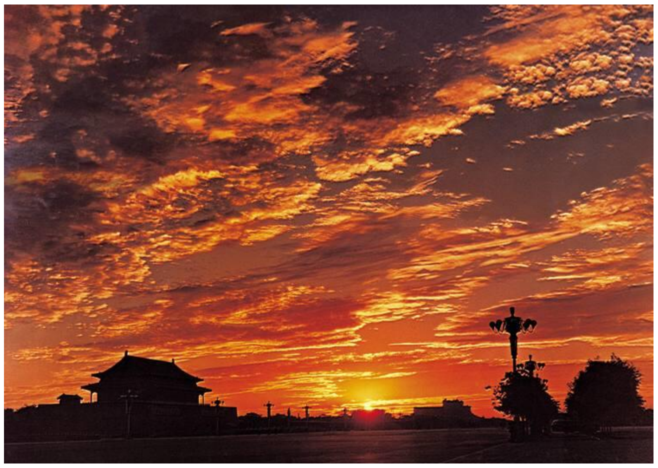

Figure 1: A rather clearly defined sunset.

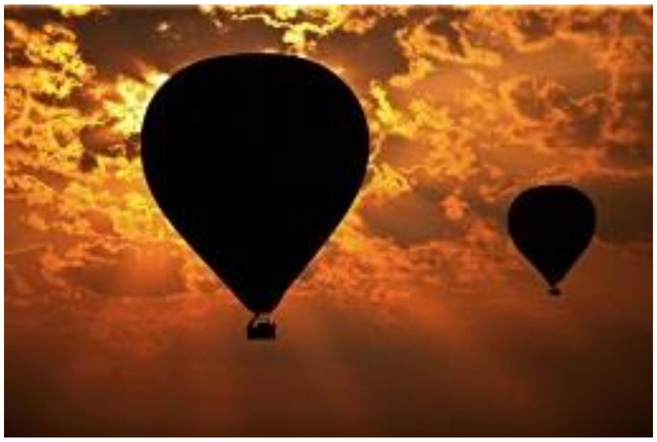

Figure 2: A less clearly defined sunset. Note occlusion of the sun by the hot air balloons.

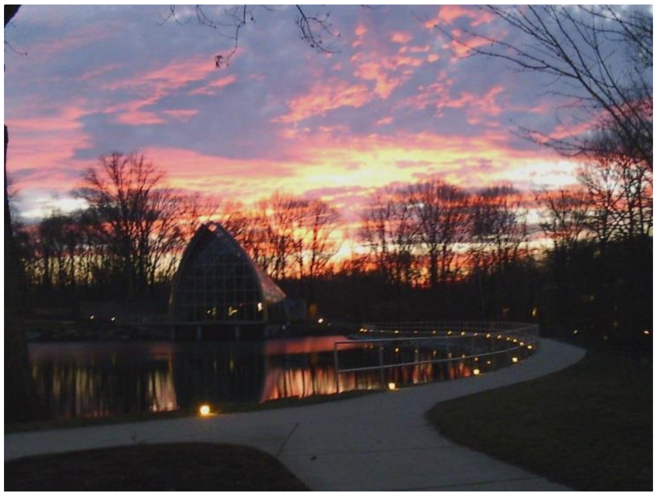

Figure 3: Another unclear sunset. Note both the occlusion of the sun and the predominance of cool colors throughout the scene.

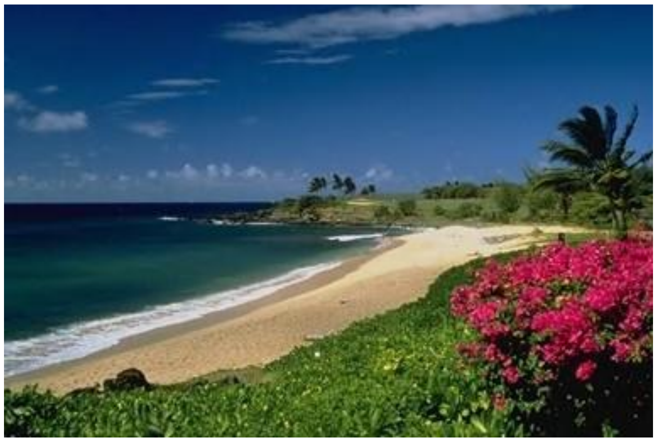

Figure 4: A rather clearly defined non-sunset. Note cool colors and lack of sun.

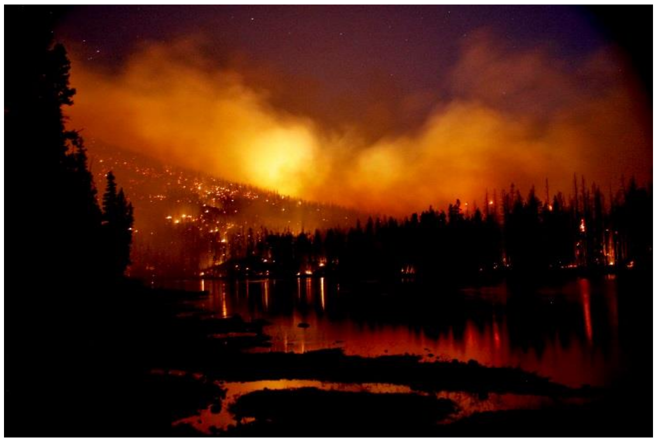

Figure 5: A rather unclearly defined non-sunset (actually a picture of a forest fire). Note warm colors and the presence of a sun-like artifact.

Clearly, the accurate classification of sunsets is a challenging problem. Attempting to solve it has implications for scene classification as a whole. Scene classification has many interesting applications, especially related to the classification of ambiguous features (as seen in this paper). Examples include the detection of tumors in breast ultrasound images [1] and categorization of scenes in nature (e.g., urban scenes versus beach scenes) [2].

2. NOVEL WORK

Our work is based on the use of an SVM to classify images as either containing or not containing a sunset. To improve on previous work, we used MATLAB to perform feature extraction in the LST color space and attempted to optimize the parameters of the SVM kernel we used.

Feature extraction was performed using a seven-by-seven grid, generating a total of 294 features per image (see section 3, “Process”). We used the LST color space in hopes of taking advantage of a smoother rate of change in color than the RGB color space.

We also hoped to improve our detection capabilities by tuning the parameters of our SVM. We tuned the σ value of the RBF kernel by testing a range of possible σ values and selecting the σ that maximized our TPR and minimized our FPR.

Finally, we also used a neural network to perform the same classification as our SVM. We hoped that this would provide a reference against which to test the performance of our SVM and potentially result in the discovery of a more effective classifier. Ultimately, our neural network was more accurate than our SVM in classifying images. See the Appendix for more information on the tuning process used in training and optimizing our neural network.

3. PROCESS

A critical component of our classification method involves the extraction of feature data from the images we process. Broadly, we divide each image into a seven by seven grid, creating 49 evenly-sized cells of the image, and then calculate the mean and standard deviation of each of the three color bands represented in each cell of the image. This results in the extraction of 294 total features per image.

Before attempting to extract features, we convert the color space of a given image from the RGB color space to the LST color space. This is accomplished using the following formulae expressing the relationship between each band of an RGB image and each band of an LST image (where L, S, and T are the bands of an LST image and R, G, and B are the bands of an RGB image).

$$\boldsymbol{L = R + G + B}$$
$$\boldsymbol{S = R - B}$$
$$\boldsymbol{T = R - 2G + B}$$

Equations 1, 2, and 3

With the image in LST format, feature extraction can now take place. We split the image into a seven-by-seven grid of equally-spaced "cells". Color space values lying outside of the boundary imposed by the constraint of having equal block sizes were ignored. In our algorithm, we chose to split the image into a grid starting in the upper-left corner of the image, ignoring pixels (if necessary) lying on the right and bottom borders. An example segmentation of an arbitrary image is shown in Figure 6.

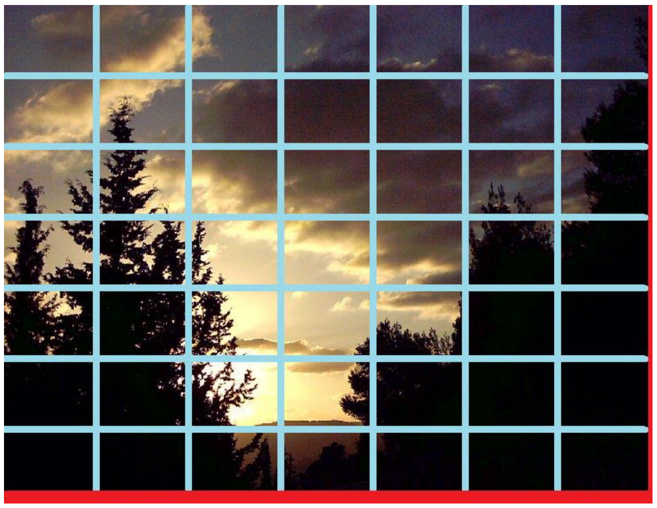

Figure 6: The seven-by-seven grid imposed on each image. Note that this image did not evenly divide into 49 cells. This resulted in the red-colored pixels being excluded from feature extraction.

Next, we calculated the mean and standard deviation of the L, S, and T bands of each of the 49 segmented cells. Note that L, S, and T bands each have a different range, unlike the R, G, and B values (which each have a range [0, 255]). L has a range of [0, 765], S has a range of [-255, 255] and T has a range of [-510, 510]. Since we would like to calculate and compare the means and standard deviations of these values, the variation in ranges could cause us to weigh each feature unequally. To circumvent this, we "normalize" the value of each band to a range of [0, 1]. This is accomplished by extracting the non-normalized features of every image in our test set, then determining the minimum and maximum values for the mean and standard deviation of each color band. Each value of a given feature is subtracted by the minimum value in the set of values for that feature (causing the minimum value to now equal zero), and then each value is divided by the maximum value in the set of values (causing the maximum value to now equal one).

Calculating the mean and standard deviation of each of the L, S, and T bands of 49 cells in a given image results in 294 total features. We used these features as inputs into our SVM (and, in our extension to our work, a neural network).

4. EXPERIMENTAL SETUP AND RESULTS

We used the process described in part 3 on a data set of 1,105 images describing both sunset and non-sunset scenes. The overall set of images was grouped according to Table 1: Grouping of input images., below.

Table 1: Grouping of input images.

Classification	Train/Test	Difficulty	Num. Images
Sunset	Training	Low	223
Sunset	Testing	Low	223
Sunset	Testing	High	92
Non-sunset	Training	Low	276
Non-sunset	Testing	Low	241
Non-sunset	Testing	High	50

Image groups classified as "training" groups were used in the initial training of the SVM. "Testing" groups were not involved in the initial SVM training, but only used to test the performance of our SVM. The difficulty of each group was subjectively determined by examining its constituent images (this process had already been performed on the data set we used). A total of 499 images were used as the training set. Not including difficult groups, a total of 464 images were used as the testing set; including difficult groups, a total of 606 images were used as the testing set.

After training, our SVM received a test image as its input, and output a classification as its output. An output of positive one indicated that the given image did represent a sunset, while an output of negative one indicated that the image did not contain a sunset. Our SVM also output an α value that reflected the confidence with which it had classified an image. The more positive an α value, the more confident our SVM was that the image was a sunset; the more negative, the less confident. This analysis assumes a threshold of zero when considering alpha. We had some success using thresholds other than zero to improve the performance of our SVM. These alternative thresholds were discovered through the use of a receiver operating characteristic (ROC) curve (see the Appendix for more information).

We evaluated three kernels for use in our SVM: the RBF, polynomial, and linear kernels. Initial experiments using each kernel showed that the RBF kernel had the highest baseline accuracy. We then used MATLAB's "svmcv" function to tune σ, the parameter associated with the RBF kernel (additional information on our parameter tuning is available in section 5.1 of the Appendix). We ultimately used a value of 1.6 as the optimal σ value for our SVM.

After tuning our SVM, we achieved a TPR of 91.11% and an FPR of 16.51% using a decision threshold of zero. Varying the decision threshold improved our overall accuracy, as shown in the ROC curve described in Figure 7: ROC curve for our trained and tuned SVM. The threshold varied from -10 to 10, with a step size of 0.05. The area underneath the SVM's ROC curve was 0.8381 (determined using a Riemann sum approximation).

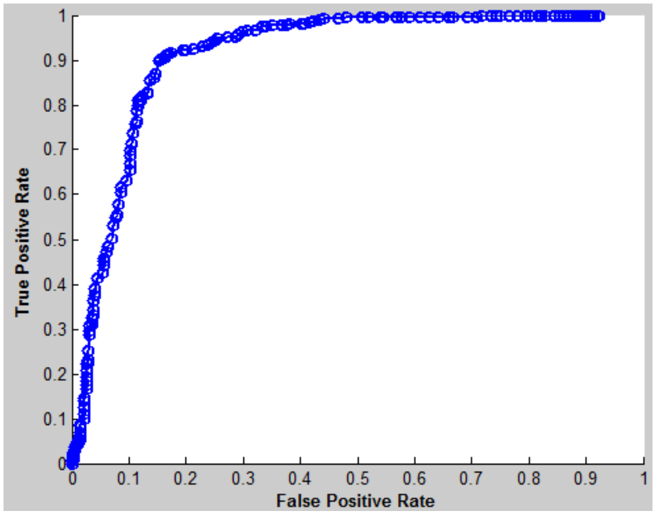

Figure 7: ROC curve for our trained and tuned SVM. The threshold varied from -10 to 10, with a step size of 0.05.

5. DISCUSSION AND CONCLUSION

Our SVM ultimately proved to be fairly accurate in classifying sunset and non-sunset images. It correctly identified 91.11% of the sunset-containing images and limited false identifications of sunset images to 16.51%.

Figures Figure 8Figure 9Figure 10Figure 11 show the most and least accurate positive and negative classifications generated by our SVM. Our SVM classified Figure Figure 8 with the most confidence of any sunset (α = 0.042). It is observed to display very warm colors and a bright sun object. Figure 9 was the image classified most confidently as a non-sunset (α = -14.29). It is composed of very cool colors and has no obvious sun-like artifacts.

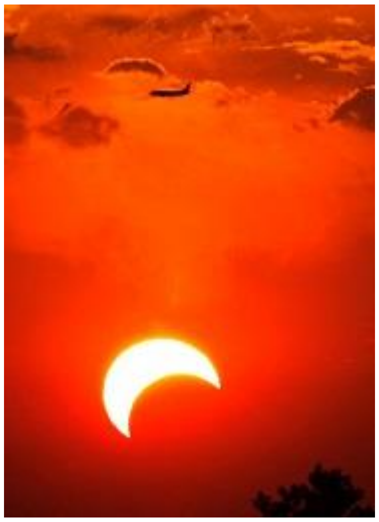

Figure 8: Most confidently accurately classified sunset image.

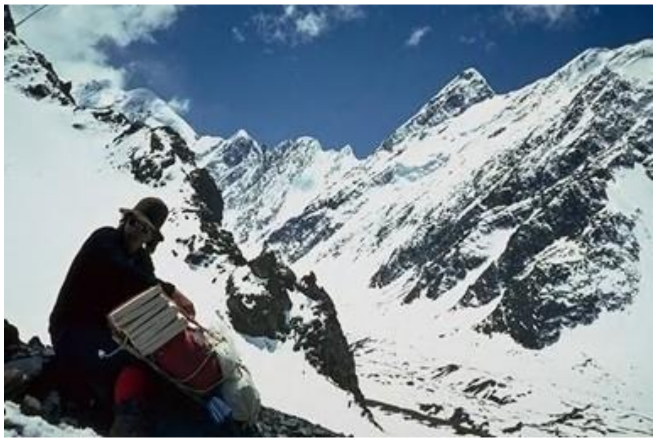

Figure 9: Most confidently accurately classified non-sunset image.

Figure 10 displays the non-sunset image that our SVM most confidently inaccurately classified as a sunset (α = 0.0026). It is composed of warm colors and a large moon that could be mistaken for a sun object. Figure 11 shows the sunset image our SVM most confidently classified as a non-sunset (α = -12.35). It displays relatively cool colors and the sun object is occluded by a tree (which itself takes up a large portion of the image).

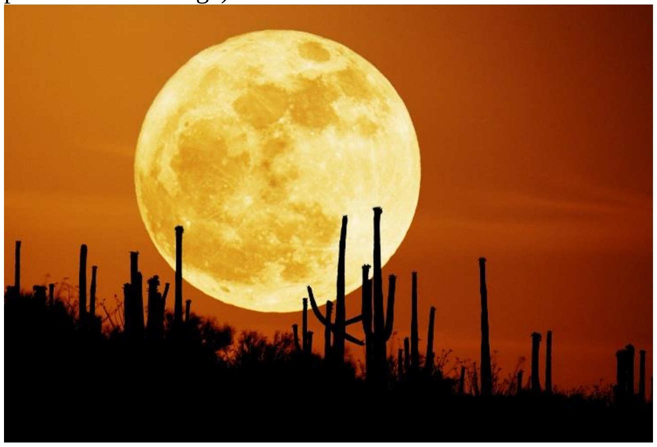

Figure 10: Worst +

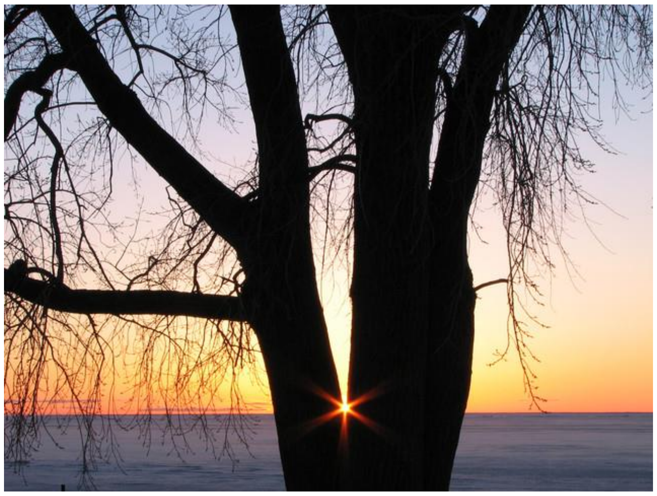

Figure 11: Worst -

Much of the difficulty in constructing our SVM lay in its optimization. Determining the correct initial ranges to begin optimizing σ involved a great deal of trial-and-error. We also encountered minor technical difficulties in ensuring enough identifying information was transmitted from the feature extraction stage. To save processing time, the features of each image in our entire image set were extracted and saved to disk. This did allow for faster use of our classifying algorithm, but we would frequently need data we had lost by only storing the extracted features. For example, only storing feature data caused us to lose the subjective difficulty rating we had associated with each image (see Table 1 in Section 4). To resolve this problem, we created separate vectors containing the data we wanted to transmit, and wrote them to disk along with the feature data.

In the short term, future work in sunset classification could be done within the paradigm of SVM usage. We believe further optimization of the SVM σ value is possible by testing greater ranges of values with a lower step size. However, the lack of success we had testing ranges narrower than [0.5, 3] may indicate that there is not much performance left to be gained in this area (see section 5.1 of the appendix for more information).

An additional potential short-term strategy could also lie in the usage of alternative color spaces. Greater variation among sunset images could perhaps be discovered by choosing a color space other than RGB or LST.

Finally, a greater number of training images could be used in training our SVM. A relatively small training set (of 499 images) was used to train our SVM.

In the long term, alternatives to the usage of an SVM could be explored. As mentioned in Section 2 (and further explained in section 5.2 of the Appendix), we also examined the use of a neural net in classifying sunset images. A wider study could be made of a great variety of classifiers. This would require a large time investment (assuming each classifier is properly tuned, as we attempted with our SVM classifier), but could potentially identify a classifier better suited to sunset classification.

5. APPENDIX

5.1. SVM Tuning

As mentioned in section 4, we made use of the “svmcv” function to tune our SVM. This was accomplished by dividing our training image group into discrete sets (named “folds”) and iteratively designating one of these folds as a “validation” set. The σ of our RBF kernel was then varied across a given range using an arbitrary step size (the value by which the current tested σ was multiplied by at each iteration), and the σ corresponding to the most accurate performance of the SVM on the designated “validation” set was stored. The process repeated until each fold had been designated as the validation set exactly one time. The lowest σ value found during this process was then reported as the optimal σ for the specified range.

We first used svmcv to examine a wide range of potential σ values. The initial search range used was [1, 100], with a step size of 1.25. This caused svmcv to report an optimal σ of approximately 1.8. We then narrowed our range, searching for values within the range [0.5, 3]. This resulted in the report of an optimal σ of 1.6. Furthering narrowing our search range failed to produce σ values resulting in a higher accuracy, so we accepted 1.6 as an optimal value for σ.

5.2. Neural Network Tuning

As an extension on the work done with support vector machines, we additionally developed an artificial neural network (ANN) method of sunset detection. The “patternnet” variant of ANN, a partially pre-built two-layer feedforward ANN consisting of sigmoid transfer functions in the hidden and output layers (included with MATLAB and designed for pattern recognition), was used due to its suitability for our problem domain. In consideration of memory constraints, 100 neurons were used in the hidden layer.

Our primary considerations with regard to configuration and optimization were the training function and parameters. For the training function, a resilient backpropogation training algorithm was used due to its strong performance in pattern recognition tasks. A resilient backpropogation algorithm only takes into account the sign of the partial derivative of the classification error function, ignoring the magnitude. This eliminates the main problem associated with steepest decent algorithms when encountering sigmoid functions, which is that the gradient can have a very small magnitude and therefore very little effect on the weights and biases. For the training parameters, the default values consisted of a minimum gradient of e^{-06}, a learning parameter of 0.01, a max number of validation failures at 100, maximum epochs at 1,000, and data division of 70% training, 15% validation, and 15% test. Because we were provided with our own test data set, there was no need to allocate a great amount of data from our training set to testing. This allowed us to reallocate the data division to 80% training, 20% validation and 0% test, giving the training function a much larger useful set. In order to prevent premature conclusion of our training process, the minimum gradient and maximum validation failures were changed to 0 and 100 respectively. The learning ratio was found to be ideal at its default value with brute force testing.

With the ANN optimally configured for the purpose of sunset detection, we began training using our training image set and then testing with our test image set. We used the performance analytics provided with MATLAB alongside our own analytic work to inspect the results of the ANN. With default biasing and thresholds, the performance was slightly underwhelming with a significantly lower true positive rate at 81.4%, and false positive rate at 10.3% than the SVM’s 91.11% and 16.51%. However, varying the threshold value through the use of an ROC curve showed that the neural network gave more accurate overall results. The area under the neural network’s ROC curve was found to be 0.9262, which is greater than that of the SVM (0.8381).

Based upon these results, we believe that an ANN is well suited to the task of sunset detection. Given that this was simply an extension to our larger development of an SVM method, we were not able to explore every possible optimization option available to us, and it is possible that, with more time, an even more successful ANN could be found.

. 6. REFERENCES

[1] J. Ding, H. D. Cheng, J. Huang, Liu, J. Liu and Y. Zhang, "Breast Ultrasound Image Classification Based on Multiple-Instance Learning," *Journal of Digital Imaging*, vol. 25(5), pp. 620-627, 2012.

[2] M. R. Boutell, J. Lueo, X. Shen and C. M. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757-1771, 2004.