SUNSET DETECTOR

Duy Le, Tyra Correia

CSSE463 Image Recognition

24 July 2024

ABSTRACT

With the growing prominence of automation in image processing, companies such as Eastman Kodak have looked towards employing machine/deep learning techniques to improve the efficiency of their editing software. Background images play a crucial role in determining the editing techniques used in images. Sunsets, in particular, have a substantial impact on the form of color correction used in editing. This project therefore addresses the challenge of automatic sunset detection in photos uploaded by customers of Eastman Kodak Company. Accurate sunset detection is crucial for photo-editing companies because standard photo-finishing algorithms often incorrectly remove the warm colors of sunsets. This study explores two techniques for this task: traditional image recognition using feature extraction and SVM, and deep learning with pre-trained convolutional neural networks (CNNs) through transfer learning. This approach offers a promising solution for enhancing photo-editing algorithms and improving customer satisfaction.

1. INTRODUCTION

Automated photo editing aims to improve the quality of images quickly and consistently across large volumes of photos. Such programs must use feature extraction and classification algorithms in order to guide the editing techniques used. Feature detection is an essential part of photo-editing processes for companies like Eastman Kodak. When applying color correction to digital images, detecting background features such as sunsets can ensure vibrant and warmer colors are enhanced alternatively. Applying background classification to images can furthermore create groups for large assortments of images taken in different locations belonging to a single client. Scene classification therefore aids in automated photo editing and organization.

Background detection is particularly challenging due to the variability in the color, intensity and composition associated with images in the same category. With respect to the sunset detection algorithm, non-sunset photos with warm tones may be misclassified leading to false positives. As per Fig 1, an example of a false positive case notes that a cat was wrongly classified as a sunset due to the orange tones of its fur. The false identification of such sunset images can therefore be attributed to the similarities in hues.

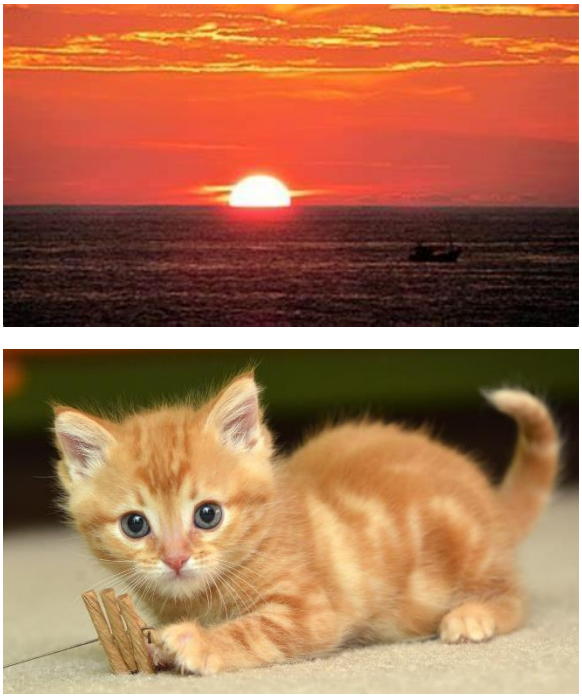

Fig 1. (a) Sunset Image (b) Falsely classified cat image

The solution proposed to this problem leverages traditional machine learning techniques combined with deep learning approaches. By using feature extraction and Support Vector Machines alongside transfer learning using pre-trained CNNs, it is possible to create an effective sunset detection algorithm. The inputs used to train and test the algorithm will be a set of digital images obtained from Flickr. The resulting output is a categorical classification used to identify whether an image is a sunset or not. The goal of the

project is therefore to develop and evaluate the accuracy of the image recognition system. In doing so, such an algorithm would prove highly beneficial to Kodak for utilization in any automated photo-editing programs.

2. PROCESS

The method used to address feature extraction and classification of sunset vs non-sunset images can be broken down into five key steps. The image is divided into subgrids, wherein their LST values are extracted. Statistical data regarding the mean and standard deviation are computed and placed into a feature vector matrix. The matrix may then be normalized to ensure standardization and run through a support vector machine for classification. The ROC curves for the resulting output may be used to assess the subsequent performance of the algorithm by tuning the box constraints and kernel parameters.

2.1 Feature Structures

To improve the efficiency of the machine learning classifier used in the SVM it is necessary to preprocess the images via feature segmentation and statistical calculations.

The algorithm initially segments the image into a 7 x 7 grid. The use of a 7 x 7 grid allows for data to be grouped into smaller pieces when processed through the SVM. To factor into account image dimensions that may not be divisible by 7, a floor function was used to compute the subgrid width and height. When parsing through the subgrids, the function looped over columns and rows. The starting x or y value was set to the current row -1 multiplied by the subgrid width or height +1. Therefore, given that the current subgrid is 3, we may find the last pixel location from subgrid 2 and increment by one to find the first value of the pixel in subgrid 3. In order to account for edge pixels at the end of a column or row, an if statement was used to ensure that the end value was set to the edge value. This ensures that the edge pixel is always accounted for due to errors in divisibility.

2.2 RGB to LST

Prior to statistical data being extracted from the subgrid, the LST values are calculated based on the RGB channels of the original image. Utilizing LST values is an alternative to working with the RGB space. The formula used for this conversion can be found below:

$$L = R + G + B \qquad \text{(Equation 1. L formula)}$$

$$S = R - B \qquad \text{(Equation 2. S formula)}$$

$$T = R - 2G + B \qquad \text{(Equation 3. T formula)}$$

An example of the conversion between RGB and LST values can be found in the table below:

Table 1. RGB vs LST comparison

Channel	Value
R	123
G	54
B	147
L	324
S	-24
T	162

The L value determines the total RGB composition in the image which indicates the brightness across all the channels. The S value measures the difference in red and blue channels which shows the variation between the red and blue hues. The T value may measure the contribution of the green channel to the overall image as it is weighted more than the red and blue channels. It therefore allows a comparison between the green and magenta channels. As yellow is composed of red and green channels, accounting for this difference is especially important when remapping the color space.

2.3 Feature Vector

Statistical data may exhibit less variance when applied to smaller subgrids as well. The mean and standard deviation were then calculated for the corresponding LST values found and placed into a feature vector with the following format:

$[img_1m_L11,\ img_1s_L11,\ img_1m_s11,\ img_1s_s11,\ img_1m_T11,\ img_1s_T11,\ img_1m_L12,\ \ldots\ ;$

$img_2m_L11,\ img_2s_L11,\ img_2m_s11,\ img_2s_s11,\ img_2m_T11,\ img_2s_T11,\ img_2m_L12,\ \ldots\ ;$

$img_Nm_L11,\ \ldots$

An example of the feature vector of the first block in image 1142546537_d51a76ee4c_z.jpg prior to normalization can be found in Table 2.

Table 2. Mean and Variation for segment 1 in 1142546537_d51a76ee4c_z.jpg

Segmentation (49 blocks)	Bands	Mean and Deviation	Data
Block 1	L	Mean	348.5511
		Deviation	31.1488
	S	Mean	-44.4212
		Deviation	16.3656
	T	Mean	-9.3548
		Deviation	4.9335

The algorithm therefore produces a 294-dimension vector which is stored into a larger 294 x nImages vector.

2.3 Normalization

Normalization is essential in order to ensure all features are weighted on a consistent scale. The use of normalization allows the system to prevent objects from dominating images or its sensitivity to outliers in the image. Additionally, the ranges for the LST values greatly differ from each other. In order to address this issue, normalization is a pivotal tool that establishes a baseline to standardize the values obtained upon converting to the LST space. The images were normalized by looping over the six feature types associated with a subgrid. The following formula was then applied to each value from the six features:

$$x_k = (x_k - x_{\min}) / (x_{\max} - x_{min})$$

(Zhang, 202X)

The feature matrix was then updated with the normalized values and cast typed to a double.

2.4 SVM Analysis

Once the preprocessing steps have been completed, the data may be run through a support vector machine. A support vector machine with a kernel parameter of 5 and a Box constraint of 30 was used to produce a net with a given hyperplane. The predict function then operated on the net and the validation set to yield a set of detected classes and their corresponding distances.

After obtaining the detected classes, the labels of the classes, which were “sunset” and “non-sunset” were converted to the values 1 and -1 as the binary classification is preferred for calculating the confusion matrix. The converted classes then were used to find accuracy as well as the confusion matrix.

The code was run at multiple kernel widths and box constraints to obtain the most fine-tuned hyperparameter. We would experiment with those different hyperparameters on the validation set. A fine-tuned hyperparameter would ensure reasonable accuracy without using too many support vectors. As a result, it was set to a kernel parameter of 7 and a box constraint of 100. It is also possible to obtain the number of support vectors by obtaining all the alphas, and counting the values of the support vectors that are greater than 0.

2.5 ROC Curves

The use of ROC curves allows one to vary the thresholds of a given SVM output, therefore plotting the true positive vs false positive rate. The SVM trained by the fine-tuned parameters would be used to predict the test set. The detected classes obtained from the prediction would be used to calculate the accuracy and confusion matrix, which includes TP, FN, FP, and TN.

The set of thresholds used for this ROC is {-5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5}. The detected classes were reassigned according to each new threshold. The set of converted classes of each threshold were then used to calculate the TP and FP.

Plotting the ROC curve for the sunset detector compares how many sunsets were accurately detected vs how many sunsets were misclassified for different threshold values. For a threshold of -5, the TP and FP are 1 and 0.932 respectively. The accuracy for this threshold is 0.533. On the other hand, the threshold -1 yields an accuracy of 0.856.

3. CLASSIFICATION

3.1 Support Vector Machines

Support Vector Machines are a form of classification analysis in machine learning that is most effective when applied to high dimensional data. The theory behind SVMS involves locating the optimal hyperplane that yields the maximal margin of separation between two desired classes (Hearst et al, 1998). The hyperplane can be determined by the support vectors which are points of data that can be found closest to the decision boundary. Maximizing the margin of the distance between the hyperplane and the support vectors therefore ensures that a classifier is better able to have a more generalized model that can adapt to more variations in the data and is more robust to noise. SVMs have the capacity to produce non-linear hyperplanes which can map input data into a higher dimensional space (Hsu, 2016). These non-linear hyperplanes may be found using radial basis functions as used in the sunset detection algorithm as per the formula below:

$$K(x, x_i) = \exp\left[-\frac{|x - x_i|^2}{\sigma^2}\right]$$

(Zhang, 2012)

The box constraint in the kernel function therefore represents a tradeoff between maximizing the margin vs reducing the level of classification errors on the data. Higher box constraints often lead to fewer misclassifications but more overfitting. Lower box constraints allow for more misclassification but have a wider margin. The box constraints of 100 were found to be most optimal in the case of the sunset detector.

It is also possible to adjust the kernel size to influence the area of impact on the points around a decision binary. Using larger kernels can result in a smoother boundary that considers more distant points. Using a small kernel indicates that the decision boundary will only account for nearby points. The kernel width therefore plays an important role when mapping the hyperplane to a higher dimensional space in order to draft decision boundaries and their corresponding support vectors.

3.2 Convolution Neural Networks

Convolution neural networks employ deep learning models best suited for image classification. The architecture of a CNN can be composed of several layers such as the convolutional layer, the activation layers, the pooling layers and the fully connected layers. The convolutional layer is able to first apply filters to an input image and consequently extract its feature map. The activations layer uses ReLu activation function and implements thresholding at zero which is useful when working with noisy data. The pooling layers then reduce the spatial dimensions of the feature maps therefore capture the most relevant information to the classifier. The fully connected layers finally use the extracted features to make classification decisions.

The two sets of techniques implemented to address sunset detection via the use of convolution neural techniques capitalize on transfer learning and feature extraction as methods to aid classification. Transfer learning leverages knowledge acquired from pre-existing datasets and allows for modifications to be made on the last few layers of the network to tune the network towards the classification problem. It is hence possible to preserve deeper layers responsible for extracting features like edges and shapes while customizing the final layers. The benefit of this approach allows one to save time in training a network from scratch and the need for large amounts of data to enable this process.

Feature extraction is another significant tool in image classification that allows an algorithm to be run up to a certain layer before terminating. Features within a convolutional neural network may contain high level or low-level features. Deeper layers may typically contain high level features, whereas low level features may be stored in shallower layers. Classifications may be made by resizing the input images, extracting their features and utilizing them as inputs to an SVM. The feature extraction method often requires less computational time and is highly effective on smaller datasets.

4. EXPERIMENTAL SETUP

The images in the dataset have been obtained from a collection of online digital images from the website Flickr. The selection of sunset images from the sunset central pool contains 19379 sunset images captured by 1600+ photographers. The images were filtered so that black-and-white photos, strong reflections, close-ups, or images that lacked a yellow-toned sun were removed. Around 460 photos reflected these non-detectable scenes. The sunsets used are therefore intended to most resemble background scenes typically present in evening time photography sessions.

The non sunset photos were obtained from the photography club of Flickr pool. The collection of non sunset images included 9585 photos captured by 400 photographers. The first 800 photos were used for the training set. The subsequent 300 photos composed the validation set and the remaining 500 photos were used for the test set. The number of images used in the training set was the highest to ensure the algorithm is capable of learning using a large data pool. The validation pool, while smaller, allows for the system to test the accuracy of the image classification algorithm.

The image classification system developed has the capacity to provide high overall accuracy using feature detection. The sunset detection program is capable of being applied to a variety of sunset and non-sunset images.

Upon running the program, images are divided into 7x7 subgrids. Segmenting the image in this manner is capable of preserving finer details across different regions in the image by better improving the grouping of similar object groups. This technique may however be sensitive to the alignment of objects in the image which may be split across multiple subgrids. Images with similar information run the risk of being processed redundantly due to uniformity across the subgrids.

The system is capable of extracting the LST values, then calculating the mean and standard deviation of them respectively. Utilizing the mean and standard deviation is a simple statistical baseline for the average pixel information and the relationship it has to other pixels in a grid. However, the mean and standard deviation may fail to consider drastic changes in lighting or color within an image. The lack of robustness in this technique provides limited information about the input image. Incorporating more advanced statistical information may consequently yield better results.

Prior to running the feature vector through the SVM, the feature is normalized to ensure that the algorithm is scale invariant. In this circumstance, the feature vector loops over the features of a given subgrid and normalizes them to a range between 0 and 1 by subtracting the min value from the given feature and dividing it by the max value of its associated matrix. As the kernel function used in the support vector machine relies on inner products, normalization is therefore a necessary task (HSU, 2016). However, the use of normalization may potentially result in loss of information depending on the technique used.

With SVMs it is possible to determine classifications of images. SVMs may use kernel functions to prevent overfitting in high-dimensional feature spaces. It is possible to use radial kernel functions to develop non-linear hyperplanes that aid in this process. Furthermore, these radial kernel functions have parameters such as the kernel width and sigma that can easily be tuned to change the accuracy produced. Moreover, the system is capable of yielding confusion matrices to the output of the sunset detection algorithm. The downside of utilizing SVMs can include its sensitivity to the parameters that are selected for the kernel and the sigma value. It is therefore necessary to tune the network according to the input data.

It is additionally possible to generate an ROC curve in order to visualize the performance of the system. The curve additionally represents the tradeoff made between the true positive and false positive rates. By utilizing the ROC curve, it is possible to make decisions on the thresholds that are selected. This provides a better indicator of the impact on the output when adjusting the margin for classification.

5. RESULTS

5.1 SVM Results

To improve the accuracy of the sun detector while using as few support vectors as possible, we experiment with many different values for the box constraint and kernel parameters on the validation set through a wide search. The results of the wide search have been narrowed down to a box constraint of 100 and a kernel parameter of 7 as per Table 3. Although they yield less accuracy than that of training with the kernel parameter of 2, the chosen hyperparameters use much fewer support vectors to avoid overfitting.

Table 3. Wide Search: the accuracy and number of support vectors given box constraints and kernel parameters

Kernel parameter Box constraint	9	7	5	2
100	0.8967; 388	0.9167; 386	0.9117; 386	0.9267; 645

80	0.9017; 387	0.9067; 384	0.9083; 389	0.9267; 645
50	0.9000; 405	0.9033; 396	0.9117; 399	0.9267; 645
30	0.9000; 416	0.9017; 401	0.9167; 409	0.9267; 645
20	0.8967; 419	0.9050; 411	0.9117; 411	0.9267; 645

Table 4. Fine Search: the accuracy and number of support vectors given box constraints and kernel parameters

Kernel parameter Box constraint	7
101	0.915000; 387
100.75	0.915000; 387
100.5	0.915000; 386
100.25	0.915000; 386
100	0.916667; 386
99.75	0.916667; 386
99.5	0.916667; 386
99.25	0.916667; 386
99	0.916667; 386

After fine-tuning with different data as per Table 4, the box constraint with the value of 99.75 and the kernel parameter with the value of 7 give reasonable accuracy. With these hyperparameters, the accuracy of the sun detector on the validation test is 0.9167 with the number of 386 support vectors over 1232 training images.

On the other hand, using the fine-tuned hyperparameters for the testing set yields an accuracy of 0.839 on classification. Although the accuracy of the test set is lower than that of the validation set, the margin of difference is not too large. If the accuracy of the test set was a lot lower than the validation set, then this would indicate overfitting.

From the ROC curve, we found out that the threshold -1 would yield the greatest accuracy in the set, which is 0.856. It is also visualized as the data point that is closest to the coordinate (TP=1, FP=0)

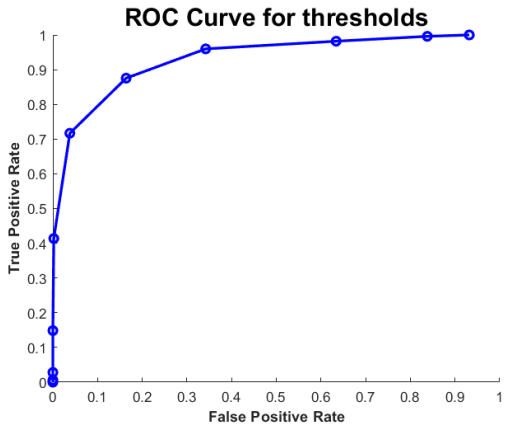

Fig. 2: ROC curve for thresholds -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5

5.2 CNN Results

5.2a Transfer Learnings

The transfer learning process used the pretrained network AlexNet and replaced the last three layers with a fully connected layer, softmax layer and classification layer. It is possible to customize the layer by setting the fully connected layer to have the same number of classes as the dataset, which in this case is two. Adjusting the rate at which the weights are learnt at the fully connected layer allows the network to learn new layers at a faster rate. The mini batch size was set to 15 and the epoch size was set to 8. As the epoch size was increased, so was the batch size to prevent the algorithm from running too slow. The learning rate on the last three layers was set to 20, and there were 82 iterations per epoch.

The overall time to predict the output ended up being about 15 mins when run on the Intel ® UHD Graphics card. The resulting accuracy on the validation set was 95.3% and the accuracy on the test set was 94.3%. Given the run time, the transfer learning method yields relatively accurate results.

The epoch graph can be found in Fig 3 below:

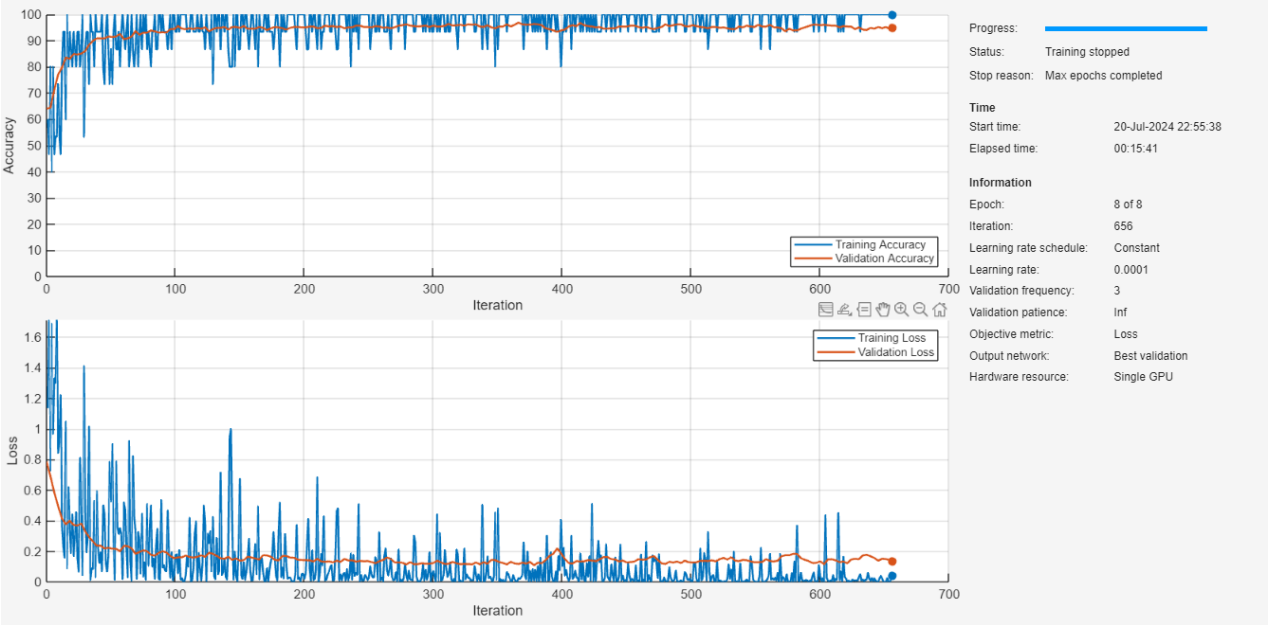

Fig 3. Accuracy vs Iteration per Epoch

The ROC curve was found as per Fig 4 and indicates promising results. Due to the shorter run time, the axes on the curve have been clipped to a False Positive rate of 0.12. The relationship between the True positive rate at 0.98 and the False positive rate of under 0.1 therefore indicates good performance of the CNN.

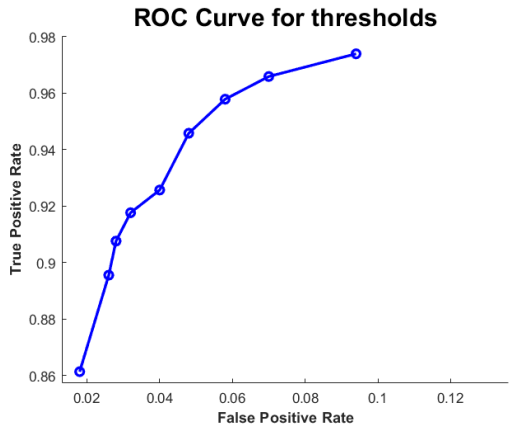

Fig 4. ROC curve for Transfer Learning

As per the confusion matrix in Table 5, a result of 286 true positive and true negative values were accompanied by 14 false positive and false negative values.

Table 5. TP,FP,TN,FN values for Transfer Learning

	Positive	**Negative**
True	286	286
False	14	14

5.2b Feature Extraction

The process of feature extraction involves using convolutional neural networks to extract features by running them up to a restricted amount of features. Once features have been extracted they may be classified by the use of support vector machines. Feature extraction through the use of convolutional neural networks may use pre-existing networks such as AlexNet and are often useful when used in combination with traditional machine learning models.

Feature extraction was implemented using feature extraction by using activations on the layer fc6. When running the algorithm with the fully connected layer fc7, the accuracy found was around 81%. As fc7 is at a higher layer, it is likely to contain high level features. In order to improve the accuracy, activations were run on the fully connected layer fc6. A key aspect of the feature extraction process involves normalizing the

feature set. Failure to normalize the feature set yielded accuracy rates of 50%.

The inputs of the feature extraction algorithm were pulled into the SVM classifier, which utilized the kernel parameter of 9 and the box constraint of 15 to achieve an accuracy 96.62% with 270 support vectors during a wide search. The results of the wide grid search to locate the ideal hyperparameters can be found in Table 6.

When performing a finer grid search, the box constraint of 14.75 and the kernel parameter of 9 was able to provide the same accuracy but with one less support vector. The results of the fine search can be found in Table 7.

Table 6. Wide Search: the accuracy and number of support vectors given box constraints and kernel parameters

Kernel parameter Box constraint	9	7	5	2
100	0.955000;	0.955000; 295	0.941667; 340	0.500000; 1079
80	0.955000; 282	0.955000; 295	0.941667; 340	0.500000; 1079
50	0.953333; 284	0.955000; 295	0.941667; 340	0.500000; 1079
30	0.953333; 283	0.956667; 298	0.941667; 340	0.500000; 1079
20	0.958333; 269	0.956667; 303	0.941667; 340	0.500000; 1079
15	0.961667; 270	0.961667; 294	0.943333; 339	0.500000; 1079

Table 7. Fine Search: the accuracy and number of support vectors given box constraints and kernel parameters

Kernel parameter Box constraint	9
16	0.961667; 274
15.75	0.961667; 273
15.5	0.961667; 272
15.25	0.961667; 272
15	0.961667; 270
14.75	0.961667; 269
14.5	0.960000; 269
14.25	0.960000; 269
14	0.960000; 272

The resulting ROC curve for feature extraction can be found below in Fig 5 and showcases a high true positive rate for low false positives.

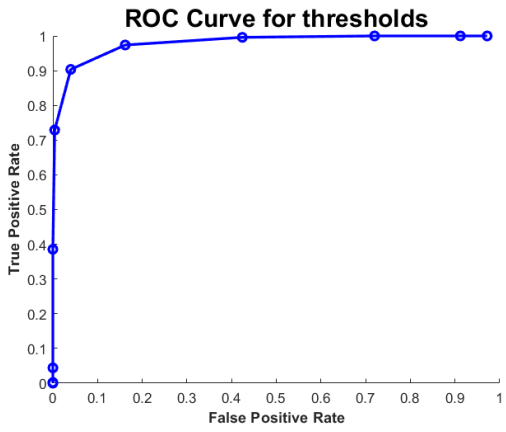

Fig 5. ROC curve for Feature Extraction

6. DISCUSSION

6.1 SVM

The LST-SVM trained with the fine-tuned hyperparameters has an accuracy of 0.839 on the test-set images. This means that 83.9 percent of the images are classified correctly, while the others are falsely predicted. The following images are sample cases that represent successful and unsuccessful classification by the SVM.

True Positive Images:

The images below have correctly been classified as true positive images. The first image in Fig 6 has a score of -1.942 indicating that the image is far away from the margin, making it more difficult to identify. This image does not have a sun which might indicate why its score is further away from the margin. The second picture in Fig 7 has a distance of -0.1427, suggesting it is close to the margin, and therefore easier to classify.

Fig 6. True Positive Image 1

Score: -1.942868787878559

Fig 7. True Positive Image 2

Score: -0.142715213125263

False Negative Images:

While the system accurately identified true positive and true negative images for most images, a small portion of false positives and false negatives were present. The first false negative picture in Fig 7 demonstrates a large amount of backscatter making it difficult to identify the sunset in the image. Additionally, the algorithm may classify the backscatter as objects rather than a background in the image. The foreground of the netted face also makes classification difficult on this image. The image has a score of 2.1875 which suggests that the image is also further away from the decision boundary and harder to classify as a sunset.

The second false negative image in Fig 8 has been misclassified due to the lack of sun present in the picture. Despite the sky being primarily composed of pinks and oranges, the image has been classified as a non-sunset. This is also evident in the image's score of 0.6139 which is close to the decision boundary. As the image does not have multiple objects in the foreground and effectively captures the scene, the image's score is closer to the margin.

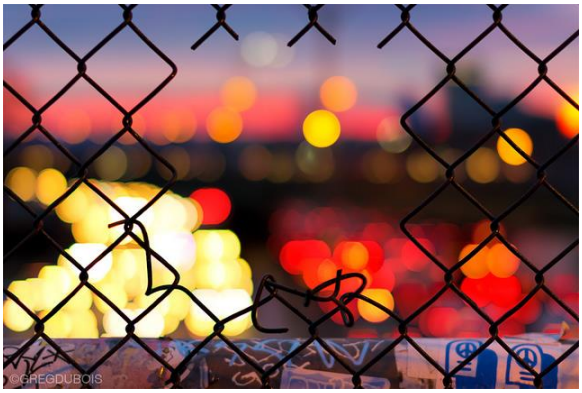

Fig 7. False Negative Image 1

Score: 2.187493985279326

Fig 8. False Negative Image 2

Score: 0.613976914619735

False Positive Images:

The false positive images classified by the system encountered misclassification with their image content. The first example in Fig 10 illustrates a hiking scene with large amounts of open sky. Due to the editing techniques used in the image, the sky appears sepia-toned resulting in misclassification. The image additionally has a score of -0.12103 which suggests

that it is located close to the decision boundary and is more susceptible to misclassification.

The second image is that of a sunflower with primarily yellow tones. This image was wrongly classified as a sunset due to the presence of warmer tones. Its corresponding score is -1.0961 suggesting it is further away from the decision boundary.

Fig 9. False Positive Image 1

Score: -0.121038304282087

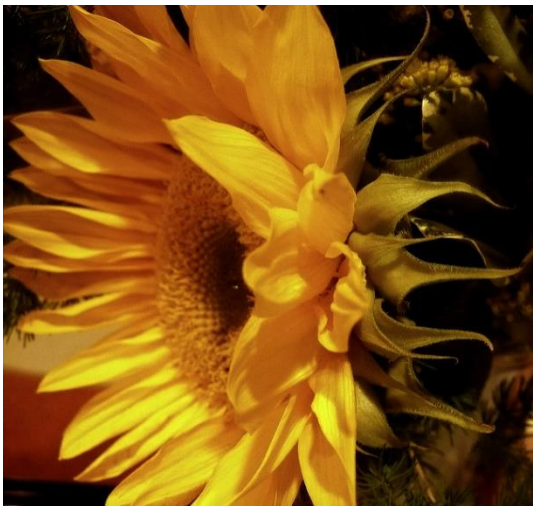

Fig 10. False Positive Image 2

Score: -1.096139339335316

True Negative Images:

The classification of true negatives represents the number of correctly identified non sunset images. The first image below in Fig 11 is an image of a sky which contains a lot of blue. Due to the color of the image, its score is therefore a 5.5653 suggesting it is further away from the margin for classification. The second image in Fig 12 is also that of sunflowers, which have been correctly identified as a non-sunset. The score of the sunflower image was 0.6876 which implies the image is close to the decision boundary. This may primarily come from the presence of yellow in the flower petal. However, due to the high presence of green and blue compared to the previous sunflower image, the algorithm was able to classify the image correctly.

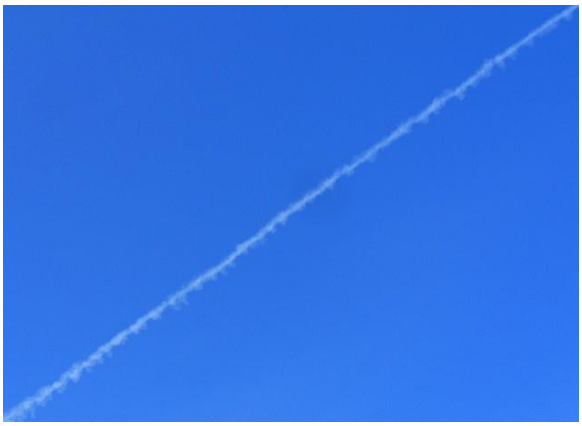

Fig 11. True Negative Image 1

Score: 5.565280101068944

Fig 12. True Negative Image 2

Score: 0.686765208531789

6.2 Feature Extraction

By using CNNs to extract features and classify them using SVMs, it is possible to develop a highly accurate image identification process. The implementation of the Feature Extraction methods yields more accuracy in comparison to the LST-SVM method. The following images are the sample cases that represent successful and unsuccessful classification by the feature extraction method.

True Positive Images:

The images below have correctly been classified as true positive images. The first image Fig 13 exhibits

typical characteristics of deep sunsets, with deep shadows and vivid orange hues. The image possesses a score of 2.385 and therefore can be easily classified as a sunset. The second image in Fig 14 lies closer to the decision boundary with a score of 0.072. As per the image, the sunset is obstructed by dark objects and is therefore harder to identify. The SVM, however, was able to use the features produced by the CNN to accurately classify both images.

Fig 13. True Positive Image 1

Score: 2.385414

Fig 14. True Positive Image 2

Score: 0.072437

False Negative Images:

False negatives occur when images with true classes of sunsets are wrongly classified as non-sunsets. The first image in Fig 15 depicts a sunset reflected by a body of water. The image has a score of 1.350 due to the large presence of golden tones in the image. Convolutional neural networks may not be equipped to classify such an image as its feature set may not align with those of a classic sunset.

The second image from Fig 16 represents a sunset but contains a lot of open blue sky. Due to the presence of blue tones and the obstruction of trees the resulting image has a score of 0.537 which is closer to the margin and more likely to be misclassified.

Fig 15. False Negative Image 1

Score: 1.350419

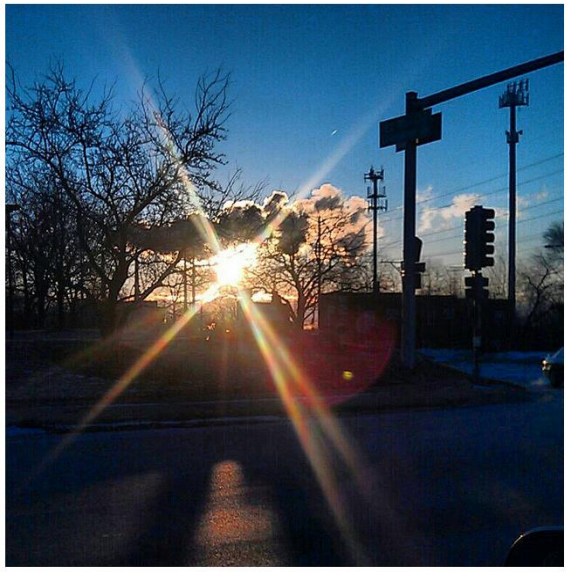

Fig 16. False Negative Image 2

Score: 0.537137

False Positive Images:

The classification of false positives emerges from images that are not sunsets but have been wrongly classified as so. The first image in Fig 17 was incorrectly classified in the LST-SVM algorithm as well. The image was taken in the daytime but has yellow tones due to the applied editing. As a result, it has an associated score of 1.030. The second image from Fig 18 has been misclassified due to the presence of orange tones in the image. As the waves reflect the

light in the image, it is likely that this may have been mistaken for the sun. The image has a score of 0.047 which places it incredibly close to the decision boundary compared to all other images.

Fig 17. False Positive Image 1

Score: 1.030439

Fig 18. False Positive Image 2

Score: 0.047022

True Negative Images:

Images that have been correctly identified as non-sunsets indicate true negatives. The images below in Fig 19 and Fig 20 contain strong purple/blue tones which allow for them to be classified as non-sunsets. The first image possesses a score of 1.782 indicating strong association as a non-sunset. The second image has a score of 0.236 as it is clearly depictive of a sky which may place it closer to the decision margin.

Fig 19. True Negative Image 1

Score: 1.781608

Fig 20. True Negative Image 2

Score: 0.235824

7. CONCLUSION

The combination of traditional machine learning techniques with deep learning techniques allowed for a greater correct classification rate than otherwise. Image recognition has been greatly employed in modern photo editing programs and continues to make an impact on photography companies worldwide. Utilizing an automated means of background detection therefore provides valuable information on the type of editing techniques used.

The solution to the problem of automating background detection was effectively adjusted to detect sunsets. By segmenting images and extracting their corresponding statistical data it is possible to use SVMs to create decision boundaries for classification. The process used to address the problem effectively uses radial basis kernel functions to exert control over the way decision boundaries are created. If Kodak were to use this method, certain parameters such as the box constraint and kernel can be tweaked to achieve the best possible detection rate. The threshold may additionally be adjusted as well to improve the accuracy of the system. This method of image recognition will prove useful in automated image editing functions.

Convolution neural networks provide a strong alternative to image classification by utilizing deep learning methods. The use of transfer learning allows for the existing layers within neural networks to be modified in order to adapt the algorithm to a set of data, However, the long computational time associated with this method proves inefficient. It is therefore more valuable to utilize a feature extraction approach that runs the convolutional neural network up to a desired layer. The feature extraction method seems to work best for smaller datasets and has a significantly lower run time. The capacity to use feature extraction alongside machine learning techniques such as support vector machines allows more control over the hyperparameters used for classification compared to the transfer learning approach. Feature extraction therefore provides a noteworthy middle ground to the use of deep learning and machine learning algorithms in classification problems.

For future works, the computation time for transfer learning methods can be improved by training on more robust computers. This also allows more experiments with hyperparameters in order to produce a finer tuned algorithm. Additional preprocessing methods may also be implemented to offset editing techniques used in images. It may also be possible to address the limitations of dataset inputs by using AI generated images of sunsets and non sunsets to train the CNN.

REFERENCES

[1] Hearst, M. A et al, 1988. "Support vector machines," in IEEE Intelligent Systems and their Applications, vol. 13, no. 4, pp. 18-28.

[2] Zhang, Y, 2012. Support Vector Machine Classification Algorithm and Its Application. In: Liu, C., Wang, L., Yang, A. (eds) Information Computing and Applications. ICICA 2012. Communications in Computer and Information Science, vol 308. Springer.

[3] Hsu, C et al, 2016. A Practical Guide to Support Vector Classification Chih-Wei Hsu, Chih-Chung Chang, and Chih-Jen Lin. Department of Computer Science National Taiwan University, Taipei 106, Taiwan

[4] Boutell, M & Gray, J, 2003. Sunset Scene Classification Using Simulated Image Recomposition. 2003 International Conference on Multimedia and Expo. ICME '03.