Needle In A Haystack

[Electrical Substation Recognition In Google Maps]

Zhengyi Wang
M.S. in Software Engineering
Carnegie Mellon University
NASA Ames Research Center, Moffett Field
California 94035
zwang1

Carlos Ruiz
Ph.D. in Electrical & Computer Engineering
Carnegie Mellon University
NASA Ames Research Center, Moffett Field
California 94035
cruizdom

ABSTRACT

Object recognition and image classification are well-studied topics in the Computer Vision field. Literature is full of algorithms and Machine Learning models that generalize well for such type of problems. However, the majority of these projects rely on a huge labeled dataset that is used to train their models with accuracy.

In this paper, we introduce an iterative approach for cases in which the lack of ground-truth prevents one from obtaining enough training samples for Machine Learning algorithms to actually *learn* from the data. In particular, we implement such approach to detect/recognize electrical substations in a set of satellite images taken from Google Maps.

1. INTRODUCTION

Image classification algorithms have been proven useful for a variety of tasks, ranging from health diagnosis to fault detection in a manufacturing plant. Yet in this paper, we introduce a new application area: earthquake risk assessment. In a collaboration with the United States Geological Survey (USGS), this project aims at finding the locations of main power supplies (*e.g.*, electrical substations) to predict the impact of an earthquake in a region.

USGS is working on an algorithm to predict the most likely location of the next earthquake's epicenter. In order to assess the risk associated with each probable location, it would be very helpful to overlay a map layer containing all (or the main) power substations in a region. However, electrical companies refuse to provide such sensitive information due to potential security threats. Therefore, a different approach needs to be taken in order to obtain the location of these stations. In this paper, we propose training a Machine Learning model to find the substations from a collection of satellite images (Figure 1).

In a previous course project, a group of students developed some MatLab code to achieve the same goal. However, their algorithm had several drawbacks. First of all, it required a very high computation time (2-3 seconds per image) to extract features, which would make the project almost unfeasible given the huge amount of images per area (just the city of Sunnyvale alone is covered by over 15,000 tiles at the deepest zoom level). In addition, it is based on SURF and Harris feature detection, which is not shift-invariant. This is a huge downside since tile images at the deepest zoom level do not cover a whole substation (only part of it), and thus using the location of SURF points in the image does not help identify it. As a result we propose an algorithm based on color analysis and relative position of lines as features to feed with a Machine Learning classifier. This ensures shift-invariance and is more prone to a null false negative rate. In the next section, we introduce our algorithm and feature selection implementation.

Figure 1: Sample satellite image of a substation

2. ALGORITHM DESCRIPTION

Our classifier takes a tile as input, first preprocessing to extract color and lines features. Then go through three machine learning algotithms, which are SVM, Decision Tree and Extra Decision Tree. After get result from three classifier, we found they still have tons of false positive so we improve the system to add the last majority voting step to reduce false positive. Finally the output is whether or not this tile contains a substation. Figure 2.

2.1 Feature Selection

Most substations are very small when looking from the satellite view of Google Map. You cannot even recognize them until zoom into the level 20 which the most detailed level of Google Map. At the zooming level 20, only Sunny-

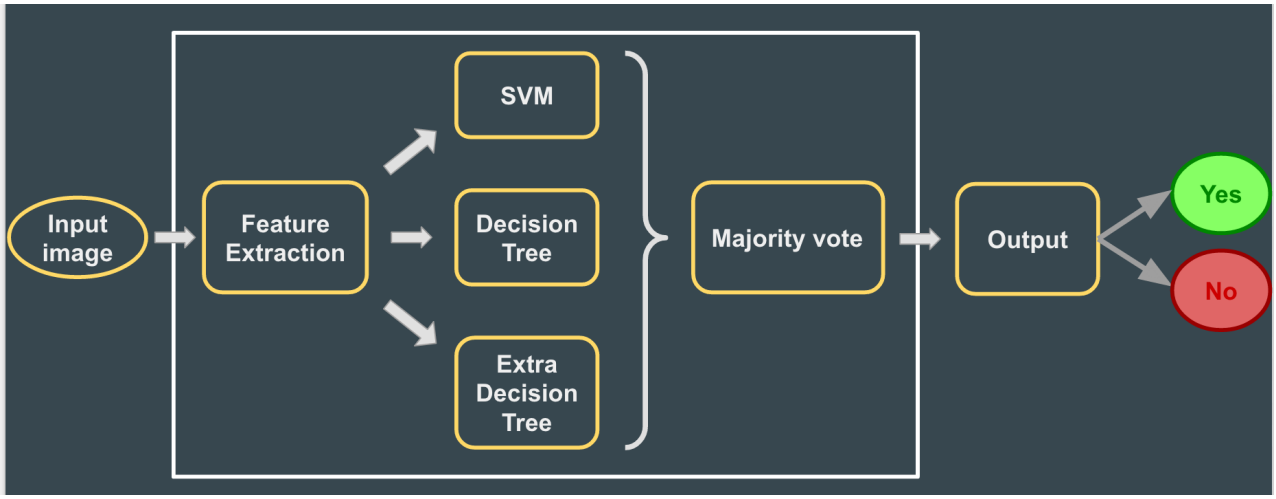


Figure 2: System diagram. Note that due to the lack of ground-truth, we do not have enough data to accurately train a Machine Learning classifier. We therefore combine the outputs of three different models (SVM, Decision Tree, Extended DT) to improve the overall accuracy.

vale region covers around 15000 images, among which there are only 10 substations according to the supporting document we found.

Images we download directly from Satellite view of Google Map is not of good quality and their clearness, brightness, resolution or contrast vary a lot due to the whether of the day they are photographed. This increased the difficulty of our project significantly requiring us to do some research on pre-processing images before extract features.

Figure 1. This is a typical substation image. By looking at it we can think of some features to distinguish from other objects from the map. 1.The background of all substation are almost gray and the wires are all write; 2.The wires always appears in a set of 3 wires because we use three phase electricity; 3. The wires are either perpendicular or parallel to each other. Therefore we decided to extract features as HSV colorspace histogram and the spacial distribution of lines.

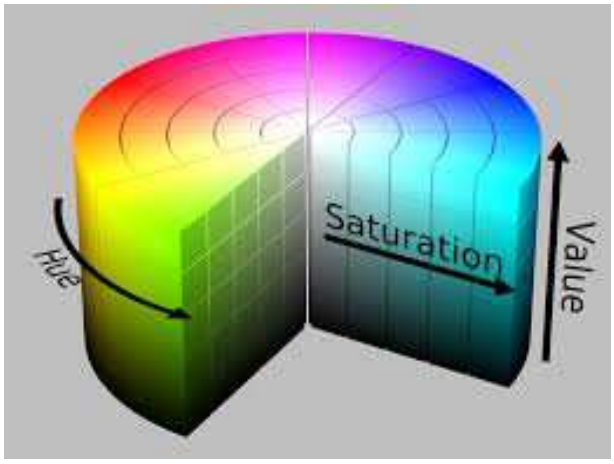


Figure 3: HSV colorspace

2.1.1 *Feature* ***Color***

Color may give important information in object detection application. In our application color information is also valuable. It is useful to distinguish substation from forests, waters, and some other non-gray human-made constructions.

At first we considered RGB color space. The problem with RGB color space is that when comparing the histogram of a picture of all gray with a picture of one third of red, one third of green and one third of blue, they would be exact same. Because gray is combined with red, green and blue. Therefore another color space is needed and we choose to use HSV color space.

In HSV color space, H stands for Hue, which is what we actually call ”color”. S stands for Saturation, showing how transparent the color is. And V stands for Value, which means how strong the color is. Shown as Figure 3.

Here is an example showing how we determine what color is a image in the HSV color space. When set a threshold for HSV values respectively as a small cube to only filter the blue color. Some pixels consisting of the swimming pool in left side image will pass this filter while no pixel in the right image falls into this color range. Figure5a

Then we divide the HSV color space into several small cubes. If in H axis devide into 3 range, S in 2 and V also 2. We will get 12 cudes in total. We count how many pixels in an image fall into each small cubes and plot and histogram for the pixel numbers in each cube. Figure 5b

In the above example, 12 bins in histogram will be 12 features when feeding into SVM or Decision Tree Classifier.

In practice, we experimented with several iterations to see how many small cubes works the best. We use 52 positive samples and 62 negative samples as training dataset. 10 positive and 10 negative samples as testing dataset. Positive samples are all substation of a part of substation images and negative images are randomly chosen from the Google map, which could be forests, lakes, buildings. Result is shown as Table 1. The correctness cannot represent the true accuracy because the testing data is too small but we can get an idea of which combination of bins works better. Too little features can not provide enough information while too many will lead to over fit.

Table 1: Algorithm accuracy vs. # HSV histogram bins

H.S.V	Accuracy for Positive	Accuracy for Negative
2.2.2	60%	100%
3.3.3	90%	80%
4.3.3	80%	80%
4.4.4	80%	90%
5.4.4	80%	90%
5.5.5	95%	95%
6.4.4	50%	95%
6.6.6	50%	90%

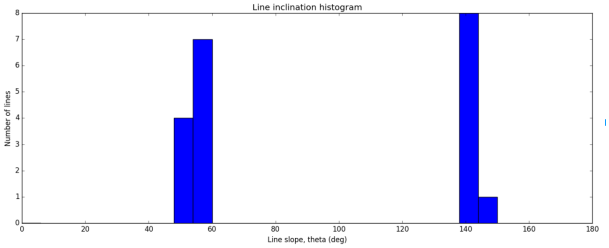
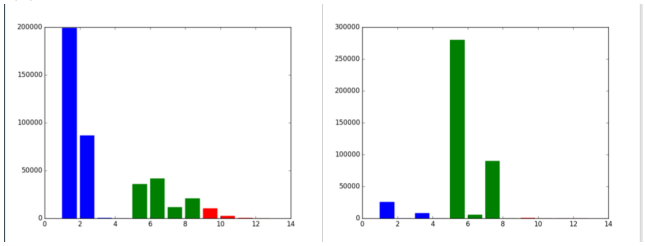


Figure 4: Sample histogram of line slope (θ) for the image in Figure 6. Note both line clusters are perpendicular (90Âž difference).

(a) Two sample images with different distribution in HSV color



(b) The HSV histogram for each image in 5a

Figure 5: Comparison of HSV histograms of two images

Since 5-5-5 works best so we choose this combination which has too many features as 125, although a lot bins contains 0 pixels and provide no information for training, it is fine since our algorithm will choose the ones with most information gain.

There are other color features we considered as variance, entropy and spacial histogram, but don't have time to explore will be our feature work to improving our correctness.

2.1.2 *Feature* ***Lines***

As mentioned in the introduction, the very obvious feature that can be easily noticed is the lines in the images of substations. We are using three-phase electricity so the wires always appear in a set of 3 and parallel to each other. One group of lines are perpendicular with the other one.

At first we tried using Detect HoughLines directly on the Canny edge of the original picture. It works for some images but not for the other. The problem with that practice is that each image has different brightness, contrast, and resolution which may depend on the whether of the day they are photographed. In order to detect the lines that are meaningful we need manually tune the threshold for each picture, which is impossible to do when running the algorithm for thousands of images. If not using the correct threshold we will either detect no lines or tons of lines all over the images. Another problem is that some pictures are taken in a really sunny day and has strong shadow. The white wires are hardly visible while instead the shadow is clearly recognizable.

To solve this problem, we do preprocessing for each image. With each input image, we first check its mean HSV to decide whether we will look for white wires or dark gray shadows. Filter image with the experimental threshold the mask all other colors and only remain the color we need to detect lines. Then do Canny edge on the filtered image and finally detect Hough lines. This process is proven to be significantly reducing the noise. The result of a sample image is shown in Figures 6

In order to use the feature as perpendicularity. We calculate each lines relative angle to the left upper point. In the example 4, one group of lines are at the angle of 45 degrees and another group are at the angel at almost 135 degrees. The difference between them are 90 degrees. With several iterations we finally choose to use the numbers of lines in each angles to plot a histogram with 30 bin

We also extract the feature as distance, but since each of the lines are not always recognized and some times one line could be recognized as multiple ones. So this feature turns out not working well. Neither does the feature of find a set of 3 lines work in practice.

2.2 Dataset

We faced the difficulty that is lack of ground truth. What we can found is a document (`http://www.pge.com/includes/docs/pdfs/b2b/energysupply/qualifyingfacilities/settlement/substations.pdf`) listing all substation names in each city, but without their actual location. The biggest problem for us is we don't have access to a lot of training data. When manually type in "substation" in Google Map, we can only find 20 big substations all around USA. By then we didn't even what is a small substation looks like. We use the download around 50 pictures from the 20 substations we found manually as our initial positive training samples. For nega-

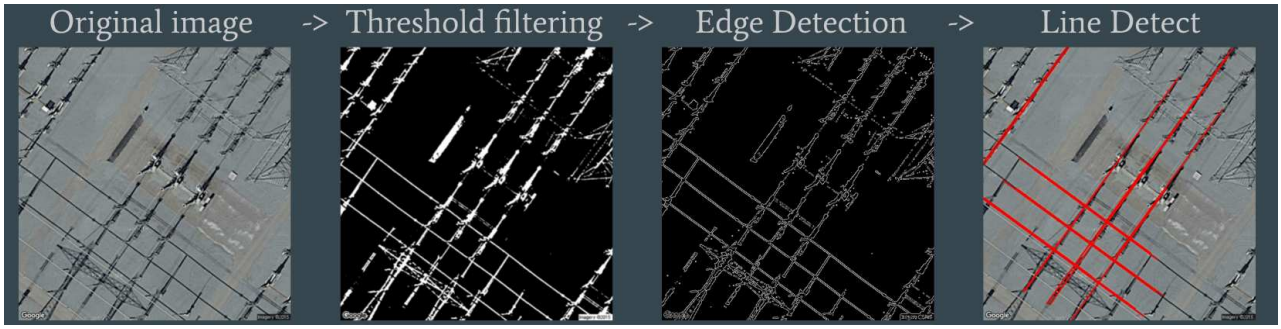


Figure 6: Preprocessing (line detection process) before extracting line features

Figure 7: Initial training samples: top row are labeled as positives; bottom row are negatives.

tive training samples we choose random pictures like forest, roads, houses. In order to have balanced training dataset, we use around 60 images as negative samples. Examples are in Figure 7. From the document we know that Sunnyvale city has 10 substations. So we download the whole Sunnyvale city at zoom 20 level which is in total around 15000 images.

Our testing dataset is highly biased. The Sunnyvale region has almost 15000 negative images while only 15 positive ones. It is not fair to use The percentage of correct prediction to measure the the accuracy of our classify system, since any naive classifier predicting every image as false would have a correctness of 99.9%. It is also not fair simply using percentage of true positive, false positive, true negative and false negative. Because even 10 negative samples out of 15000 negative ones are wrongly predicted as true and all positive ones are correctly predicted, the false positive would be bad as 50%.

Moreover, the goal of algorithm is to not miss any true positive samples. So we introduce another two metrics to measure the accuracy of our machine learning algorithm. One is positive correctness: the percentage of the correctly predicted positive samples out of the total number of positive samples. The other is negarive correctness: the percentage of the correctly predicted negarive samples out of the total number of negative samples. By that means our goal is to first and more importantly achieve the 100% positive correctness, and then trying to improving the negative correctness as much as possible.

Table 2: Accuracy of Classifier

	Positive	Negative
`SVM`	82%	78%
`Decision Tree`	80%	76%
`Extra Decision Tree`	75%	73%
`Majority Vote`	82%	86%
`Intersection of SVM and DT`	82%	92%

2.3 Algorithm Optimization

With the limitation of numbers of positive samples in training dataset, training result in early iterations were really bad for both SVM and Decision Tree algorithm. We experimented with different setting of histogram bins, color cube numbers, and threshold for detecting lines. Also we find some training samples may be confusing for our machine learning algorithms, so we carefully re-choose our training samples, especially for negative ones. Through many iterations we managed to find all the 15 positive images in Sunnyvale and exact 10 substations(some substations are cut into 2 or 3 images).

With the same training dataset and same features, we use three algorithm to training our classifier. SVM, is the most popular algorithm with image recognition and it has been proved to learn and works well with small dataset. Decision Tree is a very intuitive algorithm and it will pick the features with the most information gain. Extra Decision Tree, is a improved Decision Tree algorithm, which can better avoid over fitting when too many features are used.

Table 2 shows the metrics of correctness for three algorithms with same training features and initial training dataset on the testing result of the sunnyvale 15000 images. Since they all output a tons of false positive images we improved our algorithm by adding a majority voting. If at least two classifier predict certain image as positive then we determine it as positive. It also works doing intersection of two classifier. This approach reduced the false positive significantly and at the same time does not harm the positive correctness.

Since now we have 15 more positive samples from Sun-

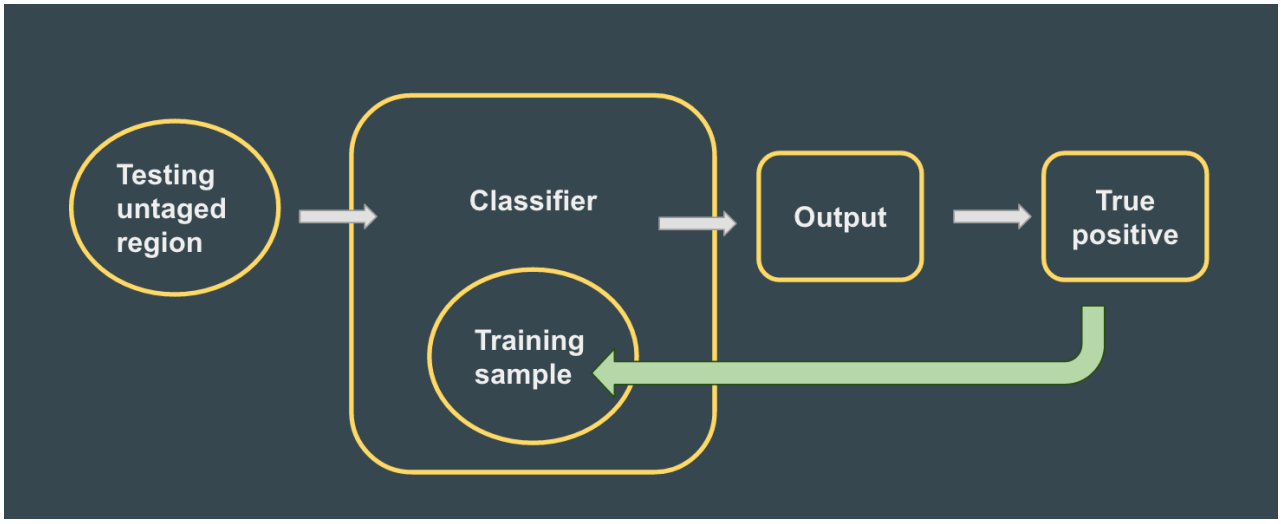


Figure 8: Iterative adding training data

Table 3: Accuracy of Classifier

	Positive	Negative
SVM	93%	83%
Decision Tree	100%	81%
Extra Decision Tree	85%	78%
Majority Vote	94%	88%
Intersection of SVM and DT	94%	91%

nyvale, we can add them into our training dataset to train other regions. We download the San Fransisco City, which is about 38614 images, run our majority vote algorithm with it.

We got 20 more positive samples from San Fransisco so we add them back into our training data and redo for Sunnyvale. Our next step is to do Berkeley. Iteratively we get more and more positive training samples. Figure 8 showing the iterative adding training data process.

2.4 Results

After using more training data which we get from San Fransisco and testing on Sunnyvale again. Result is shown in Figure 9 as true positive and Figure 10 as false positive. We improved our accuracy as shown in Table 3 and detect all 10 substations marked in Figure 11.

Finally testing on sunnyvale we found all 10 positive substations as marked in Figure 11. The average time for predicting each image is 100 ms.

3. TEAMWORK

During the project we worked closely with each other as a team on everything. For the report, Carlos wrote the Abstract, Introduction and Line Feature parts. Zhengyi wrote Color Feature, dataset and Algorithm optimization parts.

Figure 11: Final result: Location of the 10 substations found in Sunnyvale, CA.

Figure 9: Sample images of True Positive results in Sunnyvale region

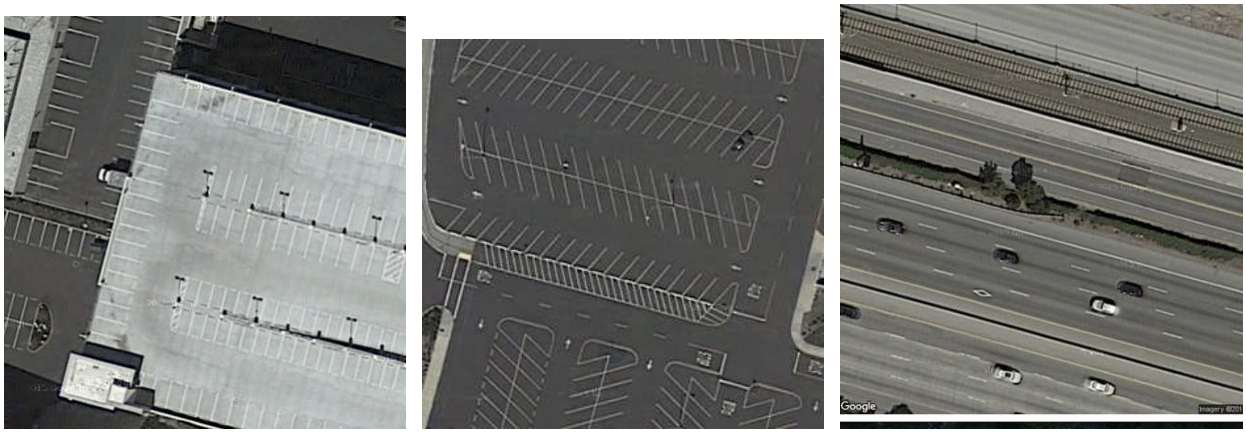


Figure 10: Sample images of False Positive results in Sunnyvale region