MACHINE LEARNING FOR AUTOMATIC TARGET DETECTION

Machine learning (ML) applications, from object recognition and caption generation, to automatic language translation and driverless cars, have increased dramatically over the last few years, powered mainly by the increase of computing power (using graphics processing units (GPUs)), reduced cost of storage, wider availability of training data, and development of new training techniques for the ML models.

This blog offers a deeper dive into the Machine Learning training process for performing automatic target detection. Samples of automatic target detection were recently presented at the Machine Learning: Automate Remote Sensing Analytics to Gain a Competitive Advantage webinar.

In the last five years, Harris Corporation has made a multi-million dollar investment into applying ML to solve customer challenges using remote sensing data. In response to the increased interest from our customers in evaluating how ML can solve their problems using geospatial data, I set out to train some of my co-workers on how to build an ML model to perform automatic feature detection on 2D overhead imagery. This training was crucial for our Solutions Engineers (SEs) to be able to prototype custom solutions for our customers and/or to integrate ML with our other powerful image analytics software like ENVI/IDL.

Figure 1: Pictured from the left in the Broomfield, CO office: Jeff McKissick (SE); Zach Norman (SE); Pedro Rodriguez (SE); Rebecca Lasica (Partners Manager); and, Dan Platt (SE).
In just a few hours, and with each trainee using a small Red Hat Virtual Machine of 4GB RAM and 2 CPUs, we were able to complete the entire process, from gathering the training data, building the ML model, and finally classifying a subset of the selected raster dataset.

For the raster dataset we used a high resolution satellite image from DigitalGlobe, Inc. (0.3 GSD, 4-band (RGBN), WorldView-3) from São Paulo, Brazil. To gather the training data (positives and negatives), we used a custom ENVI extension to chip 35x35 pixels samples and augment the training data as shown in Figure 2 at right.

For data augmentation, we simply rotated each image chip 90 degrees (4 rotations). For the sake of time, we only selected 100 positives and 200 negatives, which after the data augmentation we had 1,200 training chips (400 positives and 800 negatives). From the 1,200 training chips 10 percent were used for validation, 20 percent for testing, and 70 percent for doing the actual training of ML model. As seen in the heatmap shown in Figure 3 below, the ML classifier resulting from the limited training dataset (1,200 samples x 70% training = 840 training samples) performed very poorly as it contained many false negatives (missed detections) and some false positives (wrong detections).

In order to highlight the true potential of our ML technology, I decided to train the crosswalk classifier with a larger training data set. For this, I increased the training data by 5 times, so instead of just having 100 positives and 200 negatives, the new training set had 500 positives and 1,000 negatives. I also rotated each image chip by 10 degrees (36 rotations) instead of just every 90 degrees (4 rotations) which augmented the total image samples to 54,000. Table 1 below summarizes the data set used in both cases.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Positives</th>
<th>Negatives</th>
<th>Rotations</th>
<th>Total Samples</th>
<th>Training Samples (70%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>100</td>
<td>200</td>
<td>4</td>
<td>1200</td>
<td>840</td>
</tr>
<tr>
<td>Large</td>
<td>500</td>
<td>1000</td>
<td>36</td>
<td>54000</td>
<td>37800</td>
</tr>
</tbody>
</table>

Table 1: Data Set Characteristics
The next step was to determine the number of iterations (mini-batch updates) that were needed to complete an epoch. One epoch consists of one full training cycle on the training data set. To calculate iterations per epoch we use the following formula:

$$\text{IterPEp} = \frac{\text{TS}}{\text{BS}} = \frac{37,800}{128} \approx 239 \text{ iterations per Epoch}$$

where,

- $\text{TS}$ = Training Samples
- $\text{BS}$ = Batch Size

It’s difficult to prescribe a minimum number of epochs for training a new model since it will vary depending on the difficulty of the problem, quality of the data, chosen network architecture, etc. As a starting point, I began with 42 epochs and to calculate the total number of iterations I used the following formula:

$$\text{Iter} = \text{Ep} \times \text{IterPEp} = 42 \times 239 = 10,038 \sim 10,000 \text{ iterations}$$

The required number of epochs can be determined by watching the validation accuracy as the training proceeds with increasing number of iterations. This validation accuracy can be plotted with IDL as Receiver Operating Characteristic (ROC) curves as seen in Figure 4 below:

ROC curves feature false positive rate on the X axis and true positive rate on the Y axis. This means that the top left corner of the plot represents the “ideal” Machine Learning classifier, which has a false positive rate of zero, and a true positive rate of one. It can be seen from Figure 4 above, that the accuracy of the crosswalk classifier increased with the number of iterations. At 30,000 iterations (about 126 epochs), the ROC indicated that enough training was achieved with an overall accuracy (ACC) of 98.84 percent. Figures 5 and Figure 6 below show the results of the 30k iteration classifier in a dense urban scene and in a highway scene, respectively. This crosswalk classifier proved to be very robust against confusers, like other similar street marking, and occlusions, like partially hidden crosswalks in the shadows. I challenge you to find the crosswalks manually in the “before” (left) scenes of Figures 5 and Figure 6. You can later validate your answers in the “after” (right) scene that was analyzed using ML. Can you imagine manually identifying all the crosswalks in the city of Sao Paulo?

**Figure 5:** Urban Scene, before and after crosswalk detection
Automatic target detection is one of our most basic ML solutions, which usually involves searching for particular features in a large dataset, therefore applicable to many real world challenges. This type of solution is even more relevant with the “Big Data” surge in which studies indicate that only 0.5 percent of all data generated gets ever used or analyzed (1). It is clear that future business advantages in every industry will arise when companies are able to automatically analyze this surge of data. Machine Learning is not meant to replace industry professionals, but to off load some of the tedious task to the computer, so they can focus their expert attention to analysis and not on “snailing” large datasets searching for particular features. ML can also run 24/7 and is highly scalable to available computing resources.

I want to emphasize that at Harris Corp. we are not merely delivering software on disk, but an end-to-end solution to deliver answers to specific industry problems. To answers questions like, “How many utility poles need servicing?” “Which blades in a wind farm have damage?” or “How are the road conditions near me?” All of these are questions we have been able to accurately answer for our customers.

If you would like to know more about how we have implemented Machine Learning to address other real-world problems, watch the webinar that my co-worker Will Rorrer and I hosted in January 2017: Machine Learning: Automate Remote Sensing Analytics to Gain a Competitive Advantage

References: