Machine Learning Applied to Terrain Classification for Autonomous Mobile Robot Navigation

John Rogers, Andrew Lookingbill
CS 229 Final Project

I. Introduction

We work on the Stanford AI Lab team for the DARPA-funded Learning Applied to Ground Robotics (LAGR) project. Each of the eight competing teams in this program write code for a standardized robot platform equipped with short-range sensors and two pairs of stereo cameras. The long term focus of the program is the advancement of the state-of-the-art in computer vision and offroad autonomous mobile robot navigation.

Once a month the government team runs 3 tests in a row on the robot platform with each team's software at a remote location. The robot is expected to demonstrate inter- and intra-run learning. A priori information about the testing sites is limited. These requirements make the application of online machine learning algorithms attractive. In addition, the large volume of log data available from previous tests makes these algorithms easier to compare and evaluate. This paper describes our work with online machine learning algorithms for long-distance monocular perception. Section II describes our general classification pipeline, Sections III-V describe three different online algorithms we tested, and Section VI reports the results of running these algorithms on a standardized set of videos.

II. Approach

The LAGR robot platform benefits from reliable stereo information coming from its two pairs of Point Grey Bumblebee cameras. This information can be used to train a classifier which operates on all the pixels in a monocular image. Fig. 1a shows a single frame from a stereo pair of images from a logged image sequence. The left and right images are passed to a stereo algorithm which returns X,Y, and Z coordinates in the robot's frame of reference. Using this information along with a flat ground plane assumption it is trivial to determine which pixels in the right-hand input image correspond to objects which rise more than 10cm above the ground plane on which the robot rests. These pixels can safely be assumed to belong to obstacles. Pixels corresponding to points close to the ground plane belong to traversable terrain. A set of these stereo-based classifications are shown in Fig. 1b where green denotes traversable terrain and red indicates obstacles.

The short baseline of the stereo systems on the LAGR robot platform limit their range, however. They are unable to classify objects reliably at distances greater than 5m from the robot. A terrain classifier that classified pixels anywhere in an image, after having been trained with stereo classifications would be a powerful tool allowing correct scene segmentation at great distances from the robot. These long-range pixel classifications can be dropped into a 2D hazard map such as the one shown in Fig. 1c. This map can then be used as the input to a global planner such as D*. In this way, the long-range pixel classifications give the planner more useful information which, in turn, would allow greater navigation speeds.

This general learning framework, where the training data comes from examination of short-range stereo data is generic, and can be modified to test a variety of online learning algorithms. The following sections describe our experimentation with three such algorithms.

III. Perceptron

Our implementation of the kernelized Perceptron learning algorithm was formulated as follows:

$$prediction_o n_e xample_k = sgn\left(\Sigma_{i=1}^{k-1}\alpha^i\beta^i K(x^i,x^k)\right) \tag{1}$$

where α^i is one if the i'th example was misclassified and zero otherwise. For the data reported in this paper, a Gaussian kernel was used to allow for non-linear decision boundaries without much risk of overfitting.

Fig. 2 shows example input-output frame pairs from three different video sequences. The top frame in each pair is the original monocular input image, while the bottom frame is the output of the trained classifier on that image. The nature of the test video sequences as well as our metric for quantifying the quality of the classification is discussed in the Results section.

IV. Forgetron

As an alternative to the Perceptron algorithm we also implemented the Forgetron algorithm. [1] This algorithm performs well in the type of online classification we are trying to implement. This algorithm smoothly trades off between limited memory capacity and classifier accuracy by incorporating a decay parameter as well as a finite history of comparison examples..

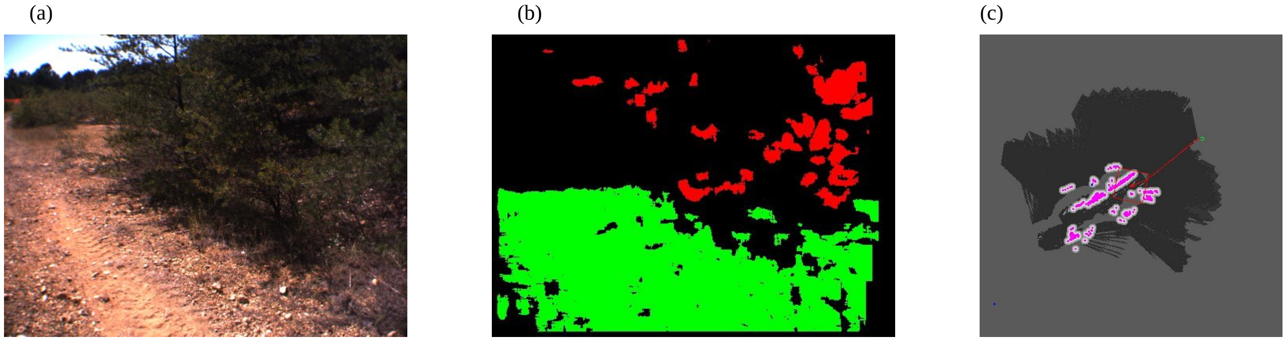

Fig. 1. (a) Right image from a stereo pair (b) Stereo-classifications for pixels (c) Information from classified pixels being placed in a global map

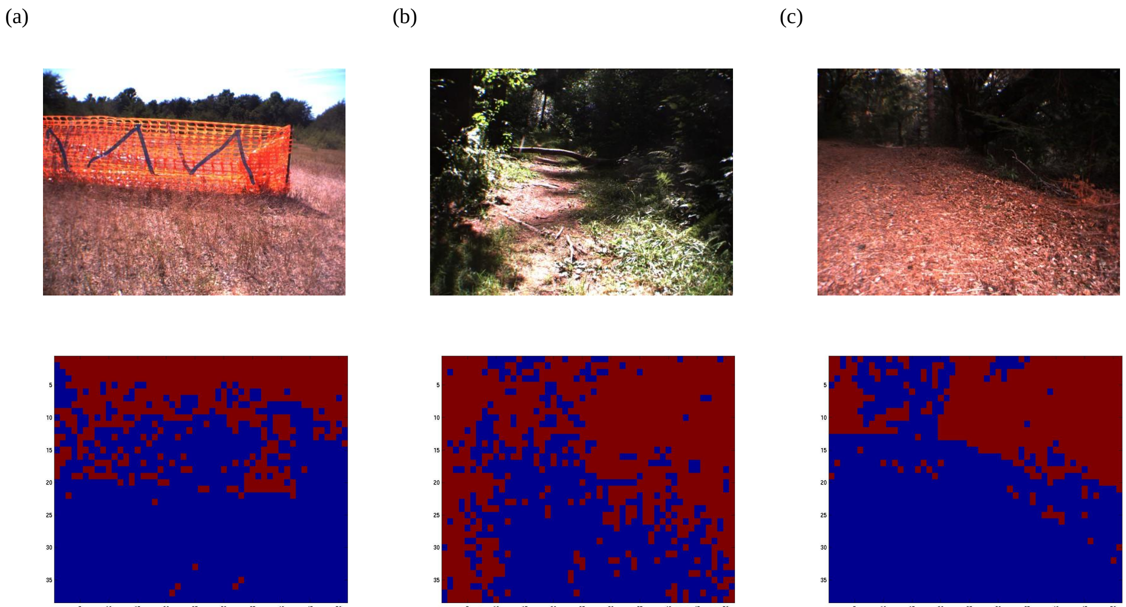

Fig. 2. Input-output classification pairs from three different image sequences classified using the kernelized Perceptron learning algorithm. Red indicates obstacles while blue indicates traversable terrain.

Our implementation of the Forgetron algorithm was formulated as follows:

$$prediction_o n_e xample_k = sgn\left(\Sigma_{i=k-1}^{k-1-B}\gamma^{i-k+1}\alpha^i\beta^i K(x^i, x^k)\right) \qquad (2)$$

where α^i is one if the i'th example was misclassified and zero otherwise, and γ is the decay parameter. For the data reported in this paper we employed a Gaussian kernel for the same reasons as described in the Perceptron section above.

Fig. 3 shows example input-output frame pairs from the same three video sequences that were used before. The top frame in each pair is the original monocular input image, while the bottom frame is the output of the trained classifier on that image.

V. Naive Bayes

In addition to the two algorithms described above we also implemented a Naive Bayes classifier along the lines of the one implemented for spam filtering in class. We maintain a large table of structures which has an entry for each of the 16 million colors in the colorspace. Any pixel which is classified by stereo as obstacle or traversable terrain has its respective counter incremented in the histogram table. This makes training very efficient; it is well suited for online operation.

The naive Bayes classifier produces inferences on 10x10 pixel blocks. We chose this size to allow regions to flow together smoothly while removing a lot of false classifications on smaller single pixels. When operating on a 10x10 pixel region, the classifier attempts to ascertain the joint log-likelihood of the data conditioned on this block belonging to each class. Whichever class gives the higher likelihood is selected as the classification for this pixel block. Fig. 4 shows the usual example input-output frame pairs.

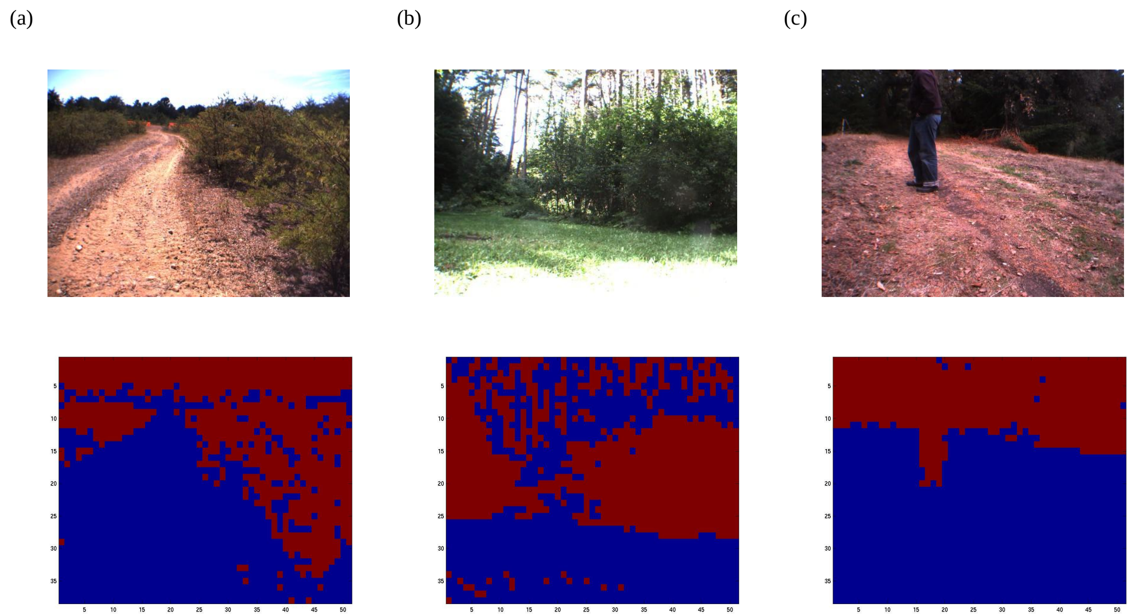

Fig. 3. Input-output classification pairs from three different image sequences classified using the kernelized Forgetron learning algorithm. Red indicates obstacles while blue indicates traversable terrain.

VI. Results

To compare the effectiveness of these three online learning algorithms, we selected three different video sequences from our log archives. The first video sequence was recorded during a government test in a wooded area. The second sequence was recorded during a government test where the robot was driving in pine duff. The third sequence was recorded during one of our local testing runs at Upper Steven's Creek, in the Santa Cruz mountains.

The three video sequences comprise 1400 frames in total. To gauge our algorithm's performances, we set aside every tenth frame in each data set as a testing frame and carefully generated a hand-labeled comparison image for each one. This hand-labeled image classifies terrain all the way to the horizon. For each of the approaches we trained on the non-test frames until we encountered one of the frames for which we had hand-labeled data. We then classified the frame using the training accumulated thus far (simulating the actual behavior of the learning algorithm during a robot run). After classification we compared the hand-labeled frame with the machine-classified frame and accumulated statistics about the false positive and false negative rates.

The Forgetron and Naive Bayes implementations were trained on all 1400 images, and tested on 140. The Perceptron implementation, however, was only trained on 350 frames and then tested on 140 frames because of memory limitations.

The results of the comparison are shown in the table in Fig. 5

Examining the data reveals that the Naive Bayes classifier was biased towards false negatives. However, the combined error rate of the Naive Bayes classifier (0.1688) was significantly lower than that of the Perceptron (0.26) and the Forgetron(0.31). It may seem that this error rate is high, but this is a very difficult application of these learning algorithms. They are trained on only the extreme near field information which gets useful stereo classification. The algorithms are then tested on full images with ranges out to the horizon. The best algorithms for use in this application will be ones who generalize well from near to far range examples.

VII. Conclusions

Due to the accuracy of the Naive Bayes classifier on the types of test data we provided, and its ease of implementation, we decided to pursue a C implementation of the algorithm to run directly on the robot. This code has been incorporated into our 2D segmentation module and was included in the last code shipment to the Government.

Future work includes the addition of texture information or different color space representations as features for the classifier. We purposefully made our implementation as modular as possible to facilitate these future changes. Including texture energy information will allow us to correctly classify obstacles even in environments where the obstacles and the traversable terrain are similar colors which will become more important in the Government testing as the winter ensues.

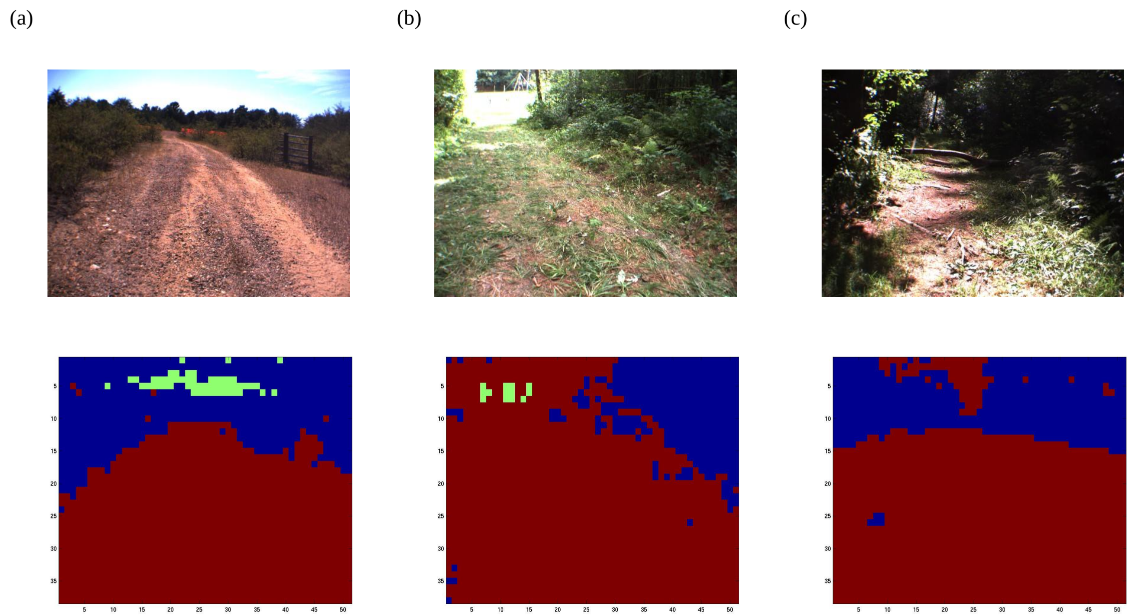

Fig. 4. Input-output classification pairs from three different image sequences classified using the Naive Bayes classification algorithm. Red indicates obstacles while blue indicates traversable terrain.

Fig. 5. Classification error rates for the three online learning approaches

ACKNOWLEDGMENTS

The authors gratefully acknowledge the assistance and guidance of Sebastian Thrun and Andrew Ng.

REFERENCES

[1] Dekel, O., Shalev-Shwartz, S., Singer, Y.," The Forgetron: A Kernel-Based Perceptron on a Fixed Budge," to appear NIPS 2005.