A Method of Evolving Novel Feature Extraction Algorithms for Detecting Buried Objects in FLIR Imagery Using Genetic Programming

A. Paino*, J. Keller*, M. Popescu**, K.Stone*
*University of Missouri, Electrical and Computer Engineering Department, Columbia, MO
**University of Missouri, Health Management and Informatics Department, Columbia, MO

Abstract

In this paper we present an approach that uses Genetic Programming (GP) to evolve novel feature extraction algorithms for greyscale images. Our motivation is to create an automated method of building new feature extraction algorithms for images that are competitive with commonly used human-engineered features, such as Local Binary Pattern (LBP) and Histogram of Oriented Gradients (HOG). The evolved feature extraction algorithms are functions defined over the image space, and each produces a real-valued feature vector of variable length. Each evolved feature extractor breaks up the given image into a set of cells centered on every pixel, performs evolved operations on each cell, and then combines the results of those operations for every cell using an evolved operator. Using this method, the algorithm is flexible enough to reproduce both LBP and HOG features. The dataset we use to train and test our approach consists of a large number of pre-segmented image "chips" taken from a Forward Looking Infrared Imagery (FLIR) camera mounted on the hood of a moving vehicle. The goal is to classify each image chip as either containing or not containing a buried object. To this end, we define the fitness of a candidate solution as the cross-fold validation accuracy of the features generated by said candidate solution when used in conjunction with a Support Vector Machine (SVM) classifier. In order to validate our approach, we compare the classification accuracy of an SVM trained using our evolved features with the accuracy of an SVM trained using mainstream feature extraction algorithms, including LBP and HOG.

Keywords: forward looking infrared imaging; FLIR; IR; genetic programming; HOG; LBP; image feature extraction

1. INTRODUCTION

Detection of buried targets continues to represent a challenging problem and multiple sensing modalities such as ground penetrating radar, metal detectors and infrared (IR) have been employed to address it. Among the variety of sensors used to detect buried objects, IR is gaining in popularity due to the recent advances in un-cooled camera technology and multiband sensors.

Infrared cameras have been used in a variety of applications such as ground target recognition [1, 2], flying target tracking [3], remote sensing [4] and landmine detection [5-9]. IR object detection is based on the temperature difference between the target and the surrounding background. More specifically, the detection of buried objects is based on differences in soil texture, spectral composition and temperature between the area above the target and background [5].

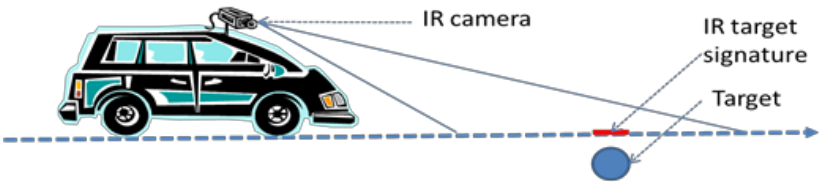

Figure 1: LWIR camera mounted in front of a moving vehicle to detect buried targets based on their IR signature

In this paper we use a long wave IR (LWIR) camera installed on a vehicle (see Fig. 1) to detect objects buried in the road ahead. The vehicle drives on a road that has buried targets which exhibit certain IR signatures (see Fig. 2 for example images). Our goal is to cue the driver for possible buried objects of interest. While our ultimate goal is to cue the driver in real time, in this paper we present only a retrospective (offline) IR video processing methodology.

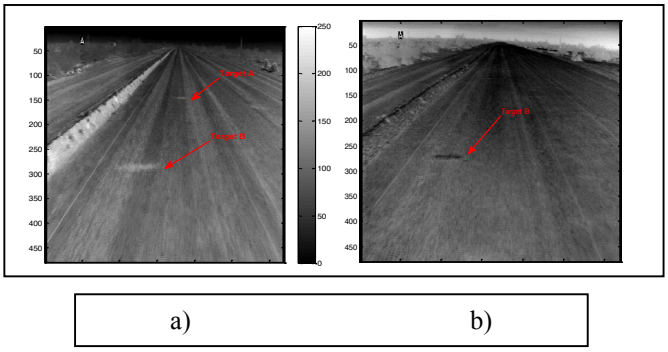

Figure 2: Typical vehicle mounted IR images taken at different time of the day: a) at 11 am, b) at 7 am

In previous papers [7-9] we presented several approaches to solve this problem. In one of our most recent efforts [6] we used the Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP) image feature extraction algorithms in conjunction with a Support Vector Machine (SVM) to classify 384x384 images that had been extracted from the FLIR video feed. As we implemented this algorithm, we noticed a high-level similarity between the HOG and LBP algorithms. Basically, we noted that both algorithms compute statistics for every sub-window of a fixed size in a given image and then aggregate the statistics into a histogram. Going further than this, we realized that each of the statistics computed for every sub-window could be represented as parse trees using basic arithmetic and comparison operators. Specifically, both HOG and LBP can be thought of as forests of parse trees with the functional nodes being n-ary arithmetic or comparison operators and the terminal nodes being a specific pixel intensity value or a random constant. In the case of HOG, the image undergoes a filter operation prior to evaluation by these parse trees, but so too can the image in an LBP algorithm evaluation if the identity filter is used.

After we realized the basic form of these algorithms could easily be represented by parse trees, the natural next step for our lab to take was to investigate applying Genetic Programming (GP) to evolve new image feature extraction algorithms. This paper presents the findings of that investigation. The remainder of this paper is structured as follows: in section 2 we present the available dataset, in section 3 we describe in detail the methodology of our approach, in section 4 we describe our results and in section 5 we provide concluding remarks.

2. DATASET

The data used in this paper was collected at an arid United States Army Test Site with a LWIR camera installed on a vehicle (see Fig. 1). The 1200 meter lane had 50 buried targets. We collected data from 7 runs on the lane. Typical frames from the collected movies are shown in Fig. 2. We can see two targets in Fig. 2.a and one target in Fig. 2.b. It is interesting to note that, since the two frames were taken at different time of the day, not only the target signatures differ between them but so also does the sky and the left roadside berm appearance.

Each run had a frame level ground truth, that is the (x, y) location in each frame of each of the 50 buried targets. We have previously [7] developed a method of segmenting video frames into 384x384 image chips that either do or do not contain a buried target. If the image chip does contain a target, we call that image chip a true target (TT). Otherwise, we call the image chip a false alarm (FA).

3. METHODOLOGY

The genetic program evaluates its candidate solutions against a set of 129x65 image chips. These image chips are extracted from individual frames of the FLIR video feed according to the procedure noted below. The segmented image chips are then run evaluated by each candidate solution during each generation of the algorithm's run. The method of evaluation is described below. Following this, a description of the genetic program is given.

3.1. Candidate Solution Representation and Evaluation

Candidate solutions are represented as a forest of parse trees, with each tree representing the formula for a single statistic. Details on the rules for each of these parse trees is given in Table 1, and a sample representation is given by Figure 3. This structure was chosen for the candidate solutions because it is flexible enough to represent either the Histogram of

Oriented Gradients (HOG) or Local Binary Pattern (LBP) feature descriptors. In other words, the HOG and LBP algorithms are particular instances of this more general candidate solution structure.

Table 1: Details of candidate solution initialization and representation

Parameter	**Value**
Functional nodes	Addition, subtraction, multiplication, division, greater than, less than, unary negation
Terminal nodes	Image pixel intensity (in 9x9 sub-window), randomly initialized constant
Initialization	Full tree
Maximum tree depth	4

When a candidate solution is evaluated against a given image chip, the statistic from each of its parse trees is computed for every 9x9 sub-window in the image chip. Once the sub-window around every pixel in the image is computed, a histogram is computed based on the aggregation of the statistic set across all sub-windows. The parameters representing this histogram are then used as features for a Support Vector Machine (SVM).

The fitness of a candidate solution is defined as the average correct classification accuracy across a 5-fold cross validation performed on the dataset described previously, with some modification. Namely, we have modified the composition of the dataset such that there is roughly an equal number of true targets (TTs) and false alarms (FAs) in each fold. This left us with a dataset of 457 image chips, with about 90 chips per fold. However, we did not always use the full dataset when evaluating the fitness of the population. Instead, for every generation we constructed a random subset of 200 image chips by randomly sampling the full dataset. We then used this reduced dataset to evaluate the fitness of every member of the population. After this completed, we then evaluated the fitness of the highest-performing candidate solution using the full dataset of 457 images. This fitness value has no effect on the behavior of the genetic algorithm, and is just used to track overall progress.

The reasoning behind using a reduced, randomly sampled dataset to perform the fitness evaluation for every candidate solution is two-fold. The primary reason it was introduced to the system was to reduce the amount of time required to perform the fitness evaluation, allowing us to run the algorithm for larger populations. Only after doing that did we realize we could randomly select a different subset of the image chip dataset on every generation to introduce a "dynamic environment" [8] for the candidate solutions to evolve in. By using a non-static fitness function, we are biasing our search towards discovering candidate solutions that generalize well.

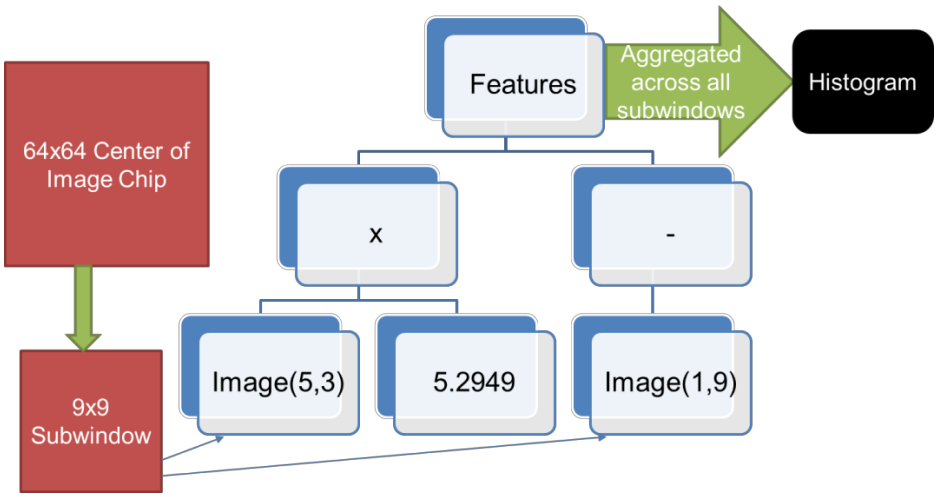

Figure 3: Outline of candidate solution evaluation

3.2. Genetic Programming Algorithm

The genetic algorithm (GA) is fairly traditional in its setup, with the high level details of its organization outlined in Table 2. This table includes the parameters for our largest test, in which we ran the GA for 100 generations with a population size of 160. During this test we mutated an offspring with probability 0.01, and performed crossover between two parents with probability 0.95.

The operation of mutation proceeds by traversing every node in the parse tree forest for a given candidate solution and modifying that node's value with probability p_m. If a node is selected for modification, it is reinitialized in a uniformly random manner to a new functional or terminal node, depending on where the node is in the tree. While mutating a single node in a candidate solution may not produce that much of a change, it is important to remember that an arbitrary number of nodes will be modified whenever a candidate solution undergoes mutation. In this way, we still allow for dramatic alterations in a candidate solution during mutation

On every generation of the GA, the crossover operator creates λ offspring solutions from the previous generation's population of μ solutions. With probability p_x, each call to the crossover operator produces a new offspring by mixing two parent solutions together. The remainder of the time, the offspring is just the first parent sent to the crossover operator. The parent solution used in crossover are chosen at random using tournament selection. In our implementation, tournament selection proceeds by selecting 5 members of the population at random, and then selecting the one with the top performance. This approach allows for the possibility of sub-optimal solutions being chosen as parents, which a more elitist approach such as roulette wheel selection would significantly decrease the probability of occurring.

Once the parents have been selected for crossover, a new offspring is created by transplanting a sub-tree of the second parent onto a random location in the first parent. We implemented this by first selecting a node from each parent using uniform random sampling, and then moving, or transplanting, the chosen node and its sub-tree from the second parent into the location of the chosen node in the first parent.

After we have completed crossover and mutation for a given generation, we then evaluate the fitness of every member in the population as well as every offspring produced this generation. Our fitness evaluation step differs from the prototypical GA in that we change the dataset which is used in fitness evaluation in every generation. We do this by selecting 200 image chips at random from the full dataset of 457 image chips before fitness evaluation, and then use this set of 200 image chips to evaluate the fitness of every candidate solution. This technique both reduces the computational overhead of fitness evaluation and provides a dynamic environment for the population to evolve in, as mentioned previously. Once the fitness of every candidate solution in a generation has been computed, we perform survival selection on the $(\mu+\lambda)$ candidate solutions by selection by keeping the top μ candidate solutions in terms of fitness.

Table 2: Description of Genetic Algorithm

Parameter	Value
Population size μ	160
Number of generations	100
Fitness function	Average classification accuracy using an SVM to classify sample image chip dataset
Probability of mutation p_m	0.01
Probability of crossover p_x	0.95
Parent selection	Tournament selection of size 5
Survival selection	(μ+λ) selection

4. RESULTS

4.1. Genetic Programming Algorithm Performance

The algorithm was executed for 100 generations for a population of 160 candidate solutions in an offline mode. There is a large overhead associated with this system due to the large number of fitness evaluations to be performed, but since this process never needs to be run in a real-time system this is acceptable. The progress of the algorithm was tracked per generation based on the top fitness value achieved during each generation, and is shown in blue in Figure 4. Also shown in Figure 4 is the baseline performance of the original HOG and LBP features being used, which is represented as a horizontal orange line. As can be seen in the figure, the performance of the top candidate solution per generation tends to increase along a roughly logarithmic growth curve. This suggests that while additional generations may have yielded improved fitness values, we were already close to the optimal performance of the algorithm and had passed the point of diminishing returns within respect to the number of generations.

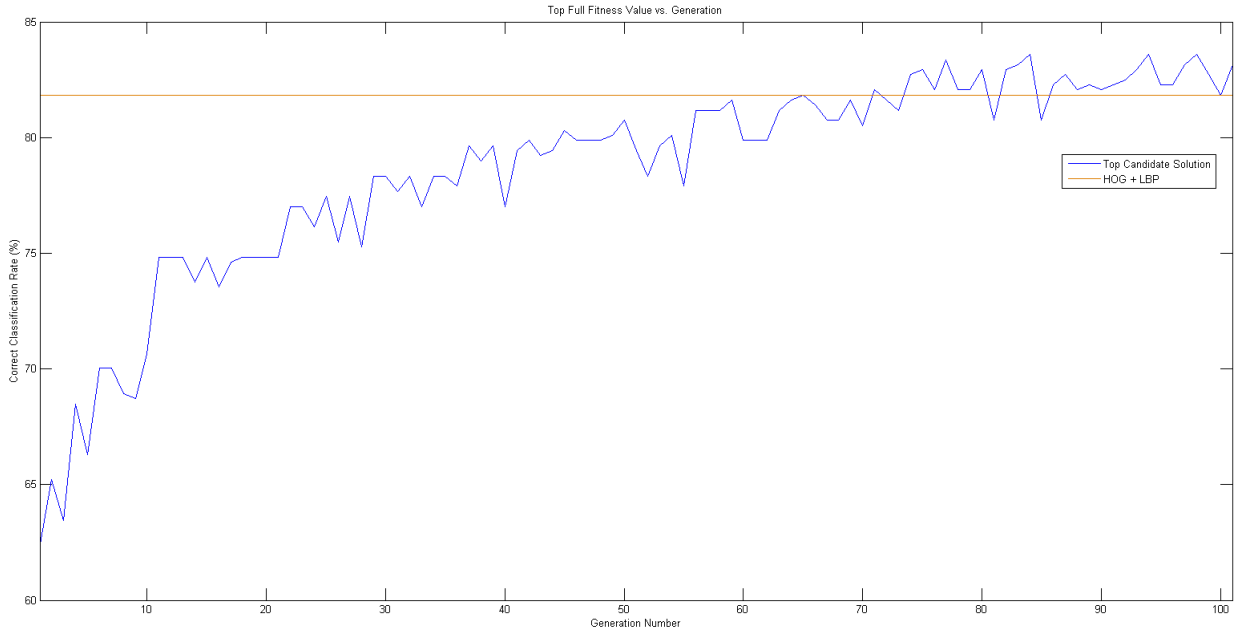

Figure 4: Top evolved solution performance compared with benchmark fitness of HOG and LBP trained classifier

4.2. Top Evolved Solution

The top evolved solutions were interesting in that the dimension of the feature vector they produced tended to be quite low (around 10) when compared to that of HOG and LBP combined (140 features). The individual statistics in the top solutions were typically indecipherable expressions, sometimes as many as 4 levels deep in the parse tree. An example of one of these expressions is given in Figure 5. This was disappointing to us, as we had hoped that our system would be able to discover new classes of image feature extraction algorithms. Unfortunately, even if these candidate solutions were functioning in a completely novel manner, it is very difficult to tell due to the obfuscated nature of their representations.

```
(img(7,5)*(((((img(7,1)+img(6,2))-img(4,9))-
(img(2,4)+img(9,6)))+img(3,3))/((0.42945/(img(1,1)-
0.64567))+(img(4,9)/(img(7,9)+0.055534)))))
```

Figure 5: Sample evolved statistic

In this way, the evolved solutions are much more similar to the "black boxes" typically seen when using Convolutional Neural Networks (CNNs) to extract features from images [9]. These networks learn non-linear transforms on the underlying image space whose representation is meaningless to a human researcher after training. The primary differences between our approach and CNNs is that we have a broader scope of operators available (any arithmetic function versus addition, multiplication and sigmoidal function) and that the learning process in our case is done in a massively parallel, less-directed fashion. Whereas training in a CNN is guaranteed to improve performance, each new generation in our system may inadvertently decrease performance due to the dynamic nature of our fitness function (as seen in Figure 4). However, our system is better at avoiding local minima in terms of classification accuracy as there are several solutions being tested in parallel as opposed to the single network being trained in typical CNN setups.

Another way of looking at the evolved solutions is that they each produced a set of filtered image chips, with one corresponding to each feature in the evolved solution. A filtered image chip can be produced by creating a new image from an existing image chip such that the pixel intensity is equal to one of the statistics in a candidate solution. An example of these filtered images is given in Figure 6.

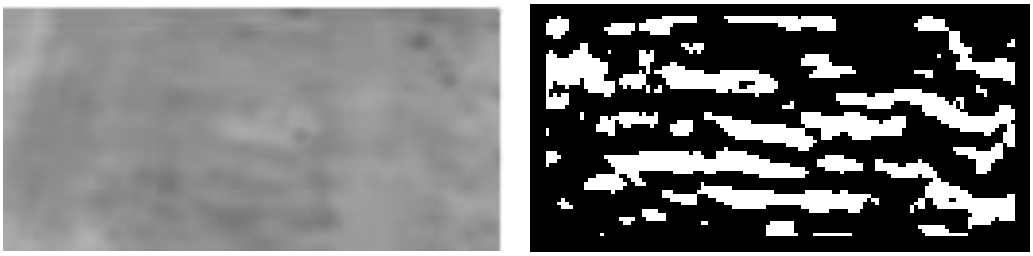

Figure 6: Sample image chip (left) with result of filtering the image with one statistic of a candidate solution (right)

4.3. Comparison with HOG and LBP Algorithms

As can be seen in Figure 4, after approximately 75 generations our top performing candidate solutions begin to outperform the benchmark classifier using HOG and LBP features. The best classification accuracy achieved by evolved features on the complete dataset was 84.03%, an improvement of 2.2% over the classification accuracy of an SVM trained using HOG and LBP features (81.83%). This result is encouraging, and goes a long towards validating our approach.

This result suggests that it may be possible to generate "custom" image feature extraction for any specific dataset that outperform "generic" image feature extraction algorithms such as HOG and LBP. As evidenced by our top solution having a feature vector of length 9, it is also possible to dramatically reduce the dimensionality of the feature vector using this approach.

The primary drawback in using the approach presented here is that it requires a very large number of computations to be performed. Fortunately, the nature of the system is conducive to being run in parallel which would greatly reduce the runtime if the researcher has access to a massively parallel computer. However, this system is not very appropriate for

situations where the researcher either does not have access to a sufficiently fast computer or where the researcher does not have time to wait for the system to run long enough to produce a useful result. In these cases, more generic and proven feature extraction algorithms such as HOG and LBP should be used, as it is unknown whether an evolved solution will give too dramatic of an increase over their performance (such as in our case, where the total classification error began at roughly 20%, and was only reduced by about 10%).

5. CONCLUSIONS

We have presented here a method of generating novel image feature extraction algorithms that are automatically tailored to a specific problem domain. In the application presented here, we have demonstrated that this approach is capable of producing new algorithms that outperform some of the more popular feature extraction algorithms, namely the Histogram of Oriented Gradients (HOG) and Local Binary Pattern (LBP) algorithms. Additionally, the optimal solution produced by our system reduced the dimensionality of the feature vector from 140 for the combined HOG and LBP case to just 9. This system could conceivably be applied to any image classification dataset, but the large time and computational complexity requirements of the system should be taken into account when deciding between this system and traditional measures such as HOG and LBP. Moving forward, this approach could be extended by introducing new arithmetic operators or including filters in the representation of the candidate solutions. The techniques shown here are more basic in order to serve as validation for the concept of our system in general.

6. ACKNOWLEDGMENTS

This Work funded by ARO grant 57940-EV to support U. S. Army RDECOM CERDEC NVESD.

References

[1] Li B., Chellapa R., Zheng Q., Der S., Nasrabadi N., Chan L., Wang L., “Experimental Evaluation of FLIR ATR Approaches—A Comparative Study”, *Computer Vision and Image Understanding* 84, 5–24 (2001).

[2] Stone, K., Keller, J. M., Popescu, M., Havens, T.C., Ho, K. C., "Forward Looking Anomaly Detection via Fusion of Infrared and Color Imagery", Proc. of SPIE 7664, 2010.

[3] Yu Y., Guo L., “Infrared Small Moving Target Detection Using Facet Model and Particle Filter”, Proc of 2008 Congress on Image and Signal Processing, pp. 206-210.

[4] Schmuggea,T., French, A., Ritchie, JC, Rango A., Pelgrum H., “Temperature and emissivity separation from multispectral thermal infrared observations”, Remote Sensing of Environment 79 (2002) 189– 198.

[5] Winter,E. M., Fields,D. J., Carter, M. R., Benett, C. L., Lucey,P. G., Hohnson,J. R., Horton,K. A., and Bowman, A. P., “Assessment of Techniques for Airborne Infrared Land Mine Detection”, Proc.of the Third International Airborne Remote Sensing Conference and Exhibition, Copenhagen, Environmental Research Institute of Michigan, Ann Arbor, Vol. II, 1997, pp. 44-51.

[6] Brian Tuomanen, Kevin Stone, Timothy Madison, Mihail Popescu, James Keller, “Buried target detection in FLIR images using Shearlet features”, Proc. SPIE 8709, Detection and Sensing of Mines, Explosive Objects, and Obscured Targets XVIII, 870919 (June 7, 2013); doi:10.1117/12.2014773.

[7] Lewis, D.B.; Keller, J.M.; Popescu, M.; Stone, K., "Dirt road segmentation using color and texture features in color imagery," Computational Intelligence for Security and Defence Applications (CISDA), 2012 IEEE Symposium on , vol., no., pp.1,6, 11-13 July 2012; doi: 10.1109/CISDA.2012.6291522

[8] Kazarlis S. and V. Petridis, Varying fitness functions in genetic algorithms: studying the rate of increase in the dynamic penalty terms, Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, Berlin, Springer Verlag, 211-220, 1998.

[9] Lawrence, S.; Giles, C.L.; Ah Chung Tsoi; Back, A.D., "Face recognition: a convolutional neural-network approach," Neural Networks, IEEE Transactions on , vol.8, no.1, pp.98,113, Jan 1997; doi: 10.1109/72.554195