Focus-of-Attention Strategies for Finding Discrete Objects in Multispectral Imagery

Neal R. Harvey and James Theiler

Space and Remote Sensing Sciences,
Los Alamos National Laboratory,
Los Alamos, NM 87544, USA

ABSTRACT

Tools that perform pixel-by-pixel classification of multispectral imagery are useful in broad area mapping applications such as terrain categorization, but are less well-suited to the detection of discrete objects. Pixel-by-pixel classifiers, however, have many advantages: they are relatively simple to design, they can readily employ formal machine learning tools, and they are widely available on a variety of platforms. We describe an approach that enables pixel-by-pixel classifiers to be more effectively used in object-detection settings. This is achieved by optimizing a metric which does not attempt to precisely delineate every pixel comprising the objects of interest, but instead focusses the attention of the analyst to these objects without the distraction of many false alarms. The approach requires only minor modification of exisiting pixel-by-pixel classifiers, and produces substantially improved performance. We will describe algorithms that employ this approach and show how they work on a varitety of object detection problems using remotely-sensed multispectral data.

Keywords: multispectral imagery, object detection, machine learning, image processing, target recognition

1. INTRODUCTION

In increasing quantities and with increasingly higher quality, remote sensing imagery is providing quantitative renditions of our environment, of our agriculture, of our cities, and of our highways and airports and shopping centers and golf courses. The fundamental challenge of remote sensing is to identify what's in the scene, given an image taken from a distant vantage point. How healthy are our crops, what is growing in our forests, where are the invasive species concentrated? But in addition to these broad-area features, there is considerable interest in exploiting high resolution remote sensing imagery to identify locations of targets: What is the density of saguaro cactii in the desert, how many cars are on the highway and on what desolate hillside is the airplane that went missing two days ago?

Multispectral (and to an even greater extent, hyperspectral) imagery provides the opportunity to treat images as collections of individual pixels. Each pixel provides spectral information about one position (physical location) in the scene, and each pixel can be analyzed independently of the others. For applications where this is appropriate, tools from machine learning can very naturally be applied. Training data is a collection of pixels which have been labelled either from ground truth campaigns or from an image analyst's expertise. The classifier is a function that takes a pixel as input (that is, a vector whose components correspond to the value of that pixel in each of the spectral channels), and provides a label as output.

Some image-oriented machine learning tools employ spatial context as part of the input (one approach[1] achieves this by producing new feature or "scratch" channels that are obtained from image processing operations – such as smoothing or dilation – applied to the raw spectral channels). But these also provide as output a separate label for each pixel.

For object detection problems, what is ultimately desired is not necessarily an image in which each pixel has been individually labelled as object or not-object. Often, the desired output is not even an image at all, but a list of object locations. Nonetheless, a pixel-by-pixel classification of the image can still be extremely useful

Authors' emails: harve@lanl.gov, jt@lanl.gov

to the analyst. If there is an adequately low false alarm rate then the analyst can manually check out all the target-identified pixels and identify which ones correspond to real targets.

But it bears remarking that the analyst's effort in doing this is not simply proportional to the number of incorrectly labelled pixels. A labelled image archive with false alarms scattered widely across the data will take a long time to evaluate; on the other hand, if the same number of false alarm pixels are all confined to one corner of one of the images in the archive, then the evaluation can proceed much more rapidly.

Our goals in this paper are: first, to characterize the utility of pixel-by-pixel classification results in a way that accounts for the effort on the part of the analyst to follow up on true detections and false alarms; and second, to propose simple modifications to existing pixel-based algorithms so that they will produce more useful pixel-by-pixel classifications.

2. OBJECT RECOGNITION

We begin with the statement that object *recognition* involves (at least) three different elements: *detection*, *segmentation*, and *identification.* The first step is to determine that there is an airplane in the image; the second step is to outline the contours of the airplane by specifying which pixels correspond to the airplane and which to the background clutter; the third step is to identify the aircraft type (*e.g.,* an F-15).

Our interest here is in automating just the detection task. For a lot of practical scenarios of interest (though certainly not all), the analyst can perform the segmentation and the identification relatively quickly. But the initial detection of particular targets in a large archive can be can be tedious and error-prone.

Our aim is to find the needle in the haystack, not to find out who put it there. To belabor the metaphor: our aim is to focus the analyst's attention on the candidate needles, so the analyst can concentrate on the details of the needle, not the distractions of the haystack.

Perhaps the most straightforward way to automate the search for objects in images is to train a classifier to identify which pixels are on-target and which pixels are off-target. This classifier is a function that takes as input the values of pixels in the local neighborhood of a pixel and produces as output a one or a zero, depending on whether or not the object is predicted to be in that pixel. A perfect classifier would identify those pixels that corresponded to objects in the image – it would identify all the pixels associated with each object, and would not identify any pixels that were not associated with any objects of interest. There are two kinds of errors in this scheme: missed detections and false alarms.

But for objects that are bigger than a single pixel, this is overkill!

To be useful to an analyst, an object *detector* need identify only a single pixel for each object. Once the attention has been focused, there is little gain to the analyst to have multiple pixels identified for a single object. Indeed, since the ultimate goal might be to produce a list of objects (and their positions), more than a single pixel per object might even incur an extra-pixel cost.

That is to say, the problem of object detection is actually easier than the identification, on a pixel-by-pixel basis, of which pixels correspond to the objects of interest and which do not.

Fig. 1 illustrates the different kinds of error:

- Missed detection: Since the goal is to detect the objects, a missed detection is always an error, and in some cases, the worst kind of error. We set the penalty for this error at 1.0.
- Isolated false alarm: The isolated false alarm draws the analyst's attention and requires some effort to resolve that it is a false alarm. We set the penalty for this error at λ, and that value can be greater or less than 1.0 depending on the scenario.
- Proximal false alarms: Two false alarm pixels that are adjacent also draw the analysts attention and demand resolution; but the cost of the effort to produce this resolution for two adjacent pixels is often much less than twice the effort for a single pixel.

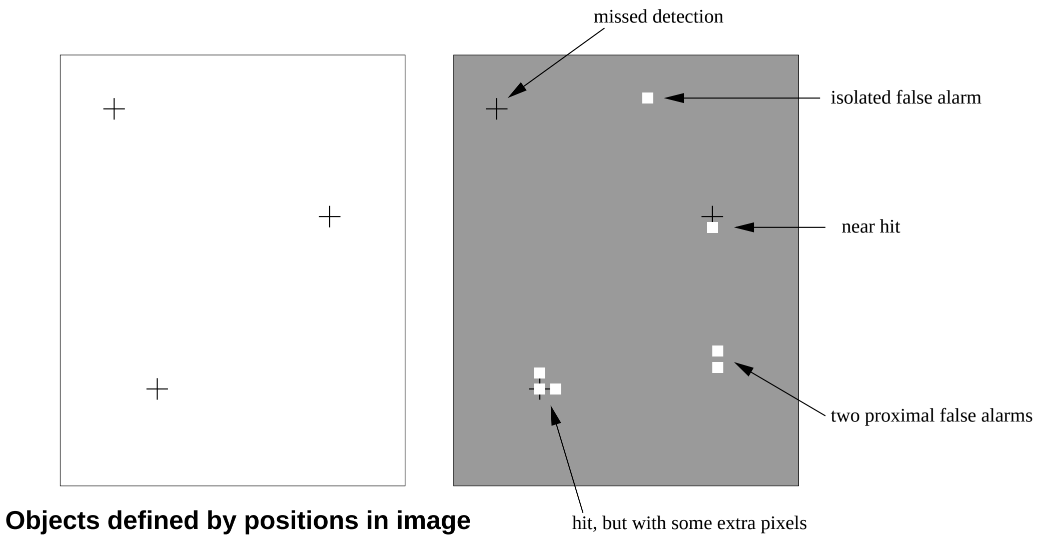
missed detection
isolated false alarm
near hit
two proximal false alarms
hit, but with some extra pixels
Objects defined by positions in image

Figure 1. Objects are defined by single-pixel positions in the image. An ideal pixel-by-pixel classifier will identify the pixels that define the objects and no other pixels. There are a number of qualitatively different ways that a pixel-by-pixel classifier can deviate from ideal. But some deviations are more expensive than others, from the point of view of an analyst's effort to confirm or reject a given "hit."

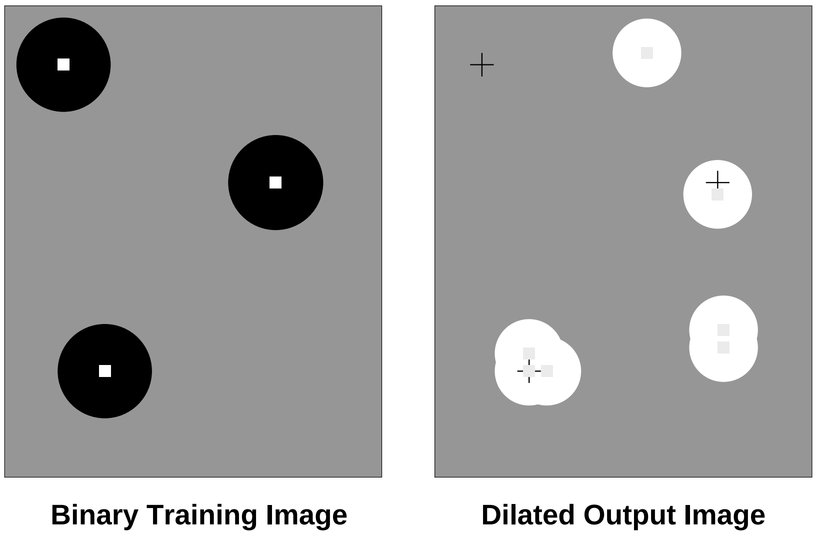

Figure 2. Binary training image and dilated output image.

- Near hit: Although the near hit nominally corresponds to both a missed detection and a false alarm, in practice it is almost as good as true detection.
- Extra pixels: Although the extra pixels are nominally false alarms, they can be rapidly resolved, and are certainly less costly than isolated false alarms. In general, one might like the penalty for an extra pixel to be near zero, but one could make argument for either sign. The extra pixels might be a benefit if the conjunction of several of them are somehow indicative of the size of the object or the estimated reliability of the detection.

3. SCORING STRATEGIES

The simplest scoring strategy is the pixel-by-pixel metric. Each pixel is treated independently and there are only two kinds of errors: missed detections and false alarms. We employ the parameter λ for the relative cost of a false alarm versus a missed detection. Choosing λ is clearly scenario-specfic. An algorithm that attempts to optimize missed detections and false alarms might have to employ such a parameter, but for our results, we will show ROC curves.

But we note that even the simple pixel-by-pixel metric can be finessed by an appropriate markup strategy, and we note that there are (at least) two such strategies that are appropriate for the object detection problem.

3.1. Markup strategies

We use the metaphor of "painting pixels" because it is colorful, it simplifies the explanation of what is done, and it actually corresponds to how our software works.

The most direct (and probably the most common) markup strategy is shown in Fig. 3(b). Here the entire object of interest is labelled (shown here as white) – that is, every pixel that corresponds to the object is identified as a target pixel. The background of non-target pixels is painted shown as gray. Unpainted pixels are agnostic, and it is not uncommon for a boundary between the object (white) and background (gray) pixels to remain unpainted.

Truer to the spirit of object detection, a second approach to markup is shown in Fig. 3(c). Here, instead of delineating the entire object, a single (white) pixel provides the location of the object. The background (gray) markup identifies the parts of the image where there are no objects. Around each object's single-(white-)pixel, there is a (black) disk which is not marked up at all. The radius of this disk is typically twice the size of the object of interest.

3.2. Loss function: two interpretations

Each missed detection has a cost of 1, and each false alarm has a cost of λ. This leads to:

$$L = n_{\mathrm{md}} + \lambda n_{\mathrm{fa}} \tag{1}$$

where n_{md} is the number of missed detections, and n_{fa} is the number of false alarms. The two terms in the loss function represent two qualitatively different scenarios, and it is not always obvious how to choose λ.

An alternative interpretation of this loss function is to again assert that there are two kinds of costs: missed detections and all alarms (true as well as false). Whether or not the alarm turns out to be false, there is a cost associated with having the analyst follow-up. If we assign each missed detection a cost of 1 and each alarm a cost of λ', then

$$L' = n_{\mathrm{md}} + \lambda'(n_{\mathrm{fa}} + n_{\mathrm{ta}}) \tag{2}$$

where n_{ta} is the number of true alarms.

Here $n_{\mathrm{obj}} = n_{\mathrm{md}} + n_{\mathrm{ta}}$ is the total number of actual objects, and is a fixed quantity, so we can write

$$L' = n_{\mathrm{md}} + \lambda'(n_{\mathrm{fa}} + n_{\mathrm{obj}} - n_{\mathrm{md}}) \tag{3}$$

or

$$L' = (1 - \lambda')n_{\mathrm{md}} + \lambda' n_{\mathrm{fa}} + \lambda' n_{\mathrm{obj}} \tag{4}$$

But since n_{obj} is fixed, then the transform $L = (L' - \lambda' n_{\mathrm{obj}})/(1 - \lambda')$ defines an equivalent loss function. If we further define $\lambda = \lambda'/(1 - \lambda')$ (or, equivalently, $\lambda' = \lambda/(1 + \lambda)$), then we have

$$L = n_{\mathrm{md}} + \lambda n_{\mathrm{fa}} \tag{5}$$

which mimics Eq. (1) and shows that both interpretations are equivalent. Note that in this second interpretation, it is important that $\lambda' < 1$. This makes sense: if an alarm (false or otherwise) were more expensive than a missed detection, then it would not be cost-effective even to find the real targets.

[[An exercise in obviousness, but I have to admit I wasn't at first entirely sure they were equivalent.]]

3.3. Reconstruction metric

The reconstruction metric is actually related to what is called the "multiple-instance" problem[2–4] in machine learning. In this metric an object is a (typically compact and contiguous) set of pixels; and the classification associated with the object is given by the "or" of the classifications assigned to each of the pixels. That is, if any of the pixels in the object are "hits" then the object is detected. Any hits outside the object are false alarms.

To apply the reconstruction metric, the objects in the test image must be fully marked up and separately identified.

Since the output of many pixel-by-pixel classifiers is a continuous real-valued quantity, this scoring is implemented by replacing all the pixels in each object with the maximum-valued pixel in that object.

This usually permits a much higher threshold than would be needed to hit all or most of the pixels in an object; and an increased threshold will in general produce fewer false alarms outside the object.

3.4. Dilation metric

The dilation metric is similar in spirit to the reconstruction metric, but it aims to be simpler, and uses the single-pixel-per-object markup. The idea is to take the solution that is obtained and dilate it by a fixed radius. This means that any (pre-dilation) hits need only be within that fixed radius in order for the single-pixel center to be hit by the (post-dilation) solution. The radius of this dilation is usually taken to be roughly half the size of the object of interest – that way, in similarity to the reconstruction metric, the pre-dilation solution only needs to hit a single pixel on the target for it to be counted as a hit in the final scoring. Note that the disk of unmarked data in the single-pixel markup should be at least twice the radius of the dilation.

One advantage (compared to the reconstruction metric) is that the penalty for proximal false alarms is much less than for isolated false alarms. That is, a dozen false alarms in twelve contiguous pixels will be much less than a dozen false alarms spread out over the image archive.

A disadvantage is that the size of the dilation is not tailored to the shape of the object of interest; so if the object is not particularly compact, the fit may not be very good.

4. ALGORITHMS

Three algorithms were used to optimize classifiers for the object detection tasks/experiments described in this paper:

- Genie,[1,9,10] with the "standard" pixel-based metric. The classifiers obtained using this algorithm are referred to with the expression *Genie Vanilla* in the remainder of this paper.
- Genie, with the "object" reconstruction-based metric, as described above. The classifiers obtained using this algorithm are referred to with the expression *Genie Object* in the remainder of this paper.
- Afreet,[5] with the "standard" pixel-based metric. The classifiers obtained using this algorithm are referred to with the expression *Afreet* in the remainder of this paper.

Genie is a system developed at Los Alamos National Laboratory, for the automated generation of feature extraction/classification tools. At its core, Genie utilizes a hybrid evolutionary-algorithm-based system capable of searching for image processing pipelines optimized for specific image feature extraction tasks.

Afreet, like Genie, is a machine learning system developed for pixel-by-pixel image classification. Afreet differs from Genie in that it uses a Support Vector Machine[6] with a hill-climbing method for constructing useful spatio-spectral image features, instead of genetic programming[7] and more standard statistical classifiers (such as a Fisher linear discriminant[8]), which Genie uses for the same task.

5. DATA SETS AND NUMERICAL EXPERIMENTS

5.1. Data Sets

Two data sets were employed:

Aircraft Data: 10-band data ranging from 0.42μm (blue) to 2.35 μm (short-wave IR) was collected over the tarmac at Nellis AFB in Nevada, using the Daedalus 1268 multi-spectral scanner (`http://www.sensytech.com`). These data were provided by DOE Remote Sensing Laboratory, NV, USA.

Car Data: 3-band (RGB) data were provided by the British Ministry of Defence's Defence Science and Technology Laboratory (DSTL) (http://www.dstl.gov.uk). The original images were taken using a wet-film SLR camera from a helicopter somehwere over England. The digital data was obtained from scanning the film negatives.

5.2. Numerical Experiments

We conducted a set of experiments, with the aim of determining whether, by making the simple modifications described above to some pixel-by-pixel classifiers, we could obtain improved performance on object-detection tasks.

We set ourselves the tasks of comparing several different feature-extraction algorithms' performance in detecting objects of interest within the two data sets described above. For the aircraft data, there were 3 detection tasks: detect the A-10, F-15 and F-16 aircraft in the imagery. For each aircraft type, approximately half of the aircraft of that type in the (single available) image were used during training, and the remaining aircraft were used for testing. For the car data, there was one detection task: detect the red cars in the images. One entire image was used for training purposes and two other complete images were used for testing.

Figure 3 shows examples of the training/testing mark-up (*i.e.,* the labelled data) that was provided for the training of the algorithms and the calculation of the performance metrics. Figure 3 (a) shows an extracted region of band 2 of the original Daedalus data. One can clearly see an A-10 aircraft taking up a large fraction of this image segment. Figure 3 (b) shows the labels provided by a human analyst for this image, which were used for training the classification algorithms and in the calculation of the reconstruction- and dilation-based performance metrics. The white pixels show those deemed by the analyst to belong to the object of interest (*i.e.,* they belong to the "true" class). The gray pixels show those deemed by the analyst to belong to the background (*i.e.,* they belong to the "false" class). The black pixels show those pixels not marked by the analyst as belonging to either class.

In general, the radius of the dilation is taken to be roughly half the size of the object of interest, and the radius of the "don't care" region is typically double that. For these datasets, we dilate with radii of 17 pixels for the cars, 25 pixels for the F-16 aircraft, 30 pixels for the A-10 aircraft, and 45 pixels for the F-15 aircraft.

6. RESULTS

Table 1 shows the performance of the different classifier algorithms, for the various metrics, on the aircraft detection tasks. The entries in the table are the minimum false-alarm rates for a detection rate of at least 0.9; *i.e.,* these are single points on the ROC curves for each classifier result, and performance metric. These results are provided with respect to both training and testing data.

Table 2 shows the performance of the different classifier algorithms, for the various metrics, on the red car detection problem. As with Table 1, the entries in the table are the minimum false-alarm rates for a detection rate of at least 0.9, and are provided for both training and testing data.

Figure 4 show some typical ROC curves, and the DR=0.9 threshold that was used to produce the data in Tables 1 and 2.

Figure 5 shows the actual classification results, for the A-10 aircraft detection problem, for a small region of the Daedalus image data. Each image corresponds to the particular points on the ROC curves, as shown in

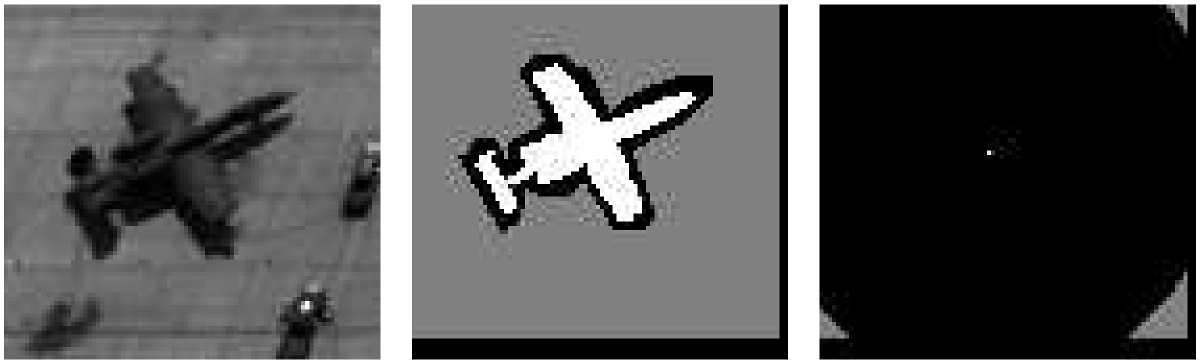

Figure 3. (a) Extracted region of raw Daedalus data (band 2), on which results are overlaid in Fig. 5. (b) Extracted region of raw Daedalus data, showing "true" and "false" object pixels for reconstruction- and pixel-based metrics. White pixels represent "true" pixels, gray pixels represent "false" pixels and black pixels are those which haven't been given a label. (c) Extracted region of raw Daedalus data, showing "true" and "false" object pixels for dilation-based metric. White pixels represent "true" pixels, gray pixels represent "false" pixels, and black pixels are those which haven't been given a label.

Table 1. Performance of classifiers on Aircraft-detection, according to the various metrics: Minimum FAR that achieves a DR of $\geq$ 0.90.

A-10						
	Training Performance			Testing Performance		
	Genie-vanilla	Genie-Object	Afreet	Genie-vanilla	Genie-object	Afreet
Pixel-Based	0.0023824	0.0394900	**0.0023824**	0.0023182	0.0047292	**0.0046031**
Reconstruction-based	0.0005963	**0.0000000**	0.0001847	0.0006978	**0.0000004**	0.0002890
Dilation-Based	0.0713459	**0.0000000**	0.0245751	0.0502306	**0.0007664**	0.0230410
F-15						
	Training Performance			Testing Performance		
	Genie-vanilla	Genie-Object	Afreet	Genie-vanilla	Genie-object	Afreet
Pixel-Based	0.0316502	0.0500145	**0.0281712**	0.0019190	0.0136419	**0.0013171**
Reconstruction-based	0.0139340	**0.0003006**	0.0069867	0.0006253	**0.0000140**	0.0003690
Dilation-Based	0.4146870	**0.2635340**	0.3006040	0.0606033	**0.0240667**	0.0318361
F-16						
	Training Performance			Testing Performance		
	Genie-vanilla	Genie-Object	Afreet	Genie-vanilla	Genie-object	Afreet
Pixel-Based	**0.0240822**	0.2022360	0.0257433	0.0037264	0.0538510	**0.0036191**
Reconstruction-based	0.0044319	**0.0000093**	0.0022148	0.0004646	**0.0000016**	0.0004132
Dilation-Based	0.1085360	**0.0164662**	0.1156440	0.0178323	**0.0025055**	0.0293159

Table 1 (*i.e.,* for the point on the ROC curve with a minimum FAR and a DR $\geq$ 0.90), for each classifier and performance metric, with respect to the test data.

Figure 6 shows the actual classification results, for the F-16 aircraft detection problem, for a small region of the Daedalus image data. Each image corresponds to the particular points on the ROC curves, as shown in Table 6 (*i.e.,* for the point on the ROC curve with a minimum FAR and a DR $\geq$ 0.90), for each classifier and performance metric, with respect to the test data.

Table 2. Performance of classifiers on red car-detection, according to the various metrics: Minimum FAR that achieves a DR of ≥ 0.90.

	Training Performance			Testing Performance		
	Genie-vanilla	Genie-Object	Afreet	Genie-vanilla	Genie-object	Afreet
Pixel-Based	0.002885	0.609230	**0.001334**	0.0029379	0.596666	**0.0010193**
Reconstruction-based	0.0001628	**0.0000087**	0.0004300	0.0912444	**0.0000024**	0.0000667
Dilation-Based	0.0075280	**0.0024457**	0.0053554	0.0007284	**0.0004206**	0.0012013

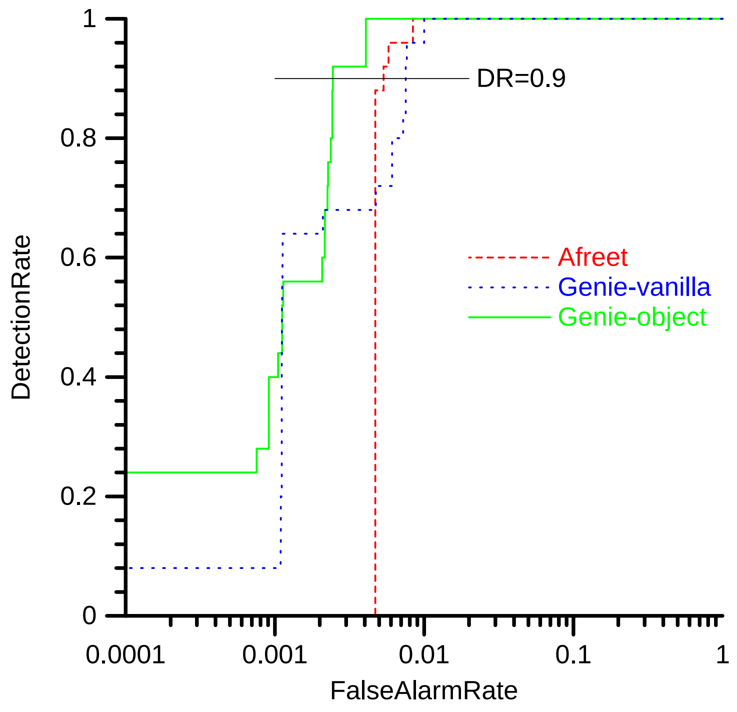

Figure 4. Detection rate versus false alarm rate for the red car detection problem. These are in-sample results (*i.e.* for the training data), and are based on the dilation metric. The horizontal line at DR=0.9 indicates the point on the ROC curves where the performance was measured. Where the ROC curves intersect this line provides the minimum false alarm rate for DR>0.9.

7. CONCLUSIONS

We have described and demonstrated how, by making relatively simple modifications to pixel-by-pixel classifiers, it is possible to enable them to be used more effectively for object-detection problems. These simple modifications involve optimizing a metric that focusses the attention of an analyst to the objects, but does not attempt to precisely delineate the objects of interest. In this way one can develop classifiers that are well-suited to the detection of discrete objects, having substantially improved performance, but, at the same time also retain many of the advantages of pixel-by-pixel classifers, such as design simplicity, wide availablity, and the option of employing formal machine learning tools.

The reconstruction metric uses markup that is directly related to the object identification task, and it does not require the specification of dilation radii. On the other hand, it imposes upon the analyst the task of producing far more detailed markup than is needed for the dilation metric.

All algorithms described in this paper were trained with the same markup data, where the analyst had been careful to fully-delineate the objects of interest. Future work will include development of detection algorithms

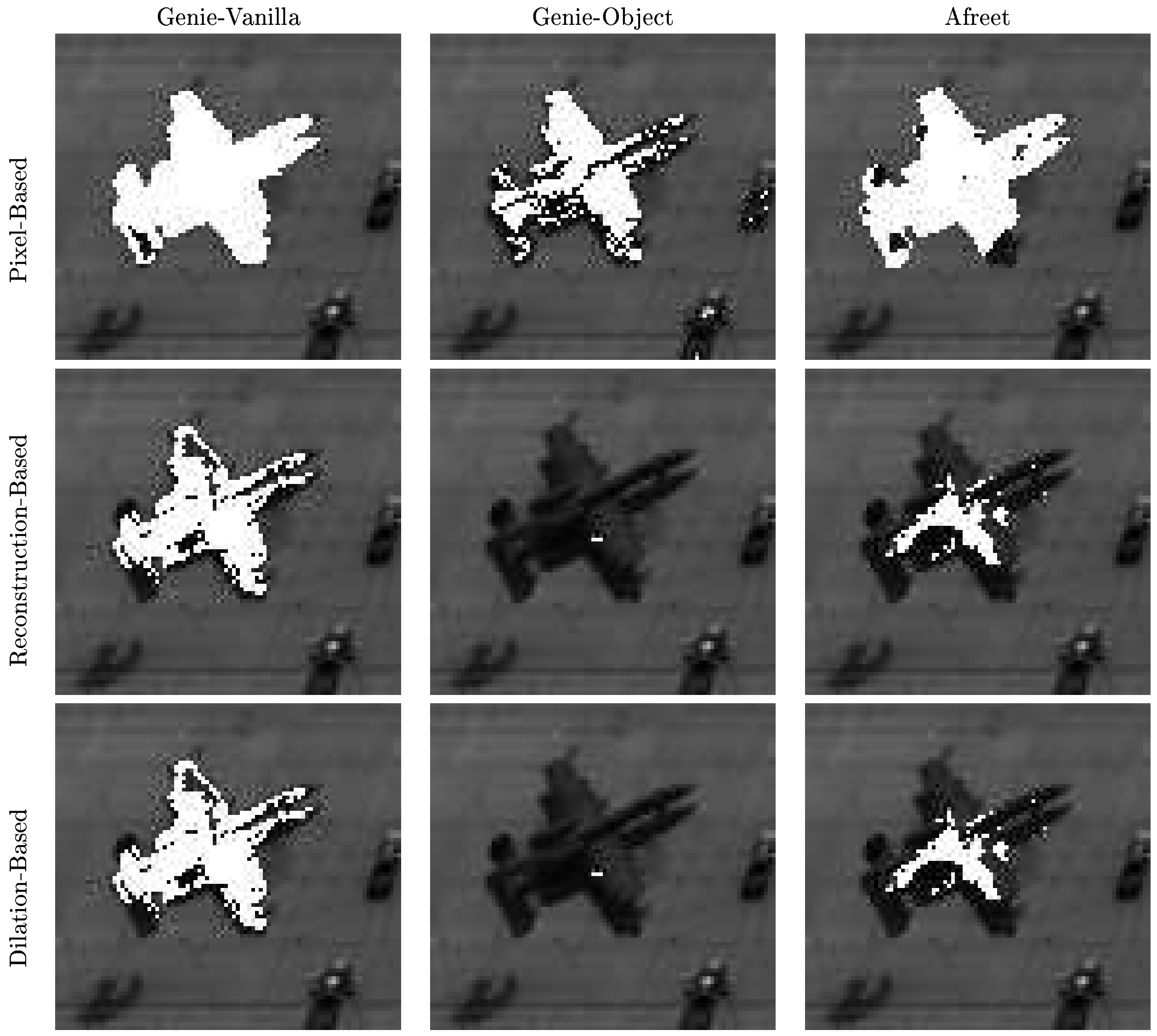

Figure 5. Results of classifiers on A-10 aircraft detection, according to the various metrics: Minimum FAR that achieves a DR of ≥ 0.90, on test data. White pixels are "hits." The bright spot on the truck in the lower right part of the image is part of the image itself, and is not a false alarm. Each column of images correspond to the same solution, but at different thresholds as needed to producem minimum FAR at DR ≥ 0.90 according to the different metrics. Note that the pixel-based metric produces solutions that try to fill out the whole extent of the airplane; just by re-adjusting the threshold, fewer pixels are identified as part of the target.

based on the easier-to-produce dilation-based markup.

ACKNOWLEDGMENTS

We are grateful to the Los Alamos ISIS (Intelligent Searching of Images and Signals) team for the Genie and Afreet software that was used in this project. We thank Dr. Peter Harvey of the Defence Science and Technology Laboratory, MOD, UK; and we acknowledge the DOE (Department of Energy) Remote Sensing Laboratory for providing data that was used in this study. We also appreciate the ISIS and RWML teams, and John Irvine of SAIC (Science Applications International Corporation), for many productive discussions of object detection issues. Finally we acknowledge funding from the Los Alamos LDRD (Laboratory Directed Research and Development) program and from DOEs DAPS (Deployable Adaptive Processing Systems) project.

REFERENCES

1. N. R. Harvey, J. Theiler, S. P. Brumby, S. Perkins, J. J. Szymanski, J. J. Bloch, R. B. Porter, M. Galassi, and A. C. Young, "Comparison of GENIE and conventional supervised classifiers for multispectral image feature extraction," *IEEE Trans. Geosci. and Remote Sens.* **40**, pp. 393–404, 2002.
2. T. G. Dietterich, R. H. Lathrop, and T. Lozano-Perez, "Solving the multiple instance problem with axis-parallel rectangles," *Artificial Intelligence* **89**(1-2), pp. 31–71, 1997.
3. O. Maron and A. L. Ratan, "Multiple-instance learning for natural scene classification," in *Proc. 15th International Conf. on Machine Learning*, pp. 341–349, Morgan Kaufmann, San Francisco, CA, 1998.
4. Q. Zhang and S. Goldman, "EM-DD: An improved multiple-instance learning technique," *Neural Information Processing Systems* **14**, p. 10, 2001.
5. S. Perkins, N. R. Harvey, S. P. Brumby and K. Lacker, "Support Vector Machines for Broad Area Feature Extraction in Remotely Sensed Images", in *Proc. SPIE 4381*, pp. 286–295, 2001.
6. N. Cristianini, J. Shawe-Taylor, "Introduction to Support Vector Machines: And Other Kernel-Based Learning Methods", *Cambridge University Press*, Cambridge, U.K., 2000.
7. J. R. Koza, "Genetic Programming : On the Programming of Computers by Means of Natural Selection (Complex Adaptive Systems)", *MIT Press*, Cambridge, MA, USA, 1992.
8. C. M. Bishop, *Neural Networks for Pattern Recognition*, pp. 105–112, Oxford University Press, 1995.
9. J. Theiler, N. R. Harvey, S. P. Brumby, J. J. Szymanski, S. Alferink, S. Perkins, R. Porter and J. J. Bloch, "Evolving Retrieval Algorithms with a Genetic Programming Scheme", in *Proc. SPIE 3753*, pp. 416–425, 1999.
10. N. R. Harvey, S. Perkins, S. P. Brumby, J. Theiler, R. B. Porter, A. C. Young, A. K. Varghese, J. J. Szymanski and J. Bloch, "Finding golf courses: The ultra high tech approach", in *Evolutionary Image Analysis, Signal Processing and Telecommunications*, Poli, et al. Springer-Verlag, 2000.

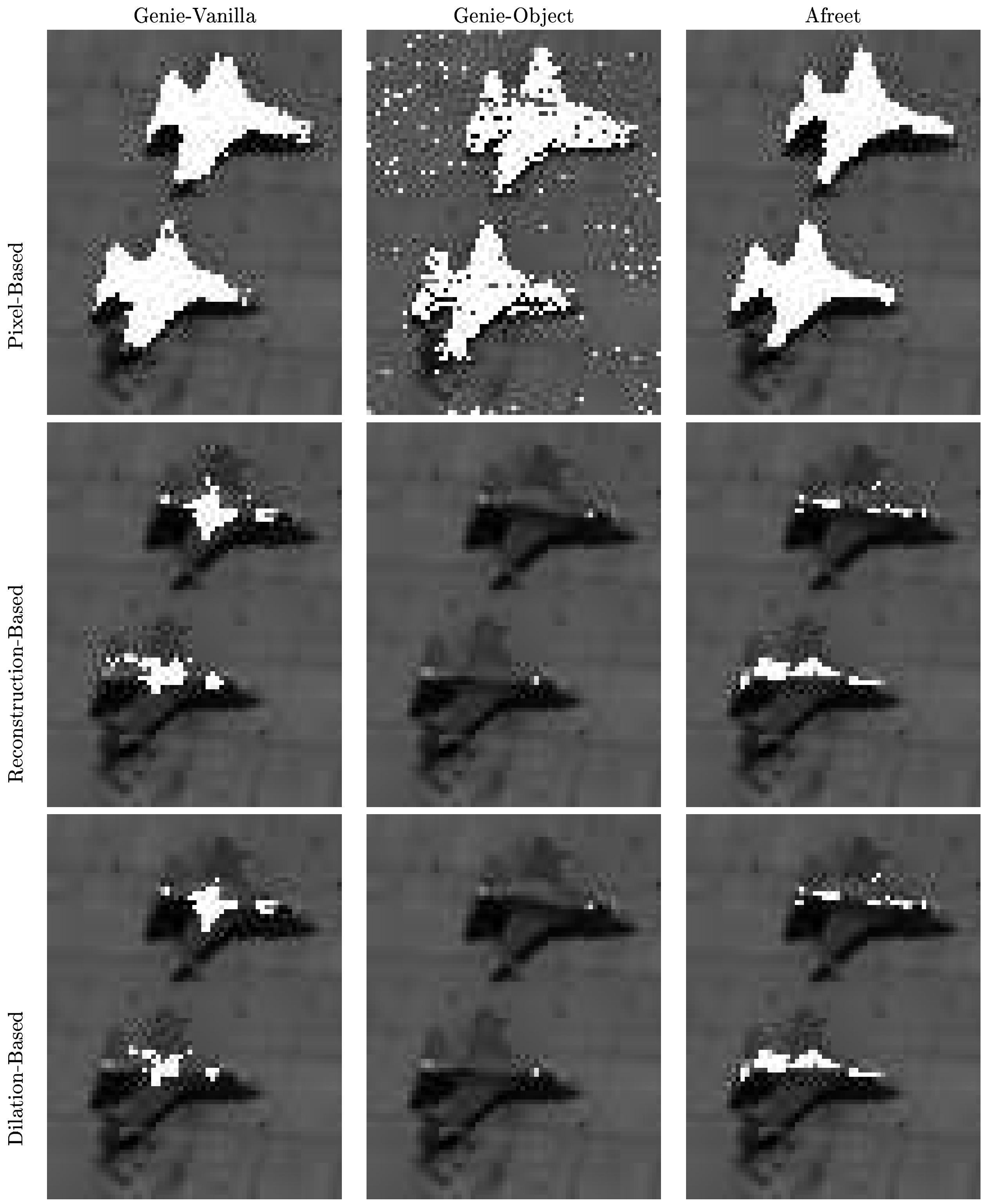

Figure 6. Results of classifiers on F-16 aircraft detection, according to the various metrics: Minimum FAR that achieves a DR of ≥ 0.90, on test data.