Integrating Visual Learning Within a Model-based ATR System

Mark Carlotto* and Mark Nebrich, General Dynamics Mission Systems

Abstract

Automatic target recognition (ATR) systems, like human photo-interpreters, rely on a variety of visual information for detecting, classifying, and identifying manmade objects in aerial imagery. We describe the integration of a visual learning component into the Image Data Conditioner (IDC) for target/clutter and other visual classification tasks. The component is based on an implementation of a model of the visual cortex developed by Serre, Wolf, and Poggio. Visual learning in an ATR context requires the ability to recognize objects independent of location, scale, and rotation. Our method uses IDC to extract, rotate, and scale image chips at candidate target locations. A bootstrap learning method effectively extends the operation of the classifier beyond the training set and provides a measure of confidence. We show how the classifier can be used to learn other features that are difficult to compute from imagery such as target direction, and to assess the performance of the visual learning process itself.

1. Introduction

The Image Data Conditioner (IDC) is flexible multi-stage object detection and classification system that can process a wide range of imagery using application-specific processing chains (Carlotto et al 2010). The design of IDC is based on a geometrical approach to ATR that uses 3D models to control detection, segmentation, and classification (Carlotto 2015). Objects of interest (e.g., ground vehicles, ships at sea, etc.) are modeled as rectangular boxes whose dimensions are Gaussian random variables. A fast geometrical predictor estimates the size and shape of model objects in the image, which controls the detection and segmentation algorithms. Segmentation fits oriented rectangles (length x width @ pose) to candidate object regions. Detections are assigned size/shape classes by comparing measured to predicted region length and width in the pose direction. Each detection has a score based on its salience (related to the contrast of the region relative to the background) and shape (how well the segmented region fits the projected dimension of a model).

ATR systems, like human photo-interpreters, rely on location, size, shape, shadow, tone/color, texture, pattern, height/depth and site/situation/association (Colwell 1997) for detecting, classifying, and identifying manmade objects in aerial imagery. Using only geometrical models, IDC cannot distinguish targets that are similar in size/shape from clutter. In this paper we describe the integration of a visual learning component into IDC for classifying targets and clutter (Fig. 1). This new component operates on chips extracted by IDC at detection locations

* mark.carlotto@gd-ms.com
and provides additional features that are difficult to compute and must be learned from training data for the objects of interest.

![Model-based ATR and visual learning components within Image Data Conditioner (IDC)](image)

**Fig. 1 Model-based ATR and visual learning components within Image Data Conditioner (IDC)**

Our visual learning approach, which is based on an implementation of a part of the model of the visual cortex developed by Serre, Wolf, and Poggio (2004), is discussed in Section 2. Machine learning is often complicated by a variety of “nuisance” factors including rotation and scale differences between images. Section 3 describes an approach for generating chips normalized with respect to object translation, scale, and orientation that reduce the computational complexity of the learning process. The ability to extend machine learning beyond the training set is key to maintaining a high level of performance across a wide range of scenes that may encompass different environmental and sensor collection conditions. Section 4 demonstrates the effectiveness of a bootstrap learning approach that generates a large number of classifiers by sub-sampling the training set. Each classifier is analogous to a “voter” where the majority of votes decide whether a chip contains target or clutter, and the number of votes provides a measure of classification confidence. Section 5 goes on to show how this approach can be used to learn other features that are difficult to compute from imagery such as target direction.

![First three layers of standard cortex model modified for 2-class learning (e.g., target/clutter classification)](image)

**Fig. 2 First three layers of standard cortex model modified for 2-class learning (e.g., target/clutter classification)**
2. Target/Clutter Classification

The standard model of the visual cortex described by Serre, Wolf, and Poggio (2004) is an implementation of a multi-layered theory of object recognition (Serre, et al 2005). Here we use the first three layers of the model (S1, C1, and S2), with modification, as the basis of our target/clutter classifier (Fig. 2).

![Diagram of S1, C1, and S2 layers](image)

**Fig. 3 Implementation of S1 and C1 layers in the standard model for a 128x128 pixel input chip (single receptive field)**

The standard model is designed to operate over an image by decomposing the receptive field into a number of smaller patches. Our modified algorithm operates on a single 128x128 pixel patch (a chip) that is passed through a bank of 16x16 Gabor filters (S1 layer) and then on to a modified “C1” layer (Fig. 3). A set of 8x8 non-overlapping 16x16 pixel max operations are performed on each filtered image. The 4x1 max of each pixel max over is computed over 4 of the 16 scales. This is done for each orientation. The 4 scale and 4 orientation values in the 8x8 area are stored in a lexicographically-ordered \( N = 4x4x8x8 = 1024 \) element feature vector. In the standard model the next stage of processing (S2 layer) consists of groups of C1 cells that become “centers” for learning. It is at this point where our implementation differs from the standard model. This stage of processing – the stage in which learning occurs – is described in Section 4.

3. Object-Based Chipping

In overhead imagery, objects can occur at any location and orientation. Consider the detection of ships in Quickbird imagery (Fig. 4). Objects may range in size from small fishing boats to container ships. Learning to differentiate ships from clouds, waves, and other kinds of clutter is
complicated by a variety of “nuisance factors”, which include translation, rotation, and scale differences. Without explicit knowledge of these factors, machine learning must in some way account for possible variations; e.g., by training on all translations, rotations, and scales, which greatly increases computational complexity.

Fig. 4 IDC detections of possible ships in Quickbird imagery. Red boxes are projected size of best matching ship class and green boxes are segmented regions.

Fig. 5 Object-based chipping provides chips normalized with respect to translation, rotation, and scale variations.
As shown in Fig. 5 we solve this problem by using IDC to extract image chips at detection locations, using region pose to rotate the chip by minus the pose angle so the object is always oriented in the same direction, and using region length to scale the chip to be a fixed size (in pixels). We also normalize the brightness within the chip to account for differences in atmospheric transmission, scene brightness, and other factors not explicitly accounted for in the training set. In using the standard model as input to the learning algorithm the max operation in the C1 layer reduces sensitivity to residual translation errors, i.e., the target not being exactly in the middle of the chip. Fig. 6 shows examples of target and clutter chips along with a visual depiction of their corresponding C1 layer features.

![Sample target chips (top) and clutter chips (bottom) with C1 features (right of the respective chip)](image)

![S2 layer in standard model replaced by 2-class classifier based on perceptron learning within a bootstrap aggregating framework](image)
4. Bootstrap Learning Using Perceptrons

Cover’s Theorem (Cover 1965), which is important in computational learning, is based on the idea that patterns in a high-dimensional space are more likely to be linearly separable than in a low-dimensional space, provided that the space is not densely populated. If two classes (e.g., targets and clutter) are linearly separable, perceptron learning (Rosenblatt 1957) will find a linear classifier \( \mathbf{w} \) that separates the two classes

\[
    y = \begin{cases} 
        1, & \mathbf{w}^T \mathbf{x}^* \geq 0 \\
        -1, & \mathbf{w}^T \mathbf{x}^* < 0 
    \end{cases}
\]  

If the classes are not linearly separable the perceptron will make mistakes. As shown in Fig. 7 we enhance perceptron learning by means of a boosting technique known as bootstrap aggregating or “bagging” (Breiman 1996). Bagging randomly selects a subset of exemplars from the training set. The exemplars are used to train a perceptron classifier. The weights are saved and the process repeated for another randomly selected subset. This off-line process is performed M times. At run time, for each chip extracted from the image, the decision function applies the M previously saved weight vectors to the C1 features \( \mathbf{x}^* \) computed from the chip, makes a clutter/target decision, and adds votes deciding that the chip is a target if the number of decisions for targets exceed a threshold. We have found that by using a large number of classifiers created in this way a consistent level of performance can be maintained outside of the training set, even if individual perceptrons sometimes make mistakes.

Fig. 8 Normalized target (top) and clutter (bottom) chips from Quickbird image. Chips are 128x128 pixels.
Fig. 9 Effect of increasing the number of classifiers on classification performance per trial (left) and averaged over multiple trials (right)

Consider the set of target and clutter exemplars in Fig. 8. Fig. 9 shows the performance of the bagging classifier over ten trials using different numbers of classifiers. In each trial a new classifier is computed from a random sampling of the training set and tested against a different random sampling. Notice the probability of error (classifying a target as clutter, or clutter as target) decreases as the number of classifiers increases. As the number of classifiers increases the variation in performance also decreases. Evidently the greatest gain over the simple perceptron appears to occur when the number of classifiers is about an order of magnitude more than the number of exemplars. Fig. 10 illustrates this effect. Clutter patterns tend to average out leaving a symmetrical rectangular pattern of weights that tend to respond more strongly to targets than clutter.

Fig. 10 Averaged feature vectors for M=1, 10, 100, and 1000 classifiers (left to right).

5. Learning Target Direction

The IDC segmentor computes pose angle modulo 180°; i.e., it cannot differentiate two objects pointing in opposite directions. Vehicles, ships, and other kinds of manmade objects have a definite heading, or direction they are oriented in/moving toward. We can use the same method trained with different exemplars to determine if a target is oriented right (heading = pose) or left (heading = pose + 180°). The performance for M=100 classifiers computed from 30 forward (target oriented right) and 42 reverse (target oriented left) exemplars is plotted in Fig. 11. Notice the forward and reverse decision curves intersect at 63 votes out of 100 rather than zero votes as one might have expected. The bias is caused by the imbalance between the number of forward and reverse exemplars.
A way of understanding this is to compare the decision process to an election. A voter (weight vector) casts a vote yea or nay – target or clutter, or target oriented right or left – for a candidate (feature vector). Elections are decided by a majority of votes. However if voters are trained by seeing more target candidates than clutter candidates, they might tend to favor the former. This can be accommodated by a threshold $T$ that reflects target or clutter bias in the training set.

6. Measuring Confidence

Recent experiments with convolutional neural networks (CNNs) reveal that they can be fooled by unrelated images or by small imperceptible changes made to images in the training set (Nguyen et al, 2015). A nice feature of the bootstrap classifier (Fig. 11) is that the number of votes

$$ V = \frac{1}{2} \sum_{m=0}^{M-1} \text{sgn}(w^T x^*) + 1 $$

(2)

can be used to estimate classification confidence

$$ \text{Conf} = \begin{cases} 
0.5 + \frac{V - T}{M - T}, & V \geq T \\
0.5 - \frac{T - V}{M + T}, & V < T 
\end{cases} $$

(3)

Fig. 12 shows several results from the ship classification data set (Fig. 8). Chips that look like a ship (top row) have a large number of classifiers voting for ships (high ship confidence values) while those that look like clutter (bottom row) have a large number of classifiers voting for clutter and so have low ship confidence values. Notice there are 2 false alarms in the top row. Although the clutter set did contain an example of a fishing net (bottom row in Fig. 8), the second from the right detection in the top row is a fishing net. Adding additional examples of fishing nets to the clutter set tends to reduce the occurrence of this type of false alarm. The false alarm at the top right is discussed in the next section.
7. Confidence-based Learning

The performance of the visual learning component in IDC depends critically on the performance of the detector and segmentor since our implementation of the standard model assumes chips have been centered and normalized with respect to scale and rotation. Classification confidence thus depends on how well these algorithms work.

The confidence value (Eq. 3) provides a partial answer about the performance of the classifier. Confidence-based learning (Wiki 2015) measures the correctness of a learner’s knowledge and confidence in that knowledge. In an educational setting a confidence-based learning plan is designed to increase retention and minimize the effects of guessing, which skews traditional assessments – distinguishing between what individuals think they know and what they actually know. Here we wish to assess the correctness of an algorithm’s knowledge and confidence in that knowledge – to distinguish between what an algorithm knows and does not know based on a determination of whether an algorithm works or does not work.

Fig. 12 Classification confidence for ship classification data set.

Fig. 13 Two examples each of ships segmented correctly (left) and incorrectly (right)
One approach is to develop a classifier that recognizes when a segmentation is good or bad. Two sets of exemplars (Fig. 13) are used: 1) examples of correctly segmented objects (i.e., ships oriented horizontally and scaled to lie entirely within the chip, and 2) examples of incorrectly segmented targets (e.g., ships not oriented horizontally or not scaled to lie entirely within the chip (e.g., ships with wakes, or parts of ships).

![Images of segmentation examples](image)

Fig. 14 Preliminary confidence-based learning results showing target/clutter (t/c) and segmentation (seg) confidence values

Fig. 14 shows some preliminary results that indicate the performance of the detection and segmentation algorithms can evidently be learned. The second detection from the left in Fig. 14 has a very low segmentation confidence, which may account for its earlier misclassification (Fig. 13, top right). Future work will develop methods of combining and using this knowledge to better quantify the confidence and performance of the overall ATR system.

### 8. Conclusion

IDC is able to detect and classify a wide range of objects; e.g., from small fishing boats to container-sized ships. Learning to visually differentiate ships from clouds, waves, and other kinds of clutter is complicated by a variety of nuisance factors. Without explicit knowledge of these factors, machine learning must train over all possible variations, which greatly increases computational complexity. Although one could use an ATR system to simply chip out data for downstream machine learning we have shown the advantage of using information provided by IDC to preprocess (normalize) chips prior to the visual learning/classification stage. It may be possible to use this information in other ways as well.

Rather than rely on a single complex CNN classifier, IDC employs a large number of simple classifiers constructed from random subsets of the training data. Each classifier votes for one of the classes. The final classification decision is based on the total number of votes exceeding a threshold where the number of votes provides a statistically-based measure of confidence. We show the performance and trial-to-trial consistency of this classification method increases with the number of classifiers.

Beyond target/clutter, target left/right, and segmentation good/bad decisions, we are exploring additional applications of our visual learning component for differentiating objects based on specific visual cues like patterns or structural complexity (e.g., fishing, cargo, or cruise ships).
References


