ABSTRACT

Automatic Target Detection and Recognition (ATD / ATR) has been a sought after but by and large unreachable capability within the DoD for more than half of a century. These capabilities were not realized historically due to a lack of sufficient low SWaP-C computing hardware, sensors, and algorithm capabilities. Fortunately, in recent years – hardware and algorithmic advances have been made that have paved the way for all of these capabilities. However, thanks to recent advancements in machine learning algorithms (e.g., Deep Convolutional Neural Networks (DCNNs)), the ubiquity of powerful processors (e.g., Nvidia Jetson GPU solutions), and low SWAP-C UAS hardware / sensors have lead to significant advancements that place real-time UAS-based ATR within reach.

To this end, Opto-Knowledge Systems Inc (OKSI) has developed and demonstrated real-time ATR capabilities for sUAS platforms based upon Deep Convolutional Neural Networks (DCNNs). In this paper, we provide an overview of the recent ATR work conducted by OKSI. We start by laying out the design of the EO/IR sensor payload for the sUAS platform. Next, we describe the relevant intrinsic and extrinsic calibration procedures for the system. Then, we include an overview of the DCNN algorithms and describe the experiments that we designed for testing the ATR proposed ATR solution. Finally, we provide flight results showing ATR in real-time on an sUAS platform using our EO/IR sensor. Demonstrating real-time ATR on an sUAS platform provides the baseline for a number of future advancements.

1.0 Introduction

The DoD requires greater capabilities for Automated Target Detection, Recognition, and Identification (ATR/DRI – we will use ATR throughout this paper) personnel, vehicles, and other targets of interest from Unmanned Aerial Systems (UAS’s) in order to increase autonomy. ATR capabilities have been a challenging domain for The Army (and the DoD) for decades, however, recent advances in algorithms and low-SWAP-C sensors/computing hardware have led to an exponential increase in these capabilities. Effort this end, OKSI has completed some initial work which leverages advances made in ATR since the advent of deep learning in 2012 in order to significantly enhance the ATR capabilities of Class 1 / Class 2 UAS Platforms. Our solution is hardware agnostic and will be compatible with any EO or IR sensors available, respectively. In this paper, we show preliminary results for Deep Learning based ATR in detecting various objects from streaming EOIR imagery. We demonstrate the capability on multiple EO/IR imaging devices in OKSI’s possession and provide a path forward for hardening these capabilities into sUAS platforms and other application domains. The capabilities proposed are amenable to real-time implementation on-board an sUAS platform carrying a low SWAP-C NVIDIA Jetson TX2 processor.

Deep learning methods enable high-quality ATR by leveraging the base capabilities of traditional shallow neural networks (which perform well at specialized mapping tasks) and extend these networks to include multiple layers of hidden nodes. Here, each layer of hidden nodes provides a layer of abstraction. For ATR from imagery for example, the raw inputs to a Deep Network are

---

raw pixel values from an imaging sensor. Then, the first layer of the Deep Network processes all of the image pixels in order to determine edges. Next, the edges are passed down to the next layer in the Deep Network which detects the presence of corners and contours from the edge data passed from the layer above. The corners and contours information is then passed down to yet another hidden layer (or multiple hidden layers) which detect object parts (e.g., shapes, etc.) within the contours present. Frequently another layer is added over the top, which learns which specific shapes and/or combinations of shapes correspond to various objects that the network is trying to identify. Lastly, a pooling layer votes all of the output nodes together in order to provide the ATR output (i.e., whether or not an object of interest is present within the image frame).

Although the results for ATR with Deep Learning have been extremely promising (see Background section), these methods are still new and developing reliable networks is non-trivial. Two of the primary challenges associated with training Deep Nets for ATR are related. First, Deep Nets require copious amounts of accurately ‘labeled’ data sets in order to be properly trained. These data sets are expensive to collect. Secondly, along with the requirement of a great deal of labeled training data comes the requirement for massive amounts of computing power in order to train these networks. For example, some of the state-of-the-art deep networks can take a couple weeks of computing to train (though they run in real-time once they are trained). In addition to these challenges are issues of how to design the layers, feedbacks, activation functions, and how to tune the regularization parameters of each network in order to optimize the overall performance. In this paper, OKSI implements deep learning capabilities for ATR and demonstrates the capabilities on various EOIR sensors.

This paper is organized as follows. First, we provide a high-level background on deep learning for ATR. Next, we discuss the hardware used to collect imagery, the training data sets we utilized to train our ATR networks, and the computational resources used for training. We then provide example results of detecting multiple objects of interest within EOIR imagery over a number of settings. Lastly, we provide conclusions and a path forward for further developing these capabilities.

2.0 Background

True automated target recognition (ATR) has eluded the military (and others) for decades, and aided target recognition (AiTR) was settled on as the norm, as depicted in Ref. 2 and Figure 1. However, recent dramatic improvements in machine learning techniques, and the emergence of dedicated low-cost computing chips 3, has monumentally shifted the paradigm towards fully automated (rather than aided) ATR capabilities. Not only can the newer paradigm outperform the older ATR methods, but machines can also outperform humans 4 and assist in those monotonous

---


4 As of 2015, machine learning methods successfully achieved performance greater than a standard human observer in the ImageNet Large Scale Visual Recognition Competition (ILSVRC). This competition evaluates algorithms for object detection and image classification at large scale (e.g., millions of images of various classes of objects).
searches to locate targets in cluttered and busy scenery, as well as with other targeting task performance (TTP)\textsuperscript{5,6}.

![Annotated image: left two images have a sedan and pickup truck, on left and right, respectively. Right two images show an SUV, pickup truck, van, and SUV\textsuperscript{2}.](image)

**Figure 1.** Annotated image: left two images have a sedan and pickup truck, on left and right, respectively. Right two images show an SUV, pickup truck, van, and SUV\textsuperscript{2}.

### 2.1. **ATR and Deep Convolutional Neural Networks**

Automatic Target Recognition (ATR) refers to the automatic (unaided) processing of sensor data to locate and classify targets of interest. This has been an area of interest for the DoD for well over 50-years. Traditionally, limits in both algorithmic capabilities and hardware capabilities have rendered ATR methods infeasible for real-world applications. Fortunately, in recent years there has been an explosion in both low SWaP-C hardware availability as well as significant advances in ATR algorithmic capabilities (e.g., Figure 2). OKSI proposes to leverage these advancements in order to provide a hardened ATR capability for sUAS platforms.

![Performance of winners for the ImageNet Large Scale Visual Recognition Competition (ILSVRC) from 2010 – 2015.\textsuperscript{7} Deep learning methods were introduced in 2012, handily beating all other state-of-the-art methods. Since 2012, performance has been improved further to the point that Deep Learning (DL) methods perform better than an average human on these tasks.](image)

**Figure 2.** Performance of winners for the ImageNet Large Scale Visual Recognition Competition (ILSVRC) from 2010 – 2015.\textsuperscript{7} Deep learning methods were introduced in 2012, handily beating all other state-of-the-art methods. Since 2012, performance has been improved further to the point that Deep Learning (DL) methods perform better than an average human on these tasks.

Over the past several decades, there have been numerous approaches proposed to solve ATR – approximately one new primary method per decade:

- **1970’s:** Statistical pattern recognition
- **1980’s:** Template matching, advance correlation filters (e.g., optical correlators)
- **1990’s:** Combinations of template matchers and neural networks (e.g., human vision models


\textsuperscript{7} The ILSVRC database contains tens of thousands of images of different types / classes of objects.
and model-based approaches)

- 2000’s: Support Vector Machines (SVMs) and feature-based methods.

As of 2010, the state-of-the-art in automatic image detection and classification algorithms was a computationally expensive feature-based SVMs. Each of these methods had provided significant improvement for ATR over its predecessors – but it wasn’t until the advent of DL in the past decade that ATR algorithms have shown promise in providing performance on par with human observers within the real-world. Specifically, automatic image detection and classification algorithms have seen over a 600% improvement in performance over the past 5-years with the advent of DL techniques such as those OKSI employed during this effort. With the methods used, the ‘detection’ and ‘classification’ steps have been fused into a single step with methods such as DCNNs – providing a complete all-in-one solution for real-world applications.

### 3.0 Data Collection Hardware & Training

In this section we provide a high-level overview of the DCNN algorithms, computing hardware, and the imaging hardware used to enable the ATR demonstrated in this work.

#### 3.1. Deep Convolutional Neural Network (DCNN) for ATR

The task of ATR can be formulated as a combination of image classification and object localization, which have historically been extremely challenging tasks for computers to undertake. Traditional approaches utilized a bevy of manually engineered image features fed into neural networks, SVMs or a boosted ensemble of classifiers. However, despite initially promising results, improvements in classification using classical techniques, performance quickly stagnated and interest waned (e.g., Figure 3).

In recent years, a breakthrough in classification performance came in the form of Convolutional Neural Networks (CNNs), which has since then been applied with great success to a number of historically challenging computer tasks such as speech recognition, image classification, and natural language processing, surpassing human performance in many cases.

![Figure 3. Classifier trends: Interest in several types of classifiers based on their on-line search history. Interest in traditional classifiers was decreasing for nearly a decade before the advent of Deep Learning methods. This is because the community realized that traditional ATR techniques were inadequate for addressing the general ATR problem. Fortunately, as seen in 2012 –DL methods were demonstrated to provide superior performance to traditional ATR algorithms with a fraction of the effort. DL methods have been gaining popularity since 2012.](image)

In this work we exploit the powerful learning ability of deep DCNNs for target recognition by training a network to perform classification and localization of targets. Training a DCNN utilizes
a large annotated corpus of collected target data, publicly available datasets, and simulated data. In the following, we will provide an overview of DCNNs for ATR.

A neural network is a biologically-inspired computational model that consists of a connected network of artificial neurons. Each neuron has a number of inputs and, like its biological counterpart, will produce an activation (“fire”) based on a weighted sum of those inputs, plus a bias. With a sufficiently large network of these artificial neurons, the network can "learn" arbitrarily complex functions that map between the inputs and outputs given by a training dataset. However, the large number of weights and biases (parameters) can make a traditional neural network prone to overfitting, where the network "memorizes" the mappings from inputs to outputs but fails to make a meaningful generalization, causing it to fail on novel data.

\[
E = \frac{1}{2n} \sum_x ||y(x) - y'(x)||^2 \quad \frac{\partial E}{\partial y'} = (y' - y)
\]

Learning is done via an optimization method known as backpropagation, where the derivatives of the training errors of the network are propagated backwards through the network layers in order to reduce the overall training error (Figure 5). However, the problem of vanishing gradients arises as an ever-shrinking derivative is propagated through the layers of the network, causing the network to learn exponentially more slowly.
Convolutional neural networks take further inspiration from biological models of vision by simulating the receptive field of a neuron through the use of convolutional kernels, limiting its inputs to a spatially localized region of the input. The use of convolutions helps address the problem of overfitting by reducing the number of learned parameters and, in tandem with the use of rectified linear units, encourages sparsity in the network which often correlates with improved generalization. The layers of a typical CNN (Figure 6) include convolutional layers, pooling layers, activation or ReLU layers, and a final fully connected layer.

<table>
<thead>
<tr>
<th>( \begin{pmatrix} a &amp; b &amp; c \ d &amp; e &amp; f \ g &amp; h &amp; i \end{pmatrix} \ast \begin{pmatrix} 1 &amp; 2 &amp; 3 \ 4 &amp; 5 &amp; 6 \ 7 &amp; 8 &amp; 9 \end{pmatrix} )</th>
<th>( [2,2] = (i \ast 1) + (h \ast 2) + (g \ast 3) + (f \ast 4) + (e \ast 5) + (d \ast 6) + (c \ast 7) + (b \ast 8) + (a \ast 9). )</th>
</tr>
</thead>
</table>

Figure 7. A 3x3 convolutional kernel (left) is applied to a 3x3 partition of the image (right) producing a single value. The key feature of a convolutional neural network is the eponymous convolutional layer. Convolutions, also known as kernels or filters, can in the case of images be thought of as the activation of a neuron in response to only a localized region of the image (Figure 7). The use of the same set of convolutions across the image encourages discovery of more local correlations, makes those correlations translationally invariant, and reduces the number of parameters that need to be learned, which can protect against overfitting. The parameters of the convolutional kernels are learned through training of the neural network.

The pooling layer produces a spatially reduced version of its input, commonly using an operation known as max pooling which selects the largest value from a local part of the input (Figure 4). By down-sampling the input, pooling decreases the number of parameters of the network, guarding against overfitting, while also reducing the computational needs of the network.

The activation layer, a staple of neural network architectures, computes the activation function and can be seen as the digital analogue of how a biological neuron fires. While traditionally a sigmoid function (Figure 8) was used as the activation function, the problem of vanishing gradients (a consequence of backpropagation and exacerbated by the limits of computer accuracy) made rectified linear units, or ReLU, a much more attractive option.

\[ \sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \]

Figure 9. The softmax function represents a probability distribution over K classes (or labels).

The convolutional, pooling, and activation layers can be thought of as creating a high-level feature representation of the salient content of the image, mathematically capturing its "essence" or "concepts". To compute the output from this high-level representation, we allow the final layer of the network to be fully connected, as it would be in a traditional neural network (Figure 6).

The loss layer incorporates the loss function that defines how the data and its labels affect the structure of the network during training. In the traditional scheme where one needs to predict a
class or label from a finite set, the softmax loss function is used for calculating the training error (Figure 9). The loss layer is used to provide information to update the weights throughout the network during training.

In this work, we build upon DCNN methods similar to those outlined above coupled with a hybrid of existing pre-labeled training data from the community as well as some additional labeled training data collected by OKSI. To train the network, we leverage OKSI’s in-house TITAN compute server, which has four NVIDIA Titan 1080 GPUs.

3.2. Data and DCNN Training

The goal of the effort is to enable ATR on class 1 and class 2 sUAS platforms. To achieve this ATR capability we chose to leverage recent advances in DCNNs. DCNNs have demonstrated state-of-the-art performance for ATR. Although they are computationally expensive to train, they are able to run in real-time on low-SWAP-C compute hardware. In this section, we provide a high-level overview of the DL code framework, training data, and compute resources used in this effort.

In this effort, we leveraged the TensorFlow library for deep learning. The TensorFlow library is a numerical computation library that is built on the concept of data flow graphs. Individual nodes within the graph represent mathematical operations (e.g., the activation functions within a neural network), while multidimensional data arrays form the graph edges (e.g., the multidimensional arrays of pixels in our images). The primary benefit of this library is that it enables the deployment of code to multiple CPUs or GPUs in a desktop, server, or mobile device without having to write custom low-level code (Figure 10).

![Figure 10. Single machine and distributed system structure framework for TensorFlow [2].](image)

We leveraged together data from online data sets such as ImageNet and Project COCO, as well as data sets collected by OKSI on ground vehicles and aerial vehicles for training data. We set-up a DCNN framework and trained it on the OKSI compute server (Figure 13) in order to achieve reliable ATR for specific classes of objects.
Figure 11. TensorBoard graph visualization of a convolutional neural network model from TensorFlow framework provided by [2]. The TensorFlow framework provides a robust toolset for developing deep networks.

Aerial platforms capture targets from different vantage points than ground-based vehicles. However, there is significantly more labeled training data available for targets of interest from ground-based vehicles for aerial vehicles. To this end, we leveraged the concept of transfer learning, which is well-known throughout the community. Here, we first trained networks on particular classes of objects using primarily ground-based imagery. We then did a second round of training over our aerial imagery data sets (e.g., Figure 12). This enabled us to achieve high-quality ATR performance on targets from aerial platforms.

![Example images captured during OKSI's recent sUAS data collection for a DCNN ATR proof-of-concept demo.](image)

The first step of the ATR pipeline determined whether or not a target is present within an image of interest and if so, which region the target resided in (approximately). The next step in the pipeline worked to generate a tightly bounded silhouette cut-out of the target. The targets can then be pulled out of the imagery and overlaid with color to highlight the targets for human observers (e.g., for decision making).
4.0 ATR on sUAS Data

For this effort, we demonstrated the ATR capability for a few key-classes of objects (though we trained for others). Specifically, cars, trucks, and people. These generic classes sufficiently demonstrated the concept of DL for ATR in a variety of settings. Generally speaking, to train a CNN we require a few hundred thousand images on a particular high-level object class of interest in order to achieve high-quality ATR with CNNs.

![Figure 13](image1.png)

Figure 13. OKSI utilized our in-house sUAS platform near Thousand Oaks, CA to collect a simple DCNN test dataset (left). The DCNN was trained on OKSI's supercomputer (right).

![Figure 14](image2.png)

Figure 14. Examples of the ATR output recognizing a car (left), a truck when the car is present (center), and a car when the truck is present (right). This algorithm runs at 30 Hz on a standard processor after training.

![Figure 15](image3.png)

Figure 15. Localization of target (car) within a scene from different vantage points.

After training a DCNN, we processed some example frames captured by OKSI’s sUAS platform of a car and truck within a rural environment to provide a baseline demonstration of the ability to first detect the presence of a target of interest within a scene (Figure 14) and to then localize the object within the scene (Figure 15-Figure 16). These were visible images captured on-board one
of OKSI’s sUAS platforms. Through this effort, we were able to demonstrate the successful detection of and distinction between a car, truck, and people. The initial CNN layer is trained to detect the ‘presence’ of objects of interest within a scene and to provide an estimate of where the object of interest lies within the scene via a heat map (e.g., Figure 14). Although this is useful, for military applications – having a clear and precise knowledge of the location of the target of interest is key to aide human operators with decision making. To this end, we implemented a second phase of the pipeline which employs an additional deep network to provide a tight mask bound around the target after the general location is provided by the first network. It is these tightly bounded target detection frames that we provide as demonstrations for the rest of the paper.

![Figure 16. Localized detection of car (left), people (left), and truck (right) within sUAS imagery.](image)

![Figure 17. Example of reliable ATR using OKSI’s algorithms for detecting a truck. As can be seen, these algorithms are able to reliably detect a truck even when it is partially occluded (left).](image)

![Figure 18. Automatic detection of partially occluded car (left) and a car in the background (right).](image)
As can be seen in Figure 14-Figure 16, the proposed ATR algorithms are able to reliably detect objects such as a truck, car, and humans within visible imagery in a controlled setting. The next step that we took was to train a DCNN for thermal IR images to demonstrate the ability of our algorithms to enable ATR within day/night settings. The key challenge with training for multiple modalities is developing a classifier that is robust to contrast changes in terms of both lighting throughout the day/night, as well as contrast changes due to how different modalities of imagers view the same scene.

Once we trained a DCNN network for detecting key objects of interest within MWIR imagery, we conducted some tests on ground imagery captured near OKSI. In these experiments, we were able to demonstrate the ability of the trained network to automatically detect trucks, cars, and people under different circumstances (with and without occlusion). Examples of detection of a truck can be observed in Figure 17.

![Figure 19](image1.png)  
Figure 19. Example automated detection of difficult to see car in the background behind trees (left).

![Figure 20](image2.png)  
Figure 20. sUAS platforms capable of real-time ATR at OKSI (left) and example image from real-time ATR in-flight on OKSI drones (right).

As a part of this effort, OKSI has been testing and training several different ATR algorithms for detection of vehicles and humans in various environment. We are currently working to significantly expand the capabilities of the networks that we have trained. To achieve this, we are training networks on millions of images collected by the community, the DoD, and OKSI over the years. We anticipate being able to readily detect and discern military vehicles, weapons, and other
objects within imagery using these algorithms in the near future as we train them on large representative datasets.

5.0 Conclusions and Future Work

In this effort, we leveraged state-of-the-art techniques in machine learning (particularly Deep Convolutional Neural Networks) to enable baseline ATR for a range of target taxonomies. Specifically, we demonstrated the detection of people and vehicles in both visible and thermal IR data sets over various terrains. Our current work focuses on domains / settings where the target occupies a sufficiently large portion of the image field-of-view (e.g., the target is on the order of tens of pixels in size). Although these results are promising, there is still a great deal of work to be done in order to transition these capabilities into an operational system. These enhancements include training these networks to cover DoD-specific target classes, and enhancing the networks themselves to optimize the performance (e.g., to obtain a high-detection rate with a low false-alarm rate for standard ATR).

Acknowledgements: This work was funded by OKSI IR&D.

References