ERDC/GRL TN-22-1
June 2022

Adversarial Artificial Intelligence: Implications for Military Operations

by Harland Yu and Saransh Chopra

BACKGROUND: Artificial intelligence and machine learning algorithms are at the forefront of current research to help military analysts deal with triaging ever larger amounts of data from deployed sensors. These automated approaches will become increasingly embedded into the military decision making process, which makes it crucial to understand how these algorithms generate outputs and how sensitive they are to perturbations during training or classification. In other words, humans must have a 'theory of mind' for these sets of approaches in order to begin to trust them enough to make life or death decisions. Research in this area is known as adversarial examples for artificial intelligence / machine learning.

Previous works in this domain focused on degrading classification performance with respect to added noise to new data. Some of these works achieved notable results on image data by subtly increasing noise, such that the image appeared unaltered to the human eye, but significantly impacted performance (Athalye et al. 2017). Povolny and Trivedi (2020) achieved similar results, but made a small visually obvious change to induce a degradation in performance. One notable work examined the effects of an increase in physical scale of the sensed environment (such as the large areas recorded for remote sensing platforms) on adversarial perturbations (Czaja et al. 2018).

This technical note (TN) describes an initial foray into understanding how physical changes to the appearance of military vehicles resulted in performance degradation for a convolutional neural network (CNN). The military vehicles chosen were the M2 Bradley Infantry Fighting Vehicle and the M1064 Mortar Carrier. As stand-ins for the actual vehicle, plastic scale models were used, each a 1/35 scale replica. The results of this research have yielded a curated training and test data set of images related to the M2 and M1064, trained models based on a combined ResNet / Inception implementation from the Keras project, and adversarial examples mocked up using the scale models with images taken by a smartphone.

DATA AND METHODS: The research presented in this TN focuses on perturbations and images within the visible spectrum. This sensing approach will probably one of an ensemble approach that would be used to perform vehicle classification. Other sensor modalities such as infrared could also have been examined, but would have been difficult to obtain in the limited time for an initial study. Additionally, many of the techniques and approaches described in previous works were demonstrated on these types of images.

Sample images for training, tests, and evaluation data sets were obtained through a Google image search using the queries 'm2 bradley' and 'm1064 mortar carrier.' The image URLs were then copied and used as arguments to the *wget* command, which downloaded all images into a directory. Non-relevant images were culled from the directory and the remainder were manually processed

using the LabelImg open-source software package.[1] LabelImg allows the user to draw bounding boxes around subjects of interest in an image file and to generate a separate XML file that saves the annotations. Information included in the annotations file includes the position of the bounding box and the associated class label for the subject in the bounding box. An example of processing an image using LabelImg is shown in Figure 1.

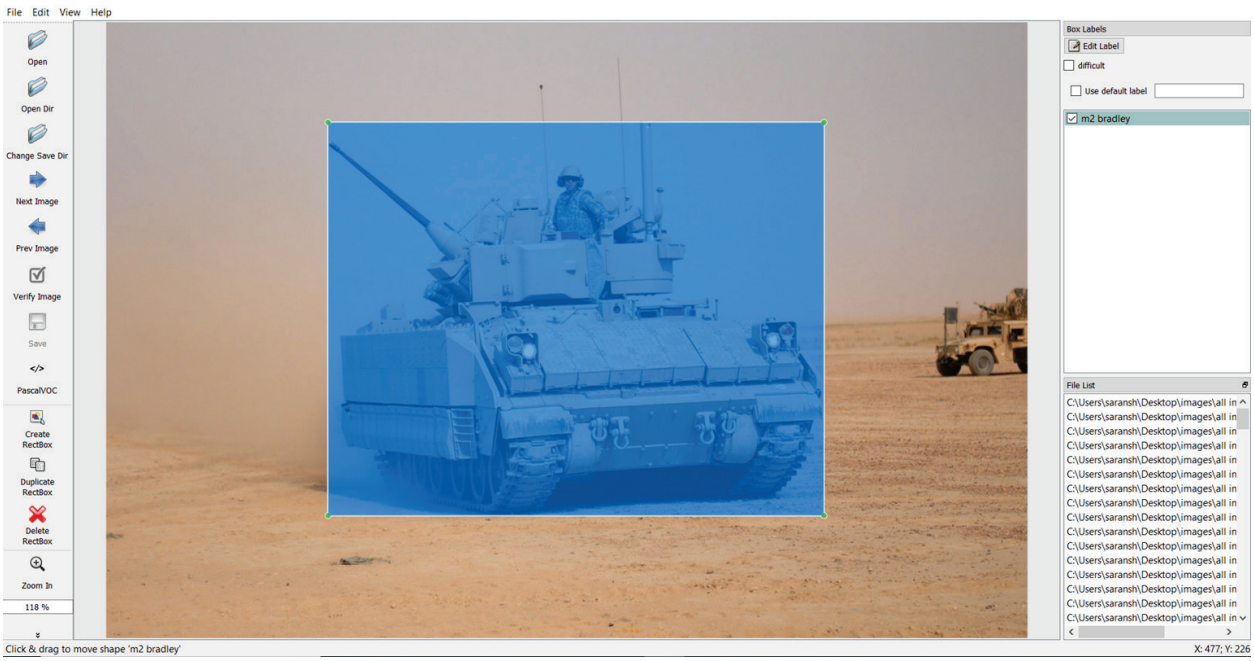

Figure 1. LabelImg software.

Processing the image data in this fashion affords flexibility as the original image remains untouched. Accuracy during model training, test, and evaluation may increase since the bounding box can be used to effectively crop the image in memory. This processed data set was then split into separate training, test, and evaluation data sets. A screenshot showing a sample of the training data set is shown in Figure 2.

[1] Mahmoudian and tzutalin. 2020. "A graphical image annotation tool." *A graphical image annotation tool.* 5. https://github.com/tzutalin/labelImg.

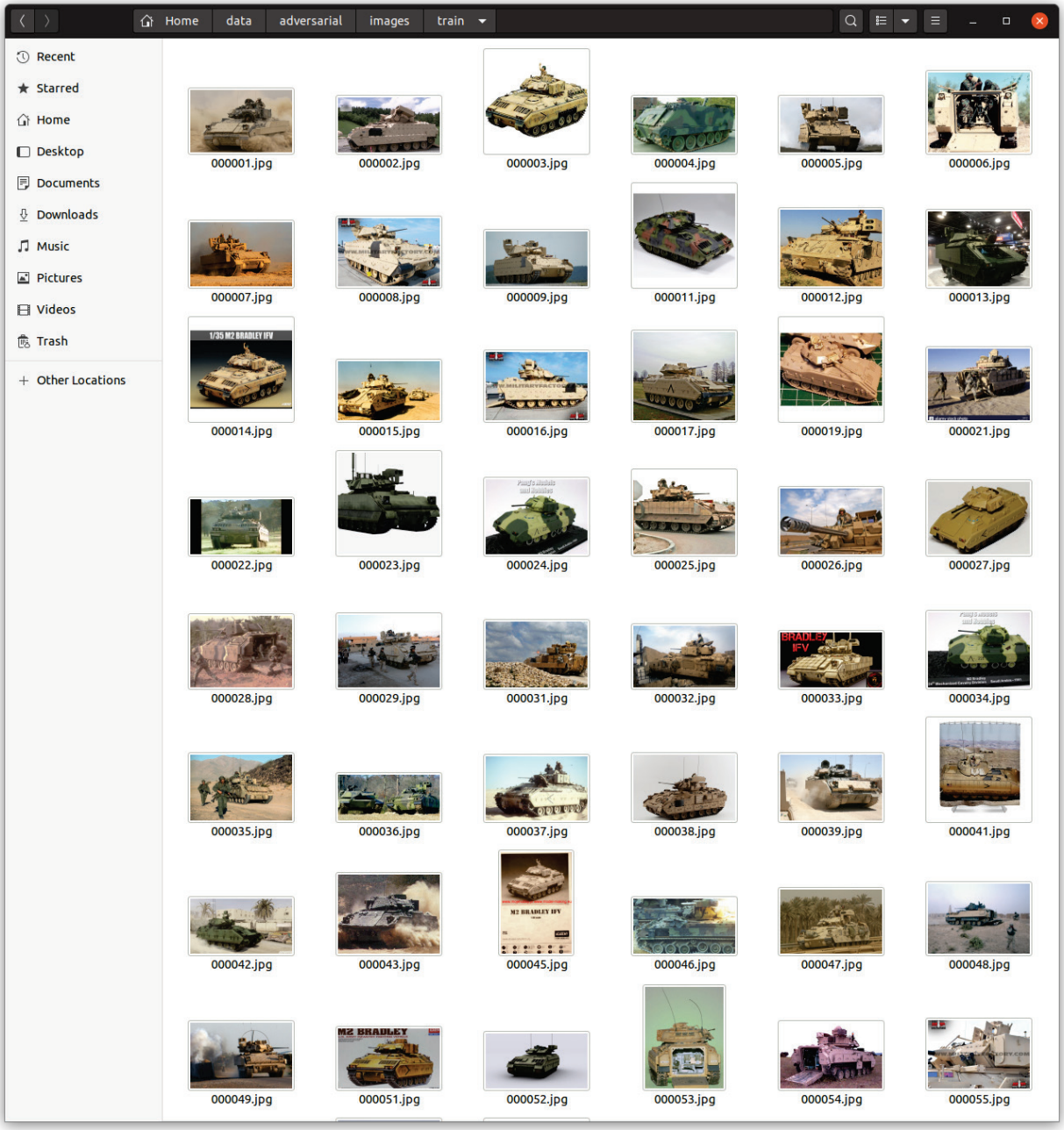

Figure 2. Sample of images in training data set.

Adversarial examples were obtained by modifying the appearance of the scale models using colored plastic tape. Three colors were used – yellow, red, and white – and were applied in various patterns across the entire body. A smartphone was used to photograph the modified models in various orientations and settings. The resulting collection of images were also processed using LabelImg to identify the areas of interest in a particular image. Additionally, filters were applied to the images that added noise or blur in order to further gauge the effect of these perturbations on performance. Instances of these adversarial examples are shown in Figure 3.

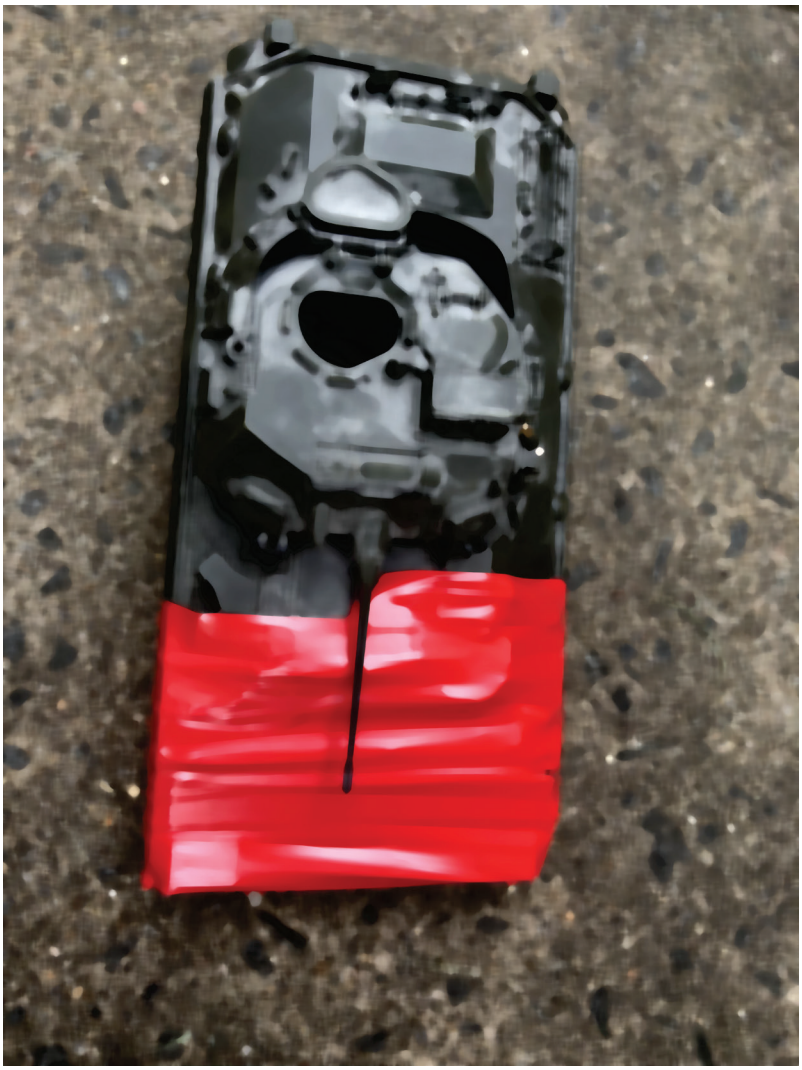

Figure 3. Adversarial example.

Convolutional neural networks were selected for evaluation on image classification due to their robustness on this type of task (Raghu and Schmidt 2020). The basic theory behind this approach is to break up the input image into tiles. These tiles can then be grouped together in various permutations (e.g. 3 x 3 or 2 x 2) in different layers of the network. As the outputs of different layers are combined, the network eventually learns which tiles or groupings of tiles are most predictive to a particular class of images. The experiments used two different CNNs. One was an implementation of Faster R-CNN ResNet 101 (FRCNN) from the TensorFlow Model Zoo[1] pre-trained on the Common Objects in Context (COCO) data set[2]. Note that the COCO data set was created to test a similar task to the one outlined in this TN – custom object detection although it does not appear to contain images of military vehicles. The model was trained further on the training data set in an attempt to understand the effects of transfer learning on training time and accuracy. The second model was an implementation that combined two CNNs – ResNet and

[1] *https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md/*.
[2] *http://arxiv.org/abs/1405.0312*

Inception V2 (RIV2) from the Keras project (Lee 2020). This model was trained solely on this training data.

TensorFlow was chosen as the underlying architecture to train, test, and evaluate the models. The main advantage of TensorFlow is its extensive support in terms of documentation, ease of installation, and available model implementations. Additionally, TensorFlow is able to take advantage of graphics processing units (GPUs) in order to dramatically reduce training time, which is useful when training the RIV2 model.

RESULTS: Training was stopped at epoch 2,629 for FRCNN to graph loss function 'noise.' And, training was stopped for RIV2 after 49 epochs once the loss function started to increase after 8 epochs – this is a user-defined setting that is useful for preventing overfitting of the model to training data. A graph of the loss versus training epoch for FRCNN and the training results for RIV2 are shown in Figures 4 and 5, respectively.

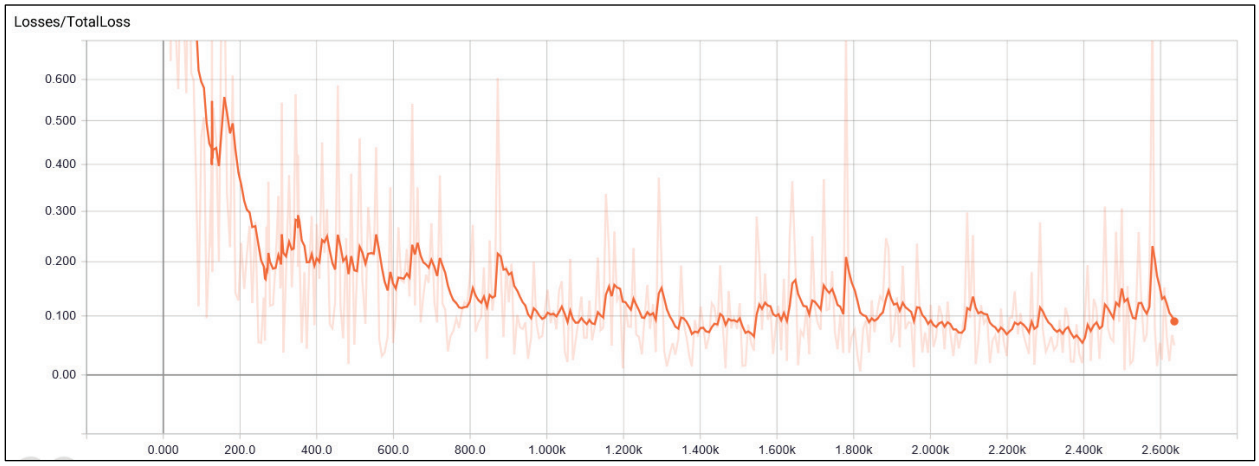

Figure 4. Loss versus training epoch, FRCNN.

Checkpoint files were created once training was complete in order to save the state of the models. These are stored in the Hierarchical Data Format version 5 (HDF5) format using the h5py Python package. The checkpoint files can then be used at any time for test or evaluation.

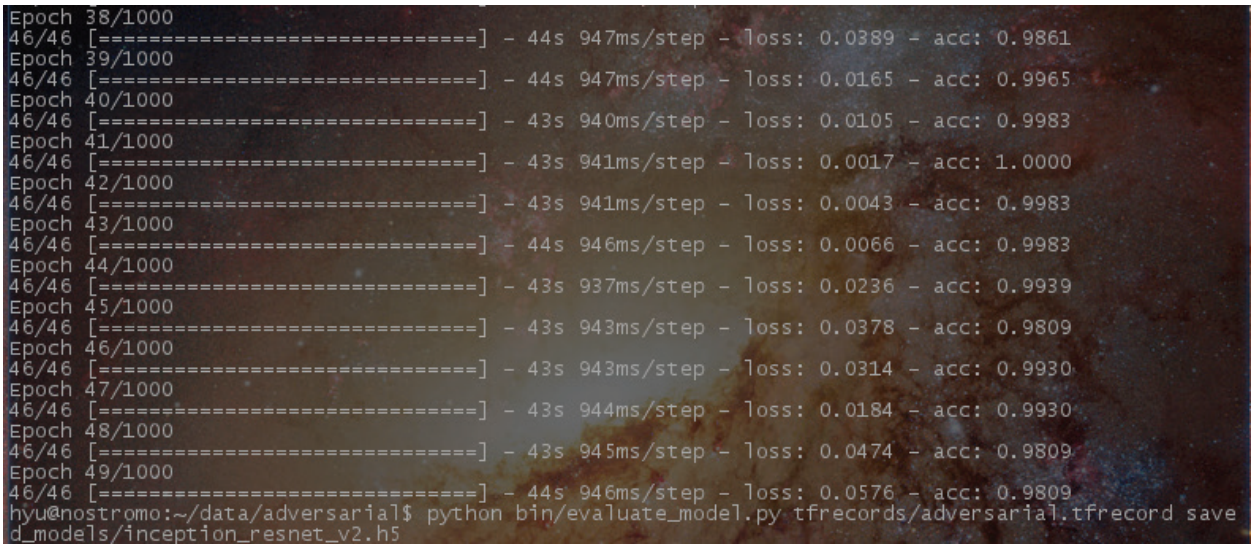

Figure 5. Training results RIV2.

Tested on the adversarial data set, both the FRCNN and RIV2 models exhibited significantly degraded performance. For FRCNN, total accuracy was reduced to 36% whereas, RIV2 achieved ~50%. A breakdown of FRCNN performance is shown in Figure 6 and RIV2 performance is shown in Figure 7. Note that for the chart in Figure 6, the 'mislabeled' category means the model supplied the wrong vehicle label (either M2 or M1064). The 'missed' category means the model did not classify the image as 'M2' or 'M1064' at all. This was because the model was trained on the COCO dataset for a more general classification task. Characterizing RIV2 is much more straightforward since only two labels were possible, either 'M2' or 'M1064.'

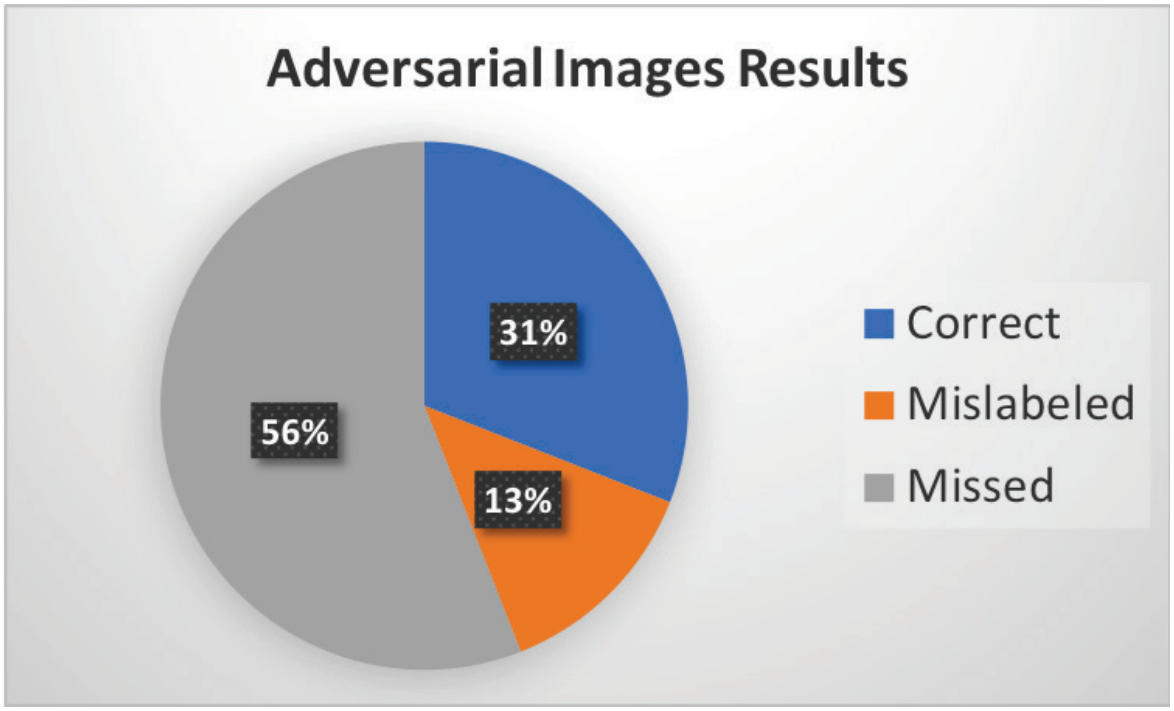

Figure 6. Classification performance breakdown, FRCNN.

```
2018-11-20 15:46:50.550829: I tensorflow/core/common_runtime/gpu/gpu_device.cc:1097] Created Tens
orFlow device (/job:localhost/replica:0/task:0/device:GPU:0 with 10809 MB memory) -> physical GPU
 (device: 0, name: TITAN V, pci bus id: 0000:01:00.0, compute capability: 7.0)
Evaluating model saved_models/inception_resnet_v2.h5, num_records: 84, num_steps: 4
4/4 [==============================] - 9s 2s/step
Evaluate results: [4.8745211362838745, 0.5033333376049995]
```

Figure 7. Evaluation results, RIV2.

The performance of RIV2 was also examined further to see which images it misclassified within the adversarial data set. A sample of this output is shown in Figure 8.

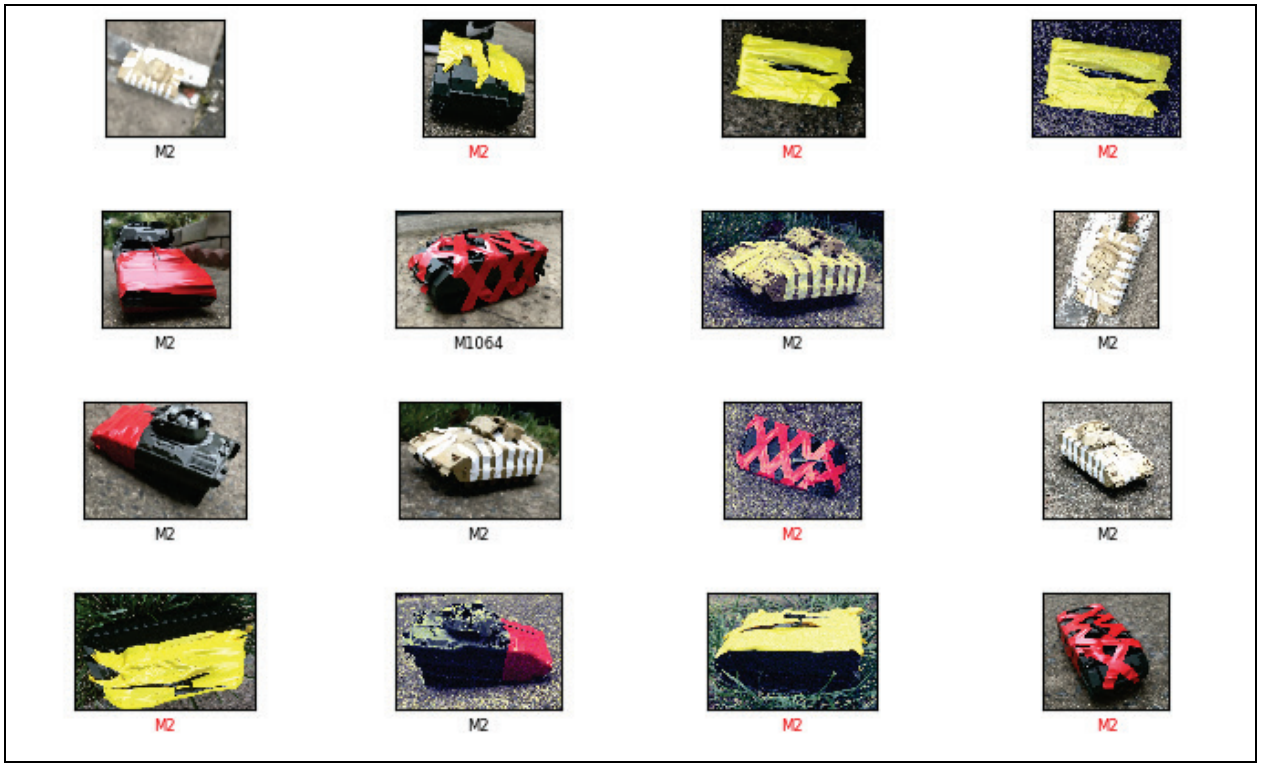

Figure 8. Sample classification results, RIV2.

What was surprising is how robust RIV2 could be to some perturbations. Note the images of the desert camouflage M2 Bradley with scattered white tape and the woodland camouflage M2 Bradley with red tape across the front. However, one particular camera angle for the desert camouflage adversarial example class managed to force the classifier to miss every single time, which can be seen in Figure 9. Overall accuracy when attempting to classify the adversarial examples using RIV2 was ~50%.

Figure 9. Misclassification camera perspective.

SUMMARY AND CONCLUSIONS: Convolutional neural networks can be quite robust in limited instances to overt perturbations to a target. However, overall performance can degrade to the point where random guessing would be just as effective in image classification tasks. Both FRCNN and RIV2 reached this point when evaluated against the full adversarial example data set.

Future work should include attempting to analyze the models' activation atlases (Carter et al. 2019), which can give clues as to which features are the most predictive. Other approaches within the domain of explainable artificial intelligence may also prove useful in this regard. Additionally, work determining the minimal number of physical changes to the vehicle's appearance required for misclassification to occur should also be explored.

TensorFlow proved to be a stable and performant system to train, test, and evaluate deep learning models. All models described in this TN used the 1.x API, but a newer API has already been released. Future work should attempt to use the newer 2.x API wherever possible, although the algorithms may be superseded by more current and robust versions by then.

Clearly, the use of artificial intelligence in military applications shows immense promise, but can also be laced with unintended pitfalls. It remains to be seen whether these algorithms can be trusted to help guide decision makers without careful consideration to hidden biases or blind spots created by training, test, or evaluation data and processes.

REFERENCES

Athalye, A., L. Engstrom, A. Ilyas, and K. Kwok. 2017. "Synthesizing robust adversarial examples." *CoRR* abs/1707.07397. http://arxiv.org/abs/1707.07397.

Carter, S., Z. Armstrong, L. Schubert, I. Johnson, and C. Olah. 2019. "Activation Atlas." *Distill* (Distill Working Group) 4. doi:10.23915/distill.00015.

Czaja, W., N. Fendley, M. J. Pekala, C. Ratto, and I.-J. Wang. 2018. "Adversarial Examples in Remote Sensing." *CoRR* abs/1805.10997. http://arxiv.org/abs/1805.10997.

Lee, T. H. 2020. "ResNet Inception V2." *ResNet Inception V2.* 5. https://github.com/keras-team/keras-applications/blob/master/keras_applications/inception_resnet_v2.py.

Lin, T-Y., M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. 2014. "Microsoft COCO: Common Objects in Context." *CoRR* abs/1405.0312. http://arxiv.org/abs/1405.0312.

Povolny, S., and S. Trivedi. 2020. "Model Hacking ADAS to Pave Safer Roads for Autonomous Vehicles." *Model Hacking ADAS to Pave Safer Roads for Autonomous Vehicles.* 2. https://www.mcafee.com/blogs/other-blogs/mcafee-labs/model-hacking-adas-to-pave-safer-roads-for-autonomous-vehicles/.

Raghu, M., and E. Schmidt. 2020. "A Survey of Deep Learning for Scientific Discovery." *A Survey of Deep Learning for Scientific Discovery*. http:// arxiv.org