Toward Automatic Threat Recognition for Airport X-ray Baggage Screening with Deep Convolutional Object Detection

Kevin J Liang[1]*, John B. Sigman[1], Gregory P. Spell[1], Dan Strellis[2],
William Chang[2], Felix Liu[2], Tejas Mehta[2], and Lawrence Carin[1]
[1]Duke University [2]Rapiscan Systems

Abstract

For the safety of the traveling public, the Transportation Security Administration (TSA) operates security checkpoints at airports in the United States, seeking to keep dangerous items off airplanes. At these checkpoints, the TSA employs a fleet of X-ray scanners, such as the Rapiscan 620DV, so Transportation Security Officers (TSOs) can inspect the contents of carry-on possessions. However, identifying and locating all potential threats can be a challenging task. As a result, the TSA has taken a recent interest in deep learning-based automated detection algorithms that can assist TSOs. In a collaboration funded by the TSA, we collected a sizable new dataset of X-ray scans with a diverse set of threats in a wide array of contexts, trained several deep convolutional object detection models, and integrated such models into the Rapiscan 620DV, resulting in functional prototypes capable of operating in real time. We show performance of our models on held-out evaluation sets, analyze several design parameters, and demonstrate the potential of such systems for automated detection of threats that can be found in airports.

1. Introduction

The Transportation Security Administration (TSA) oversees the safety of the traveling public in the United States of America. One of the most visible functions of the TSA is security screening of travelers and their personal belongings for potential threats. Handsearching each passenger's bag would be both time-consuming and intrusive, so X-ray scanner systems such as the Rapiscan 620DV are deployed to remotely provide an interior view of baggage contents. Many real threats are captured nationwide: in 2018, for example, 4239 firearms were found in carry-on bags, and more than 80% of these were loaded [36]. These numbers have steadily grown in recent years as air traffic has continued to increase nationally. The capability of finding these objects effectively is an important concern for national security.

Currently, the detection of prohibited items relies on Transportation Security Officers (TSOs) to visually pick out these items from displayed image scans. This is challenging for several reasons. First, the set of prohibited items that TSOs must identify is quite diverse: firearms; sharp instruments; blunt weapons; and liquids, aerosols, and gels (LAGs) with volumes exceeding the TSA-established thresholds all pose security concerns. Second, the majority of scans are benign, yet TSOs must remain alert for long periods of time. Third, because X-ray scans are transmission images, the contents of a bag appear stacked on top of each other into a single, often cluttered scene, which can render identification of individual items difficult. The Rapiscan 620DV provides dual perpendicular views to ameliorate this problem, but depending on the orientations, views can still be non-informative. Finally, given the need to maintain passenger throughput, evaluation of a particular scan should not take excessively long.

For the aforementioned reasons, an automatic threat detection algorithm to aid human operators in locating prohibited items would be useful for the TSA, especially if it can be readily integrated into the existing fleet of deployed scanners. Fundamentally, the TSOs both localize and identify dangerous items in an image, which are the same objectives of *object detection* [13, 12, 30, 24, 9, 17]. Object detection has long been considered a challenging task for computers, but advances in deep learning [14] in recent years have resulted in enormous progress. Specifically, Convolutional Neural Networks (CNNs) [21] have proven extremely useful at extracting learned features for a wide variety of computer vision tasks, including object detection. As a result, the TSA is interested in assessing the feasibility of deploying algorithms that can automatically highlight objects of interest to TSOs [1].

Most deep learning methods require a large training dataset of labeled examples to achieve good performance [33]; for object detection, this means data comprising both images and bounding boxes with class labels.

*Correspondence to: `kevin.liang@duke.edu`

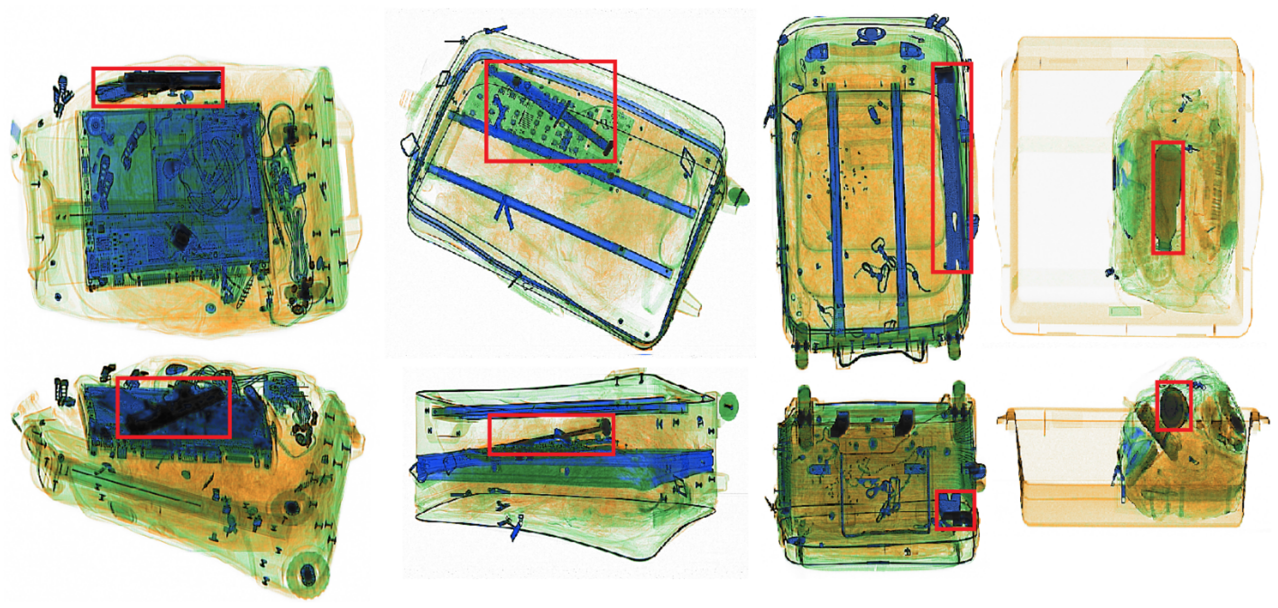


Figure 1: Example scans of bags in false color containing a firearm (handgun), sharp (knife), blunt (crow bar), and LAG (bottle of liquid), from the left to right, with (top row) top view and (bottom row) side view shown. Ground truth locations of threats in each image are bounded with a red box. Scans produced by a laboratory prototype not in TSA configuration.

While many such datasets exist for Red-Green-Blue (RGB) natural scenes (*e.g.* [11, 23, 8]), none contain threats in X-ray luggage, and so a sizable data collection effort was necessary for this endeavor. We assembled a large variety of cluttered bags (*e.g.* clothing, electronics, etc.) with hidden threats (firearms, sharps, blunts, LAGs), and scanned these with the Rapiscan 620DV. Each threat in the scans was then annotated with a tight bounding box and labeled according to class. This dataset was then used for training and evaluating object detection models.

In this work, we present the results of a research effort in collaboration with the TSA to develop a deep learning-based automated threat detection system. We first describe the Rapiscan 620DV scanner and the data collection process. We then introduce the deep learning algorithms we used to perform object detection and how we integrated them into a Rapiscan 620DV prototype, for live testing. Finally, we present experimental results on a number of models we tested on the collected data. The resulting prototype system has shown great promise, and technology like this may one day be deployed by the TSA to airports nationally.

2. Data Collection

2.1. Rapiscan 620DV X-Ray Scanning System

The Rapiscan 620DV X-ray screening system is designed for aviation and high-security applications. It comprises a tunnel 640 mm wide and 430 mm high, equipped with a 160 kV / 1 mA X-ray source that achieves a steel penetration of 33mm and a wire resolution of about 80 micrometers (40 American Wire Gauge). The scanner produces two views through the near-orthogonal orientation of the fan-shaped beams from the X-ray sources. These projections generate a horizontal and vertical view of the object under inspection, both of which can be used to identify the contents of a bag. X-ray detectors collect both high and low X-ray energy data, which allows for material discrimination. Examples are shown in Figure 1.

While it is possible to use the high and low energy image scans as direct inputs to our model, we instead choose to use the pre-processed RGB coloration typically shown to human TSOs. This coloring uses the relationship between the linear attenuation coefficient and photon energy to estimate effective atomic number (Z), transforming the image into one where material properties can be more readily inferred: for example, organic materials tend to have low Z, while metallic materials tend to have higher Z. According to Rapiscan's proprietary coloring scheme, metallic objects are colored blue, organic materials are tinted orange, and materials with effective Z (Z_{eff}) between these two are shaded green. Using this false coloring as our input achieves two objectives: (i) encoding of additional human knowledge of material properties, which are highly informative for threat detection (firearms, sharps, and blunts, for example, often contain metallic components) and (ii) aligning the image input color distribution closer to the pre-trained weights, which were trained on RGB natural scenes.

Threat Type	Total Threats	Total Images
Blunts	10	3366
Firearms	43 (assembled) + 19 (disassembled)	3480
LAGs	70	3456
Sharps	40	3484

Table 1: Total number of unique threat items and number of images collected for each threat.

2.2. Scan Collection and Annotation

Baggage scans were collected at various sites, occurring over multiple collection events. This data collection targeted several of the TSA's designated threat categories: firearms (*e.g.* pistols), sharps (*e.g.* knives), blunts (*e.g.* hammers), and LAGs (*e.g.* liquid-filled bottles). A diverse set of unique items from each class were selected to provide coverage for each threat type; for example, the firearms set included both assembled and disassembled guns. To simulate the diversity of real-world traffic, a variety of host bags was used, including roller, laptop, and duffel bags. Each was filled with diverse assortments of benign items, such as clothing, shoes, electronics, hygiene products, and paper products. Threats were added to each host bag in different locations and orientations, as well as with imaginative concealments, to simulate the actions of potentially malicious actors. Under the assumption that threat objects are typically rare, most bags contained only one threat, as in the examples shown in Figure 1.

Given the time-consuming nature of assembling bags for scanning, a single bag was used to host different unique threats for multiple scans, with a minor exchanging of benign clutter between insertions. Each bag was also scanned in several different poses (*e.g.* flipped or rotated). These strategies allow for more efficient collection of more scans and encourage our models to learn invariance to exact positioning within the tunnel. Total number of threats scanned are summarized in Table 1.

After the scans were collected, each image was hand-annotated by human labelers, where each label consisted of both the threat class-type, as well as the coordinates of the bounding box. Each box was specified to be as tight as possible in each view, while still containing the full object; in the case of objects like sharps and blunts, this definition included the handle, for instances in which there was one. In total, the entire data collection effort of assembling, scanning, and labeling bags took over 400 worker hours.

3. Methods

3.1. Convolutional Neural Networks

The advent of deep convolutional neural networks (CNNs) [21] has resulted in a quantum leap in the field of computer vision. Across virtually all computer vision tasks, the incorporation of CNNs into model designs has resulted in significant performance gains; consequently, CNNs play a significant role in almost every recent computer vision algorithm. Unlike classical methods that rely upon carefully selected, human-engineered features, machine learning (and deep learning) methods learn these features from the data itself. CNNs in particular are designed to learn hierarchical representations [38], resulting in a feature extractor that produces highly informative, abstract encodings that can be used for downstream tasks, such as classification [19]. Additionally, the learned visual features are highly transferable: for example, CNN weights learned for the classification task of ImageNet [10] can serve as a good initialization for other datasets or even other related computer vision tasks [37, 28, 13]. Doing so can considerably reduce the number of training examples needed for the desired task. In the setting of automatic threat detection at TSA checkpoints, this is especially significant, as we must assemble, scan, and label each training sample ourselves; pre-trained networks allow us to significantly cut down man-hours and costs.

CNN Architecture	Top-1 Accuracy	Number of parameters
Inception V2 [18]	73.9	10.2 M
ResNet-101 [15]	77.0	42.6 M
ResNet-152 [15]	77.8	58.1 M
Inception ResNet V2 [34]	80.4	54.3 M

Table 2: ImageNet classification accuracy and number of parameters for each of the CNNs architectures considered in our experiments. Adapted from [17].

There are several design considerations for CNNs. Most obvious is model performance: how good are the features the CNN extracts for the downstream task? In general, there is a positive correlation between the number of CNN layers (depth) and parameters with overall performance [32, 15], though architectural choices can play a significant role as well [39]. However, finite hardware memory and processing time limit model size. We consider several popular CNN architectures in our experiments, summarized in Table 2.

3.2. Object Detection

Localizing and classifying objects in a scene is a canonical research area in computer vision. In this context, localization refers to the production of a *bounding box* which is as tight as possible while still containing the entire object, while classification is the identification of which of a predetermined set of classes the object belongs to. Formally, given an image X, the goal of object detection is to predict the class c_i of each object indexed by i, as well as the center and dimensions (x_i, y_i, w_i, h_i) of a bounding box.

Modern object detectors are almost exclusively built

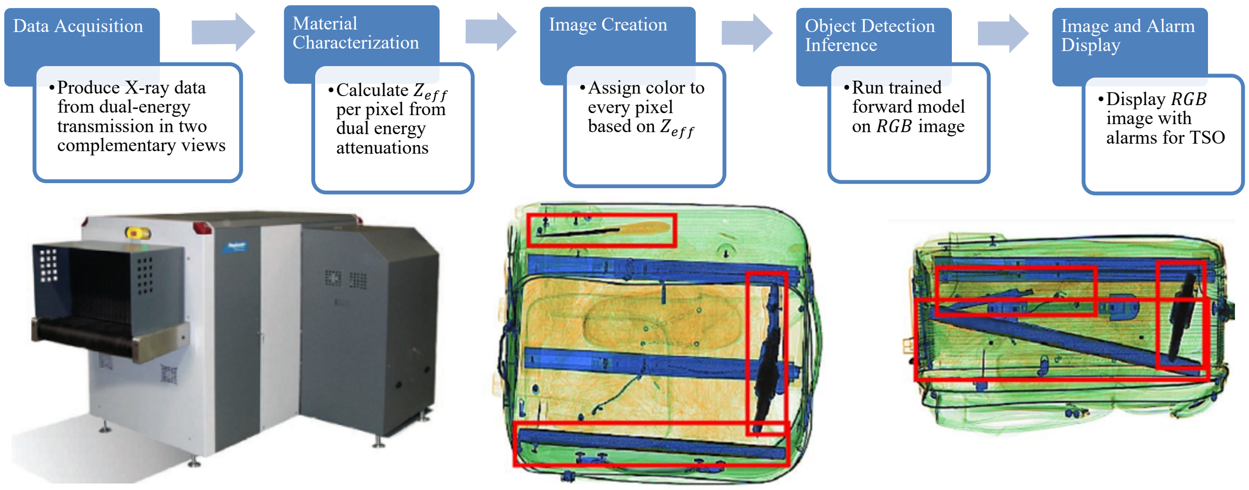

Data Acquisition
• Produce X-ray data from dual-energy transmission in two complementary views
Material Characterization
• Calculate Z_{eff} per pixel from dual energy attenuations
Image Creation
• Assign color to every pixel based on Z_{eff}
Object Detection Inference
• Run trained forward model on RGB image
Image and Alarm Display
• Display RGB image with alarms for TSO

Figure 2: Diagram of the prototype Rapiscan 620DV X-ray screening system with threat recognition capability. Dual-energy X-Ray information yields false-color images of two views, which are fed to a trained deep convolutional object detector. Detections above threshold are displayed for the user. Scans produced by a laboratory prototype not in TSA configuration.

upon CNN backbones. The specific CNN architecture used is often readily interchangeable, with the choice of CNN depending on the trade-off between accuracy with speed and memory. How predictions are made from the features extracted by the CNN can vary, and various object detection *meta-architectures* [17] have been recently proposed, of which we highlight two notable ones here.

Faster R-CNN: Faster R-CNN [30] makes predictions in a two-stage process. In first stage, called the Region Proposal Network (RPN), a set of reference boxes of various sizes and dimensions (termed *anchor boxes*) are tiled over the entire image. Using features extracted by a CNN, the RPN assigns an "objectness" score to each anchor based on how much it overlaps with a ground-truth object, as well as a proposal of how each anchor box should be adjusted to better bound the object. The N_p box proposals with highest objectness scores are then passed to the second stage, where N_p is a hyperparameter controlling the number of proposals. In the second stage, a classifier and box refinement regressor yield final output predictions. Non-maximal suppression reduces duplicate detections.

Single Shot MultiBox Detector (SSD): Unlike Faster R-CNN, which performs classification and bounding box regression twice, Single-stage detectors like SSD [24] combine both stages. This eliminates the proposal stage to directly predict both classes and bounding boxes at once. This reduction tends to make the network much faster, though sometimes at the cost of accuracy.

3.3. Evaluation

The two-part nature of the object detection task–localization and classification–requires evaluation metrics that assess both aspects of detections. The quality of an algorithm-produced predicted box (B_p) with a ground-truth bounding box (B_{gt}) is formalized as the Intersection over Union (IoU) $= \text{area}(B_p \cap B_{gt})/\text{area}(B_p \cup B_{gt})$.

A true positive (T_p), false positive (F_p), and false negative (F_n) are defined in terms of the IoU of a predicted box with a ground-truth box, as well as the class prediction. A true positive proposal is a correctly classified box that has an IoU above a set threshold (*e.g.* 0.5), a false positive proposal either misclassifies an object or does not achieve a sufficiently high IoU, and a false negative is a ground-truth object that was not properly bounded (with respect to IoU) and correctly classified.

At a particular IoU threshold, the *precision* and *recall* of the model may be computed as the proportion of proposed bounding boxes that are correct and the proportion of ground truth objects that are correctly detected, respectively. These quantities are: $\text{Precision} = T_p/(T_p + F_p)$, $\text{Recall} = T_p/(T_p + F_n)$. Precision-recall (PR) curves are constructed by plotting both quantities over a range of operating point thresholds. We present these curves in Section 5 to provide a sense for model performance. Additionally, we may quantitatively summarize model performance through mean Average Precision (mAP). Average Precision (AP) is the area-under-the-curve (AUC) of the PR curve for a single class, and mAP is the mean of the APs across all classes.

3.4. Rapiscan 620DV Integration

In order to take a concrete step towards the TSA's goal of potentially deploying the deep learning-based automated threat detector, we also worked to integrate the algorithm with the Rapiscan 620DV. The Rapiscan 620DV has an on-

Model	Speed (s/scan)	mAP	Sharps	Blunts	Firearms	LAGs
SSD-InceptionV2	0.042	0.7523	0.408	0.918	0.757	0.907
Faster-RCNN-ResNet101	0.222	0.9166	0.766	0.976	0.944	0.973
Faster-RCNN-ResNet152	0.254	0.9244	0.786	0.980	0.947	0.976
Faster-RCNN-InceptionResNetV2	0.812	0.9410	0.818	0.983	0.962	0.985

Table 3: Inference speed and mAP of the considered feature extractor and meta-architecture combinations. Timing measured on a Nvidia GeForce GTX 1080 Graphical Processing Unit (GPU).

board computer and monitors to construct and display images from the output of the X-ray photon detectors, as well as algorithms for explosives detection. We wish to leave these functionalities untouched, simply overlaying an additional detection output on screen. Therefore, we pipe the constructed scan images to our model, perform inference, and project the predictions to the display (see Figure 2).

To achieve threat recognition, we export a trained model and run it in parallel with existing software. The system computer hardware was upgraded to an Intel i7 CPU and a Nvidia GeForce GTX 1080 GPU in order to support the TensorFlow [2] implementation of the model graph. This allows for a single integrated machine to perform all of the computation for the 620DV, unlike previous implementations that require an additional auxiliary machine to perform the deep neural network computation [22]. While the resulting integrated system has been used for live demos, the experimental results we report in this paper were computed with a held-out test set.

4. Related Work

The development of computer-aided screening for aviation security has garnered much attention in the past two decades. We focus here specifically on efforts to locate and classify potential threats in X-ray images.

Initial work using machine learning to classify objects in X-ray images leveraged hand-crafted features fed to a traditional classifier such as a Support Vector Machine (SVM). In particular, [7] used Bag-of-Visual-Words (BoVW) and an SVM to classify X-ray baggage with feature representations such as Difference of Gaussians (DoG) in conjuction with scale-invariant feature transform (SIFT) [25]. Further BoVW approaches are used for classification in [35], [6], [26], [20].

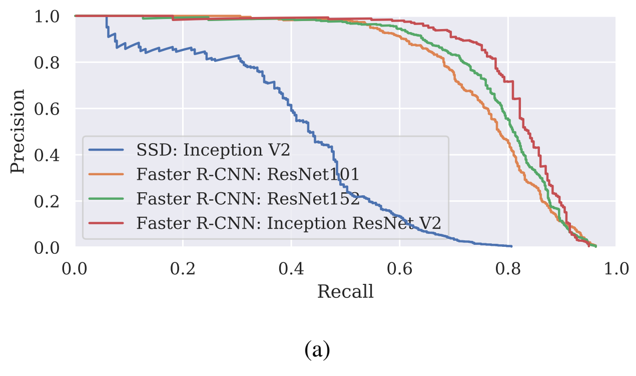
While deep learning has been applied to general image analysis for at least a decade, its adoption for X-ray security image analysis is relatively recent. Still, there are several works that apply deep learning to baggage screening. In [31], the authors provide a review of methods for automating X-ray image analysis for cargo and baggage security, pointing to the use of CNNs as a promising direction. The first application of deep learning to an X-ray baggage screening context was for classifying manually cropped regions of X-ray baggage images that contained different classes of firearms and knives, with additional benign classes of camera and laptop [4]. To perform classification, [4] fine-tuned a pre-trained CNN to their unique datasets, leveraging transfer learning to improve training with a limited number of images compared to the size of datasets that CNNs are typically trained on. In [4], the authors compare their classification performance to the BoVW methods mentioned above.

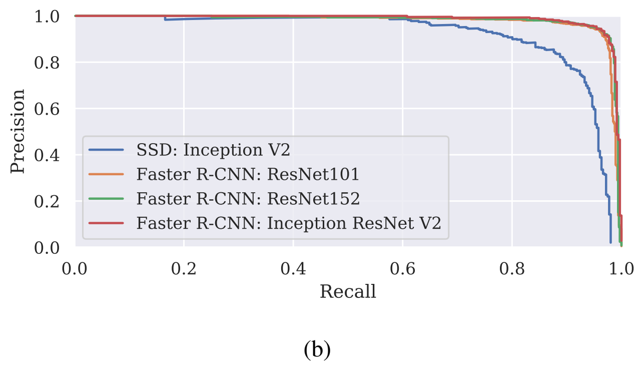
The work of [4] is extended in [3, 5] to examine the use of deep object detection algorithms for X-ray baggage scans. The authors address two related problems: binary identification of objects as firearms or not and a multi-class problem using the same classes as [4]. They expand the CNN classification architectures investigated to include VGG [32] and ResNet [15], and they further adapt Faster R-CNN [30], R-FCN [9], and YOLOv2 [29] as CNN-based detection methods to X-ray baggage. However, these experiments were done in simulation on pre-collected datasets, without any integration into the scanner hardware. They also do not take advantage of the X-ray scanner's multiple views.

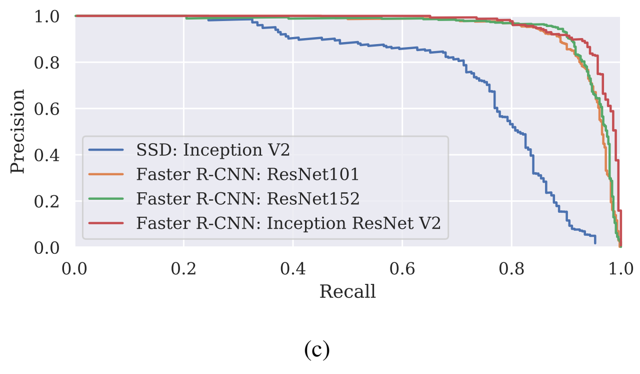
Concurrent with this work, the TSA has sought to incorporate deep learning systems at U.S. airport security checkpoints in other efforts. In [22], the authors present data collection efforts for firearms and sharps classes and compare the performance of five object detection models. Relative to [22], we also include blunt weapons and LAGs categories, and we train a single four-class detector, rather than training an individual detector for each category.

5. Experiments

For our experiments and in-system implementation, we use Google's code base [17] of object detection models, implemented in TensorFlow [2]. We initialize each model with pre-trained weights from the MSCOCO Object Detection Challenge [23] and then fine-tune them to detect each of the target classes (firearms, sharps, blunts, LAGs) simultaneously, which allows us to perform detection four times as fast as if we trained a separate algorithm for each. Since we initialize with weights pre-trained on MSCOCO, we







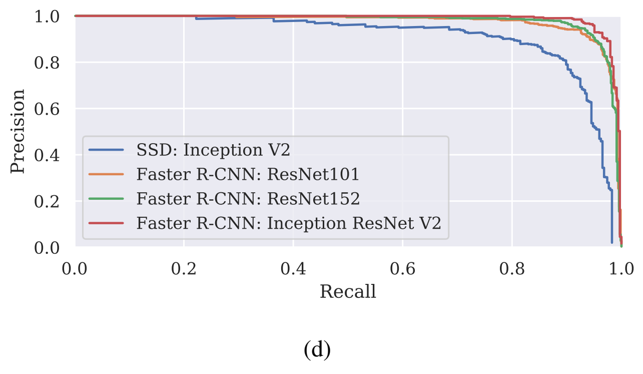
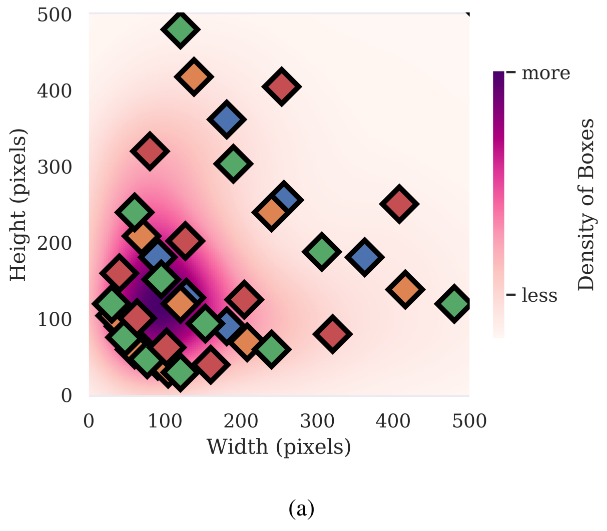


Figure 3: PR curves for four meta-architecture/feature extractor combinations. (a) Sharps, (b) Blunts (c) Firearms (d) LAGs.



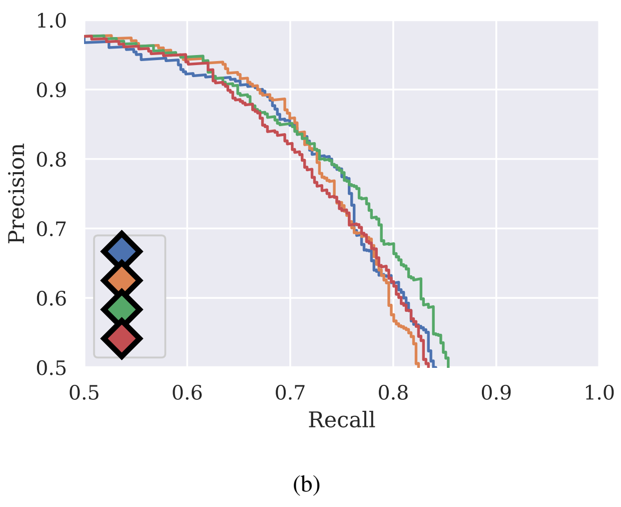


Figure 4: (a) Heatmap indicating density of bounding box dimensions for the training set. Various anchor box distributions are shown; each color indicates a different anchor box experimental setting. Natural image defaults in blue. (b) Precision-recall curves for default anchor boxes and engineered set on sharps. Colors correspond to the distributions shown in (a).

pre-process each image by subtracting from each pixel the channel-means of the MSCOCO dataset; this aligns our pre-processing with that performed on images for the MSCOCO Challenge.

For all Faster R-CNN algorithms, we use a momentum optimizer [27] with a learning rate of 0.003 for 130,000 steps, reducing it by a factor of 10 for 40,000 steps, and reducing by another factor of 10 for a final 30,000 steps.

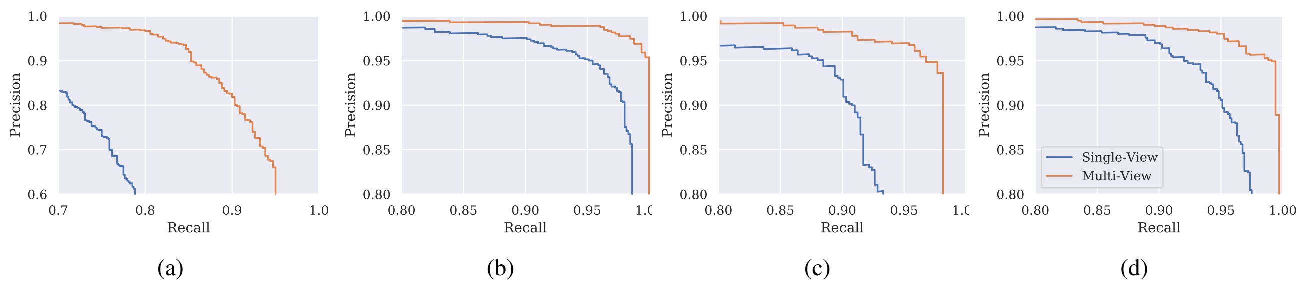


Figure 5: Precision-recall comparison of single view (blue) versus with multi-view (orange) detection for (a) sharps (c) blunts (c) firearms (d) LAGs . Note that the multi-view graphs shown here are a choice of analysis, and not a different technique. The training and inference of Faster R-CNN are the same in both traces.

For the SSD model, we used 200,000 steps of an RMSprop optimizer [16] with an exponential decay learning rate starting at 0.003, and decaying by 0.9 every 4000 steps. During training, a batch size of 1 was used for all Faster R-CNN models, and a batch size of 24 was used for SSD.

From the $13,786$ images collected, we create a $70/10/20$ train-validation-test split, which we use for all experiments. We take care to ensure the two images (views) of a particular bag remain in the same split.

5.1. Feature Extractor and Meta-architecture

As discussed in Section 3, there are many options for both CNN feature extractor and the object detection meta-architecture, each with its own advantages and disadvantages. See [17] for extensive comparisons on MS COCO [23]. For the collected X-ray scan dataset, we choose to analyze several high-performing combinations. Detection performance is measured in terms of AP for each of the classes of interest, and mAP for overall performance is also calculated. We also measure processing time per scan to project practical passenger wait times.

We summarize the results in Table 3 and Figure 3. Overall, Faster R-CNN with Inception ResNet V2 has the highest mAP, while SSD with Inception V2 performed the worst. In general, faster models are less accurate, which may be seen in the "Speed" column of Table 3. Faster R-CNN with the two smaller feature extractors (ResNet101 and ResNet152) achieve nearly the same performance on sharps as ResNet Inception V2, but at more than three times the speed. While the speed of single-stage models is suitable for video frame rates, we found this to be unnecessary for checkpoint threat recognition and to sacrifice too much accuracy.

5.2. Anchor Boxes

As discussed in Section 3.2, bounding box predictions are typically made relative to anchor boxes tiled over the image. The object detection algorithms we have considered were primarily designed for finding common objects (*e.g.* people, animals, vehicles) in natural scenes, with datasets like PASCAL VOC [11] or MS COCO [23] in mind.

Threat	Single View AP	Multiple Views AP
Sharps	0.786	0.935
Blunts	0.980	0.995
Firearms	0.947	0.984
LAGs	0.976	0.994

Table 4: Single View vs. Effective Multiple View APs.

The anchor box distribution is commonly held to act as a kind of "prior" over the training data. In YOLO V2 [29], anchors are learned by k-means clustering, and some of the performance gains of this model are credited to this improvement. We chose several configurations of anchor boxes to better match the distribution of our training data, and display those configurations alongside training dataset bounding box dimension density in Figure 4a. The dataset used for these experiments was smaller than the dataset used for the main findings as described in Table 1. Training, test, and validation sets were drawn from a pool of images containing 2768 Sharps, 1788 LAGs, 1800 Blunts, and 3080 Firearms. This does not impact our conclusions stated in the next paragraph.

During model validation, some of these configurations showed modest gains for sharps, but these did not generalize during testing. The sharps class PR curves for the anchor box distributions in 4a are shown in 4b. We find that performance is robust to different anchor configurations, showing that even with a different box size distribution, Faster R-CNN is able to learn accurate bounding box regressors.

6. Discussion

The results we have shown bear implications for a pilot real-world deployment of this technology. In Table 3, we showed test AP on sharps and timing for four feature extractor/meta-architecture pairs. In a possible real-world

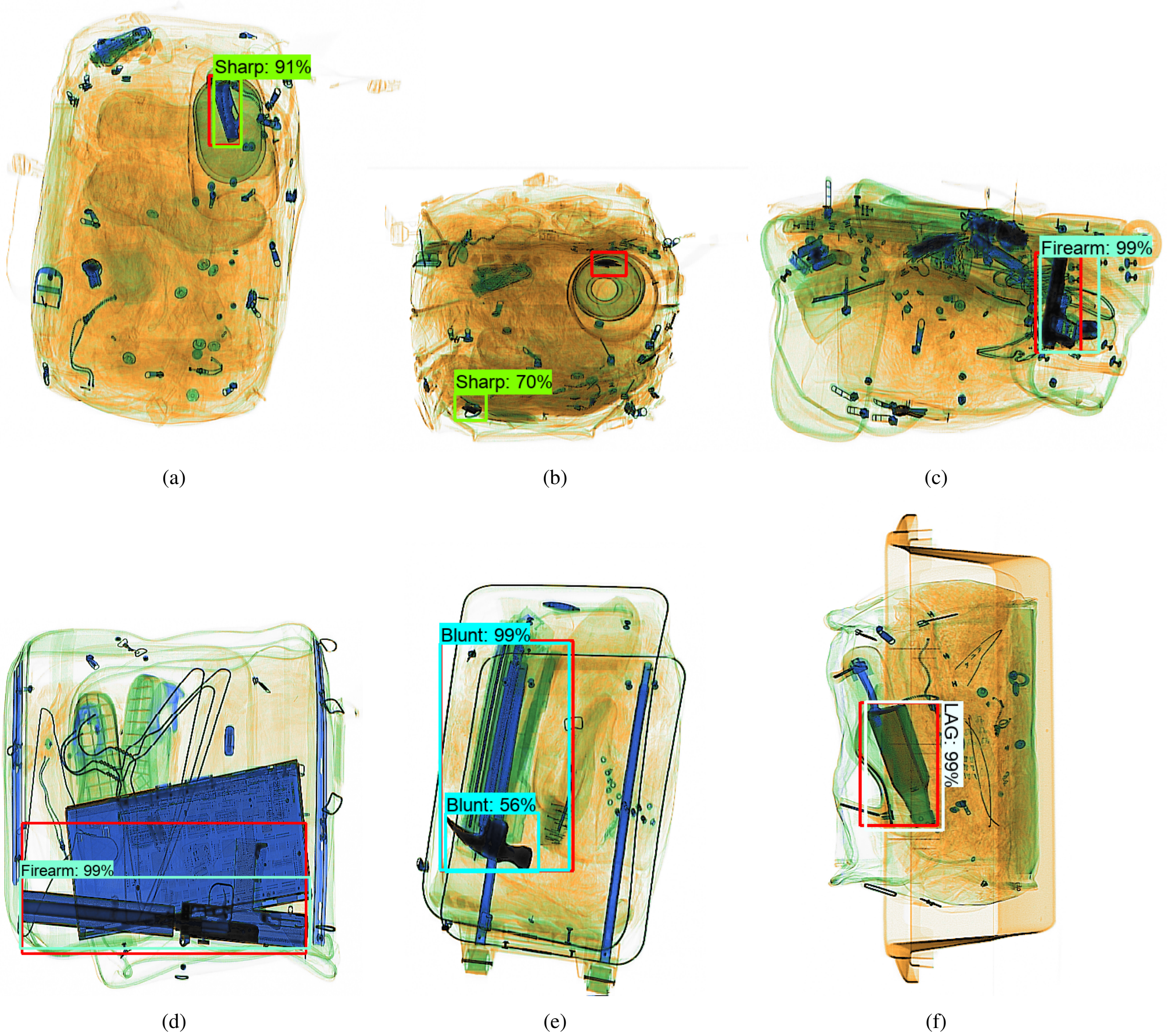


Figure 6: Sample detections from the Faster R-CNN model with ResNet152 as the feature extractor. Ground truth bounding boxes are shown in red. (a-b) Sharp detection. (c-d) Firearm detection. (e) Blunt weapon detection. (f) LAG detection. The color of predicted box and the label indicate the predicted class. Scans produced by a laboratory prototype not in TSA configuration.

system, we strive for inference rates which would not impact screening time and security checkpoint throughput. Because of the long evaluation time of the Faster R-CNN with InceptionV2 model ($\sim$ 800 ms seconds per bag), we recommend use of Faster R-CNN with ResNet152 ($\sim$ 250 ms per bag) for its performance/speed tradeoff. For the remainder of the Discussion section, we will show results only from this model.

6.1. Multiple View Redundancy

Unlike typical object detection research benchmarks, the Rapiscan 620DV provides two views along nearly perpendicular axes of the same scanned object. Within the context of threat detection in X-ray images, this is especially important, as individual views may occasionally be uninformative due to perspective or clutter. Leveraging the two separate views can improve overall performance.

In order to account for the multiple views, we consider a true positive in any view to be a true positive in all views. False positives are added independently across all views. Note that this is not describing a change to the training of the algorithm, nor the inference process. Rather, by performing our analysis in this way, we hope to better represent how the system might work in a potential real-world deployment, when both views are available to a TSO. We show the improvements in PR between single-view and multi-view evaluation in Figure 5 and summarize the AP in Table 4.

6.2. Sample Detections

In Figure 6, we display selected detections from the fully trained Faster R-CNN with ResNet152 as the feature extractor, for a number of threat classes.

In Subfigures (6a-6b), we display two views of the same scan. The very small profile of the folding knife in one view (6b) makes detection challenging for the trained object detector (though there is a low-confidence false alarm). However, the knife presents a more clear profile in Subfigure 6a, and is detected there. This motivates what we call "multi-view" analysis, which we discuss further in Section 6.1.

Subfigure 6e shows a blunt threat which is detected twice. The larger detection, which encompasses the head and handle of a hammer, is a true positive, because the IoU of this detection is greater than 0.5. The other detection in this image, however, only covers the hammer's head. While the presence of the hammer merits an alarm, the detection does not overlap enough with the ground truth, and is therefore a false positive. Some of the training data included hammer heads disconnected from a handle. It may be harder for the CNN to learn to bound hammers with handles or hammer heads only.

To demonstrate detections of the remaining threat classes Subfigure 6f shows a detected LAG, and Subfigures 6c and 6d show scans with firearms. Note that the machine pistol in Subfigure 6d is not as well localized, compared to the firearm in 6c, likely due to the obscuring presence of a laptop. However, such an alarm still makes the threat readily visible to a human operator.

7. Conclusion

We have investigated use of state-of-the-art techniques for the challenging task of threat detection in bags at airport security checkpoints. First, we collected a significant amount of data, assembling by hand many bags and bins which simulate everyday traffic. These concealed a wide variety of threats. We scanned each bag to produce X-ray images, and annotated both views of the scan. We then trained multiple modern object detection algorithms on the collected data, exploring a number of settings and engineering them for the task at hand. We have presented the results of evaluating the model on held-out validation and test data.

In general, we do not find single stage methods to be accurate enough as a security screening method, and their frame rate advantages are superfluous in this application. There are variants of the Faster R-CNN which can run on commercially available computer hardware, and still achieve accurate threat recognition.

In addition to the evaluation presented in Section 5, the TSA has also tested prototype Rapiscan 620DV systems with directly integrated trained models. These results have shown the promise of deep learning methods for automatic threat recognition. Further, they illustrate that the TSA, using X-ray scanners such as the Rapiscan 620DV, has the capability to bring these new technologies to airport checkpoints in the near future.

Acknowledgment

This research has been funded by the Transportation Security Administration (TSA) under Contract #HSTS04-16-C-CT7020. The authors would like to thank the TSA, specifically Armita Soroosh and Suriyun Whitehead, for their administrative support.

References

[1] Passenger Screening Algorithm Challenge. Available online at `https://www.kaggle.com/c/passenger-screening-algorithm-challenge/overview/description`.

[2] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous distributed systems, 2015. Software available from tensorflow.org.

[3] Samet Akcay and Toby P Breckon. An Evaluation of Region Based Object Detection Strategies within X-ray Baggage Security Imagery. In *In Proceedings of the IEEE International Conference on Image Processing (ICIP)*, September 2017.

[4] Samet Akçay, Mikolaj E Kundegorski, Michael Devereux, and Toby P Breckon. Transfer Learning Using Convolutional Neural Networks for Object Classification within X-ray Baggage Security Imagery. In *In Proceedings of the IEEE International Conference on Image Processing (ICIP)*, September 2016.

[5] Samet Akcay, Mikolaj E Kundegorski, Chris G Willcocks, and Toby P Breckon. Using Deep Convolutional Neural Network Architectures for Object Classification and Detection within X-ray Baggage Security Imagery. *IEEE*

Trans. Info. Forens. Sec., 13(9):2203–2215, 2018. doi: 10.1109/TIFS.2018.2812196.

[6] Muhammet Bastan, Wonmin Byeon, and Thomas Breuel. Object Recognition in Multi-View Dual Energy X-ray Images. In *In Proceedings of the British Machine Vision Conference (BMVC)*, January 2013.

[7] Muhammetand Baştan, Mohammad Reza Yousefi, and Thomas M. Breuel. Visual Words on Baggage X-Ray Images. *Computer Analysis of Images and Patterns.*, pages 360–368, 2011.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[9] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-FCN: Object Detection via Region-based Fully Convolutional Networks. In *In Proceedings of the Neural Information Processing Systems Conference (NIPS)*, December 2016.

[10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A large-scale hierarchical image database. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2009.

[11] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010. doi:10.1007/s11263-014-0733-5.

[12] Ross Girshick. Fast R-CNN. In *In Proceedings of the International Conference on Computer Vision (ICCV)*, December 2015.

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[14] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT press, Cambridge, MA, USA, 1st ed edition, 2016. `http://www.deeplearningbook.org`, ISBN: 9780262035613.

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[16] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural Networks for Machine Learning – Lecture 6a – Overview of Mini-Batch Gradient Descent, 2012. Available online at `https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`.

[17] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, and et al. Speed/Accuracy Trade-offs for Modern Convolutional Object Detectors. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[18] Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *In Proccedings of the International Conference on Machine Learning (ICML)*, 2015.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *In Proceedings of the Neural Information Processing Systems Conference (NIPS)*, December 2012.

[20] Mikolaj E. Kundegorski, Samet Akçay, Michael Devereux, Andre Mouton, and Toby P. Breckon. On Using Feature Descriptors as Visual Words for Object Detection within X-ray Baggage Security Screening. In *In Proceedings of the International Conference on Imaging for Crime Detection and Prevention (ICDP)*, November 2016.

[21] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.*, 1(4):541–551, 1989. doi:10.1162/neco.1989.1.4.541.

[22] Kevin J Liang, Geert Heilmann, Christopher Gregory, Souleymane Diallo, David Carlson, Gregory Spell, John Sigman, Kris Roe, and Lawrence Carin. Automatic Threat Recognition of Prohibited Items at Aviation Checkpoints with X-Ray Imaging: a Deep Learning Approach. In *In Proceedings of the SPIE Conference on Anomaly Detection and Imaging with X-Rays (ADIX) III*, April 2018.

[23] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common Objects in Context. In *In Proceedings of the European Conference on Computer Vision (ECCV)*, June 2014.

[24] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. SSD: Single Shot MultiBox Detector. In *In Proceedings of the European Conference on Computer Vision (ECCV)*, October 2016.

[25] David G. Lowe. Object Recognition from Local Scale-Invariant Features. In *In Proceedings of the International Conference on Computer Vision (ICCV)*, September 1999.

[26] Domingo Mery, Erick Svec, and Marco Arias. Object Recognition in Baggage Inspection Using Adaptive Sparse Representations of X-ray Images. *Image and Video Technology*, pages 709–720, 2016.

[27] Ning Qian. On the Momentum Term in Gradient Descent Learning Algorithms. *Neural Comput.*, 12(1):145–151, 1999. doi:10.1016/S0893-6080(98)00116-6.

[28] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[29] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[30] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *In Proceedings of the Neural*

Information Processing Systems Conference (NIPS), December 2015.

[31] Thomas W Rogers, Nicolas Jaccard, Edward J Morton, and Lewis D Griffin. Automated X-ray Image Analysis for Cargo Security: Critical Review and Future Promise. *J. Xray Sci. Technol.*, 25(1):33–56, 2017. doi:10.3233/XST-160606.

[32] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *In Proceedings of the International Conference on Learning Representations (ICLR)*, May 2015.

[33] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting Unreasonable Effectiveness of Data in Deep Learning Era. In *In Proceedings of the International Conference on Computer Vision (ICCV)*, October 2017.

[34] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In *In Proceedings for the Workshop of the International Conference on Learning Representations (ICLR)*, May 2016.

[35] Diana Turcsany, Andre Mouton, and Toby P. Breckon. Improving Feature-Based Object Recognition for X-Ray Baggage Security Screening Using Primed Visual Words. In *In Proceedings of the IEEE International Conference on Industrial Technology (ICIT)*, February 2013.

[36] Jay Wagner. TSA Year in Review: A Record Setting 2018. Available online at `https://www.tsa.gov/blog/2019/02/07/tsa-year-review-record-setting-2018`.

[37] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *In Proceedings of the Neural Information Processing Systems Conference (NIPS)*, December 2014.

[38] Matthew D. Zeiler and Rob Fergus. Visualizing and Understanding Convolutional Networks. In *In Proceedings of the European Conference on Computer Vision (ECCV)*, September 2014.

[39] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. Learning Transferable Architectures for Scalable Image Recognition. In *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.