Automated passenger screening based on TSA passenger scan data

Vashistha Harsh, Bhattacharyya Deepanjan, Patki Prachi, Manjunatha Sushmitha
University of Maryland Baltimore County
{harsh5,deep8,patki1,sushmi1}@umbc.edu

I. Abstract

Full body scan is a normal security procedure employed at the airports in USA. These body scanners use millimeter wave technology to get a 360° scan of the passengers. The scans can penetrate clothing and can reveal objects attached to the body. However, detecting something being attached to the body is more or less a manual process and consumes time and decrease the competence. Due to this limitation, security queues at the airports usually end up being unnecessarily long and time consuming. Our service is an attempt to automate this process using machine learning and reduce the time it takes to go through such security procedures. It will not only reduce the hours wasted in security queues but will also reduce the manual labor involved in the process and uphold the privacy of all those going through the scanners.

Keywords— WADL, REST, Service Oriented Architecture, Convolutional Neural Network, Deep Learning Network, Transfer Learning

II. Introduction

In 2017,the global air traffic passenger demand has increased by over 7.5% compared to the earlier year. With the increase of air traffic passenger demand, security becomes a prime concern. The TSA (Transportation Security Administration) department of Homeland Security of USA works for providing security measures by detecting individuals in possession of threatening item.

The state of the art surveillance system developed for airport security to detect and identify threatening objects, which are concealed on the human body has a huge advantage compared to conventional metal detectors which fail to detect nonmetallic objects such as plastic explosives and plastic guns. This system is based on millimeter-wave scan technology and a holographic imaging algorithm to provide multi-dimensional images of objects hidden beneath clothing. The algorithm is based on image processing filters and artificial neural networks that locates and segments threats and places them on either a silhouette of the person or a wire-frame humanoid representation. Our project involves accurately predicting the threat zone present in a fully scanned body image.Threat here is any abnormal object present on the clothes or hidden inside the clothes. This system is ideally suited for mass transportation centers such as airport checkpoints that require high throughput rates.

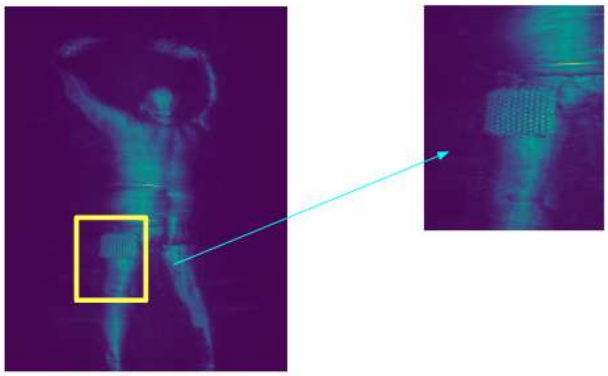

Figure 1. Image in the left shows an full body image form a certain angle of the total 3D image with a threat present in it. Image on the right shows a zoomed image of just the zone for a certain angle in which the threat is present.

III. Related Work

The applications of CNN are varied and stretch along different fields. It is used prominently in the clinical applications which serves the purpose of understanding the causes of diseases and find a remedy over the same from various kinds of images (MRI, DiCom, etc.). We studied about the classification of brain CT-Scan images which contained data of patients with Alzheimers disease or Lesion or normal aging. The reason behind choosing CNN on medical images as related work is the similarity of the medical image data and our data. Where the base object remains constant throughout, and the differentiating factor is an object or pattern with size very less as compared to base image. The research acquainted us with the fact that when CNNs are trained with a proper regularizer, they can achieve high performance on visual recognition tasks without depending on the handcrafted features. The work addressed in the [1] helped us understand the version of CNN used for scanned images and thereby provided a base for implementing the project.

IV. Method

While it is very easy for humans to identify places, things or environments, Computer face difficulty for the same task. But with the help of machine learning and computer vision technologies, image recognition is possible. Image recognition implies resembling the working of prefrontal cortex in human brain and thus, is complex. For instance, to devise self-driving cars, the overhead is to differentiate

between various instances of videos and photos. Solution for this is neural network which sacrifices ability to generalize for a feasible output.

The convolutional network is preferred over conventional network as the computational overhead of conventional network makes the network less accurate. Filtering in convolutional networks makes image processing computationally manageable and these networks can also be made to generalize well over different kinds of objects or classifications.

The concept of CNN was best suited for the project as the project involved looking for low level features like curves and edges. These, features then help to distinguish the abnormal object on or inside the clothes of passengers entering the security checkpoint. Also, with the current interest and research going into CNNs for image recognition tasks and other related areas of audio/video, these network provides a ready base for this task. There are multiple libraries and frameworks such as Tensorflow, Caffe, etc. which makes it feasible to quickly setup one such network and start training.

V. Implementation

A. Dataset

The data set contains body scan images produced by scanning devices and is acquired via Kaggle. The total number of images present in the dataset are 1247 (each image has 16 angles and can be divide into 17 zones) including training and testing data. Each aps image has a dimension 512x600x16. Data set is available in three different formats. These formats are as follows:

- .a3daps = combined image angle sequence file (41.2MB per file)
- .aps = projected image angle sequence file (10.3MB per file)
- .a3d = combined image 3D file (330MB per file)

We decided to go ahead with .aps (10.3MB per file) and a3daps (41.2MB per file) which is feasible for computation. Each aps file has 16 processed images equally spaced at angle 22.6° around the axis covering 360° around the person while a3daps has 64 raw scan projections equally spaced at 5.625°. The result of this is an image file that, when played back plane-by-plane, appears like the object is spinning on the screen. The Figure in 13 is the representation on single aps file.

Along with this we had zone wise threat labels in the data set. Each image had 17 labels representing the presence of any object in the corresponding body zone.The dimensions defining 17 zones are provided by TSA. 2 shows a snippet from the csv file of labels for one of the aps image.

B. Data-set description

The data-set distribution in terms of 'safe images' i.e. images with no threats present in any of the zones vs. images with a threat present in at least one of the 17 zones is as shown in Figure 3. The threat count distribution over 17 zones is as shown in Figure 4 below. It is observed that most of the times the threat is present in 'zone 1', 'zone 13' and 'zone 14'. But overall the data set is evenly distributed.

	Id	Probability
2	00360f79fd6e02781457eda48f85da90_Zone1	0
3	00360f79fd6e02781457eda48f85da90_Zone10	0
4	00360f79fd6e02781457eda48f85da90_Zone11	0
5	00360f79fd6e02781457eda48f85da90_Zone12	0
6	00360f79fd6e02781457eda48f85da90_Zone13	0
7	00360f79fd6e02781457eda48f85da90_Zone14	1
8	00360f79fd6e02781457eda48f85da90_Zone15	0
9	00360f79fd6e02781457eda48f85da90_Zone16	0
10	00360f79fd6e02781457eda48f85da90_Zone17	0
11	00360f79fd6e02781457eda48f85da90_Zone2	0
12	00360f79fd6e02781457eda48f85da90_Zone3	0
13	00360f79fd6e02781457eda48f85da90_Zone4	0
14	00360f79fd6e02781457eda48f85da90_Zone5	0
15	00360f79fd6e02781457eda48f85da90_Zone6	0
16	00360f79fd6e02781457eda48f85da90_Zone7	0
17	00360f79fd6e02781457eda48f85da90_Zone8	0
18	00360f79fd6e02781457eda48f85da90_Zone9	0

Figure 2. Dataset Threat labels

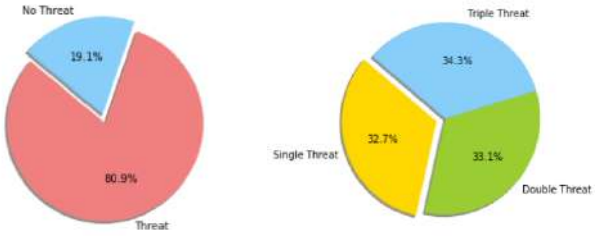

Figure 3. Threat distribution in data-set

C. Data-processing

In order to enhance image features and remove noise. Several image processing techniques were applied on the data.

C.1 RGB to Grey Conversion

Originally the images in the dataset were in the form of RGB images. RGB to Greyscale conversion and contrast adjustments are done to extract features obtained by luminance difference.

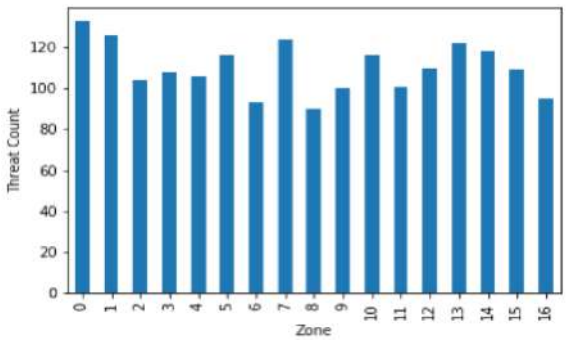

Figure 4. Zone wise threat count

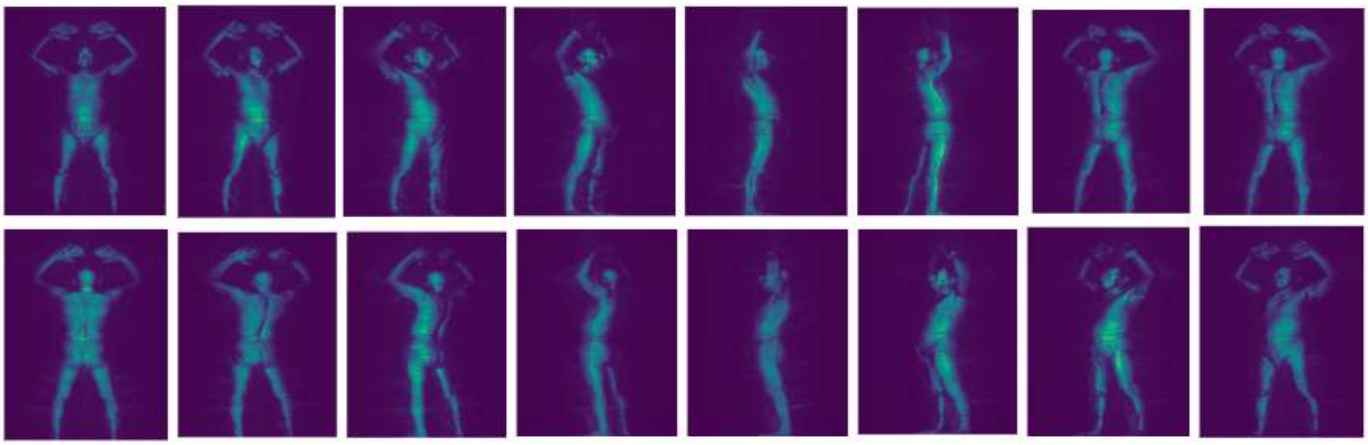

Figure 5. Representation of a single .aps file for images from all the angles each separated by 22.5 degrees

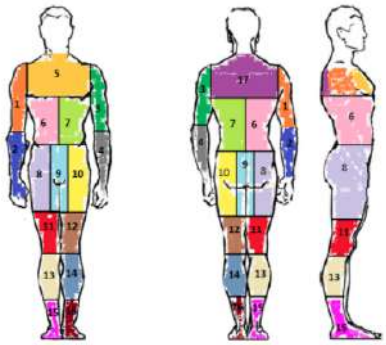

Figure 6. Zone wise threat labels

C.2 Thresholding

To enhance the human figure out of the images we applied constant threshold values on numpy arrays of images. A threshold for pixel is kept 50 as all the pixel values less than 50 are nothing but background noise.

C.3 Image Augmentation

Since we had limited number of images in data set, just 1247 in total, we used image augmentation techniques by shifting an image horizontally and/or flipping horizontally/vertically.

D. Approach

D.1 Feeding the data into CNN

We had to consider to major factors while feeding the data into CNN:

• Zones: First approach was to give full images as an input to CNN. As the threat object is very trickily hidden under the clothes of a person.There was a minor difference between safe image and image with threat when looking at images.The model was unable to tell the difference looking at the full images. Therefore, we decided to go for the option to feed the model with zone wise segments of images. After receiving an input image, it is segmented into 17 pieces as shown in Figure 6. Following is the representation of one of the segments.

Examples of different segments

• Angles: Initially, all the 16 angles from an aps file were considered for each model. As this was taking long time for computation we decided to consider three angles(i.e. angle taking front view, back view and side view for each zone). The thought behind doing this is, these three angles are primary contributors in detecting any threat present in that zone.

Figure 7. Zone wise Data Arrangement

• Data Arrangement: Next task was to find a way to coordinate angles and zones. As we didnt have the labels zone wise separated in the given labels. We separated labels zone wise and divided a single CSV labels file into 17 different CSV files. To get the zone information, we first

created a set of dimensions to crop the full image into 17 segments. After that we wrote separate script which give output as multidimensional array that contains all the file ids that are present in the dataset. Each id points to 17 arrays representing 17 zones. Each zone is further pointing to its numpy array and its respective labels. The dataset arrangement flow is as shown in Figure 7.

E. Machine Learning Model

We tried different approaches to train a machine learning model. The basic algorithm is Convolutional Neural Network in every approach.

E.1 Randomized weights

The idea behind this approach was to build a model that detects threats by finding inconsistencies. When we see an image for threat, we just dont see the whole body, but rather concentrate on each part at a time. We wanted to approach the problem from a similar point of view. For example, we started with zone 13 which shows the left calf of an image. The model tried to learn the shape and generic texture of it, which would then enable it to detect objects as it would be distorting the shape or texture.

We divided the whole image based on zones. Then we trained a model for each zone. The zonal image was pushed into a CNN pipeline. The pipeline consists of a total of 9 layers. Three convolution layers, two pooling layers followed with by two fully connected layers sandwiching a dropout layer. The initial convolutional layers learn the shape and texture. The fully connected layer gives the final output a binary classification, 1 for threat or 0 for no-threat. A total of 17 models were trained. Total dataset used had 1247 instances. 900 of them were used as training dataset and 247 images were used as the test data set. The remaining 100 were unlabeled which we used to evaluate the model.

```
# Input image tensor

network = input_data(shape=[None, dimY, dimX, 1],
                     data_preprocessing=img_prep,
                     data_augmentation=img_aug)
```

Figure 8. Input Tensor

```
# Step 1: Convolution

network = conv_2d(network, 32, 3, activation='relu')

# Step 2: Max pooling

network = max_pool_2d(network, 2)
```

Figure 9. Convolution and Max-pool Layer

When an input image was fed for threat detection, it is broken into 17 images zone wise and fed to respective 17 models to evaluate and the results are combined as a binary array of length 17. The model architecture is as shown in Figure 10.

E.2 Transfer Learning with fine tuning

Because of the limited number of training data, we decided to use a pre-trained ImageNet Model with VGG16 Architecture. In this model the initial layer were frozen to extract basic features (edges, shapes) and last two layers were fine tuned. In the transfer Learning model the input is full body image. The a3dps image set was used and the input is given in batches. A total of 64 images were there for each input comprising the 360 degrees. 16 batches consist of images at 90 degree separations angles (1 to 16), (17 to 32), (33 to 48) and (49 to 64) respectively. The architecture of this model is as shown in Figure 11.

F. Web Service Description

A web service is a collection of open protocols and standards used for exchanging data between applications or systems. Software applications written in various programming languages and running on various platforms can use web services to exchange data over computer networks like the Internet in a manner similar to inter-process communication on a single computer. It is a unit of managed code that can be remotely invoked using HTTP, that is, it can be activated using HTTP requests. Web services allows you to expose the functionality of your existing code over the network. Once it is exposed on the network, other application can use the functionality of your program.

F.1 Interoperability

Web services allow various applications to talk to each other and share data and services among themselves. Other applications can also use the web services. For example, a VB or .NET application can talk to Java web services and vice-versa. Web services are used to make the application platform and technology independent.

F.2 Standardized Protocol

Web services use standardized industry standard protocol for the communication. All the four layers (Service Transport, XML Messaging, Service Description, and Service Discovery layers) use well-defined protocols in the web services protocol stack. This standardization of protocol stack gives the business many advantages such as a wide range of choices, reduction in the cost due to competition, and increase in the quality.

F.3 Low Cost Communication

Web services use SOAP over HTTP protocol, so you can use your existing low-cost internet for implementing web services. This solution is much less costly compared to proprietary solutions like EDI/B2B. Besides SOAP over HTTP, web services can also be implemented on other reliable transport mechanisms like FTP.

F.4 Service Architecture

The architecture used was a standard three layer architecture. Front end middle layer and a back end. As the machine learning models were implemented using python so

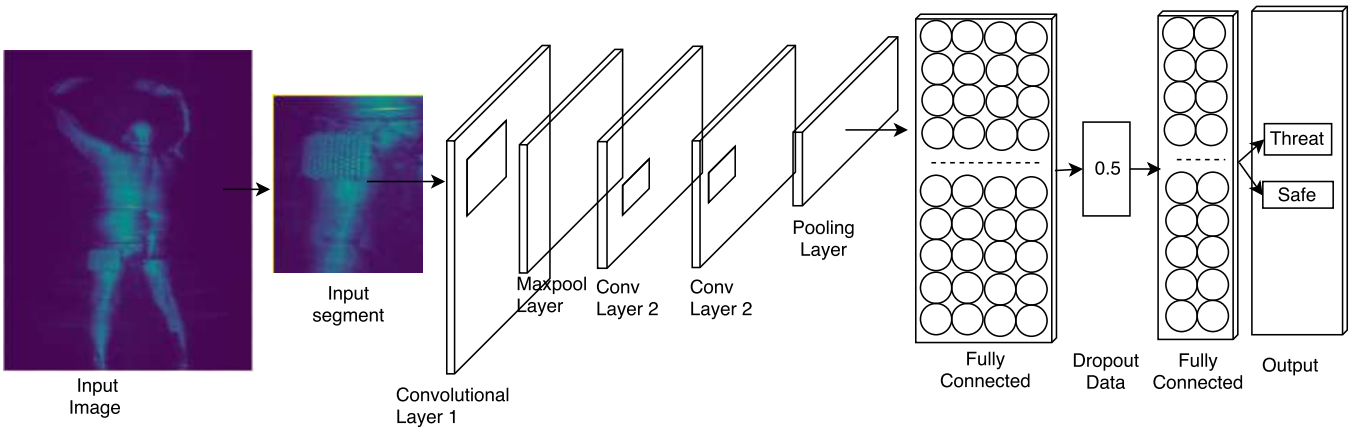

Figure 10. CNN model architecture with Randomized weights

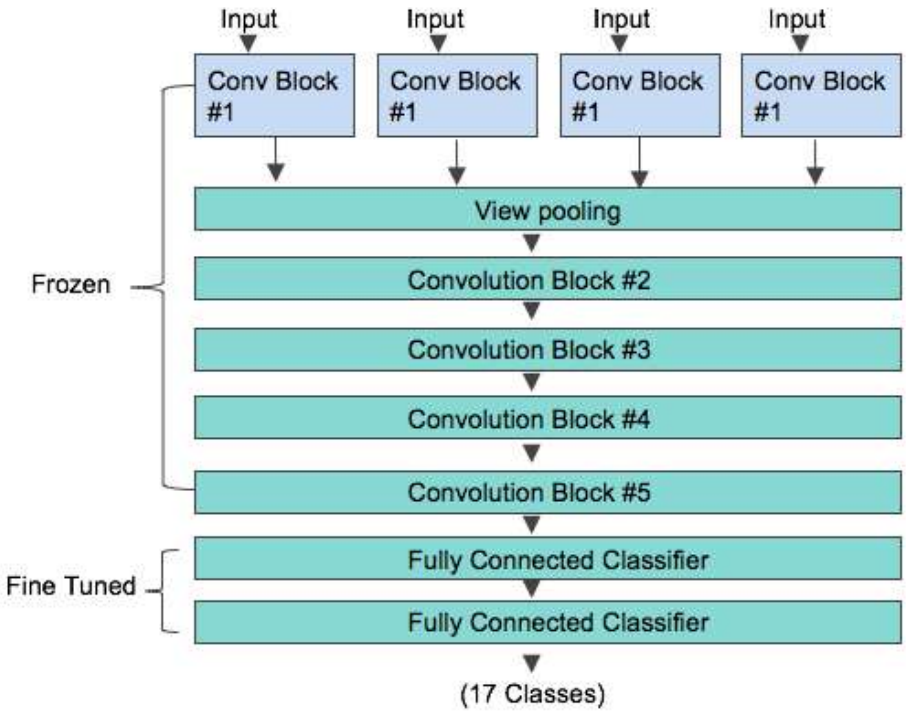

Figure 11. Transfer Learning Model:ImageNet(Vgg16)

we opted to use python as our middle layer service provider architecture as well.

1. Front End:
We created a web application using HTML5 CSS3/Bootstrap, Javascript/Jquery that can interact with our middle layer and provide a user interface for a user to interact with our system. Basic features included were login, registration and evaluating threat zones. After successful registration and correspondingly logging in, one lands on the request page to see a list of past requests their status and also add new request. A request comprises of uploading a image file to evaluate threat. Once the evaluation is complete the threat zones would be highlighted for that particular image.

2. Middle Layer:
The middle layer was made primarily on python flask. It allowed us to natively interact with our models as well as expose rest api's that our front end was using. Being service based enables flexible application development via just consuming our services as per user needs.

2 a. Registration: Service type POST. Input: User Name, First name, Last Name, Password and Email. This service was meant to create the initial registration for a user.

2 b. Login: Service type POST. Input: User Name and Password. This service was meant to authenticate a registered user.

2 c. Analyze: Service type POST. Input: Image file (aps,a3dps, a3d). This service takes in the image and spits out the request Id which is used to give details about the request.

2 d. Request Details: Service type GET. Input: Request id. This service gives out a status output(Processed/Not Processed). If processed also gives out the zone wise threat probability.

2 e. Request List: Service type GET. Input: User id. Gives out a list of all the requests made by this particular user.

2 f. Request Feedback: Service type POST. Input: Request Id and updated zone-wise probability. This service updates the zone-wise threat zone probability for a particular request id, which is then used to reuse the model.

3. The Back end:
The back end consisted of AWS EC2 instance on top of which our middle layer was hosted. The front end was supported by the AWS S3 platform. For the database we used AWS dynamoDb for scalability and for queuing with high throughout we opted for AWS SQS which is a queuing service which we used to queue our incoming requests.

VI. Results

A. Accuracy of the model

The train and test data ratio in the dataset is 80:20. The accuracy is plotted according to zones. At present, we have not evaluated all the 17 zones. We are considering eight major zones where threat probability is more that the other zones. Also, we are considering three primary angles for each zone which are front, back and respective side of the zone. The plots below show accuracy respective to the zones. The accuracy is calculated by considering true positives and true negatives predictions on test images. The model most accurately predicted the threat present in the zone 7 which was area around the abdominal region. Whereas, we observed decrease in accuracy in the zone 8 and 10 in the areas around hips. After looking at the training data we realized that because of the presence of pockets and other cloth lines present on the bottom-wear of the subjects, it is ambiguous to tell if it is cloth pattern or threat object. The average accuracy is 87.20%.

B. No.of epochs vs. Accuracy

We have trained the model for different number of epochs in the range 1 to 500. The model acquired the best accuracy at 100 epochs. Figure 12 shows the changes in accuracy w.r.t. increase in the number of epochs.

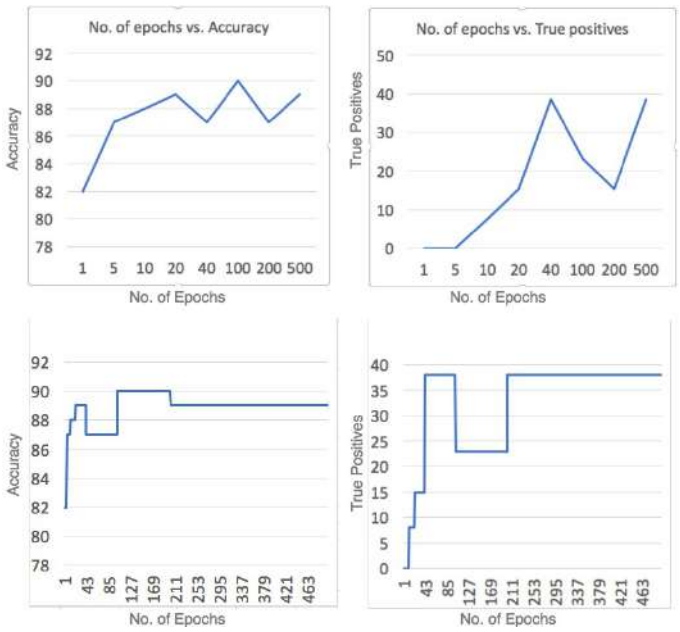

Figure 12. No. of Epochs vs. Accuracy

VII. Conclusion

This report presented an implementation of machine Learning model which takes an input as segments(zones) of scanned images and predicts if a zone has a threat present in it. Final prediction is an array of binary classification where '0' stands for safe images and '1' stands for an image with a threat. We have achieved an average accuracy of 87.2%.This accuracy is achieved on 100 epochs. This model can be taken as good starting point to developed an deep CNN integrated with all the zones and angles to achieve better Accuracy and Sensitivity which can make it a potential application in TSA scanning systems.

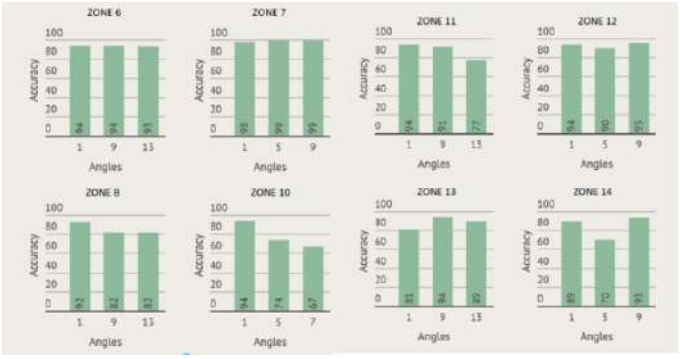

Figure 13. Zone wise accuracies

VIII. Future Work

The first task that we are planning to implement as a part of future work is building models to detect the threats in all the 17 zones.Currently we have implemented 8 zones.

The integration of these 17 models using a concept 'Multiple Instance Learning'.Where the labels will be assigned to bag of instances.A bag is labeled positive if at least one

instance in that bag is positive, and the bag is labeled negative if all the instances in it are negative.

After covering all the zones the model should be exposed to to all the 16 angles which are provided as a part of single .aps file.Presently we are considering three angles (i.e. angle taking front view,back view and side view for each zone) for each zone.

We started with 'Transfer Learning Approach' to build this model.but later shifted to Randomized weights approach.Implementing the model with 'Transfer learning approach with fine tuning will be a good step to check the improvement in accuracy and sensitivity.

References

[1] Daniel Hammack and Julian de Wit. *Predicting Lung Cancer.* https://eliasvansteenkiste.github.io/machine%20learning/lung-cancer-pred/

[2] D.M. Sheen, D.L. McMakin T.E. Hall. *Three-dimensional millimeter-wave imaging for concealed weapon detection.* IEEE Transactions on Microwave Theory and Techniques (Volume:49, Issue:9, Sep 2001)

[3] Boris Kapilevich, Moshe Einat. *Detecting Hidden Objects on Human Body Using Active Millimeter Wave Sensor.* IEEE Sensors Journal (Volume:10, Issue:11, Nov 2010)

[4] Hang Su, Subhransu Maji, Evangelos Kalogerakis, Erik Learned-Miller. *Multi-view Convolutional Neural Networks for 3D Shape Recognition.* CoRR (Volume: abs/1505.00880, 2015)