Gurleen Dhody and Steve Babcock

4/29/2017

GunVision

INTRODUCTION

Our project aims to create a smart security camera capable of detecting a raised firearm using computer vision techniques such as background subtraction, template matching, and neural networks. It could be used in convenience stores or banks to send an early alert to police when a robbery is in progress. We have a sizable code base built using C++, OpenCV, and OpenGL and Caffe. The project has been compiled on Mac OS X and Linux Machine (IUB BigRed2), but should be very portable to other operating systems. Many assumptions about the problem have been made, and the solving philosophy we choose strives to perform well within these bounds. However, it is not concerned about performance outside the assumptions. These include

1. fixed location of the camera
2. consistent light source
3. few windows/doors in the background
4. the objects moving about the scene are humans
5. a buffer of 3-4 feet to the camera is generally maintained
6. the left/right position of a cashier relative to the scene is known
7. the camera is centered upon the position where a robber is likely to stand
8. this view is perpendicular to the line between the cashier and robber
9. and this view is close to parallel with the ground

Obviously, many physical locations such as outdoors and tight quarters will not be viable for deployment of our project. Though we believe many of the most important locations that our project could benefit do fit these assumptions.

Current surveillance and control systems still require human supervision and intervention. This work presents a novel automatic handgun detection system in videos/images appropriate for both, surveillance and control purposes. We reformulate this detection problem into the problem of minimizing false positives and solve it by building the key training data-set guided by the results of a deep Convolutional Neural Networks (CNN) classifier (Google Inception v1 Net)

BACKGROUND AND RELATED WORK

The most similar of the research papers studied is [1] Automated Detection of Firearms and Knives in a CCTV Image written by Grega, Matiolanski, Guzik, and Leszczuk. Their algorithm is significantly longer: background detection, Canny edge detection, sliding window, scaling, PCA, neural network, candidate regions, MPEG-7 classifier, spa/temp filtering, and decision. Similarities exist between our two approaches. Both begin with background subtraction, and use a sliding window technique later. However, they proceed to detect human bodies, and then search around the body for a weapon. They also have a much broader definition of the problem - attempting to detect lowered handguns and non-perpendicular views. Their results show a sensitivity of 35.98% and specificity of 100%. Our results will show a higher sensitivity and lower specificity. The higher result is likely regarding our narrower definition of the problem. Another difference is they only noted using an umbrella and bag as random objects. While we used several objects with similar dimensions to that of a gun barrel, and held in a way

to mimic a handgun. Overall, it is difficult to compare the two results due to these differences. On further exploration, we found two sub-areas where surveillance of hand guns take place.

The first and traditional sub-area in gun detection focuses on detecting concealed handguns in X-ray or millimetric wave images. The most representative application in this context is luggage control in airports. The existent methods achieve high accuracies by using different combinations of feature extractors and detectors, either using simple density descriptors, border detection and pattern matching or using more complex methods such as cascade classifiers with boosting. The effectiveness of these methods made them essential in some specific places. However, they have several limitations. As these systems are based on metal detection, they cannot detect non-metallic guns. They are expensive to be used in many places as they require to be combined with X-ray scanners and Conveyor belts. They are not precise because they react to all metallic objects.

The second sub-area addresses gun detection in RGB images using classical methods. The few existent papers essentially apply methods such as SIFT (Scale-Invariant Feature Transform) combined with Harris interest point detector. For example, the authors in [6,7] developed an accurate software for pistol detection in RGB images. However, their method is unable to detect multiple pistols in the same scene. The used technique consists of first, eliminating non-related objects to a pistol from the segmented image using K-mean clustering algorithm then, applying SURF (Speeded Up Robust Features) method for detecting points of interest. Similarly, the authors in [8] demonstrated that BoWSS (Bag of Words Surveillance System) algorithm has a high potential to detect guns. They first extract features using SIFT, cluster the obtained functions using K-Means clustering and use SVM (Support Vector Machine) for the training.

We divided our approach into 2 parts

- Approach 1 – Smart Feature/Template Matching
- Approach 2 – Deep Learning Google Net v1

APPROACH - 1

The detector is essentially a chain of sub-algorithms where positive results from one sets off the next step, and negative results break the flow. First, we anchor the camera, then generate a background binary map, next search for a raised arm, and finally scan the end of the arm for a gun. The anchoring is performed by selecting eight pixels spread around the edges of the scene. If less than three pixels do not match the values of the previous frame, the camera is said to be unanchored, and this frame will not be processed further. Enough consecutive unanchored frames can result in the deletion of a background image if one has been generated.

A background image holds value we assume to be the unmoving parts of a scene. In our code, it is a matrix of RGB values just like a normal image. Except one extra property is added to each pixel - a certainty value which is initialized to zero. As more video frames can in, each new pixel is compared to its corresponding background pixel. If they are similar, that background pixel's certainty is increased - decreased otherwise. This similarity has an inverse relationship to the maximum difference between the three-color channels of the pixels. When the net certainty of a background rises above a threshold, it is said to be generated, and can be used to produce binary maps of a scene. Given a new video frame, any pixel similar to its corresponding background pixel is assumed to be part of the background and re-assigned to a zero. Otherwise, the pixel is assumed to be an active part of the scene and re-assigned to a one. Also, if the certainty of a background pixel falls below a threshold, its value is replaced by the next incoming

frame and its certainty is reset to zero. This allows for adaptation to changes in lighting, camera position, and new permanent additions to a scene.

Next, we use dynamic programming to rapidly convolve an arm template through the binary map searching for a raised arm. Below is a rough visual representation of this template where white is positive (1), gray is neutral (0), and black is negative (-1).

Each column of the template is uniform. So, moving it over simply requires the next column to be added, and the oldest column to be subtracted from a running total. The assumption made here is that a robber is unlikely a hold a gun at his or her hip.

Finally, we use a series of gun templates to scan the end of a detected arm for a weapon. This process is divided into four parts: matching the whole template, score of the barrel section of the template, color channel balance, and smart template matching. Below is an example of a gun template which is a matrix of 1's where a gun or hand should be, 0's forming a buffer around the 1's, and -9's in the areas where nothing should be seen by the background subtractor.

At the place where this template scores the highest (and is above a threshold), we consider the score of the right side or gun barrel. This is done because the left side will match highly against a hand holding any object. The barrel is the most identifying part of a gun, but if we were to convolve a barrel template by itself, unexpected results often happen. That is, it may match well against the top/bottom of an arm or random noise. Also since the position of the cashier is known in our assumptions, the program is started with no arguments when the gun is expected to be aimed stage left of the camera. It is started with any argument otherwise, and the input will be mirrored horizontally before passed to any detection algorithm.

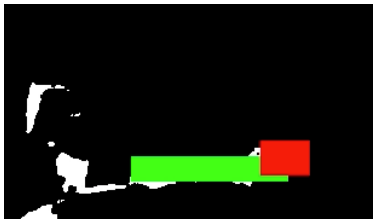
Next, we analyze the values of the three color channels in the original video frame at the locations where the gun barrel is thought to exist. If there is a significant difference in these values, we ignore the frame in regard to containing a gun. Our assumption here is that a robber is unlikely to possess a blue gun or a red gun, etc. We are only interested in detecting black, gray, or silver weapons.

Finally, the detector has a training mode that will save to a file the contents of the binary map in the form of a vector at the location a gun has been detected. After generating many of these recorded at different distances (and potentially with different gun models), we have much more exact ideas of what a gun looks like to our background subtractor at its current physical location. The entire collection can then be compared future recognitions of a gun by rearranging the image segment into a vector, and calculating minimum euclidian distance. If it is below a threshold, we conclude there is a raised handgun in the scene.

We performed two rounds of testing. An initial round was in Steve's room at a distance of 8-9 feet with five rounds and five objects. About a week later, the final round occurred in the Mac lab of Lindley Hall's basement. It used five objects and ten rounds at distances of 4-5 feet,

6-7 feet, 8-10 feet, 12-15 feet, and 18+ feet. Our program draws a green rectangle when detecting an arm, and a smaller red rectangle upon detecting a gun. Reviewing the testing video, and counting the presence of these rectangles determines the statistics. For example…





APPROACH – 2

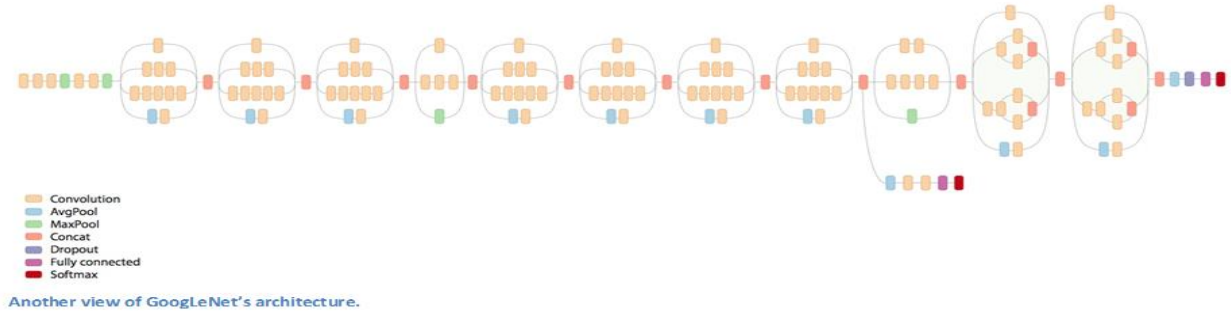
We took the pretrained Google-Net v1 CNN model on ImageNet and changed The Data Layer, its 1st convolution layer and the output layer which now has 2 classes. Weather it will be Handgun (0) or not (1).

Since all the annotated images had a size of 112*112*3 the Data Layer was changed. The following convolution layer was kept the same but the stride was changed to 1 compared to original model. Whereas the loss3 layer was modified to have 2 outputs (classes)

Let x and y be the input images and corresponding output class labels, the objective of the training is to iteratively minimize the average loss defined as:

$$J(w) = (1/N)\ E\ L(f(w; x_i), y_i) + \lambda R(w)$$

Where N is the number of data instances (mini-batch) in every iteration. L is the loss function, f is the predicted output of the network depending on the current weights w, and R is the weight decay with the Lagrange multiplier λ. We use the Stochastic Gradient Descent (SGD), which is commonly used in deep CNNs to update the weights.



Another view of GoogLeNet's architecture.

The other parameters changed specifically for our training were:

- Learning Rate 0.00075
- Top modified convolution layer learning rate 5
- Loss3 layer learning rate 3
- Step size 1000
- Gamma 0.9
- Max Iteration 8000
- Batch Size 24
- Shuffle True
- Crop False
- Test Iteration 1000
- Test Size 100

Database Creation

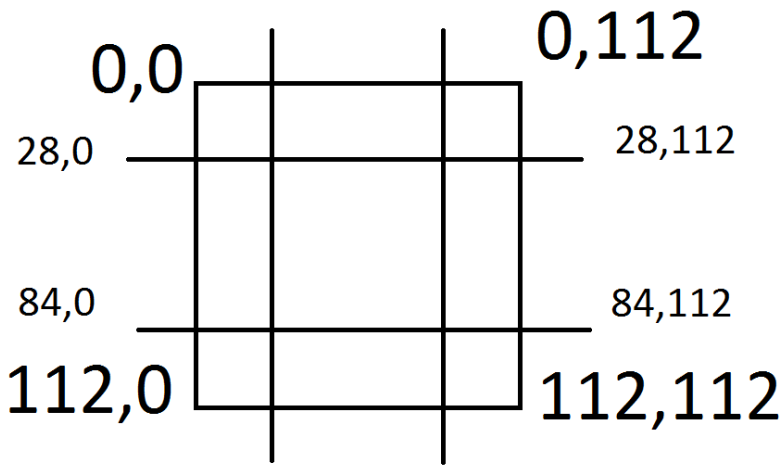
For our problem at hand we had no dataset to begin. So, we constructed a new a dataset for our problem. We annotated manually around 2500+ images.

To download such images, we scrapped http://www.imfdb.org/wiki/Main_Page. We wrote a web scrapping script to download 100,000+ pistol/compact/handgun/revolver images. To start with we selected the Berretta handgun class and only annotated images from that subset. Simultaneously we also mixed match some revolvers and compact pistols in our dataset.

We also annotated form the video dataset present at this link http://kt.agh.edu.pl/~grega/guns/

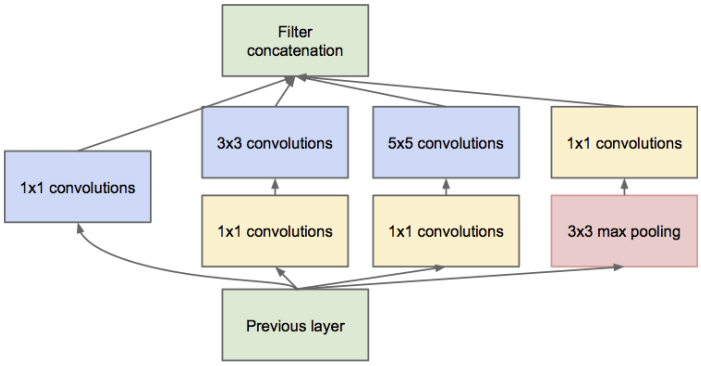
All annotated images were scaled to 112*112. For fine training, all dataset combined was small 2500+. So, we decided to bump the size of the dataset by data augmentation.

Randomly taking an annotated image we introduced rotation and skewness into it by projecting into a different plane. Simultaneously we also mirrored the images on the y-axis.



A general 20 to 30-degree rotation of the image was done, to add skewness we selected 4 random points in the top-left top-right bottom-left and bottom-right areas as shown in the pic above. After that we projected the image back to 112*112. This added random projection into the image making it non-linear for the neural net to overfit. On an approximation, each image was augmented to 20+ images.

We also decided on the choice of passing an augmented image only once through the network to prevent overfitting. Our main motive to use googlenet was that since GoogleNet offers inception model it can decide itself the filter size. This is intuitive as the firearm being the same class is in different shapes and sizes; Handguns/compact/revolver/pistols.



For the negative dataset, we found a Gesture dataset. Which did a pretty good job as a negative dataset. http://www.gregrogez.net/research/egovision4health/gun-71/

Our Final Dataset:

Gun images

Negative Samples

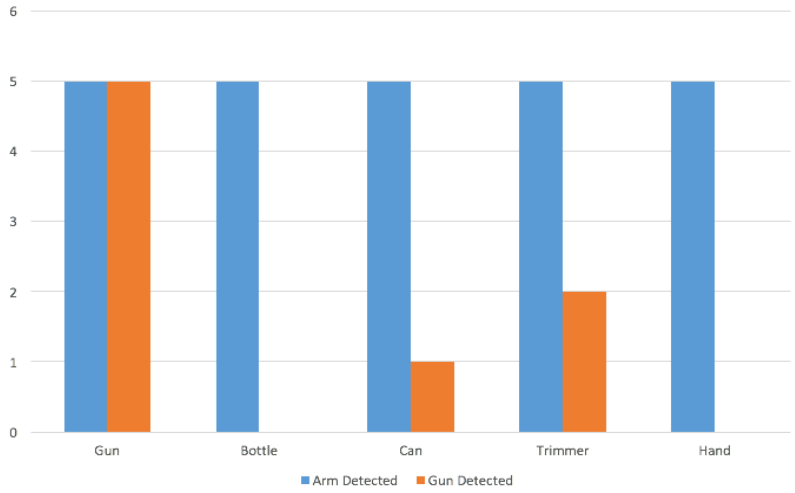
RESULTS APPROACH – 1

Left to Right: Hand, Can, Gun, Trimmer, Bottle

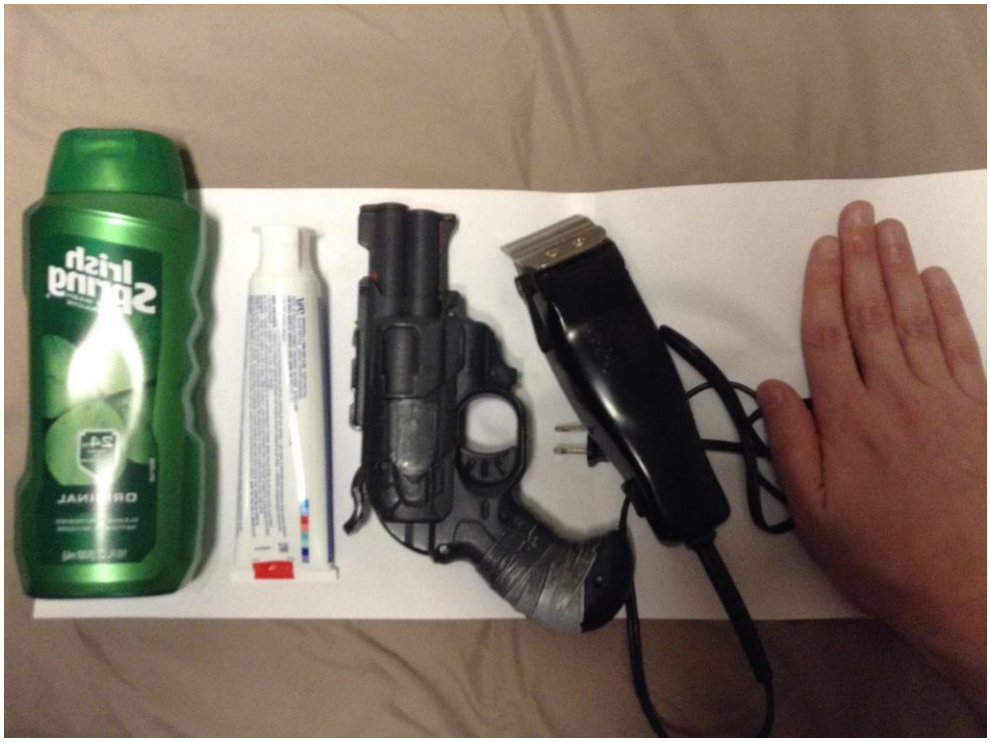
Location: Steve’s Room

Testing Images

Results - Out of Five Rounds

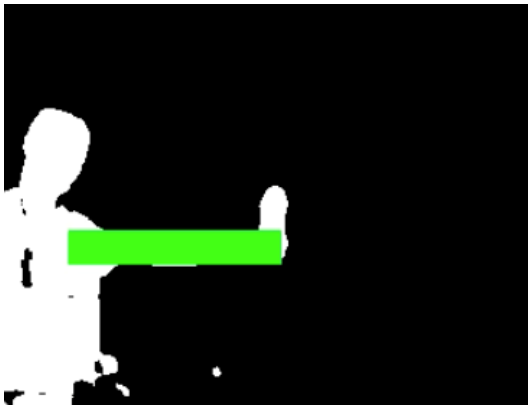


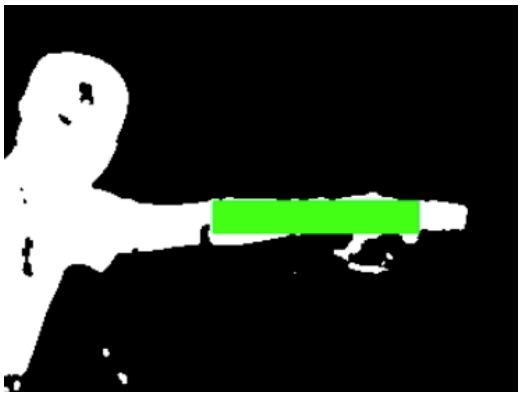
Left to Right: Shampoo, Toothpaste, Gun, Razor, Hand

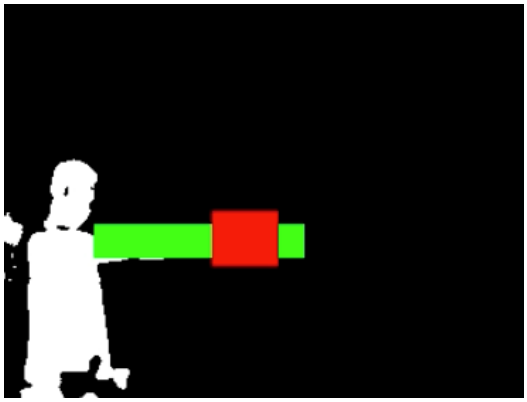


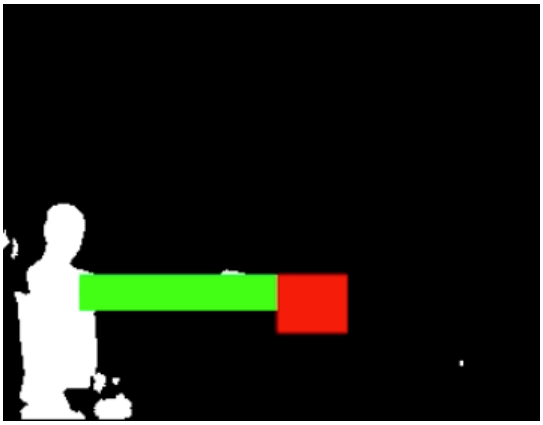
Location - Lindley Hall Mac Lab

Testing Images

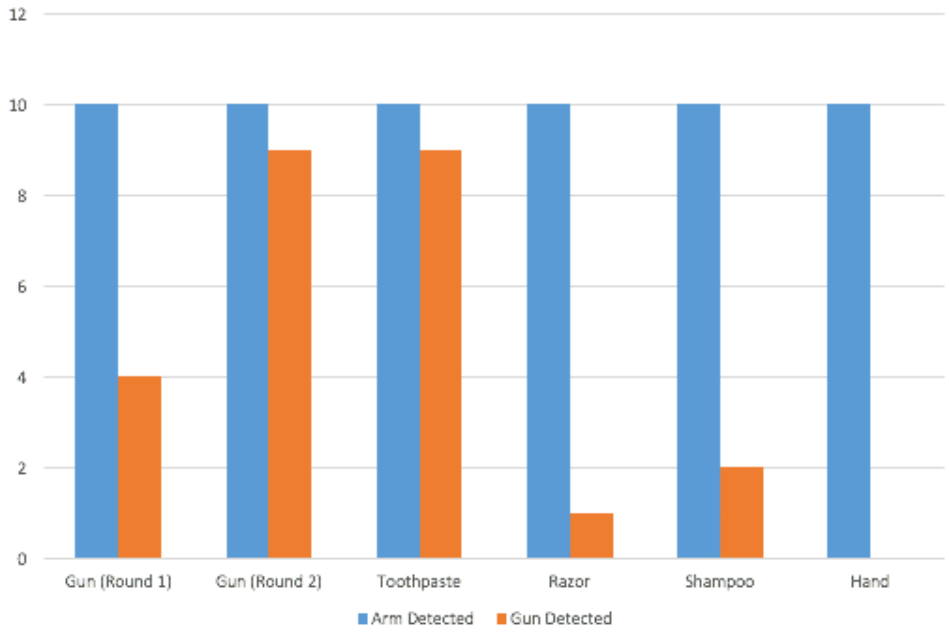




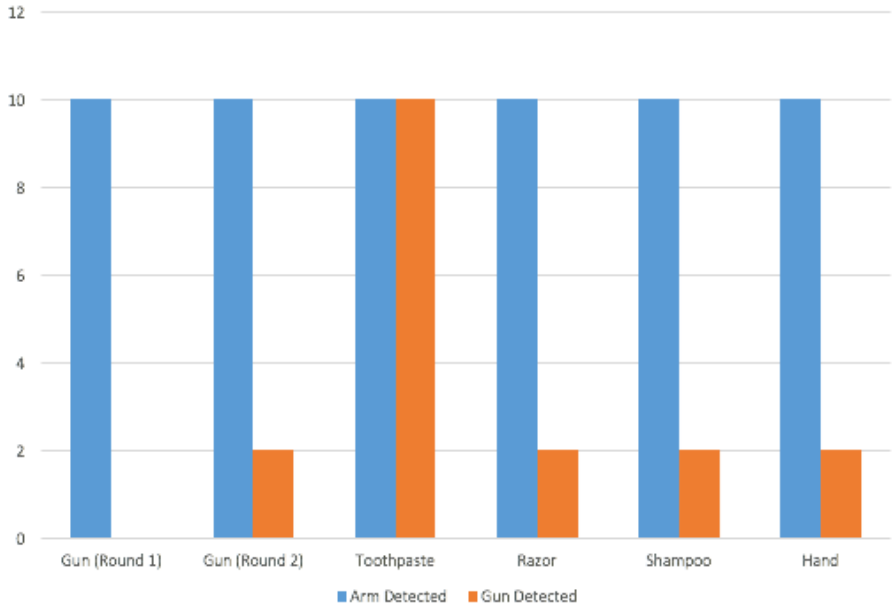




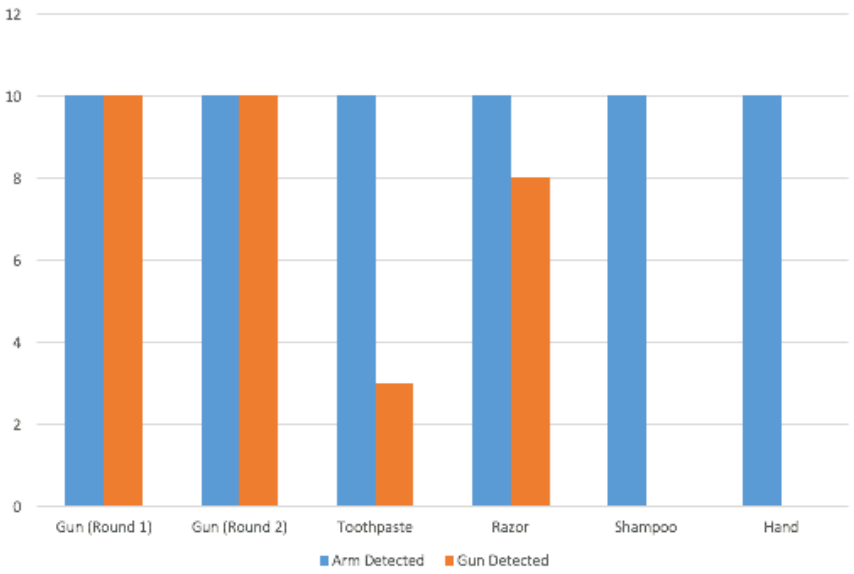
All Final Results Are Out of Ten Attempts



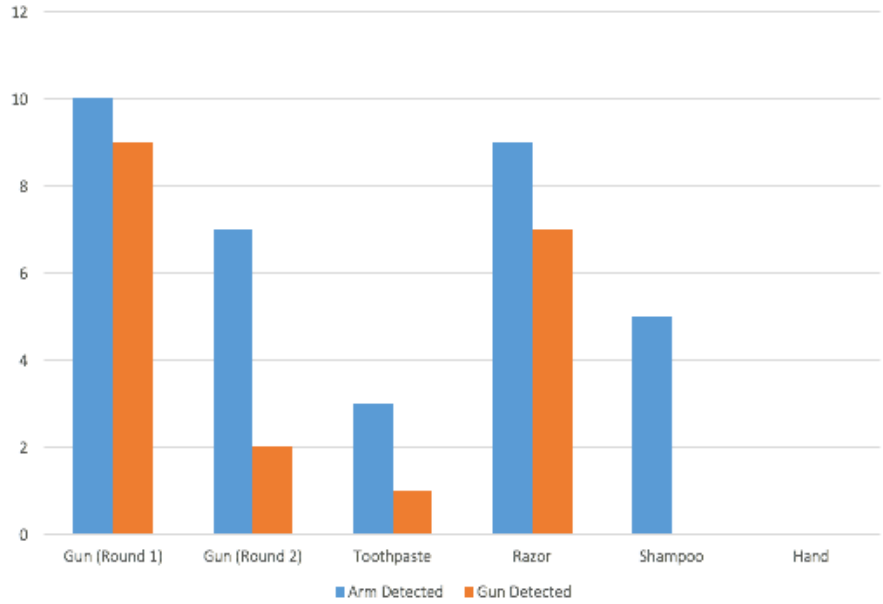
Results: 4-5 feet



Results: 6-7 feet



Results: 8-10 feet



Results: 12-15 feet

- Final Sensitivity: 4-15 feet
- Raised Arm: 214/240 = 89%
- Handgun: 46/80 = 57.5%
- Final Specificity: 4-15 feet
- Other Objects: 113/160 = 70%

Download Testing Videos

Initial Testing https://drive.google.com/open?id=0B69-xfrx3aQmSFpNcER0QWRoMUk

Final - Gun Round 2 https://drive.google.com/open?id=0B69-xfrx3aQmZWxOSm5KYmN2OXM

Final - Gun Round 1 https://drive.google.com/open?id=0B69-xfrx3aQmVjZuT0FMWjZZTVk

Final - Hand https://drive.google.com/open?id=0B69-xfrx3aQmR2ZtWHJndGVRanc

Final - Toothpaste https://drive.google.com/open?id=0B69-xfrx3aQmNFI2YXVucTIxS1k

Final - Razor https://drive.google.com/open?id=0B69-xfrx3aQmeUVkOGY1cVNDaVE

Final - Shampoo https://drive.google.com/open?id=0B69-xfrx3aQmbnVSeU5fS0MzSVk

GunVision Code

GunVision Code https://github.com/thugsatbay/GunVision

RESULTS APPROACH – 2

Training for 8000 iterations on a batch size of 24 we trained on approximately 190,000 images where 45,000+ (1300 original images) were of the positive class. We kept around original 100+ annotated images completely unseen images just for testing (6000+ augmented images). While training we also had maintained a separate 15000+ images for validation. Because of validation

we found out that at around 7000 iterations of the googlenet made the loss stagnant and there was no point in further training.

Our results are as follows:

$$precision = \frac{True\ Positives}{True\ Positives + False\ Positives},$$

$$recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

- Precision – 98.44%
- Recall – 99.01%

DISCUSSION

The initial test results were very promising, and we believe this occurred because the location coincided with the place most of the code was written. Steve's room also has a rather confined space, so varying distances were hard to simulate. After the initial tests, the templates were further refined, and thresholds for gun detection increased. Expectations for final testing were high. However, moving to the lab in Lindley Hall served as a rude awakening in some regards. Initially, virtually no guns were being detected at all. After dropping the thresholds to identify a gun, false positives became numerous and even overtook real detections at some distances. In fairness, the test subject did hold the razor, toothpaste, trimmer, and hand in a way to mimic a gun as best as possible.

None of the smart training filters generated in Steve's room seemed very effective, and were regenerated. We believe this is because the camera was mounted generally lower than the levels the gun was raised to for initial tests, and mounted equal or higher for final tests. The

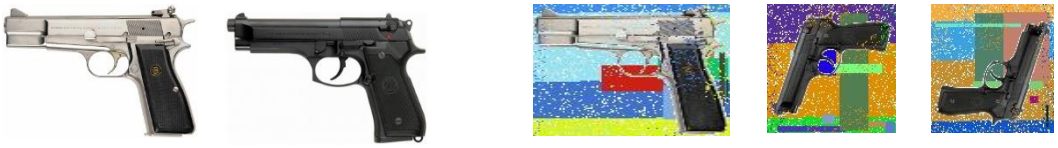
background also varied in color greatly between the two locations. Steve's room provided a mostly brown background, but the Mac lab had stark white walls and computer screens that were very dark. The gun would often blend in to the background when held in front of a screen. Also, there were several windows and even a few people studying in the background during final testing. One benefit of the final location was it had dozens of light sources. This allowed for much clearer binary maps at the 6-7 and 4-5 foot ranges. In Steve's room, there is only one light source, and standing that close to the camera often creates noisy images by interrupting it.

The conclusive results showed some flaws in the gun template matching strategy. Three templates were used: far, near, and regular. The regular template was optimized for 8-10 feet. The near template turned out to work best at the 4/5 foot range. In between these distances, the gun detection hit a dead spot. Only 2/20 guns were detected at 6-7 feet. Templates performed inconsistently at the 12-15 foot distance for both arm and gun. This proved to be the boundary at which arm detection functioned. An arm must be fully outstretched and holding an object in a way that extends its silhouette for the detector to register at this distance. Any further back, arm detection doesn't work at all with the current setup. These distances also provide much less detailed silhouettes of the gun. Nevertheless, gun detection can be accurate around 12-15 feet as demonstrated in the first round of final testing. We assume the test subject was likely standing closer to 12 feet this round. Towards 15 feet, the accuracy of both arm and gun detection plummeted. Side note - six other templates were tried where the gun was held slightly up and down at the fear, near, and regular distances. However, these seemed to only increase false positive rates, and were not included in the official testing.

For Approach-2 we are getting a high precision rate because we feel that the model has learned a good deal how a hand holding a gun looks. Though the negative class sampling is not

the perfect dataset to train against. A more accurate dataset will be hand holding different objects in the same orientation as our gun dataset.

To improve our learning methods, we tried an alternative approach where we tried making the network learn the profile images of guns and then learn on the proposed dataset [5]. We found that the network learned quickly using this approach in about 3000 iterations which required less image augmentation. Though the profile images had white background so we needed to do some augmentation to introduce color noise to the images. This prevented the network to learn the white noise.



The network learned the gun shape using the same parameters we used for proposed dataset, Image size being 112*112.

CONCLUSION

Upon conclusion, background subtraction and template matching can provide a clever idea of where a gun is most likely to be in a scene which fits our assumptions. However, the final testing showed these techniques alone cannot accurately determine whether there truly exists a gun at this specific location in an image. It may seem possible to obtain satisfactory results if enough time is spent configuring parameters to an exact camera height, distance, and background. Though these tailored parameters become of little use when placed in an unfamiliar environment. We believe this work could be useful in providing input to a more accurate gun detection algorithm. We have shown a deep neural network approach that can decide hand held

firearms from other handheld objects with a high degree of certainty. Another possibility may be to replace the smart template generated from background subtraction with templates generated from a Sobel filter/edge detection. Perhaps SIFT could be involved in some regard, and still function in real time. We believe the idea of going from background subtraction to detecting a raised arm is novel, and could benefit others working on this problem. This strategy is very computationally efficient, makes sense, and helps reduce false positives. Also, a narrow problem definition and prominent level of assumptions are different from most others' approach. Some may scoff at this philosophy, but we see it as still being very applicable to real world situations.

With respect to the CNN further improvements can be a better negative class dataset. And breaking negative class into 2 parts. Hand holding various objects and human body parts (torso, arm). The reason being when trying to find guns in an image. The negative regions will be of human body parts and other objects.

In general, we believe the idea of a smart security camera could very valuable to our modern society. Currently, a verbal description of an attacker is typically given to emergency responders/police. This can lead to false arrests and unneeded confrontations between law enforcement and civilians who happen to fit the given description. Our society currently has all the technology to make verbal descriptions of someone robbing a bank or convenience store obsolete. However, we simply are not using it in that regard. Hopefully, this project can inspire more computer scientists, community leaders, and law enforcement to believe emergency response can be guided by computer vision.

REFERENCES

[1] Grega, Michal, Matiolanski, Andrzej, Guzik, Piotr and Leszczuk Mikolaj. Automated Detection of Firearms and Knives in a CCTV Image. Multidisciplinary Digital Publishing Institute - Sensors. Published 1 January 2016.

[2] Dalal, Navneet and Triggs, Bill. Histograms of Oriented Gradients for Human Detection. Computer Vision and Pattern Recognition. Published 20 June 2005.

[3] Grega, M., Lach, S., Sieradzki, R. Automated recognition of firearms in surveillance video. In Proceedings of the 2013 IEEE International Multi- Disciplinary Conference on Cognitive Methods in Situation Awareness and Decision Support (CogSIMA), San Diego, CA, USA, 25–28 February 2013; pp. 45–50.

[4] Chen, C.; Zhuang, Y.; Xiao, J. Silhouette representation and matching for 3D pose discrimination—A comparative study. Image Vis. Comput. 2010, 28, 654–667.

[5] Automatic Handgun Detection Alarm in Videos Using Deep Learning Roberto Olmos 1 , Siham Tabik 1 , and Francisco Herrera1,2

[6] Rohit Kumar Tiwari and Gyanendra K Verma. A computer vision based framework for visual gun detection using harris interest point detector. Procedia Computer Science, 54:703–712, 2015.

[7] Rohit Kumar Tiwari and Gyanendra K Verma. A computer vision based framework for visual gun detection using surf. In Electrical, Electronics, Signals, Communication and Optimization (EESCO), 2015 International Conference on, pages 1–5. IEEE, 2015.

[8] Nadhir Ben Halima and Osama Hosam. Bag of words based surveillance system using support vector machines. International Journal of Security and Its Applications, 10(4):331–346, 2016.