Quest Journals
Journal of Software Engineering and Simulation
Volume 8 ~ Issue 7 (2022) pp: 44-53
ISSN(Online) :2321-3795 ISSN (Print):2321-3809
www.questjournals.org

Research Paper

Object Detection for Crime Scene Evidence Analysis

Sule Sani

ABSTRACT
Computer forensics is quickly becoming used for many different areas of criminal investigations. The accurate recording of crime scene details is crucial to providing investigators with information which they may will assist in reconstructing the scene. The key aspect of analyzing visual evidence from crime scenes is detecting instances of semantic objects of a certain class in digital images and videos using Machine Vision. Object detection is the key module in most visual-based surveillance applications and security systems. However, due to the presence of a large volume of data, the task of detecting objects of interest is very tedious for law enforcement agencies. Therefore, this research work presents an object detection model based on Convolutional Neural Network to detect objects in a crime scene without external control. The deep learning model in this research is trained on the Microsoft Common Objects in Context Dataset of comprising of over 70 classes of objects.This system can achieve the test accuracy of 90 % for the datasets we have that is verymuch competitive with other systems for this particular task.

***R**eceived 01 July, 2022; **R**evised 10 July, 2022; **A**ccepted 12 July, 2022 © The author(s) 2022.*
P**ublished with open access at **www.questjournals.org

3.3 Object Detection Using Convolutional Neural Network

We retrain on a custom image dataset the state-of-the-art algorithm for real-time object detection which was presented in a 2016 research paper by(Redmon.) And his colleagues decided to name this algorithm YOLO (You Only Look Once) due to its real time object detection feature. The YOLO creators claim the convolutional neural network processes images at real-time with about 45 frames per second (much faster than the R-CNN and even the Faster R-CNN).

The YOLO algorithm divides an input image into an S X S matrix. Whereby each cell of the matrix is a combination of pixels in the image which predicts an object. Combination of pixels or a grid cell in the matrix predicts a fixed number of boundary boxes to specifically localize an object on the entire input image. For each boundary box you output 5 elements (x, y, w and h) including C.

Where:

(x, y) = coordinates for the center of the box (object localization) *w = width of the object in the image* *h = height of the object in the image*

C = class confidence score of the prediction

The major concept of the algorithm is to predict a (7, 7, 30) tensor with a CNN network having 24 convolutional layers and 2 fully connected layers. It predicts several boundary boxes and outputs the boxes with a confidence score greater than 0.25 as the final prediction (class confidence score).

Class confidence score = box confidence score x conditional class probability

However, the algorithm still has losses (errors) when selecting the boundary box with a class confidence score greater than 0.25 is calculated by the loss function.

The Yolo algorithm calculates the loss function as a sum-squared of three errors for each grid cell prediction which includes:

Classification loss: Classification loss of an object detected in the input image is a squared sum error of the class conditional probabilities for each class.

Prediction accuracy of the algorithm is amazing as it uses one network for a prediction, however it can be retrained for layer to layer to improve its accuracy.

Yolo has various versions with different levels of accuracy when trained and tested on large datasets like COCO, Image Net and so on. However, YOLO was deployed for this research because of its highly optimized architecture that makes it compactible with systems of low computational abilities during the training and testing phase of the model. The optimized architecture of YOLO also contributes to the feasibility of integrating the model.

3.4 Network Design of YOLO Object Detection Model

Further research was conducted resulting in the December 2016 paper "YOLO9000: Better, Faster, Stronger," by Redmond and Farhadi, both from the University of Washington, that provided a number of improvements to the YOLO detection method including the detection of over 9,000 object categories by jointly optimizing detection and classification.To understand the YOLO algorithm, first we need to understand what is actually being predicted. Ultimately, we aim to predict a class of an object and the bounding box specifying object location. Each bounding box can be described using four descriptors:

1. Center of the box (**bx, by**)
2. Width (**bw**)
3. Height (**bh**)
4. Value **c** corresponding to the class of an object

Along with that we predict a real number **pc**, which is the probability that there is an object in the bounding box.YOLO doesn't search for interested regions in the input image that could contain an object, instead it splits the image into cells, typically 19x19 grid. Each cell is then responsible for predicting K bounding boxes.

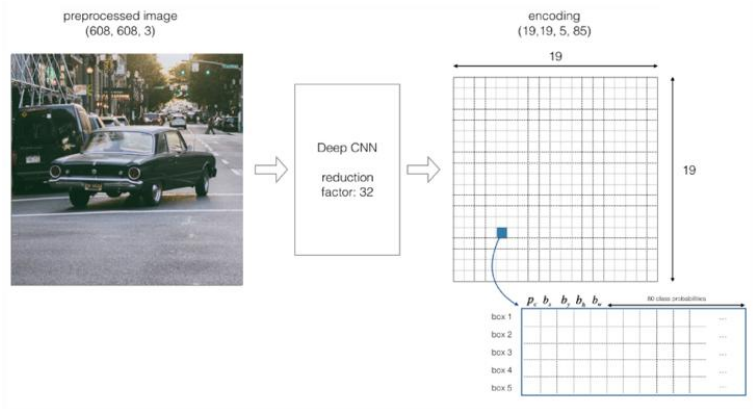

Figure 3. 1: YOLO Network Design

Here we take K=5 and predict possibility for 80 classes

An Object is considered to lie in a specific cell only if the center co-ordinates of the anchor box lie in that cell. Due to this property the center co-ordinates are always calculated relative to the cell whereas the height and width are calculated relative to the whole Image size.During the one pass of forwards propagation, YOLO determines the probability that the cell contains a certain class. The equation for the same is:

$$score_{c,i} = p_c \times c_i$$

Probability that there is an object of certain class 'c'. The class with the maximum probability is chosen and assigned to that particular grid cell. Similar process happens for all the grid cells present in the image.After computing the above class probabilities, the image may look like this:

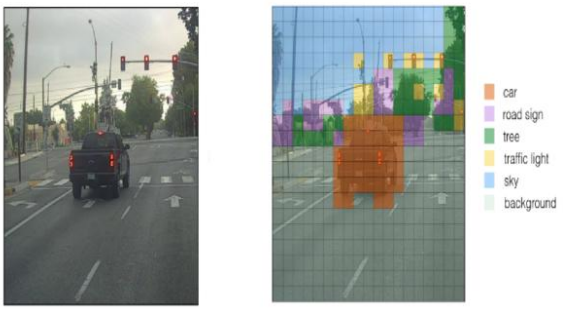

Figure 3.2: Before and after prediction

This shows the before and after of predicting the class probabilities for each grid cell. After predicting the class probabilities, the next step is Non-max suppression, it helps the algorithm to get rid of the unnecessary anchor boxes, like you can see that in the figure below, there are numerous anchor boxes calculated based on the class probabilities.

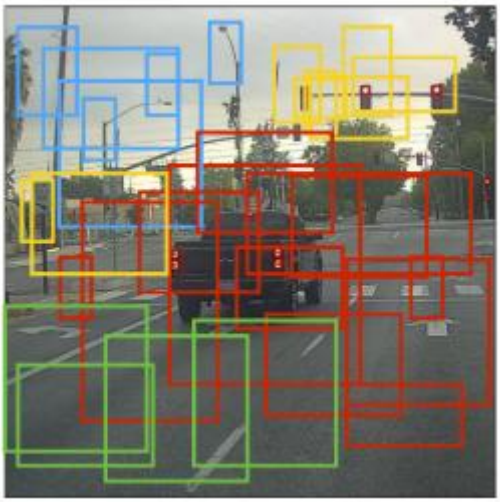

Figure 3.3: Anchor boxes

To resolve this problem Non-max suppression eliminates the bounding boxes that are very close by preforming the IoU(Intersection over Union) with the one having the highest class probability among them.

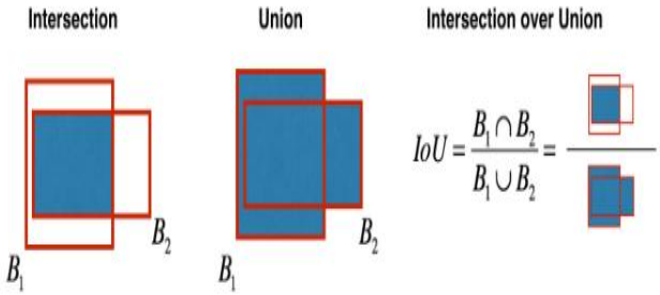

Figure 3.4: IoU operation

It calculates the value of IoU for all the bounding boxes respective to the one having the highest class probability, it then rejects the bounding boxes whose value of IoU is greater than a threshold. It signifies that those two bounding boxes are covering the same object but the other one has a low probability for the same, thus it is eliminated.
Once done, algorithm finds the bonding box with next highest class probabilities and does the same process, it is done until we are left with all the different bounding boxes.

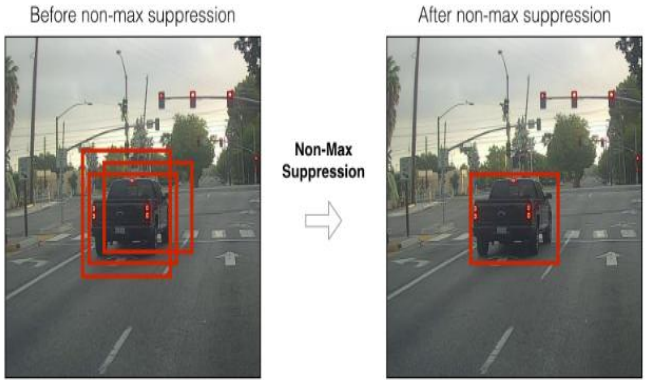

Figure 3.5: Before and After of Non-max suppression

After this, almost all of our work is done, the algorithm finally outputs the required vector showing the details of the bounding box of the respective class. The overall architecture of the algorithm can be viewed below:

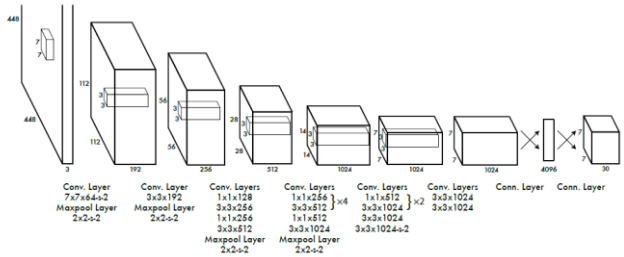

Figure 3.6: YOLO Network Architecture

YOLO Architecture, the most important parameter of the Algorithm, its Loss function is shown below. YOLO simultaneously learns about all the four parameters it predicts (discussed above).

$$\lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

So this was all about the YOLO Algorithm. We discussed all the aspects of Object detection along with the challenges we face in that domain. We then saw some of the algorithms that tried to solve some of these challenges but were failing in the most crucial one-Real time detection (speed in fps). We then studied the YOLO algorithm which outperforms all the other models in terms of the challenges faced, its fast-can work well in real-time object detection, follows a regression approach. Still improvements are being made in the algorithm. It is specifically designed to achieve an incredibly high speed of 220fps.

3.5 Dataset for the proposed system

COCO is a large-scale object detection, segmentation, and captioning dataset. COCO has several features: Object segmentation, recognition in context andSuper pixel segmentation .The Microsoft Common Objects in Context (MS COCO) dataset contains 91 common object categories with 82 of them having more than 5,000 labeled instances. In total the dataset has 2,500,000 labeled instances in 328,000 images. In contrast to the popular ImageNet dataset, COCO has fewer categories but more instances per category. This can aid in learning detailed object models capable of precise 2D localization. The dataset is also significantly larger in number of instances per category than the PASCAL VOC and SUN datasets. Additionally, a critical distinction between our dataset and others is the number of labeled instances per image which may aid in learning contextual information overall.

3.5.1 Data Pre-processing and Training

Data Preprocessing is a technique that is used to convert the raw data into a clean data set. In other words, whenever the data is gathered from different sources it is collected in raw format which is not feasible for the analysis.Pre-processing refers to the transformations applied to our data before feeding it to the algorithm. The data set can be further used as the training data. The training data is used to make sure the machine recognizes patterns in the data, the cross-validation data is used to ensure better accuracy and efficiency of the algorithm used to train the machine. For this research we are you using pre-trained waits from YOLO. The result of the preprocessing is an extremely fast and accurate model with MAP measured at .5 IOU .YOLO is on par with Focal Loss but about 4x faster.

3.6 Methodology and Work Procedure

The methodology describes the method adopted to achieve the stated objectives of the proposed system. The selected methodology for this project is the Cross Industry Standard Process for Data Mining due to its sequential and iterative approach to problem solving applying data science and machine learning algorithms which is relevant to activities carried out in this research. We systematically employ the scientific methods identified with this methodology.

Figure 3.2 shows the sequential process of the methodology.

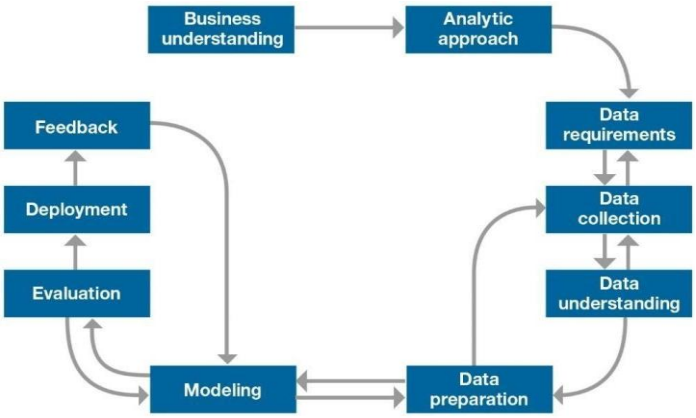

Figure 3.7 Cross Industry Process for Data Mining Methodology(Aggarwal)

The methodology is referred to as the standardized approach for data science projects due to its systematic development and delivery process. The system development process is considered effective as the entire system, as the entire system is divided into four modules. The various stages of the methodology include the following as shown in Figure 3.7:

Business Understanding: Here we analytically understand the problem at hand during crime evidence analysis which can be improved by accurately detecting evidence related objects found at a crime scene.

Analytic Approach: Having clearly stated the problem we seek to solve we select an algorithm suitable for a concise solution. The required solution for the problem involve object detection through image classification, thereby we center our research on developing and training convolutional neural network for object detection from digital forensic images using the You Only Look Once (Yolo) algorithm.

Data Requirements: After an algorithm is selected the required format of data to train and test an effective model with YOLOis images with the ".jpg" extension.

Data Collection: This phase is where we collect image data to prepare our dataset. We use the Google Advance Search tool to download the MS COCO dataset.

Data Understanding: Here we try to understand to collected dataset of 5 class of objects to know how to process them for training and testing the object detection model.

Data Preparation: At this phase we try to prepare the dataset by creating annotations for every image collected in the dataset. Annotations are corresponding ".xml" files which gives a detailed description of corresponding images in the dataset, details like the position of the desired object in the image using boundary boxes.

Modelling: This is where we develop the object detection model by training with the training set of images collected in the dataset.

Evaluation: At this phase we test the accuracy of the model having trained with the training set.

Deployment: The deployment phase is where we utilize the trained model on a Django web app developed in the research for crime scene evidence analysis and documentation.

Feedback: Having implemented the model on a Django web application we get feedback from users in the domain of study to access the significance of our achieved result.

Justification of this methodology for the research work includes:

I. Providing a guiding strategy for the development process
II. Improving the reusability of software modules
III. Changing to the software can be easily implemented
IV. Easy integration of machine learning model on the system for deployment.

3.7 Tools

Anaconda Environment

Anaconda is a free and open-sourcedistribution of the Python and R programming languages for scientific computing (data science, machine learning applications, large-scale data processing, predictive

analytics, etc.), that aims to simplify package management and deployment. Package versions are managed by the package management system conda. Anaconda distribution comes with more than 1,500 packages as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator, as a graphical alternative to the command line interface (CLI).

Open CV Python

OpenCV (Open source computer vision) is a library of programming functions mainly aimed at real-time computer vision. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface. There are bindings in Python, Java and MATLAB/OCTAVE. The API for these interfaces can be found in the online documentation.

Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Built by experienced developers, it takes care of much of the hassle of Web development, so you can focus on writing your app without needing to reinvent the wheel. It's free and open source. Django was designed to help developers take applications from concept to completion as quickly as possible. Django takes security seriously and helps developers avoid many common security mistakes. Some of the busiest sites on the Web leverage Django's ability to quickly and flexibly scale.

EXPERIMENTS AND RESULT

In this section, we describe the datasets, the experimental setup and the results achieved.

4.1 System Requirement

In order to achieve the stated objectives of the system, some requirements are necessary. These requirements can be classified into two (2) categories: hardware and Software Requirements.

4.1.1 Hardware Requirement

Table 1: Hardware Requirement for the system

SN	Minimum Requirements
1.	Intel Pentium 2.0GHZ Core i5 Processor or higher
2.	Minimum of 4GB RAM
3.	Ethernet Wireless Card
4.	250 GB Hard Disk Drive or 120 GB Solid State Drive
5.	PS4 Controller / Xbox Controller
6.	USB 2.0 / USB 3.0 Connection Port and cable

4.1.2 Software Requirement

Table 2: Software Requirement for the system

SN	Requirements	Software
1.	Development Environment	Anaconda Python IDE and Visual Studio Code Text Editor
2.	Web Tools and Technologies	Django Web Development Framework and Web Brower (Google Chrome / Mozilla Firefox)
3.	Scripting Languages	Python, and JavaScript
4.	Machine Learning	Yolo Weights , Cfg File and Darknet
6.	Libraries and Dependencies	Pillow,gunicorn,WhiteNoise,Keras,Numpy,Matploit, OpenCV python, Imageio, Cloudinary,Tkinter, pip,Conda, Libboost-python and H264DECODER library
7.	Operating System	Windows 7, Windows 8, Windows 8.1, Windows 10 , Ubuntu 15 LTS

4.2 System Testing

This involves qualitative evaluation of the new system with respect to the user requirements gathered and system specification. This process ensures the system is error free, produces expected results, highly secure operability, efficient and reliable. System testing involves the software validation and verification, the process is carried out in four phases: unit testing, integration testing, system testing and acceptance testing.

Unit testing: the system is divided into distinct manageable units. Each of these units are tested separately with different parameters to ensure error free modules of the software.

Integration Testing: having tested the system units for correctness effective intercommunication of these units is what this phase of the software testing handles. It ensures that each unit of system communicates with other units as specified. At this phase, the system units are fused together as a single entity.
System testing: at these phases the system is compiled as a product and tested as a single product. This can be done by functionality testing, performance testing and security testing.
Acceptance: having tested the compiled system as a single product, the product is ready to be handed over to the user. At this phase, the user evaluates the system against user requirements.

4.3 Dataset

Our network is pre-trained with the MS-COCO dataset. This is an image segmentation, recognition and captioning dataset by Microsoft Corporation, which is a collection of more than 300,000 images and 80 object categories with multiple objects per image. Then, we tested the method in two different test-sets: a subset of images containing 12 indoor objects extracted from ImageNet, which we called it ImageNet-RoomObjects, and the Karina dataset. We briefly describe the test-sets along with the datasets.
Test-Set: ImageNet-Room Objects: This is a collection of 1345 images with 12 object categories that are commonly found in an indoor environment, i.e. bedroom. We randomly selected images from the ImageNet, which is a huge dataset with a collection of 14,197,122 images with more than 1000 object classes, and each object class in this dataset contains thousands of images.
Karina Dataset: This is a video dataset that was created to evaluate object recognition in environments which are similar to those that might appear in child pornography. The dataset contains 16 videos of 3min which are filmed in 7 different rooms, and it contains 40 different categories, which represents to some of the objects that can be found most commonly in an indoor environment i.e. a bedroom.

Figure 4.1 object detection results on ImageNet-Indoor Objects.

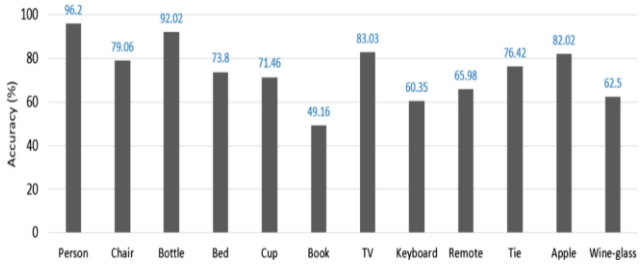

Figure 4.2 Detection accuracy (in percentage) of each class.

4.4 Experimental Setup

We have used the pre-trained network architecture of YOLO to detect objects in our created test-set and the Karina dataset. The network was trained based on the following parameters: base learning rate: 0.001, learning policy: step, gamma: 0.1, momentum: 0.9, weight decay: 0.0005 and iterations 490000. All the experiments were carried out using the Caffe deep learning framework in NVidia Titan X GPU and in an Intel Xeon machine with 128 GB RAM.

4.5 Detection Accuracy

We present the detection accuracy (in percentage) of each class in the ImageNetRoomObjects test-set, which is the percentage of true positives in each class of the test-set. Figure 4.3shows

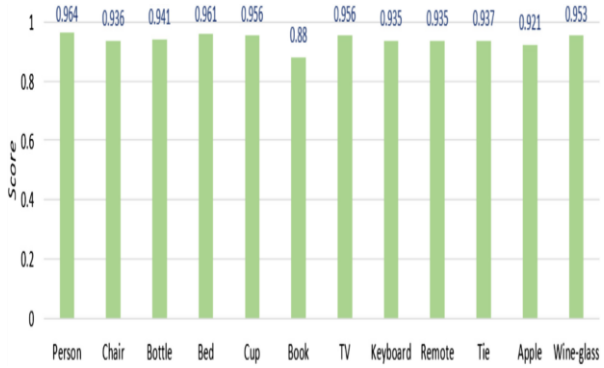

Figure 4.3the detection accuracies for all the 12 considered classes

We obtained an average Accuracy of 74.33%. We also present the mean of confidence scores for each class. Figure5 shows the samples of object detection in each of the class category.

Table 3. Detection time (in seconds) for each object class along with the number of images per class.

Class	Person	Chair	Bottle	Bed	Cup	Book	TV	Keyboard	Remote	Tie	Apple	Wine-glass
Time	31.35	26.8	37.02	35.6	21.65	21.65	27.22	31.62	24.74	28.27	27.94	34.53
Total images	104	100	110	129	121	120	130	110	105	114	106	105

4.4 Detection Time

This is the amount of time taken to detect specific objects in an image. Table 3 shows the time taken to detect objects specific to each of the class. For example, there are 104 images of person in the test-set, and the time needed to detect a person in those images was 31.35s. The system takes 356.35s in a GPU platform to detect objects in all the 1345 images, which also includes the time to propose regions. We also observe that, it takes an average of 2s to detect objects in a single image in a CPU environment, and 0.12s in a GPU environment, saving approximately 90% of the needed time.

4.5 Experiments on the Karina Dataset

To test the system, we have created a test-set from the Karina dataset by extracting image frames (size 640 × 480) from the videos. Out of 40 object categories we select 6 classes i.e.bed, book, toy car, teddy, person and cup. In Table2 we present the detection accuracy for each class along with the total number of images present in each category. Figure8 shows some samples, where we were able to detect some indoor objects like (a) cup, (b) bed, (c) book, (d) toy car, (e) remote, (f) doll and (g) teddy bear. Since the resolution of the images is low, we performed only preliminary test. In the future, we would like to apply image super-resolution techniques to enhance the resolution before detection.

Table 4. Detection accuracy for each object class in the Karina dataset.

Class	Person	Book	Toy car	Teddy bear	Cup	Bed
Accuracy	81.3%	28.42%	5.4%	18.2%	45.61%	20.23 %
Total images	100	120	105	80	109	91

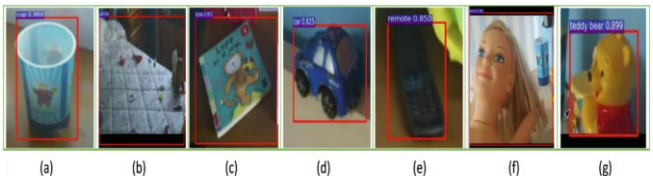

Figure 4.4 Examples of object detection results on the Karina dataset. Red squares overlaid on input images mark the bounding boxes of the detected objects.

4.6 Discussion

We have evaluated our object detection system in ImageNet-Room Objects and the Karina Dataset, and we made several observations and conclusions based on our experiments.First of all, our object detection system is pre-trained on the MS-COCO dataset. While testing on the ImageNet-Room Objects, we observed that the system was able to detect objects with an average accuracy of 74.33%, where the highest accuracy was 96.2% obtained with the class person, and the lowest was 49.16% yielded with books. However, this performance can be further improved by fine tuning the architecture using datasets with respect to each class category, and we will address it in our future work.We have also evaluated the performance of the system in the Karina dataset. But, due to the presence of low resolution images, the average accuracy obtained was 33.19%. In real world scenarios, we cannot expect all the images to be of good resolution, and it is a difficult task for the forensic department to recreate crime scenes using such images. In the future, we will handle this issue by applying image-super resolution techniques to enhance the image quality, which will ease the task for police officers to detect objects effectively even in low quality images.

The average detection time per image was 0.12s in NVidia Titan X GPU, which makes the system suitable to be used as a real-time application. Furthermore, the system might be an application for a forensic department, which can help police officers to automatically detect objects of interest from large scale datasets in real-time

4.7 Performance Evaluation

Table 5: Performance Evaluation

Metrics	Existing System	Proposed System
Reliability	Low	High
Speed	Medium	High
Efficiency	Medium	High
Security	Low	Medium
Accuracy	Low	High

SUMMARY, RECOMMENDATION AND CONCLUSION

5.1 Summary

Crime evidence analysis is an essential activity carried out during crime event reconstruction. Forensic analysis physical crime scenes involve analysis of visual documents like images and videos containing object found at a crime scene. Manual analysis of these objects in visual documentations can be very tedious resulting to erroneous results when carried out with the nude eye. To solve this, machine learning algorithms have played major roles in this research area but the need for a real time and faster object detection system is required to improve achieved results thus far.

As an improvement, a research is conducted using the convolutional neural network algorithm for object detection of already captured images. This research retrain the state of the art You Only Look Once (YOLO) object detection algorithm on a custom dataset of 80 class of objects.

.The machine learning object detection model trained and tested in this research is envisaged to be used on the e camera for object detection of images taken at the crime scene.

5.2 Recommendations

The object detection model is highly recommended for integration on digital forensic systems like video surveillance system, forensic analysis cameras and so on. The research goes further to create web app for forensic case documentation which is recommended for crime evidence analysis on physical crime scenes.

5.3 Conclusion

In this work, we presented a real-time system which can detect objects related to indoor environments (i.e. bedroom). We have used the state-of-the-art network architecture of YOLO algorithm, which can compute region proposal within the network itself. Due to this property, the algorithm can be widely used to develop real-time object detection applications. To evaluate the system, we have created a test-set "ImageNet-RoomObjects" comprising of images commonly found in an indoor environment, and we achieved state-ofthe-art accuracy. The system has also been tested on the Karina dataset, but we have achieved poor accuracy due to the low quality of the images. In future works, we will address this issue by applying image super-resolution techniques, and we will train a new model containing a large number of categories based on the additional object types that the police might found interesting during their crime scene research. Finally, the method can be used as a surveillance application to detect objects of interest in videos and images in real time for analyzing various crime scenes.

REFERENCE

[1]. Aggarwal, M. Cross-Industry Process for Data Mining. 2017, https://medium.com/@thecodingcookie/cross-industry-process-for-data-mining-286c407132d0.

[2]. Bkassiny, Mario, et al. "A Survey on Machine-Learning Techniques in Cognitive Radios." IEEE Communications Surveys and Tutorials, 2013, doi:10.1109/SURV.2012.100412.00017.

[3]. Bostanci, Erkan. 3D Reconstruction of Crime Scenes and Design Considerations for an Interactive Investigation Tool. no. December 2015, 2015, http://arxiv.org/abs/1512.03156.

[4]. Brian Carrier. "Defining Digital Forensic Examination and Analysis Tools Using Abstraction Layers Brian." International Journal of Digital Evidence, vol. 1, no. 4, 2003, pp. 1–30, doi:10.1017/CBO9781107415324.004.

[5]. Buczak, Anna L., and Erhan Guven. "A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection." IEEE Communications Surveys and Tutorials, 2016, doi:10.1109/COMST.2015.2494502.

[6]. Dai, Qieyun, and Derek Hoiem. "Learning to Localize Detected Objects." Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2012, pp. 3322–29, doi:10.1109/CVPR.2012.6248070.

[7]. Duce, D. A., et al. The Use of Artificial Intelligence in Digital Forensics : An. 2010, pp. 35–41.

[8]. Felzenszwalb, Pedro, et al. "A Discriminatively Trained, Multiscale, Deformable Part Model." 26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR, 2008, doi:10.1109/CVPR.2008.4587597.

[9]. Forsyth, David. "Object Detection with Discriminatively Trained Part-Based Models." Computer, 2014, doi:10.1109/MC.2014.42.

[10]. Gkioxari, Georgia, et al. "Contextual Action Recognition with R∗CNN." Proceedings of the IEEE International Conference on Computer Vision, 2015, doi:10.1109/ICCV.2015.129.

[11]. Hinton, Geoffrey E., et al. "A Fast Learning Algorithm for Deep Belief Nets." Neural Computation, 2006, doi:10.1162/neco.2006.18.7.1527.

[12]. Homem, Irvin. "Towards Automation in Digital Investigations: Seeking Efficiency in Digital Forensics in Mobile and Cloud Environments." Diss. Department of Computer and Systems Sciences, Stockholm University, 2016.

[13]. Huang, Jonathan, et al. "Speed/Accuracy Trade-Offs for Modern Convolutional Object Detectors." Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, 2017, doi:10.1109/CVPR.2017.351.

[14]. Karpathy, Andrej. "CS231n Convolutional Neural Networks for Visual Recognition." Stanford University, 2016, pp. 1–22, http://cs231n.github.io/.

[15]. LeCun, Yann, et al. "Convolutional Networks and Applications in Vision." ISCAS 2010 - 2010 IEEE International Symposium on Circuits and Systems: Nano-Bio Circuit Fabrics and Systems, 2010, doi:10.1109/ISCAS.2010.5537907.

[16]. Luo, Bin, et al. "Advances in Brain-Inspired Cognitive Systems." Cognitive Computation, vol. 8, no. 5, Springer International Publishing, 2016, doi:10.1007/s12559-016-9431-7.

[17]. Marra, Francesco, et al. "Counter-Forensics in Machine Learning Based Forgery Detection." Media Watermarking, Security, and Forensics 2015, 2015, doi:10.1117/12.2182173.

[18]. Marsland, Stephen. "Machine Learning: An Algorithmic Perspective." Machine Learning: An Algorithmic Perspective, Second Edition, 2014, doi:10.1201/b17476.

[19]. Mathworks. What Is Deep Learning. 2016, https://www.mathworks.com/discovery/deep-learning.html.

[20]. McCulloch, Warren S., and Walter Pitts. "A