Machine Learning Model Powered Shoplifting Detection For Shops, Airports, BookStores Final Report

BERAT BERKAY ERKEN
191180758

GÖKAY DİNDAR
181180024

ABSTRACT

According to U.S. Chamber of Commerce's survey the rate of the shoplifting increased over the past year. Shoplifting is at serious retail crime for all retailers, which reduces profitability. Many shop owners tried to decrease shoplifting rates by using CCTV cameras however it is not a complete effective solution to highlight shoplifting activities. We are suggest that using machine learning model to find shoplifter and his actions.

1. INTRODUCTION

Object Discovery is feting an case of object classes over a wide range of image data using computational ways or the raw eye. Object discovery and image processing have been a frequent exploration over the times due to their numer- ous practical operations. Increasing crime rate became the main subject of the project. Thanks to the YOLOv3(You Look Only Once) algorithm, it was decided whether an event was criminal(shoplifting) or normal by object recognition and object tracking.

2. MOTIVATION

In most retail stores nowadays, there are clear signs that state that shoplifters will be prosecuted and that the shop is monitored with cameras. Yet, despite these anti-theft measures, billions of potential profits are lost each year due to shoplifting. our time. To make shop's profit high we decided to decrease the number of stolen goods.

3. DATASET

Kaggle dataset of UCF Crime. The dataset contains images extracted from every video from the UCF Crime Dataset. Every 10th frame is extracted from each full-length video and combined for every video in that class. All the images are of size 64*64 and in .png format.

The dataset has a total of 14 Classes :

1.Abuse 2. Arrest 3. Arson 4. Assault 5. Burglary 6. Explosion 7. Fighting 8. Normal Videos 9. RoadAccidents 10. Robbery 11.Shooting 12.Shoplifting 13.Stealing 14. Vandalis

The total image count for the train subset is 1,266,345.

The total image count for the test subset is 111,308.

Kaggle dataset of yolo-coco.

This is ready to use data with weights and configuration along with coco names to detect objects with YOLO algorithm.

80 names of objects (labels) that can be Detected on the image.

(person bicycle car motorbike aeroplane bus train truck boat traffic light fire hydrant stop sign parking meter bench bird cat dog horse sheep cow elephant bear zebra giraffe backpack umbrella handbag tie suitcase frisbee skis snowboard sports ball kite baseball bat baseball glove skateboard surfboard tennis racket bottle wine glass cup fork knife spoon bowl banana apple sandwich orange broccoli carrot hot dog pizza donut cake)

4. MATHEMATICAL METHOD

Convolutional neural networks are very good at picking up on patterns in the input image, such as lines, gradients, circles, or even eyes and faces. It is this property that makes convolutional neural networks so powerful for computer vision. Unlike earlier computer vision algorithms, convolutional neural networks can operate directly on a raw image and do not need any preprocessing.!

The YOLOv3 algorithm first separates an image into a grid. Each grid cell predicts some number of boundary boxes (sometimes referred to as anchor boxes) around objects that score highly with the aforementioned predefined classes. Each boundary box has a respective confidence score of how accurate it assumes that prediction should be and detects only one object per bounding box. The boundary boxes are generated by clustering the dimensions of the ground truth boxes from the original dataset to find the most common shapes and sizes.

a)No More Softmax Means Multiple Belonged Class

Softmaxing classes rests on the assumption that classes are mutually exclusive, or in simple words, if an object belongs to one class, then it cannot belong to the other. This works fine in COCO dataset. However, when we have classes like Person and Women in a dataset, then the above assumption fails. This is the reason why the authors of YOLO have refrained from softmaxing the classes. Instead, each class score is predicted using logistic regression and a threshold is used to predict multiple labels for an object. Classes with scores higher than this threshold are assigned to the box.

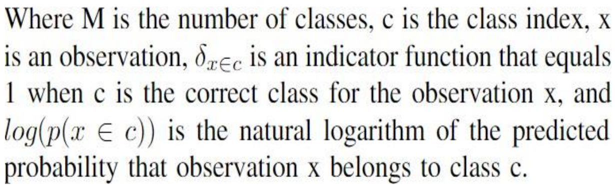
b) Convolutional Kernels

The YOLO algorithm is named “you only look once” because its prediction uses 1×1 convolutions; this means that the size of the prediction map is exactly the size of the feature map before it.

c) Loss Function

Cross-entropy for the loss function. The cross-entropy loss function is calculated as follows.

$$-\sum_{c=1}^{M} \delta_{x\in c} log(p(x \in c))$$



Where M is the number of classes, c is the class index, x is an observation, $\delta_{x\in c}$ is an indicator function that equals 1 when c is the correct class for the observation x, and $log(p(x \in c))$ is the natural logarithm of the predicted probability that observation x belongs to class c.

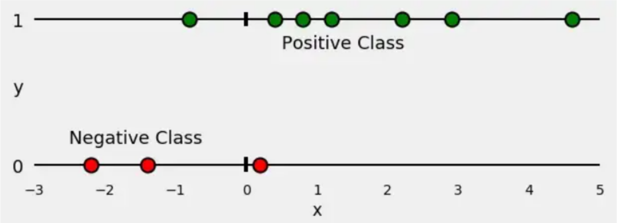
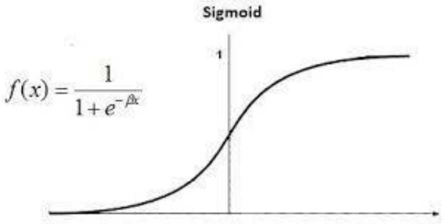


Figure 2: splitting the data!

d) Bounding Box Regression

Using logistic regression (instead of the softmax function)for predicting an objectness score for every bounding box

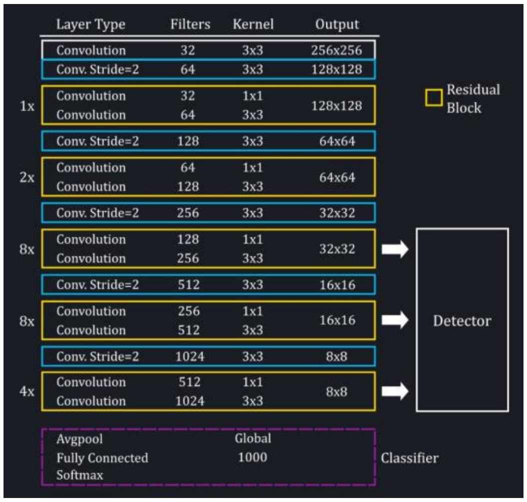
$$p(x) = \frac{1}{1 + e^{-(x-\mu)/s}}$$



5. SOFTWARE APPLICATION

To decide whether if a person stealing or not we are going to use live video streams from shop’s CCTV cameras then proceed the video frames with YOLO and pose analysis algorithms. One of the most well-liked model architectures and object identification techniques is (YOLO). The primary reason for its popularity is that it makes use of one of the greatest neural network architectures to create high accuracy and overall processing speed. For posture detection we will use Conventional Neural Networks and we will combine the results with YOLO.

YOLOv3 mainly consists of two parts: a feature extractor and a detector. The image is first fed into the feature extractor, a convolutional neural network called Darknet-53. Darknet-53 processes the image and creates feature maps at three different scales. The feature maps at each scale are then fed into separate branches of a detector. The detector’s job is to process the multiple feature maps at different scales to create grids of outputs consisting of objectness scores, bounding boxes, and class confidences.



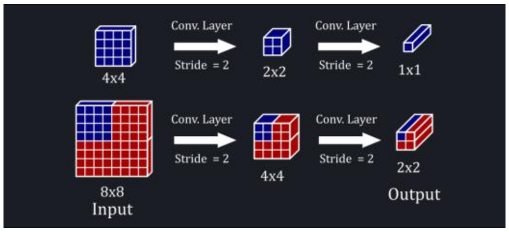
	Layer Type	Filters	Kernel	Output
	Convolution	32	3x3	256x256
	Conv. Stride=2	64	3x3	128x128
1x	Convolution	32	1x1	128x128
	Convolution	64	3x3	
	Conv. Stride=2	128	3x3	64x64
2x	Convolution	64	1x1	64x64
	Convolution	128	3x3	
	Conv. Stride=2	256	3x3	32x32
8x	Convolution	128	1x1	32x32
	Convolution	256	3x3	
	Conv. Stride=2	512	3x3	16x16
8x	Convolution	256	1x1	16x16
	Convolution	512	3x3	
	Conv. Stride=2	1024	3x3	8x8
4x	Convolution	512	1x1	8x8
	Convolution	1024	3x3	
	Avgpool		Global	
	Fully Connected		1000	
	Softmax			

The first part of YOLOv3 is the feature extractor: Darknet-53. It uses 53 convolutional layers. With so many layers, the feature extractor needs to use residual blocks to mitigate the vanishing gradient problem.

When there are more layers in the network, the value of the product of derivative decreases until at some point the partial derivative of the loss function approaches a value close to zero, and the partial derivative vanishes. We call this the vanishing gradient problem.

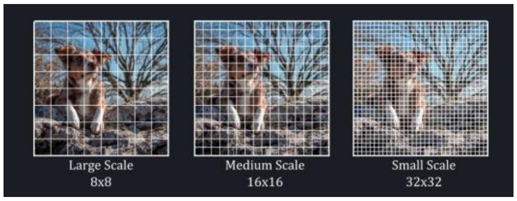
The Darknet feature extractor, therefore, mainly consists of convolutional layers grouped together in residual blocks alternating between 3x3 and 1x1 convolutional layers. Instead of using max pooling or average pooling layers to downsample the feature map, Darknet-53 uses convolutional layers with strides of 2 which skips every other output to halve the resolution of the feature map. Additionally, batch normalization is used throughout the network to stabilize and speed up training while also regularizing the model.

The end portion of the network is only used when YOLOv3 is used as a classifier rather than for object detection. For object detection, three outputs at different feature resolutions (in this case 32x32, 16x16, and 8x8) to be fed into the detector. These outputs come from the last residual block at a specific resolution which would have the most complex features. These three outputs allow for multi-scale detection.



YOLOv3 divides the input image into grid cells where if an object’s centrepoint lands on a particular grid cell, that grid cell will be responsible for detecting it with its three (or more) anchor boxes. These grid cells arise from the architecture of Darknet-53, which is illustrated in a simplified example in figure. In figure

On the top, the input is reduced from a 4x4 to a 1x1 resolution. The same operations acting on an 8x8 input creates a 2x2 output. Each component of this output then represents a grid cell of the input image.

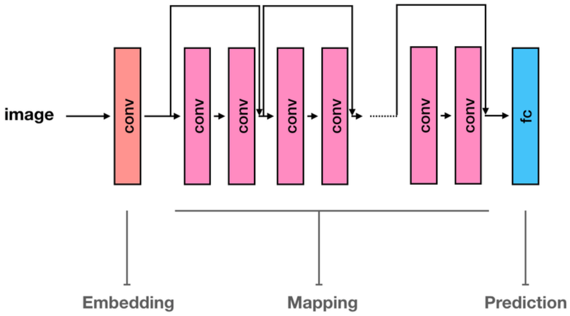


With multi-scale predictions, YOLOv3 creates three different sets of grid cells for each scale. Many of these grid cells will overlap and each scale will specialize in different object sizes. The small scale output has smaller anchor boxes to handle smaller objects while the larger grid cells use much larger anchor boxes to handle larger objects. This allows YOLOv3 to detect objects of vastly different sizes.

When we developing our model to detect crime scenes we used Resnet Architecture with sample crime or normal key tagged frames. Then we proced with resnet. ResNet (Residual Network) is a type of convolutional neural network (CNN) that is trained to perform image classification tasks. It was introduced in 2015 by researchers at Microsoft Research and has since become widely used in the field of deep learning, particularly for image classification tasks.

ResNets are designed to address the problem of vanishing gradients, which is a common issue in very deep neural networks. In a deep neural network, the gradients of the parameters with respect to the loss function can become very small, making it difficult to update the parameters and improve the model. ResNets solve this problem by introducing a shortcut connection, or a skip connection, between the layers of the network. This allows the gradients to bypass one or more layers, allowing the network to learn more effectively.

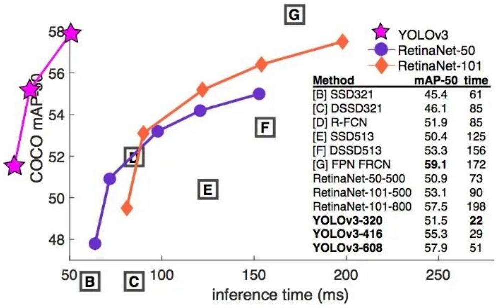
ResNets have been successful in achieving state-of-the-art results on a number of image classification benchmarks, including the ImageNet dataset. They have also been applied to other tasks such as object detection, semantic segmentation, and even machine translation.



(Resnet Skip Connections)

a)Testing

YOLO v3 performs faster at COCO50 mean average precision benchmark than others.



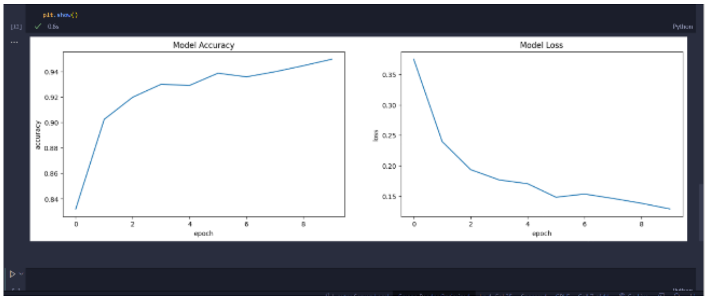
Method	mAP-50	time
[B] SSD321	45.4	61
[C] DSSD321	46.1	85
[D] R-FCN	51.9	85
[E] SSD513	50.4	125
[F] DSSD513	53.3	156
[G] FPN FRCN	**59.1**	172
RetinaNet-50-500	50.9	73
RetinaNet-101-500	53.1	90
RetinaNet-101-800	57.5	198
YOLOv3-320	51.5	**22**
YOLOv3-416	55.3	29
YOLOv3-608	57.9	51

Frame Classification With Resnet

When we plotting train model accuracy score and model loss:

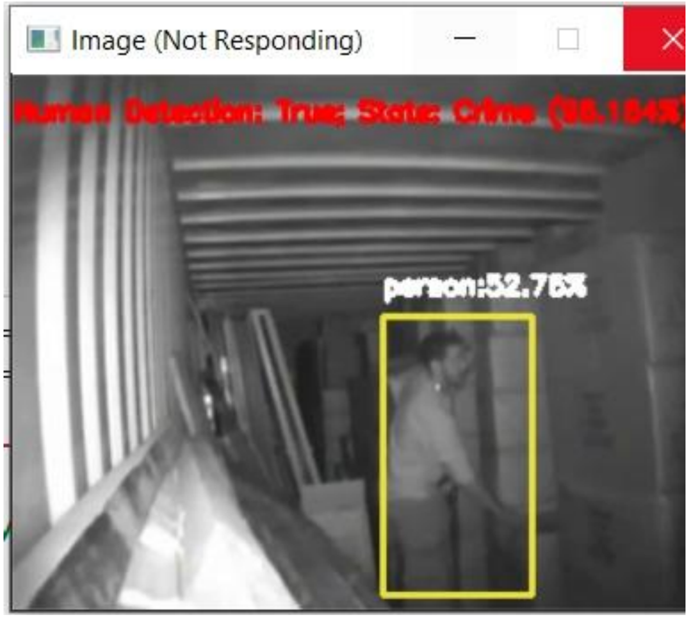
```
  Epoch 1/10
  114/114 [==============================] - 642s 6s/step
  - loss: 0.3751 - accuracy: 0.8321
  Epoch 2/10
  114/114 [==============================] - 625s 5s/step
  - loss: 0.2396 - accuracy: 0.9024
  Epoch 3/10
  114/114 [==============================] - 592s 5s/step
  - loss: 0.1935 - accuracy: 0.9197
  Epoch 4/10
  114/114 [==============================] - 569s 5s/step
  - loss: 0.1766 - accuracy: 0.9299
  Epoch 5/10
  114/114 [==============================] - 561s 5s/step
  - loss: 0.1704 - accuracy: 0.9290
  Epoch 6/10
  114/114 [==============================] - 547s 5s/step
  - loss: 0.1481 - accuracy: 0.9386
  Epoch 7/10
  114/114 [==============================] - 563s 5s/step
  - loss: 0.1534 - accuracy: 0.9357
  Epoch 8/10
  114/114 [==============================] - 590s 5s/step
  - loss: 0.1462 - accuracy: 0.9397
  Epoch 9/10
  114/114 [==============================] - 555s 5s/step
  - loss: 0.1382 - accuracy: 0.9445
  Epoch 10/10
  114/114 [==============================] - 526s 5s/step
-   loss: 0.1289 - accuracy: 0.9496
```

Our Loss Function 'binary_crossentropy'.



```
plt.show()
```

[12] ✓ 0.6s Python



REFERENCES

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. ArXiv:1506.02640 [Cs]. http://arxiv.org/abs/1506.02640

[2] Li, E. Y. (2019, December 30). Dive Really Deep into YOLO v3: A Beginner's Guide. https://towardsdatascience.com/dive-really-deep-into-yolo-v3-a-beginners-guide-9e3d2666280e

[3] You Look Only Once(YOLO) (2022, December 3) https://www.neuralception.com/objectdetection-yolo/ .

[4] yolo-coco-data (Weights and Configuration to use with YOLOv3) https://www.kaggle.com/datasets/valentynsichkar/yolo-coco-data

[5] UCF Crime Dataset (Real-world Anomaly Detection in Surveillance Videos) https://www.kaggle.com/datasets/odins0n/ucf-crime-dataset

[6] Machine Learning Model Powered Shoplifting Detection For Shops, Airports, BookStores YOLOv3: Real-Time Object Detection Algorithm (Guide) https://viso.ai/deep-learning/yolov3-overview/

[7] What's new in YOLO v3? https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

[8] Aerial Images Processing for Car Detection using Convolutional Neural Networks: Comparison between FasterR-CNN and YoloV3 https://www.researchgate.net/publication/336602731_Aerial_Images_Processing_for_Car_Detection_using_Convolutional_Neural_Networks_Comparison_between_Faster_R-CNN_and_YoloV3

[9] Computing the Loss Binary Cross Entropy https://towardsdatascience.com/understanding-binary-cross-entropy-log-loss-a-visual-explanation-a3ac6025181a