International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-4, April 2019

Finding Survivors In Flood Affected Areas During Response Operations By Deep Learning Approach

Mohammad Nasim, GV Ramaraju

Abstract— This study explains an approach of finding survivors in flood affected areas during response operations by Deep-Learning technique. Most of the flood disasters affect a large scale livelihood by damaging physical assets, infrastructures and creating local environmental disturbance. We don't have much control to avoid this natural disaster but we can plan it in efficient way at the time of rescue operations on the affected areas and move the people to a safer place where time is an important factor. Here we are assuming that the local transport is completely damaged, there is no internet or network through which we can connect with the people in the affected areas and many people are stuck into it. It means all connections to connect a person has been damaged by the flood. We have more data than any other time in recent history following a debacle on account of automations and accessible developed satellite system that can take picture of disasters as aerial, but we are still studying on ways to process this raw data so that it is valuable for relief efforts. Deep learning technique [11] is what enables Artificial intelligence to identify patterns in images, videos, sounds, and other information utilizing a neural system that reflects our own dim issues. This study relates to identify moving human patterns in a video that has been captured from a flying drone in the affected area through pattern recognition techniques and that can produce a quality analytical output. This technique can be used in various disasters to find the moving human pattern or any particular moving object pattern.

Index Terms— ***Deep learning, drone, flood, rescue, disaster***

I. INTRODUCTION

Fig. 1.Real time object identification using YOLO Algorithm

Drone video data can be used to train a deep learning system [4]. After the preparation, a profound learning machine can anticipate the event of alive questions in influenced regions of debacle utilizing a convolution neural system [8].

Manuscript published on 30 April 2019.
***** Correspondence Author (s)
Mohammad Nasim*, Department of CSE, Lingayas Vidyapeeth, Faridabad, India
GV Ramaraju, Lingayas Vidyapeeth, Faridabad, India

No person can stop the occurrence of natural or human made disaster like flood. We can just take active measures to slow down their impact [9]. By incorporating deep learning applications, for example, drone, specialists can get ongoing information on zones that are going to get hit by a calamity. Opportune and adequate measures would then be able to be taken to spare lives and property. Moreover, automatons can likewise follow explicit regions, for example, woods and thin geological areas, with the goal that uncommon help can be stretched out to such troublesome landscapes. Today, Deep learning is all stage-set, ready to show its potential in specific regions too like flood disaster on the board.

II. PROBLEM STATEMENT

The present surge the board proposition incorporates satellites that collect data from zones that are probably going to experience the ill effects of fiascos. Be that as it may, how proficient are the aftereffects of this proposition? The present catastrophe structure unsuccessful to:

- Offer advantageous data continuously [3],
- Assemble information from numerous destinations in the meantime, and
- Suggest proactive steps for flood disaster prevention [6]

Additionally, the current flood disaster management systems do not offer clear and crisp images of disaster prone regions [5] [7]. This is why; its high time experts implement deep learning in disaster management to overcome these current issues.

III. RESEARCH DESIGN AND METHODOLOGY

A. Conceptual Architecture

YOLO is the third object identification calculation in YOLO (i.e. You Only Look Once) family [1]. It enhanced the exactness with numerous traps and is progressively fit for identifying little objects [2].

The algorithm:

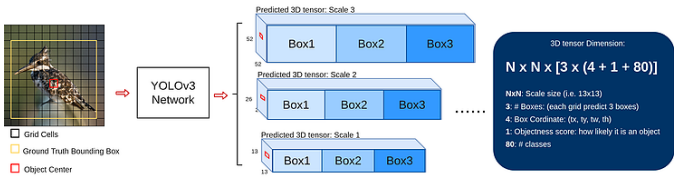

Fig. 2.YOLO Algorithm

YOLO Process Flow:

First, during training, YOLO organize is sustained with info pictures to anticipate 3D tensors (which is the last element outline) to 3 scales, as appeared in the centre one in the above chart. The three scales are intended for distinguishing objects with different sizes. Here we take the scale 13x13 for instance. For this scale, the input image is divided into 13x13 grid cells, every grid cell corresponds to a 1x1x255 voxel inside a 3D tensor. Here, 255 comes from (3x (4+1+80)). Values in a 3D tensor such as rectangle box coordinate, objectness score and class confidence are shown on the right of the diagram. Second, if the centre of the objects ground truth bounding rectangle falls in a certain grid cell (i.e. the red one on the bird image), this grid cell is responsible for predicting the object's bounding rectangle. The relating objectness score is "1" for this matrix cell and "0" for other. For every lattice cell, it is appointed with 3 earlier boxes of various sizes. What it realizes amid preparing is to pick the correct box and figure exact offset/coordinate. However, how does the matrix cell realize which box to pick? There is a standard that it just picks the case that covers ground truth jumping square shape most. Ultimately, how to pick the underlying size of those 3 earlier boxes? The creator utilizes K-mean bunching to arrange the aggregate boxes from COCO dataset to 9 groups before preparing. These outcomes in 9 sizes browsed 9 group, 3 for 3 scales. This earlier data is useful for system to find out how to compile box counterbalance/arrange correctly in light of the fact that naturally, awful decision of box estimate makes it harder and longer for the system to learn.

B. *Technical Architecture*

Below is a diagram of YOLO's network architecture Fig 3. It is a component learning based system that embraces 75 convolutional layers as its most amazing asset. No completely associated layer is utilized. This structure makes it conceivable to manage pictures with any sizes. Additionally, no pooling layers are utilized. Rather, a convolutional layer with stride 2 is utilized to down sample the component delineates, estimate invariant element forwardly. Moreover, a ResNet-alike structure and FPN-alike structure is likewise a key to its exactness enhancement.

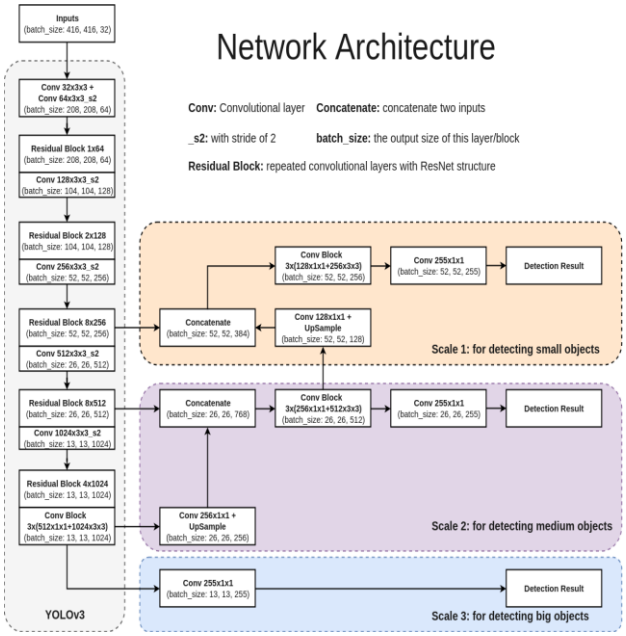

Fig. 3.YOLO Network Architecture

C. *Scales: handling objects of different sizes*

There are objects of various sizes on the pictures. Some are enormous and some are little. It is attractive for the system to identify every one of them. Along these lines, the system should be equipped for "seeing" objects that are of various sizes. As the system goes further, its element delineates littler. In other words, the more profound it goes, the harder it is to distinguish littler articles. Instinctively, it is smarter to identify the articles at various component maps before little protests wind up vanishing. As in SSD, object identification is done on various features maps all together catch separate. Be that as it may, the objects are not totally important at various profundities. So, what is the meaning of this? All things considered, with system profundity expanding, the highlights change from low-level highlights (edges, hues, unpleasant 2D positions. and so forth) to abnormal state highlights (semantic-important data: hound, feline, vehicle. and so forth) with profundity expanding. So making forecasts on highlight maps at various profundity sounds it can do recognition for multi-scale objects, however it isn't as precise not surprisingly. In YOLO, this is enhanced by receiving a FPN-like structure.

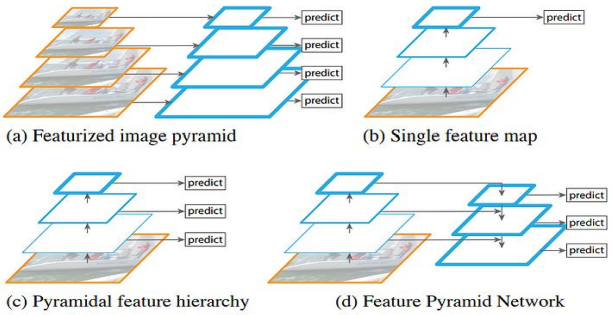

Fig. 4.YOLO Structure

D. *Multi-scale Feature Learning Illustration*

As appeared on the above delineation, there are 4 fundamental structures for multi-scale object learning.

- The clearest one. Build a picture pyramid and info each pyramid level to singular system uniquely intended for its scale. Thus, it is moderate in light of the fact that each dimension needs its very own system or process.
- The forecast is done toward the finish of the component outline. This structure can't deal with different scales.
- The forecast is done on object maps at various profundity. This is embraced by SSD. The forecast is finished by utilizing the highlights adapted up until now and further highlights at more profound layers can't be used.
- Similar to (c) however further objects are used by up sampling the component delineates converged with current element outline. This is fascinating because it let current feature map to see its features in "future" layers and utilize both to do accurate prediction. With this technique, the modal is progressively able to capture the object's information, both low-level and high-level.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

In YOLO, there are 3 scales used in (d) form. This helps detect small objects effectively. As appeared in the git underneath, little vehicles and individuals can be recognized effectively. ResNet-alike structure: a superior method to get a handle on good features.

In YOLO, a ResNet-alike structure (called Residual Blocks in the YOLO Architecture Diagram) is utilized for Feature study. Essentially a Residual Block comprises of a few convolutional layers and alternate way ways [10]. A case of an alternate route way is outlined underneath.

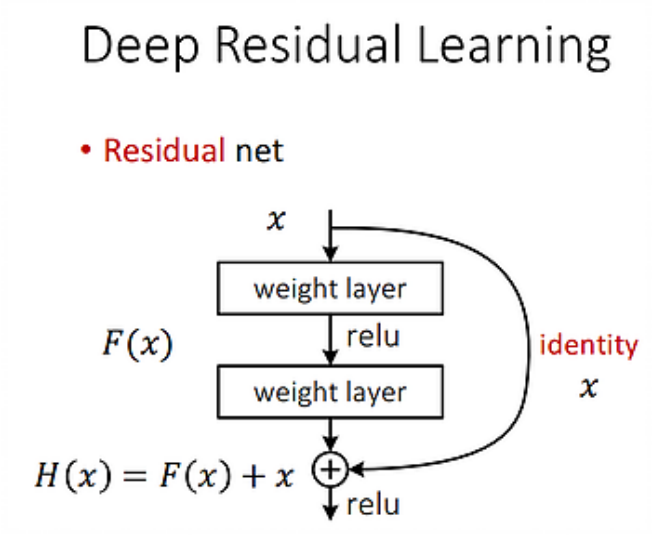

Fig. 5. Residual Blocks inside the Architecture Diagram

The bended bolt on the privilege speaks to an easy route way. Without this, it is an exemplary CNN organize, which takes in the element one by another. As the framework goes further, the harder it is to take in the items well. On the remote possibility that we incorporate a substitute path as showed up, the layers inside the simple course acknowledges what to add to the old segment in order to make better component. Along these lines, a mind boggling part H(x), which used to be delivered autonomous, is right now shown as x, where x is the old component starting from the backup course of action and F(x) is the "supplement" or the "extra" to learn now. This makes it simpler for the system to take in the highlights steadily, particularly in profound systems. This system changes the aim of learning. In this way, rather than taking in a total complex element, the new aim is to take in the supplemental "leftover" that is utilized to be indicated the old element, which disentangled multifaceted nature for learning.

E. No softmax layer: multi-label classification

Softmax layer is supplanted by 1x1 convolutional layer with calculated capacity. By utilizing a softmax, we need to expect that each yield just has a place with precisely ONE of the classes. In any case, in some dataset or situations where the marks are semantically comparable (for example Lady and Male person), preparing with softmax probably won't let the system sum up the information conveyance well. Rather, a calculated capacity is utilized to adapt to multi-mark order.

IV. SYSTEM DEVELOPMENT

A. Correlation with Other Detectors

YOLO is to a great degree quick and exact. In mean Average Precision estimated at 0.5 Intersection over Union YOLO is comparable to Focal Loss yet about 4 times quicker. Also, we can without much of a stretch tradeoff among speed and precision just by making the extent different every time of the standard, no need to re-process it.

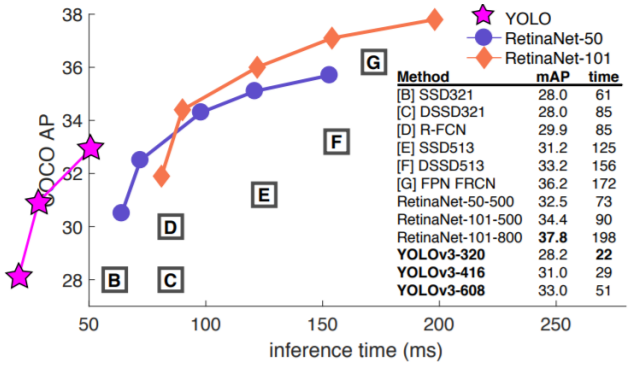

Fig. 6.YOLO runs essentially quicker than other discovery techniques with similar execution.

Performance on the COCO Dataset

Model	Train	Test	mAP	FLOPS	FPS	Cfg	Weights
SSD300	COCO trainval	test-dev	41.2	-	46		link
SSD500	COCO trainval	test-dev	46.5	-	19		link
YOLOv2 608x608	COCO trainval	test-dev	48.1	62.94 Bn	40	cfg	weights
Tiny YOLO	COCO trainval	test-dev	23.7	5.41 Bn	244	cfg	weights
SSD321	COCO trainval	test-dev	45.4	-	16		link
DSSD321	COCO trainval	test-dev	46.1	-	12		link
R-FCN	COCO trainval	test-dev	51.9	-	12		link
SSD513	COCO trainval	test-dev	50.4	-	8		link
DSSD513	COCO trainval	test-dev	53.3	-	6		link
FPN FRCN	COCO trainval	test-dev	59.1	-	6		link
Retinanet-50-500	COCO trainval	test-dev	50.9	-	14		link
Retinanet-101-500	COCO trainval	test-dev	53.1	-	11		link
Retinanet-101-800	COCO trainval	test-dev	57.5	-	5		link
YOLOv3-320	COCO trainval	test-dev	51.5	38.97 Bn	45	cfg	weights
YOLOv3-416	COCO trainval	test-dev	55.3	65.86 Bn	35	cfg	weights
YOLOv3-608	COCO trainval	test-dev	57.9	140.69 Bn	20	cfg	weights
YOLOv3-tiny	COCO trainval	test-dev	33.1	5.56 Bn	220	cfg	weights
YOLOv3-spp	COCO trainval	test-dev	60.6	141.45 Bn	20	cfg	weights

Fig. 7.Execution on the Data of COCO

B. Recognition using a pre-established Standard

This examination would oversee us via distinctive articles along with YOLO framework using a pre-established standard. On this off chance which we don't as of now have Darknet53 introduced, we ought to complete that first. Otherwise then again as opposed to perusing such simply run:

github replica darknet53
cd darknet53
create

Simple! We as of now having configuration record for YOLO inside config/subcatelog. We have to load pre-prepared weight document. Otherwise after that simply run this:

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

weget YOLO.mass

At that point run the finder!

./darknet53 finder config/YOLO.config YOLO.mass data/puppy.jpg

We would create few yields in this manner:

```
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32  0.299 BFLOPs
    1 conv     64  3 x 3 / 2   416 x 416 x  32   ->   208 x 208 x  64  1.595 BFLOPs
    .......
  105 conv    255  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 255  0.353 BFLOPs
  106 detection
truth_thresh: Using default '1.000000'
Loading weights from yolov3.weights...Done!
data/dog.jpg: Predicted in 0.029329 seconds.
dog: 99%
truck: 93%
bicycle: 99%
```

Fig. 8. Detector outcome

Darknet53 produce the feature it recognized, its guarantee, and up to what degree it takes to search those. You didn't consolidate Darknet53 along with OpenCV therefore it can't demonstrate the revelations direct. Or maybe, it commits them inside predictions.png. We can view this to visualize the recognized articles. As we are utilizing Darknet53 on the PC it will take nearly 7-13 seconds for each picture. If we modify the PC adjustment, it will be significantly speedier. The recognize arrange is a summary for a continuously expansive adjustment of this course. It is nearly equal to the request

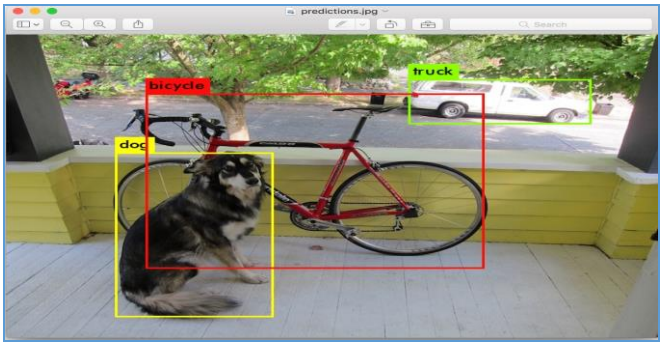

Fig. 9. Darknet53 produce

./darknet53 finder testing config/coco.Info config/YOLO.config YOLO.mass data/puppy.jpg

C. Numerous Images

Rather than providing a picture on the order line, you can abandon it clear to attempt numerous pictures consecutively. Or maybe we would see an impel when configuration and burdens are done stacking:

```
./darknet detect cfg/yolov3.cfg yolov3.weights
layer     filters    size              input                output
    0 conv     32  3 x 3 / 1   416 x 416 x   3   ->   416 x 416 x  32  0.299 BFLOPs
    1 conv     64  3 x 3 / 2   416 x 416 x  32   ->   208 x 208 x  64  1.595 BFLOPs
    .......
  104 conv    256  3 x 3 / 1    52 x  52 x 128   ->    52 x  52 x 256  1.595 BFLOPs
  105 conv    255  1 x 1 / 1    52 x  52 x 256   ->    52 x  52 x 255  0.353 BFLOPs
  106 detection
Loading weights from yolov3.weights...Done!
Enter Image Path:
```

Fig. 10. Type a picture path example information/horses.jpg to have it anticipate rectangle for particular that picture.

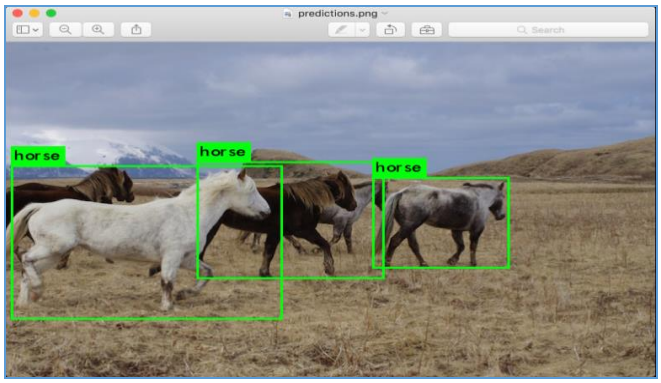

Fig. 11. Darknet prints output

When it is completed, it would instigate us for another approaches to endeavor particular pictures. Type Ctrl-C command to leave the application when you are completed.

D. Teach the standard

Presently you can teach! This is the direction:

./darknet53 finder teach config/coco.info config/YOLO.config darknet53.conv.74

If you have to use diverse gpus run:

./darknet53 finder teach config/coco.info config/YOLO.config darknet53.conv.74 -gpus 0,1,2,3

If we wanted to end and wanted to restart teaching from one checkpoint:

./darknet53 finder teach config/coco.info config/YOLO.config reinforcement/YOLO. reinforcement -gpus 0,1,2,3

V. CONCLUSION

2017 saw some genuine battle for the best Object Detection appear with RetinaNet (another type of stage locator), Faster Region-CNN with Feature Pyramid Network along with ResNext as the backbone and Mask Region-CNN with ResNext backbone and after that RetinaNet with the ResNext backbone finishing the diagrams with a Mean Average Precision of 61 on dataset of COCO for 0.5 Intersection over Union. RetinaNet being a one-orchestrate pointer was snappier than the rest. With no new kind of YOLO in 2017, 2018 ran with best RetinaNet (the one I referenced above) and after that now YOLO! YOLO gives a Mean Average Precision of 57.9 on dataset of COCO for Intersection over Union 0.5. For examinations essentially suggest the table underneath:

	backbone	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L
Two-stage methods							
Faster R-CNN+++ [3]	ResNet-101-C4	34.9	55.7	37.4	15.6	38.7	50.9
Faster R-CNN w FPN [6]	ResNet-101-FPN	36.2	59.1	39.0	18.2	39.0	48.2
Faster R-CNN by G-RMI [4]	Inception-ResNet-v2 [19]	34.7	55.5	36.7	13.5	38.1	52.0
Faster R-CNN w TDM [18]	Inception-ResNet-v2-TDM	36.8	57.7	39.2	16.2	39.8	**52.1**
One-stage methods							
YOLOv2 [13]	DarkNet-19 [13]	21.6	44.0	19.2	5.0	22.4	35.5
SSD513 [9, 2]	ResNet-101-SSD	31.2	50.4	33.3	10.2	34.5	49.8
DSSD513 [2]	ResNet-101-DSSD	33.2	53.3	35.2	13.0	35.4	51.1
RetinaNet [7]	ResNet-101-FPN	39.1	59.1	42.3	21.8	42.7	50.2
RetinaNet [7]	ResNeXt-101-FPN	**40.8**	**61.1**	**44.1**	**24.1**	**44.2**	51.2
YOLOv3 608 × 608	Darknet-53	33.0	57.9	34.4	18.3	35.4	41.9

. Fig: 12: Mean Average Precision Comparisons 2018!

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

A. What has been changed? What are the supposed Incremental corrections?

- **Anticipate of Bouncing Boxes:** YOLO simply like other deep learning tools utilizes dimension groups to produce Anchor Boxes. Presently as YOLO is a solitary system the misfortune for objectiveness and grouping should be determined independently yet from a similar system. YOLO estimates the tolerance points utilizing strategic relapse where 1 implies finish cover of bouncing box earlier over the real ground object. It will anticipate just 1 holding box earlier for one real ground object (dissimilar to Faster Region CNN) and any issues in this would bring about for both grouping and finding (tolerance) misfortune. There would likewise be other jumping box priors which would have tolerance score more than the limit yet not exactly the best one, for these mistake will bring about for the recognition misfortune and not for the grouping misfortune.
- **Class Probabilities:** YOLO utilizes autonomous calculated classifiers for every class rather than a customary soft-max surface. It has been done to create the arrangement multi-mark order. What does it mean and how this includes esteem? Take a precedent, where a lady is appeared in the image and the standard is prepared for both individual and lady, here soft-max will prompt the class probabilities been isolated into two classes with state 0.4 and 0.45 likelihoods. However, autonomous classifiers understand this problem and they provide a yes versus no likelihood for every class, similar to what's the likelihood that in the image where there is a lady will result 0.8 and what's the likelihood that in the image if there is an individual will result 0.9 and you can mark the item as both individual and lady.
- **Expectations crosswise over scales:** To help findings on changing scales YOLO estimates bounding rectangles at three distinct scales. At that point objects are separated from every scale by utilizing a strategy like that of FPN. What's the strategy? let's see the details below.

We will pick the object delineate two layers past & up-sample the same by two times. You likewise pick an element delineate prior in the system and union it along with your up-sampled highlights utilizing component insightful expansion. These strategies enable you to receive increasingly significant semantic data from the up-sampled highlights & better grained data from the before highlight outline. We at that point add a couple of more convolutional layers to process this joined element delineate, in the end foresee a comparative tensor, albeit now double the size. We play out a similar structure once again to foresee rectangle of last scale. Along these lines our expectations for the third scale benefited by all the earlier calculation and additionally fine-grained highlights from right off the bat in the system.

Along these lines, YOLO gains the capacity to more readily anticipate at different scales utilizing the above technique. The jumping box priors created utilizing dimension groups are partitioned into three scales, such that there would be three bouncing rectangle priors for every scale & in this way there are 9 bouncing box priors as total.

- **Feature Extractor:**

Backbone	Top-1	Top-5	Bn Ops	BFLOP/s	FPS
Darknet-19 [13]	74.1	91.8	7.29	1246	**171**
ResNet-101[3]	77.1	93.7	19.7	1039	53
ResNet-152 [3]	**77.6**	**93.8**	29.4	1090	37
Darknet-53	77.2	**93.8**	18.7	**1457**	78

Table 2. **Comparison of backbones.** Accuracy, billions of operations, billion floating point operations per second, and FPS for various networks.

Fig: 12: Mean ImageNet Outcome

As should be obvious, Darknet53 is superior to ResNet-101 yet 1.5 occasions quicker & is as exact as ResNet152 however two times quicker.

B. What has been modified?

- The normal exactness for little features enhanced, it is currently superior to Faster Region-CNN yet Retina-Net is still better in this range.
- As MAP expanded limitation issues diminished.
- Expectations at various scales or viewpoint proportions for same item enhanced as a result of the expansion of FPN like method.
- Mean Average Precision expanded altogether.

C. What are the challenges?

- Processing of the mill precision for middle and wide objects could be enhanced as middle is five percent and gigantic is ten percent behind the scene.
- Mean average precision count between 0.5 to 0.95 overlay over Union could be broadened.
- The utilization of DarkNet53 & in like manner YOLO is right now in C, perhaps in near future a python solution too.

D. Summary

YOLO is quick, has at standard exactness with good 2 phase identifiers (on 0.5 Intersection over union) and all these pushes it an incredible object identification standard. Utilization of Object identification in spaces like disaster management especially in flood and so forth require the standards to be extremely fast (a little trade off on exactness is alright) yet YOLO is additionally exceptionally precise. This makes it the best standard to pick in these sort of uses where speed is vital either for the real time ground objects or the information is simply too huge. A decent exactness with the best speed makes YOLO a go to object recognition demonstrate in flood rescue management.

REFERENCES

1. 'Image Preprocessing for Efficient Training of YOLO Deep Learning Networks','Hyeok-June Jeong ; Kyeong-Sik Park ; Young-Guk Ha','2018 IEEE International Conference on Big Data and Smart Computing (BigComp)','Year: 2018','Pages: 635 - 637'
2. 'An object detection system based on YOLO in traffic scene','Jing Tao ; Hongbo Wang ; Xinyu Zhang ; Xiaoyu Li ; Huawei Yang','2017 6th International Conference on Computer Science and Network Technology (ICCSNT)','Year: 2017','Pages: 315 - 319,
3. 'Realtime object detection in IoT (Internet of Things) devices','Erke Arıbaş ; Evren Dağlarl','2017 25th Signal Processing and Communications Applications Conference (SIU)','Year: 2017','Pages: 1 - 4'

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication (BEIESP)

4. 'Real-time Implementation of YOLO+JPDA for Small Scale UAV Multiple Object Tracking','Shuoyuan Xu ; Al Savvaris ; Shaoming He ; Hyo-sang Shin ; Antonios Tsourdos','2018 International Conference on Unmanned Aircraft Systems (ICUAS)','Year: 2018','Pages: 1336 - 1341'
5. 'Flood disaster response and decision-making support system based on remote sensing and GIS','Hu Zhuowei ; Li Xiaojuan ; Sun Yonghua ; Gong Zhaoning ; Wang Yanhui ; Zhu Liying','2007 IEEE International Geoscience and Remote Sensing Symposium','Year: 2007','Pages: 2435 - 2438'
6. 'A flood emergency response system based on flood disaster evaluation models','Li Fang ; Jianwei Yue ; Jiawen Dai ; Guobin Ma ; Zhuoyuan Yu','2010 18th International Conference on Geoinformatics','Year: 2010','Pages: 1 - 4'
7. 'Research on Methods of Quick Monitoring and Evaluating of Flood Disaster in Poyang Lake Area Based on RS and GIS','Youliang Chen ; Xiaosheng Liu','2008 IEEE International Symposium on Knowledge Acquisition and Modeling Workshop','Year: 2008','Pages: 1105 - 1108'
8. 'A neural network based prediction model for flood in a disaster management system with sensor networks','S. Mandal ; D. Saha ; T. Banerjee','Proceedings of 2005 International Conference on Intelligent Sensing and Information Processing, 2005','Year: 2005','Pages: 78 - 82'
9. 'Study on Decision Support System of Flood Disaster Based on the Overall Process Emergency Management','Qing Yang ; Qiang Zhang','2007 International Conference on Wireless Communications, Networking and Mobile Computing','Year: 2007','Pages: 6126 - 6129'
10. 'K. He, X. Zhang, S. Ren, and J. Sun.','Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition','pages 770–778', 'Year: 2016'
11. ‘'Deep Learning Could Help First Responders Offer Critical Aid in the Wake of Disasters' , 'https://futurism.com/titan-more-common-earth-thought', 'Aylin Woodward', '20-Jan-18'

AUTHORS PROFILE

First Author: Mohammad Nasim I have extensive live experience of 20+ years for many large scale system projects in developing and architecting software solutions using various latest COTS and open source technologies and products. I am working here on a clients facing role and working as a senior architect in Pubic authority of electricity and water, Muscat, Oman. I do have working and architect knowledge on data center (Custom/Cloud), platforms (SASS, IAAS, PASS) , Technology (Dotnet, Open Source), storage (Mobile ,Server) , database (Cloud, Oracle, Microsoft SQL, PostgreSQL) , web services (REST, Micro-services) , security(Internet) and application performance and testing. I have completed my Master’s Degree of Technology in Computer Science (M.Tech.-CSE) from Shri Venkateshwara University, Gajraula, U.P., India. Now I am persuing my Philosophy in Doctorate in Computer Science (Ph.D.-CSE) from Lingaya’s University, Faridabad, Haryana , India. I have participated in IInd International Conference of System Modelling and Advancement in Research Trends (SMART) organized by Teerthanker Mahaveer University, Moradabad, U.P., INDIA and Academic Journals Online in Nov 13 and presented a paper titled "Mobile GIS on ArcGIS technologies for Android”. I have participated in International Conference of Advance Research and Innovation organized by International Journal of Advance Research and Innovation, Delhi, INDIA and Delhi Technological University, Delhi, INDIA in Feb 14 and presented a paper titled "Cross Platform Mobile GIS System for Data Collection based on GPS and emerging GIS Technologies". I am member of IEEE since Mar 2016 – Present.