Real Time Detection Of Nilgai

Darshak Patel(201701459) ,Jay Kadia(201701225), Dhruv Mendpara(201701146)

Mentor: Prof. Pankaj Kumar

Abstract

We made a model for real life and real-time Nilgai detection. The idea of our Nilgai detection work will help farmers to protect their crops from native wild animals and help for road-safety to warn drivers from possible accidental threat with wild animals. We used YOLOv3 Darkent-53 model which was pre-trained on COCO dataset. We used transfer learning approach to custom train. For training, we created our own dataset.

Introduction

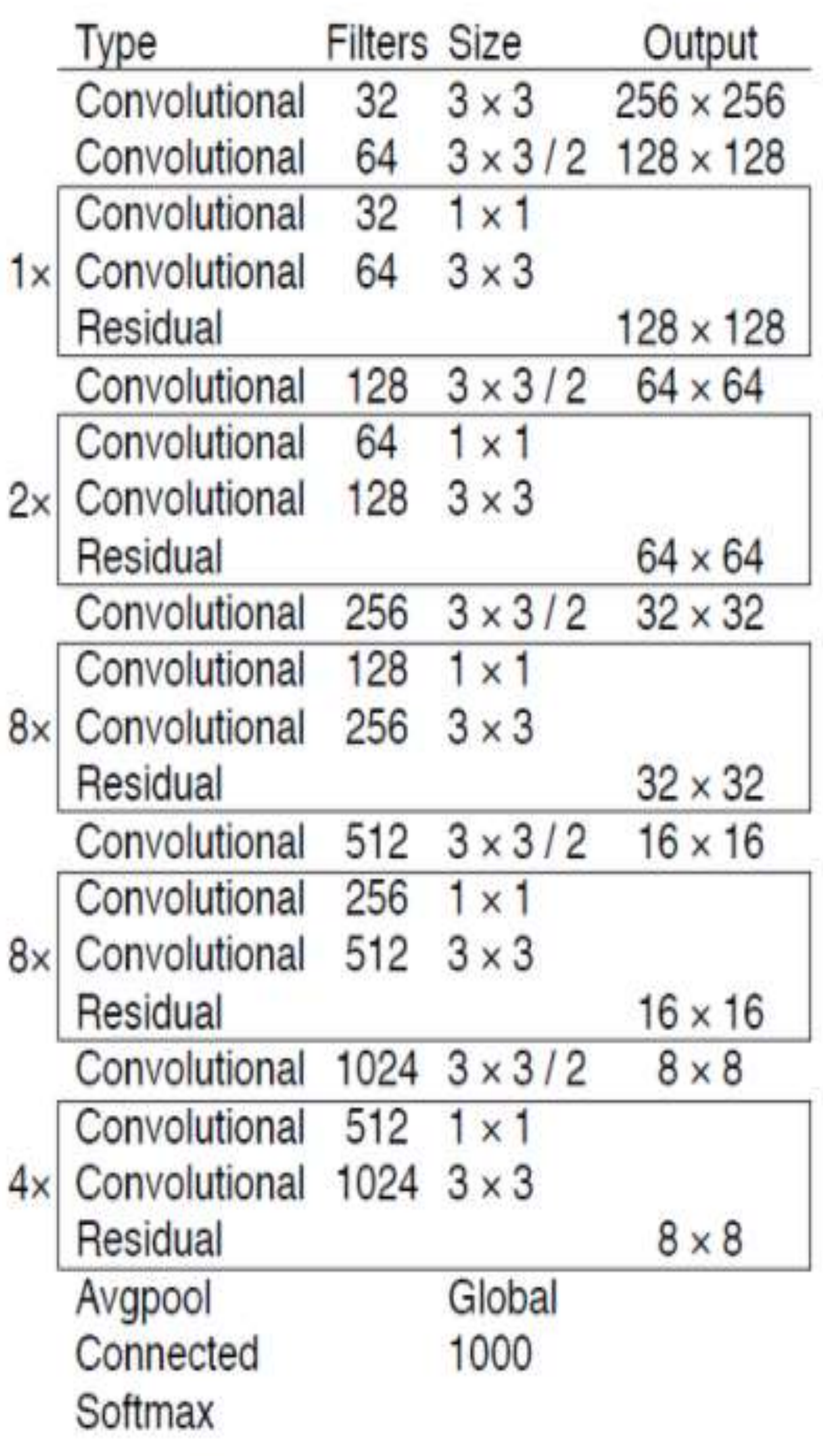
In many incidents, farmers crops were destroyed by wild animals like pig, monkey, nilgai and other animals. Usually, these animals enter the farms and destroy their corps. To stop such loses, many farmers put electric fencing around their farms but it is not viable option as it has a risk to take human and animal life. So we come with and idea to develop a model and train it on one of the animal(nilgai) dataset. And to use in real life we can develop a camera traps system which can make some sound that will drive nilgai away from the farm after detecting it.

During this work, we realized that there were no dataset available for Nilgai.

Choice of CNN Architecture

First we think to work with VGG19 or YOLOv3 model but, In VGG19 it classify the whole image as Nilgai or not a Nilgai. While in YOLO model will classify the object from the given frames of images. And YOLO is a real time faster technique compared to VGG19, so we used YOLO algorithm, i.e. YOLOv3.

Figure 1a. Shows the architecture of YOLOv3. It has 53 convolution layers, so it is named as Darknet-53. Darknet is an open-source neural network framework written in C and CUDA. DarkNet-53 has shortcut connections as well as a better object detector with feature map upsampling and concatenation.



	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
1×	Convolutional	64	3 × 3	
1×	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
2×	Convolutional	128	3 × 3	
2×	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
8×	Convolutional	256	3 × 3	
8×	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
8×	Convolutional	512	3 × 3	
8×	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
4×	Convolutional	1024	3 × 3	
4×	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Figure 1a.

Collecting and Preprocessing the Database

Since, there were no datasets available for Nilgai, we used the help of Google to search images of nilgai and used Google chrome extension called "Download all Images" to download all the images from webpage. Then we manually filtered all the Nilgai images. We collected around 1100 images of Nilgai.
To train Yolo model we need to label every object in the image. We used open source software **labelimg** and **makesense.ai** to manually label all the images. For every image these software will create .txt file which contains class and dimension of the object.
Format of each row in the txt file is shown below.

CLASS	X_CENTER	Y_CENTER	WIDTH	HEIGHT

Training

We trained our model with the help of Darknet Open source neural network. Darknet can be used with Google colab environment. Training was done on Tesla K80 GPU provided by Google colab. To train the model changes were made to darknet-53 model for only one class.

YOLOv3 Configuration Parameters:

Batch Size : 64,which means 64 images used in one iteration.

Subdivisions: 16, which means GPU will process subdivision number of images at any time.

Width, Height, Channels: 416,416,3.

Momentum & Decay: 0.9,0.005

Learning Rate: 0.001

Number of Iterations: 40000

We have trained YOLOv3 for 4000 iterations on Tesla K80 GPU for 12 hours.

Results

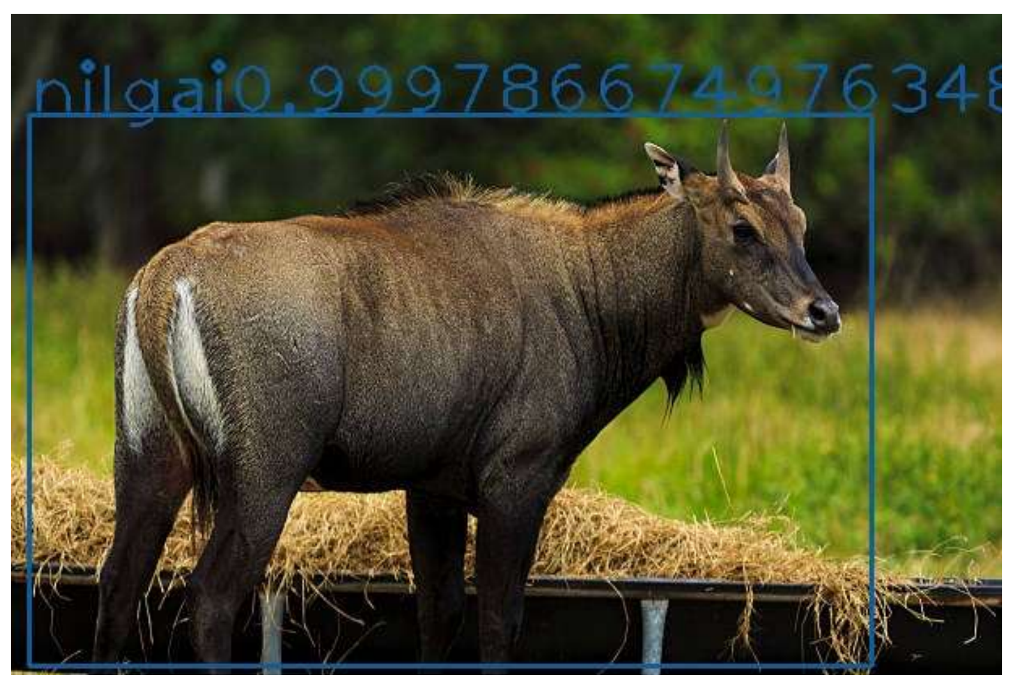
For all the experiments, we have used laptop with 4GB RAM, 64-bit operating system and Intel® Core™ i5-inside CPU @2.2GHz.

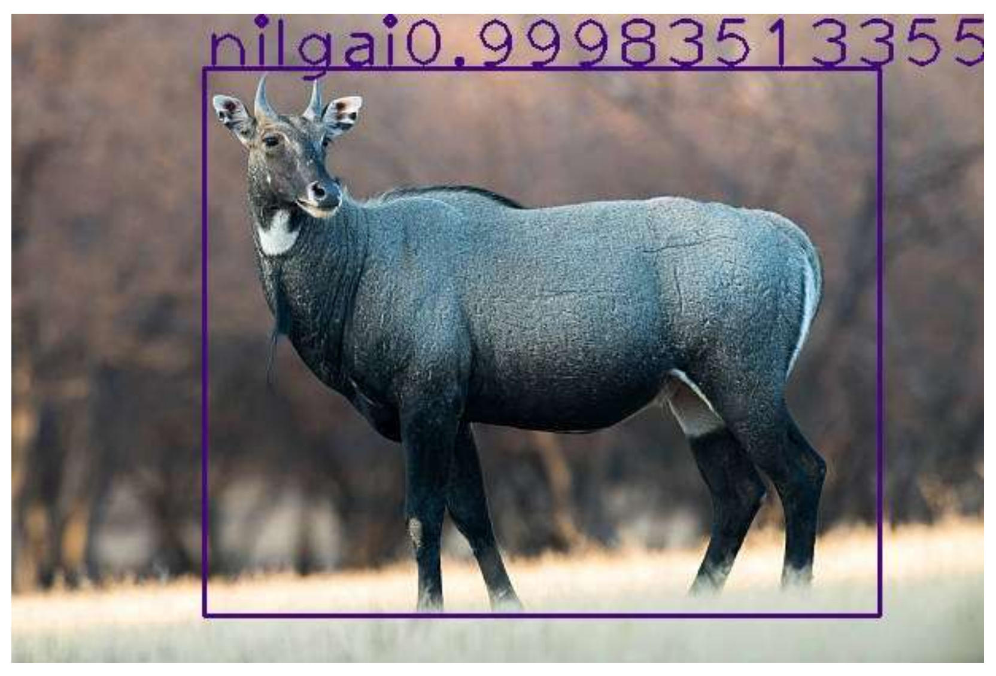
we use IOU (intersection over union) metric to measure the accuracy of the object detector.

At 2230 iteration Avg. loss was 0.143382. and accuracy was 90.36%.

We created a program in python to capture the live feed from the webcam or give input as image file to the model and it will detect nilgai from the given live feed or image.

Here are some results shown below.





Conclusion and Future Work

In the present work, we designed techniques to detect nilgai in real time. But, Due to the pandemic we couldn't test it in the real life.

We can also use custom YOLOv3 to extend our nilgai database. And we can improve the accuracy with much good dataset collected in real life.

Bibliography

Web-based Software:
1. https://www.makesense.ai/
2. https://github.com/tzutalin/labelImg
3. https://github.com/pjreddie/darknet.git
4. J. Redmon. Darknet: Open source neural networks in c. http://pjreddie.com/darknet/, 2013–2016

Paper:
1. Joseph Redmon , Ali Farhadi. *YOLOv3: An Incremental Improvement.* arXiv:1804.02767 .