Second Capstone Project

Yolov3 object detection on videos

Springboard 2019

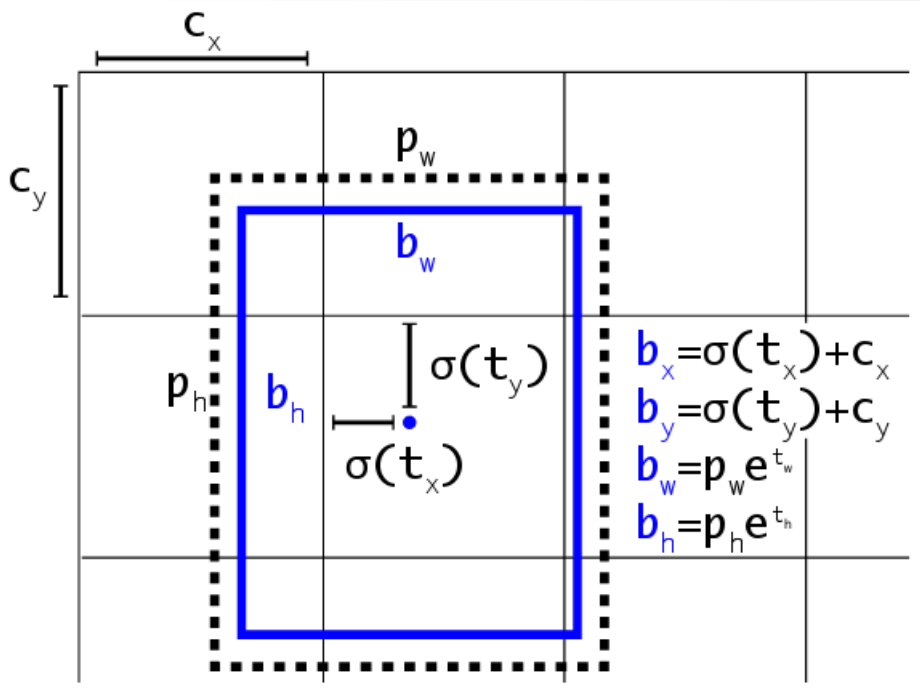
Yolo & Yolov3

- YOLO ("you only look once") is a popular algorithm used to accurately detect and identify objects on images while also being able to run in real-time. The algorithm requires only one forward propagation pass through neural network to make predictions. The resulting image includes boxes around objects with name tags.

- YOLOv3 is improvement to YOLO algorithm. It uses a few improvements for training and increase performance, including: multi-scale predictions, a better backbone classifier, and more.

Bounding Box Prediction

- The network predicts 4 coordinates for each bounding box, tx, ty, tw, th
- The offset from top left corner of the image is measured by (cx,cy) and the bounding box prior width and height are measured by pw, ph
- The equation for calculating final box coordinat are given in here (see image on the right)
- YOLOv3 predicts an objectness score for each bounding box using logistic regression



Source: https://pjreddie.com/media/files/papers/YOLOv3.pdf

Class Prediction

- Each box predicts the classes the bounding box may contain using multi-label classification. An independent logistic classifier is used instead of Softmax. Binary cross-entropy loss is used for the class predictions during the training.

Predictions Across Scales

- YOLOv3 predicts boxes at 3 different scales
- Several convolutional layers are added from the base feature extractor
- The last of these predicts a 3-d tensor encoding bounding box, objectness, and class predictions

Feature Extractor

- New network is a hybrid approach between the network used in YOLOv2, Darknet-19, and that newfangled residual network stuff
- The network uses successive 3×3 and 1 × 1 convolutional layers but has some shortcut connections as well and is significantly larger. It has 53 convolutional layers and called Darknet-53!

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

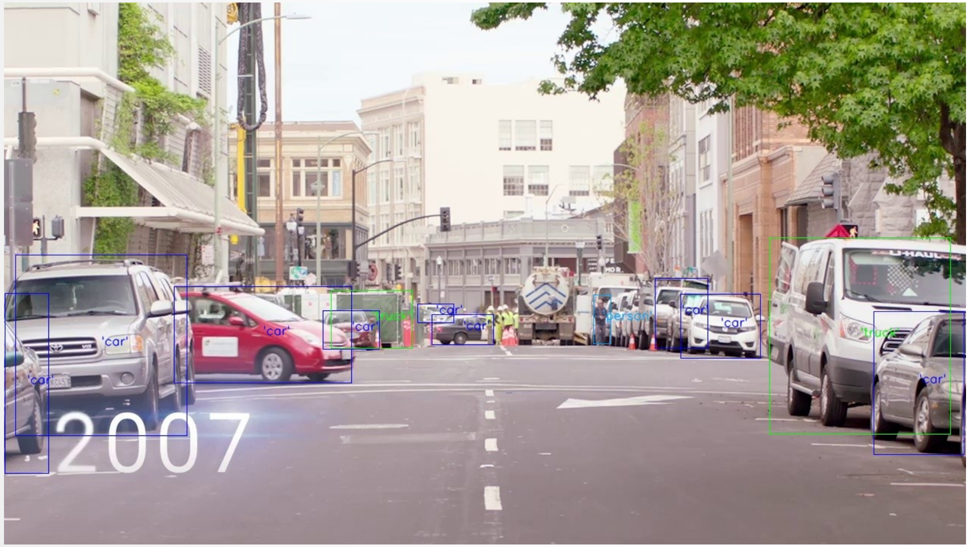
Table 1. **Darknet-53.**

Source: https://pjreddie.com/media/files/papers/YOLOv3.pdf

Description of the Dataset

- Videos with mp4 format that include cars and persons will be used as an input to docker container where python application will run to detect and identify objects on the given video frame.
- The resulting processed video with detected and tagged objects will be produced as an output video in avi format.
- As an example, video with mp4 format can be downloaded from the youtube using the following online video grabber tool: https://www.videograbber.net/, and then used for testing purposes.
- We recommend to create new directory, rename newly downloaded video to test.mp4 and store the video in that new directory. We will use the path to the directory as an input docker volume mapping into container. Thus, the video will be available inside the container for the analyses.

Results



Source: https://www.youtube.com/watch?v=6FZH652qYkA

Summary of Findings

- We can clearly see accurately detected and tagged objects on the video frames. These are car and truck objects.
- We used open cv library to brake the video into frames and analyze them with darknet open source neural networks. Then, we combined processed frames into resulting video.
- The processing took place in docker container. The docker image for the container was build with python3.5 base. Also, it includes all darknet files downloaded from the source and installed with make command.

Conclusion and Open Questions

- We build docker application based on darknet open source neural networks (yolov3) to detect and tag objects on videos.
- We used open-cv for image processing.
- Obtained resulting video contained accurately detected and tagged objects.
- Video processing could be accelerated by reducing video quality and resizing frames.
- Updating python script (darknet.py) to include multi-threading could also speed up the overall application run time.