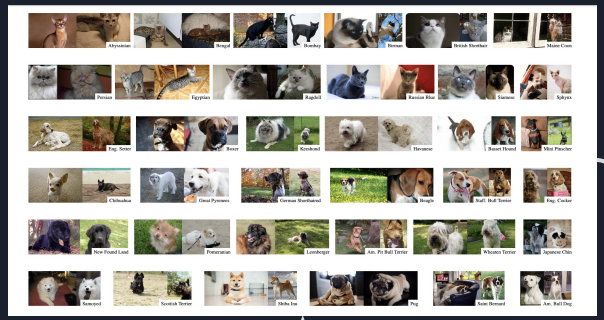
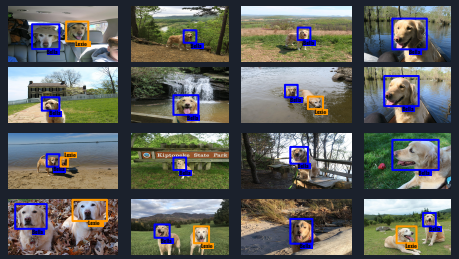
Implementing Pet Detection on the NVIDIA Jetson Nano

By Brandon Chestnut and Jason Cox
Governor's School for Science and Technology
Mentor: Gavin Alberghini, MITRE Corporation

Overview

Oxford Pet Dataset





Personal Pet Images

Desktop Training

TensorFlow

YOLO

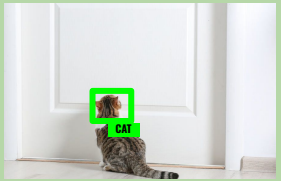
Jetson Operation

TensorRT

TensorFlow Lite

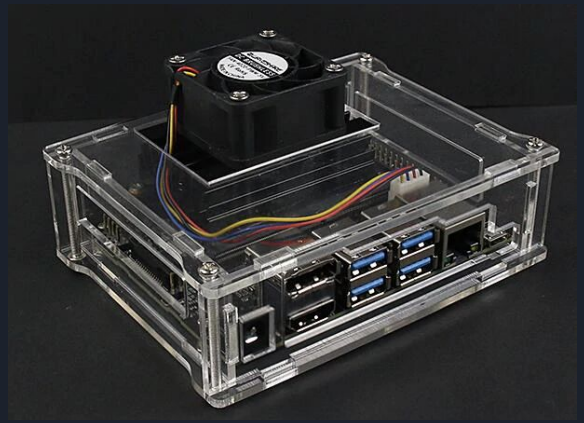
YOLO

Automated Pet Doorbell



NVIDIA Jetson Nano

- RAM
  - 2GB RAM (shared)
- CPU
  - 4-core ARM processor (1.43 Ghz)
- GPU
  - 128 core
- Power
  - 5-10W power consumption



- Machine Learning Focus
  - Having a discrete GPU allows the matrix calculations of neural networks to be run significantly faster than on a CPU
  - The Jetson Nano OS (Jetpack) comes preinstalled with CUDA GPU drivers and even tensorRT

https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/

Literature Review

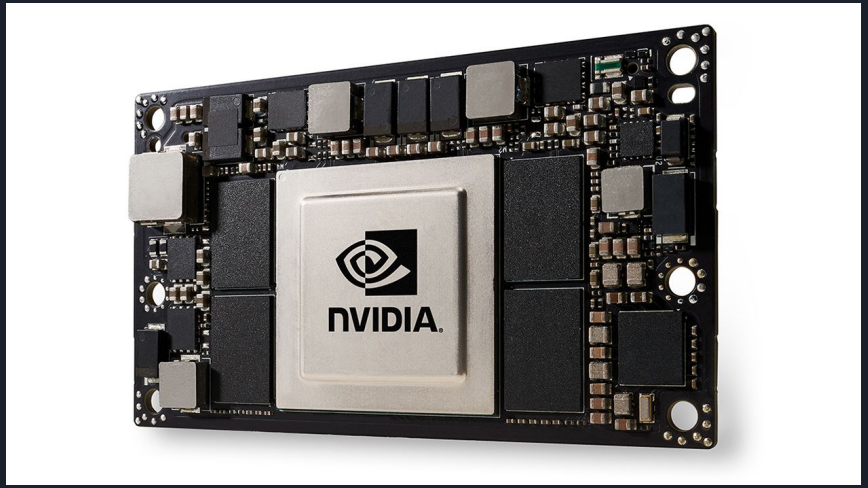
Mittal's survey of 122 papers regarding the NVIDIA Jetson series

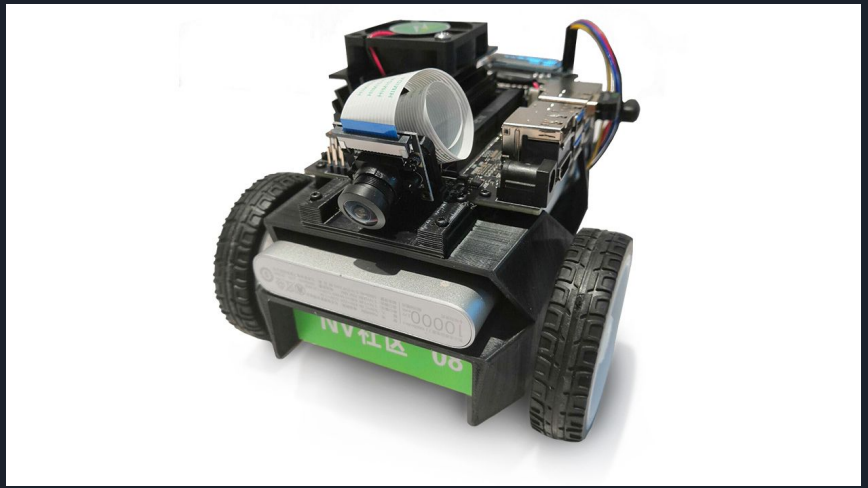
Program Optimizations

- Optimal CNN (size, lightweight)
- Reduce image resolution

Future Jetson Improvements

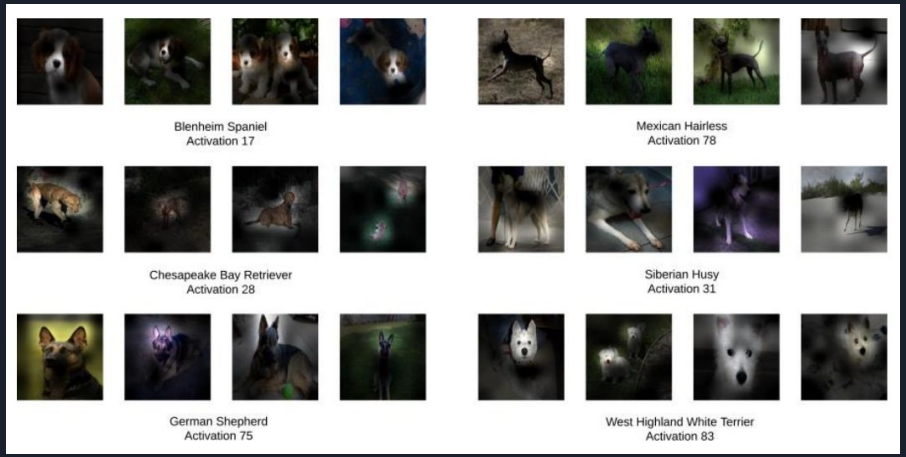
- Smaller feature sizes
- Non-volatile memory
- Closer integration between CPU and GPU





Similar Projects

- Gyanendra, Verma, and Gupta 2018, demonstrate the use of self-learned Deep Convolutional Neural Networks in wild animal detection.
- https://www.researchgate.net/publication/324960511_Wild_Animal_Detection_Using_Deep_Convolutional_Neural_Network



- Higa 2019 uses Convolutional Neural Networks on a dataset of dogs to classify their breeds
- https://core.ac.uk/download/pdf/214315107.pdf

Research Questions

1: Can we achieve real-time pet recognition on the Nvidia Jetson Nano 2GB?

2: Compare the processes of pet recognition between Tensorflow and YOLO. Are there any measurable differences in results for practical application?

3: Is there a large performance difference between Tensorflow optimization packages?

Importance/Application

Research Value

- Evaluates performance on NVIDIA Jetson, which is the future for mobile machine learning
- Contribute to existing knowledge base of machine learning and embedded systems
- Proves the beneficial use of NVIDIA Jetson in everyday life

Statistics

- 41 dogs die from being left outside
- ~350 dogs have to be rescued

Commercial value

- This system can help address these statistics by alerting owners when their dog is waiting at the door.

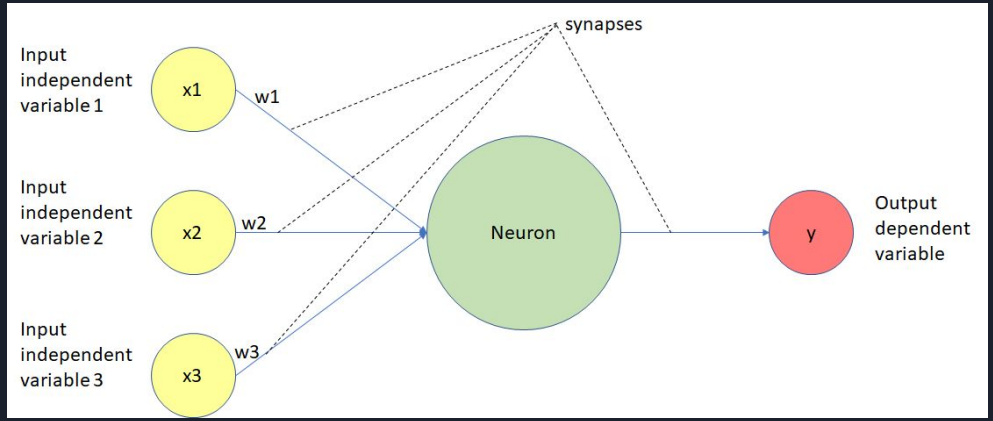
Neural Network Basics

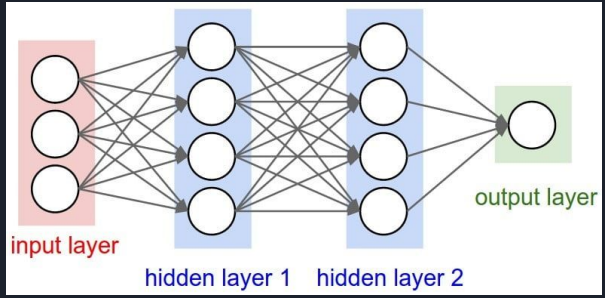
Neuron

- Nodes in the neural network that receive multiple input signals and performs calculations to produce an output with them.

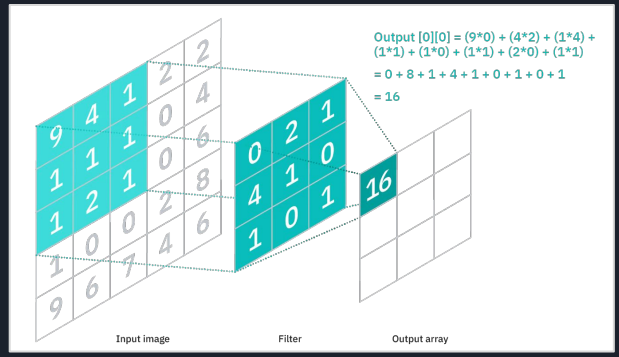
Deep Neural Networks (DNN)

- A hierarchical (layered) organization of neurons that are connected to other neurons.





Neural Network Basics cont.

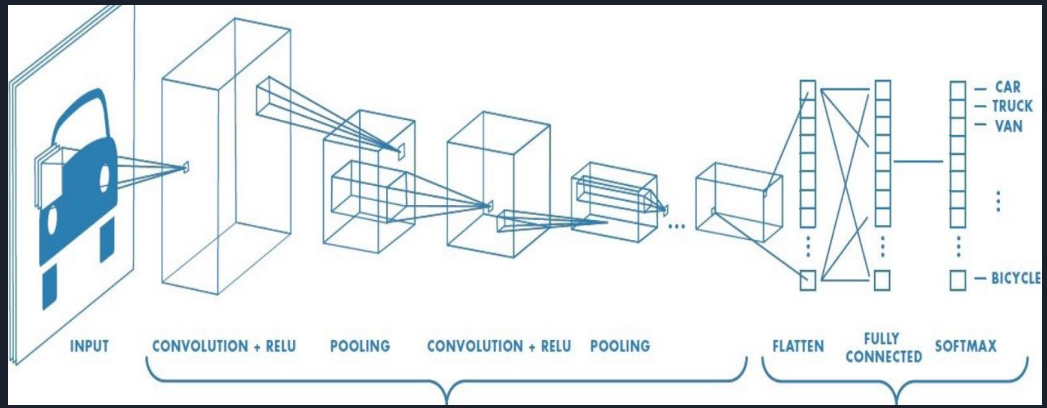


Convolutional Neural Network (CNN)

- A Deep Learning algorithm that looks for different aspects in an image, assigns an importance to each, and differentiates an aspect from others.

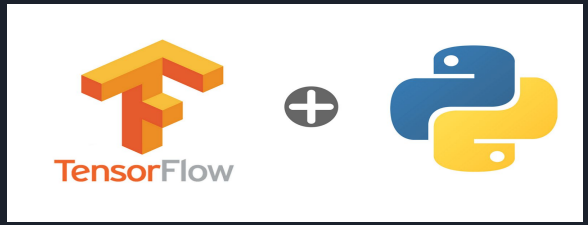
Transfer Learning

- Combining a CNN to differentiate attributes with a DNN to complete categorization with the CNN output



Tensorflow

Python with Tensorflow



Definition

- Open source machine learning platform especially for python

Benefits

- Tends to be faster than other modules
- Many example programs provided
- Pretrained model from the Tensorflow Model Zoo
- Using a pretrained model allows for much better results in object detection since the models are professionally built and trained on millions of images.

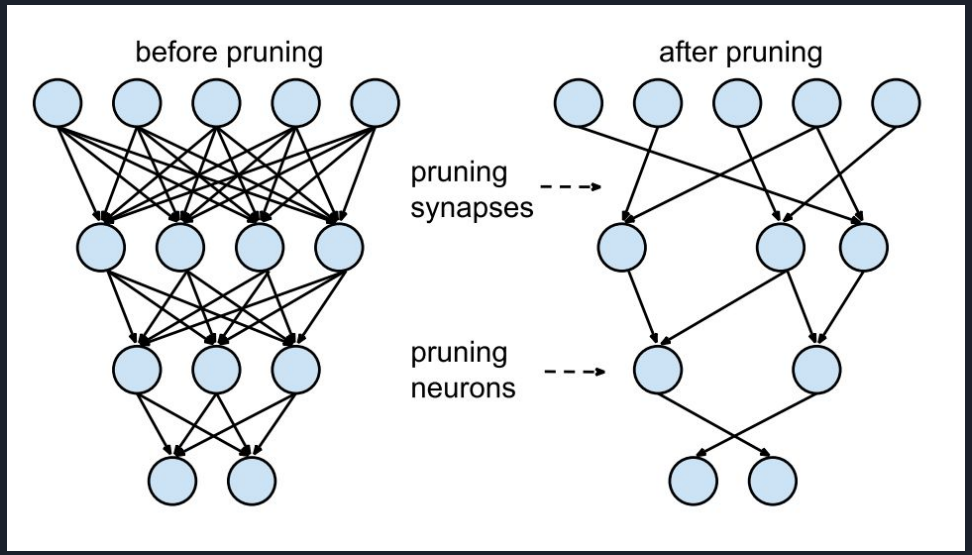
COCO-trained models

Model name	Speed (ms)	COCO mAP[^1]	Outputs
ssd_mobilenet_v1_coco	30	21	Boxes
ssd_mobilenet_v1_0.75_depth_coco ☆	26	18	Boxes
ssd_mobilenet_v1_quantized_coco ☆	29	18	Boxes
ssd_mobilenet_v1_0.75_depth_quantized_coco ☆	29	16	Boxes
ssd_mobilenet_v1_ppn_coco ☆	26	20	Boxes
ssd_mobilenet_v1_fpn_coco ☆	56	32	Boxes
ssd_resnet_50_fpn_coco ☆	76	35	Boxes
ssd_mobilenet_v2_coco	31	22	Boxes
ssd_mobilenet_v2_quantized_coco	29	22	Boxes
ssdlite_mobilenet_v2_coco	27	22	Boxes
ssd_inception_v2_coco	42	24	Boxes
faster_rcnn_inception_v2_coco	58	28	Boxes
faster_rcnn_resnet50_coco	89	30	Boxes
faster_rcnn_resnet50_lowproposals_coco	64		Boxes
rfcn_resnet101_coco	92	30	Boxes
faster_rcnn_resnet101_coco	106	32	Boxes
faster_rcnn_resnet101_lowproposals_coco	82		Boxes
faster_rcnn_inception_resnet_v2_atrous_coco	620	37	Boxes
faster_rcnn_inception_resnet_v2_atrous_lowproposals_coco	241		Boxes
faster_rcnn_nas	1833	43	Boxes
faster_rcnn_nas_lowproposals_coco	540		Boxes
mask_rcnn_inception_resnet_v2_atrous_coco	771	36	Masks
mask_rcnn_inception_v2_coco	79	25	Masks
mask_rcnn_resnet101_atrous_coco	470	33	Masks
mask_rcnn_resnet50_atrous_coco	343	29	Masks

Tensorflow Lite (tfLite)

Definition

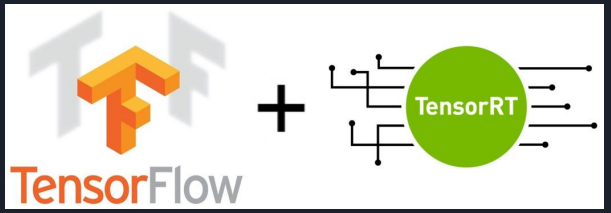
- Model optimizer for running algorithms on small devices (IOS, Android, Linux).



Optimization methods

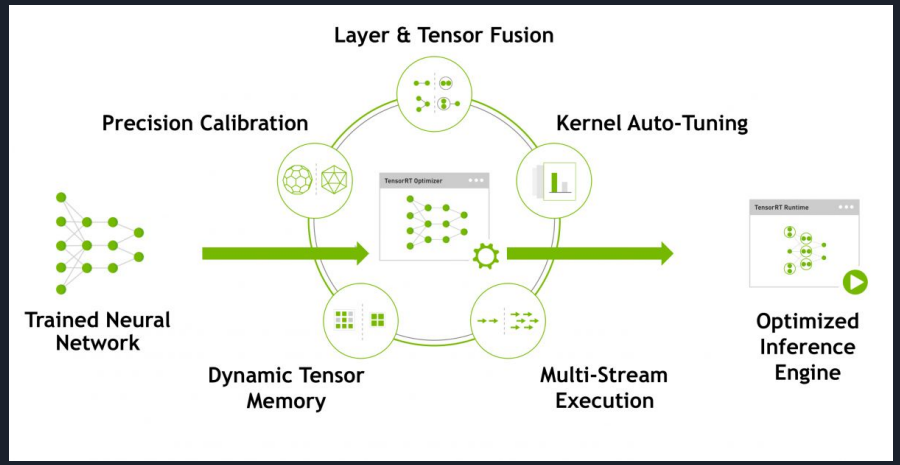
- Quantization
  - Reducing precision of numbers within the model (ex: downgrading from float32 to float16)
- Pruning
  - Removing minor model parameters to make for more efficient compression
- Clustering
  - Clustering layer weights together and using the centers of those clusters for new weights, thus reducing complexity

Tensorflow + TensorRT (TF-TRT)



Definition

- Machine Learning model optimizer for running models on devices with NVIDIA GPUs (primarily the Jetson family)

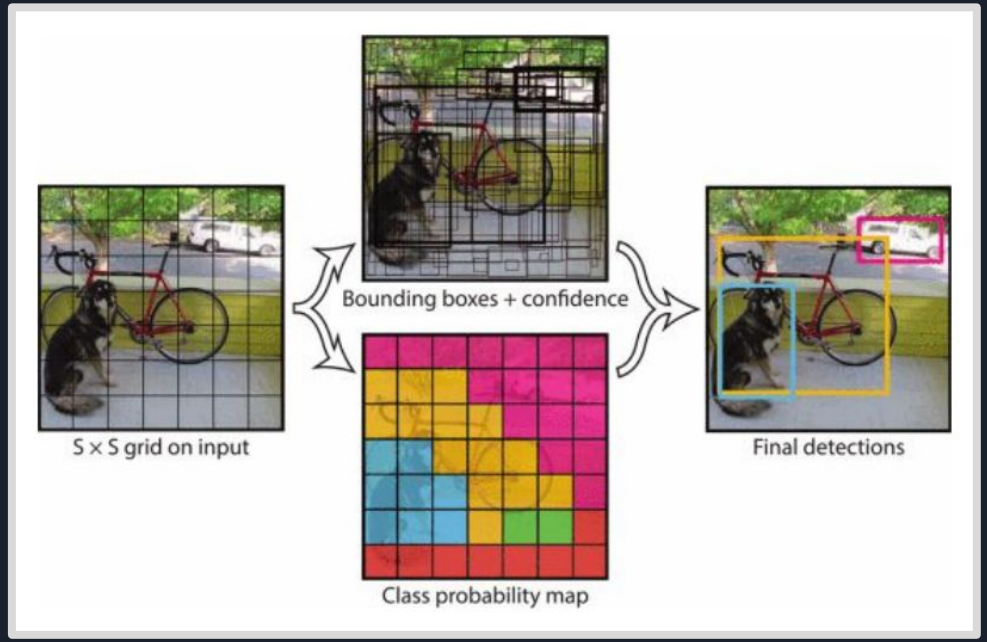


Optimization Techniques

- Reducing precision
  - Reducing precision of numbers within model
- Layer and Tensor Fusion
  - Optimizes GPU memory by fusing nodes of model
- Kernel auto-tuning
  - selecting best layers and algorithms based on kernel
- Dynamic Tensor Memory
  - allocating memory to tensors only while they are in use

YOLO

- You Only Look Once (YOLO) is a type of convolutional deep learning algorithm
- Divides the image into regions and and makes predictions for each region
- Single neural network applied to the full image
  - Informed by global context
  - Extremely fast
- Still preserves accuracy, making ideal for Nvidia Jetson Nano



Comparison to Other Detectors

- COCO (Common Objects in Context) is a large, universal dataset. The dataset consists of 328K images.
- Testing done on Pascal Titan X
- Demonstrates competitive accuracy while maintaining high speeds when compared to other popular methods

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

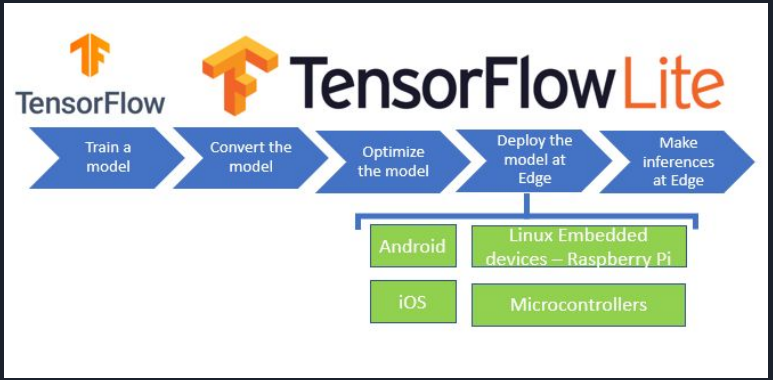
Workflow

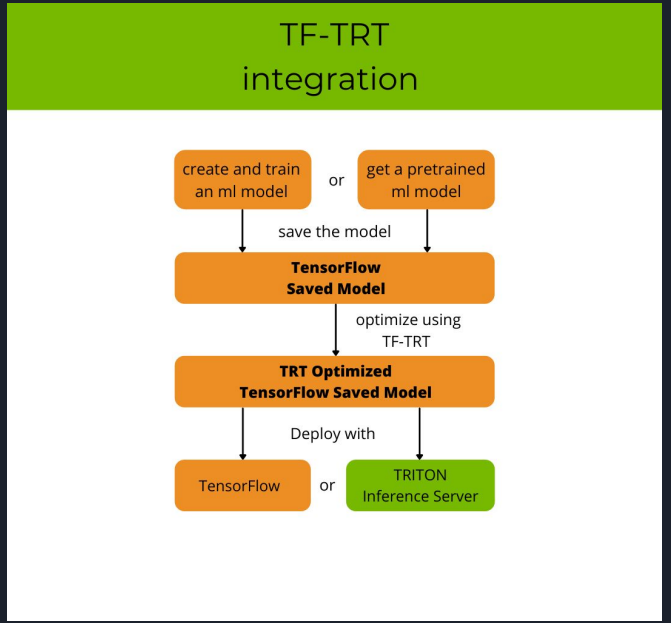
YOLO

- No conversion necessary
- Train model using AWS virtual machine, download weights that can be deployed on Jetson

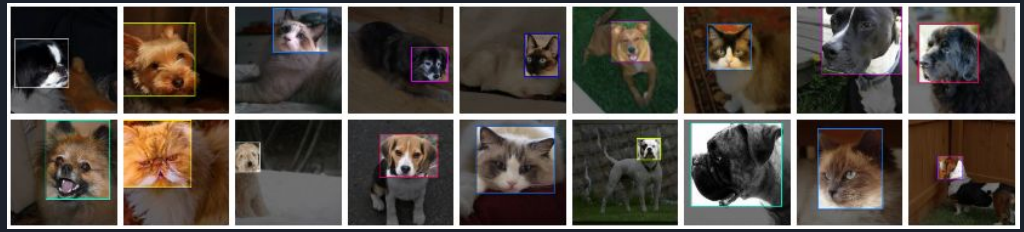
Tensorflow

- Train model using desktop Tensorflow, convert it with tfLite or tensorRT, and deploy it on the Jetson.





Dataset



- Oxford pet breed dataset
  - Pre annotated
  - 2,576 training images
  - 368 testing images
  - 37 breeds
    - No Bella breed (Golden Retriever)
- Custom pictures of Jason's dog Bella
  - Annotated by hand
  - 107 training images
  - 19 testing images

Measurements

Precision
- How many of the detections were true

Recall
- What percentage of all truths did model predict correctly?

Intersection over Union (IoU)
- Amount of overlap between prediction box and amount of overlap

Average Precision
- Compares precision and recall to calculate the accuracy of the model for each class

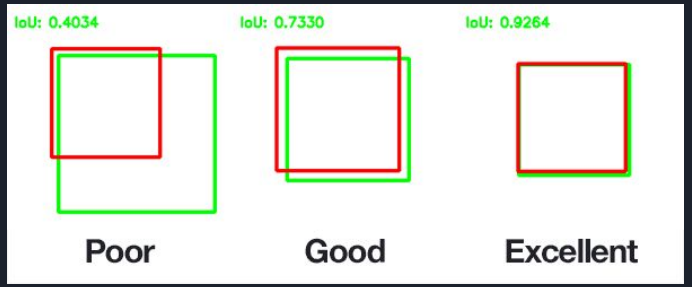
Mean Average Precision
- Mean of the average precisions for all classes

		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

$$\text{Precision} = \frac{tp}{tp + fp}$$

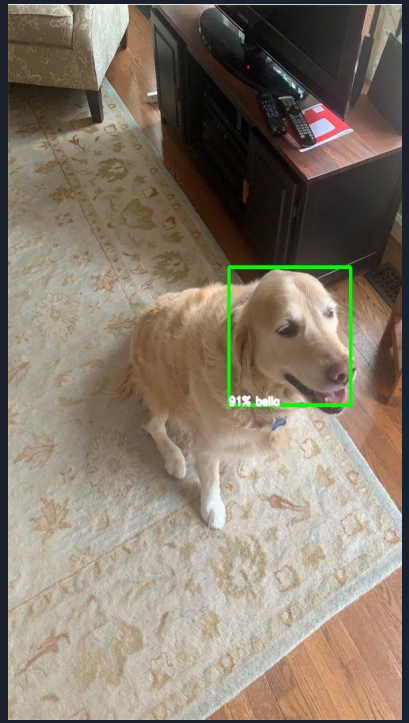
$$\text{Recall} = \frac{tp}{tp + fn}$$

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$

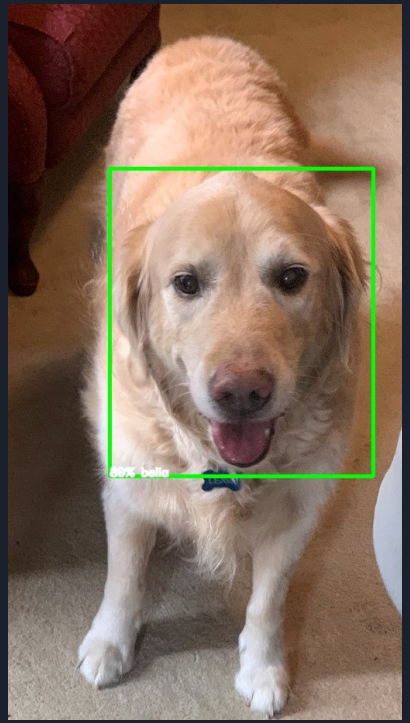


Testing and Analysis

tfLite



TensorRT



Yolo

Results for each algorithm/model

Algorithm	FPS	mAP	AP for Bella	Memory usage (GB)	GPU usage	CPU usage
Yolo	3.8	0.497	0.8474	1.026	100%	30% (Average for cores)
tfLite	0.78	0.620	0.420	0.127	0%	25% (100% one core)
TensorRT	3.82	0.846	0.848	**4.9	100%	50%
*Tensorflow		0.751	0.384			

*only run on desktop
**used Swap memory past 2GB

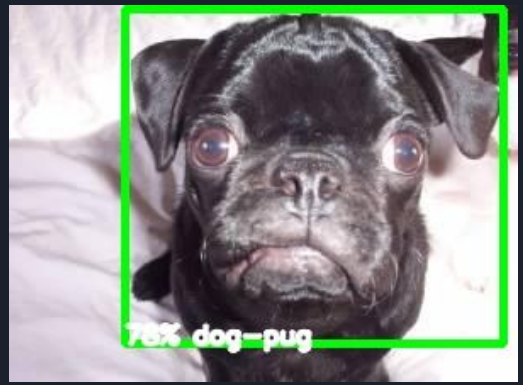
Understanding the results and what they mean

Surprising Results

- TensorRT beat Tensorflow, which was unexpected
- Tensorflow Lite lost more accuracy compared to TensorRT, however the difference in memory and computing usage may justify this accuracy loss depending on the situation.

Project as a whole

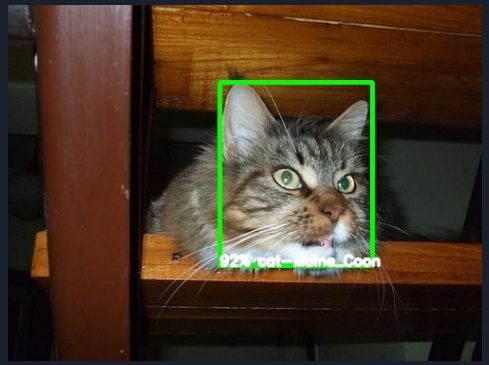
- Animal detection and identification is very much achievable on the NVIDIA Jetson with a frame rate that exceeds what is needed, which shows that this automated pet doorbell can very much be made into a working and affordable product.
- Though all algorithms worked, it is probably best to use the YOLO route if starting from scratch and Tensorflow if someone already has experience with it.

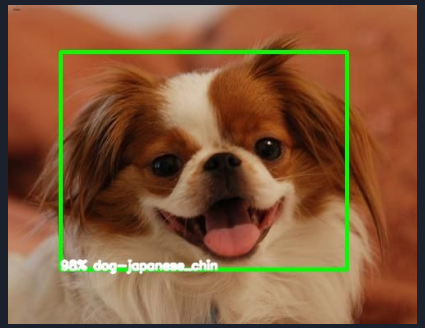


Directions for Future work

Limitations

- Dataset
  - Similar dog to Bella was not labelled
  - Bella training images possibly different than reality
- Tensorflow
  - TensorRT ran out of memory half way through optimizing
  - TF-TRT used 2.5x the available RAM



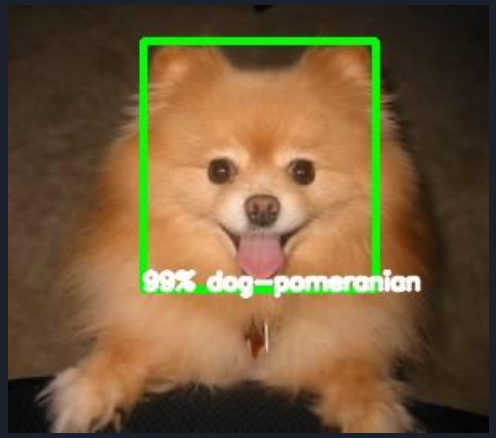


Future Work

- Polishing system
  - Create commercial version
  - Make user friendly UI
- Optimizing algorithms
  - Optimize source code
  - Use larger dataset
  - Use TensorRT on desktop to have enough memory to fully optimize model
  - Investigate TensorRT outperforming Tensorflow
- Other methods
  - Use dedicated TensorRT (less loading time than TF-TRT)
  - Consider using facial recognition specific algorithms instead

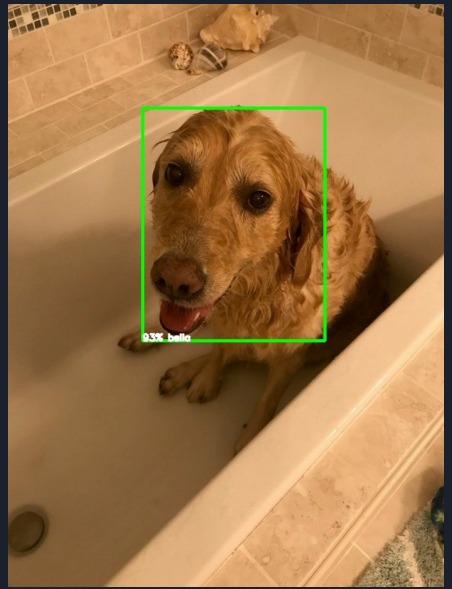
Recap of mentorship accomplishments

- Introduction
  - Installing a virtual machine
  - Linux OS and command line
  - kNN
    - Ensemble kNN
  - Github
  - Analyzing machine learning results
  - Background research on NVIDIA Jetson performance/use
  - Configure NVIDIA Jetson
- Project
  - Set out goals to complete project
  - YOLO (Jason)
    - Installing YOLO object detection and requirements
    - Creating object detection algorithms with trained weights from AWS machine
    - Use this model to perform tests on custom dataset
  - Tensorflow (Brandon)
    - Installing Tensorflow object detection along with CUDA and CUDNN
    - Creating object detection models on Tensorflow desktop with the Tensorflow Model Zoo
    - How to train these object detection models
    - Exporting them as tfLite
    - Using the tfLite model with a provided helper script
    - Converting tensorRT model and using it directly with no helper script
  - Converting predictions into usable metrics



Lessons and skills learned

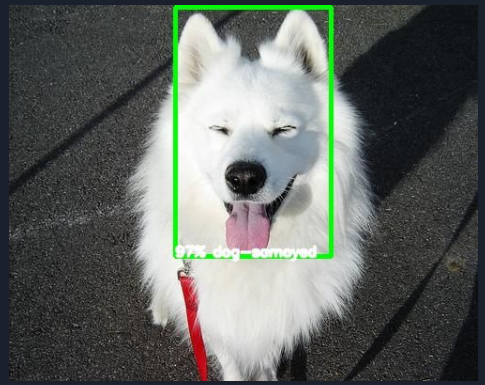
- How to
  - use linux for software development
  - use github for software collaboration
  - build a kNN algorithm from scratch
    - build ensemble algorithms
  - start/layout project development for a client
- How object detection data is saved and analyzed for metrics
- Jason
  - Experience with YOLO algorithm and knowledge that can be applied to other work
  - Practice with debugging and solving problems in this kind of workspace
- Brandon
  - Deep dive into the Tensorflow ecosystem
    - Can now be transferred to other types of models and applications
  - Utilizing opencv to read in and save images and videos



Acknowledgements

Mentor: Mr. Alberghini
- Machine Learning Development Operations Engineer with the MITRE corporation

Mrs. Vobrak
- Research teacher that allowed us to engage in the mentorship from the beginning

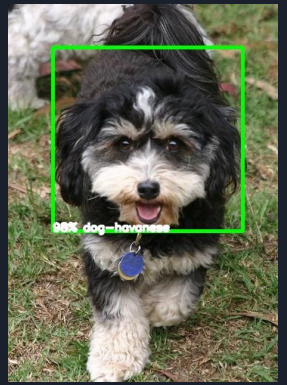


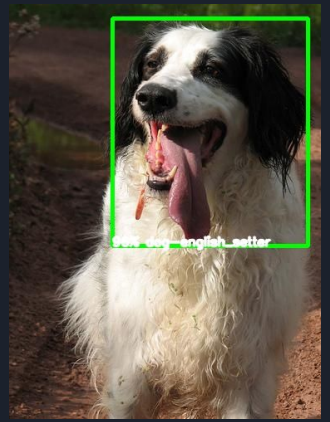
References

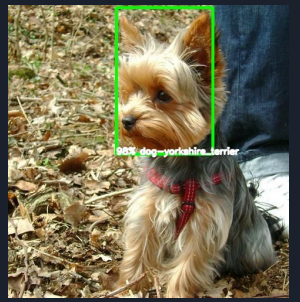
- Labs, S. (2019, March 21). Understanding deep learning: DNN, RNN, LSTM, CNN and R-CNN. Medium. Retrieved November 29, 2021, from https://medium.com/@sprhlabs/understanding-deep-learning-dnn-rnn-lstm-cnn-and-r-cnn-6602ed94dbff.
- Higa, X. (2019). Dog Breed Classification Using Convolutional Neural Networks: Interpreted Through a Lockean Perspective (Doctoral dissertation, Lake Forest College).
- Verma, G. K., & Gupta, P. (2018). Wild animal detection using deep convolutional neural network. In Proceedings of 2nd international conference on computer vision & image processing (pp. 327-338). Springer, Singapore.
- Nicholas Renotte. (2021, April 9). Tensorflow Object Detection in 5 Hours with Python | Full Course with 3 Projects [video]. Youtube. https://www.youtube.com/watch?v=yqkISICHH-U
- Koech, K. E. (2022, March 28). On object detection metrics with worked example. Medium. Retrieved April 3, 2022, from https://towardsdatascience.com/on-object-detection-metrics-with-worked-example-216f173ed31e
- Darknet github: https://github.com/AlexeyAB/darknet
- YOLO tutorial: https://medium.com/analytics-vidhya/train-a-custom-yolov4-tiny-object-detector-using-google-colab-b58be08c9593#4be1

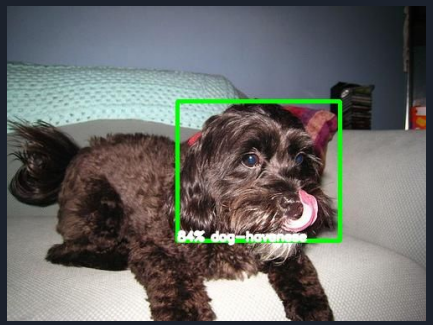
Questions?

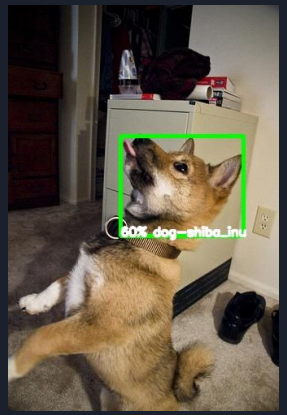
Cute dog examples

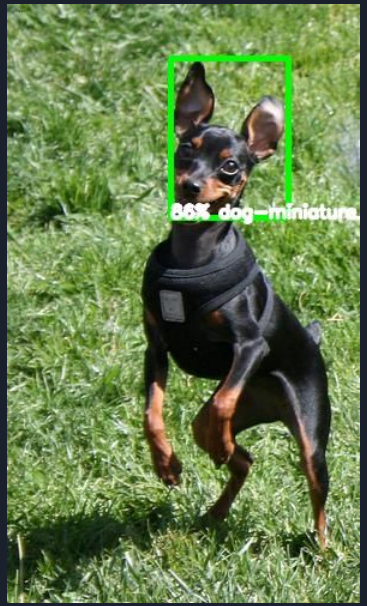




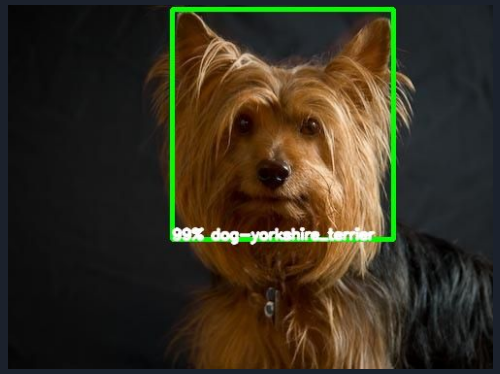


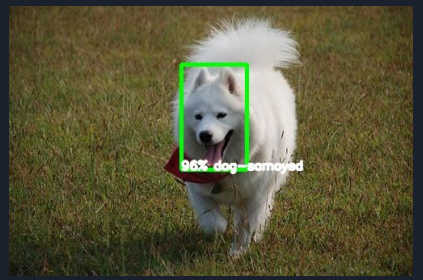


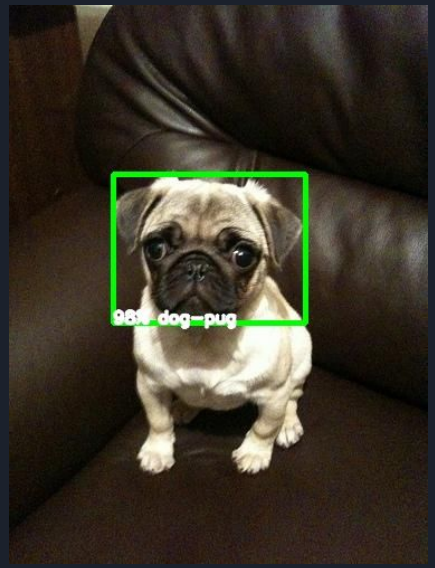




More cute dogs







Cute cat examples