Study on Object Detection and Few-shot Learning

Supervisor: Dr. Jana Kosecka
Presented by: Anowarul Kabir

Outline

1. Object Detection

→ Approach

→ Result

2. Few-shot Leaning

→ Approach

→ Result

Object Detection

* Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks; Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun

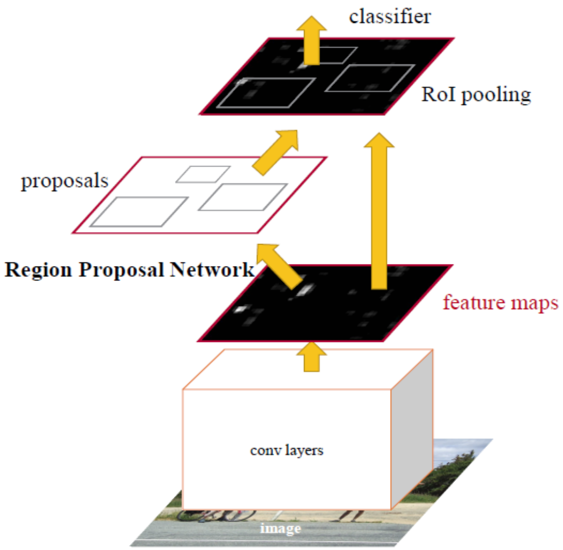
* Training set: Ground truth image with bounding box.

* Convolutional feature extractor: It uses Convolutional Neural Network (e.g. VGG16) for extracting the features from the image.

* Region Proposal Network (RPN): RPN takes the feature maps as input and uses fixed number of boxes (called Anchors) to output some object proposals.

* Classifier: Finally the classifier and regressor find the class and object occurrence from region proposals using regions of interest layer.

* Objective function: Minimizing a loss function in two steps.
The classification loss over two classes (object or not object).
The regression loss over the position of predicted box and
ground truth box.



Object Detection with Fine-grained Data

* Fine-grained data: Involves subordinate categories of a class. e.g. species of cats, dogs, birds, plants

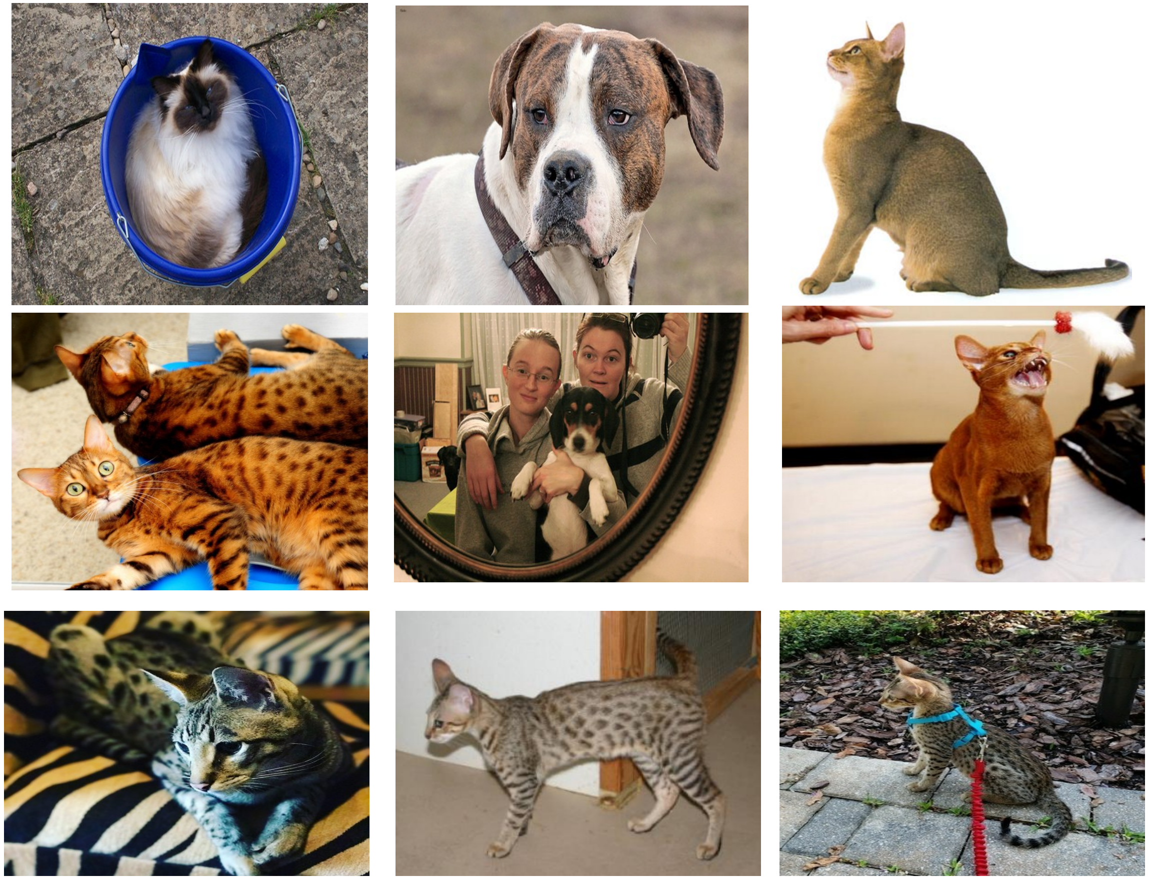
* Cats and Dogs; Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, C. V. Jawahar

* Dataset statistics:

- 37 classes (Cat: 12, Dog: 25)
- Train set: 2208 images
- Validation set: 552 images
- Test set: 920

* Extended dataset statistics:

- 38 classes (Cat: 13, Dog: 25)
- Train set: 2328 images
- Validation set: 582 images
- Test set: 970
- Newly added 200 images



Results

On pet dataset

Feature extractor	Set	Images	mAP@IoU= 0.5:0.95	mAP@IoU= 0.5	mAP@IoU= 0.75
ResNet101	Validation	552	0.705	0.875	0.831
ResNet101	Test	920	0.706	0.908	0.853

On extended dataset

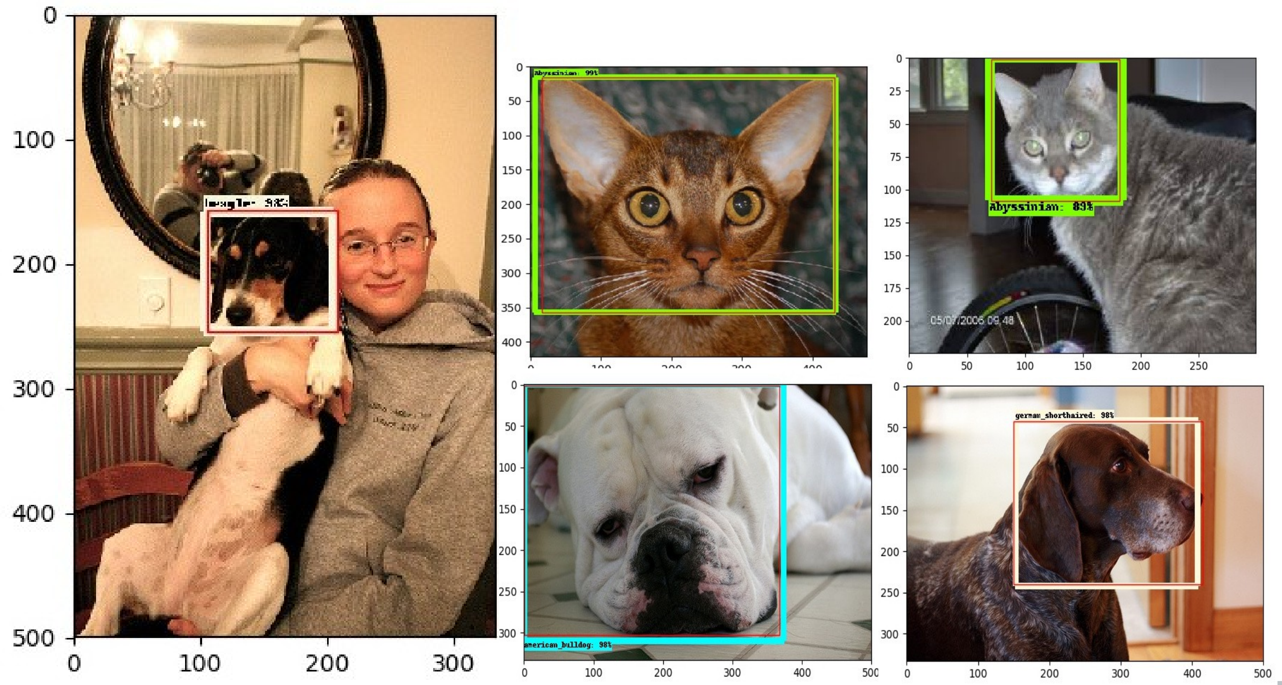
Feature extractor	Set	Images	mAP@IoU= 0.5:0.95	mAP@IoU= 0.5	mAP@IoU= 0.75
ResNet101	Validation	582	0.718	0.907	0.853
ResNet101	Test	970	0.712	0.912	0.849

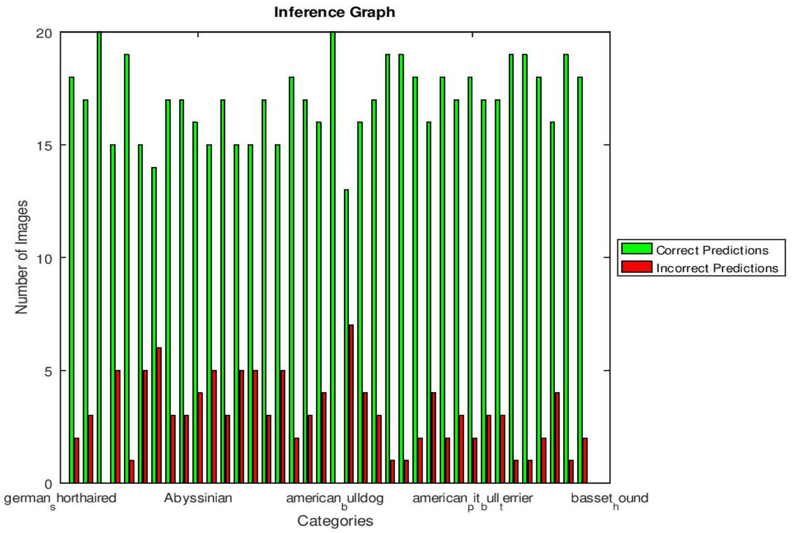
Outputs

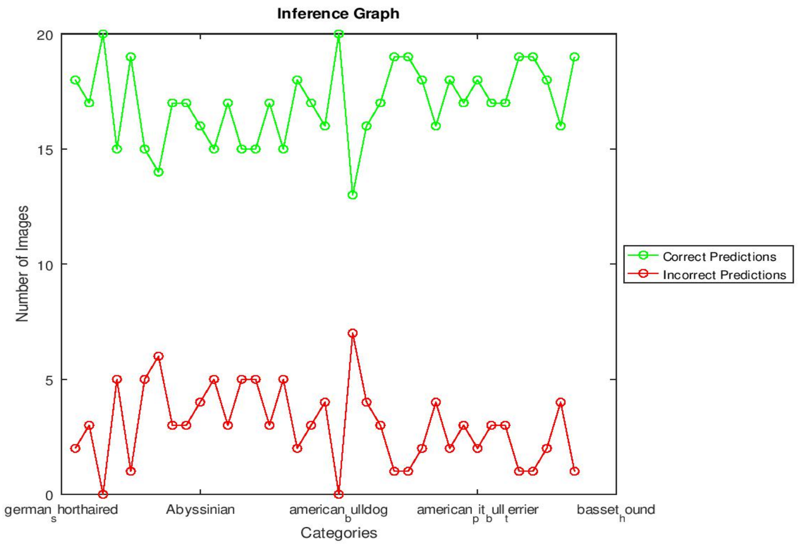
* At right: The graph is showing correct and incorrect prediction number per classes (each class contains 20 images)

* At Below: Some example outputs generated by the detector. Each output has three components:

1. Ground truth bounding box (red)
2. Predicted bounding box (other colors)
3. Predicted class name and objectness score







Few-shot Learning using Prototypical Networks

* Few-shot learning: The model learns to classify new classes where the training set involves a small number of examples per class.

* Prototypical Networks for Few-shot Learning; Jake Snell, Kevin Swersky, Richard S. Zemel

* Prototypical Networks: It learns a metric space to classify new examples by performing distance measurement from prototype representation of each class.

* Prototype representation: For each input we first map the input into an embedding space using neural network and take the mean in the embedding space for each class. Each of this mean is called prototype representation for each class.

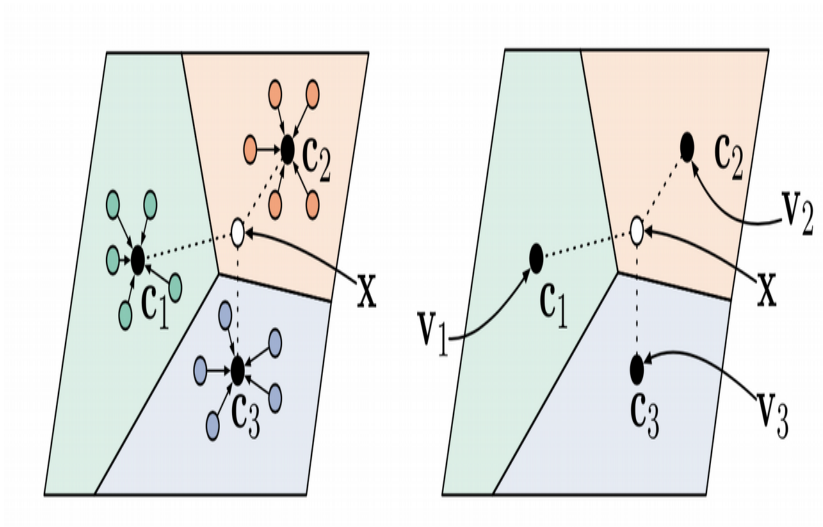
* Classification prediction: For a query point we will calculate the nearest prototype embedding which will eventually be the class prediction.

* Objective function: Loss *J* is calculated by:

$$J = J + \frac{1}{N_C N_Q}[d(f_\phi(x), c_k) + \log(\sum_{k'} (\exp(-d(f_\phi(x), c_{(k')}))))]$$

$c_k = prototype, f_\phi = embedding\ function, x = an\ example, d = distance,$

$N_C = number\ of\ classes\ per\ episode, N_Q = number\ of\ query\ example\ per\ class$



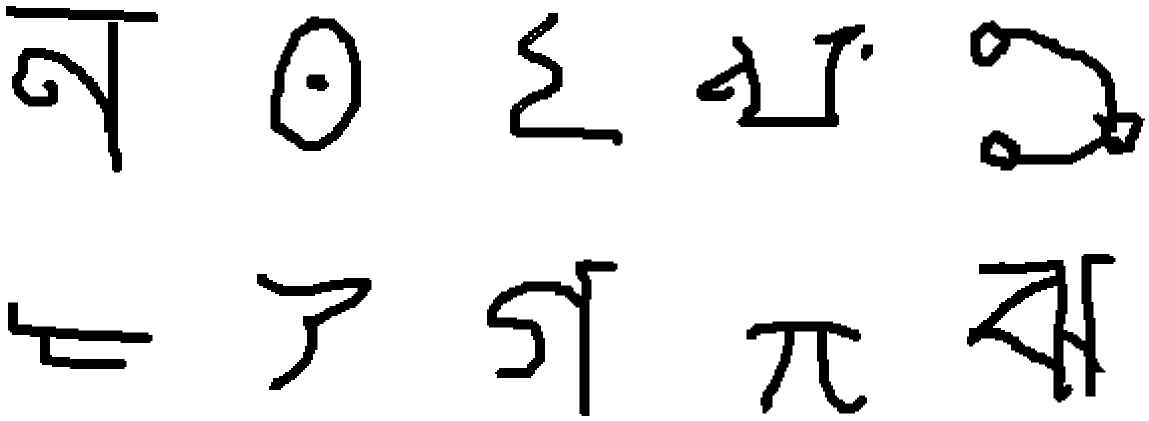
(a) Few-shot

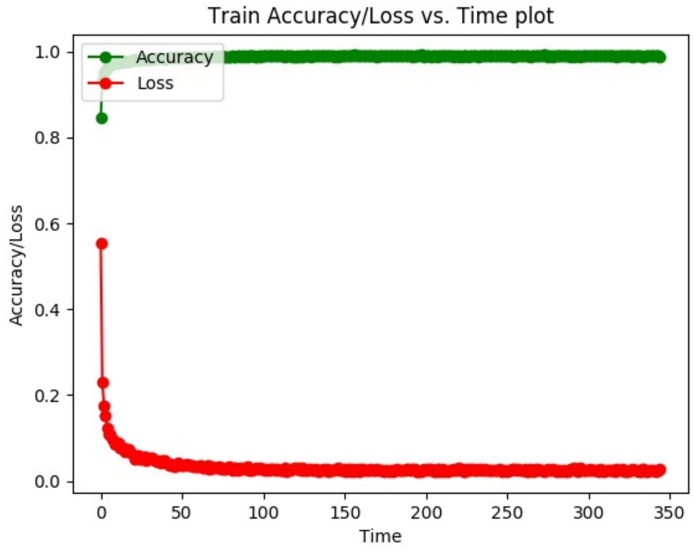
(b) Zero-shot

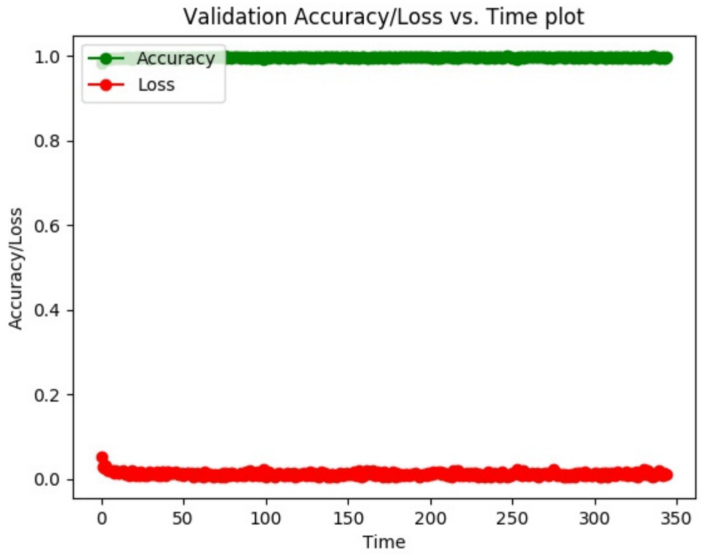
Results

*The Omniglot Challenge: A 3-Year Progress Report; Brenden M. Lake, Ruslan Salakhutdinov, Joshua B. Tenenbaum

	Output Process	Accuracy	Loss
Train	60-way 5-shot	0.994	0.012
Validation	60-way 5-shot	0.990	0.024




Train Accuracy/Loss vs. Time plot


Validation Accuracy/Loss vs. Time plot

Acknowledgment

* I would like to express great appreciation to Tensorflow Object Detection Api.

* I would like to offer thanks to the authors of the Prototypical Networks paper for making their code public.