



DEEP
LEARNING
INSTITUTE



ODTÜ
METU

Object Detection with DIGITS

Dr. Alptekin Temizel
DLI Certified Instructor
Associate Professor, Graduate School of Informatics, METU

8 January 2018

WHAT THIS LAB IS

- Discussion/Demonstration of object detection using Deep Learning
- Hands-on exercises using Caffe and DIGITS

TAKE AWAYS

- You can setup your own object detection workflow in Caffe and adapt it to your use case
- Know where to go for more info
- Familiarity with Caffe

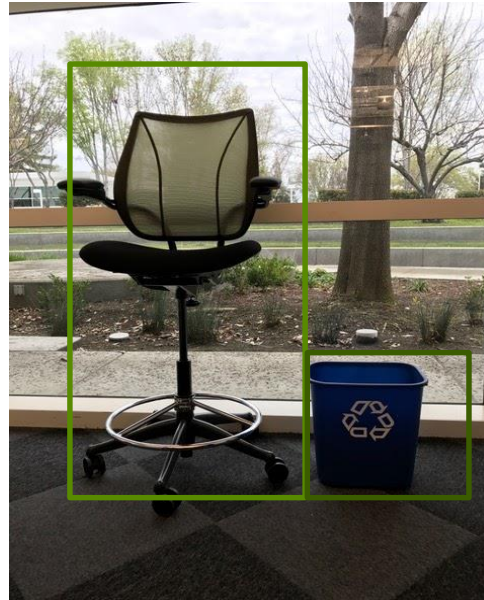
OBJECT DETECTION

COMPUTER VISION TASKS

**Image
Classification**



Object Detection



**Image
Segmentation**



(inspired by a slide found in cs231n lecture from Stanford University)

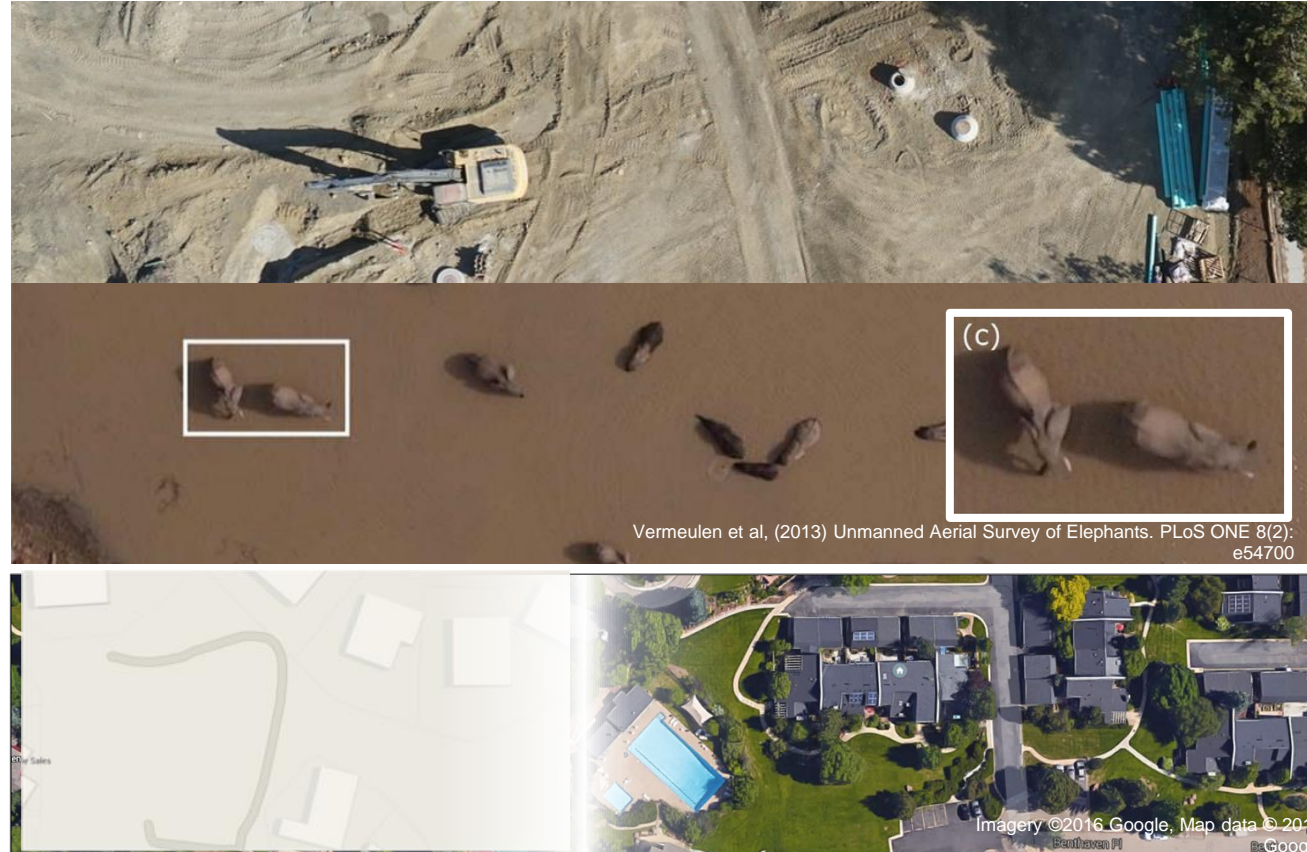
OBJECT DETECTION

- Object detection can identify and classify one or more objects in an image
- Detection is also about localizing the extent of an object in an image
 - Bounding boxes / heat maps
- Training data must have objects within images labeled
 - Can be hard to find / produce training dataset

OBJECT DETECTION IN REMOTE SENSING IMAGES

Broad applicability

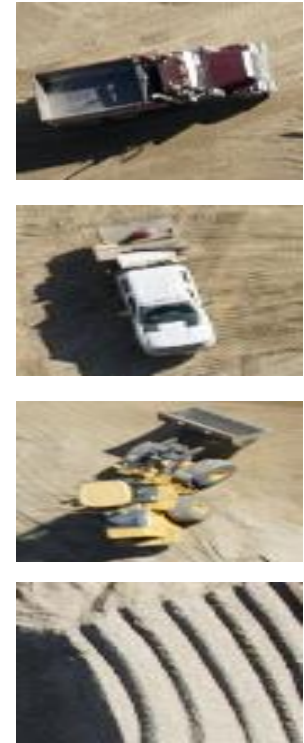
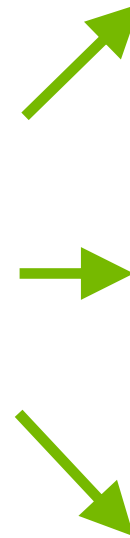
- Commercial asset tracking
- Humanitarian crisis mapping
- Search and rescue
- Land usage monitoring
- Wildlife tracking
- Human geography
- Geospatial intelligence production
- Military target recognition



OBJECT DETECTION

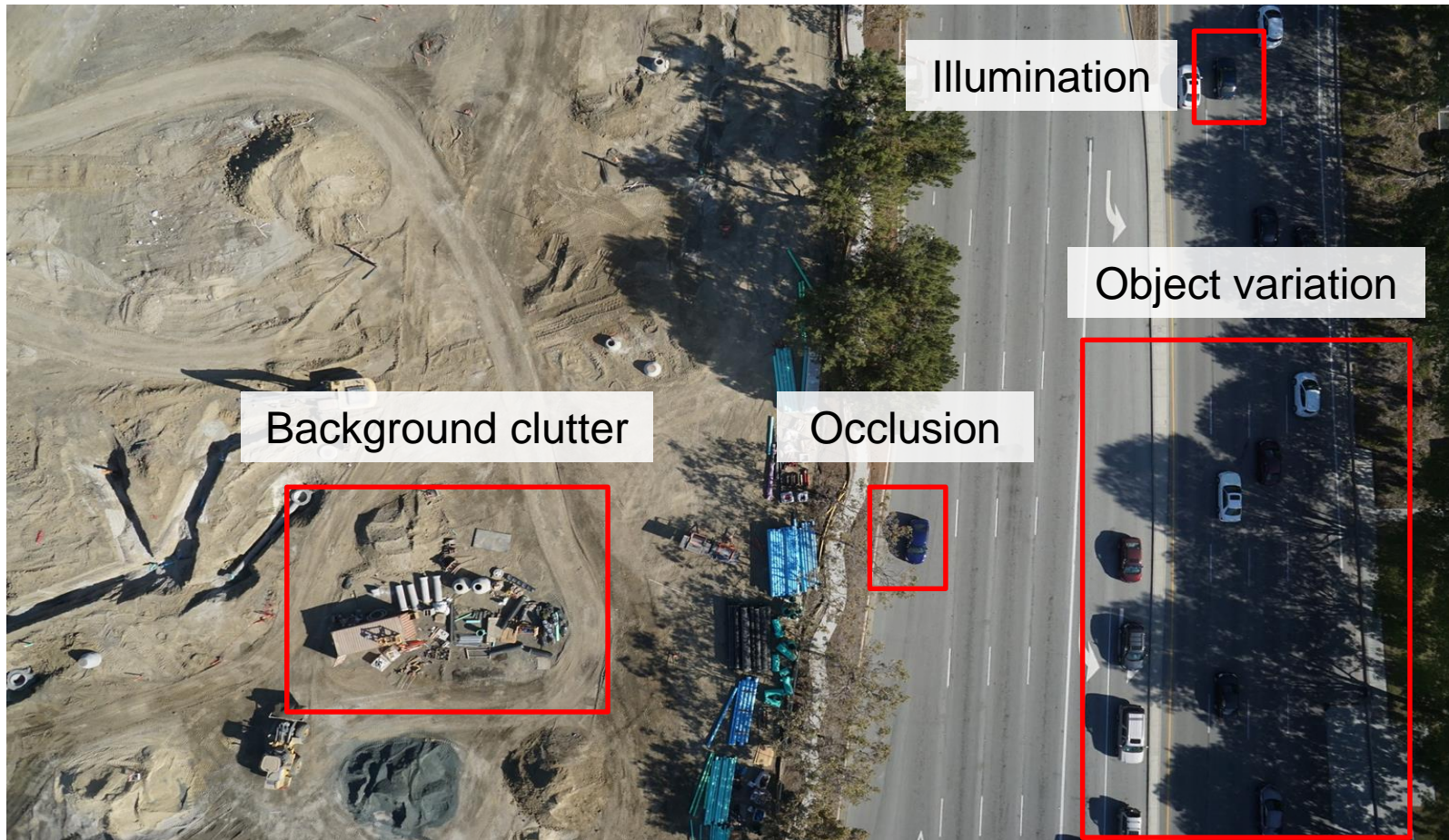


GENERATE CANDIDATE DETECTIONS



**EXTRACT
PATCHES**

CHALLENGES FOR OBJECT DETECTION



CAFFE

WHAT IS CAFFE?

An open framework for deep learning developed by the Berkeley Vision and Learning Center (BVLC)



- Pure C++/CUDA architecture
- Command line, Python, MATLAB interfaces
- Fast, well-tested code
- Pre-processing and deployment tools, reference models and examples
- Image data management
- Seamless GPU acceleration
- Large community of contributors to the open-source project

caffe.berkeleyvision.org
<http://github.com/BVLC/caffe>

CAFFE FEATURES

Deep Learning model definition

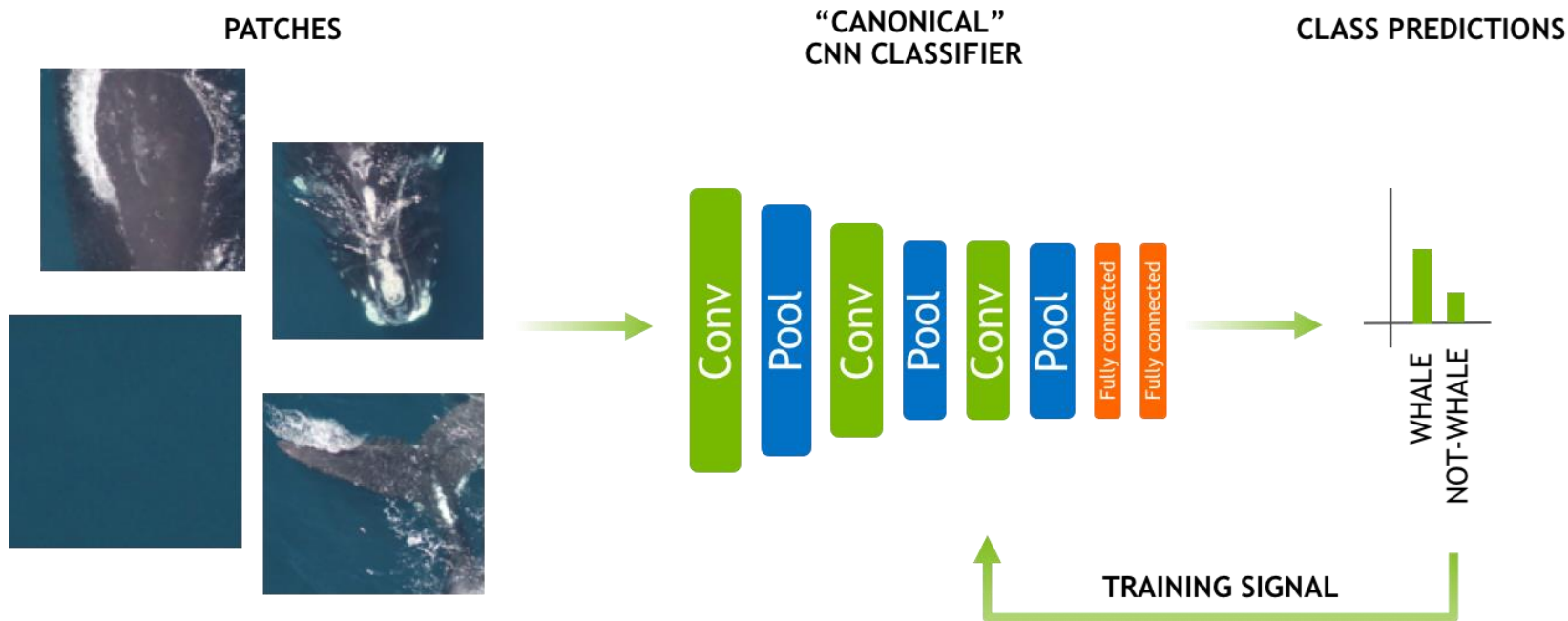
Protobuf model format

- Strongly typed format
- Human readable
- Auto-generates and checks Caffe code
- Developed by Google
- Used to define network architecture and training parameters
- No coding required!

```
name: "conv1"
type: "Convolution"
bottom: "data"
top: "conv1"
convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
        type: "xavier"
    }
}
```

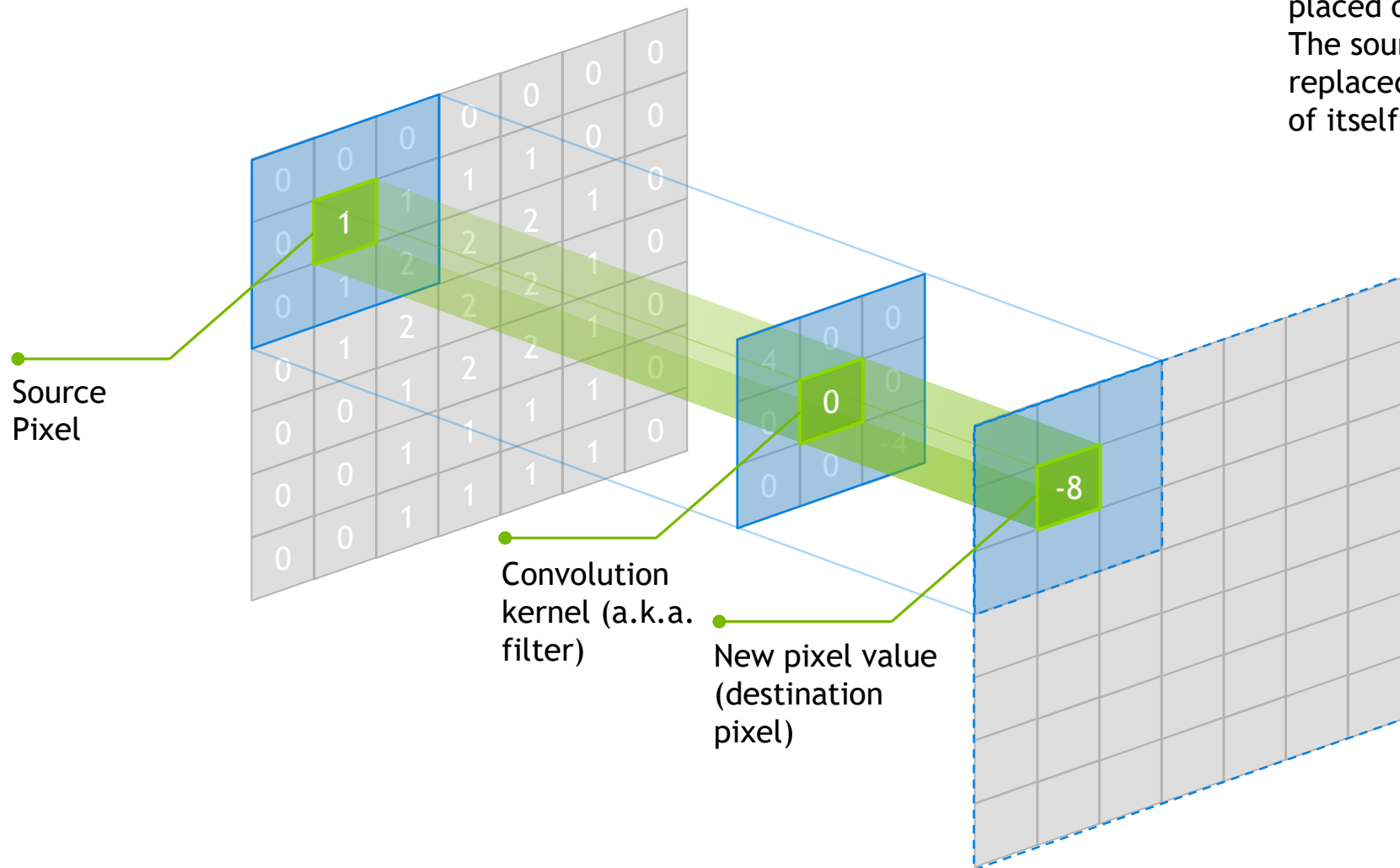

LAB DISCUSSION / OVERVIEW

TRAINING APPROACH 1 - SLIDING WINDOW



CONVOLUTION

Center element of the kernel is placed over the source pixel. The source pixel is then replaced with a weighted sum of itself and nearby pixels.



TRAINING APPROACH 1 - POOLING

- Pooling is a down-sampling technique
 - Reduces the spatial size of the representation
 - Reduces number of parameters and number of computations (in upcoming layer)
 - Limits overfitting
- No parameters (weights) in the pooling layer
- Typically involves using MAX operation with a 2 X 2 filter with a stride of 2

TRAINING APPROACH 1 - DATASETS

- Two datasets
 - First contains the wide area ocean shots containing the whales
 - This dataset is located in data_336x224
 - Second dataset is ~4500 crops of whale faces and an additional 4500 random crops from the same images
 - We are going to use this second dataset to train our classifier in DIGITS
 - These are the “patches”

TRAINING APPROACH 1 - TRAINING

- Will train a simple two class CNN classifier on training dataset
- Customize the Image Classification model in DIGITS:
 - Choose the Standard Network "AlexNet"
 - Set the number of training epochs to 5

TRAINING APPROACH 1 - SLIDING WINDOW

- Will execute code shown below
 - Example of how you feed new images to a model
 - In practice, would write code in C++ and use TensorRT

```
import numpy as np
import matplotlib.pyplot as plt
import caffe
import time
```

```
MODEL_JOB_NUM = '20160920-092148-8c17' ## Remember to set this to be the job number for your model
DATASET_JOB_NUM = '20160920-090913-a43d' ## Remember to set this to be the job number for your dataset
```

```
MODEL_FILE = '/home/ubuntu/digits/digits/jobs/' + MODEL_JOB_NUM + '/deploy.prototxt' # Do not change
PRETRAINED = '/home/ubuntu/digits/digits/jobs/' + MODEL_JOB_NUM + '/snapshot_iter_270.caffemodel' # Do not change
MEAN_IMAGE = '/home/ubuntu/digits/digits/jobs/' + DATASET_JOB_NUM + '/mean.jpg' # Do not change
```

```
# load the mean image
mean_image = caffe.io.load_image(MEAN_IMAGE)
```

```
# Choose a random image to test against
RANDOM_IMAGE = str(np.random.randint(10))
IMAGE_FILE = 'data/samples/w_' + RANDOM_IMAGE + '.jpg'
```

CAPTURING MODEL / DATASET NUMBER

1. Model number can be found here

1. Dataset number will be different, but found in same location

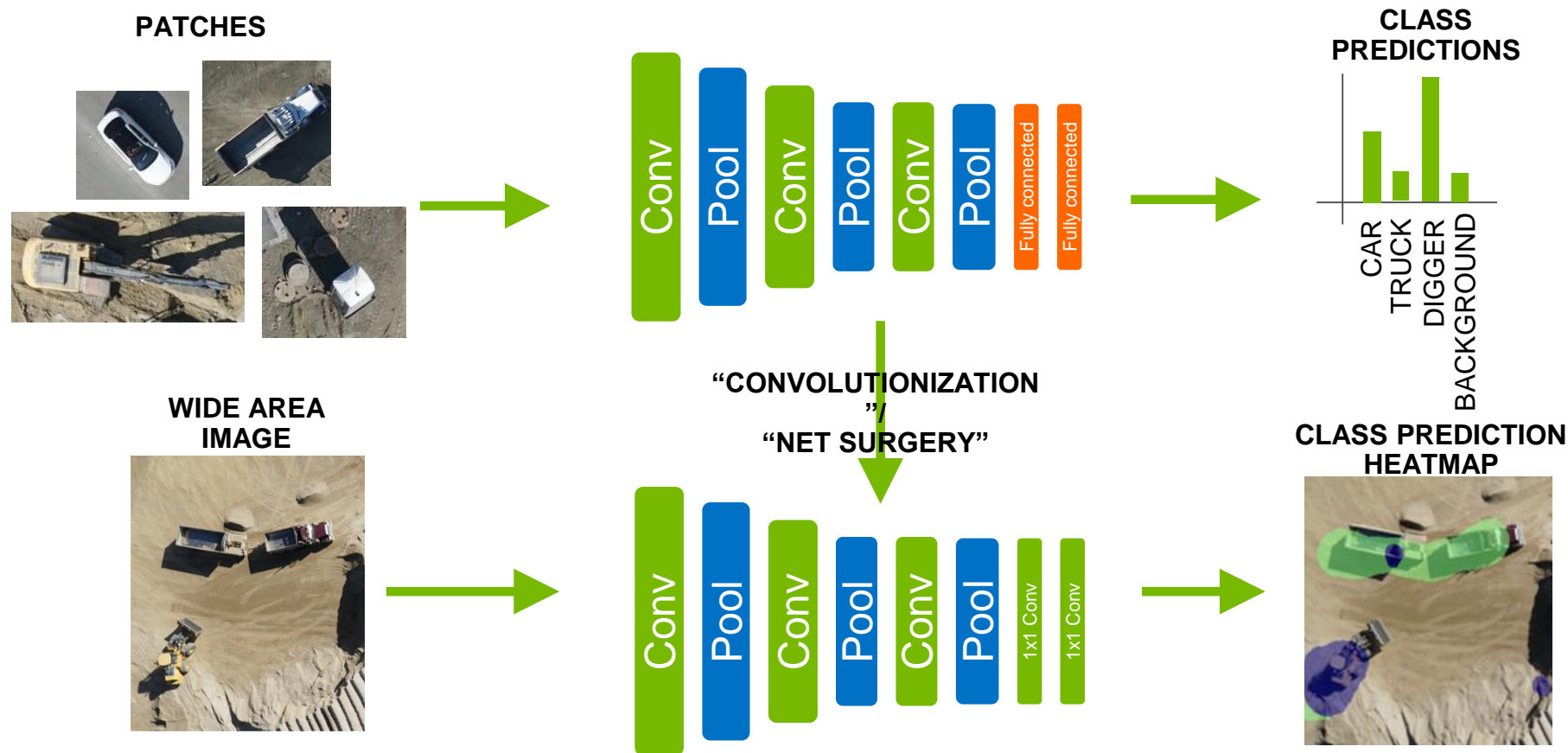
The screenshot shows the AWS Digits console interface. The browser address bar displays the URL: `ec2-54-161-216-120.compute-1.amazonaws.com:5000/models/20160722-180900-cc67`. The console header includes the 'DIGITS' logo, the 'Generic Image Model' tab, and user information 'ckillam (Logout)' with 'Info' and 'About' links. The main content area shows the model name 'whale_detectnet' with the owner 'ubuntu'. There are 'Clone Job' and 'Delete Job' buttons. Below this, there are three sections: 'Job Directory' showing the path `/home/ubuntu/digits/digits/jobs/20160722-180900-cc67` and 'Disk Size'; 'Dataset' showing 'whale_full'; and 'Job Status Done' showing 'Initialized at Jul 22 2016, 06:09:00 PM (1)'.

TRAINING APPROACH 2

- Candidate generation and classification
- Alternative to classification CNN using sliding window approach
- Discussed in lab instructions, but no lab task associated with this approach

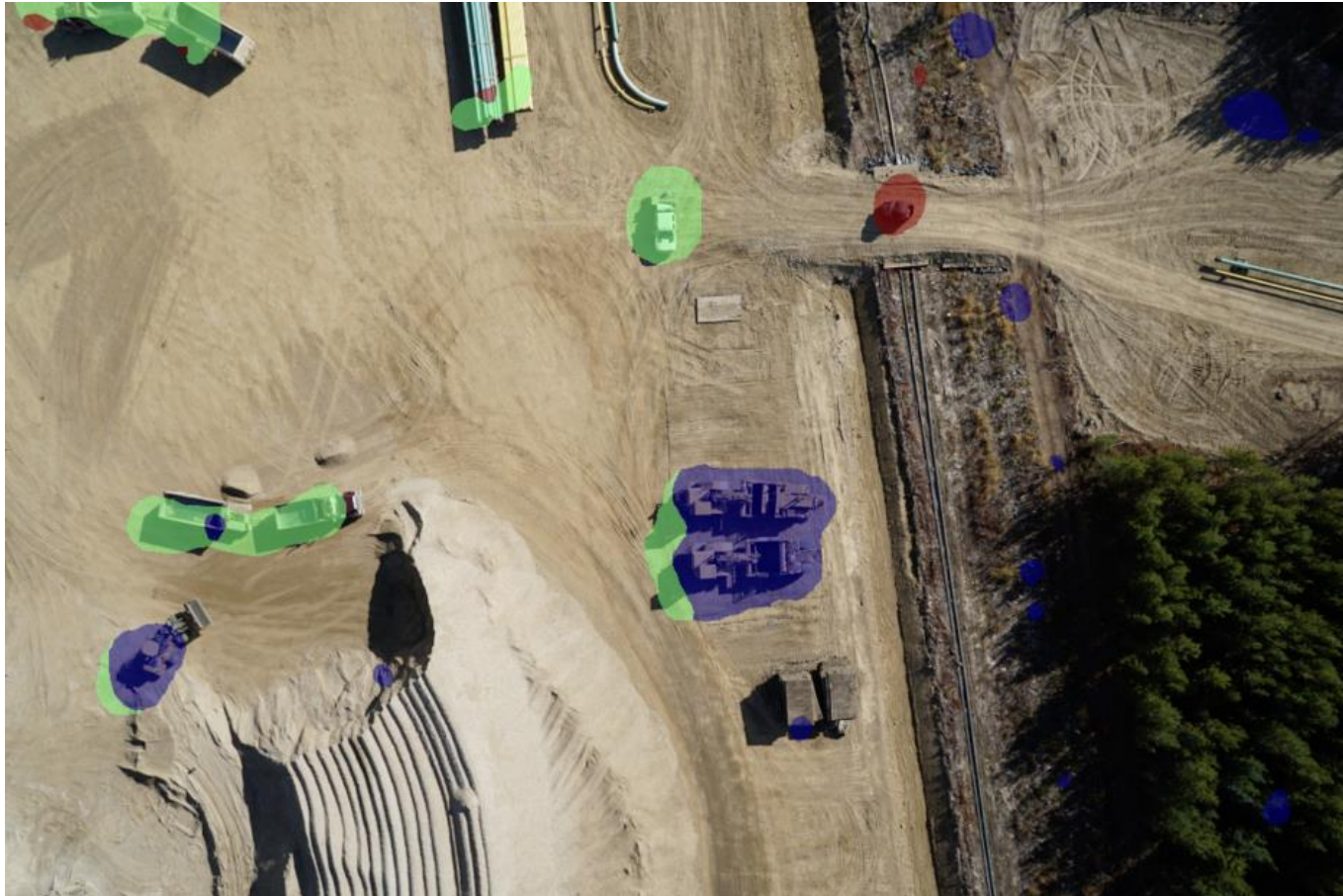
TRAINING APPROACH 3

Fully-Convolutional Network (FCN)

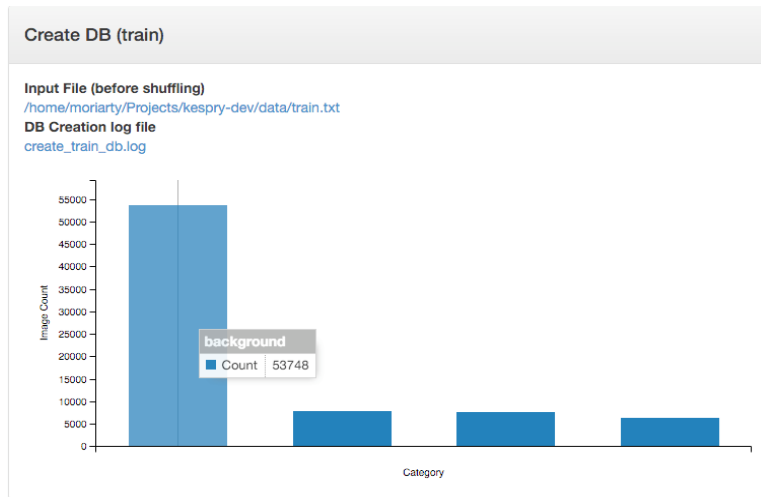


TRAINING APPROACH 3 - EXAMPLE

Alexnet converted to FCN for four class classification



TRAINING APPROACH 3 - FALSE ALARM MINIMIZATION

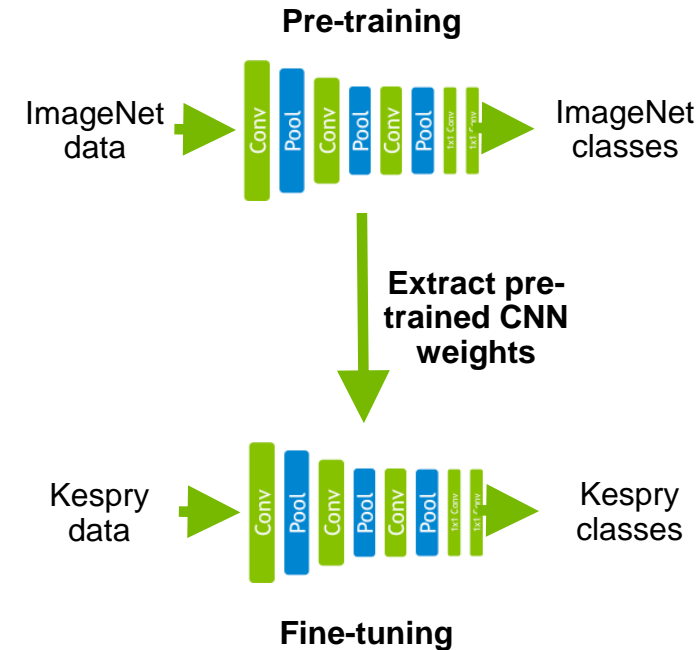


$$E = -\frac{1}{N} \sum_{n=1}^N H_{l_n} \log(\hat{p}_n)$$

Imbalanced dataset and
InfogainLoss

Data augmentation

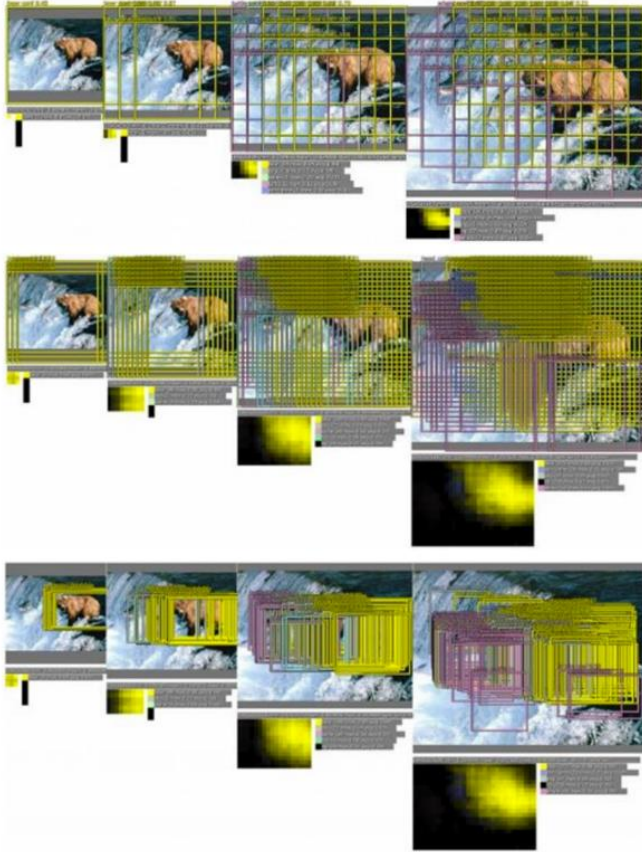
Random scale, crop, flip,
rotate



Transfer learning

TRAINING APPROACH 3 - INCREASING FCN PRECISION

Multi-scale and shifted inputs



OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks, Sermanet et al., 2014

greedy merging
procedure

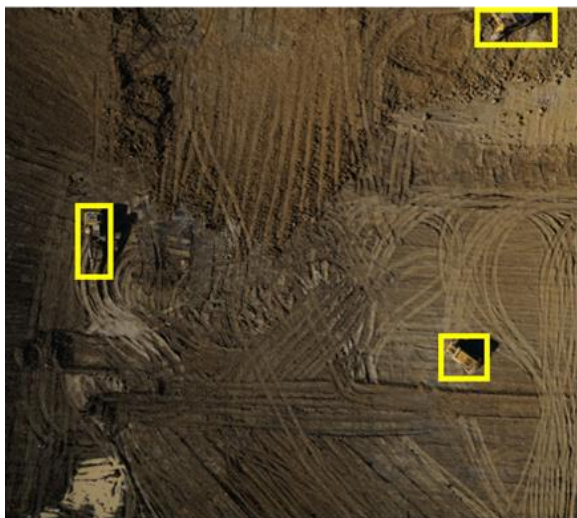


TRAINING APPROACH 4 - DETECTNET

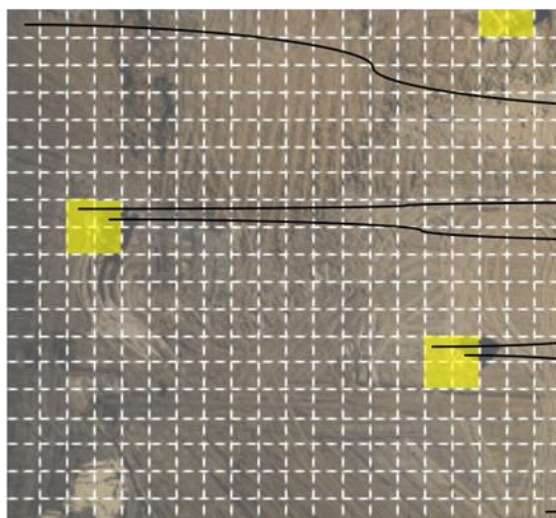
- Train a CNN to simultaneously
 - Classify the most likely object present at each location within an image
 - Predict the corresponding bounding box for that object through regression
- Benefits:
 - Simple one-shot detection, classification and bounding box regression pipeline
 - Very low latency
 - Very low false alarm rates due to strong, voluminous background training data

TRAINING APPROACH 4 - DETECTNET

Train on wide-area images with bounding box annotations



Training image with bounding box annotations



Bounding boxes mapped to grid squares

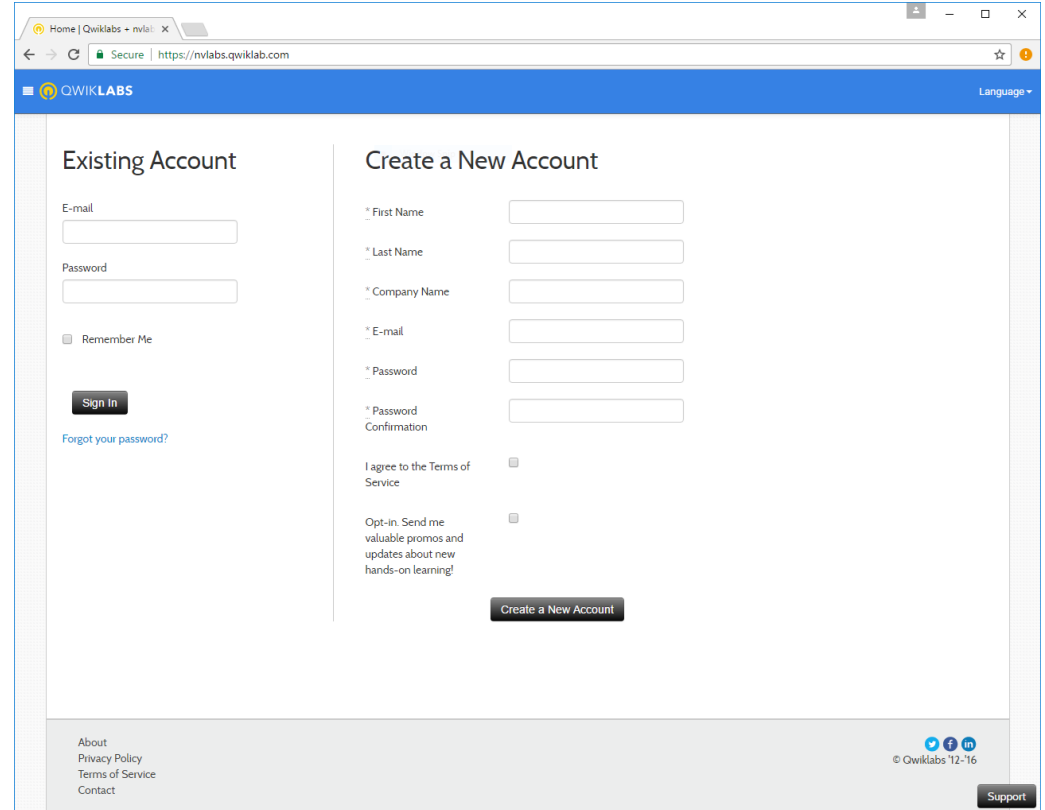
Bounding box coordinates in pixels relative to center of grid square

class	x_1	y_1	x_2	y_2	coverage
dontcare	0	0	0	0	0
...
digger	-2	-8	18	24	1
digger	-18	-8	2	24	1
...
digger	-6	-8	22	24	1
digger	-24	-8	8	24	1
...
dontcare	0	0	0	0	0

DetectNet input data representation

NAVIGATING TO QWIKLABS

1. Navigate to:
<https://nvlabs.qwiklab.com>
1. Login or create a new account



The screenshot shows a web browser window with the URL <https://nvlabs.qwiklab.com>. The page has a blue header with the Qwiklabs logo and a "Language" dropdown. The main content area is divided into two columns: "Existing Account" and "Create a New Account".

Existing Account:

- E-mail:
- Password:
- ☐ Remember Me
-
- [Forgot your password?](#)

Create a New Account:

- * First Name:
- * Last Name:
- * Company Name:
- * E-mail:
- * Password:
- * Password Confirmation:
- ☐ I agree to the Terms of Service
- ☐ Opt-in: Send me valuable promos and updates about new hands-on learning!
-

Footer:

- [About](#), [Privacy Policy](#), [Terms of Service](#), [Contact](#)
- © Qwiklabs 12-16
- [Support](#)

ACCESSING LAB ENVIRONMENT

1. Select the event specific In-Session Class in the upper left

1. Click the “Approaches to Object Detection Using DIGITS” Class from the list

*** Model building may take some time and may appear to initially not be progressing ***

The screenshot shows the Qwiklabs interface. At the top, there's a navigation bar with 'QWIKLABS' and tabs for 'IN SESSION 3', 'UPCOMING 1', and 'TAKEN 3'. Below this, the 'In-Session Class' dropdown is set to 'Deep Learning Labs'. A list of classes is displayed, including 'Introduction to Deep Learning', 'Approaches to Object Detection using DIGITS' (marked as 'Running'), 'Identifying Whale Sounds with Audio Classification', 'Deep Learning Network Deployment', 'Introduction to RNNs', 'Exploring TensorFlow on GPUs', 'Introduction to Deep Learning with R and MXNet', and 'Signal Processing using DIGITS'. The 'Approaches to Object Detection using DIGITS' class is highlighted in green. To the right of the class list, there's a detailed view of the selected class, showing its description, duration (90 min), access time (115 min), setup time (8 min), and level (Beginner). The bottom of the interface contains links for 'About', 'Privacy Policy', 'Terms of Service', and 'Contact', as well as social media icons and a 'Support' button.

LAB REVIEW

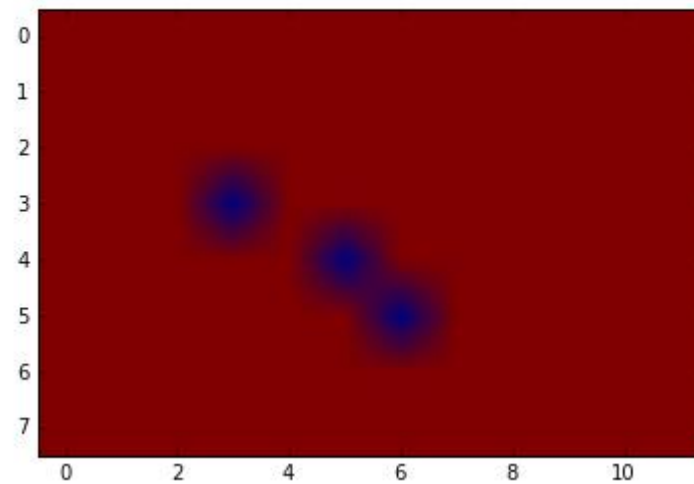
TRAINING APPROACHS

- Approach 1:
 - Patches to build model
 - Sliding window looks for location of whale face



Total inference time: 10.5373151302 seconds

Total inference time: 10.5373151302 seconds



TRAINING APPROACHS

- Approach 3:
 - Fully-convolution network (FCN)

```
241 {
242   layer {
243     name: "pool5"
244     type: "Pooling"
245     bottom: "conv5"
246     top: "pool5"
247     pooling_param {
248       pool: MAX
249       kernel_size: 3
250       stride: 2
251     }
252   }
253   layer {
254     name: "fc6"
255     type: "InnerProduct"
256     bottom: "pool5"
257     top: "fc6"
258     param {
259       lr_mult: 1
260       decay_mult: 1
261     }
262     param {
263       lr_mult: 2
264       decay_mult: 0
265     }
266     inner_product_param {
267       num_output: 4096
268       weight_filler {
269         type: "gaussian"
270         std: 0.005
271       }
272       bias_filler {
273         type: "constant"
274         value: 0.1
275       }
276     }
277   }
278   layer {
279     name: "relu6"
280     type: "ReLU"
281     bottom: "fc6"
282     top: "fc6"
283   }
```

```
layer {
  name: "conv6"
  type: "Convolution"
  bottom: "pool5"
  top: "conv6"
  param {
    lr_mult: 1.0
    decay_mult: 1.0
  }
  param {
    lr_mult: 2.0
    decay_mult: 0.0
  }
  convolution_param {
    num_output: 4096
    pad: 0
    kernel_size: 6
    weight_filler {
      type: "gaussian"
      std: 0.01
    }
    bias_filler {
      type: "constant"
      value: 0.1
    }
  }
}
layer {
  name: "relu6"
  type: "ReLU"
  bottom: "conv6"
  top: "conv6"
}
```


TRAINING APPROACHS

- Approach 4:
 - DetectNet

Source image



Inference visualization



■ bbox-list

ADDITIONAL APPROACHES TO OBJECT DETECTION ARCHITECTURE

- R-CNN = Region CNN
- Fast R-CNN
- Faster R-CNN Region Proposal Network
- RoI-Pooling = Region of Interest Pooling