



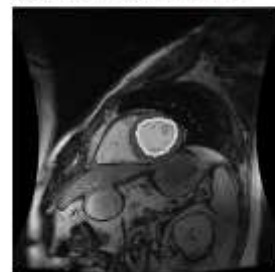
DEEP
LEARNING
INSTITUTE

MEDICAL IMAGE SEGMENTATION WITH DIGITS

Hyun gon Ryu, Jack Han
Solution Architect
NVIDIA Corporation

Overview

Segmentation



Segmentation



SEGMENTATION

Pascal VOC 2012 dataset

Source Image



Inference visualization



Legend



- #0: background
- #1: aeroplane
- #2: bicycle
- #3: bird
- #4: boat
- #5: bottle
- #6: bus
- #7: car
- #8: cat
- #9: chair
- #10: cow
- #11: diningtable
- #12: dog
- #13: horse
- #14: motorbike
- #15: person
- #16: pottedplant
- #17: sheep
- #18: sofa
- #19: train
- #20: tvmonitor
- #255: undefined/don't care



SEGMENTATION

MS COCO dataset

What is COCO?



COCO is a new image recognition, segmentation, and captioning dataset. COCO has several features:

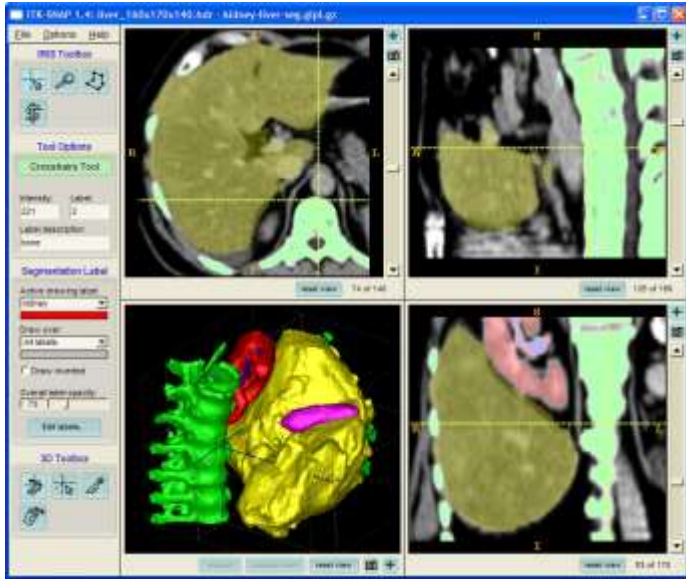
- ✓ Object segmentation
- ✓ Recognition in Context
- ✓ Multiple objects per image
- ✓ More than 300,000 images
- ✓ More than 2 Million instances
- ✓ 80 object categories
- ✓ 5 captions per image
- ✓ Keypoints on 100,000 people

Dataset examples

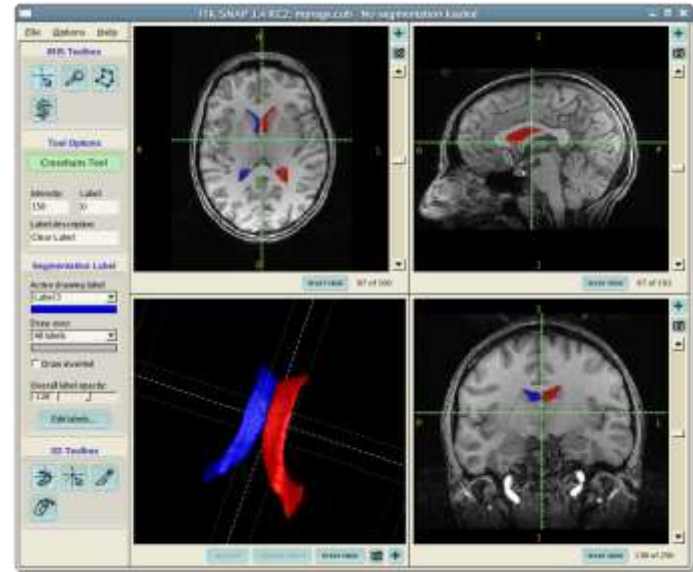


SEGMENTATION

ITK-SNAP



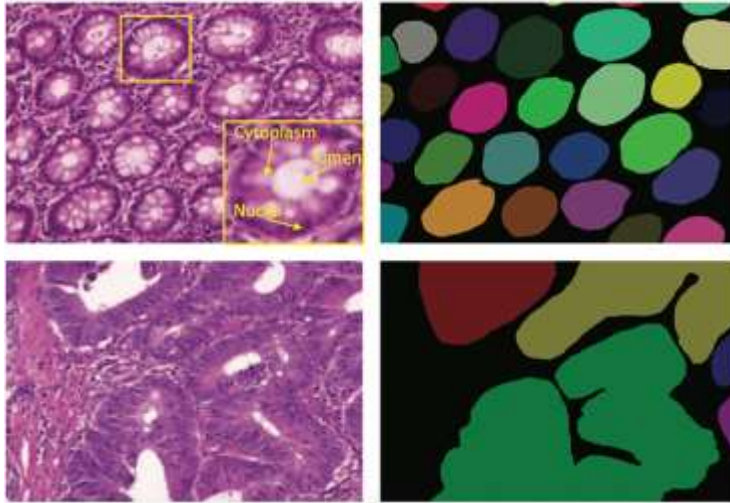
Kidney-Liver Segmentation



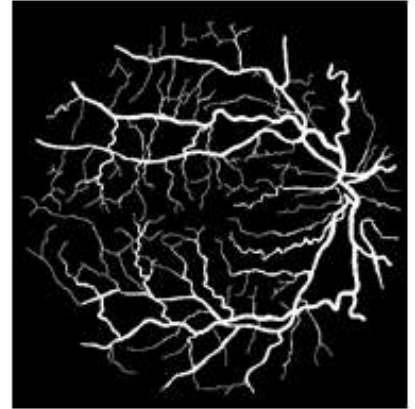
Brain Segmentation

SEGMENTATION

Medical Image



Cancer Cell



Vessel Segmentation

Dataset



CARDIAC MR LEFT VENTRICLE SEGMENTATION

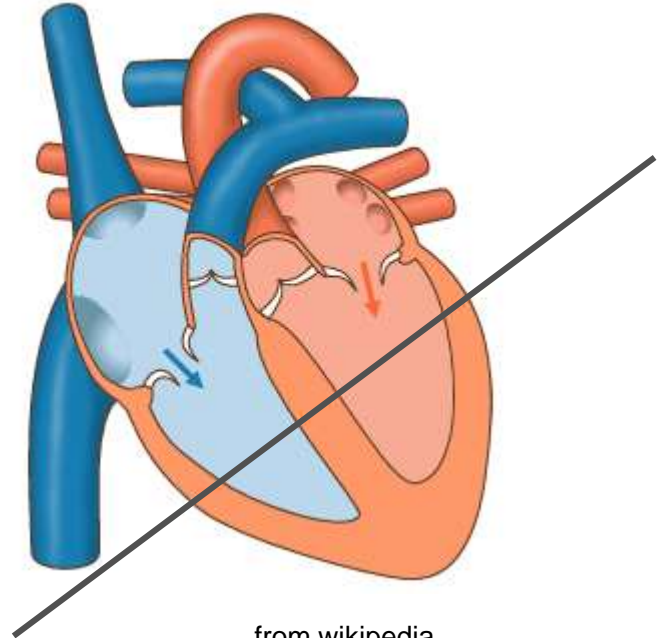
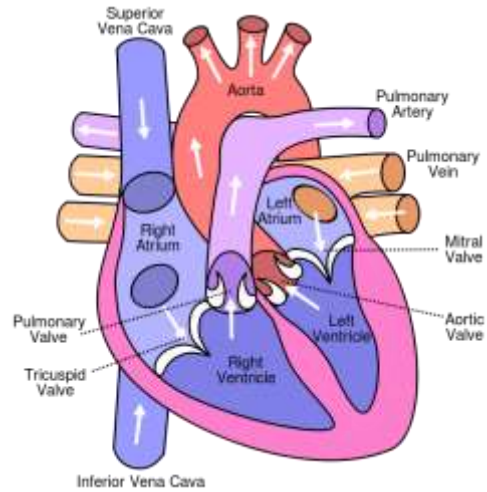
MIDAS Journal - Cardiac MR Left Ventricle Segmentation Challenge



<http://hdl.handle.net/10380/3070>

http://smial.sri.utoronto.ca/LV_Challenge/Home.html

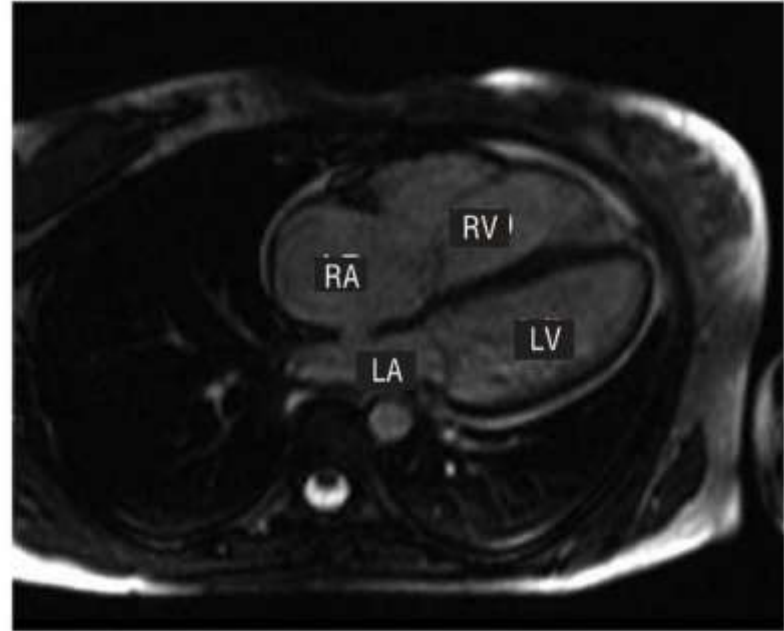
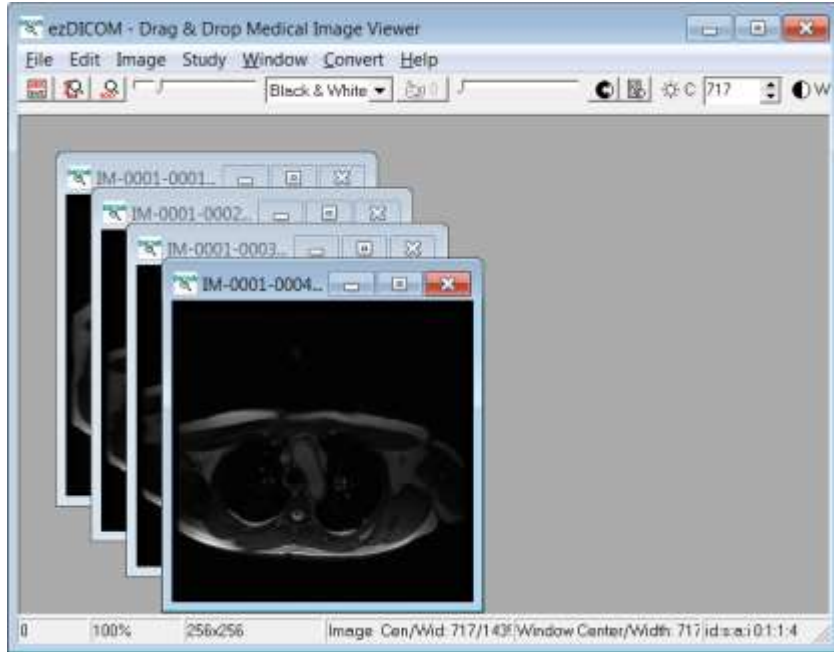
SLICE VIEW



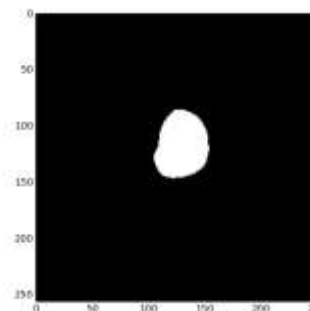
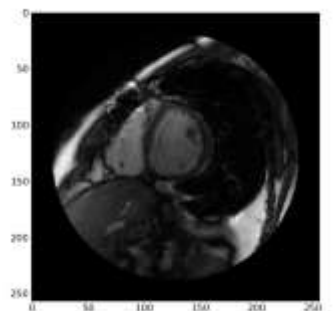
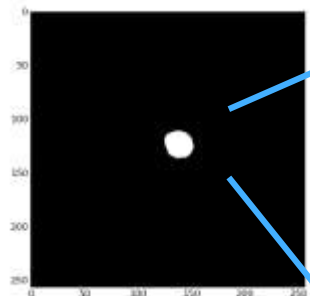
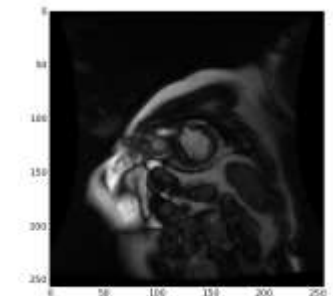
from wikipedia

DICOM VIEWER

ezDICOM



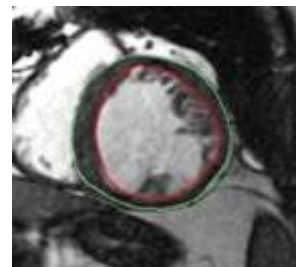
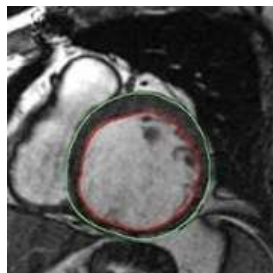
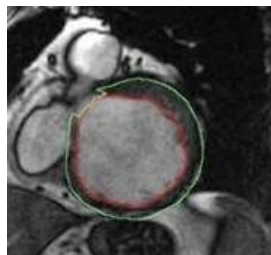
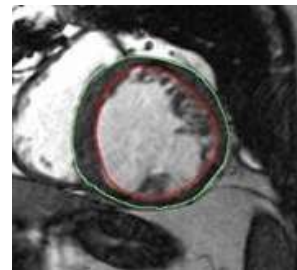
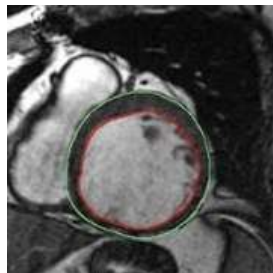
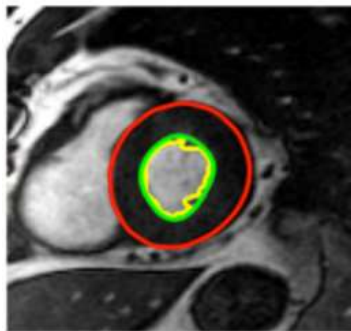
DATASET WITH CONTOUR



X	Y
136.00	120.00
136.50	120.00
137.00	120.00
137.50	120.50
138.00	120.50
138.50	121.00
139.00	121.50
139.50	121.50
140.00	121.50
140.50	122.00
141.00	123.00
141.50	123.50
142.00	124.00
142.50	124.50
142.50	125.00
143.00	125.50
143.50	126.00
143.50	126.50
144.00	127.00
144.50	127.50
144.50	128.00
145.00	128.50

.	.
.	.
.	.

DATASET WITH CONTOUR



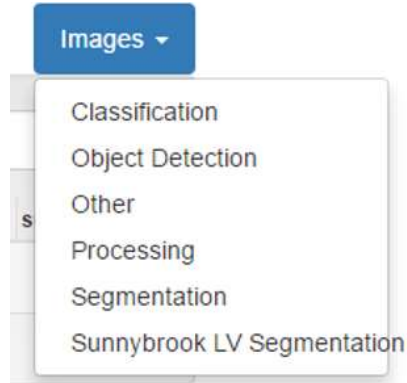


LAB

DIGITS PLUGINS

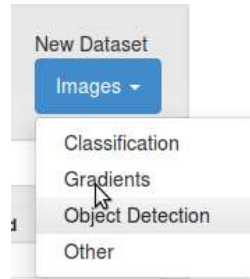
DIGITS Plugins
Image : Sunnybrook LV Segmentation

plugins/data/sunnybrook



DIGITS Plugins
Image : Regression

plugins/data/imageGradients



DIGITS Plugins
Text

plugins/data/textClassification



PREPARE DATA

DIGITS New Dataset

New Sunnybrook LV Segmentation Dataset

Image folder ⓘ

/data/challenge_training

Contour folder ⓘ

/data/Sunnybrook Cardiac MR Database ContoursPart3

Channel conversion ⓘ

Grayscale

% for validation ⓘ

10

Feature Encoding ⓘ

None

% for validation ⓘ

10

Feature Encoding ⓘ

None

Label Encoding ⓘ

None

Encoder batch size ⓘ

1

Number of encoder threads ⓘ

4

DB backend

LMDB

Enforce same shape ⓘ

Yes

Group Name

Dataset Name

Sunnybrook

Create

DATASET

DIGITS

Generic Dataset

Sunnybrook

Owner: ckilam

Get Sunnybrook TensorBoard

Job Information

Job Directory

/jobs/20170307-211644-1b1d

Dataset size

100 MB

Create train_db DB

Entry Count

234

Feature shape ⓘ

(1, 256, 256)

Label shape ⓘ

(1, 256, 256)

labels DB

/jobs/20170307-211644-1b1d/train_db/labels

features DB

/jobs/20170307-211644-1b1d/train_db/features

DB create log file

Job Status Overview

- Initialized
- Running 1
- Done at 0

Total - 53 s

Create train_db

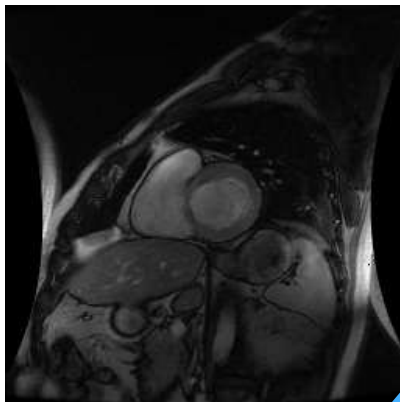
Create val_db

Create test_db

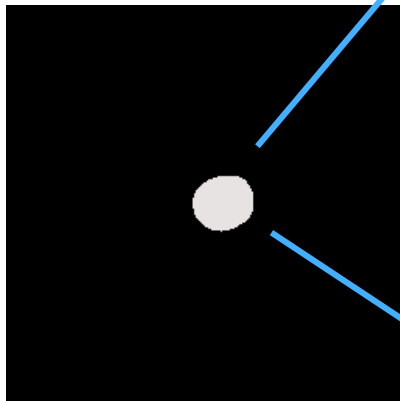
Notes

None ⓘ

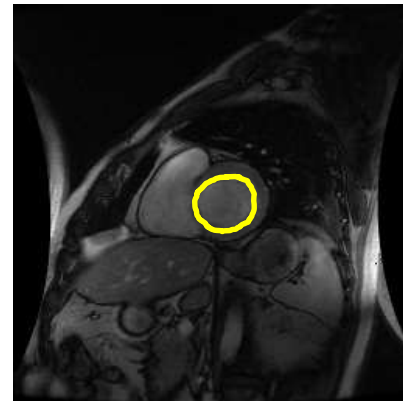
Image



contour



X	Y
130.50	118.00
131.00	118.00
131.50	117.50
132.00	117.50
132.50	117.50
133.00	117.50
133.50	117.50
134.00	117.50
134.50	117.50
135.00	117.50
135.50	117.50
136.00	117.50
136.50	117.50
137.00	117.50
137.50	118.00
138.00	118.00
138.50	118.50
139.00	118.50
139.50	119.00
140.00	119.00
140.50	119.50
141.00	119.50
141.50	120.00
142.00	120.00
142.50	120.50
143.00	121.00
143.50	121.50
144.00	121.50
144.50	122.00
145.00	122.50
.	.
.	.
.	.



CONFIGURE DL MODEL

Filter

framework

status

elapsed

s

New Model

Images ▾

Classification

Object Detection

Other

Processing

Segmentation

Sunnybrook LV Segmentation

Select Dataset ⓘ

Sunnybrook

Sunnybrook

Done 09:17:37 PM

- DB backend: Imdb
- Create train_db DB
 - Entry Count: 234
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)
- Create val_db DB
 - Entry Count: 26
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)

Python Layers ⓘ

Server-side file ⓘ

☐ Use client-side file

Solver Options ⓘ

Training epochs ⓘ

5

Snapshot interval (in epochs) ⓘ

5

Validation interval (in epochs) ⓘ

1

Random seed ⓘ

[none]

Batch size ⓘ multiples allowed

[network defaults]

Batch Accumulation ⓘ

Solver type ⓘ

Stochastic gradient descent (SGD) *

Base Learning Rate ⓘ multiples allowed

1e-4

☐ Show advanced learning rate options

Data

Subt

ims

Crop

not

CONFIGURE DL MODEL

☐ Use client-side file

Base Learning Rate ⓘ multiples allowed

1e-4

☐ Show advanced learning rate options

Standard Networks Previous Networks Pretrained Networks Custom

Caffe Torch

Custom Network ⓘ Visualize

```
1 # data layers
2 layer {
3   name: "data"
4   type: "Data"
5   top: "data"
6   include {
7     phase: TRAIN
8   }
9   data_param {
10    batch_size: 4
11    backend: Lmdb
12  }
13 }
14 layer {
15   name: "label"
16   type: "Data"
17   top: "label"
18   include {
19     phase: TRAIN
```

```
334
335 # Dice coefficient
336 #layer {
337 #   type: 'Python'
338 #   name: 'dice'
339 #   bottom: 'score'
340 #   bottom: 'label'
341 #   top: 'dice'
342 #   python_param {
343 #     module: "digits_python_layers"
344 #     layer: "Dice"
345 #   }
346 #   exclude { stage: "deploy" }
347 #}
```

Pretrained model(s) ⓘ

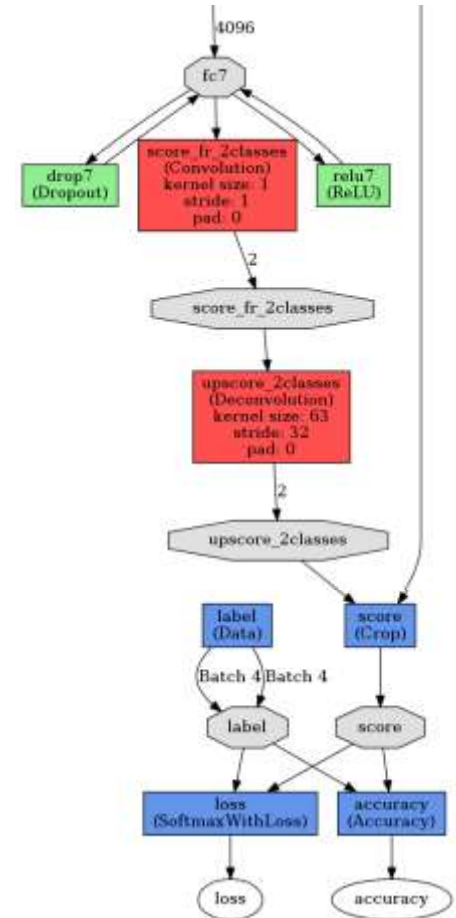
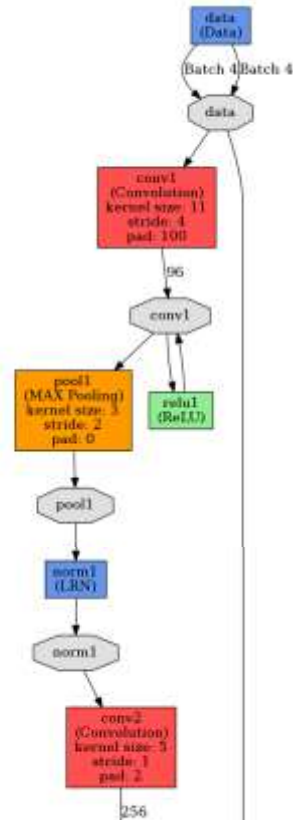
Group Name ⓘ

Model Name ⓘ

fc_alexnet-sunnybrook

Create

VISUALIZE



MONITOR TRAIN

Job Directory
jobs/20170314-134034-8042

Disk Size
0 B

Network (train/val)
train_val.prototxt

Network (deploy)
deploy.prototxt

Network (original)
original.prototxt

Solver
solver.prototxt

Raw caffe output
caffe_output.log

Dataset
Sunnybrook

Done 01:39:52 PM

- DB backend: InnoDB
- Create train_db DB
 - Entry Count: 234
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)
- Create val_db DB
 - Entry Count: 26
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)

Job Status (related)

- Initiated at 01:45:34 PM

Train Caffe Model selected ▾

Hardware

GRID K520 (#0)

Memory
1.12 GB / 3.94 GB (28.6%)

GPU Utilization
100%

Temperature
45 °C

Process #181

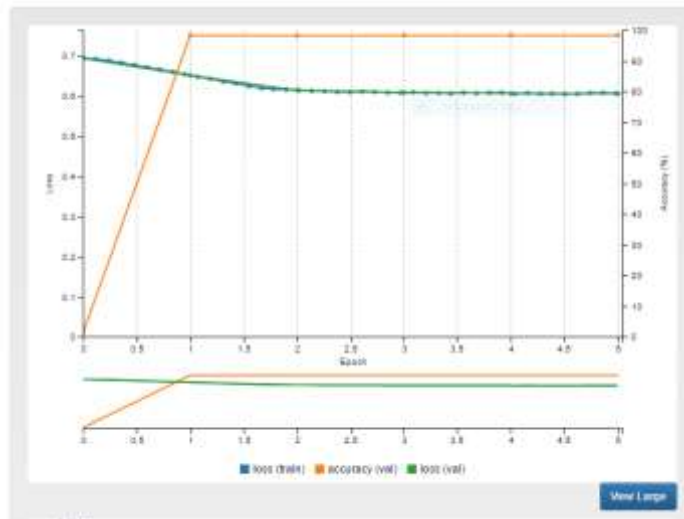
CPU Utilization
101.9%

Memory
868 MB (8.0%)

Related jobs

Generic Dataset

Sunnybrook train ▾



TEST

Trained Models

Select Model

Epoch #5

Download Model

Make Pretrained Model

Select Visualization Method

Image Segmentation

Visualization Options

Display segmented image

Colormap

From dataset

Inference Options

☒ Do not resize input image(s)

Test an image

Image file

image file

☒ Show visualizations and statistics

Test

Test a record from validation set

Record from validation set

SC-HF-NI-3

INTEGRATION

Test Model

Inference

Model box

Test image

Test image

Job Statistics

fcn_alexnet-sunnybrook image inference

- Initiated at 01:07:23 PM (1 second)
- Running at 01:07:31 PM (1 second)
- Over at 01:07:34 PM (1 second)

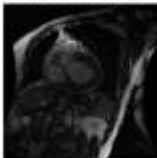
View Model box

Notes

None

Summary

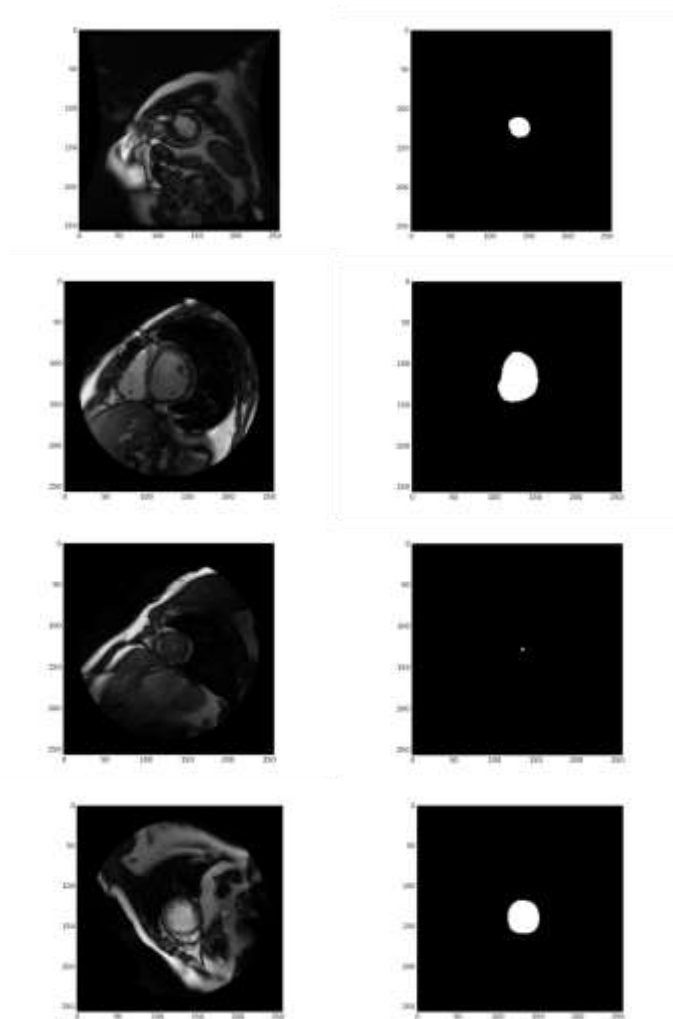
Output visualizations



REASON

Cine MR

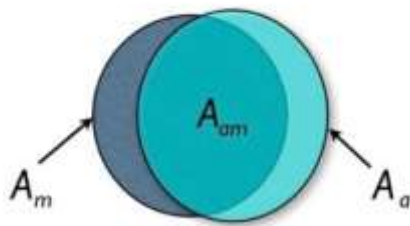
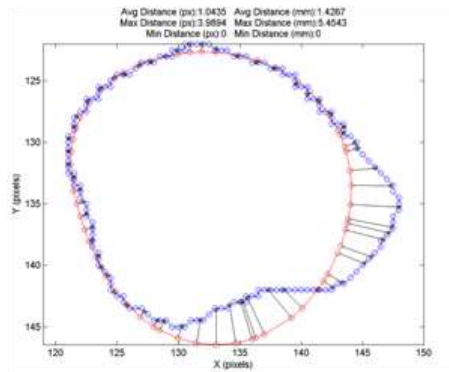
Same Object, time variance



PRACTICE 2

DICE METRIC

DICE METRIC



Manual contour
(Expert)

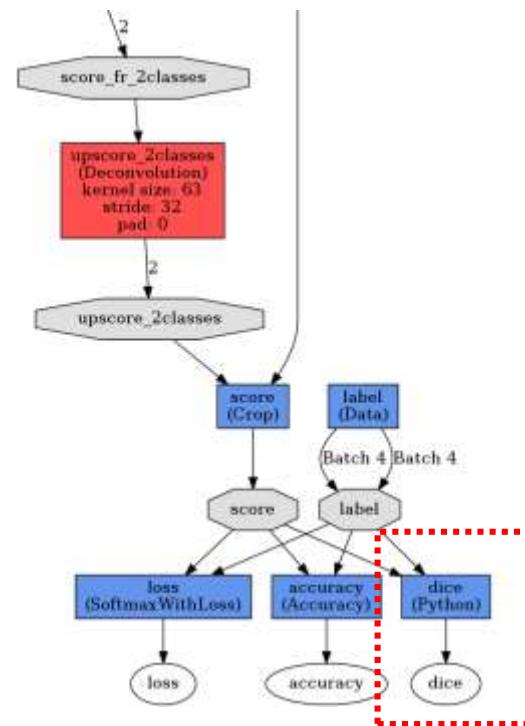
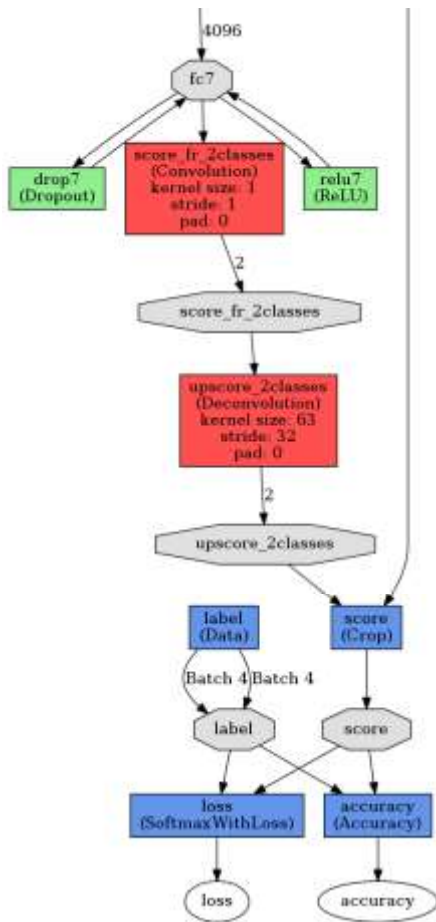
Automatic
(DL)

intersection

$$DM = 2A_{am}(A_a + A_m)^{-1}$$

Union

DICE METRIC



PYTHON LAYER

```
324
325 layer {
326   name: "accuracy"
327   type: "Accuracy"
328   bottom: "score"
329   bottom: "label"
330   top: "accuracy"
331   include { stage: "val" }
332 }
333
334
335 # Dice coefficient
336 layer {
337   type: 'Python'
338   name: 'dice'
339   bottom: 'score'
340   bottom: 'label'
341   top: 'dice'
342   python_param {
343     module: "digits_python_layers"
344     layer: "Dice"
345   }
346   exclude { stage: "deploy" }
347 }
```

Pretrained model(s) ⓘ

```
import random

import numpy as np
import caffe

class Dice(caffe.Layer):
    """
    A layer that calculates the Dice coefficient
    """
    def setup(self, bottom, top):
        if len(bottom) != 2:
            raise Exception("Need two inputs to compute Dice coefficient.")

    def reshape(self, bottom, top):
        # check input dimensions match
        if bottom[0].count != 2*bottom[1].count:
            raise Exception("Prediction must have twice the number of elements of the input.")
        # loss output is scalar
        top[0].reshape(1)

    def forward(self, bottom, top):
        #print "bottom[0].shape=%s" % repr(bottom[0].data.shape)
        #print "bottom[1].shape=%s" % repr(bottom[1].data.shape)
        label = bottom[1].data[:,0,:,:]
        # compute prediction
        prediction = np.argmax(bottom[0].data, axis=1)
        #print "prediction.shape=%s" % repr(prediction.shape)
        # area of predicted contour
        a_p = np.count_nonzero(prediction)
        # area of contour in label
        a_l = np.count_nonzero(label)
        # area of intersection
        a_pl = np.count_nonzero(prediction * label)
        #print "a_p=%f a_l=%f a_pl=%f" % (a_p, a_l, a_pl)
        # dice coefficient
        dice_coeff = 2.*a_pl/(a_p + a_l)
        top[0].data[...] = dice_coeff

    def backward(self, top, propagate_down, bottom):
        pass
```


PYTHON LAYER

Label shape (1, 256, 256)

Python Layers ?

Client-side file ?

Browse... layers.py

☒ Use client-side file

[network defaults]

Batch Accumulation ?

Solver type ?

Stochastic gradient descent (SGD)

Base Learning Rate ? multiples allowed

0.0001

☐ Show advanced learning rate options

Type : Python

New Image Model x Inference x

ec2-54-204-98-40.compute-1.amazonaws.com:5000/m

DIGITS New Model

Custom Network ? Visualize

```
305
306 # SoftMaxWithLoss ##
307 layer {
308   name: "loss"
309   type: "SoftmaxWithLoss"
310   bottom: "score"
311   bottom: "label"
312   top: "loss"
313   loss_param {
314     normalize: true
315   }
316   exclude {
317     stage: "deploy"
318   }
319 }
320 #####
321
322 #####
323
324 layer {
325   name: "accuracy"
326   type: "Accuracy"
327   bottom: "score"
328   bottom: "label"
329   top: "accuracy"
330   include { stage: "val" }
331 }
332
333
334 # Dice coefficient
335 layer {
336   type: "Python"
337   name: "dice"
338   bottom: "score"
339   bottom: "label"
340   top: "dice"
341 }
```

ENABLE DICE LAYER

```
124 #####
125
126 layer {
127   name: "accuracy"
128   type: "Accuracy"
129   bottom: "score"
130   bottom: "label"
131   top: "accuracy"
132   include { stage: "val" }
133 }
134
135 # Dice coefficient
136 layer {
137   type: "Python"
138   name: "dice"
139   bottom: "score"
140   bottom: "label"
141   top: "dice"
142   python_param {
143     module: "digits_python_layers"
144     layer: "Dice"
145   }
146   exclude { stage: "deploy" }
147 }
```



```
124 #####
125
126 layer {
127   name: "accuracy"
128   type: "Accuracy"
129   bottom: "score"
130   bottom: "label"
131   top: "accuracy"
132   include { stage: "val" }
133 }
134
135 # Dice coefficient
136 layer {
137   type: "Python"
138   name: "dice"
139   bottom: "score"
140   bottom: "label"
141   top: "dice"
142   python_param {
143     module: "digits_python_layers"
144     layer: "Dice"
145   }
146   exclude { stage: "deploy" }
147 }
```

Group Name ?

Model Name ?

fc_alexnet-sunnybrook-with_dice

Create

DL TRAINING WITH DICE

Job Directory
jobs/20170314-135230-670d
Disk Size
1.34 KB
Network (train/val)
train_val.prototxt
Network (deploy)
deploy.prototxt
Python layer
jobs/20170314-135230:
d70bd819_python_layers.py
Network (original)
original.prototxt
Solver
solver.prototxt
Raw caffe output
caffe_output.log

Dataset
Sunnybrook
Done 01:38:52 PM

- DB backend: leveldb
- Create train_db DB
 - Entry Count: 234
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)
- Create val_db DB
 - Entry Count: 26
 - Feature shape (1, 256, 256)
 - Label shape (1, 256, 256)

Job Status initialized

• initialized at 01:52:30 PM

Train Caffe Model initialized →

Hardware

GRID K520 (#0)

Memory
1.13 GB / 3.94 GB (28.7%)
GPU Utilization
100%
Temperature
60 °C

Process #350

CPU Utilization
10.10%
Memory
140 MB (5.3%)

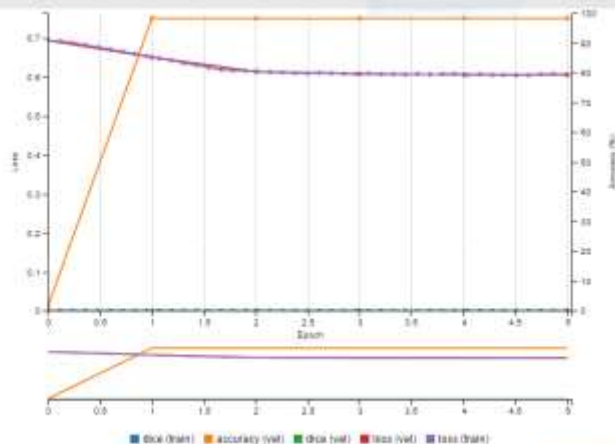
Related jobs

Generic Dataset

Sunnybrook Data →

Generic Image Model

fcnalexnet-sunnybrook Data →



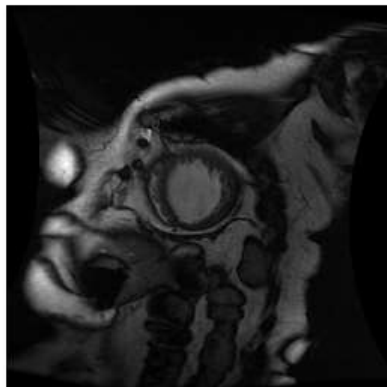
View Large

RESULT2 WITH DICE

fcn_alexnet-sunnybrook-with_dice Image Inference

Summary

Output visualizations



Job Status Done

- Initialized at 01:55:00 PM (1 second)
- Running at 01:55:01 PM (3 seconds)
- Done at 01:55:04 PM
(Total - 4 seconds)

Infer Model Demo ➡

Notes

None





PRACTICE 3

USE PRE-TRAINED PARAMETERS

FOR ALEXNET (COLOR)

The screenshot shows the AWS Digits console interface for creating a new dataset. The browser tabs include 'for_alexnet-sunnybrook', 'New Sunnybrook LV Segmentation Dataset', and 'Inference'. The URL is 'ec2-54-204-98-40.compute-1.amazonaws.com:5000/datasets/generic/new/image-sunnybrook?clone=20170314-133857.7Mib'. The page title is 'New Sunnybrook LV Segmentation Dataset'. The configuration form includes the following fields:

- image folder**: /data/challenge_training
- Contour folder**: /data/Sunnybrook Cardiac MR Database ContoursPart3
- Channel conversion**: RGB (highlighted with a red dashed box)
- % for validation**: 10
- Feature Encoding**: None
- Label Encoding**: None
- Encoder batch size**: (field is partially visible)

CONFIGURE DL MODEL WITH PRE-TRAINED

Python Layers ⓘ

Client-side file ⓘ

layers.py

☒ Use client-side file

Solver type ⓘ

Stochastic gradient descent (SGD) ▼

Base Learning Rate ⓘ multiples allowed

0.0001

☒ Show advanced learning rate options

Policy

103 stride: 1

Pretrained model(s) ⓘ

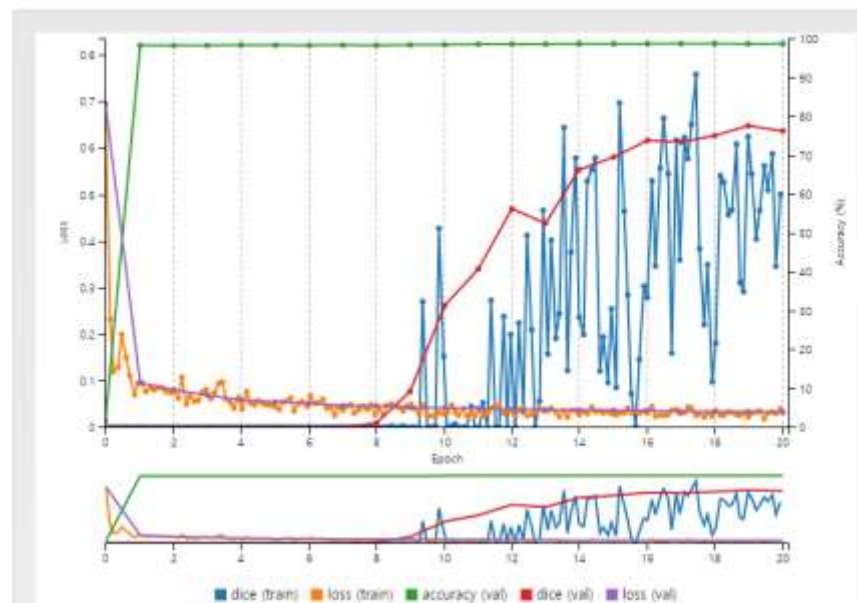
/data/fcn_alexnet.caffemodel

Group Name ⓘ

Model Name ⓘ

fc_alexnet-sunnybrook-with_dice-pretrained

DICE(VAL)



RESULT WITH PRE-TRAIN

Summary

Output visualizations



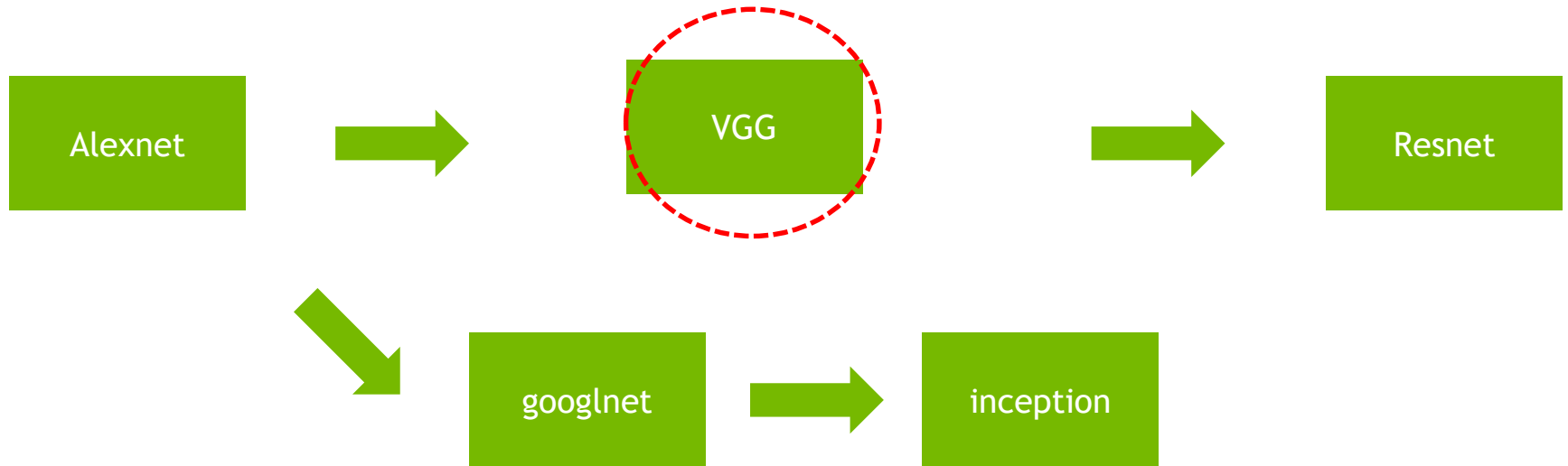
□ left ventricle



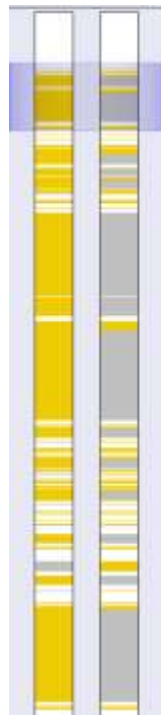
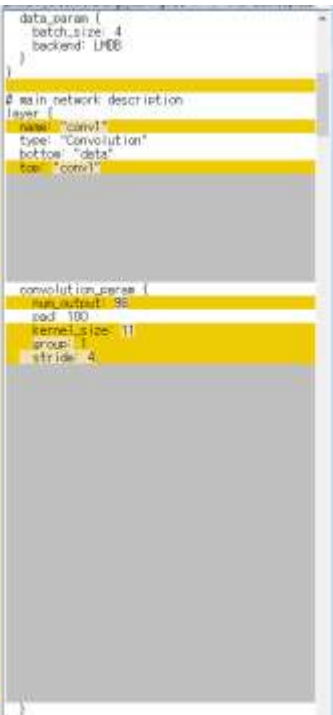
PRACTICE 4

MORE FINE DL MODEL

FCN-8S



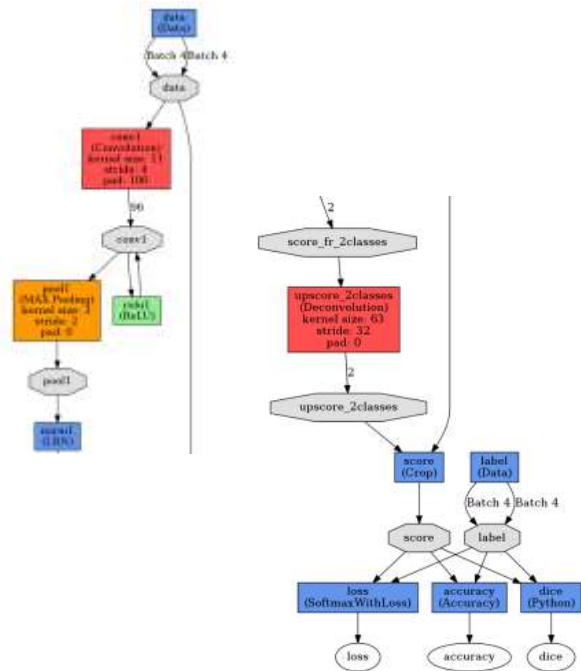
DIFF

 A vertical diagram of a VGG-style convolutional neural network. It shows a sequence of layers: an input layer, two convolutional layers with 3x3 kernels, a ReLU layer, another convolutional layer with a 3x3 kernel, another ReLU layer, and a final convolutional layer with a 3x3 kernel. The layers are represented by vertical bars of varying widths and colors (yellow, grey, white).	<pre>data_param { batch_size: 4 backend: LMDB } # main network description layer { name: "conv1_1" type: "Convolution" bottom: "data" top: "conv1_1" param { lr_mult: 1 decay_mult: 1 } param { lr_mult: 2 decay_mult: 0 } convolution_param { num_output: 64 pad: 100 kernel_size: 3 stride: 1 } } layer { name: "relu1_1" type: "ReLU" bottom: "conv1_1" top: "conv1_1" } layer { name: "conv1_2" type: "Convolution" bottom: "conv1_1" top: "conv1_2" param { lr_mult: 1 decay_mult: 1 } param { lr_mult: 2 decay_mult: 0 } convolution_param { num_output: 64 pad: 1 kernel_size: 3 stride: 1 } }</pre>	 A vertical diagram of an Alexnet-style convolutional neural network. It shows a sequence of layers: an input layer, a convolutional layer with an 11x11 kernel, a ReLU layer, another convolutional layer with an 11x11 kernel, another ReLU layer, and a final convolutional layer with a 3x3 kernel. The layers are represented by vertical bars of varying widths and colors (yellow, grey, white).
---	--	--

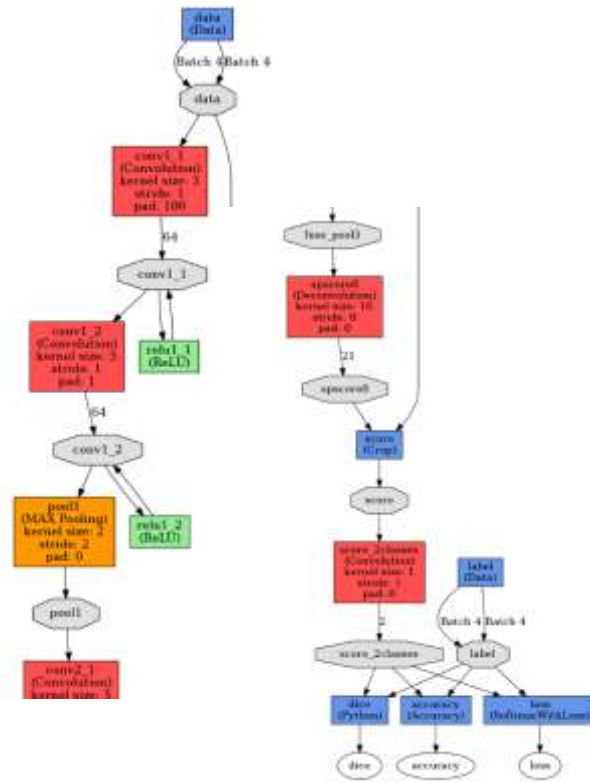
VGG style

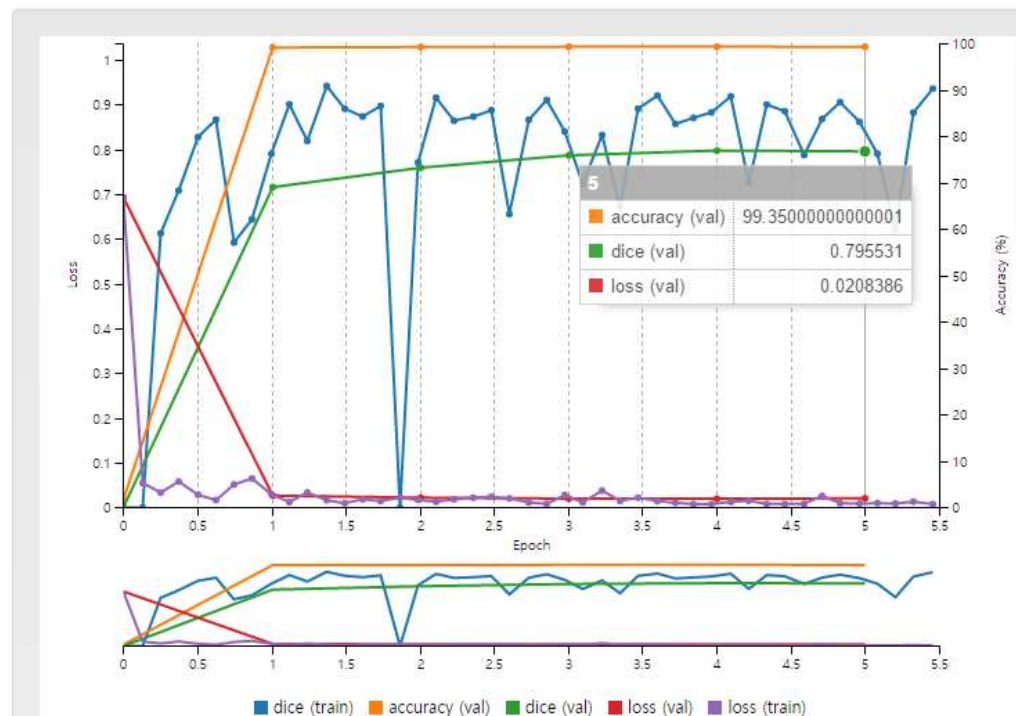
Alexnet

Alxenet



VGG style

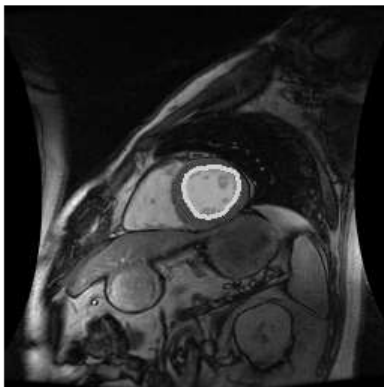




RESULT

Summary

Output visualizations

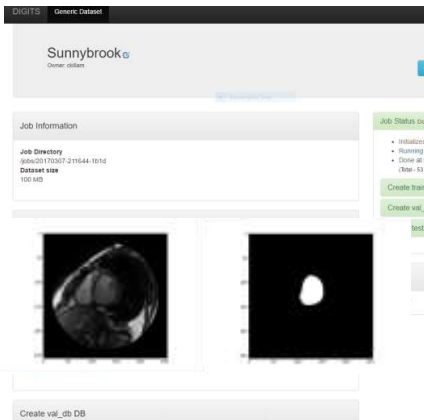


□ left ventricle

MEDICAL IMAGE SEGMENTATION WITH DIGITS

summary

Prepare Dataset

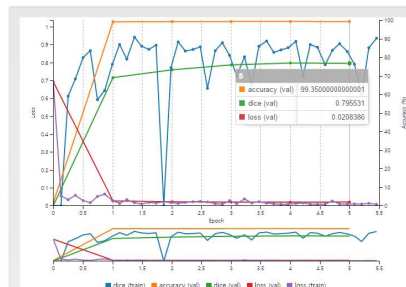


Configure DL Model



With python layer(DICE metric)

DL Training



Segmentation



WHAT'S NEXT

TAKE SURVEY

ACCESS ONLINE LABS

Check your email to access more DLI training online.

ATTEND WORKSHOP

Visit www.nvidia.com/dli for workshops in your area.

JOIN DEVELOPER PROGRAM

Visit <https://developer.nvidia.com/join> for more.



DEEP
LEARNING
INSTITUTE

www.nvidia.com/dli