NVIDIA交互式GPU训练平台： DIGITS

AGENDA

开源免费的深度学习管理工具

1. DIGITS诞生背景及功能简介
2. DIGITS 的训练数据集导入
3. DIGITS 的基本模型训练操作：图片分类
4. DIGITS 用于目标检测
5. DIGITS 用于图像分割
6. DIGITS 迁移学习的使用
7. DIGITS Plugin的使用

传统的深度学习开发界面

- DL frameworks, Caffe, etc. aimed at computer scientist not data scientist
- Juggle multiple files & windows
- Handcrafted visualizations
- Manual log file parsing
- Manual experiment logging
- Model editing in Lua IDE files

```
    -- stage 1 : filter bank -> squashing -> L2 pooling -> normalization
    model:add(nn.SpatialConvolutionMM(nfeats, nstates[1], filtsize, filtsize))
    model:add(nn.Tanh())
    model:add(nn.SpatialLPPooling(nstates[1],2,poolsize,poolsize,poolsize,poolsize))
    model:add(nn.SpatialSubtractiveNormalization(nstates[1], normkernel))

    -- stage 2 : filter bank -> squashing -> L2 pooling -> normalization
    model:add(nn.SpatialConvolutionMM(nstates[1], nstates[2], filtsize, filtsize))
    model:add(nn.Tanh())
    model:add(nn.SpatialLPPooling(nstates[2],2,poolsize,poolsize,poolsize,poolsize))
    model:add(nn.SpatialSubtractiveNormalization(nstates[2], normkernel))

    -- stage 3 : standard 2-layer neural network
    model:add(nn.Reshape(nstates[2]*filtsize*filtsize))
    model:add(nn.Linear(nstates[2]*filtsize*filtsize, nstates[3]))
    model:add(nn.Tanh())
    model:add(nn.Linear(nstates[3], noutputs))
  end
else

  error('unknown -model')

end

----------------------------------------------------------------------
print '==> here is the model:'
print(model)
```

```
==> here is the model:
nn.Sequential {
  [input -> (1) -> (2) -> (3) -> (4) -> (5) -> (6) -> (7) -
  (1): nn.SpatialConvolutionMM(3 -> 64, 5x5)
  (2): nn.Tanh
  (3): nn.Sequential {
    [input -> (1) -> (2) -> (3) -> (4) -> output]
    (1): nn.Square
    (2): nn.SpatialAveragePooling(2,2,2,2)
    (3): nn.MulConstant
    (4): nn.Sqrt
  }
  (4): nn.SpatialSubtractiveNormalization
  (5): nn.SpatialConvolutionMM(64 -> 64, 5x5)
  (6): nn.Tanh
  (7): nn.Sequential {
    [input -> (1) -> (2) -> (3) -> (4) -> output]
    (1): nn.Square
    (2): nn.SpatialAveragePooling(2,2,2,2)
    (3): nn.MulConstant
    (4): nn.Sqrt
  }
  (8): nn.SpatialSubtractiveNormalization
  (9): nn.Reshape(1600)
  (10): nn.Linear(1600 -> 128)
  (11): nn.Tanh
  (12): nn.Linear(128 -> 10)
}
```

```
#28p        a02 Take the trueNLL model w/ kappa_true=0.76 from expt#20  -> true + MSE under settings
                d   all data,       55 epochs,  epochSize 500,  regime with LR=0.2
th main_custom_noTest.lua -cache devAnnealAugFinal -data ../data-1024-fulltrain -nGPU 4 -name TrueNLL
nDonkeys 6 -nEpochs 55 -epochSize 500 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /
multiGPU/highres/learn_1024_anneal/devAnnealAug/EverNLLtoTrueNLL_fixedLR,CRfinal=trueCR,LR=0.1,batchS
nGPU=4,netType=1024/nin-preall-multi-fixedLR/05-48_on_25-Jul-15/model_100.t7 -CRinitial trueCR -CRfi
1 -manualSeed 4359 | tee expt28p_ap02_TrueNLLtoTrueMSEfinal.log


#29p        a03 Take the 62411+0.75MSE model  w/ kappa_true=0.78 from expt#24 -> true + MSE under se
                b   regular data,   55 epochs,  epochSize 200,  regime with LR=0.2
th main_custom.lua -cache devAnnealAug -data ../data-1024 -nGPU 4 -name CR62411MSE3by4_seed864521_exp
epochSize 200 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /home/mnt/retinopathy/ima
learn_1024_anneal/devAnnealAug/pretrained62411MSE3by4_trainFurther,CRfinal=6,2,4,1,1,CRinitial=6,2,4,
MSEinitial=0.75,batchSize=32,epochSize=300,nDonkeys=6,nGPU=4,netType=1024/nin-preall-multi-fixedLR/0
CRinitial 6,2,4,1,1 -CRfinal trueCR -MSEinitial 0 -MSEfinal 1 -manualSeed 864521| tee expt29p_a03_CR


#30p        ap06    Take the 62411+0.75MSE model  w/ kappa_true=0.78 from expt#24 -> true + MSE under
                d   all data,       55 epochs,  epochSize 500,  regime with LR=0.2
th main_custom_noTest.lua -cache devAnnealAugFinal -data ../data-1024-fulltrain -nGPU 4 -name CR62411
nDonkeys 6 -nEpochs 55 -epochSize 500 -batchSize 32 -LR 0.2 -netType 1024/nin-preall-multi -retrain /
multiGPU/highres/learn_1024_anneal/devAnnealAug/pretrained62411MSE3by4_trainFurther,CRfinal=6,2,4,1,1
MSEfinal=0.75,MSEinitial=0.75,batchSize=32,epochSize=300,nDonkeys=6,nGPU=4,netType=1024/nin-preall-mu
model_55.t7 -CRinitial 6,2,4,1,1 -CRfinal trueCR -MSEinitial 0 -MSEfinal 1 -manualSeed 351987 | tee e
log

```

NVIDIA’S DIGITS

交互式深度学习GPU训练系统

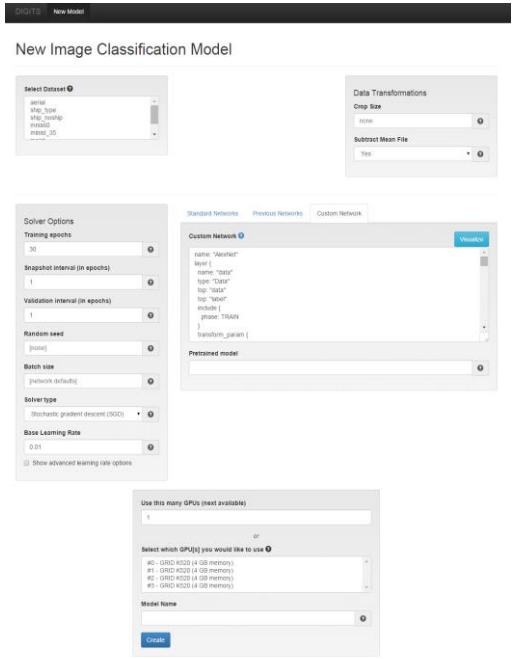
- 简化通用的深度学习任务如：
  - 管理数据
  - 在多GPU系统上设计并训练神经网络
  - 使用高级可视化界面，监控实时性能
- 完全的交互式界面，使得数据科学家可以专注在设计及训练网络，节约编程及调试代码的时间
- 开源、免费

NVIDIA DIGITS

交互式深度学习GPU训练平台

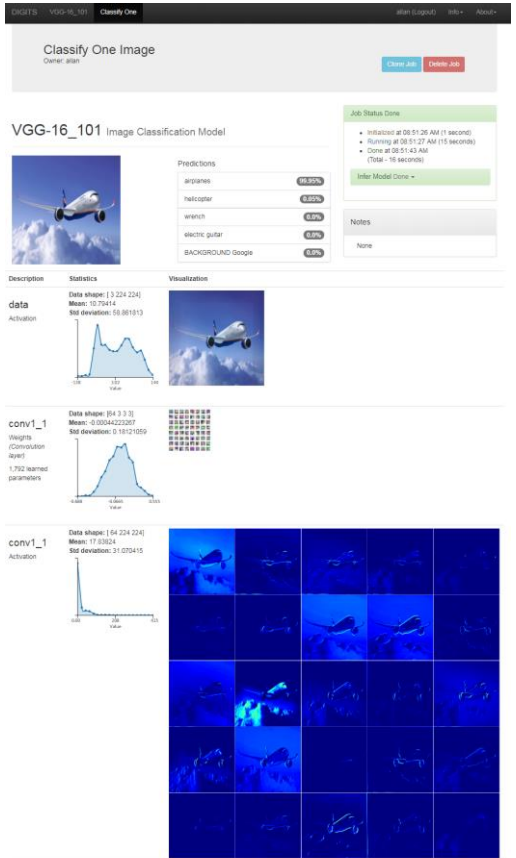
数据加载

配置神经网络



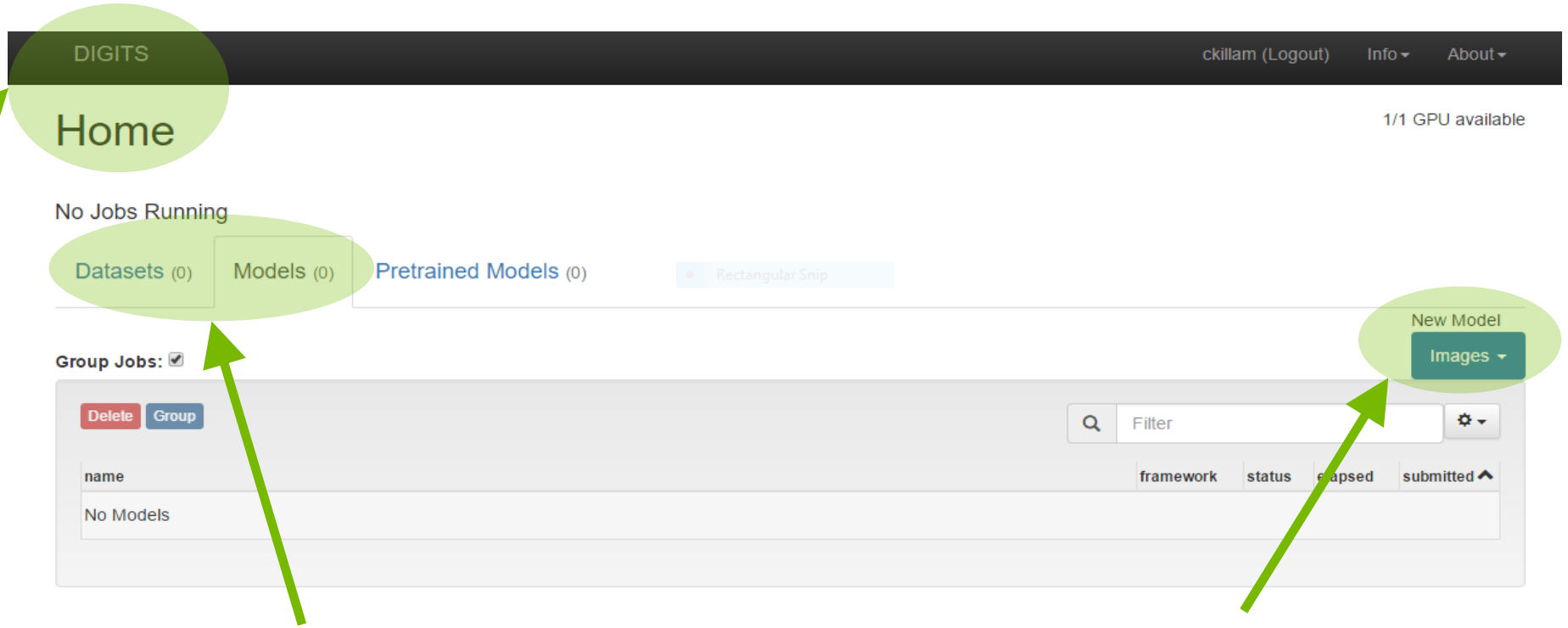
监控训练过程

校验训练精度



DIGITS - HOME

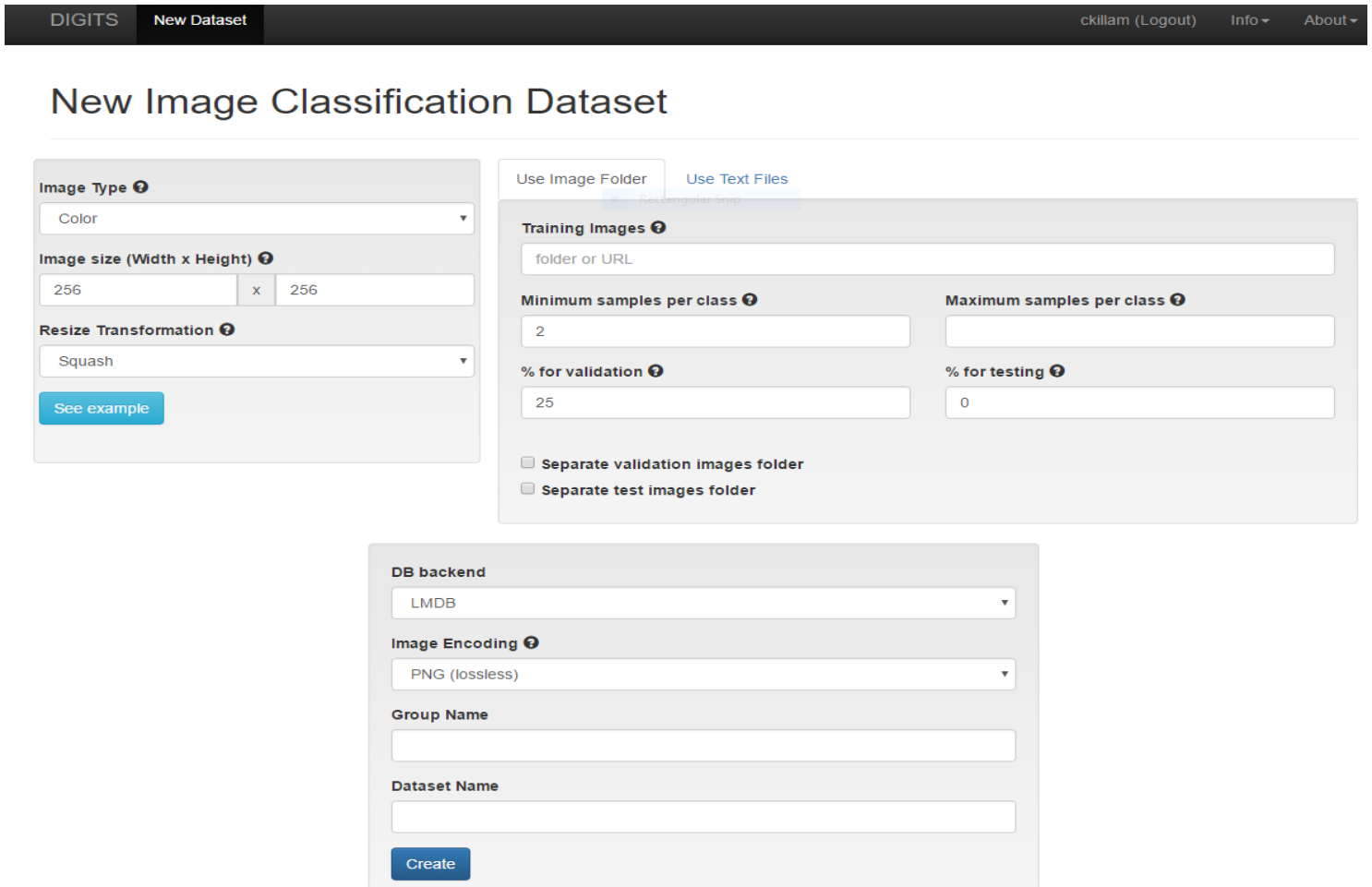
Clicking DIGITS will bring you to this Home screen



Click here to see a list of existing datasets or models

Clicking here will present different options for model and dataset creation

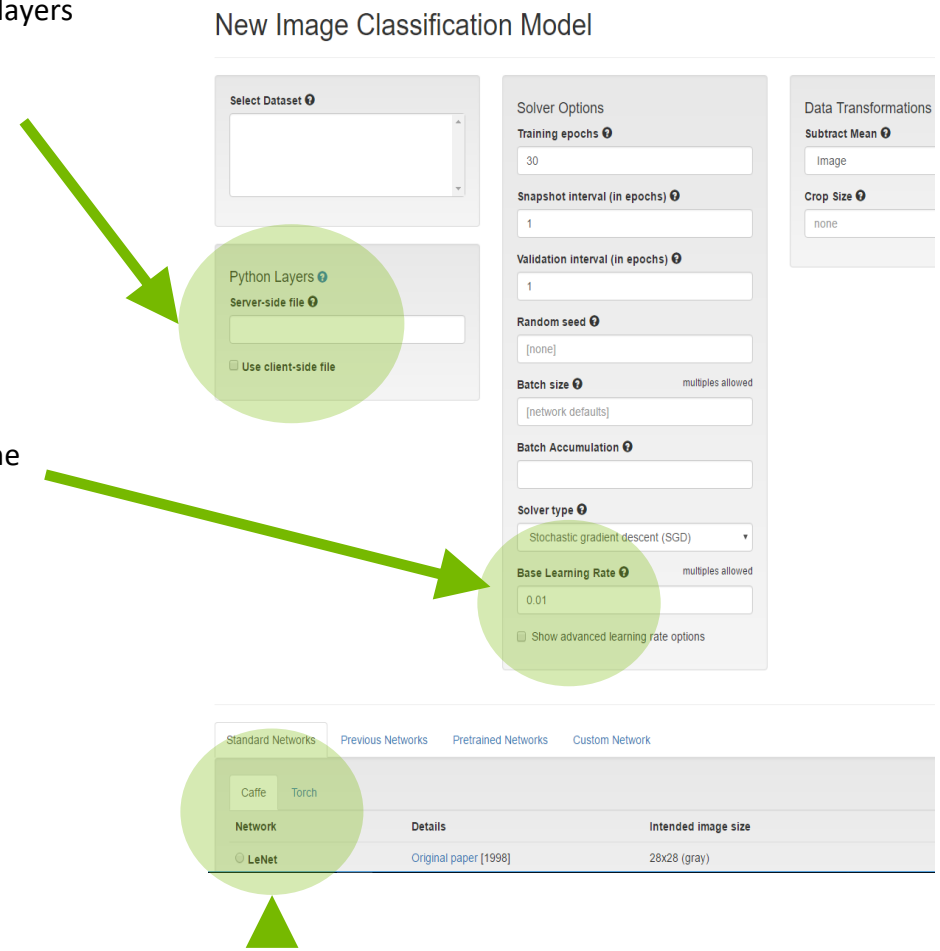
DIGITS - DATASET



Different options will be presented based upon the task

DIGITS - MODEL

Define custom layers with Python



Can anneal the learning rate

Differences may exist between model tasks

DIGITS- 模型编程入口及可视化

DIGITS | New Model

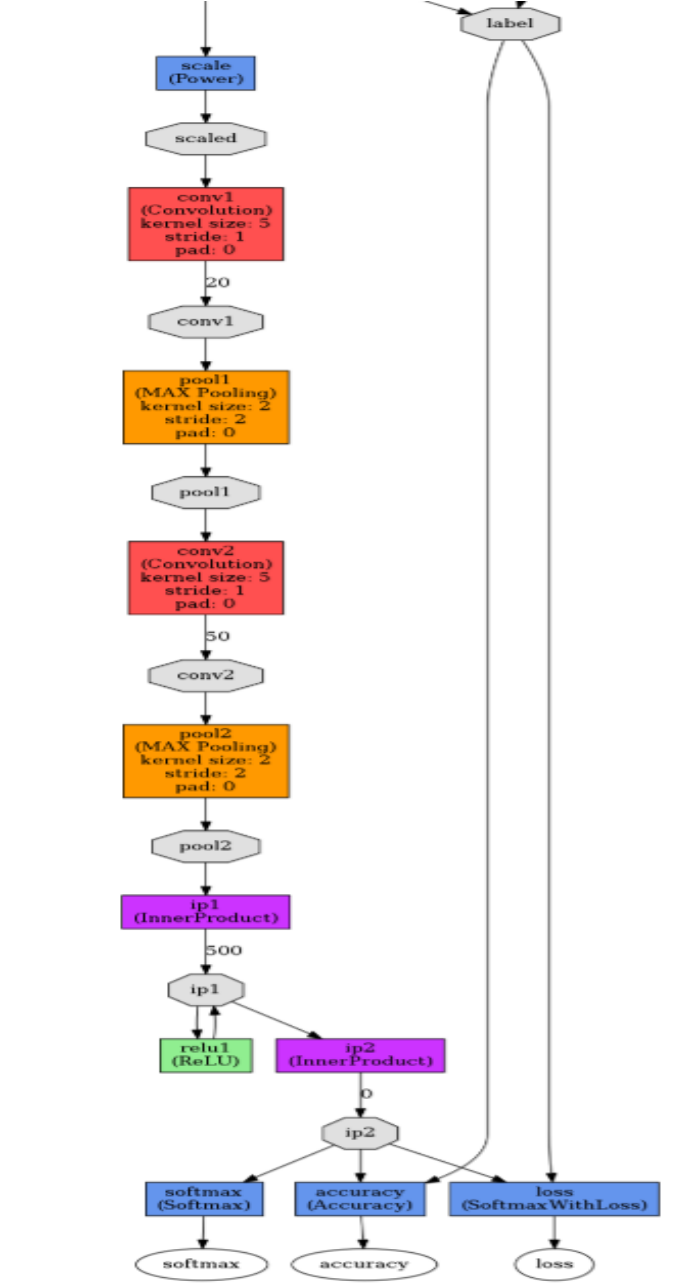
Standard Networks | Previous Networks | Pretrained Networks | Custom Network

Caffe | Tensorflow

Custom Network Visualize

```
  top: "scaled"
  type: "Power"
  power_param {
    # 1/(standard deviation on MNIST dataset)
    scale: 0.0125
  }
}
layer {
  name: "conv1"
  type: "Convolution"
  bottom: "scaled"
  top: "conv1"
  param {
    lr_mult: 1
  }
  param {
    lr_mult: 2
  }
  convolution_param {
    num_output: 20
    kernel_size: 5
    stride: 1
    weight_filler {
      type: "xavier"
    }
    bias_filler {
      type: "constant"
    }
  }
}
layer {
  name: "pool1"
  type: "Pooling"
  bottom: "conv1"
  top: "pool1"
  pooling_param {
    pool: MAX
    kernel_size: 2
    stride: 2
  }
}
layer {
  name: "conv2"
```

Pretrained model(s)

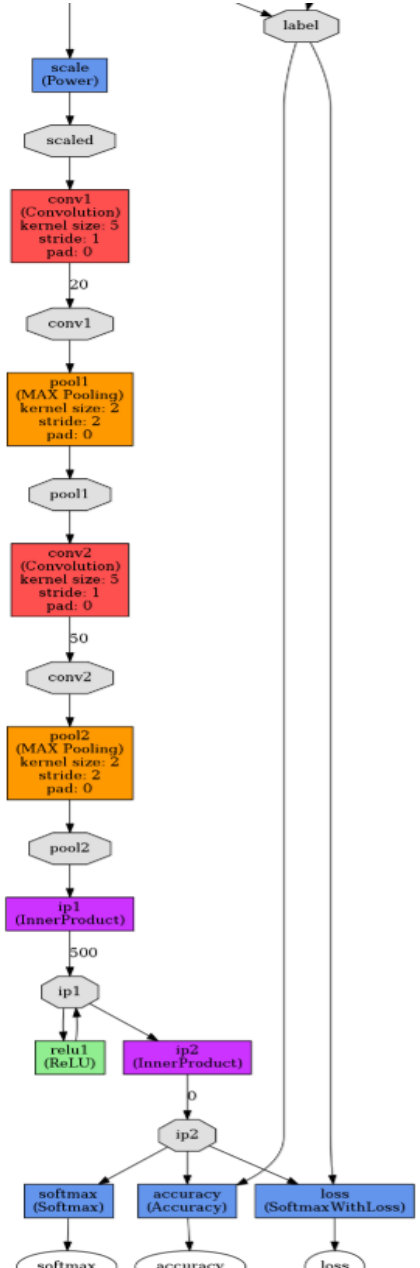


MODIFY THE NETWORK

Adding filters and ReLU layer

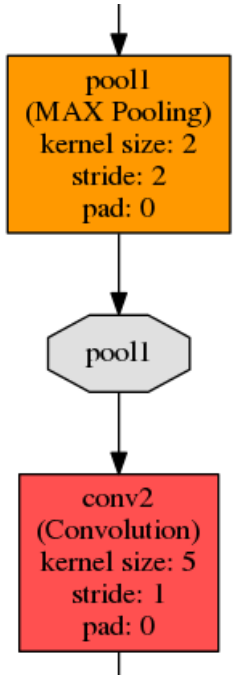
```
layer {
        name: "pool1“
        type: "Pooling“
        …
}

layer {
        name: "reluP1"
        type: "ReLU"
        bottom: "pool1"
        top: "pool1"
}

layer {
        name: "reluP1“
```

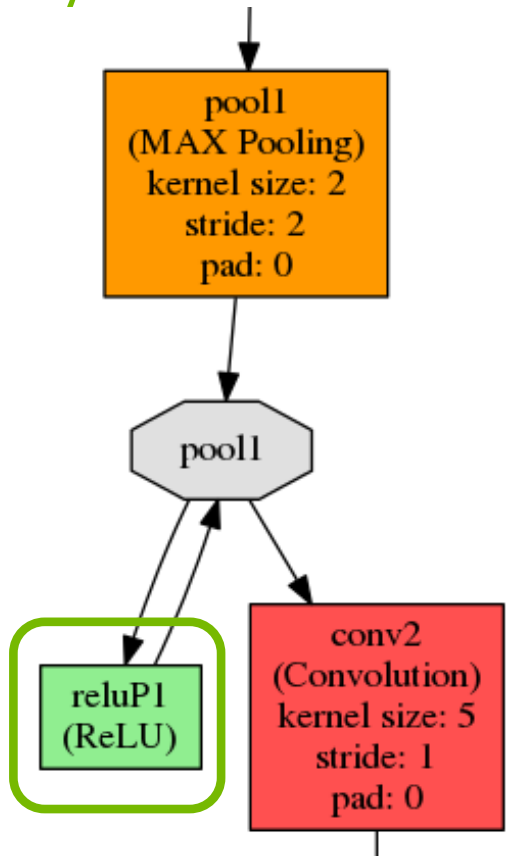
```
layer {
  name: "conv1"
  type: "Convolution"
        ...
        convolution_param {
        num_output: 75
        ...
layer {
        name: "conv2"
        type: "Convolution"
        ...
        convolution_param {
        num_output: 100
        ...
```



MODIFY THE NETWORK

Adding ReLU Layer

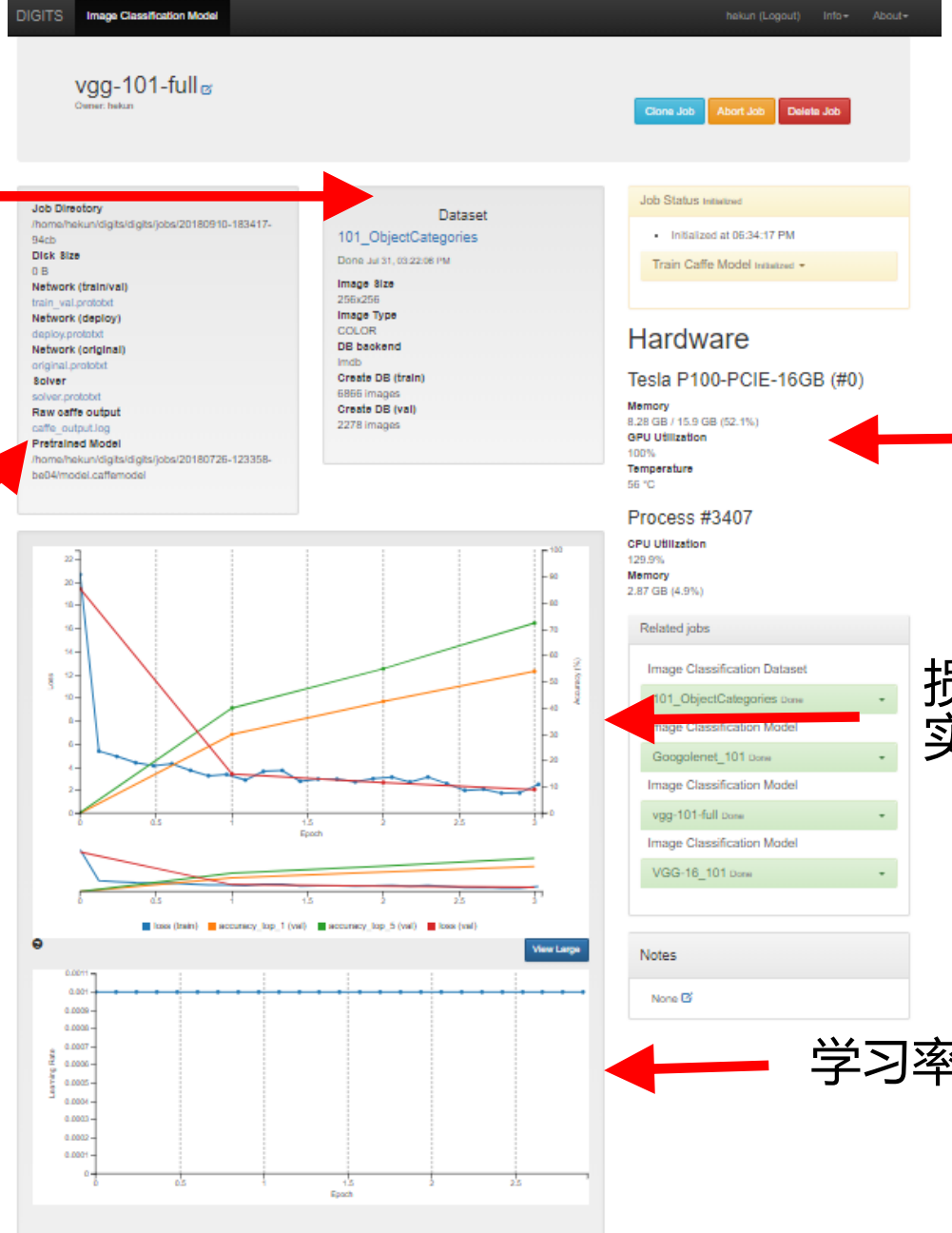




DIGITS – 训练过程监控

使用的数据集信息

模型训练配置文件



GPU工作状态

损失函数及精度的实时显示

学习率

DIGITS 的功能

图像分类

98% Dog

2% Cat

Classify images into classes or categories

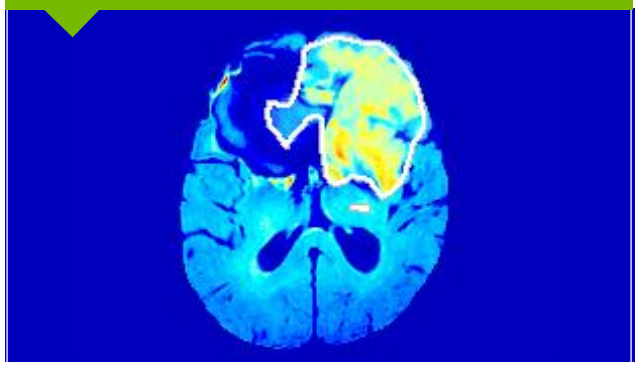
Object of interest could be anywhere in the image

目标检测

Find instances of objects in an image

Objects are identified with bounding boxes

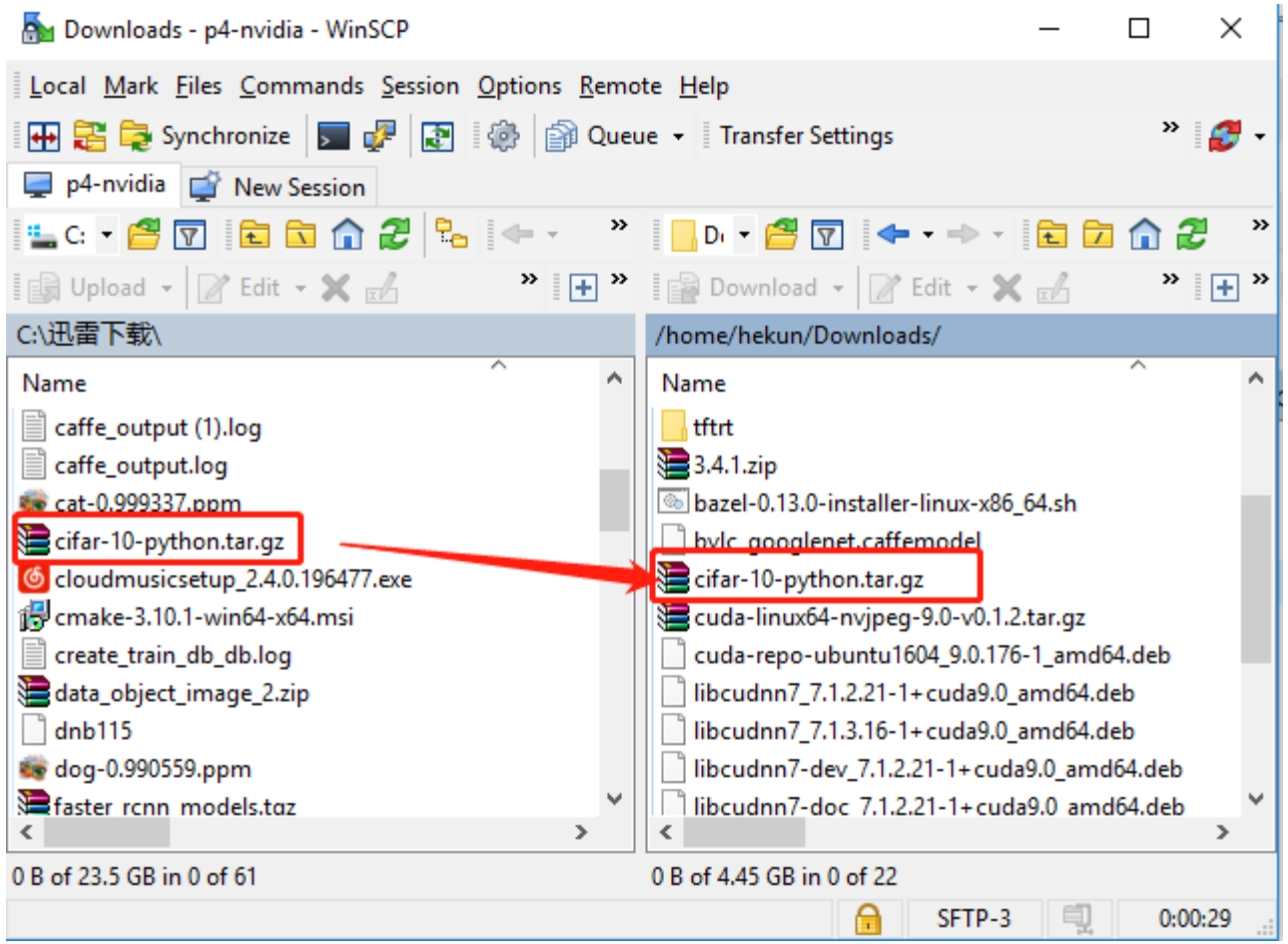
图像分割



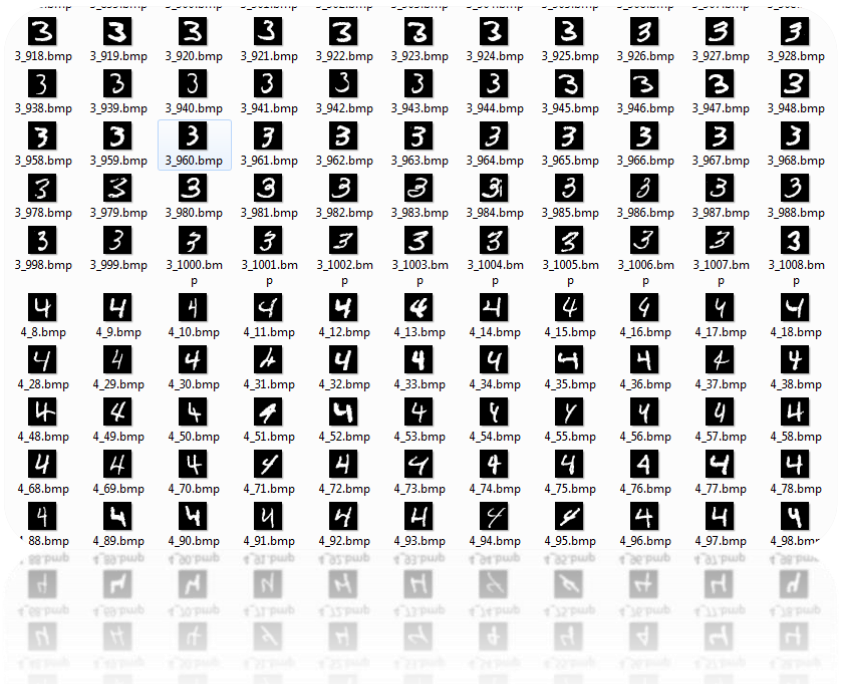
Partition image into multiple regions

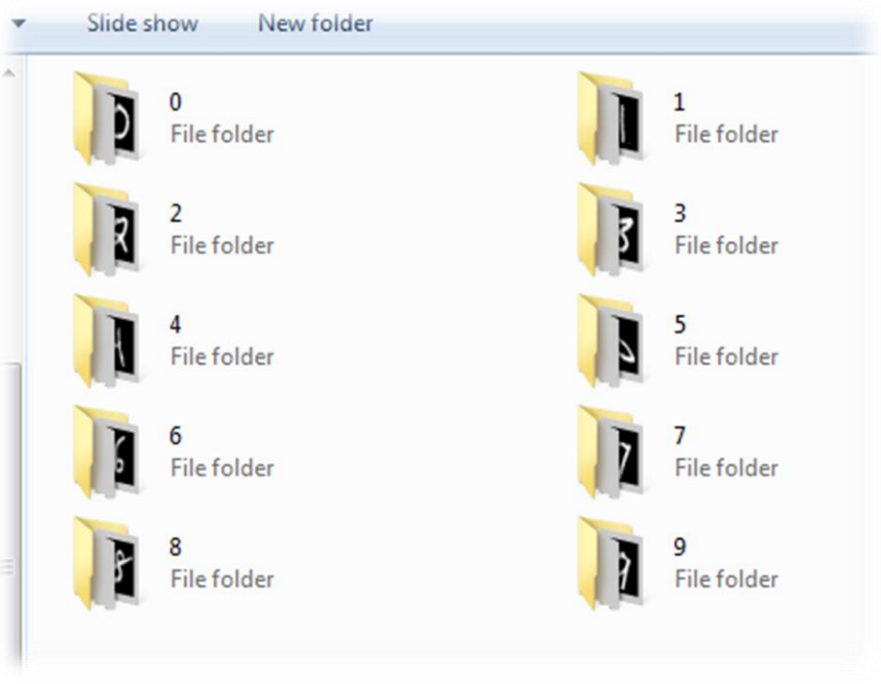
Regions are classified at the pixel level

DIGITS 的训练数据集导入



DIGITS： 图片分类的训练数据集格式

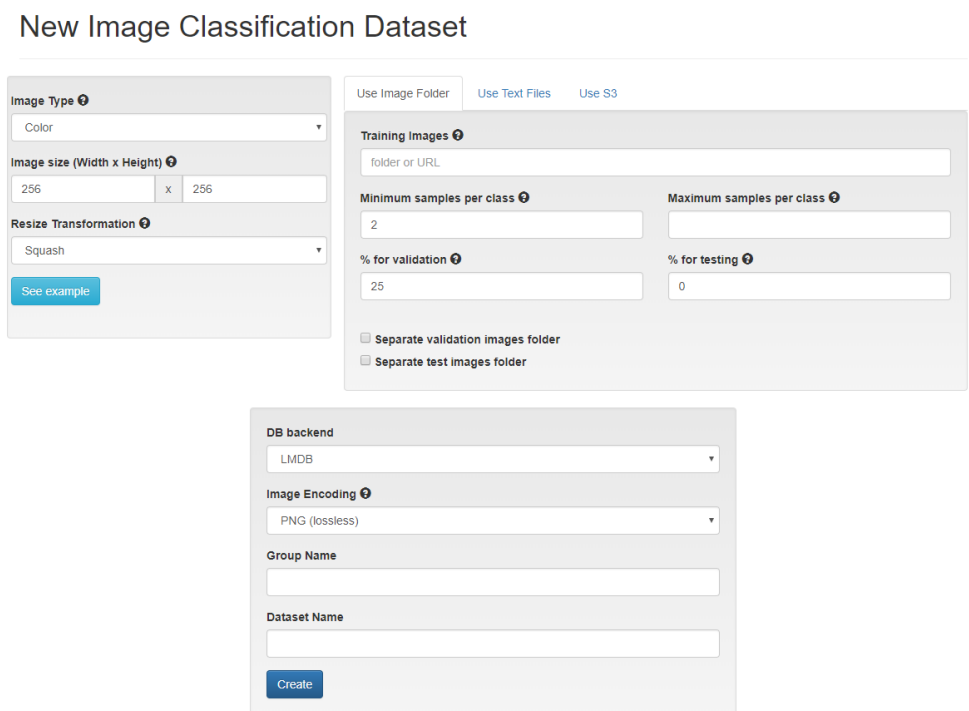




DIGITS： 图片分类

创建用于训练的数据集：

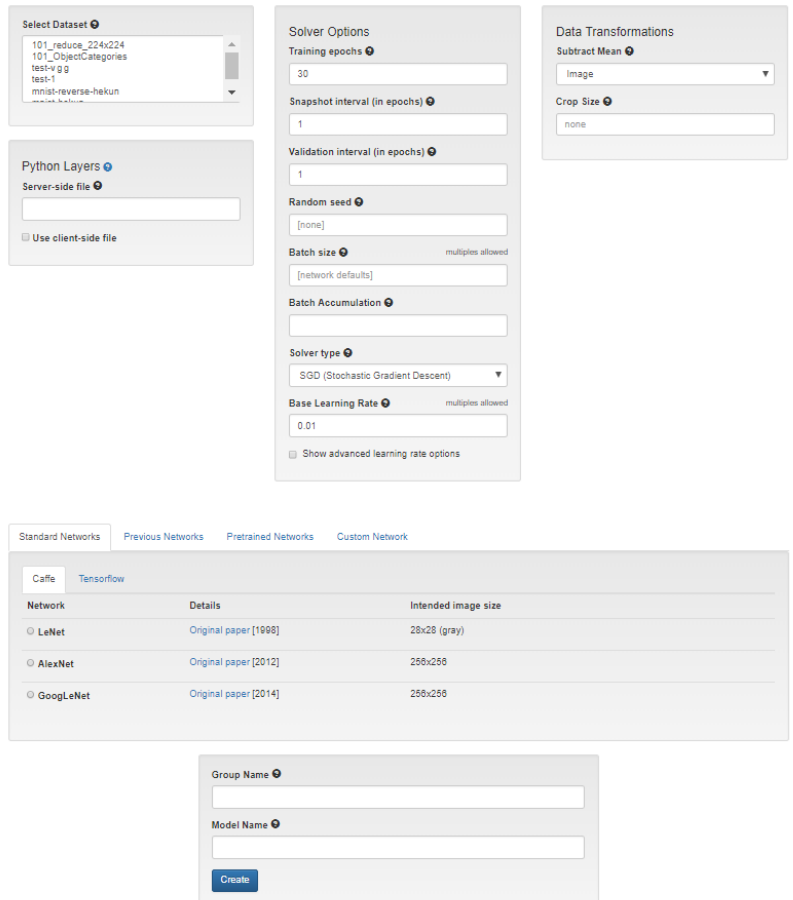
1. 定义图像类型、尺寸、变换方式等
2. 定义图片数据集在服务器中的地址或者网络中的地址
3. 定义训练集和验证集的比例大小以及详细信息
4. 定义图片的编码格式以及训练集的格式
5. 给数据集命名



DIGITS： 图片分类

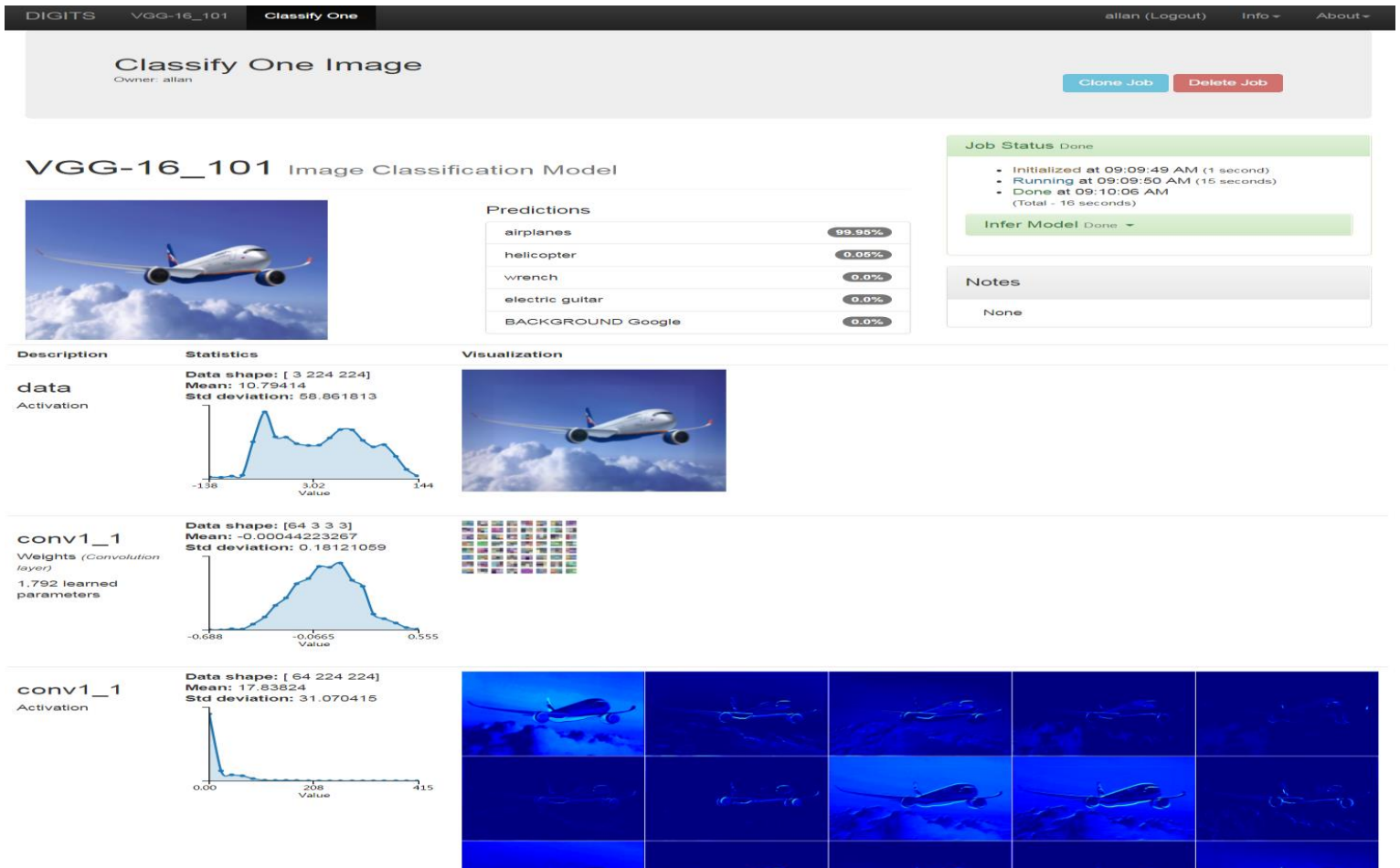
定义训练参数：

1. 选择已经创建好的数据集
2. 定义Epoch,batchsize,base_lr等训练参数
3. 选择使用的训练框架
4. 选择或者自定义网络结构
5. 给训练的模型命名



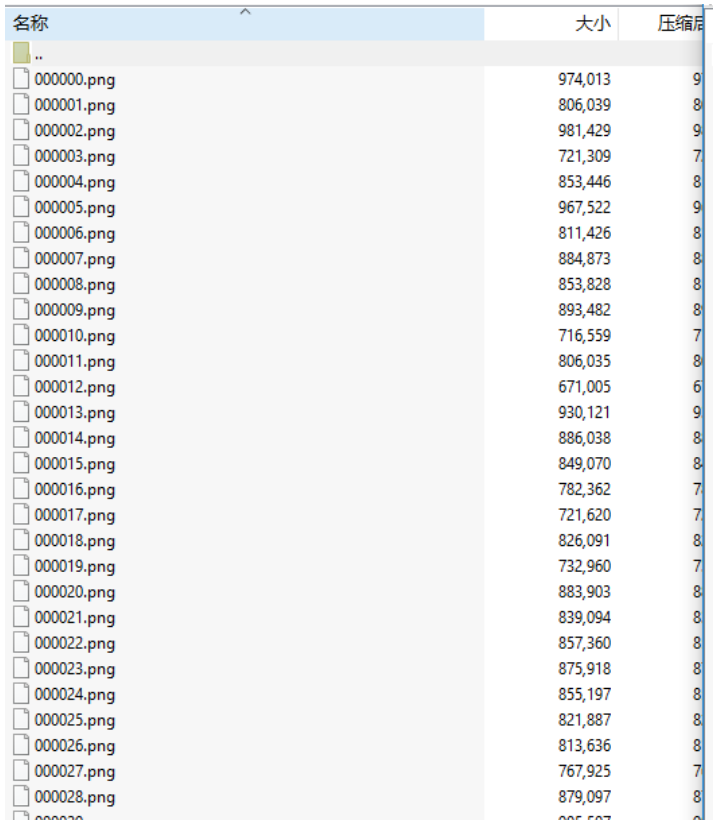
NVIDIA DIGITS

图片分类校验结果输出

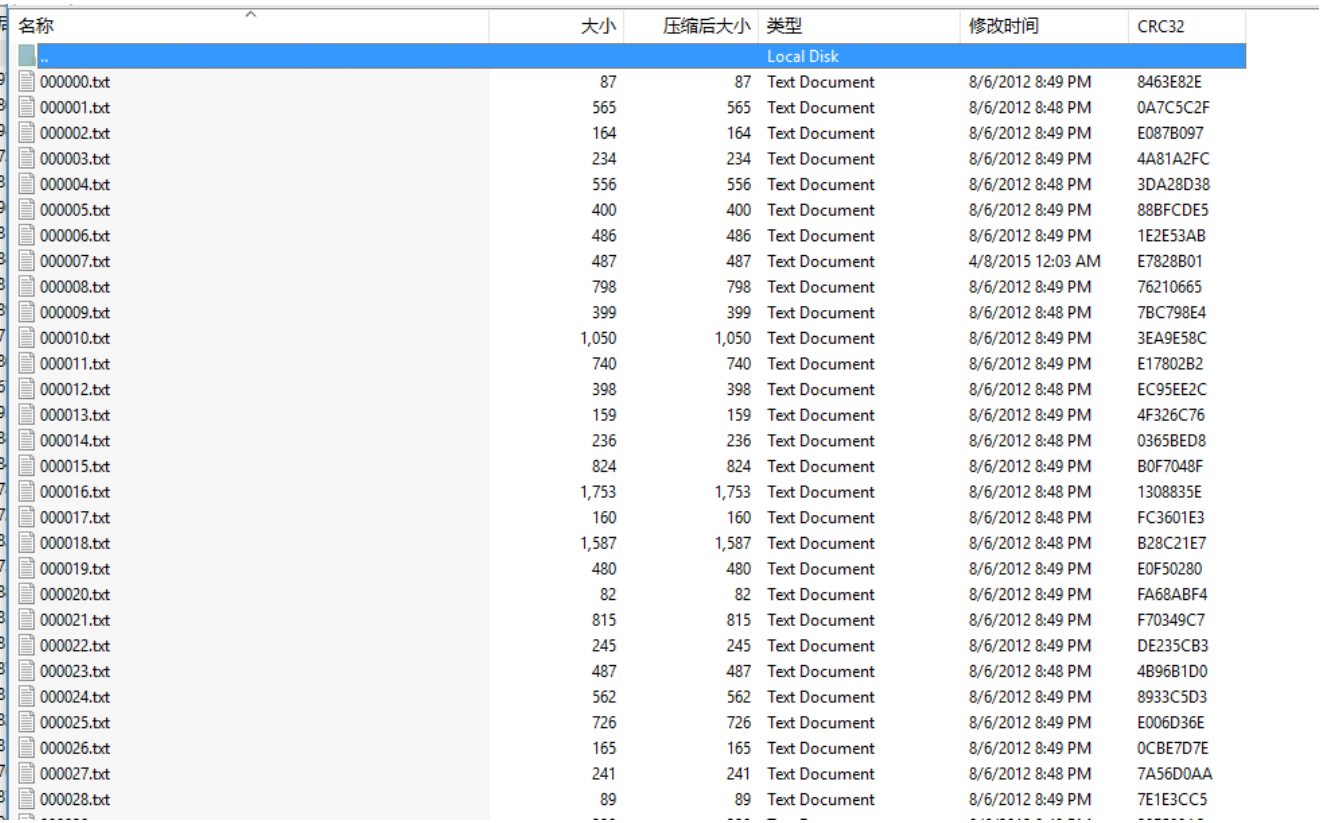


DIGITS： 目标检测的训练数据集格式

每张图片对应有一个和它相同名字的label文件，label文件中包含了对应图片中目标的类别和代表位置的矩形框信息



名称	大小	压缩后
..		
000000.png	974,013	9
000001.png	806,039	8
000002.png	981,429	9
000003.png	721,309	7
000004.png	853,446	8
000005.png	967,522	9
000006.png	811,426	8
000007.png	884,873	8
000008.png	853,828	8
000009.png	893,482	8
000010.png	716,559	7
000011.png	806,035	8
000012.png	671,005	6
000013.png	930,121	9
000014.png	886,038	8
000015.png	849,070	8
000016.png	782,362	7
000017.png	721,620	7
000018.png	826,091	8
000019.png	732,960	7
000020.png	883,903	8
000021.png	839,094	8
000022.png	857,360	8
000023.png	875,918	8
000024.png	855,197	8
000025.png	821,887	8
000026.png	813,636	8
000027.png	767,925	7
000028.png	879,097	8

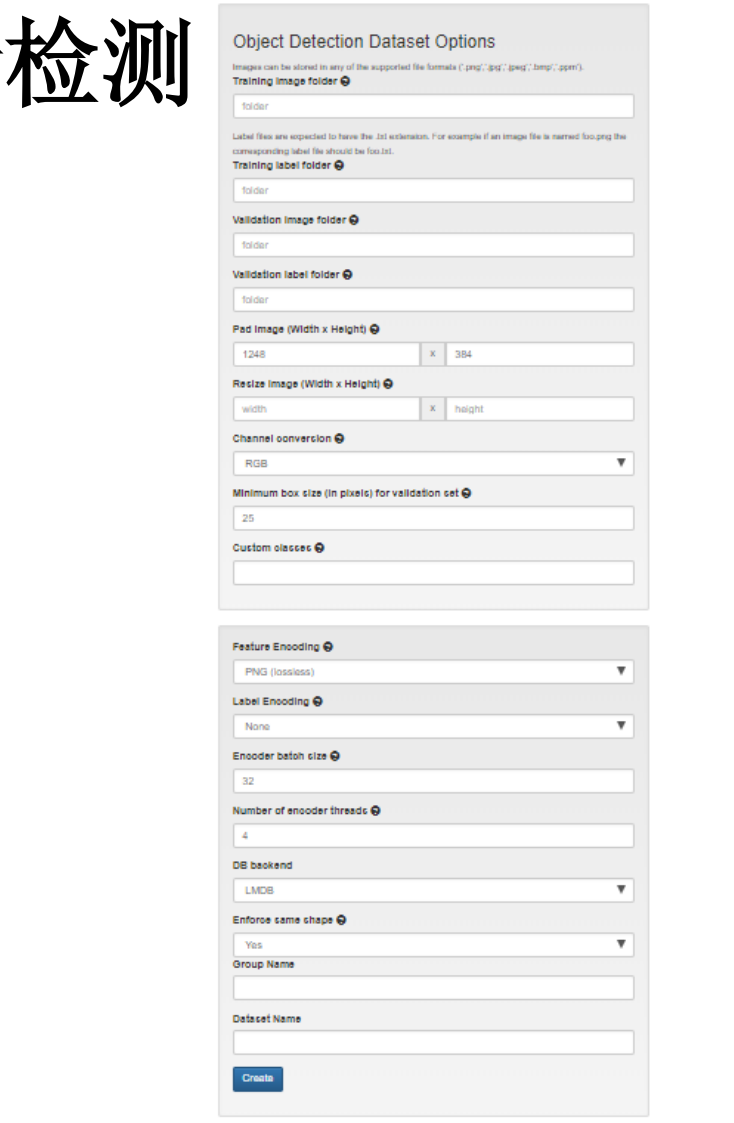


名称	大小	压缩后大小	类型	修改时间	CRC32
..			Local Disk		
000000.txt	87	87	Text Document	8/6/2012 8:49 PM	8463E82E
000001.txt	565	565	Text Document	8/6/2012 8:48 PM	0A7C5C2F
000002.txt	164	164	Text Document	8/6/2012 8:49 PM	E087B097
000003.txt	234	234	Text Document	8/6/2012 8:49 PM	4A81A2FC
000004.txt	556	556	Text Document	8/6/2012 8:48 PM	3DA28D38
000005.txt	400	400	Text Document	8/6/2012 8:49 PM	88BFCDE5
000006.txt	486	486	Text Document	8/6/2012 8:49 PM	1E2E53AB
000007.txt	487	487	Text Document	4/8/2015 12:03 AM	E7828B01
000008.txt	798	798	Text Document	8/6/2012 8:49 PM	76210665
000009.txt	399	399	Text Document	8/6/2012 8:48 PM	7BC798E4
000010.txt	1,050	1,050	Text Document	8/6/2012 8:49 PM	3EA9E58C
000011.txt	740	740	Text Document	8/6/2012 8:49 PM	E17802B2
000012.txt	398	398	Text Document	8/6/2012 8:48 PM	EC95EE2C
000013.txt	159	159	Text Document	8/6/2012 8:49 PM	4F326C76
000014.txt	236	236	Text Document	8/6/2012 8:48 PM	0365BED8
000015.txt	824	824	Text Document	8/6/2012 8:49 PM	B0F7048F
000016.txt	1,753	1,753	Text Document	8/6/2012 8:48 PM	1308835E
000017.txt	160	160	Text Document	8/6/2012 8:48 PM	FC3601E3
000018.txt	1,587	1,587	Text Document	8/6/2012 8:48 PM	B28C21E7
000019.txt	480	480	Text Document	8/6/2012 8:49 PM	E0F50280
000020.txt	82	82	Text Document	8/6/2012 8:49 PM	FA68ABF4
000021.txt	815	815	Text Document	8/6/2012 8:49 PM	F70349C7
000022.txt	245	245	Text Document	8/6/2012 8:49 PM	DE235CB3
000023.txt	487	487	Text Document	8/6/2012 8:48 PM	4B96B1D0
000024.txt	562	562	Text Document	8/6/2012 8:49 PM	8933C5D3
000025.txt	726	726	Text Document	8/6/2012 8:49 PM	E006D36E
000026.txt	165	165	Text Document	8/6/2012 8:49 PM	0CBE7D7E
000027.txt	241	241	Text Document	8/6/2012 8:48 PM	7A56D0AA
000028.txt	89	89	Text Document	8/6/2012 8:49 PM	7E1E3CC5

DIGITS： 目标检测

创建用于训练的数据集：

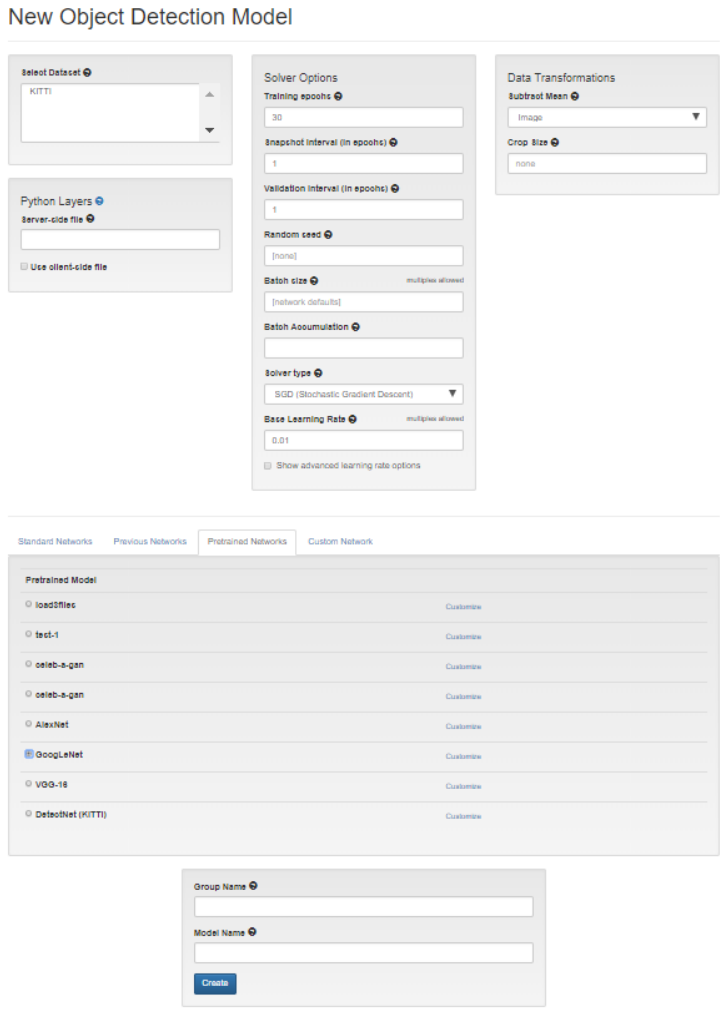
1. 定义图像类型、尺寸、变换方式等
2. 定义图片数据集在服务器中的地址
3. 定义样本标签在服务器中的地址
4. 定义训练集和验证集的比例大小以及详细信息
5. 定义图片的编码格式以及训练集的格式
6. 给数据集命名



DIGITS： 目标检测

定义训练参数：

1. 选择已经创建好的数据集
2. 定义Epoch,batchsize,base_lr等训练参数
3. 选择使用的训练框架
4. 选择或者自定义网络结构
5. 选择是否使用Pretrained Model
6. 给训练的模型命名



NVIDIA DIGITS

目标检测校验结果输出

DIGITS | detector-epochs-100 | Test One

Infer One Image

Owner: hekun

detector-epochs-100 Generic Image Model

Found 5 bounding boxes.

Source image

Inference visualization

bbox-list

DIGITS | Generic Image Model

Trained Models

Select Model

Epoch #100 | Download Model | Make Pretrained Model

Select Visualization Method

Bounding boxes

Raw Data

Bounding boxes

GAN

Image output

Image Segmentation

Object Detection

Show visualizations and statistics

Test

Visualization Options

Draw a bounding box around a detected object. The expected network output is a dictionary, keyed by class, where the values are lists of bounding boxes for that image. e.g. {'class1': [[l, t, r, b, confidence], ...], ...}

Test a single image

Image Path

/home/hekun/digits/examples/object-detection/kitti-

Upload image

Browse...

Show visualizations and statistics

Test One

Test a Database

LMDB path

Specify path to LMDB database on server

Test a list of images

Upload Image List

Browse...

Accepts a list of filenames or urls (you can use your val.txt file)

Image folder *(optional)*

Relative paths in the text file will be prepended with this value before reading

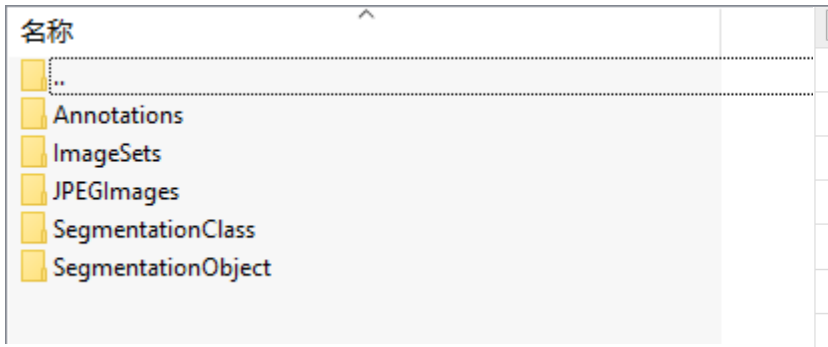
Number of images use from the file

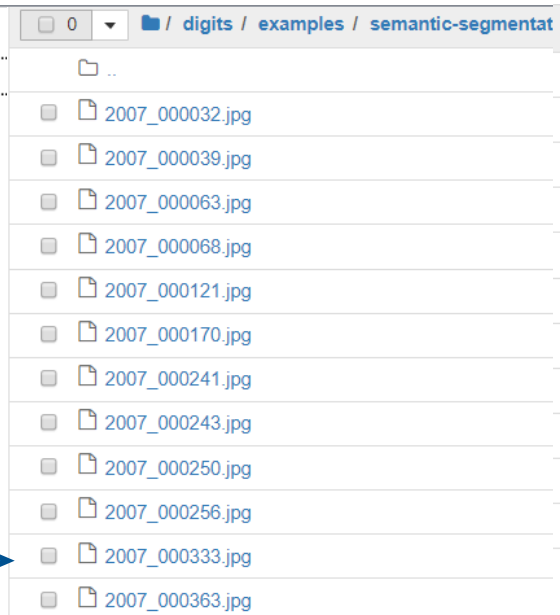
All

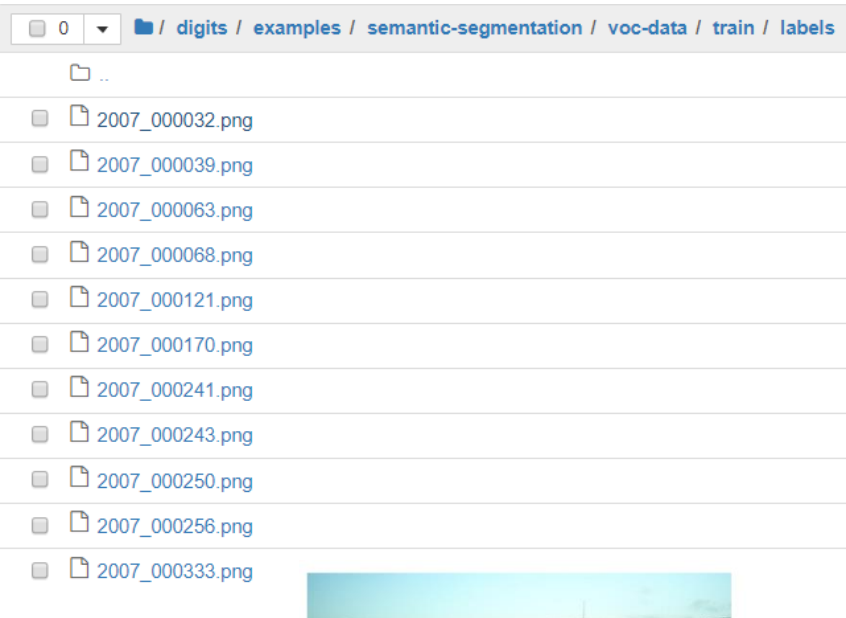
Leave blank to use all

Test Many

DIGITS： 图像分割的训练数据集格式







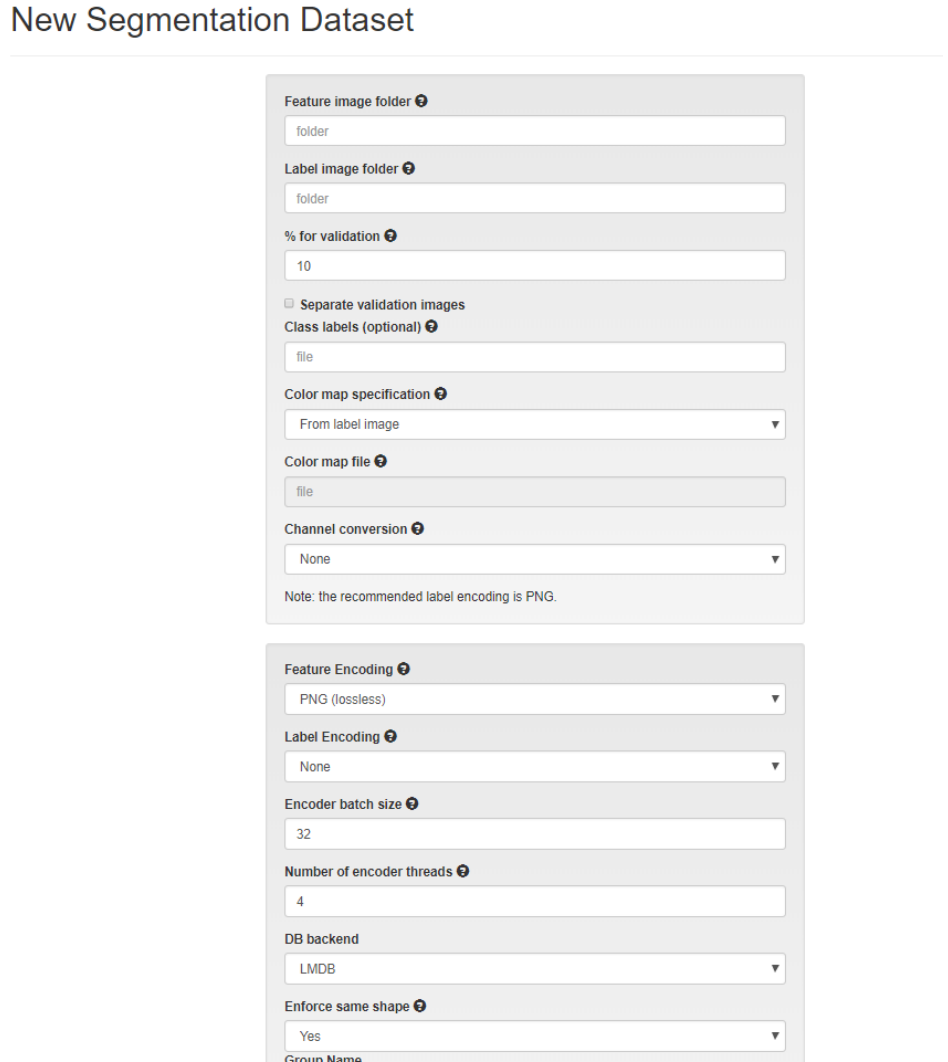

```
$ ./prepare_pascal_voc_data.sh /data/VOCtrainval_11-May-2012.tar ./voc-data
```

图像分割数据集中包含了图片样本，以及他们的label文件。这里的label文件与之前的目标检测不同的地方在于，这个label文件本身也是一个图片的形式。它标注了对应的图片样本中，目标的边界。

DIGITS： 图像分割

创建用于训练的数据集：

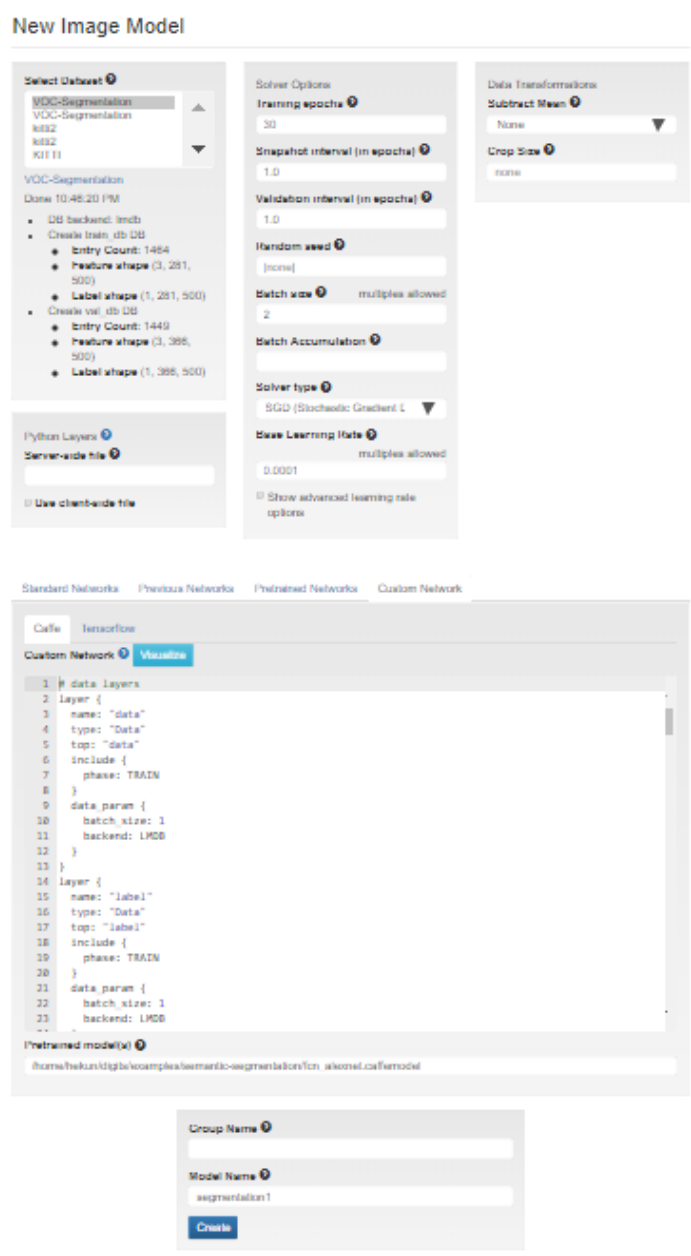
1. 定义图片数据集在服务器中的地址
2. 定义样本标签在服务器中的地址
3. 定义训练集和验证集的比例大小以及详细信息
4. 定义图片的编码格式以及训练集的格式
5. 定义样本标签的编码方式
6. 给数据集命名



DIGITS： 图像分割

定义训练参数：

1. 选择已经创建好的数据集
2. 定义Epoch,batchsize,base_lr等训练参数
3. 选择使用的训练框架
4. 选择或者自定义网络结构
5. 选择是否使用Pretrained Model
6. 给训练的模型命名



NVIDIA DIGITS

图像分割校验结果输出

DIGITS segmentation3 Test One hekun (Logout) Info About

Infer One Image

Owner: hekun

Clone Job Delete Job

segmentation3 Generic Image Model

Job Status Done

- Initialized at 06:41:22 AM (1 second)
- Running at 06:41:23 AM (8 seconds)
- Done at 06:41:31 AM

(Total - 9 seconds)

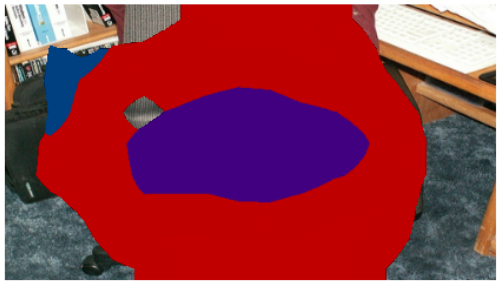
Infer Model Done

Notes

None

Source image

Inference visualization



chair dog tvmonitor

Trained Models

Select Model

Epoch #28

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)

Select Inference form

Segmentation

Show visualizations and statistics

Test

Test a single image

Image Path

/home/hekun/digits/examples/semantic-segmentati

Upload image

Browse...

Show visualizations and statistics

Test One

Test a list of images

Upload Image List

Browse...

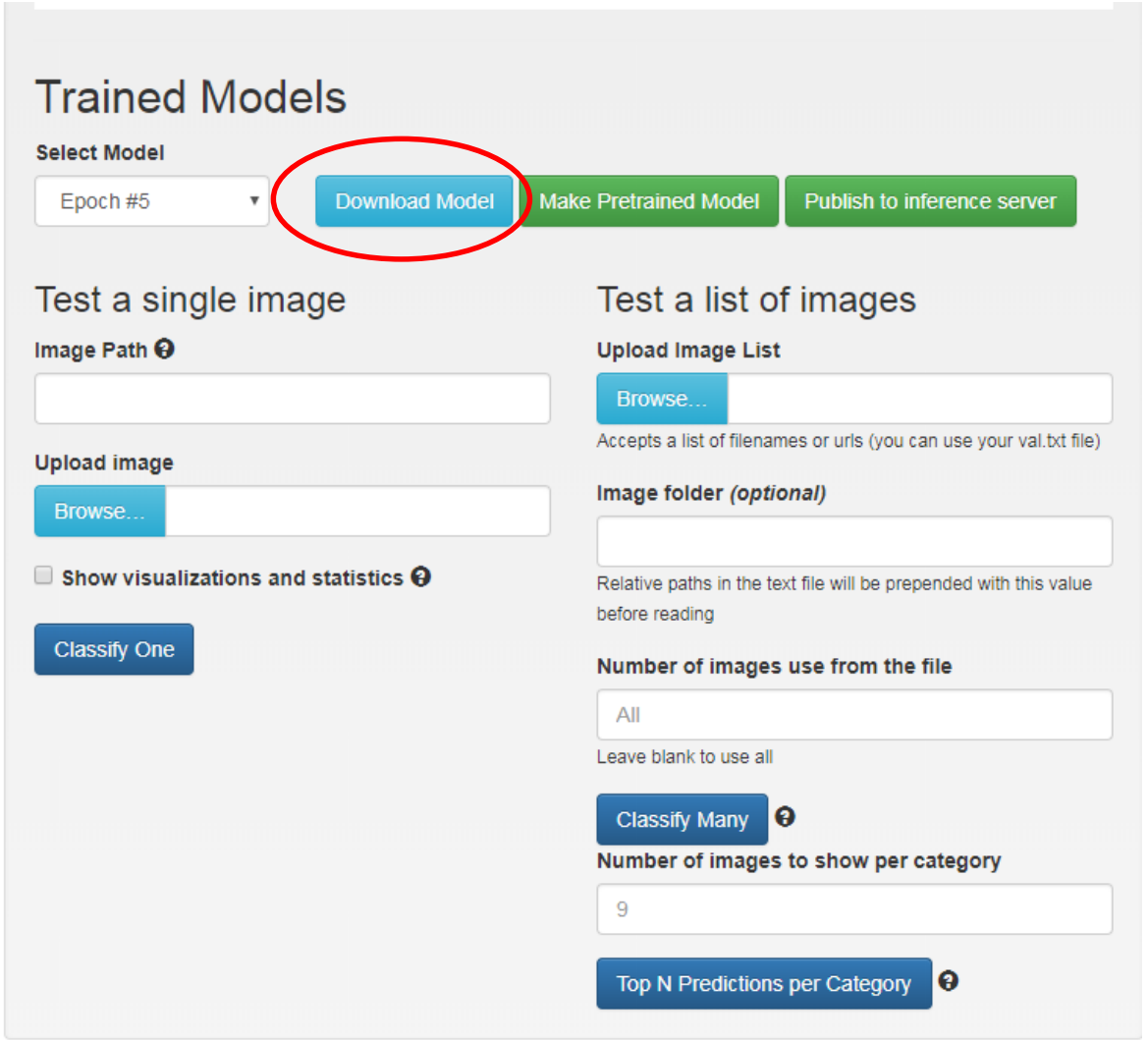
Accepts a list of filenames or urls (you can use your val.txt file)

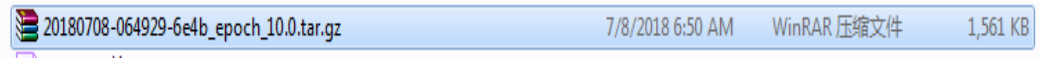
Image folder *(optional)*

Relative paths in the text file will be prepended with this value before reading

Number of images use from the file

NVIDIA DIGITS 模型输出

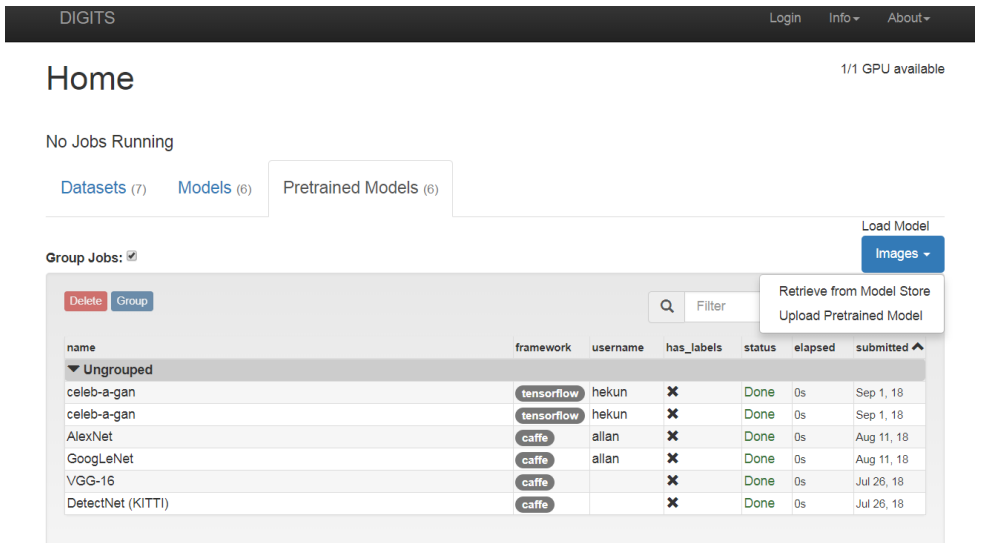


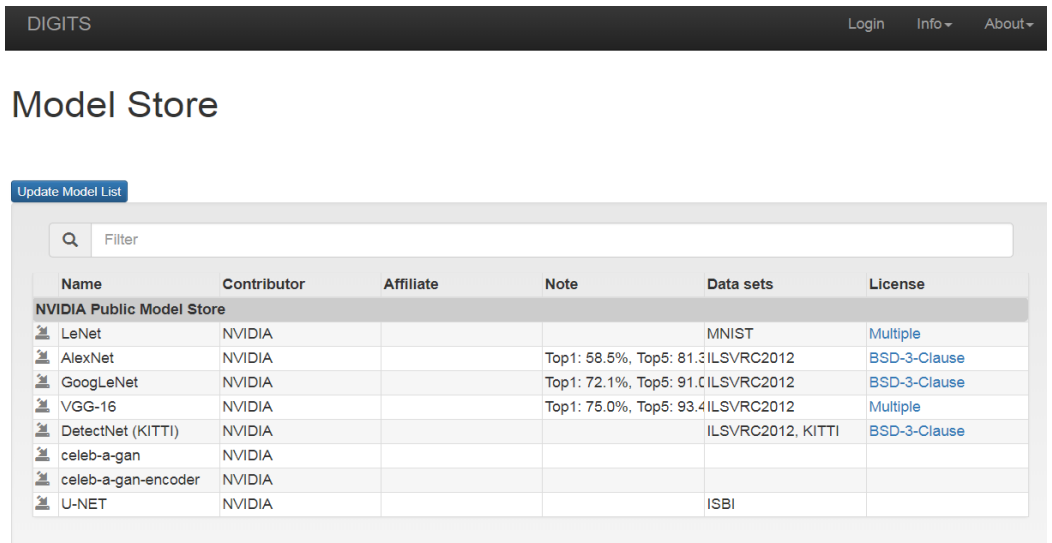


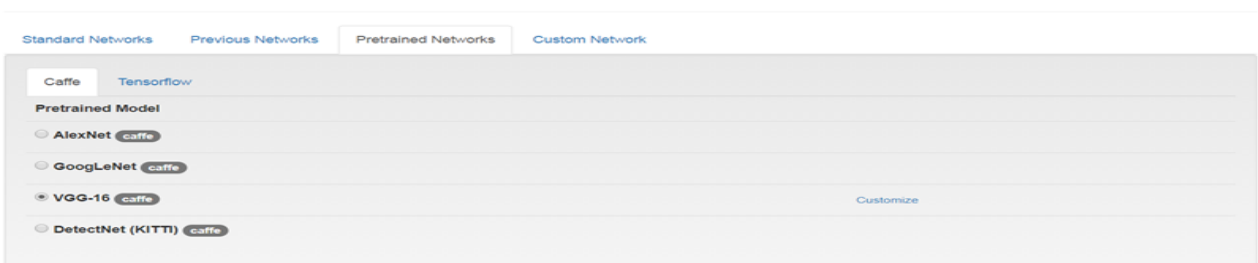
名称	大小	压缩后大小	类型	修改时间	CRC32
..			Local Disk		
deploy.prototxt	1,823	?	PROTOTXT File	7/8/2018 6:49 ...	
info.json	769	?	JSON File	1/1/1970 8:00 ...	
labels.txt	20	?	Text Document	7/8/2018 6:47 ...	
mean.binaryproto	3,147	?	BINARYPROTO File	7/8/2018 6:48 ...	
original.prototxt	2,346	?	PROTOTXT File	7/8/2018 6:49 ...	
snapshot_iter_7040.caffemodel	1,725,144	?	CAFFEMODEL File	7/8/2018 6:50 ...	
solver.prototxt	300	?	PROTOTXT File	7/8/2018 6:49 ...	
train_val.prototxt	2,600	?	PROTOTXT File	7/8/2018 6:49 ...	

迁移学习在DIGITS里的使用

加载model store里的模型





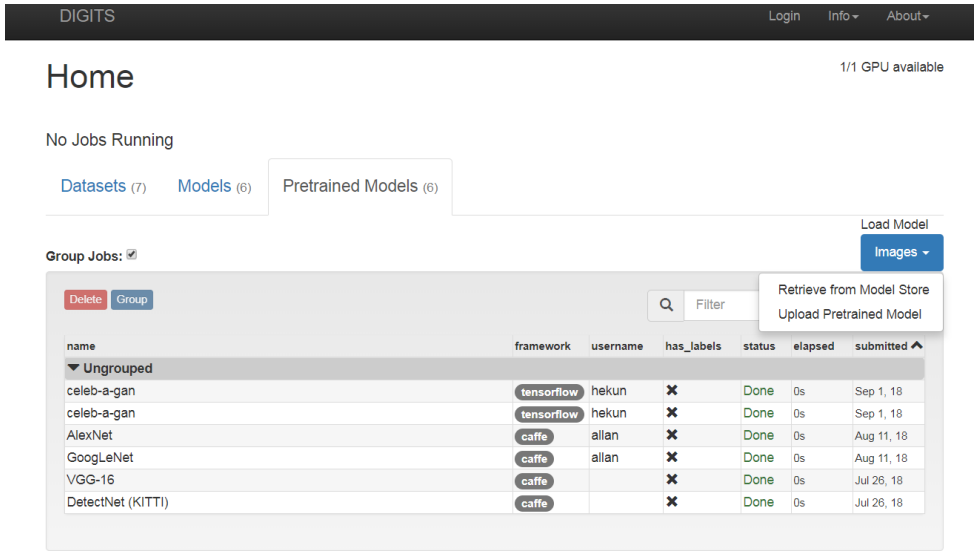


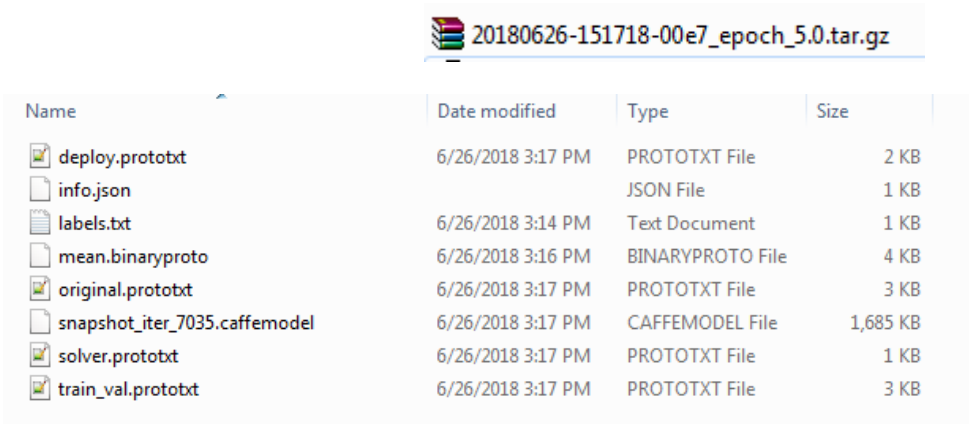
- Image Classification: VGG-16,
- Object Detection: DetectNet

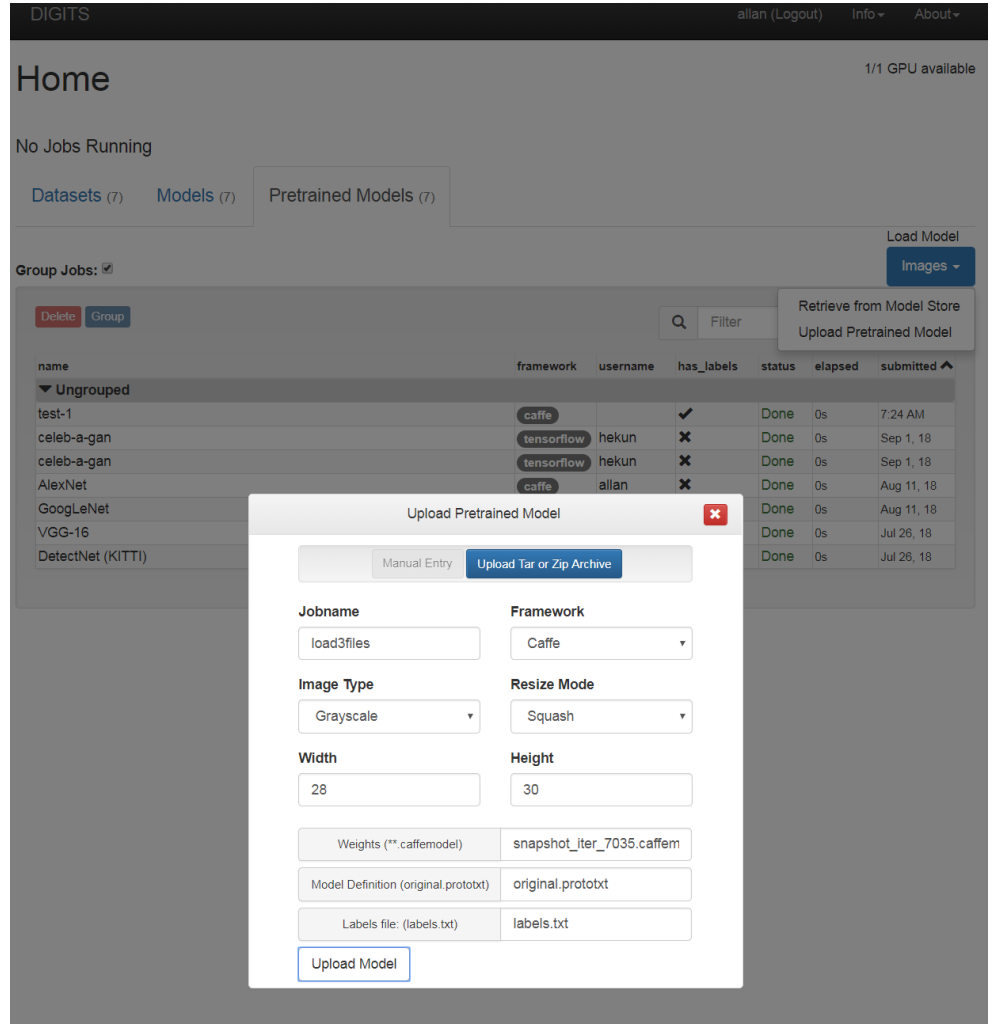
NVIDIA

迁移学习在DIGITS里的使用

加载自己的模型



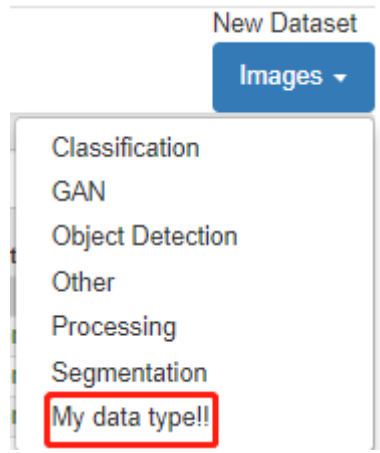




使用Plugin导入数据及自定义显示输出

GAN?
医疗影像分析?
文字语义识别?
更多类型的模型训练?

Plugin



Below is an example file tree for a data plugin:

```
sunnybook/
├── digitsDataPluginSunnybrook/
│   ├── templates
│   │   ├── dataset_template.html
│   │   └── inference_template.html
│   ├── __init__.py
│   ├── data.py
│   └── forms.py
├── MANIFEST.in
└── setup.py
```

```python
@subclass
class DatasetForm(Form):
    """
    A form used to create a Sunnybrook dataset
    """

    def validate_folder_path(form, field):
        if not field.data:
            pass
        else:
            # make sure the filesystem path exists
            if not os.path.exists(field.data) or not os.path.isdir(field.data):
                raise validators.ValidationError(
                    'Folder does not exist or is not reachable')
            else:
                return True

    image_folder = utils.forms.StringField(
        u'Image folder',
        validators=[
            validators.DataRequired(),
            validate_folder_path,
        ],
        tooltip="Specify the path to the image folder"
    )

    contour_folder = utils.forms.StringField(
        u'Contour folder',
        validators=[
            validators.DataRequired(),
            validate_folder_path,
        ],
        tooltip="Specify the path to the contour folder"
    )

    channel_conversion = utils.forms.SelectField(
        'Channel conversion',
        choices=[
            ('RGB', 'RGB'),
            ('L', 'Grayscale'),
        ],
        default='L',
        tooltip="Perform selected channel conversion."
    )

    folder_pct_val = utils.forms.IntegerField(
        u'% for validation',
        default=10,
        validators=[
```

如何得到DIGITS

- **Simple way:**
- OS – Ubuntu16.04
- Download link：
  https://developer.nvidia.com/digits

Others (from source code)：

- Download NVIDIA-Caffe：
  https://github.com/NVIDIA/caffe
- Download Digits：
  https://github.com/NVIDIA/DIGITS
- More Examples：
  https://github.com/NVIDIA/DIGITS/tree/master/examples

Recommended HW/SW environment:

- GPU Compute Capability > 3.5 (Kepler and later)， cuDNN v7
- OS – Ubuntu16.04

深度学习培训 @ GTC CHINA 2018

6大主题全中文首发 | 6 折限时特惠 | 全配置GPU云实验环境 | 全球标准培训证书

自动驾驶感知
深度学习模型训练及部署至自动驾驶汽车核心DrivePX

计算机视觉升级版
快速入门深度学习完整实践流程

自然语言处理
用深度学习方法完成文本分类和机器翻译

游戏开发
训练深度强化学习代理玩星际争霸2

多数据类型
1天解锁计算机视觉和自然语言处理两大技能

创建数字内容
为媒体娱乐业注入“深度学习”新活力

查看课程和报名

培训咨询