Model_Test_Prediction

January 30, 2018

1 Implementing Deep Learning CNN Model to Classify 14 common types of Thorax Diseases from NIH Chest Xray Dataset

- https://www.nih.gov/news-events/news-releases/nih-clinical-center-provides-one-largest-publ

- https://stanfordmlgroup.github.io/projects/chexnet/

```
In [1]: """
        The main CheXNet model implementation.
        """
        import os
        import numpy as np
        import torch
        import torch.nn as nn
        import torch.backends.cudnn as cudnn
        import torchvision
        #import torchvision.transforms as transforms
        from torchvision import models, transforms
        from torch.utils.data import DataLoader
        from read_data import ChestXrayDataSet
        from sklearn.metrics import roc_auc_score
        from PIL import Image
```

```
In [2]: print(torch.__version__)
```

```
0.3.0.post4
```

```
In [19]: ! pwd
```

```
/home/ubuntu/CheXNet
```

```
In [20]: ! ls
```

```
Data                          localization   read_data.py   sample_submission.csv
Model_Training.ipynb  model.pth.tar  read_data.pyc  testfile.ipynb
README.md               model.py             results.csv
```

```
In [5]: !ls -l Data/images/test_ |wc -l
```

```
12387
```

```
In [25]: CKPT_PATH = 'model.pth.tar'
         N_CLASSES = 14

         CLASS_NAMES = [ 'Atelectasis', 'Cardiomegaly', 'Effusion', 'Infiltration', 'Mass', 'No
                         'Pneumothorax', 'Consolidation', 'Edema', 'Emphysema', 'Fibrosis', 'Pl


         DATA_DIR = 'Data/images/NIH Chest X-ray Dataset '
         TEST_IMAGE_LIST = './Data/labels/test_list.txt'
         BATCH_SIZE = 64
```

```
In [26]: class DenseNet121(nn.Module):
             """Model modified.
             The architecture of our model is the same as standard DenseNet121
             except the classifier layer which has an additional sigmoid function.
             """
             def __init__(self, out_size):
                 super(DenseNet121, self).__init__()
                 self.densenet121 = torchvision.models.densenet121(pretrained=True)
                 num_ftrs = self.densenet121.classifier.in_features
                 self.densenet121.classifier = nn.Sequential(
                     nn.Linear(num_ftrs, out_size),
                     nn.Sigmoid()
                 )

             def forward(self, x):
                 x = self.densenet121(x)
                 return x
```

In [27]:

```
def compute_AUCs(gt, pred):
    """Computes Area Under the Curve (AUC) from prediction scores.
    Args:
        gt: Pytorch tensor on GPU, shape = [n_samples, n_classes]
          true binary labels.
        pred: Pytorch tensor on GPU, shape = [n_samples, n_classes]
          can either be probability estimates of the positive class,
          confidence values, or binary decisions.
    Returns:
        List of AUROCs of all classes.
    """
    AUROCs = []
    gt_np = gt.cpu().numpy()
    pred_np = pred.cpu().numpy()
    for i in range(N_CLASSES):
        AUROCs.append(roc_auc_score(gt_np[:, i], pred_np[:, i]))
    return AUROCs
```

In [28]:

```
!nvidia-smi
```

```
Mon Jan 29 19:02:32 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 367.48                 Driver Version: 367.48                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla K80           Off  | 0000:00:1E.0     Off |                    0 |
| N/A   49C    P0    72W / 149W |    320MiB / 11439MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|    0      4130    C   /home/ubuntu/anaconda2/bin/python              318MiB |
+-----------------------------------------------------------------------------+
```

```
In [29]: cudnn.benchmark = True

         # initialize and load the model
         model = DenseNet121(N_CLASSES).cuda()
         model = torch.nn.DataParallel(model).cuda()

         #model = DenseNet121(N_CLASSES)
         #model = torch.nn.DataParallel(model)

         if os.path.isfile(CKPT_PATH):
             print("=> loading checkpoint")
             checkpoint = torch.load(CKPT_PATH)
             model.load_state_dict(checkpoint['state_dict'])
             print("=> loaded checkpoint")
         else:
             print("=> no checkpoint found")
```

```
=> loading checkpoint
=> loaded checkpoint
```

```
In [30]: normalize = transforms.Normalize([0.485, 0.456, 0.406],
                                          [0.229, 0.224, 0.225])

         transform = transforms.Compose([
                                         transforms.Resize(256),
                                         #transforms.Scale(256),
                                         transforms.TenCrop(224),
                                         #ten_crop(224),
                                         transforms.Lambda
                                         (lambda crops: torch.stack([transforms.ToTensor()(crop
                                         transforms.Lambda
                                         (lambda crops: torch.stack([normalize(crop) for crop i
                                         ])

         test_dataset = ChestXrayDataSet(data_dir=DATA_DIR,
                                         image_list_file=TEST_IMAGE_LIST,
                                         transform= transform)
         test_loader = DataLoader(dataset=test_dataset, batch_size=BATCH_SIZE,
                                  shuffle=False, num_workers=8, pin_memory=True)
```

```
In [31]: #??transforms
```

```
In [32]: # initialize the ground truth and output tensor
         gt = torch.FloatTensor()
         gt = gt.cuda()
         pred = torch.FloatTensor()
         pred = pred.cuda()

         # switch to evaluate mode
         model.eval()
```

```
Out[32]: DataParallel(
           (module): DenseNet121(
             (densenet121): DenseNet(
               (features): Sequential(
                 (conv0): Conv2d (3, 64, kernel_size=(7, 7), stride=(2, 2), padding=(3, 3), bia
                 (norm0): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
                 (relu0): ReLU(inplace)
                 (pool0): MaxPool2d(kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), dilatio
                 (denseblock1): _DenseBlock(
                   (denselayer1): _DenseLayer(
                     (norm.1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (64, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer2): _DenseLayer(
                     (norm.1): BatchNorm2d(96, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (96, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer3): _DenseLayer(
                     (norm.1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (128, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer4): _DenseLayer(
                     (norm.1): BatchNorm2d(160, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (160, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer5): _DenseLayer(
                     (norm.1): BatchNorm2d(192, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (192, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer6): _DenseLayer(
                     (norm.1): BatchNorm2d(224, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (224, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                 )
                 (transition1): _Transition(
                   (norm): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
                   (relu): ReLU(inplace)
                   (conv): Conv2d (256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                   (pool): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, coun
                 )
                 (denseblock2): _DenseBlock(
                   (denselayer1): _DenseLayer(
                     (norm.1): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (128, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer2): _DenseLayer(
                     (norm.1): BatchNorm2d(160, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (160, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer3): _DenseLayer(
               (norm.1): BatchNorm2d(192, eps=1e-05, momentum=0.1, affine=True)
               (relu.1): ReLU(inplace)
               (conv.1): Conv2d (192, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
               (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
               (relu.2): ReLU(inplace)
               (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer4): _DenseLayer(
  (norm.1): BatchNorm2d(224, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (224, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer5): _DenseLayer(
  (norm.1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer6): _DenseLayer(
  (norm.1): BatchNorm2d(288, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (288, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer7): _DenseLayer(
  (norm.1): BatchNorm2d(320, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (320, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer8): _DenseLayer(
  (norm.1): BatchNorm2d(352, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (352, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer9): _DenseLayer(
                 (norm.1): BatchNorm2d(384, eps=1e-05, momentum=0.1, affine=True)
                 (relu.1): ReLU(inplace)
                 (conv.1): Conv2d (384, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                 (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                 (relu.2): ReLU(inplace)
                 (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
               )
               (denselayer10): _DenseLayer(
                 (norm.1): BatchNorm2d(416, eps=1e-05, momentum=0.1, affine=True)
                 (relu.1): ReLU(inplace)
                 (conv.1): Conv2d (416, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                 (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                 (relu.2): ReLU(inplace)
                 (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
               )
               (denselayer11): _DenseLayer(
                 (norm.1): BatchNorm2d(448, eps=1e-05, momentum=0.1, affine=True)
                 (relu.1): ReLU(inplace)
                 (conv.1): Conv2d (448, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                 (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                 (relu.2): ReLU(inplace)
                 (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
               )
               (denselayer12): _DenseLayer(
                 (norm.1): BatchNorm2d(480, eps=1e-05, momentum=0.1, affine=True)
                 (relu.1): ReLU(inplace)
                 (conv.1): Conv2d (480, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                 (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                 (relu.2): ReLU(inplace)
                 (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
               )
)
(transition2): _Transition(
  (norm): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (relu): ReLU(inplace)
  (conv): Conv2d (512, 256, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (pool): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, coun
)
(denseblock3): _DenseBlock(
  (denselayer1): _DenseLayer(
    (norm.1): BatchNorm2d(256, eps=1e-05, momentum=0.1, affine=True)
    (relu.1): ReLU(inplace)
    (conv.1): Conv2d (256, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
    (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
    (relu.2): ReLU(inplace)
    (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
  )
  (denselayer2): _DenseLayer(
                     (norm.1): BatchNorm2d(288, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (288, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer3): _DenseLayer(
                     (norm.1): BatchNorm2d(320, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (320, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer4): _DenseLayer(
                     (norm.1): BatchNorm2d(352, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (352, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer5): _DenseLayer(
                     (norm.1): BatchNorm2d(384, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (384, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer6): _DenseLayer(
                     (norm.1): BatchNorm2d(416, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (416, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer7): _DenseLayer(
                     (norm.1): BatchNorm2d(448, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (448, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer8): _DenseLayer(
               (norm.1): BatchNorm2d(480, eps=1e-05, momentum=0.1, affine=True)
               (relu.1): ReLU(inplace)
               (conv.1): Conv2d (480, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
               (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
               (relu.2): ReLU(inplace)
               (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer9): _DenseLayer(
  (norm.1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (512, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer10): _DenseLayer(
  (norm.1): BatchNorm2d(544, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (544, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer11): _DenseLayer(
  (norm.1): BatchNorm2d(576, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (576, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer12): _DenseLayer(
  (norm.1): BatchNorm2d(608, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (608, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer13): _DenseLayer(
  (norm.1): BatchNorm2d(640, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (640, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer14): _DenseLayer(
               (norm.1): BatchNorm2d(672, eps=1e-05, momentum=0.1, affine=True)
               (relu.1): ReLU(inplace)
               (conv.1): Conv2d (672, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
               (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
               (relu.2): ReLU(inplace)
               (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer15): _DenseLayer(
  (norm.1): BatchNorm2d(704, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (704, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer16): _DenseLayer(
  (norm.1): BatchNorm2d(736, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (736, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer17): _DenseLayer(
  (norm.1): BatchNorm2d(768, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (768, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer18): _DenseLayer(
  (norm.1): BatchNorm2d(800, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (800, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer19): _DenseLayer(
  (norm.1): BatchNorm2d(832, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (832, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer20): _DenseLayer(
                     (norm.1): BatchNorm2d(864, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (864, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer21): _DenseLayer(
                     (norm.1): BatchNorm2d(896, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (896, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer22): _DenseLayer(
                     (norm.1): BatchNorm2d(928, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (928, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer23): _DenseLayer(
                     (norm.1): BatchNorm2d(960, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (960, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer24): _DenseLayer(
                     (norm.1): BatchNorm2d(992, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (992, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                 )
                 (transition3): _Transition(
                   (norm): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
                   (relu): ReLU(inplace)
                   (conv): Conv2d (1024, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
                   (pool): AvgPool2d(kernel_size=2, stride=2, padding=0, ceil_mode=False, coun
                 )
                 (denseblock4): _DenseBlock(
                   (denselayer1): _DenseLayer(
               (norm.1): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True)
               (relu.1): ReLU(inplace)
               (conv.1): Conv2d (512, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
               (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
               (relu.2): ReLU(inplace)
               (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer2): _DenseLayer(
  (norm.1): BatchNorm2d(544, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (544, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer3): _DenseLayer(
  (norm.1): BatchNorm2d(576, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (576, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer4): _DenseLayer(
  (norm.1): BatchNorm2d(608, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (608, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer5): _DenseLayer(
  (norm.1): BatchNorm2d(640, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (640, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer6): _DenseLayer(
  (norm.1): BatchNorm2d(672, eps=1e-05, momentum=0.1, affine=True)
  (relu.1): ReLU(inplace)
  (conv.1): Conv2d (672, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
  (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
  (relu.2): ReLU(inplace)
  (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
)
(denselayer7): _DenseLayer(
                     (norm.1): BatchNorm2d(704, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (704, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer8): _DenseLayer(
                     (norm.1): BatchNorm2d(736, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (736, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer9): _DenseLayer(
                     (norm.1): BatchNorm2d(768, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (768, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer10): _DenseLayer(
                     (norm.1): BatchNorm2d(800, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (800, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer11): _DenseLayer(
                     (norm.1): BatchNorm2d(832, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (832, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer12): _DenseLayer(
                     (norm.1): BatchNorm2d(864, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (864, 128, kernel_size=(1, 1), stride=(1, 1), bias=False
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer13): _DenseLayer(
                     (norm.1): BatchNorm2d(896, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (896, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer14): _DenseLayer(
                     (norm.1): BatchNorm2d(928, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (928, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer15): _DenseLayer(
                     (norm.1): BatchNorm2d(960, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (960, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                   (denselayer16): _DenseLayer(
                     (norm.1): BatchNorm2d(992, eps=1e-05, momentum=0.1, affine=True)
                     (relu.1): ReLU(inplace)
                     (conv.1): Conv2d (992, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
                     (norm.2): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True)
                     (relu.2): ReLU(inplace)
                     (conv.2): Conv2d (128, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1,
                   )
                 )
                 (norm5): BatchNorm2d(1024, eps=1e-05, momentum=0.1, affine=True)
               )
               (classifier): Sequential(
                 (0): Linear(in_features=1024, out_features=14)
                 (1): Sigmoid()
               )
             )
           )
         )
```

```
In [33]: for i, (inp, target) in enumerate(test_loader):
             target = target.cuda()
             gt = torch.cat((gt, target), 0)
             bs, n_crops, c, h, w = inp.size()
             input_var = torch.autograd.Variable(inp.view(-1, c, h, w).cuda(), volatile=True)
```

```
    output = model(input_var)
    output_mean = output.view(bs, n_crops, -1).mean(1)
    pred = torch.cat((pred, output_mean.data), 0)
```

```
In [34]: !nvidia-smi
```

```
Mon Jan 29 19:29:04 2018
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 367.48                  Driver Version: 367.48                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla K80           Off  | 0000:00:1E.0     Off |                    0 |
| N/A   72C    P0   110W / 149W |   1425MiB / 11439MiB |     30%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID  Type  Process name                               Usage      |
|=============================================================================|
|    0      4130    C   /home/ubuntu/anaconda2/bin/python             1409MiB |
+-----------------------------------------------------------------------------+
```

```
In [35]: AUROCs = compute_AUCs(gt, pred)
         AUROC_avg = np.array(AUROCs).mean()
         print('The average AUROC is {AUROC_avg:.3f}'.format(AUROC_avg=AUROC_avg))
         for i in range(N_CLASSES):
             print('The AUROC of {} is {}'.format(CLASS_NAMES[i], AUROCs[i]))
```

```
The average AUROC is 0.843
The AUROC of Atelectasis is 0.829442091979
The AUROC of Cardiomegaly is 0.916518403854
The AUROC of Effusion is 0.887048087675
The AUROC of Infiltration is 0.714316169346
The AUROC of Mass is 0.859717812613
The AUROC of Nodule is 0.787343689144
The AUROC of Pneumonia is 0.774506342568
The AUROC of Pneumothorax is 0.872677426782
The AUROC of Consolidation is 0.814235394452
The AUROC of Edema is 0.893253766662
The AUROC of Emphysema is 0.925360031027
The AUROC of Fibrosis is 0.830390420068
The AUROC of Pleural_Thickening is 0.783100509535
The AUROC of Hernia is 0.910447650098
```