```
In [1]: %reload_ext autoreload
        %autoreload 2
        %matplotlib inline
```

```
In [2]: import torch
```

```
In [3]: torch.cuda.is_available()
Out[3]: True
```

```
In [4]: import os
        from tqdm import tqdm, tnrange, tqdm_notebook
        from pathlib import Path
        import re
        import numpy as np
        import matplotlib.pyplot as plt
        #import cv2
        import sys
        import scipy.ndimage
        # from mpl_toolkits.mplot3d.art3d import Poly3Dcollection
```

```
In [5]: #import pydicom
        #from pydicom.data import get_testdata_files
        #from pydicom.filereader import read_dicomdir
        #import pydicom.pixel_data_handlers.gdcm_handler as gdcm_handler
        # ! gdcm must be installed with conda install (conda install -c conda-forge gdcm)
        # pydicom.config.image_handlers = ['gdcm_handler']
```

```
In [6]: # import nibabel as nib
```

```
In [7]: from fastai.vision import *
        from fastai.metrics import *
        from fastai.callbacks import *
```

```
In [8]: #from fastai2.data.all import *
        #from fastai2.vision.core import *
```

```
In [9]: import pandas as pd
```

Define paths

```
In [10]: path_str = '/home/ubuntu/sfr-challenge/lungs/dataset'
         #path_str = '/Users/igorgarbuz/SoftDev/sfr-challenge/dataset'
```

```
In [11]: path = Path(path_str)
```

```
In [12]: path_p = path/'Pathologiques'
```

```
In [13]: path_n = path/'Normaux'
```

```
In [14]: path_train = path_str + '/train'
```

```
In [15]: test_path = path_str + '/Pathologiques/N7Q0jai/N7Q0jai'
```

Define fixed random seed

```
In [16]: np.random.seed(42)
```

Test section ==>

```
In [17]: # cell to run the experiments
```

<== End of test section

Train network

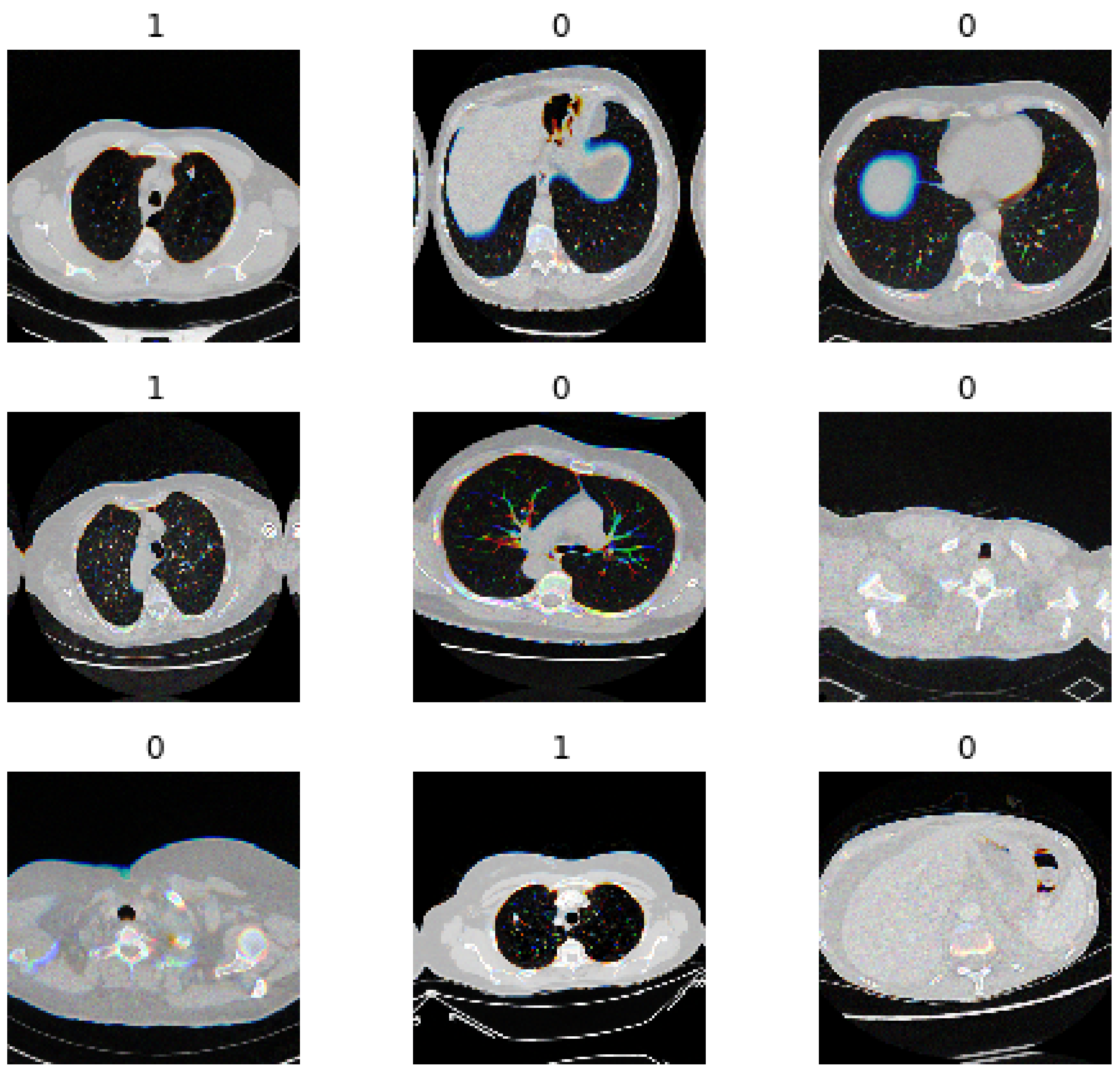
In [17]:
```
bs = 32
valid_split = 0.15
```

In [18]:
```
data = ImageDataBunch.from_folder(path/'train', ds_tfms=get_transforms(), size=224, bs=bs, valid_pc
```

In [19]:
```
pd.value_counts(data.train_dl.y.items.flatten(), sort=False)
```

Out[19]:
```
0    642
1    158
dtype: int64
```

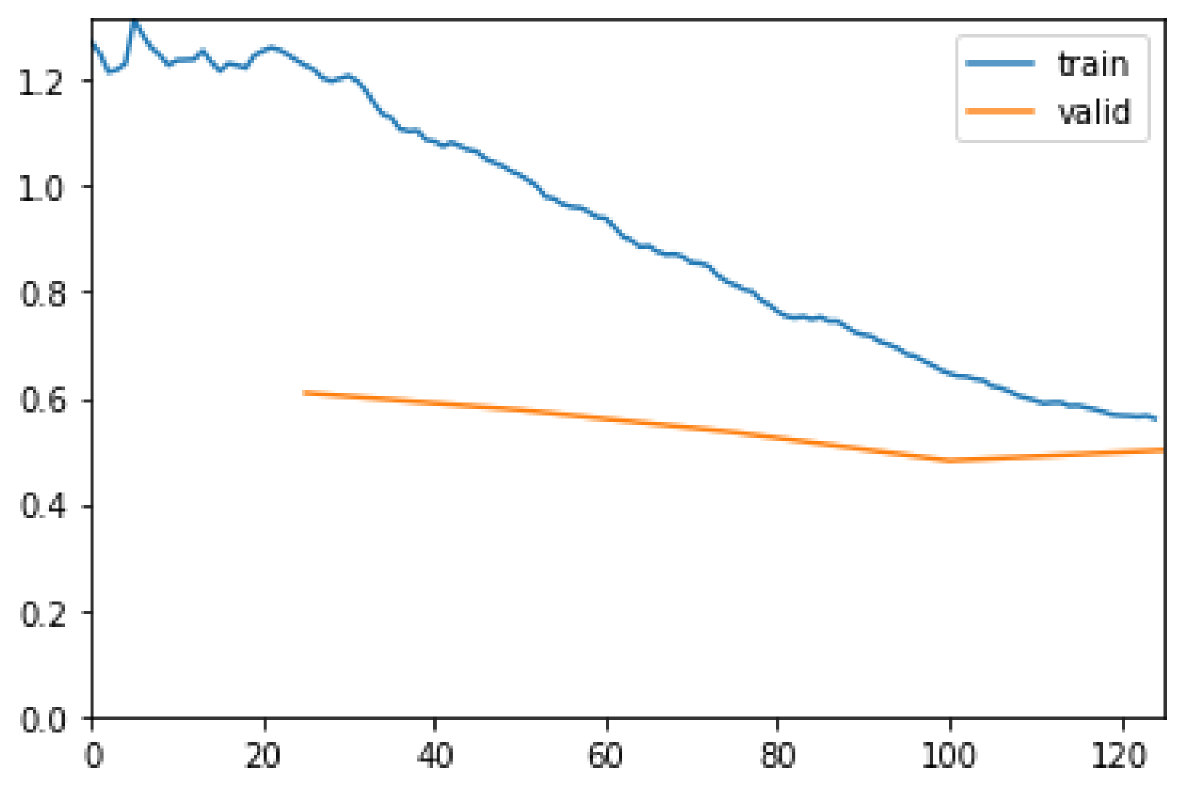
In [20]:
```
data.show_batch(rows=3, figsize=(7,6))
```



In [31]:
```
learner = cnn_learner(data, models.vgg16_bn, metrics=[error_rate, f1_score()], callback_fns=[ShowGr
#learner = cnn_learner(data, models.resnet18, metrics=[error_rate, f1_score(), AUROC()], callback_f
```

In [32]:
```
learner.fit_one_cycle(5)
```

epoch	train_loss	valid_loss	error_rate	f1_score	time
0	1.236137	0.609142	0.276596	0.338983	00:08
1	1.027317	0.577295	0.198582	0.416667	00:08
2	0.819671	0.535043	0.177305	0.468085	00:08
3	0.652925	0.481476	0.177305	0.509804	00:08
4	0.560919	0.501422	0.184397	0.535714	00:08


train
valid