4. Feature Extraction (Kaggle testing dataset)

Loading packages

In [1]:
```
from mxnet import autograd
from mxnet import gluon
import mxnet as mx
from mxnet import image
from mxnet import init
from mxnet import nd
from mxnet.gluon.data import vision

import math
import os
import shutil
import numpy as np
from collections import Counter

import numpy as np
# np.random.randint(256,480,1)[0]

# Change the following to mx.cpu() if you don't have GPU in your compute
# To use different GPU, you can try "ctx = mx.gpu(1)", where 1 is the fi
ctx = mx.gpu()
```

Setting parameters

In [2]:
```
data_dir   = 'data'
label_file = 'labels.csv'
# test_dir   = 'test'
input_dir  = 'for_test'
batch_size = 128
```

In [3]:
```
def transform_test(data, label):
    im1 = image.imresize(data.astype('float32') / 255, 288, 288)
    auglist1 = image.CreateAugmenter(data_shape = (3, 224, 224),
                                     resize     = 0,
                                     mean       = np.array([0.485,0.456,(
                                     std        = np.array([0.229,0.224,(
    im2 = image.imresize(data.astype('float32') / 255, 363, 363)
    auglist2 = image.CreateAugmenter(data_shape = (3, 299, 299),
                                     resize     = 0,
                                     mean       = np.array([0.485,0.456,(
                                     std        = np.array([0.229,0.224,(
    for aug in auglist1:
        im1 = aug(im1)
    im1 = nd.transpose(im1, (2,0,1))
    for aug in auglist2:
        im2 = aug(im2)
    im2 = nd.transpose(im2, (2,0,1))
    return (im1,im2, nd.array([label]).asscalar().astype('float32'))


def get_features_test(net1, net2, data):
    res_features = []
    inception_features = []
    labels = []
    for x1,x2,y in data:
        feature1 = net1(x1.as_in_context(ctx))
        res_features.append(feature1.asnumpy())
        feature2 = net2(x2.as_in_context(ctx))
        inception_features.append(feature2.asnumpy())
        labels.append(y.asnumpy())
    res_features = np.concatenate(res_features, axis=0)
    inception_features = np.concatenate(inception_features, axis=0)
    labels = np.concatenate(labels, axis=0)
    return res_features, inception_features, labels
```

In [4]:
```
input_str  = os.path.join('.', data_dir,  input_dir)
test_1     = vision.ImageFolderDataset(input_str, flag=1, transform=trans
loader     = gluon.data.DataLoader
test_data1 = loader(test_1, batch_size, shuffle=False, last_batch='keep'
```

In [5]:
```
from mxnet.gluon.model_zoo import vision as models
res151 = models.resnet152_v1(pretrained=True, ctx=ctx).features
with res151.name_scope():
    res151.add(gluon.nn.GlobalAvgPool2D())
res151.collect_params().reset_ctx(ctx)
res151.hybridize()
```

In [6]:
```
from mxnet.gluon.model_zoo import vision as models
import mxnet as mx
inception = models.inception_v3(pretrained=True, ctx=ctx)
inception_net = inception.features
inception_net.collect_params().reset_ctx(ctx)
inception_net.hybridize()
```

In [7]:
```
%%time

test_res151, test_inception, _ = get_features_test(res151, inception_net
test_res151 = test_res151.reshape(test_res151.shape[:2])
test_inception = test_inception.reshape(test_inception.shape[:2])
```

```
CPU times: user 12min 33s, sys: 1min 45s, total: 14min 19s
Wall time: 7min 41s
```

In [8]:
```
nd.save(os.path.join('.', data_dir, 'features_test_res.nd'), nd.array(te
nd.save(os.path.join('.', data_dir, 'features_test_incep.nd'), nd.array(
```