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

In [ ]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plot
import os

from fastai.imports import *
from fastai.dataset import *
from fastai.conv_learner import *
from fastai.plots import *
from fastai.model import *
from fastai.transforms import *
from fastai.sgdr import *
```

In [ ]:
```
PATH='data/'
```

In [ ]:
```
train_df = pd.read_csv(f'{PATH}train.csv')
test_df = pd.read_csv(f'{PATH}test.csv')
```

In [ ]:
```
valid_df = train_df.sample(frac=0.2, random_state=1337)
train_df = train_df.drop(valid_df.index)
```

In [ ]:
```
Y_train = train_df['label']
Y_valid = valid_df['label']
X_train = train_df.drop('label', axis=1)
X_valid = valid_df.drop('label', axis=1)
```

In [ ]:
```
X_train.shape,Y_train.shape,X_valid.shape,Y_valid.shape,test_df.shape
```

In [ ]:
```
img = X_train.iloc[0,:].values.reshape(28,28)
plot.imshow(img, cmap='gray')
Y_train.iloc[0]
```

In [ ]:
```
def reshape_img(matrix):
    return matrix.values.reshape(-1,28,28)
```

In [ ]:
```
def add_channel(matrix):
    matrix = np.stack((matrix,) * 3, axis = -1)
    return matrix
```

In [ ]:
```
def convert_ndarray(matrix):
    return matrix.values.flatten()
```

In [ ]:
```
X_trn = reshape_img(X_train)
X_trn = add_channel(X_trn)
X_val = reshape_img(X_valid)
X_val = add_channel(X_val)
X_t = reshape_img(test_df)
X_t = add_channel(X_t)
```

In [ ]:
```
Y_trn = convert_ndarray(Y_train)
Y_val = convert_ndarray(Y_valid)
```

In [ ]:
```
preprocessed_data = [X_trn, Y_trn, X_val, Y_val, X_t]
print([e.shape for e in preprocessed_data])
print([type(e) for e in preprocessed_data])
```

In [ ]:
```
!rm -rf {PATH}tmp
```

In [ ]:
```
arch = resnet50
sz = 28
classes = np.unique(Y_trn)
```

In [ ]:
```
tfms = tfms_from_model(arch, sz, aug_tfms=[RandomRotate(10)])
#trn_dl = DataLoader
```

In [ ]:
```
def get_augs():
    data = ImageClassifierData.from_arrays(path=PATH, trn =(X_trn, Y_trn
), val=(X_val,Y_val),
                                   classes=classes, test=X_t,tfms=tfms)
    print(type(data.aug_dl))
    x, _ = next(iter(data.aug_dl))
    return x[0]
```

In [ ]:
```
#%debug
ims = np.stack([get_augs() for i in range(6)])
plots(ims, rows=2)
```

In [ ]:
```
data = ImageClassifierData.from_arrays(path=PATH, trn =(X_trn, Y_trn), v
al=(X_val,Y_val),
                                   bs=64,classes=classes, test=X_t,tfms=tfm
s)
```

In [ ]:
```
learn = ConvLearner.pretrained(arch, data,ps=0.7,precompute=True)
```

In [ ]:
```
learn.lr_find()
```

In [ ]:
```
learn.sched.plot()
```

In [ ]:
```
lr = 1e-2
```

In [ ]:
```
learn.fit(lr, 2)
```

In [ ]:
```
learn.fit(lr, 3, cycle_len=1, cycle_mult=2)
```

In [ ]:
```
learn.precompute = False
learn.unfreeze()
```

In [ ]:
```
lrs = np.array([0.001, 0.0075, 0.01])
learn.fit(lrs,2)
```

In [ ]:
```
learn.fit(lrs,3, cycle_len=1, cycle_mult=2)
```

In [ ]:
```
learn.save('res_50_last')
```

In [ ]:
```
learn.load('res_50_last')
```

In [ ]:
```
preds, y = learn.TTA()
probs = np.mean(np.exp(preds), 0)
accuracy_np(probs, y)
```

In [ ]:
```
preds_test, y_test = learn.TTA(is_test=True)
probs_test = np.mean(np.exp(preds_test), 0)
probs_test.shape
```

In [ ]:
```
df = pd.DataFrame(probs_test)
df.index+=1
```

In [ ]:
```
df = df.assign(Label=df.values.argmax(axis=1))
```

In [ ]:
```
df = df.assign(ImageId = df.index.values)
```

In [ ]:
```
df = df.drop([0,1,2,3,4,5,6,7,8,9], axis=1)
```

In [ ]:
```
df=df[['ImageId', 'Label']]
```

In [ ]:
```
df.tail()
```

In [ ]:
```
df.to_csv(f'{PATH}sub_gary_gan_resnet50.csv', index=False)
```

In [ ]:
```
from IPython.display import FileLink, FileLinks
```

In [ ]:
```
FileLink(f'{PATH}sub_gary_gan_resnet50.csv')
```