In [19]:
```
from fastai import *        # Quick access to most common functionality
from fastai.text import *
```

In [20]:
```
torch.cuda.set_device(1)
```

In [21]:
```
path = untar_data(URLs.IMDB_SAMPLE)
```

In [22]:
```
df_train = pd.read_csv(path/'train.csv', header=None)
df_val = pd.read_csv(path/'valid.csv',header=None)
df_val.head(1)
```

Out[22]:

	0	1
0	1	This very funny British comedy shows what migh...

In [23]:
```
## Making a small example
df_train = df_train.iloc[:80,:]
df_val = df_val.iloc[:20,:]
# Add new column to simulate multi-label/multi-class
df_train['new1'] = 1
df_train['new2'] = 0
df_val['new1'] = 1
df_val['new2'] = 0

df_train = df_train[[0,'new1','new2',1]]
df_val = df_val[[0,'new1','new2',1]]
df_train.head(1)
```

Out[23]:

	0	new1	new2	1
0	0	1	0	Un-bleeping-believable! Meg Ryan doesn't even ...

In [24]:
```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 80 entries, 0 to 79
Data columns (total 4 columns):
0       80 non-null int64
new1    80 non-null int64
new2    80 non-null int64
1       80 non-null object
dtypes: int64(3), object(1)
memory usage: 2.6+ KB
```

In [25]:
```
data_path = Path('ULMFiT/imdb')
data_path
```

Out[25]:
```
PosixPath('ULMFiT/imdb')
```

In [26]:
```
import os
if not os.path.exists(data_path):
    os.makedirs(data_path)
```

In [27]:
```
df_train.to_csv(data_path/'train.csv',header=None,index=None)
df_val.to_csv(data_path/'valid.csv',header=None,index=None)
```

In [35]:
```
train_ds = TextDataset.from_df(data_path, df_train, n_labels=3)
data_lm = TextLMDataBunch.from_csv(data_path)
data_clas = TextClasDataBunch.from_csv(data_path, train='train', valid='valid',
                vocab=data_lm.train_ds.vocab, n_labels=3)
```

In [36]:
```
data_clas.train_ds.classes
```
Out[36]:
```
array([0, 1])
```
In [37]:
```
learn = RNNLearner.language_model(data_lm,
                pretrained_model=URLs.WT103, drop_mult=0.5)
learn.fit_one_cycle(1, 1e-2)
```
```
Total time: 00:01
epoch  train_loss  valid_loss  accuracy
1      4.254179    3.652339    0.272347  (00:01)
```
In [38]:
```
learn.save_encoder("lm")
```

```
In [39]:  learn = RNNLearner.classifier(data_clas)
           learn.metrics = []
           learn.load_encoder("lm")
           learn.fit_one_cycle(1, 1e-3)
```

0.00% [0/1 00:00<00:00]

epoch **train_loss** **valid_loss**

Interrupted

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-39-1c44f940481b> in <module>()
      2 learn.metrics = []
      3 learn.load_encoder("lm")
----> 4 learn.fit_one_cycle(1, 1e-3)

~/.local/lib/python3.6/site-packages/fastai/train.py in fit_one_cycle(learn, cyc_le
n, max_lr, moms, div_factor, pct_start, wd, callbacks, **kwargs)
     20     callbacks.append(OneCycleScheduler(learn, max_lr, moms=moms, div_factor
=div_factor,
     21                                         pct_start=pct_start, **kwargs))
---> 22     learn.fit(cyc_len, max_lr, wd=wd, callbacks=callbacks)
     23 
     24 def lr_find(learn:Learner, start_lr:Floats=1e-7, end_lr:Floats=10, num_it:i
nt=100, stop_div:bool=True, **kwargs:Any):

~/.local/lib/python3.6/site-packages/fastai/basic_train.py in fit(self, epochs, lr,
 wd, callbacks)
    160         callbacks = [cb(self) for cb in self.callback_fns] + listify(callba
cks)
    161         fit(epochs, self.model, self.loss_func, opt=self.opt, data=self.dat
a, metrics=self.metrics,
--> 162             callbacks=self.callbacks+callbacks)
    163 
    164     def create_opt(self, lr:Floats, wd:Floats=0.)->None:

~/.local/lib/python3.6/site-packages/fastai/basic_train.py in fit(epochs, model, lo
ss_func, opt, data, callbacks, metrics)
     92     except Exception as e:
     93         exception = e
---> 94         raise e
     95     finally: cb_handler.on_train_end(exception)
     96 

~/.local/lib/python3.6/site-packages/fastai/basic_train.py in fit(epochs, model, lo
ss_func, opt, data, callbacks, metrics)
     82             for xb,yb in progress_bar(data.train_dl, parent=pbar):
     83                 xb, yb = cb_handler.on_batch_begin(xb, yb)
---> 84                 loss = loss_batch(model, xb, yb, loss_func, opt, cb_handler
)
     85                 if cb_handler.on_batch_end(loss): break
     86 

~/.local/lib/python3.6/site-packages/fastai/basic_train.py in loss_batch(model, xb,
 yb, loss_func, opt, cb_handler)
     20 
     21     if not loss_func: return to_detach(out), yb[0].detach()
---> 22     loss = loss_func(out, *yb)
     23 
     24     if opt is not None:

~/.local/lib/python3.6/site-packages/torch/nn/functional.py in binary_cross_entropy
_with_logits(input, target, weight, size_average, reduce, reduction, pos_weight)
   1764 
   1765     if not (target.size() == input.size()):
```

```
-> 1766         raise ValueError("Target size ({}) must be the same as input size
 ({})".format(target.size(), input.size()))
   1767
   1768     return torch.binary_cross_entropy_with_logits(input, target, weight, po
s_weight, reduction)

ValueError: Target size (torch.Size([32])) must be the same as input size (torch.Si
ze([32, 2]))
```

In [ ]:
```
learn.fit_one_cycle(1)
```

In [40]:
```
learn.model.eval()
```

Out[40]:
```
SequentialRNN(
  (0): MultiBatchRNNCore(
    (encoder): Embedding(936, 400, padding_idx=1)
    (encoder_dp): EmbeddingDropout(
      (emb): Embedding(936, 400, padding_idx=1)
    )
    (rnns): ModuleList(
      (0): WeightDropout(
        (module): LSTM(400, 1150)
      )
      (1): WeightDropout(
        (module): LSTM(1150, 1150)
      )
      (2): WeightDropout(
        (module): LSTM(1150, 400)
      )
    )
    (input_dp): RNNDropout()
    (hidden_dps): ModuleList(
      (0): RNNDropout()
      (1): RNNDropout()
      (2): RNNDropout()
    )
  )
  (1): PoolingLinearClassifier(
    (layers): Sequential(
      (0): BatchNorm1d(1200, eps=1e-05, momentum=0.1, affine=True, track_running_st
ats=True)
      (1): Dropout(p=0.4)
      (2): Linear(in_features=1200, out_features=50, bias=True)
      (3): ReLU(inplace)
      (4): BatchNorm1d(50, eps=1e-05, momentum=0.1, affine=True, track_running_stat
s=True)
      (5): Dropout(p=0.1)
      (6): Linear(in_features=50, out_features=2, bias=True)
    )
  )
)
```

In [ ]:
```
preds = learn.get_preds(is_test=False)
preds[1]
```

In [ ]:
```
len(df_val), len(preds[1])
```