Text Generation using Long-Short Term Memory (LSTM) Model

1. Prepare & Explore Dataset

In [2]:

```
from tensorflow import keras
from keras.callbacks import LambdaCallback
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from keras.optimizers import RMSprop
from keras.utils.data_utils import get_file
import numpy as np
import random
import sys
import io
```

In [3]:

```
#read the corpus dataset
path = get_file(
    'nietzsche.txt',
    origin='https://s3.amazonaws.com/text-datasets/nietzsche.txt')
with io.open(path, encoding='utf-8') as f:
    text = f.read().lower()
print('corpus length:', len(text))
```

```
Downloading data from https://s3.amazonaws.com/text-datasets/nietzsche.txt (https://s3.amazonaws.com/tex
t-datasets/nietzsche.txt)
606208/600901 [==============================] - 1s 1us/step
corpus length: 600893
```

In [4]:

```
#prepare charachter-based model
chars = sorted(list(set(text)))
print('total chars:', len(chars))
char_indices = dict((c, i) for i, c in enumerate(chars))
indices_char = dict((i, c) for i, c in enumerate(chars))
```

```
total chars: 57
```

In [5]:

```
# cut the text in semi-redundant sequences of maxlen characters
maxlen = 40
step = 3
sentences = []
next_chars = []
for i in range(0, len(text) - maxlen, step):
    sentences.append(text[i: i + maxlen])
    next_chars.append(text[i + maxlen])
print('nb sequences:', len(sentences))

print('Vectorization...')
x = np.zeros((len(sentences), maxlen, len(chars)), dtype=np.bool)
y = np.zeros((len(sentences), len(chars)), dtype=np.bool)
for i, sentence in enumerate(sentences):
    for t, char in enumerate(sentence):
        x[i, t, char_indices[char]] = 1
    y[i, char_indices[next_chars[i]]] = 1
```

```
nb sequences: 200285
Vectorization...
```

2. Define the neural network architecture

In [6]:

```
# build the model: using single LSTM
model = Sequential()
model.add(LSTM(128, input_shape=(maxlen, len(chars))))
model.add(Dense(len(chars), activation='softmax'))
```

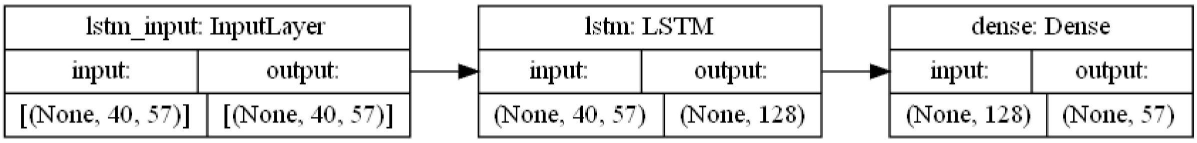
3. Compile the neural net

In [7]:

```
optimizer = RMSprop()
model.compile(loss='categorical_crossentropy', optimizer=optimizer)
```

In [8]:
```
keras.utils.plot_model(model, show_shapes=True, rankdir="LR")
```

Out[8]:



4. Fit / train the neural net

In [9]:
```
def sample(preds, temperature=1.0):
    # helper function to sample an index from a probability array
    preds = np.asarray(preds).astype('float64')
    preds = np.log(preds) / temperature
    exp_preds = np.exp(preds)
    preds = exp_preds / np.sum(exp_preds)
    probas = np.random.multinomial(1, preds, 1)
    return np.argmax(probas)
```

In [10]:
```
def on_epoch_end(epoch, _):
    # Function invoked at end of each epoch. Prints generated text.
    print()
    print('----- Generating text after Epoch: %d' % epoch)

    start_index = random.randint(0, len(text) - maxlen - 1)
    for diversity in [0.2, 0.5, 1.0, 1.2]:
        print('----- diversity:', diversity)
        generated = ''
        sentence = text[start_index: start_index + maxlen]
        generated += sentence
        print('----- Generating with seed: "' + sentence + '"')
        sys.stdout.write(generated)

        for i in range(400):
            x_pred = np.zeros((1, maxlen, len(chars)))
            for t, char in enumerate(sentence):
                x_pred[0, t, char_indices[char]] = 1.

            preds = model.predict(x_pred, verbose=0)[0]
            next_index = sample(preds, diversity)
            next_char = indices_char[next_index]

            sentence = sentence[1:] + next_char

            sys.stdout.write(next_char)
            sys.stdout.flush()
        print()
```

In [11]:
```
print_callback = LambdaCallback(on_epoch_end=on_epoch_end)

model.fit(x, y,
          batch_size=128,
          epochs=12,
          callbacks=[print_callback])
```

```
theoreminged, and popstician and now a gradanity philosophy and
the prates the givent: by it courous
rigate, and rack are inactulton for ous aspen
truild or all repariting lowaldess these concratence, who ascompine even ans afcentity of the uest. n
ow religious impain for menity, -vonewantly good.

27o  mube for a mesible that is make
who farthard see that chorost
f
----- diversity: 1.2
----- Generating with seed: "od."--this mode of reasoning savours of "
od."--this mode of reasoning savours of live:--this wouss
theolfy: cen supeiviag, but rache, this oul exferde, a certain and with
a consciouded, in even so eff-cceiry a hi
d1esless, anol only which is never tizgthl. therey reador and exmanterhasowitn meas! to weingry, come
no think
naclet,
or no cruak he is
coldo sucling schole be nehightly facce, were that the inserve, and they stard reselteng--wis, has, w
haw
```