In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import os
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Attention, Dense, Embedding, GRU, L
STM, SimpleRNN
```

In [2]:

```
# Ensure GPU is being recognized
print("Num GPUs available: ", len(tf.config.experimental.list_physical_
devices('GPU')))
```

```
Num GPUs available:  1
```

Prepare dataset

In [3]:

```
# Read in text file
with open('sherlock.txt', 'r', encoding='utf-8') as f:
    sherlock_data = f.read()

# Remove the 'table of contents' and start with stories
sherlock_data = sherlock_data[3376:]
```

Encode Features

In [4]:

```
# Dataset information
print(f'Length of data: {len(sherlock_data)} characters')
unique_chars = sorted(set(sherlock_data))
print(f'Number of unique characters: {len(unique_chars)}')
print(f'Unique Characters: {unique_chars}')
```

```
Length of data: 3378552 characters
Number of unique characters: 97
Unique Characters: ['\n', ' ', '!', '"', '&', "'", '(', ')', '*',
',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9',
':', ';', '?', 'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K',
'L', 'M', 'N', 'O', 'P', 'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y',
'Z', '[', ']', '`', 'a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j',
'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x',
'y', 'z', '£', '°', '½', 'ß', 'à', 'â', 'è', 'é', 'ê', 'î', 'ñ', 'ô',
'ö', 'û', 'ü', '’']
```

In [5]:
```
# Preview first 1000 characters
print(sherlock_data[:500])
```

```
          CHAPTER I
          Mr. Sherlock Holmes


     In the year 1878 I took my degree of Doctor of Medicine of the
     University of London, and proceeded to Netley to go through the
     course prescribed for surgeons in the army. Having completed my
     studies there, I was duly attached to the Fifth Northumberland
     Fusiliers as Assistant Surgeon. The regiment was stationed in In
dia
     at the time, and before I could join it, the second Afghan war h
ad
     broken out. On landing a
```

In [5]:
```
# Encode characters for training
index_mapping = {char:i for i, char in enumerate(unique_chars)}
sherlock_index = np.array([index_mapping[char] for char in sherlock_dat
a]) # map features to index
char_index = np.array(unique_chars)

print('Character Representation: ',sherlock_data[30:42])
print('Integer Representation: ', sherlock_index[30:42])
```

```
Character Representation:  Mr. Sherlock
Integer Representation:  [38 72 11  1 44 62 59 72 66 69 57 65]
```

Create Training Batches

In [6]:
```
# Create dataset from slices of Sherlock data
dataset_slices = tf.data.Dataset.from_tensor_slices(sherlock_index)

# Separate text into sequences
seq_len = 100
examples_per_epoch = int(len(sherlock_data)/(seq_len+1))
sequences = dataset_slices.batch(seq_len+1, drop_remainder=True)
```

In [7]:
```
# Split sequences into input and target sequences
def split_input_target(sequence):
    input_text = sequence[:-1]
    target_text = sequence[1:]
    return input_text, target_text

dataset = sequences.map(split_input_target)
```

In [8]:

```
# Break dataset into batches
BATCH_SIZE = 64
BUFFER_SIZE = 10000

dataset = dataset.shuffle(BUFFER_SIZE).batch(BATCH_SIZE, drop_remainder
=True)
dataset
```

Out[8]:

```
<BatchDataset shapes: ((64, 100), (64, 100)), types: (tf.int32, tf.in
t32)>
```

Train Models

In [9]:

```
# Model constants
vocab_size = len(unique_chars)
embedding_dim = 256
rnn_units = 1024
```

In [56]:

```
# Build RNN with GRU
def build_GRU_model(vocab_size, embedding_dim, rnn_units, batch_size):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim, batch_inpu
t_shape=[batch_size, None]),
        tf.keras.layers.GRU(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),
        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [12]:

```
# Create the model
model = build_GRU_model(vocab_size=len(unique_chars),
                        embedding_dim=embedding_dim,
                        rnn_units=rnn_units,
                        batch_size=BATCH_SIZE)

model.summary()
```

```
Model: "sequential"

_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding (Embedding)        (64, None, 256)           24832     

_________________________________________________________________
gru (GRU)                    (64, None, 1024)          3938304   

_________________________________________________________________
dense (Dense)                (64, None, 97)            99425     
=================================================================
Total params: 4,062,561
Trainable params: 4,062,561
Non-trainable params: 0

_________________________________________________________________
```

In [10]:

```
# Use sparse categorical crossentropy as loss
def loss(labels, logits):
    return tf.keras.losses.sparse_categorical_crossentropy(labels, logi
ts, from_logits=True)
```

In [14]:

```
# Compile the model
model.compile(optimizer='adam', loss=loss)
```

In [15]:

```
# Save checkpoints to access weights later
checkpoint_dir = './training_checkpoints'
checkpoint_prefix = os.path.join(checkpoint_dir, 'ckpt_{epoch}')

checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(filepath=check
point_prefix,
                                                          save_weights_on
ly=True)
```

In [16]:

```
# Train model
EPOCHS = 15
history = model.fit(dataset, epochs=EPOCHS, callbacks=[checkpoint_callb
ack])
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 45s 86ms/step - loss: 2.01
17
Epoch 2/15
522/522 [==============================] - 41s 78ms/step - loss: 1.35
87
Epoch 3/15
522/522 [==============================] - 41s 78ms/step - loss: 1.22
50
Epoch 4/15
522/522 [==============================] - 41s 79ms/step - loss: 1.16
49
Epoch 5/15
522/522 [==============================] - 41s 79ms/step - loss: 1.12
63
Epoch 6/15
522/522 [==============================] - 41s 78ms/step - loss: 1.09
70
Epoch 7/15
522/522 [==============================] - 41s 78ms/step - loss: 1.07
31
Epoch 8/15
522/522 [==============================] - 40s 76ms/step - loss: 1.05
19
Epoch 9/15
522/522 [==============================] - 39s 74ms/step - loss: 1.03
31
Epoch 10/15
522/522 [==============================] - 40s 77ms/step - loss: 1.01
66
Epoch 11/15
522/522 [==============================] - 40s 76ms/step - loss: 1.00
15
Epoch 12/15
522/522 [==============================] - 40s 77ms/step - loss: 0.98
72
Epoch 13/15
522/522 [==============================] - 42s 80ms/step - loss: 0.97
52
Epoch 14/15
522/522 [==============================] - 42s 81ms/step - loss: 0.96
37
Epoch 15/15
522/522 [==============================] - 42s 80ms/step - loss: 0.95
39
```

In [17]:

```
model = build_GRU_model(vocab_size, embedding_dim, rnn_units, batch_siz
e=1)
model.load_weights(tf.train.latest_checkpoint(checkpoint_dir))
model.build(tf.TensorShape([1, None]))
```

In [18]:

```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_1 (Embedding)      (1, None, 256)            24832     
_________________________________________________________________
gru_1 (GRU)                  (1, None, 1024)           3938304   
_________________________________________________________________
dense_1 (Dense)              (1, None, 97)             99425     
=================================================================
Total params: 4,062,561
Trainable params: 4,062,561
Non-trainable params: 0
_________________________________________________________________
```

In [17]:

```
def generate_text(model, start_string, num_generate=1000, temperature=
1.0):
    input_eval = [index_mapping[c] for c in start_string]
    input_eval = tf.expand_dims(input_eval, 0)

    generated_text = []

    model.reset_states()
    for i in range(num_generate):
        predictions = model(input_eval)
        predictions = tf.squeeze(predictions, 0)

        predictions = predictions / temperature
        predicted_id = tf.random.categorical(predictions, num_samples=
1)[-1, 0].numpy()

        input_eval = tf.expand_dims([predicted_id], 0)
        generated_text.append(char_index[predicted_id])

    return(start_string + ''.join(generated_text))
```

In [20]:
```
print(generate_text(model, start_string=u'There '))
```

```
There was Mr. Reuland I
     should not have something to decrepatid you will fail to tell yo
u than I.

     "'That with a shrink hat one
     generally is looking at the apologiency quite a bit.
     They appeared.

     Let it threatened by their light her for the tragedy.

     "When a man on the Underground, outwin lire adjustivess
     into the garret was able body and should hive reached this
     creature upon this mental detach."

     "That was the use of the express be put down your lather has
     been mistaken, but as I heard often and I was filled into it. I
gather that he
     walked so hu round, though I can be rather belay to be an idea.
The case besides, with a wooden saw the enormous
     reason why did you expect Dr. Professor Coram, will bring a
     return back to have been into the matter so such an unfortunate
Manor of the tould
     of us out of the latesteeparis again, Dr  round the room caught
the edme baskets. Fou before ever you want to
     introduce mutton, that
```

In [66]:
```
# Build RNN with LSTM
def build_LSTM_model(vocab_size, embedding_dim, rnn_units, batch_size):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim, batch_inpu
t_shape=[batch_size, None]),
        tf.keras.layers.LSTM(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),
        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [29]:

```
model2 = build_LSTM_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=BATCH_SIZE)

model2.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_4 (Embedding)      (64, None, 256)           24832     
_________________________________________________________________
lstm_2 (LSTM)                (64, None, 1024)          5246976   
_________________________________________________________________
dense_4 (Dense)              (64, None, 97)            99425     
=================================================================
Total params: 5,371,233
Trainable params: 5,371,233
Non-trainable params: 0
_________________________________________________________________
```

In [30]:

```
model2.compile(optimizer='adam', loss=loss)
```

In [31]:

```
model2.fit(dataset, epochs=EPOCHS)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 51s 98ms/step - loss: 1.91
41
Epoch 2/15
522/522 [==============================] - 50s 96ms/step - loss: 1.34
45
Epoch 3/15
522/522 [==============================] - 50s 95ms/step - loss: 1.22
29
Epoch 4/15
522/522 [==============================] - 49s 94ms/step - loss: 1.16
41
Epoch 5/15
522/522 [==============================] - 49s 94ms/step - loss: 1.12
61
Epoch 6/15
522/522 [==============================] - 49s 94ms/step - loss: 1.09
71 1s -
Epoch 7/15
522/522 [==============================] - 49s 95ms/step - loss: 1.07
23
Epoch 8/15
522/522 [==============================] - 49s 93ms/step - loss: 1.05
06
Epoch 9/15
522/522 [==============================] - 50s 95ms/step - loss: 1.03
13
Epoch 10/15
522/522 [==============================] - 51s 97ms/step - loss: 1.01
19
Epoch 11/15
522/522 [==============================] - 53s 101ms/step - loss: 0.9
946
Epoch 12/15
522/522 [==============================] - 50s 97ms/step - loss: 0.97
76
Epoch 13/15
522/522 [==============================] - 50s 97ms/step - loss: 0.96
12
Epoch 14/15
522/522 [==============================] - 51s 98ms/step - loss: 0.94
49
Epoch 15/15
522/522 [==============================] - 49s 94ms/step - loss: 0.92
94
```

Out[31]:

```
<tensorflow.python.keras.callbacks.History at 0x202c122f688>
```

In [32]:

```
model2.save_weights('./training_checkpoints/LSTM_checkpoint')
```

In [12]:

```
model2 = build_LSTM_model(vocab_size=len(unique_chars),
                          embedding_dim=embedding_dim,
                          rnn_units=rnn_units,
                          batch_size=1)

model2.load_weights('./training_checkpoints/LSTM_checkpoint')
model2.build(tf.TensorShape([1, None]))
```

In [13]:

```
model2.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding (Embedding)        (1, None, 256)            24832     
_________________________________________________________________
lstm (LSTM)                  (1, None, 1024)           5246976   
_________________________________________________________________
dense (Dense)                (1, None, 97)             99425     
=================================================================
Total params: 5,371,233
Trainable params: 5,371,233
Non-trainable params: 0
_________________________________________________________________
```

In [16]:

```
print(generate_text(model2, start_string='There '))
```

```
There got away into the house," said Holmes. "I wion my friend jealou
s, as you tell me
     that he and I have seen anything more like a long one to applaus
e. It was
     some story of the middle of it.

     "What is it, then?"

     "Well, you dejected that, there would be all that is better than
ks."

     There was a few months back to the neck in front. Take my ears a
s I
     ended by crumpling of the sharing brow.  The afternoon was in
     Mortimer Spanish desires. I understood that we shall show you to
the Franco-Midland
     Hill."

     "There is."

     "It was I have no means of geores buttons upon a new base, and h
e
     was regomsed for a money and took a
     stream of fierce-eyed, wicked chindering from as to their
     business. It has been committed in the outside-shever. There's n
o one their own
     countries from a clothes does. It ran
     argument which brought was married. A lady was
     largely in an ordinary but, as the cloud of light in the doar, w
hich lead
     in
```

In [11]:

```
# Build RNN with basic RNN implementation
def build_RNN_model(vocab_size, embedding_dim, rnn_units, batch_size):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim, batch_inpu
t_shape=[batch_size, None]),
        tf.keras.layers.SimpleRNN(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),
        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [12]:

```
model3 = build_RNN_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=BATCH_SIZE)

model3.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding (Embedding)        (64, None, 256)           24832     
_________________________________________________________________
simple_rnn (SimpleRNN)       (64, None, 1024)          1311744   
_________________________________________________________________
dense (Dense)                (64, None, 97)            99425     
=================================================================
Total params: 1,436,001
Trainable params: 1,436,001
Non-trainable params: 0
_________________________________________________________________
```

In [13]:

```
model3.compile(optimizer='adam', loss=loss)
model3.fit(dataset, epochs=15)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 39s 76ms/step - loss: 2.36
00
Epoch 2/15
522/522 [==============================] - 38s 73ms/step - loss: 1.58
49
Epoch 3/15
522/522 [==============================] - 38s 73ms/step - loss: 1.40
32
Epoch 4/15
522/522 [==============================] - 38s 73ms/step - loss: 1.31
86
Epoch 5/15
522/522 [==============================] - 38s 72ms/step - loss: 1.26
87
Epoch 6/15
522/522 [==============================] - 38s 72ms/step - loss: 1.23
63
Epoch 7/15
522/522 [==============================] - 38s 72ms/step - loss: 1.21
28
Epoch 8/15
522/522 [==============================] - 38s 72ms/step - loss: 1.19
40
Epoch 9/15
522/522 [==============================] - 38s 72ms/step - loss: 1.17
88
Epoch 10/15
522/522 [==============================] - 38s 72ms/step - loss: 1.16
59
Epoch 11/15
522/522 [==============================] - 38s 72ms/step - loss: 1.15
56
Epoch 12/15
522/522 [==============================] - 38s 72ms/step - loss: 1.14
60
Epoch 13/15
522/522 [==============================] - 38s 72ms/step - loss: 1.13
79
Epoch 14/15
522/522 [==============================] - 38s 73ms/step - loss: 1.13
07
Epoch 15/15
522/522 [==============================] - 38s 72ms/step - loss: 1.12
44
```

Out[13]:

```
<tensorflow.python.keras.callbacks.History at 0x14746279388>
```

In [14]:

```
model3.save_weights('./training_checkpoints/RNN_checkpoint')
```

In [15]:

```
model3 = build_RNN_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=1)
model3.load_weights('./training_checkpoints/RNN_checkpoint')
model3.build(tf.TensorShape([1, None]))
```

In [18]:

```
print(generate_text(model3, start_string='There '))
```

```
There LNd, I want of his pocket, I pessired bygeal.

     "A very solution."

     "In the colligants, and maid doubt the faces as it was there and
something was evidently
     had a most serious rest up, and
     the weight would have had py
     round with a very right had ruch for his mind.

     Holmes stopped at our dark man's hand. A confession of the relac
itions. I must assure no means, who had first might trie, there was a
lossly barrates for with the door and shutters below any step had reg
ret him should, had occasional
     twission to
     he live up to me that
     they are brows most look in with convally last
     things are in his eyes e might imagines we will seek-heavy stagg
erings and outside, house, as business may
     believe to say that is the paper answered. ' and was
     wrong. They saw that we are dealingshe
     object to acced is too well within the room which seemed to me t
hat I am a companion of Agent Prower McMurdo, but I shall suddenly st
aring up as his foolsca
```

In [52]:

```
# Build RNN with both GRU and LSTM
def build_LSTM_GRU_model(vocab_size, embedding_dim, rnn_units, batch_si
ze):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim, batch_inpu
t_shape=[batch_size, None]),
        tf.keras.layers.LSTM(rnn_units,
                             return_sequences=True,
                             stateful=True,
                             recurrent_initializer='glorot_uniform'),
        tf.keras.layers.GRU(rnn_units,
                            return_sequences=True,
                            stateful=True,
                            recurrent_initializer='glorot_uniform'),
        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [60]:

```
model4 = build_LSTM_GRU_model(vocab_size=len(unique_chars),
                              embedding_dim=embedding_dim,
                              rnn_units=rnn_units,
                              batch_size=BATCH_SIZE)
model4.summary()
```

```
Model: "sequential_10"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_11 (Embedding)     (64, None, 256)           24832     
_________________________________________________________________
lstm_6 (LSTM)                (64, None, 1024)          5246976   
_________________________________________________________________
gru_2 (GRU)                  (64, None, 1024)          6297600   
_________________________________________________________________
dense_10 (Dense)             (64, None, 97)            99425     
=================================================================
Total params: 11,668,833
Trainable params: 11,668,833
Non-trainable params: 0
_________________________________________________________________
```

In [61]:
```
model4.compile(optimizer='adam', loss=loss)
model4.fit(dataset, epochs=EPOCHS)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 158s 302ms/step - loss: 2.
0433
Epoch 2/15
522/522 [==============================] - 157s 301ms/step - loss: 1.
3121
Epoch 3/15
522/522 [==============================] - 153s 294ms/step - loss: 1.
1957
Epoch 4/15
522/522 [==============================] - 153s 292ms/step - loss: 1.
1404
Epoch 5/15
522/522 [==============================] - 155s 296ms/step - loss: 1.
1017
Epoch 6/15
522/522 [==============================] - 158s 303ms/step - loss: 1.
0710
Epoch 7/15
522/522 [==============================] - 164s 314ms/step - loss: 1.
0423
Epoch 8/15
522/522 [==============================] - 163s 312ms/step - loss: 1.
0144
Epoch 9/15
522/522 [==============================] - 153s 294ms/step - loss: 0.
9877
Epoch 10/15
522/522 [==============================] - 162s 309ms/step - loss: 0.
9599
Epoch 11/15
522/522 [==============================] - 161s 309ms/step - loss: 0.
9334
Epoch 12/15
522/522 [==============================] - 155s 297ms/step - loss: 0.
9064
Epoch 13/15
522/522 [==============================] - 155s 297ms/step - loss: 0.
8799
Epoch 14/15
522/522 [==============================] - 156s 299ms/step - loss: 0.
8544
Epoch 15/15
522/522 [==============================] - 155s 296ms/step - loss: 0.
8304
```

Out[61]:
```
<tensorflow.python.keras.callbacks.History at 0x202c78cecc8>
```

In [62]:
```
model4.save_weights('./training_checkpoints/LSTM_GRU_checkpoint')
```

In [53]:

```
model4 = build_LSTM_GRU_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=1)
model4.load_weights('./training_checkpoints/LSTM_GRU_checkpoint')
model4.build(tf.TensorShape([1, None]))
```

In [54]:

```
print(generate_text(model4, start_string='There '))
```

```
There was no one upon him?"

     Holmes nodded applied "There, you name!" said Holmes, "I shall s
ee him even sooner
     there shared in the street. It loved means
     that it was to far me, and his eyes fell go to keep her it upon
the
     ground followed by the mentte who had met Mr. Barclay lumps no
     telegraph with it."

     "That you," said Holmes, smiling, answer in the exerces of the f
irm. Her
     sleep would stand through my head, pael face, and against them,
es
     upon it, in spite of its
     instrument.

     "Your own villainy," he white figure. "Can't you, if you wonder
to his ruin, and
     running through the squasters?"

     "Well, it is likely a little throughory at ten o'clock't.

     Their keys and looked sternly at her excited, heard, high injury
ty was a quick step now in it was
     covered with the Navy, invite that they were told of my father
     Barrymore of Napoleon, and his persuady has gone this idea. Noth
ing more showing than
     singular knill I
```

In [46]:

```
def build_bidirectional_model(vocab_size, embedding_dim, rnn_units, bat
ch_size):
    model = tf.keras.Sequential([
        tf.keras.layers.Embedding(vocab_size, embedding_dim, batch_inpu
t_shape=[batch_size, None]),
        tf.keras.layers.Bidirectional(tf.keras.layers.LSTM(rnn_units //
2, return_sequences=True)),
        tf.keras.layers.Dense(vocab_size)
    ])
    return model
```

In [47]:

```
model5 = build_bidirectional_model(vocab_size=len(unique_chars),
                              embedding_dim=embedding_dim,
                              rnn_units=rnn_units,
                              batch_size=BATCH_SIZE)
model5.summary()
```

```
Model: "sequential_12"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_11 (Embedding)     (64, None, 256)           24832     
_________________________________________________________________
bidirectional_16 (Bidirectio (64, None, 1024)          3149824   
_________________________________________________________________
dense_12 (Dense)             (64, None, 97)            99425     
=================================================================
Total params: 3,274,081
Trainable params: 3,274,081
Non-trainable params: 0
_________________________________________________________________
```

In [48]:

```
model5.compile(optimizer='adam', loss=loss)
model5.fit(dataset, epochs=15)
```

```
Train for 522 steps
Epoch 1/15
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.
iter
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.
beta_1
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.
beta_2
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.
decay
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer.
learning_rate
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-0.embeddings
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-2.kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-2.bias
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.forward_layer.cell.kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.forward_layer.cell.recurr
ent_kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.forward_layer.cell.bias
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.backward_layer.cell.kerne
l
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.backward_layer.cell.recur
rent_kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'm' for (root).layer_with_weights-1.backward_layer.cell.bias
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-0.embeddings
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-2.kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-2.bias
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.forward_layer.cell.kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.forward_layer.cell.recurr
ent_kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.forward_layer.cell.bias
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.backward_layer.cell.kerne
l
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.backward_layer.cell.recur
rent_kernel
WARNING:tensorflow:Unresolved object in checkpoint: (root).optimizer'
s state 'v' for (root).layer_with_weights-1.backward_layer.cell.bias
WARNING:tensorflow:A checkpoint was restored (e.g. tf.train.Checkpoin
t.restore or tf.keras.Model.load_weights) but not all checkpointed va
lues were used. See above for specific issues. Use expect_partial() o
n the load status object, e.g. tf.train.Checkpoint.restore(...).expec
t_partial(), to silence these warnings, or use assert_consumed() to m
ake the check explicit. See https://www.tensorflow.org/guide/checkpoi
nt#loading_mechanics for details.
522/522 [==============================] - 40s 77ms/step - loss: 0.48
97
Epoch 2/15
522/522 [==============================] - 37s 71ms/step - loss: 0.02
39
Epoch 3/15
522/522 [==============================] - 37s 71ms/step - loss: 0.02
11
Epoch 4/15
522/522 [==============================] - 37s 71ms/step - loss: 0.01
98
Epoch 5/15
522/522 [==============================] - 37s 71ms/step - loss: 0.01
87
Epoch 6/15
522/522 [==============================] - 38s 72ms/step - loss: 0.01
78
Epoch 7/15
522/522 [==============================] - 38s 73ms/step - loss: 0.01
70
Epoch 8/15
522/522 [==============================] - 38s 72ms/step - loss: 0.01
62
Epoch 9/15
522/522 [==============================] - 38s 72ms/step - loss: 0.01
54
Epoch 10/15
522/522 [==============================] - 38s 72ms/step - loss: 0.01
45
Epoch 11/15
522/522 [==============================] - 37s 72ms/step - loss: 0.01
35
Epoch 12/15
522/522 [==============================] - 37s 71ms/step - loss: 0.01
24
Epoch 13/15
522/522 [==============================] - 37s 71ms/step - loss: 0.01
11
Epoch 14/15
522/522 [==============================] - 38s 73ms/step - loss: 0.00
95
Epoch 15/15
522/522 [==============================] - 37s 71ms/step - loss: 0.00
78
```

```
Out[48]: <tensorflow.python.keras.callbacks.History at 0x11dd7b3d248>
```

```
In [49]: model5.save_weights('./training_checkpoints/bidirectional_checkpoint')
```

In [50]:
```
model5 = build_bidirectional_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=1)
model5.load_weights('./training_checkpoints/bidirectional_checkpoint')
model5.build(tf.TensorShape([1, None]))
```

In [51]:
```
print(generate_text(model5, start_string='There '))
```

```
There hexalopexanythadves?"
 tot vene livesumaringroxinemy'HAmond
 llene.
 rFPreneneme, menfedonenebes
 H fatonodenetha
 mivenemermey

 nexar--I Brivesananes cèinèSclvif Miny -d
 stesad benereöGeereve mensAnenCin tingy petoréAsus°à’üI fa winotesed
Pon'din bfined mesinYonen.
 anexadrenoxpare, I oborexarineme! Wmes?"I ugen Ohhexaloro-Clinsunea
mexarinelachestealenx½edI Yoryon, us.
 finncimalifid--cr`'Tresthevedrvenendzzzzzzzzzzzzzzzeneshancorend.
 agy, junearZVelinemes'm, Cagrex Agy t y udwabobes; sveved, Chiswheme
nesmady cudd.
 Y.
 ddy taloponellemed finerenobal½?½Dy I sishen'sos, Trabunefey hes'Ja
minere.
 *7lin ItenoneGonconanequncrineperinenennd mesmuneshe'Slaîtowad
 umery Heroushachelo,  necloryotoggreswime ey he! gringes manenedas,
fusony
 bin'Siny.
 Cine ocralinunshes'°estedodrinedenve
 gre-glalrengeney gineny masthwheaney y hesgrmerinw-finasathenoxc."Gi
nd.
 tar` menores semonanexaloruala cholexzenenenelalos!alinopunths; puss
tonedy Cinetenexalonenellle CDuned Mreblory, I unesPenedy umer
```

In [57]:
```
model6 = build_GRU_model(vocab_size=len(unique_chars),
                        embedding_dim=embedding_dim,
                        rnn_units=rnn_units,
                        batch_size=BATCH_SIZE)
model6.compile(optimizer='RMSprop', loss=loss)
model6.fit(dataset, epochs=15)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 39s 75ms/step - loss: 1.83
43
Epoch 2/15
522/522 [==============================] - 40s 77ms/step - loss: 1.29
01
Epoch 3/15
522/522 [==============================] - 41s 78ms/step - loss: 1.18
66
Epoch 4/15
522/522 [==============================] - 39s 74ms/step - loss: 1.13
48
Epoch 5/15
522/522 [==============================] - 38s 73ms/step - loss: 1.09
90
Epoch 6/15
522/522 [==============================] - 38s 74ms/step - loss: 1.07
04
Epoch 7/15
522/522 [==============================] - 39s 74ms/step - loss: 1.04
61
Epoch 8/15
522/522 [==============================] - 39s 74ms/step - loss: 1.02
48
Epoch 9/15
522/522 [==============================] - 38s 73ms/step - loss: 1.00
53
Epoch 10/15
522/522 [==============================] - 39s 74ms/step - loss: 0.98
78
Epoch 11/15
522/522 [==============================] - 39s 74ms/step - loss: 0.97
20
Epoch 12/15
522/522 [==============================] - 39s 74ms/step - loss: 0.95
76
Epoch 13/15
522/522 [==============================] - 41s 79ms/step - loss: 0.94
50
Epoch 14/15
522/522 [==============================] - 40s 76ms/step - loss: 0.93
32
Epoch 15/15
522/522 [==============================] - 41s 78ms/step - loss: 0.92
38
```

Out[57]:
```
<tensorflow.python.keras.callbacks.History at 0x11ddf2046c8>
```

```
In [58]: model6.save_weights('./training_checkpoints/GRU_RSM_checkpoint')
```

```
In [60]: model6 = build_GRU_model(vocab_size=len(unique_chars),
                             embedding_dim=embedding_dim,
                             rnn_units=rnn_units,
                             batch_size=1)
         model6.load_weights('./training_checkpoints/GRU_RSM_checkpoint')
         model6.build(tf.TensorShape([1, None]))
```

```
In [64]: print(generate_text(model6, start_string='There '))
```

```
There a cloud heavily
     canable. Even now--n the key as he pays."

     "Oh! I realized that my wife was about to realize of my advice,
Watson?"

     I loved all the views, staring otherwise beyond their solurn con
versation. They brought no
     common man had blown to the stair.

     "That will do," said I, "for we can then take them
     at least you are, and that there are time it is
     evident that his visit has been entirely drugged to this group.
     And then anno name I have the meaning of the facts of mankind ch
ief from the West End. Lawurnil
     maid shook his head with his flight."

     "And how did he hear a letter you think before. Of course this b
eautiful
     perhaps she has spoken to the spot mixty hobbling but not blowni
ng on to the
     Barrymores.

     He approached it, and a husband, coming down to conceal it, but
gathered through the
     whole of that last man this morning.

     "I may add that I had been the moat will find a new mornings at
     Baker Street
```

In [67]:

```
model7 = build_LSTM_model(vocab_size=len(unique_chars),
                          embedding_dim=embedding_dim,
                          rnn_units=rnn_units,
                          batch_size=BATCH_SIZE)

model7.compile(optimizer='RMSprop', loss=loss)
model7.fit(dataset, epochs=15)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 51s 99ms/step - loss: 1.96
93 3s - loss:
Epoch 2/15
522/522 [==============================] - 49s 94ms/step - loss: 1.33
99
Epoch 3/15
522/522 [==============================] - 49s 94ms/step - loss: 1.21
06
Epoch 4/15
522/522 [==============================] - 49s 94ms/step - loss: 1.14
90
Epoch 5/15
522/522 [==============================] - 49s 94ms/step - loss: 1.10
88 0s - loss: 1.1
Epoch 6/15
522/522 [==============================] - 49s 95ms/step - loss: 1.07
82
Epoch 7/15
522/522 [==============================] - 50s 95ms/step - loss: 1.05
18
Epoch 8/15
522/522 [==============================] - 49s 95ms/step - loss: 1.02
87
Epoch 9/15
522/522 [==============================] - 49s 94ms/step - loss: 1.00
71
Epoch 10/15
522/522 [==============================] - 50s 95ms/step - loss: 0.98
68 1s - lo - ETA: 0s - loss:
Epoch 11/15
522/522 [==============================] - 49s 94ms/step - loss: 0.96
78
Epoch 12/15
522/522 [==============================] - 49s 95ms/step - loss: 0.94
91
Epoch 13/15
522/522 [==============================] - 49s 94ms/step - loss: 0.93
16
Epoch 14/15
522/522 [==============================] - 49s 94ms/step - loss: 0.91
37
Epoch 15/15
522/522 [==============================] - 49s 95ms/step - loss: 0.89
75
```

```
Out[67]: <tensorflow.python.keras.callbacks.History at 0x11de35343c8>
```

```
In [68]: model7.save_weights('./training_checkpoints/LSTM_RMS_checkpoint')
```

In [69]:

```
model7 = build_LSTM_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=1)
model7.load_weights('./training_checkpoints/LSTM_RMS_checkpoint')
model7.build(tf.TensorShape([1, None]))
```

In [70]:

```
print(generate_text(model7, start_string='There '))
```

```
There came down to
     Rolding's six; but I was a sinking out a slip of paper and the
           Rylodes will be back?"

     "By thumb," he said. "But the lady is sure that you would. You
     slipped at a glovely interesting path, and y-- Young October sin
ce Dr. Mortimer told
     me hell upon so much as material, red before the treasure
     is gieaties. As he said, however, otherwise, on which was they
     glancing backwaze nor any elecwing girl. It is a stout fanging i
n front of
     yellow-shot, forbidden it is close to me with the auraction of a
     large idea that he did."

     "You may make it worth this security for?" asked Holmes.

     "Father--link above. She was told in one of his
     met. Even within each business must blame it itsed window of the
heavy brows,
     and begins to clear it up. With our friend
     could lie down without her, but indoors I can."

     "What else it is ubout war, for the ladisch door over the walls
     t it. And I
     think. We must in that
```

In [74]:

```
model8 = build_RNN_model(vocab_size=len(unique_chars),
                         embedding_dim=embedding_dim,
                         rnn_units=rnn_units,
                         batch_size=BATCH_SIZE)

model8.compile(optimizer='RMSprop', loss=loss)
model8.fit(dataset, epochs=15)
```

```
Train for 522 steps
Epoch 1/15
522/522 [==============================] - 39s 74ms/step - loss: 2.59
35
Epoch 2/15
522/522 [==============================] - 38s 72ms/step - loss: 1.71
88
Epoch 3/15
522/522 [==============================] - 37s 72ms/step - loss: 1.56
33
Epoch 4/15
522/522 [==============================] - 37s 71ms/step - loss: 1.48
54
Epoch 5/15
522/522 [==============================] - 38s 72ms/step - loss: 1.43
75
Epoch 6/15
522/522 [==============================] - 37s 72ms/step - loss: 1.40
37
Epoch 7/15
522/522 [==============================] - 37s 72ms/step - loss: 1.37
88
Epoch 8/15
522/522 [==============================] - 37s 72ms/step - loss: 1.35
90
Epoch 9/15
522/522 [==============================] - 38s 72ms/step - loss: 1.34
39
Epoch 10/15
522/522 [==============================] - 37s 71ms/step - loss: 1.33
06
Epoch 11/15
522/522 [==============================] - 37s 72ms/step - loss: 1.31
90
Epoch 12/15
522/522 [==============================] - 38s 72ms/step - loss: 1.31
01
Epoch 13/15
522/522 [==============================] - 38s 72ms/step - loss: 1.30
13
Epoch 14/15
522/522 [==============================] - 38s 72ms/step - loss: 1.29
35
Epoch 15/15
522/522 [==============================] - 37s 71ms/step - loss: 1.28
76
```

```
Out[74]: <tensorflow.python.keras.callbacks.History at 0x11dc748ee88>
```

```
In [75]: model8.save_weights('./training_checkpoints/RNN_RMS_checkpoint')
```

In [76]:

```
model8 = build_RNN_model(vocab_size=len(unique_chars),
                    embedding_dim=embedding_dim,
                    rnn_units=rnn_units,
                    batch_size=1)
model8.load_weights('./training_checkpoints/RNN_RMS_checkpoint')
model8.build(tf.TensorShape([1, None]))
```

In [78]:

```
print(generate_text(model8, start_string='There '))
```

```
There all
     to but to be disguiseous aspackmissist
     know. The
     window lodge play she causence that one passive."

     "You will would be dark I can Alvarourous oftei. But they took a
ll you, Sir Claimanion. That's
     look between from his
     atmospage of
     the end of them against his hury told your confided job-gover ab
out the
     long prescinger which can she was enter off and dumbled
     of Afficeales to have you, Souses, and also question had never h
eard I've back.

     "I know that he had take tracked upon the
     last quiting. On the
     vague deal him the orders and soul-wream of this!" Sand-house in
spector to started.

     "That is you, Formall
     down gales sinctions should have no derately up the
     discover the will launch
     in the man
     admital charactering that
     det in an accues marious satually come. I think that he sporting
his friendly known house. His own
     dars and my alsoftened at, well, and
     you will trusting as not with the ins
```

In [ ]: