##Libro utilizado: The Great Gatsby by F. Scott Fitzgerald

Ernesto Reynoso Lizárraga A01639915

Libro utilizado: The Great Gatsby by F. Scott Fitzgerald

Ernesto Reynoso Lizárraga A01639915

```python
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
import os
```

Descarga de datos

```python
path_to_fileDL = tf.keras.utils.get_file('The Great Gatsby by F. Scott Fitzgerald.txt', 'https://gutenberg.org/cache/epub/64317/pg64317.txt')

text = open(path_to_fileDL, 'rb').read().decode(encoding='utf-8')
print('Longitud del texto:    {} caracteres'.format(len(text)))

vocab = sorted(set(text))
print ('El texto esta compuesto de estos {} caracteres'.format(len(vocab)))
print (vocab)
```

```
Downloading data from https://gutenberg.org/cache/epub/64317/pg64317.txt
306317/306317 [==============================] - 0s 0us/step
Longitud del texto:    296579 caracteres
El texto esta compuesto de estos 96 caracteres
['\t', '\n', '\r', ' ', '!', '#', '$', '%', '(', ')', '*', ',', '-', '.', '/', '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', ':', ';', '?', 'A', 'B'
```

Tablas de traduccion o Inversa de vocabulario

```python
char2idx = {u:i for i, u in enumerate(vocab)}
idx2char = np.array(vocab)

for char,_ in zip(char2idx, range(len(vocab))):
  print(' {:4s}: {:3d},'.format(repr(char), char2idx[char]))
```

```
    'K' :  38,
    'L' :  39,
    'M' :  40,
    'N' :  41,
    'O' :  42,
    'P' :  43,
    'Q' :  44,
    'R' :  45,
    'S' :  46,
    'T' :  47,
    'U' :  48,
    'V' :  49,
    'W' :  50,
    'X' :  51,
    'Y' :  52,
    'Z' :  53,
    '[' :  54,
    ']' :  55,
    'a' :  56,
    'b' :  57,
    'c' :  58,
    'd' :  59,
    'e' :  60,
    'f' :  61,
    'g' :  62,
    'h' :  63,
    'i' :  64,
    'j' :  65,
    'k' :  66,
    'l' :  67,
    'm' :  68,
    'n' :  69,
    'o' :  70,
    'p' :  71,
    'q' :  72,
    'r' :  73,
    's' :  74,
    't' :  75,
    'u' :  76,
    'v' :  77,
    'w' :  78,
    'x' :  79,
    'y' :  80,
    'z' :  81,
    'ç' :  82,
    'é' :  83,
    'è' :  84,
    'ô' :  85,
    '\u200a':  86,
    '—' :  87,
    '‘' :  88,
    '’' :  89,
    '“' :  90,
    '”' :  91,
    '•' :  92,
    '…' :  93,
    '™' :  94,
    '\ufeff':  95,
```

convertir texto a enteros

```python
text_as_int = np.array([char2idx[c] for c in text])

#Mostramos algunos caracteres
print('text: {}'.format(repr(text[:50])))
print('{}'.format(repr(text_as_int[:50])))
```

```
    text: '\ufeffThe Project Gutenberg eBook of The Great Gatsby\r\n'
    array([95, 47, 63, 60,  3, 43, 73, 70, 65, 60, 58, 75,  3, 34, 76, 75, 60,
```

```
69, 57, 60, 73, 62,  3, 60, 29, 70, 70, 66,  3, 70, 61,  3, 47, 63,
60,  3, 34, 73, 60, 56, 75,  3, 34, 56, 75, 74, 57, 80,  2,  1])
```

PREPARAR DATOS

```python
char_dataset = tf.data.Dataset.from_tensor_slices(text_as_int)

seq_length = 100

sequences = char_dataset.batch(seq_length+1, drop_remainder=True)


#comprobar datos
for item in sequences.take(10):
  print(repr(''.join(idx2char[item.numpy()])))
```

```
'\ufeffThe Project Gutenberg eBook of The Great Gatsby\r\n    \r\nThis ebook is for the use of anyone anywhere '
'in the United States and\r\nmost other parts of the world at no cost and with almost no restrictions\r\nw'
'hatsoever. You may copy it, give it away or re-use it under the terms\r\nof the Project Gutenberg Licen'
'se included with this ebook or online\r\nat www.gutenberg.org. If you are not located in the United Sta'
'tes,\r\nyou will have to check the laws of the country where you are located\r\nbefore using this eBook.\r'
'\n\r\nTitle: The Great Gatsby\r\n\r\n\r\n\r\nAuthor: F. Scott Fitzgerald\r\n\r\n\r\nRelease date: January 17, 2021 [eBook '
'#64317]\r\n\r\nLanguage: English\r\n\r\n\r\n\r\n\r\n*** START OF THE PROJECT GUTENBERG EBOOK THE GREAT GATSBY ***\r\n\r\n\r\n'
'\t\t\t    The Great Gatsby\r\n\t\t\t  by\r\n\t\t\t F. Scott Fitzgerald\r\n\r\n\r\n\n                          Table of Co'
'ntents\r\n\r\n\r\nI\r\nII\r\nIII\r\nIV\r\nV\r\nVI\r\nVII\r\nVIII\r\nIX\r\n\r\n\r\n\n                       Once again\r\n           '
'                    to\r\n                           Zelda\r\n\r\n  Then wear the gold hat, if'
```

```python
#Preparar datos de entrenamiento  (Entrada 0 a 99 )  (Salida 1 a 100)
def split_input_target(chunk):
  input_text = chunk[:-1]
  target_text = chunk[1:]
  return input_text, target_text

dataset = sequences.map(split_input_target)


#Visualizamos
for input_example, target_example in dataset.take(1):
  print ('Input data: ', repr(''.join(idx2char[input_example.numpy()])))
  print ('Target data: ', repr(''.join(idx2char[target_example.numpy()])))
```

```
Input data:  '\ufeffThe Project Gutenberg eBook of The Great Gatsby\r\n    \r\nThis ebook is for the use of anyone anywhere'
Target data:  'The Project Gutenberg eBook of The Great Gatsby\r\n    \r\nThis ebook is for the use of anyone anywhere '
```

```python
#imprimir dataset
print (dataset)
```

```
<_MapDataset element_spec=(TensorSpec(shape=(100,), dtype=tf.int64, name=None), TensorSpec(shape=(100,), dtype=tf.int64, name=None))>
```

```python
#agrupar en batches
BATCH_SIZE = 64
BUFFER_SIZE = 10000

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

```
<_BatchDataset element_spec=(TensorSpec(shape=(64, 100), dtype=tf.int64, name=None), TensorSpec(shape=(64, 100), dtype=tf.int64, name=None))>
```

Construir modelo RNN

```python
def build_model(vocab_size, embedding_dim, rnn_units, batch_size):
  model = tf.keras.Sequential([
      tf.keras.layers.Embedding(vocab_size, embedding_dim,
                          batch_input_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

vocab_size = len(vocab)
embedding_dim= 256
rnn_units = 1024

model = build_model(
    vocab_size = vocab_size,
    embedding_dim=embedding_dim,
    rnn_units=rnn_units,
    batch_size = BATCH_SIZE
)

#Visualizar estructura
model.summary()
```

```
Model: "sequential"
_____
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (64, None, 256)           24576

 lstm (LSTM)                 (64, None, 1024)          5246976

 dense (Dense)               (64, None, 96)            98400

=================================================================
Total params: 5369952 (20.48 MB)
Trainable params: 5369952 (20.48 MB)
Non-trainable params: 0 (0.00 Byte)
_____
```

```
# Forma de input
for input_example_batch, target_example_batch in dataset.take(1):
  print("Input: ", input_example_batch.shape, "# (batch_size, lenght)")
  print("Target: ", target_example_batch.shape, "# (batch_size, sequence_length)")
```

```
    Input:  (64, 100) # (batch_size, lenght)
    Target:  (64, 100) # (batch_size, sequence_length)
```

```
#Forma de salida
for input_example_batch, target_example_batch in dataset.take(1):
  example_batch_predictions = model(input_example_batch)
  print("Prediction: ", example_batch_predictions.shape, "# (batch_size, sequence_length, vocab_size)")
```

```
    Prediction:  (64, 100, 96) # (batch_size, sequence_length, vocab_size)
```

```
#Mostrar que el resultado es una distribucion, no un argmax
```

```
sampled_indices = tf.random.categorical(example_batch_predictions[0], num_samples=1)
sampled_indices_characters = tf.squeeze(sampled_indices,axis=-1).numpy()
print(sampled_indices_characters)
```

```
    [80 27 90 74 91 58 91 27 68 45 59  5 20 83 63 83 21 79 80 93 68 84 69 50
     43 10  0 56 38 75 58 19 58 63 32 49 29 52  9 66 63 77 95 70 72 85 47 51
     29 58 49  6 20 91 61 34 61  7 70 17 89 43 84  1 46 51 68 78 10 81 46 71
     43 60 66 15 23 29 55 74 14 61 44 48 24  2  5  1 17 70  3 26 72 20 64 67
     60 45 29 64]
```

ENTRENAMIENTO

```
def loss(labels, logits):
  return tf.keras.losses.sparse_categorical_crossentropy(labels, logits, from_logits=True)
```

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

```
checkpoint_dir = './training_checkpoints'
checkpoint_prefix = os.path.join(checkpoint_dir, "ckpt_(epoch)")
```

```
checkpoint_callback = tf.keras.callbacks.ModelCheckpoint(
    filepath=checkpoint_prefix,
    save_weights_only=True
)
```

```
EPOCHS = 50
```

```
history = model.fit(dataset, epochs=EPOCHS, callbacks=[checkpoint_callback])
```

```
    Epoch 22/50
    45/45 [==============================] - 4s 78ms/step - loss: 1.0749
    Epoch 23/50
    45/45 [==============================] - 4s 79ms/step - loss: 1.0329
    Epoch 24/50
    45/45 [==============================] - 4s 79ms/step - loss: 0.9919
    Epoch 25/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.9502
    Epoch 26/50
    45/45 [==============================] - 5s 81ms/step - loss: 0.9033
    Epoch 27/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.8597
    Epoch 28/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.8167
    Epoch 29/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.7743
    Epoch 30/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.7288
    Epoch 31/50
    45/45 [==============================] - 4s 83ms/step - loss: 0.6822
    Epoch 32/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.6466
    Epoch 33/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.6069
    Epoch 34/50
    45/45 [==============================] - 4s 82ms/step - loss: 0.5688
    Epoch 35/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.5344
    Epoch 36/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.5031
    Epoch 37/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.4755
    Epoch 38/50
    45/45 [==============================] - 4s 79ms/step - loss: 0.4518
    Epoch 39/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.4288
    Epoch 40/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.4038
    Epoch 41/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.3868
    Epoch 42/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.3728
    Epoch 43/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.3566
    Epoch 44/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.3463
    Epoch 45/50
    45/45 [==============================] - 4s 81ms/step - loss: 0.3339
    Epoch 46/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.3253
    Epoch 47/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.3164
    Epoch 48/50
    45/45 [==============================] - 5s 80ms/step - loss: 0.3068
    Epoch 49/50
    45/45 [==============================] - 4s 80ms/step - loss: 0.2996
    Epoch 50/50
    45/45 [==============================] - 4s 82ms/step - loss: 0.2912
```

Generacion de texto

```
model = build_model(vocab_size, embedding_dim, rnn_units, batch_size=1)
```

```
model.load_weights(tf.train.latest_checkpoint(checkpoint_dir))

model.build(tf.TensorShape([1,None]))
```

```
WARNING:tensorflow:Detecting that an object or model or tf.train.Checkpoint is being deleted with unrestored values. See the following logs for the spec
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._iterations
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._learning_rate
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.1
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.2
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.3
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.4
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.5
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.6
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.7
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.8
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.9
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.10
WARNING:tensorflow:Value in checkpoint could not be found in the restored object: (root).optimizer._variables.11
WARNING:tensorflow:Value sain checkpoint could not be found in the restored object: (root).optimizer._variables.12
```

```python
def generate_text(model, start_string, temp):
  num_generate = 500
  input_eval = [char2idx[s] for s in start_string]

  input_eval = tf.expand_dims(input_eval, 0)
  text_generated = []

  temperature = temp

  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)

    text_generated.append(idx2char[predicted_id])

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

```python
print(generate_text(model, start_string=u"Well", temp = 0.5))
```

```
Welll never his nose. "I'm start about from the
dark road one person and his chin in his hand.

"I'm not?" she announced as he saw that his front door was still on the
there to faded girls whom we were nears to this
afternoon. I hadn't been there before. Or perhaps in his hands had
seemed relieved to find that the two minutes that pretty soon, and then there's always the
funndless drive boy. The funchel scarcely created grass. A new world, many moved against a mile, so as
to shrink away from
```

```python
print(generate_text(model, start_string=u"Well", temp = 0.2))
```

```
Welllver' she said he knew you. He said he was raised in Louisville. Asa
Bird brought him around at the last thing that many bend
carrying light capes over their arms.

"Shall we all go in my car?" suggested Gatsby. He felt the basic fingers over his eyes and looking at him with
unfathomable delight. It was touching to see them?" demanded Tom harshly.

"I just got wised up to something funny the last two days," remarked
Wilson stare. When she mat be slowly until we were beyond the
bricag fo
```

```python
print(generate_text(model, start_string=u"Well", temp = 0.8))
```

```
Welll she said "Sh!" and week at Daisy, who was sytairs to the cestaurant, and Michaelis tent out in the half-highed back
have saying. "Why's that?"

"Daisy invited him; she had grassed reluctantly
away. It seemed to me that he was not alone—and I was alone for me. You've got to get a body were
all I among through the high in from the
beg on a chair.

"We've got to be an embarrassed, he saw him contoffull sable, tracing, of
the sun, which, as it drive me some again in a flurres of sometime,
```

```python
print(generate_text(model, start_string=u"sadly", temp = 0.5))
```

```
sadly.

"What go?" I assered her. "It's another man."

"No?" in a bottle of whisky from a locked
bureau door.

I have been drunk just twice in my life, and the second could lit for my table. That was my fault—Gatsby had
been called to the phone, and I'd enjoyed these same people over here and there was a face I had noticed
on the commuting train. I was immediately struck by the silence with the soft twilight,
but each time I tried to go I became entangled in some wild, string and familiar
```

```python
print(generate_text(model, start_string=u"sadly", temp = 0.2))
```

```
sadly.

"What do I owe you in casious n the secret of this accisingly. "What do you think we
stopped for—to admire the view?"

"I'm sick," he said down his chair.

"Look here, old sport," said Gatsby, leaning toward me, "I have to leave you here."

"No you don't understand," explained the criminal. "I wasn't even
trying."

An awoke for me to see. On
```

```
        the last flyleaf was printed the word schedule, and the date September
        12, 1.2.  believe it was something funny.”

        “You mean your wife bo
```

```
print(generate_text(model, start_string=u"sadly", temp =1.5))
```

```
        sadly abswervehered lapped a sense
        Forward’s-by EF)
                Precels Plutence (o7” Toh, and Daisy gratteders, remanded meugh to get home.”

        “Lut’s a look!” she broke out menby, “Mertly, (butures womancing?”
        demanifor 19j.353h’Sk appearance. me atticial glagolys
        jusin the inunspaintance GaBody each other. Hear ear hapsbanks and unkeriph 1.E8:1 (7!15, 8906”      Satur:

        “We bot Westfunnd. Mach because squake her hair, and a gus ouvenees—and after all
        onceived n a visiot:

        It was afterwarm thrat
```