```
# -*- coding: utf-8 -*-
"""Assignment 4.ipynb

Automatically generated by Colaboratory.

Original file is located at
    https://colab.research.google.com/drive/1-
opAkGkf3ZzX1BSkv2BuhcPMazCdVzKI
"""

import tensorflow as tf
from tensorflow.keras.datasets import imdb
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, SimpleRNN, Dense,
Bidirectional
from tensorflow.keras.preprocessing import sequence

# Load the IMDB dataset
max_features = 10000  # Consider only the top 10,000 words
maxlen = 150  # Cutoff reviews after 150 words
batch_size = 32

print('Loading data...')
(x_train, y_train), (x_test, y_test) =
imdb.load_data(num_words=max_features)
print(len(x_train), 'train sequences')
print(len(x_test), 'test sequences')

# Restrict training samples to 100
x_train = x_train[:100]
y_train = y_train[:100]

# Pad sequences to the same length
x_train = sequence.pad_sequences(x_train, maxlen=maxlen)
x_test = sequence.pad_sequences(x_test, maxlen=maxlen)

# Validate on 10,000 samples
x_val = x_test[:10000]
y_val = y_test[:10000]
x_test = x_test[10000:]
y_test = y_test[10000:]

# Create the RNN model
model = Sequential()

# Add an embedding layer
model.add(Embedding(max_features, 32))  # Embedding layer with 32-
dimensional embeddings

# Add a bidirectional SimpleRNN layer
model.add(Bidirectional(SimpleRNN(32)))

# Add a dense output layer
model.add(Dense(1, activation='sigmoid'))
```

```
# Compile the model
model.compile(optimizer='rmsprop', loss='binary_crossentropy',
metrics=['accuracy'])

# Print the model summary
model.summary()

# Train the model
history = model.fit(x_train, y_train, epochs=10, batch_size=batch_size,
validation_data=(x_val, y_val))

# Evaluate the model on test data
score, acc = model.evaluate(x_test, y_test, batch_size=batch_size)
print('Test score:', score)
print('Test accuracy:', acc)

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense, Bidirectional
from tensorflow.keras.datasets import imdb
from tensorflow.keras.preprocessing.sequence import pad_sequences

max_features = 10000  # Maximum number of words to keep based on word
frequency
max_len = 150  # Maximum length of reviews (in words) to consider

# Load the IMDB dataset
(X_train, y_train), (X_test, y_test) =
imdb.load_data(num_words=max_features)

# Cut reviews after max_len words
X_train = pad_sequences(X_train, maxlen=max_len)
X_test = pad_sequences(X_test, maxlen=max_len)

# Create the model
model = Sequential()

# Add an Embedding layer
model.add(Embedding(max_features, 128))  # Use 128-dimensional embedding
vectors

# Add a Bidirectional LSTM layer
model.add(Bidirectional(LSTM(64)))  # Use 64 units in the LSTM layer

# Add a Dense output layer
model.add(Dense(1, activation='sigmoid'))

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

# Train the model
model.fit(X_train, y_train, batch_size=32, epochs=5,
validation_data=(X_test, y_test))
```

```
# Evaluate the model
scores = model.evaluate(X_test, y_test, batch_size=32)
print('Test loss:', scores[0])
print('Test accuracy:', scores[1])
```