In [51]:
```
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import LSTM, Dense
```

In [52]:
```
model = Sequential([
    LSTM(32, input_shape=(10,1)),
    Dense(1,activation = 'sigmoid')
])
```

In [53]:
```
model.compile(optimizer = 'adam',loss = 'binary_crossentropy',
              metrics = ['accuracy'])
```

In [54]:
```
import numpy as np
X = np.random.rand(100, 10 , 1)
y = np.random.randint(0,2,(100,1))
```

In [55]:
```
model.fit(X, y, epochs = 10, batch_size = 32)
```

```
Epoch 1/10
4/4 [==============================] - 5s 16ms/step - loss: 0.6900 - accuracy: 0.5400
Epoch 2/10
4/4 [==============================] - 0s 13ms/step - loss: 0.6884 - accuracy: 0.5400
Epoch 3/10
4/4 [==============================] - 0s 13ms/step - loss: 0.6889 - accuracy: 0.5400
Epoch 4/10
4/4 [==============================] - 0s 13ms/step - loss: 0.6891 - accuracy: 0.5400
Epoch 5/10
4/4 [==============================] - 0s 14ms/step - loss: 0.6890 - accuracy: 0.5400
Epoch 6/10
4/4 [==============================] - 0s 14ms/step - loss: 0.6895 - accuracy: 0.5400
Epoch 7/10
4/4 [==============================] - 0s 14ms/step - loss: 0.6888 - accuracy: 0.5400
Epoch 8/10
4/4 [==============================] - 0s 15ms/step - loss: 0.6888 - accuracy: 0.5400
Epoch 9/10
4/4 [==============================] - 0s 12ms/step - loss: 0.6885 - accuracy: 0.5400
Epoch 10/10
4/4 [==============================] - 0s 13ms/step - loss: 0.6884 - accuracy: 0.5400
```

Out[55]:
```
<keras.src.callbacks.History at 0x22f4e52a140>
```

In [56]:
```
loss, accuracy = model.evaluate(X,y)
print(f'Test loss: {loss}, Test accuracy: {accuracy}')
```

```
4/4 [==============================] - 1s 10ms/step - loss: 0.6883 - accuracy: 0.5400
Test loss: 0.6882514357566833, Test accuracy: 0.5400000214576721
```

In [57]:
```
# EX 2
```

In [64]:
```
import tensorflow as tf
from tensorflow.keras.datasets import imdb
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Embedding, LSTM, Dense

# Load and preprocess the IMDB dataset
max_features = 20000  # Number of words to consider as features
maxlen = 100  # Cut texts after this number of words (among top max_features most common 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')

print('Pad sequences (samples x time)')
x_train = pad_sequences(x_train, maxlen=maxlen)
x_test = pad_sequences(x_test, maxlen=maxlen)
print('x_train shape:', x_train.shape)
print('x_test shape:', x_test.shape)

# Build the model
model = Sequential()
model.add(Embedding(max_features, 128, input_length=maxlen))
model.add(LSTM(64, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation='sigmoid'))

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

# Train the model
print('Train...')
```

```
model.fit(x_train, y_train,
          batch_size=batch_size,
          epochs=5,
          validation_data=(x_test, y_test))

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

```
Loading data...
Downloading data from https://storage.googleapis.com/tensorflow/tf-keras-datasets/imdb.npz
17464789/17464789 [==============================] - 344s 20us/step
25000 train sequences
25000 test sequences
Pad sequences (samples x time)
x_train shape: (25000, 100)
x_test shape: (25000, 100)
Train...
Epoch 1/5
782/782 [==============================] - 172s 208ms/step - loss: 0.4075 - accuracy: 0.8132 - val_loss: 0.3347 - val_accuracy:
0.8558
Epoch 2/5
782/782 [==============================] - 165s 211ms/step - loss: 0.2332 - accuracy: 0.9080 - val_loss: 0.3774 - val_accuracy:
0.8446
Epoch 3/5
782/782 [==============================] - 163s 208ms/step - loss: 0.1500 - accuracy: 0.9441 - val_loss: 0.4092 - val_accuracy:
0.8392
Epoch 4/5
782/782 [==============================] - 163s 209ms/step - loss: 0.1052 - accuracy: 0.9620 - val_loss: 0.4958 - val_accuracy:
0.8306
Epoch 5/5
782/782 [==============================] - 164s 209ms/step - loss: 0.0752 - accuracy: 0.9735 - val_loss: 0.6073 - val_accuracy:
0.8318
782/782 [==============================] - 22s 28ms/step - loss: 0.6073 - accuracy: 0.8318
Test score: 0.6072919368743896
Test accuracy: 0.8317599892616272
```