Análisis de Datos y Aprendizaje Máquina con Tensorflow 2.0: Redes neuronales recurrentes

2019/09/30

RNN - Clasificación de Texto

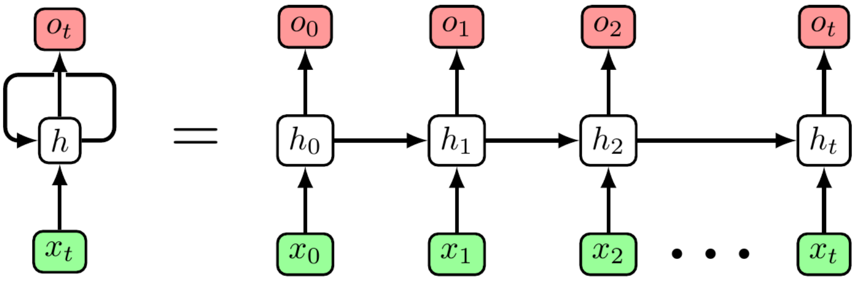
- Objetivo: Implementar RNN y LSTM para la clasificación de secuencias (texto). Aplicar inicialización de pesos y dropout en modelos recurrentes.
- Conocer efectos de inicialización y dropout en RNN y comparar con MLP

Redes Neuronales Recurrentes

- Cada capa en cada iteración comparte parámetros
- 'sparse_categorical_crossentropy' se utiliza para varias clases

Leer Dataset

```
In [1]: from tensorflow.keras.models import Model
        from tensorflow.keras.layers import LSTM, Embedding, Dense, SimpleRNN
        from tensorflow.keras.datasets import reuters
        from tensorflow.keras.models import Sequential
        from tensorflow import keras
        import numpy as np
        from tensorflow.keras.preprocessing.sequence import pad_sequences
        import matplotlib.pyplot as plt
```



RNN

11,228 noticias de Reuters, con más de 46 temas.

```
In [2]: # parámetros
        emb_dim = 64
        num_words = 8000
        max_words = 40
```

```
In [3]: (x_train, y_train), (x_test, y_test) = reuters.load_data(num_words = num_words, maxlen=ma
```

```
In [4]: x_train.shape
```

```
Out[4]: (1084,)
```

```
In [5]: print('Noticia')
        print(x_train[0])
        print('Etiqueta')
        print(y_train[0])
```

```
Noticia
[1, 245, 273, 207, 156, 53, 74, 160, 26, 14, 46, 296, 26, 39, 74, 2979, 3554, 14, 46, 4689, 4329,
Etiqueta
3
```

Palabras de noticia

```
In [6]: wordDict = {y:x for x,y in reuters.get_word_index().items()}
        res = []
        for index in x_train[0]:
            res.append(wordDict.get(index - 3))
        print('Noticia: ',res,'Longitud noticia: ', len(res))
```

```
Noticia:  [None, 'period', 'ended', 'december', '31', 'shr', 'profit', '11', 'cts', 'vs', 'loss',
```

```
In [7]: x_train = pad_sequences(x_train, maxlen=max_words, padding = 'post')
        x_test = pad_sequences(x_test, maxlen=max_words, padding = 'post')
```

```
In [8]: epoch = 40
        verbose = 1
        batch = 50
```

```
In [9]: print(x_train.shape)
        print(x_test.shape)
        print(y_train.shape)
        print(y_test.shape)
```

```
(1084, 40)
(272, 40)
(1084,)
(272,)
```

```
In [10]: x_train[0]
```

```
Out[10]: array([   1,  245,  273,  207,  156,   53,   74,  160,   26,   14,   46,
                 296,   26,   39,   74, 2979, 3554,   14,   46, 4689, 4329,   86,
                  61, 3499, 4795,   14,   61,  451, 4329,   17,   12,    0,    0,
                   0,    0,    0,    0,    0,    0,    0], dtype=int32)
```

Deep RNN

- Cuando se conectan varias capas de RNNs se modifica el parámetro 'return_sequences'
- Se inicializan los pesos con 'glorot_uniform'

```
In [11]: def deep_rnn():
             model = Sequential()
             model.add(Embedding(num_words, emb_dim))
             model.add(SimpleRNN(32, return_sequences = True, recurrent_initializer='glorot_unifo
             model.add(SimpleRNN(32, return_sequences = False, recurrent_initializer='glorot_unif
             model.add(Dense(46, activation = 'softmax'))

             model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['ac

             return model
```

```
In [12]: model = deep_rnn()
         model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding (Embedding)        (None, None, 64)          512000
_________________________________________________________________
simple_rnn (SimpleRNN)       (None, None, 32)          3104
_________________________________________________________________
simple_rnn_1 (SimpleRNN)     (None, 32)                2080
_________________________________________________________________
dense (Dense)                (None, 46)                1518
=================================================================
Total params: 518,702
Trainable params: 518,702
Non-trainable params: 0
_________________________________________________________________
```

```
In [13]: history = model.fit(x_train, y_train, batch_size = batch, validation_split = 0.3,
                             epochs = epoch, verbose = verbose)
```

```
Train on 758 samples, validate on 326 samples
Epoch 1/40
758/758 [==============================] - 2s 3ms/sample - loss: 2.7716 - accuracy: 0.6240 - val_
Epoch 2/40
```

```
758/758 [==============================] - 1s 886us/sample - loss: 1.5105 - accuracy: 0.7718 - va
Epoch 3/40
758/758 [==============================] - 1s 998us/sample - loss: 1.1607 - accuracy: 0.7718 - va
Epoch 4/40
758/758 [==============================] - 1s 937us/sample - loss: 1.0992 - accuracy: 0.7718 - va
Epoch 5/40
758/758 [==============================] - 1s 1ms/sample - loss: 1.0718 - accuracy: 0.7718 - val_
Epoch 6/40
758/758 [==============================] - 1s 824us/sample - loss: 1.0595 - accuracy: 0.7718 - va
Epoch 7/40
758/758 [==============================] - 1s 922us/sample - loss: 1.0503 - accuracy: 0.7718 - va
Epoch 8/40
758/758 [==============================] - 1s 877us/sample - loss: 1.0444 - accuracy: 0.7718 - va
Epoch 9/40
758/758 [==============================] - 1s 855us/sample - loss: 1.0403 - accuracy: 0.7718 - va
Epoch 10/40
758/758 [==============================] - 1s 870us/sample - loss: 1.0373 - accuracy: 0.7718 - va
Epoch 11/40
758/758 [==============================] - 1s 870us/sample - loss: 1.0345 - accuracy: 0.7718 - va
Epoch 12/40
758/758 [==============================] - 1s 890us/sample - loss: 1.0320 - accuracy: 0.7718 - va
Epoch 13/40
758/758 [==============================] - 1s 1ms/sample - loss: 1.0313 - accuracy: 0.7718 - val_
Epoch 14/40
758/758 [==============================] - 1s 846us/sample - loss: 1.0304 - accuracy: 0.7718 - va
Epoch 15/40
758/758 [==============================] - 1s 809us/sample - loss: 1.0293 - accuracy: 0.7718 - va
Epoch 16/40
758/758 [==============================] - 1s 900us/sample - loss: 1.0280 - accuracy: 0.7718 - va
Epoch 17/40
758/758 [==============================] - 1s 812us/sample - loss: 1.0279 - accuracy: 0.7718 - va
Epoch 18/40
758/758 [==============================] - 1s 879us/sample - loss: 1.0261 - accuracy: 0.7718 - va
Epoch 19/40
758/758 [==============================] - 1s 870us/sample - loss: 1.0267 - accuracy: 0.7718 - va
Epoch 20/40
758/758 [==============================] - 1s 817us/sample - loss: 1.0268 - accuracy: 0.7718 - va
Epoch 21/40
758/758 [==============================] - 1s 986us/sample - loss: 1.0250 - accuracy: 0.7718 - va
Epoch 22/40
758/758 [==============================] - 1s 915us/sample - loss: 1.0246 - accuracy: 0.7718 - va
Epoch 23/40
758/758 [==============================] - 1s 830us/sample - loss: 1.0241 - accuracy: 0.7718 - va
Epoch 24/40
758/758 [==============================] - 1s 882us/sample - loss: 1.0237 - accuracy: 0.7718 - va
Epoch 25/40
758/758 [==============================] - 1s 880us/sample - loss: 1.0230 - accuracy: 0.7718 - va
Epoch 26/40
758/758 [==============================] - 1s 866us/sample - loss: 1.0235 - accuracy: 0.7718 - va
```

```
Epoch 27/40
758/758 [==============================] - 1s 939us/sample - loss: 1.0239 - accuracy: 0.7718 - va
Epoch 28/40
758/758 [==============================] - 1s 958us/sample - loss: 1.0244 - accuracy: 0.7718 - va
Epoch 29/40
758/758 [==============================] - 1s 933us/sample - loss: 1.0221 - accuracy: 0.7718 - va
Epoch 30/40
758/758 [==============================] - 1s 858us/sample - loss: 1.0229 - accuracy: 0.7718 - va
Epoch 31/40
758/758 [==============================] - 1s 838us/sample - loss: 1.0228 - accuracy: 0.7718 - va
Epoch 32/40
758/758 [==============================] - 1s 922us/sample - loss: 1.0225 - accuracy: 0.7718 - va
Epoch 33/40
758/758 [==============================] - 1s 940us/sample - loss: 1.0227 - accuracy: 0.7718 - va
Epoch 34/40
758/758 [==============================] - 1s 923us/sample - loss: 1.0220 - accuracy: 0.7718 - va
Epoch 35/40
758/758 [==============================] - 1s 943us/sample - loss: 1.0213 - accuracy: 0.7718 - va
Epoch 36/40
758/758 [==============================] - 1s 1ms/sample - loss: 1.0220 - accuracy: 0.7718 - val_
Epoch 37/40
758/758 [==============================] - 1s 884us/sample - loss: 1.0227 - accuracy: 0.7718 - va
Epoch 38/40
758/758 [==============================] - 1s 975us/sample - loss: 1.0214 - accuracy: 0.7718 - va
Epoch 39/40
758/758 [==============================] - 1s 967us/sample - loss: 1.0215 - accuracy: 0.7718 - va
Epoch 40/40
758/758 [==============================] - 1s 1ms/sample - loss: 1.0213 - accuracy: 0.7718 - val_
```

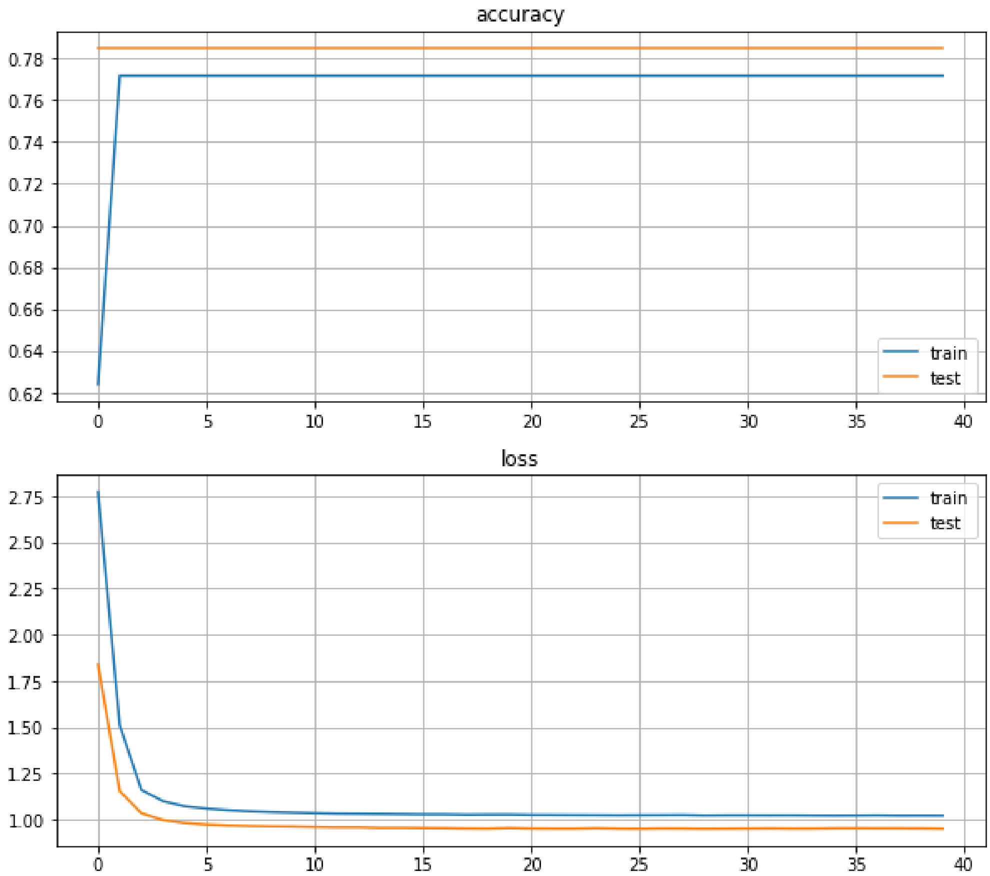
```
In [14]: test_loss, test_acc = model.evaluate(x_test,  y_test, verbose=2)

         print('\nTest acccuracy:', test_acc)
```

```
272/1 - 0s - loss: 1.1919 - accuracy: 0.8272

Test acccuracy: 0.8272059
```

```
In [15]: #plot
         plt.figure(figsize=(10,9))

         plt.subplot(211)
         plt.plot(history.history['accuracy'])
         plt.plot(history.history['val_accuracy'])
         plt.title('accuracy')
         plt.legend(['train', 'test'])
         plt.grid()
         plt.subplot(212)
         plt.plot(history.history['loss'])
```

```
plt.plot(history.history['val_loss'])
plt.title('loss')
plt.legend(['train', 'test'])
plt.grid()
```

```
plt.show()
```



LSTM

- Desempeño de LSTM con una capa vs. deep RNN

```
In [16]: def lstm():
             model = Sequential()
             model.add(Embedding(num_words, emb_dim))
             model.add(LSTM(32, return_sequences = False))
```

```
        model.add(Dense(46, activation = 'softmax'))

        model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['ac

        return model
```

```
In [17]: model = lstm()
         model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_1 (Embedding)      (None, None, 64)          512000
_________________________________________________________________
lstm (LSTM)                  (None, 32)                12416
_________________________________________________________________
dense_1 (Dense)              (None, 46)                1518
=================================================================
Total params: 525,934
Trainable params: 525,934
Non-trainable params: 0
_________________________________________________________________
```

```
In [18]: history = model.fit(x_train, y_train, batch_size = batch, validation_split = 0.3,
                             epochs = epoch, verbose = verbose)
```

```
Train on 758 samples, validate on 326 samples
Epoch 1/40
758/758 [==============================] - 2s 3ms/sample - loss: 3.7126 - accuracy: 0.5858 - val_
Epoch 2/40
758/758 [==============================] - 0s 257us/sample - loss: 2.7176 - accuracy: 0.7718 - va
Epoch 3/40
758/758 [==============================] - 0s 256us/sample - loss: 1.5607 - accuracy: 0.7718 - va
Epoch 4/40
758/758 [==============================] - 0s 251us/sample - loss: 1.1844 - accuracy: 0.7718 - va
Epoch 5/40
758/758 [==============================] - 0s 255us/sample - loss: 1.0998 - accuracy: 0.7718 - va
Epoch 6/40
758/758 [==============================] - 0s 245us/sample - loss: 1.0740 - accuracy: 0.7718 - va
Epoch 7/40
758/758 [==============================] - 0s 231us/sample - loss: 1.0598 - accuracy: 0.7718 - va
Epoch 8/40
758/758 [==============================] - 0s 243us/sample - loss: 1.0517 - accuracy: 0.7718 - va
Epoch 9/40
758/758 [==============================] - 0s 252us/sample - loss: 1.0458 - accuracy: 0.7718 - va
Epoch 10/40
758/758 [==============================] - 0s 259us/sample - loss: 1.0410 - accuracy: 0.7718 - va
Epoch 11/40
```

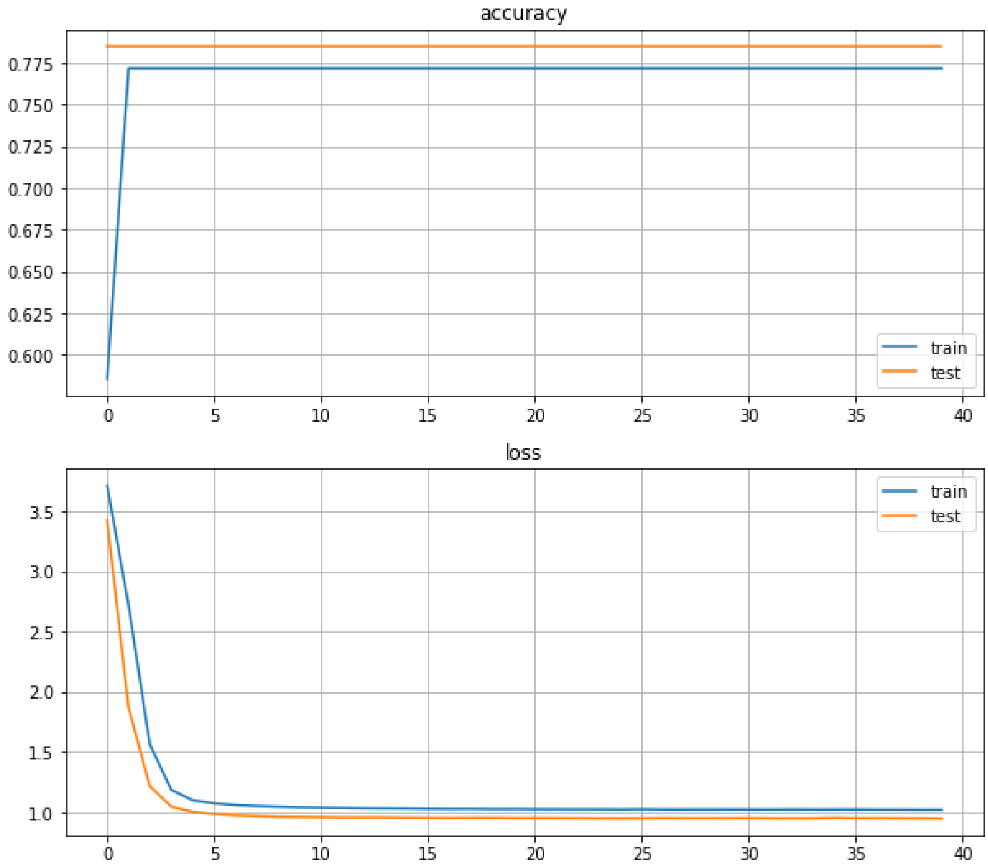
```
758/758 [==============================] - 0s 262us/sample - loss: 1.0380 - accuracy: 0.7718 - va
Epoch 12/40
758/758 [==============================] - 0s 244us/sample - loss: 1.0353 - accuracy: 0.7718 - va
Epoch 13/40
758/758 [==============================] - 0s 352us/sample - loss: 1.0332 - accuracy: 0.7718 - va
Epoch 14/40
758/758 [==============================] - 0s 307us/sample - loss: 1.0320 - accuracy: 0.7718 - va
Epoch 15/40
758/758 [==============================] - 0s 299us/sample - loss: 1.0312 - accuracy: 0.7718 - va
Epoch 16/40
758/758 [==============================] - 0s 266us/sample - loss: 1.0291 - accuracy: 0.7718 - va
Epoch 17/40
758/758 [==============================] - 0s 275us/sample - loss: 1.0282 - accuracy: 0.7718 - va
Epoch 18/40
758/758 [==============================] - 0s 352us/sample - loss: 1.0284 - accuracy: 0.7718 - va
Epoch 19/40
758/758 [==============================] - 0s 301us/sample - loss: 1.0262 - accuracy: 0.7718 - va
Epoch 20/40
758/758 [==============================] - 0s 253us/sample - loss: 1.0268 - accuracy: 0.7718 - va
Epoch 21/40
758/758 [==============================] - 0s 312us/sample - loss: 1.0253 - accuracy: 0.7718 - va
Epoch 22/40
758/758 [==============================] - 0s 328us/sample - loss: 1.0250 - accuracy: 0.7718 - va
Epoch 23/40
758/758 [==============================] - 0s 264us/sample - loss: 1.0248 - accuracy: 0.7718 - va
Epoch 24/40
758/758 [==============================] - 0s 274us/sample - loss: 1.0243 - accuracy: 0.7718 - va
Epoch 25/40
758/758 [==============================] - 0s 285us/sample - loss: 1.0233 - accuracy: 0.7718 - va
Epoch 26/40
758/758 [==============================] - 0s 293us/sample - loss: 1.0251 - accuracy: 0.7718 - va
Epoch 27/40
758/758 [==============================] - 0s 272us/sample - loss: 1.0225 - accuracy: 0.7718 - va
Epoch 28/40
758/758 [==============================] - 0s 277us/sample - loss: 1.0226 - accuracy: 0.7718 - va
Epoch 29/40
758/758 [==============================] - 0s 263us/sample - loss: 1.0229 - accuracy: 0.7718 - va
Epoch 30/40
758/758 [==============================] - 0s 268us/sample - loss: 1.0225 - accuracy: 0.7718 - va
Epoch 31/40
758/758 [==============================] - 0s 301us/sample - loss: 1.0224 - accuracy: 0.7718 - va
Epoch 32/40
758/758 [==============================] - 0s 270us/sample - loss: 1.0218 - accuracy: 0.7718 - va
Epoch 33/40
758/758 [==============================] - 0s 307us/sample - loss: 1.0229 - accuracy: 0.7718 - va
Epoch 34/40
758/758 [==============================] - 0s 286us/sample - loss: 1.0221 - accuracy: 0.7718 - va
Epoch 35/40
758/758 [==============================] - 0s 296us/sample - loss: 1.0223 - accuracy: 0.7718 - va
```

```
Epoch 36/40
758/758 [==============================] - 0s 258us/sample - loss: 1.0226 - accuracy: 0.7718 - va
Epoch 37/40
758/758 [==============================] - 0s 292us/sample - loss: 1.0210 - accuracy: 0.7718 - va
Epoch 38/40
758/758 [==============================] - 0s 244us/sample - loss: 1.0211 - accuracy: 0.7718 - va
Epoch 39/40
758/758 [==============================] - 0s 265us/sample - loss: 1.0199 - accuracy: 0.7718 - va
Epoch 40/40
758/758 [==============================] - 0s 273us/sample - loss: 1.0198 - accuracy: 0.7718 - va
```

```
In [19]: test_loss, test_acc = model.evaluate(x_test,  y_test, verbose=2)

         print('\nTest acccuracy:', test_acc)
```

```
272/1 - 0s - loss: 1.1993 - accuracy: 0.8272

Test acccuracy: 0.8272059
```

```
In [20]: #plot
         plt.figure(figsize=(10,9))

         plt.subplot(211)
         plt.plot(history.history['accuracy'])
         plt.plot(history.history['val_accuracy'])
         plt.title('accuracy')
         plt.legend(['train', 'test'])
         plt.grid()
         plt.subplot(212)
         plt.plot(history.history['loss'])
         plt.plot(history.history['val_loss'])
         plt.title('loss')
         plt.legend(['train', 'test'])
         plt.grid()




         plt.show()
```



Deep LSTM

- LSTM cuentan inicializador 'orthogonal' por defecto

```
In [21]: def deep_lstm():
             model = Sequential()
             model.add(Embedding(num_words, emb_dim))
             model.add(LSTM(32, return_sequences = True, recurrent_initializer='orthogonal'))
             model.add(LSTM(32, return_sequences = False, recurrent_initializer='orthogonal'))
             model.add(Dense(46, activation = 'softmax'))
             model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['ac

             return model
```

```
In [22]: model = deep_lstm()
         model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
```

```
Layer (type)                 Output Shape              Param #   
=================================================================
embedding_2 (Embedding)      (None, None, 64)          512000    
_________________________________________________________________
lstm_1 (LSTM)                (None, None, 32)          12416     
_________________________________________________________________
lstm_2 (LSTM)                (None, 32)                8320      
_________________________________________________________________
dense_2 (Dense)              (None, 46)                1518      
=================================================================
Total params: 534,254
Trainable params: 534,254
Non-trainable params: 0
_________________________________________________________________
```

```
In [23]: history1 = model.fit(x_train, y_train, batch_size = batch, validation_split = 0.3,
                             epochs = epoch, verbose = verbose)
```

```
Train on 758 samples, validate on 326 samples
Epoch 1/40
758/758 [==============================] - 3s 4ms/sample - loss: 3.6785 - accuracy: 0.6821 - val_
Epoch 2/40
758/758 [==============================] - 0s 286us/sample - loss: 2.3124 - accuracy: 0.7718 - va
Epoch 3/40
758/758 [==============================] - 0s 283us/sample - loss: 1.2686 - accuracy: 0.7718 - va
Epoch 4/40
758/758 [==============================] - 0s 327us/sample - loss: 1.1388 - accuracy: 0.7718 - va
Epoch 5/40
758/758 [==============================] - 0s 348us/sample - loss: 1.0823 - accuracy: 0.7718 - va
Epoch 6/40
758/758 [==============================] - 0s 298us/sample - loss: 1.0622 - accuracy: 0.7718 - va
Epoch 7/40
758/758 [==============================] - 0s 300us/sample - loss: 1.0503 - accuracy: 0.7718 - va
Epoch 8/40
758/758 [==============================] - 0s 308us/sample - loss: 1.0449 - accuracy: 0.7718 - va
Epoch 9/40
758/758 [==============================] - 0s 280us/sample - loss: 1.0382 - accuracy: 0.7718 - va
Epoch 10/40
758/758 [==============================] - 0s 286us/sample - loss: 1.0350 - accuracy: 0.7718 - va
Epoch 11/40
758/758 [==============================] - 0s 291us/sample - loss: 1.0334 - accuracy: 0.7718 - va
Epoch 12/40
758/758 [==============================] - 0s 322us/sample - loss: 1.0319 - accuracy: 0.7718 - va
Epoch 13/40
758/758 [==============================] - 0s 323us/sample - loss: 1.0290 - accuracy: 0.7718 - va
Epoch 14/40
758/758 [==============================] - 0s 288us/sample - loss: 1.0291 - accuracy: 0.7718 - va
Epoch 15/40
758/758 [==============================] - 0s 274us/sample - loss: 1.0271 - accuracy: 0.7718 - va
```

```
Epoch 16/40
758/758 [==============================] - 0s 275us/sample - loss: 1.0234 - accuracy: 0.7718 - va
Epoch 17/40
758/758 [==============================] - 0s 285us/sample - loss: 1.0145 - accuracy: 0.7718 - va
Epoch 18/40
758/758 [==============================] - 0s 285us/sample - loss: 0.9961 - accuracy: 0.7718 - va
Epoch 19/40
758/758 [==============================] - 0s 282us/sample - loss: 0.9568 - accuracy: 0.7718 - va
Epoch 20/40
758/758 [==============================] - 0s 275us/sample - loss: 0.8873 - accuracy: 0.7718 - va
Epoch 21/40
758/758 [==============================] - 0s 278us/sample - loss: 0.9047 - accuracy: 0.7718 - va
Epoch 22/40
758/758 [==============================] - 0s 294us/sample - loss: 1.0311 - accuracy: 0.7718 - va
Epoch 23/40
758/758 [==============================] - 0s 289us/sample - loss: 1.0250 - accuracy: 0.7718 - va
Epoch 24/40
758/758 [==============================] - 0s 287us/sample - loss: 1.0251 - accuracy: 0.7718 - va
Epoch 25/40
758/758 [==============================] - 0s 336us/sample - loss: 1.0201 - accuracy: 0.7718 - va
Epoch 26/40
758/758 [==============================] - 0s 329us/sample - loss: 1.0066 - accuracy: 0.7718 - va
Epoch 27/40
758/758 [==============================] - 0s 432us/sample - loss: 0.9466 - accuracy: 0.7718 - va
Epoch 28/40
758/758 [==============================] - 0s 377us/sample - loss: 0.8134 - accuracy: 0.7810 - va
Epoch 29/40
758/758 [==============================] - 0s 310us/sample - loss: 0.9883 - accuracy: 0.7836 - va
Epoch 30/40
758/758 [==============================] - 0s 288us/sample - loss: 1.0242 - accuracy: 0.7718 - va
Epoch 31/40
758/758 [==============================] - 0s 267us/sample - loss: 1.0188 - accuracy: 0.7718 - va
Epoch 32/40
758/758 [==============================] - 0s 385us/sample - loss: 1.0178 - accuracy: 0.7718 - va
Epoch 33/40
758/758 [==============================] - 0s 400us/sample - loss: 1.0166 - accuracy: 0.7718 - va
Epoch 34/40
758/758 [==============================] - 0s 435us/sample - loss: 1.0162 - accuracy: 0.7718 - va
Epoch 35/40
758/758 [==============================] - 0s 381us/sample - loss: 1.0164 - accuracy: 0.7718 - va
Epoch 36/40
758/758 [==============================] - 0s 409us/sample - loss: 1.0155 - accuracy: 0.7718 - va
Epoch 37/40
758/758 [==============================] - 0s 446us/sample - loss: 1.0130 - accuracy: 0.7718 - va
Epoch 38/40
758/758 [==============================] - 0s 426us/sample - loss: 1.0148 - accuracy: 0.7718 - va
Epoch 39/40
758/758 [==============================] - 0s 340us/sample - loss: 0.9521 - accuracy: 0.7718 - va
Epoch 40/40
```

```
758/758 [==============================] - 0s 261us/sample - loss: 0.7970 - accuracy: 0.8153 - va
```

```
In [24]: test_loss, test_acc = model.evaluate(x_test,  y_test, verbose=2)

         print('\nTest acccuracy:', test_acc)
```

```
272/1 - 0s - loss: 1.0245 - accuracy: 0.8824

Test acccuracy: 0.88235295
```

- Modificando inicializador

```
In [25]: def deep_lstm():
             model = Sequential()
             model.add(Embedding(num_words, emb_dim))
             model.add(LSTM(32, return_sequences = True, recurrent_initializer='glorot_uniform'))
             model.add(LSTM(32, return_sequences = False, recurrent_initializer='glorot_uniform')
             model.add(Dense(46, activation = 'softmax'))
             model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['ac

             return model
```

```
In [26]: model = deep_lstm()
         model.summary()
```

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_3 (Embedding)      (None, None, 64)          512000
_________________________________________________________________
lstm_3 (LSTM)                (None, None, 32)          12416
_________________________________________________________________
lstm_4 (LSTM)                (None, 32)                8320
_________________________________________________________________
dense_3 (Dense)              (None, 46)                1518
=================================================================
Total params: 534,254
Trainable params: 534,254
Non-trainable params: 0
_________________________________________________________________
```

```
In [27]: history2 = model.fit(x_train, y_train, batch_size = batch, validation_split = 0.3,
                              epochs = 40, verbose = verbose)
```

```
Train on 758 samples, validate on 326 samples
Epoch 1/40
758/758 [==============================] - 4s 5ms/sample - loss: 3.5942 - accuracy: 0.6478 - val_
```

```
Epoch 2/40
758/758 [==============================] - 0s 397us/sample - loss: 2.1352 - accuracy: 0.7718 - va
Epoch 3/40
758/758 [==============================] - 0s 412us/sample - loss: 1.2587 - accuracy: 0.7718 - va
Epoch 4/40
758/758 [==============================] - 0s 296us/sample - loss: 1.1126 - accuracy: 0.7718 - va
Epoch 5/40
758/758 [==============================] - 0s 352us/sample - loss: 1.0777 - accuracy: 0.7718 - va
Epoch 6/40
758/758 [==============================] - 0s 364us/sample - loss: 1.0595 - accuracy: 0.7718 - va
Epoch 7/40
758/758 [==============================] - 0s 442us/sample - loss: 1.0511 - accuracy: 0.7718 - va
Epoch 8/40
758/758 [==============================] - 0s 438us/sample - loss: 1.0439 - accuracy: 0.7718 - va
Epoch 9/40
758/758 [==============================] - 0s 451us/sample - loss: 1.0396 - accuracy: 0.7718 - va
Epoch 10/40
758/758 [==============================] - 0s 416us/sample - loss: 1.0362 - accuracy: 0.7718 - va
Epoch 11/40
758/758 [==============================] - 0s 328us/sample - loss: 1.0344 - accuracy: 0.7718 - va
Epoch 12/40
758/758 [==============================] - 0s 261us/sample - loss: 1.0335 - accuracy: 0.7718 - va
Epoch 13/40
758/758 [==============================] - 0s 263us/sample - loss: 1.0302 - accuracy: 0.7718 - va
Epoch 14/40
758/758 [==============================] - 0s 261us/sample - loss: 1.0296 - accuracy: 0.7718 - va
Epoch 15/40
758/758 [==============================] - 0s 271us/sample - loss: 1.0282 - accuracy: 0.7718 - va
Epoch 16/40
758/758 [==============================] - 0s 261us/sample - loss: 1.0273 - accuracy: 0.7718 - va
Epoch 17/40
758/758 [==============================] - 0s 259us/sample - loss: 1.0265 - accuracy: 0.7718 - va
Epoch 18/40
758/758 [==============================] - 0s 261us/sample - loss: 1.0272 - accuracy: 0.7718 - va
Epoch 19/40
758/758 [==============================] - 0s 268us/sample - loss: 1.0248 - accuracy: 0.7718 - va
Epoch 20/40
758/758 [==============================] - 0s 498us/sample - loss: 1.0261 - accuracy: 0.7718 - va
Epoch 21/40
758/758 [==============================] - 0s 396us/sample - loss: 1.0247 - accuracy: 0.7718 - va
Epoch 22/40
758/758 [==============================] - 0s 260us/sample - loss: 1.0237 - accuracy: 0.7718 - va
Epoch 23/40
758/758 [==============================] - 0s 253us/sample - loss: 1.0231 - accuracy: 0.7718 - va
Epoch 24/40
758/758 [==============================] - 0s 258us/sample - loss: 1.0235 - accuracy: 0.7718 - va
Epoch 25/40
758/758 [==============================] - 0s 259us/sample - loss: 1.0217 - accuracy: 0.7718 - va
Epoch 26/40
```

```
758/758 [==============================] - 0s 312us/sample - loss: 1.0144 - accuracy: 0.7718 - va
Epoch 27/40
758/758 [==============================] - 0s 269us/sample - loss: 0.9809 - accuracy: 0.7718 - va
Epoch 28/40
758/758 [==============================] - 0s 261us/sample - loss: 0.8520 - accuracy: 0.7770 - va
Epoch 29/40
758/758 [==============================] - 0s 267us/sample - loss: 0.7102 - accuracy: 0.8602 - va
Epoch 30/40
758/758 [==============================] - 0s 292us/sample - loss: 0.6853 - accuracy: 0.8417 - va
Epoch 31/40
758/758 [==============================] - 0s 293us/sample - loss: 0.6408 - accuracy: 0.8562 - va
Epoch 32/40
758/758 [==============================] - 0s 275us/sample - loss: 0.5880 - accuracy: 0.8641 - va
Epoch 33/40
758/758 [==============================] - 0s 390us/sample - loss: 0.5779 - accuracy: 0.8615 - va
Epoch 34/40
758/758 [==============================] - 0s 406us/sample - loss: 0.5824 - accuracy: 0.8654 - va
Epoch 35/40
758/758 [==============================] - 0s 270us/sample - loss: 0.5104 - accuracy: 0.8852 - va
Epoch 36/40
758/758 [==============================] - 0s 308us/sample - loss: 0.4854 - accuracy: 0.8945 - va
Epoch 37/40
758/758 [==============================] - 0s 323us/sample - loss: 0.4664 - accuracy: 0.8945 - va
Epoch 38/40
758/758 [==============================] - 0s 434us/sample - loss: 0.4511 - accuracy: 0.8905 - va
Epoch 39/40
758/758 [==============================] - 0s 414us/sample - loss: 0.4427 - accuracy: 0.8892 - va
Epoch 40/40
758/758 [==============================] - 0s 457us/sample - loss: 0.4296 - accuracy: 0.9037 - va
```

```
In [28]: test_loss, test_acc = model.evaluate(x_test,  y_test, verbose=2)

         print('\nTest acccuracy:', test_acc)
```

```
272/1 - 0s - loss: 0.9204 - accuracy: 0.8934

Test acccuracy: 0.8933824
```

- Recurrent dropout

```
In [29]: def deep_lstm():
             model = Sequential()
             model.add(Embedding(num_words, emb_dim))
             model.add(LSTM(32, return_sequences = True, recurrent_initializer='glorot_uniform',
                            recurrent_dropout=0.1))
             model.add(LSTM(32, return_sequences = False, recurrent_initializer='glorot_uniform',
                            recurrent_dropout=0.1))
             model.add(Dense(46, activation = 'softmax'))
```

```
            model.compile(loss='sparse_categorical_crossentropy', optimizer='adam', metrics=['ac

            return model
```

```
In [30]: model = deep_lstm()
         model.summary()
```

```
Model: "sequential_4"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
embedding_4 (Embedding)      (None, None, 64)          512000
_________________________________________________________________
lstm_5 (LSTM)                (None, None, 32)          12416
_________________________________________________________________
lstm_6 (LSTM)                (None, 32)                8320
_________________________________________________________________
dense_4 (Dense)              (None, 46)                1518
=================================================================
Total params: 534,254
Trainable params: 534,254
Non-trainable params: 0
_________________________________________________________________
```

```
In [31]: history3 = model.fit(x_train, y_train, batch_size = batch, validation_split = 0.3,
                             epochs = epoch, verbose = verbose)
```

```
Train on 758 samples, validate on 326 samples
Epoch 1/40
758/758 [==============================] - 4s 5ms/sample - loss: 3.4562 - accuracy: 0.6939 - val_
Epoch 2/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.8847 - accuracy: 0.7718 - val_
Epoch 3/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.2168 - accuracy: 0.7718 - val_
Epoch 4/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.0976 - accuracy: 0.7718 - val_
Epoch 5/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.0700 - accuracy: 0.7718 - val_
Epoch 6/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.0546 - accuracy: 0.7718 - val_
Epoch 7/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.0440 - accuracy: 0.7718 - val_
Epoch 8/40
758/758 [==============================] - 1s 2ms/sample - loss: 1.0232 - accuracy: 0.7718 - val_
Epoch 9/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.9878 - accuracy: 0.7718 - val_
Epoch 10/40
758/758 [==============================] - 2s 2ms/sample - loss: 0.9038 - accuracy: 0.7718 - val_
Epoch 11/40
```

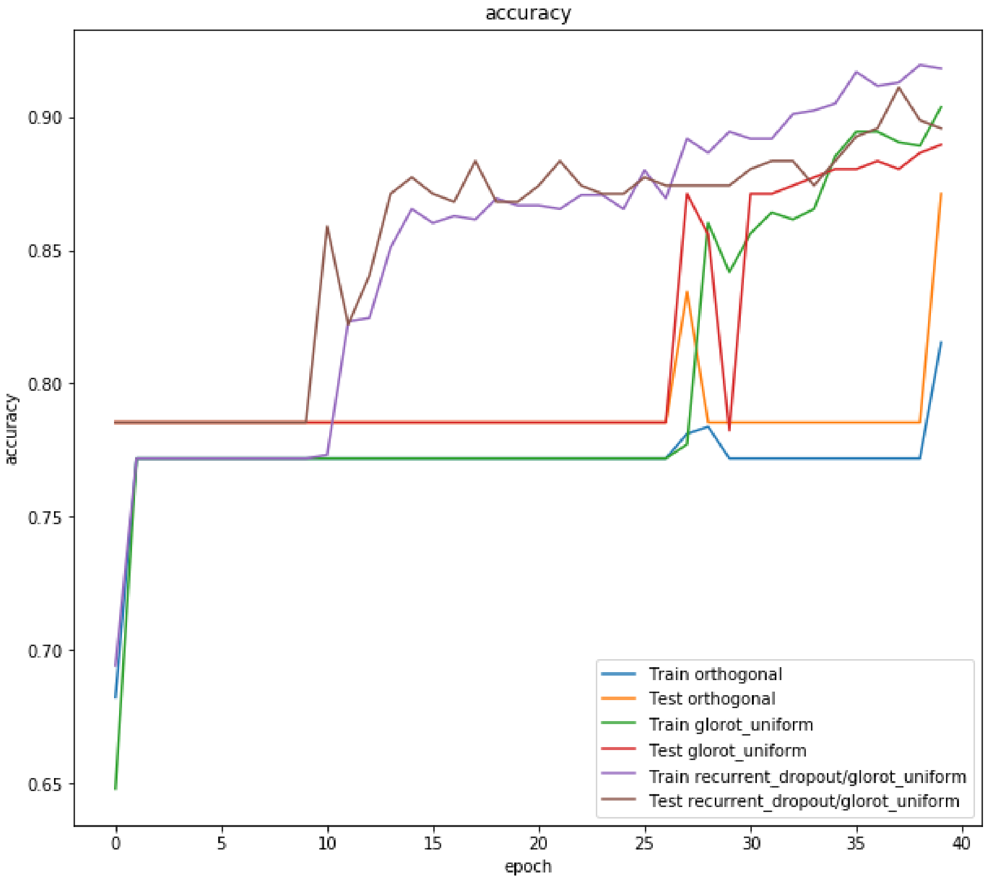
```
758/758 [==============================] - 1s 2ms/sample - loss: 0.8087 - accuracy: 0.7731 - val_
Epoch 12/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.7716 - accuracy: 0.8232 - val_
Epoch 13/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.7254 - accuracy: 0.8245 - val_
Epoch 14/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.6757 - accuracy: 0.8509 - val_
Epoch 15/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.6354 - accuracy: 0.8654 - val_
Epoch 16/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.6219 - accuracy: 0.8602 - val_
Epoch 17/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.6038 - accuracy: 0.8628 - val_
Epoch 18/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5822 - accuracy: 0.8615 - val_
Epoch 19/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5577 - accuracy: 0.8694 - val_
Epoch 20/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5408 - accuracy: 0.8668 - val_
Epoch 21/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5221 - accuracy: 0.8668 - val_
Epoch 22/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5132 - accuracy: 0.8654 - val_
Epoch 23/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5049 - accuracy: 0.8707 - val_
Epoch 24/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4926 - accuracy: 0.8707 - val_
Epoch 25/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.5031 - accuracy: 0.8654 - val_
Epoch 26/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4874 - accuracy: 0.8799 - val_
Epoch 27/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4686 - accuracy: 0.8694 - val_
Epoch 28/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4507 - accuracy: 0.8918 - val_
Epoch 29/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4365 - accuracy: 0.8865 - val_
Epoch 30/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4259 - accuracy: 0.8945 - val_
Epoch 31/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4198 - accuracy: 0.8918 - val_
Epoch 32/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4244 - accuracy: 0.8918 - val_
Epoch 33/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.4049 - accuracy: 0.9011 - val_
Epoch 34/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3981 - accuracy: 0.9024 - val_
Epoch 35/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3874 - accuracy: 0.9050 - val_
```

```
Epoch 36/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3764 - accuracy: 0.9169 - val_
Epoch 37/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3711 - accuracy: 0.9116 - val_
Epoch 38/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3615 - accuracy: 0.9129 - val_
Epoch 39/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3595 - accuracy: 0.9195 - val_
Epoch 40/40
758/758 [==============================] - 1s 2ms/sample - loss: 0.3559 - accuracy: 0.9182 - val_
```

```
In [32]: test_loss, test_acc = model.evaluate(x_test,  y_test, verbose=2)

         print('\nTest acccuracy:', test_acc)
```

```
272/1 - 0s - loss: 0.9561 - accuracy: 0.9118

Test acccuracy: 0.9117647
```

```
In [34]: #plot
         plt.figure(figsize=(10,9))
         plt.plot(history1.history['accuracy'])
         plt.plot(history1.history['val_accuracy'])


         plt.plot(history2.history['accuracy'])
         plt.plot(history2.history['val_accuracy'])

         plt.plot(history3.history['accuracy'])
         plt.plot(history3.history['val_accuracy'])


         plt.legend(['Train orthogonal', 'Test orthogonal',
                     'Train glorot_uniform', 'Test glorot_uniform',
                     'Train recurrent_dropout/glorot_uniform', 'Test recurrent_dropout/glorot_uni

         plt.title('accuracy')
         plt.ylabel('accuracy')
         plt.xlabel('epoch')

         plt.show()
```



- Mejorar test accuracy
- Investigar que son las GRU e implementarlas
- Experimentar con diferente número de capas y neuronas, mejorar los resultados
- Experimentar otros inicializadores y diferentes valores de dropout
- Probar con otro dataset