基于 RNN、LSTM、GRU 的谣言分类预测

2022 年 5 月 31 日

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

[2]:
```
data = pd.read_csv(r'E:\桌面\疫情谣言\covid19_rumors.csv')
```

[3]:
```
for i in data.columns:
    print('%s的分布:\n' % i, data[i].value_counts())
    print('---------------------')
```

```
crawlTime 的分布:
 2020-01-29    117
2020-02-19     86
2020-01-31     51
2020-01-30     43
2020-01-26     24
              …
2020-04-08      1
2020-03-29      1
2020-04-10      1
2020-04-18      1
2020-03-30      1
Name: crawlTime, Length: 91, dtype: int64
---------------------
mainSummary 的分布:
 世界卫生组织回应：目前没有证据显示狗猫等宠物会感染新型冠状病毒      6
```

```
丁香医生团队：交流要保持距离；尚未证实新冠病毒可以通过空气传播。      5
中科院辟谣：并非同一种                                    4
丁香医生团队辟谣：SARI 是「严重急性呼吸道感染」的英文缩写        3
经查证：系编造                                          3
                                                      ..
1 月 25 日上午，澎湃新闻从呼吸疾病国家重点实验室办公室获悉…      1
感染概率很低…                                           1
美国总统特朗普因感染新冠病毒在演讲过程中晕倒的说法系谣言…        1
丁香医生团队：交流要保持距离；病毒不会在空气中悬浮               1
2020 年 1 月 24 日…                                      1
Name: mainSummary, Length: 796, dtype: int64
---------------------
rumorType 的分布:
 fake     662
doubt    139
true      63
Name: rumorType, dtype: int64
---------------------
title 的分布:
 人会传染宠物                           8
带毛领或绒线的外套容易吸附病毒                   6
电吹风对手和面部吹 30 秒能消毒                  5
用了 7 天的 N95 口罩消毒可继续用                 5
服用 VC 可以预防感染                        5
                                        ..
来自抗击疫情一线消息，李留树博士刚从武汉打来电话       1
深圳 49 家医院可免费领口罩                     1
亚洲人更容易感染新型冠状病毒                     1
新冠肺炎疫苗已研制成功                        1
循环使用的地铁票会传播病毒                      1
Name: title, Length: 789, dtype: int64
---------------------
```

[4]:
```python
print('数据量: ', len(data['rumorType']))
print('***********************')
print(data.info())
```

```
数据量： 864
***********************
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 864 entries, 0 to 863
Data columns (total 4 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   crawlTime    864 non-null    object
 1   mainSummary  864 non-null    object
 2   rumorType    864 non-null    object
 3   title        864 non-null    object
dtypes: object(4)
memory usage: 27.1+ KB
None
```

[5]:
```
print(data['mainSummary'][0])
```

```
2 月 23 日…
```

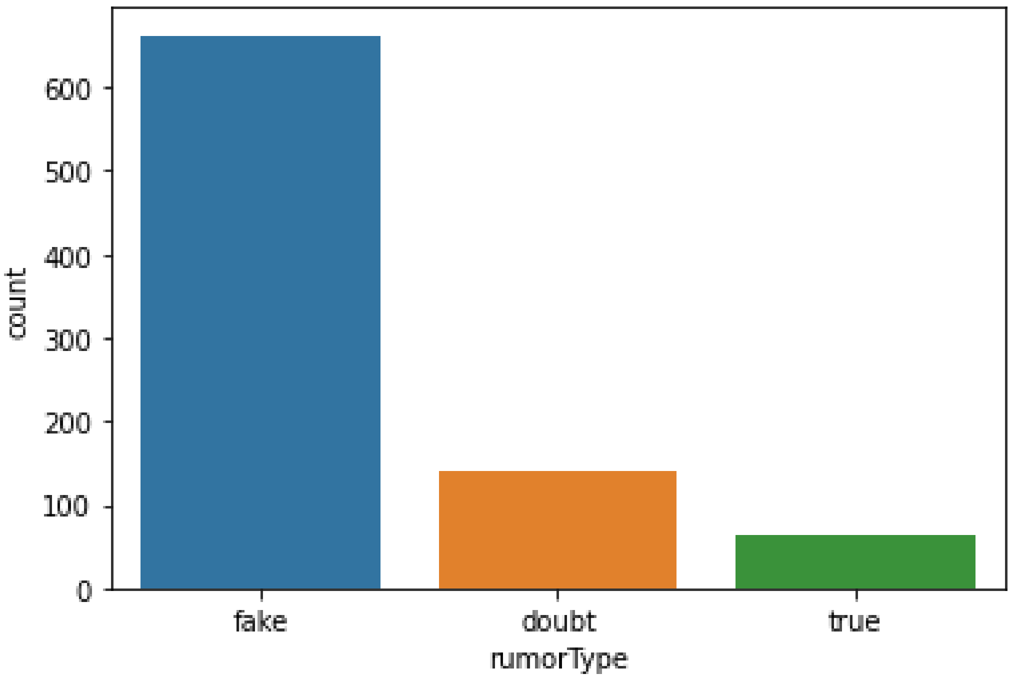
[6]:
```
print(data['title'][0])
```

```
国家体育总局下发 4 月 30 日前禁止办赛通知
```

[7]:
```
data['rumorType'].value_counts()
```

[7]:
```
fake     662
doubt    139
true      63
Name: rumorType, dtype: int64
```

[8]:
```
sns.countplot(data['rumorType'])
plt.show()
print(data['rumorType'].value_counts())
```




```
fake     662
doubt    139
true      63
Name: rumorType, dtype: int64
```

[9]:
```
labels = []
for i in range(len(data['rumorType'])):
    if data['rumorType'][i] == 'fake':
        labels.append(1)
    if data['rumorType'][i] == 'true':
        labels.append(0)
    if data['rumorType'][i] == 'doubt':
        labels.append(2)
```

[10]:
```
train = data[['mainSummary','title']].apply(lambda x:' '.join(x),axis=1)
train.head()
```

[10]:
```
0                          2 月 23 日… 国家体育总局下发 4 月 30 日前禁止办赛通知
```

```
1      世界卫生组织辟谣 ：不能，肺炎球菌疫苗和乙型流感嗜血杆菌疫苗等肺炎疫苗不能预防新冠肺炎 普通…
2      在法国，无论是“黄马甲”运动还是大罢工，目前都并未结束… 一个武汉女人终结了法国大罢工，…
3                    经查证，这是一条电脑合成的焰火视频… 武汉长江大桥燃放烟花驱疫
4      网传消息并没有罗列出将在上海哪些主干道、用什么药物进行消毒作业… 上海主干道今晚12 点大…
dtype: object
```

[ ]:

```
[11]: import string
import jieba
import re
import tensorflow as tf
import sklearn as sl
```

```
[12]: def tokenize_text(text):
    tokens = jieba.cut(text, cut_all=False)
    tokens = [token.strip() for token in tokens]
    return tokens

def remove_special_characters(text):
    tokens = tokenize_text(text)
    pattern = re.compile('[{}]'.format(re.escape(string.punctuation)))
    filtered_tokens = filter(None, [pattern.sub(' ', token) for token in␣
 ↪tokens])
    filtered_text = ' '.join(filtered_tokens)
    return filtered_text

def remove_stopwords(text):
    tokens = tokenize_text(text)
    filtered_tokens = [token for token in tokens if token not in stopwords]
    filtered_text = ' '.join(filtered_tokens)
    return filtered_text

def normalize_corpus(corpus, tokenize=False):
```

```python
    normalized_corpus = []
    for text in corpus:
        text = remove_special_characters(text)
        text = remove_stopwords(text)
        normalized_corpus.append(text)
        if tokenize:
            text = tokenized_corpus.append(text)
    return normalized_corpus
```

[13]:
```python
# 自建停用词表
with open(r'E:\桌面\停用词表.txt') as f:
    stopwords = f.read()
```

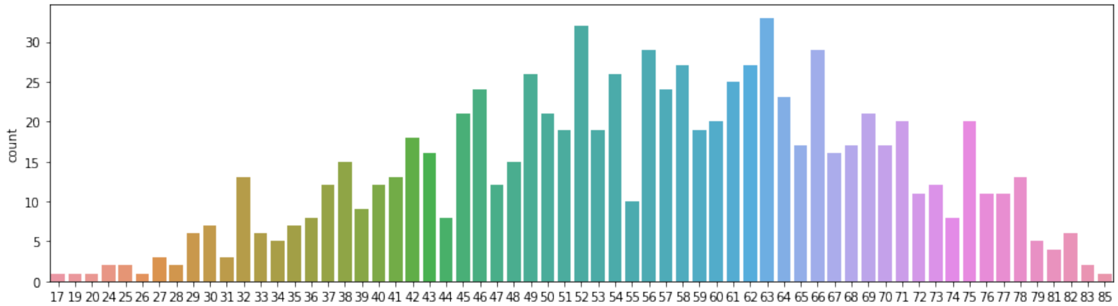
[14]:
```python
norm_corpus = normalize_corpus(train)
```

```
Building prefix dict from the default dictionary …
Loading model from cache C:\Users\WS\AppData\Local\Temp\jieba.cache
Loading model cost 1.420 seconds.
Prefix dict has been built successfully.
```

[15]:
```python
b = []
c = 0
for i in norm_corpus:
    a = len(i)
    b.append(a)
    c += a
```

[16]:
```python
fig = plt.figure(figsize=(15, 4))
sns.countplot(b)
print('词数: ', c)
```

```
词数:  48390
```



[17]:
```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_vec = TfidfVectorizer(analyzer='word', max_features=40000,␣
 ↪ngram_range=(1, 4),
                                binary=True, use_idf=1, smooth_idf=1,␣
 ↪sublinear_tf=1, norm='l2').fit_transform(norm_corpus)
```

[18]:
```
labels = np.array(labels)
```

[19]:
```
from sklearn.model_selection import train_test_split
train_x, test_x, train_y, test_y = train_test_split(tfidf_vec, labels,␣
 ↪test_size=.2, shuffle=True, random_state=42)
```

[21]:
```
a = train_x.toarray()
print('a:', a.shape)

b = test_x.toarray()
print('b:', b.shape)
```

```
a: (691, 29664)
b: (173, 29664)
```

[22]:
```
train_xx = np.reshape(a, (691,1,29664))
test_xx = np.reshape(b, (173, 1,29664))
```

[35]:
```
model = tf.keras.models.Sequential([
    tf.keras.layers.LSTM(36, return_sequences=True),
    tf.keras.layers.Dropout(0.3),
    tf.keras.layers.GRU(24, return_sequences=True),
```

```
    tf.keras.layers.Dropout(0.2),
    tf.keras.layers.SimpleRNN(12, return_sequences=True),
    tf.keras.layers.Dropout(0.1),
    tf.keras.layers.Dense(3, activation='softmax')
])
```

[36]:
```
model.compile(optimizer=tf.keras.optimizers.Adam(0.001),
              loss=tf.keras.losses.
 ↪SparseCategoricalCrossentropy(from_logits=False),
              metrics=['sparse_categorical_accuracy'])
```

[37]:
```
import os
checkpoint_save_path = 'E:\\桌面\\1.ckpt'
if os.path.exists(checkpoint_save_path + '.index'):
    print('-----load model---------')
    model.load_weights(checkpoint_save_path)
cp_callback = tf.keras.callbacks.ModelCheckpoint(filepath=checkpoint_save_path,
                                                 save_weights_only=True,
                                                 asve_best_only=True,␣
 ↪monitor='val_loss')
```

```
-----load model---------
```

[38]:
```
history = model.fit(train_xx, train_y, batch_size=32, epochs=100,␣
 ↪callbacks=[cp_callback],
                    validation_data=(test_xx, test_y), validation_freq=1,)
```

```
Train on 691 samples, validate on 173 samples
Epoch 1/100
691/691 [==============================] - 5s 8ms/sample - loss: 0.9409 -
sparse_categorical_accuracy: 0.9479 - val_loss: 1.0255 -
val_sparse_categorical_accuracy: 0.8035
Epoch 2/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.8830 -
sparse_categorical_accuracy: 0.9493 - val_loss: 1.0092 -
val_sparse_categorical_accuracy: 0.8035
Epoch 3/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.8109 -
```

```
sparse_categorical_accuracy: 0.9493 - val_loss: 0.9913 -
val_sparse_categorical_accuracy: 0.8092
Epoch 4/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.7434 -
sparse_categorical_accuracy: 0.9624 - val_loss: 0.9718 -
val_sparse_categorical_accuracy: 0.8092
Epoch 5/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.6675 -
sparse_categorical_accuracy: 0.9638 - val_loss: 0.9503 -
val_sparse_categorical_accuracy: 0.8092
Epoch 6/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.6001 -
sparse_categorical_accuracy: 0.9783 - val_loss: 0.9291 -
val_sparse_categorical_accuracy: 0.8092
Epoch 7/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.5421 -
sparse_categorical_accuracy: 0.9725 - val_loss: 0.9083 -
val_sparse_categorical_accuracy: 0.8092
Epoch 8/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.4846 -
sparse_categorical_accuracy: 0.9783 - val_loss: 0.8890 -
val_sparse_categorical_accuracy: 0.8035
Epoch 9/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.4481 -
sparse_categorical_accuracy: 0.9754 - val_loss: 0.8711 -
val_sparse_categorical_accuracy: 0.7977
Epoch 10/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.3977 -
sparse_categorical_accuracy: 0.9812 - val_loss: 0.8550 -
val_sparse_categorical_accuracy: 0.7977
Epoch 11/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.3652 -
sparse_categorical_accuracy: 0.9841 - val_loss: 0.8408 -
val_sparse_categorical_accuracy: 0.7977
Epoch 12/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.3432 -
sparse_categorical_accuracy: 0.9870 - val_loss: 0.8282 -
```

```
val_sparse_categorical_accuracy: 0.7977
Epoch 13/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.3155 -
sparse_categorical_accuracy: 0.9841 - val_loss: 0.8173 -
val_sparse_categorical_accuracy: 0.7977
Epoch 14/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.2951 -
sparse_categorical_accuracy: 0.9841 - val_loss: 0.8072 -
val_sparse_categorical_accuracy: 0.7977
Epoch 15/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.2753 -
sparse_categorical_accuracy: 0.9928 - val_loss: 0.7984 -
val_sparse_categorical_accuracy: 0.7977
Epoch 16/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.2607 -
sparse_categorical_accuracy: 0.9884 - val_loss: 0.7901 -
val_sparse_categorical_accuracy: 0.7977
Epoch 17/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.2506 -
sparse_categorical_accuracy: 0.9913 - val_loss: 0.7825 -
val_sparse_categorical_accuracy: 0.7977
Epoch 18/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.2348 -
sparse_categorical_accuracy: 0.9928 - val_loss: 0.7756 -
val_sparse_categorical_accuracy: 0.7977
Epoch 19/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.2260 -
sparse_categorical_accuracy: 0.9942 - val_loss: 0.7689 -
val_sparse_categorical_accuracy: 0.7977
Epoch 20/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.2199 -
sparse_categorical_accuracy: 0.9928 - val_loss: 0.7631 -
val_sparse_categorical_accuracy: 0.7977
Epoch 21/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.2062 -
sparse_categorical_accuracy: 0.9928 - val_loss: 0.7575 -
val_sparse_categorical_accuracy: 0.7977
```

```
Epoch 22/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1995 -
sparse_categorical_accuracy: 0.9913 - val_loss: 0.7524 -
val_sparse_categorical_accuracy: 0.7977
Epoch 23/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1829 -
sparse_categorical_accuracy: 0.9971 - val_loss: 0.7482 -
val_sparse_categorical_accuracy: 0.7977
Epoch 24/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1801 -
sparse_categorical_accuracy: 0.9957 - val_loss: 0.7439 -
val_sparse_categorical_accuracy: 0.7977
Epoch 25/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1696 -
sparse_categorical_accuracy: 0.9957 - val_loss: 0.7395 -
val_sparse_categorical_accuracy: 0.7977
Epoch 26/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1611 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7352 -
val_sparse_categorical_accuracy: 0.7977
Epoch 27/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1564 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.7313 -
val_sparse_categorical_accuracy: 0.7977
Epoch 28/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1478 -
sparse_categorical_accuracy: 0.9971 - val_loss: 0.7276 -
val_sparse_categorical_accuracy: 0.7977
Epoch 29/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1383 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7243 -
val_sparse_categorical_accuracy: 0.7977
Epoch 30/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1306 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7212 -
val_sparse_categorical_accuracy: 0.7977
Epoch 31/100
```

```
691/691 [==============================] - 2s 3ms/sample - loss: 0.1257 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7180 -
val_sparse_categorical_accuracy: 0.7977
Epoch 32/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.1224 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7148 -
val_sparse_categorical_accuracy: 0.7977
Epoch 33/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1204 -
sparse_categorical_accuracy: 0.9971 - val_loss: 0.7120 -
val_sparse_categorical_accuracy: 0.7977
Epoch 34/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1104 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.7093 -
val_sparse_categorical_accuracy: 0.7977
Epoch 35/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1048 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.7069 -
val_sparse_categorical_accuracy: 0.7977
Epoch 36/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.1040 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.7040 -
val_sparse_categorical_accuracy: 0.7977
Epoch 37/100
691/691 [==============================] - 3s 4ms/sample - loss: 0.0970 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.7019 -
val_sparse_categorical_accuracy: 0.7977
Epoch 38/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0904 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6997 -
val_sparse_categorical_accuracy: 0.7977
Epoch 39/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0846 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6978 -
val_sparse_categorical_accuracy: 0.7977
Epoch 40/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0833 -
```

```
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6955 -
val_sparse_categorical_accuracy: 0.7977
Epoch 41/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0808 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6933 -
val_sparse_categorical_accuracy: 0.7977
Epoch 42/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0779 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6914 -
val_sparse_categorical_accuracy: 0.7977
Epoch 43/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0739 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6897 -
val_sparse_categorical_accuracy: 0.7977
Epoch 44/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0739 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6877 -
val_sparse_categorical_accuracy: 0.7977
Epoch 45/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0681 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6859 -
val_sparse_categorical_accuracy: 0.7977
Epoch 46/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0634 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6840 -
val_sparse_categorical_accuracy: 0.7977
Epoch 47/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0616 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6822 -
val_sparse_categorical_accuracy: 0.7977
Epoch 48/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0572 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6804 -
val_sparse_categorical_accuracy: 0.7977
Epoch 49/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0568 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6789 -
```

```
val_sparse_categorical_accuracy: 0.7977
Epoch 50/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0534 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6776 -
val_sparse_categorical_accuracy: 0.7977
Epoch 51/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0511 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6760 -
val_sparse_categorical_accuracy: 0.7977
Epoch 52/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0518 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6744 -
val_sparse_categorical_accuracy: 0.7977
Epoch 53/100
691/691 [==============================] - 3s 4ms/sample - loss: 0.0539 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6730 -
val_sparse_categorical_accuracy: 0.7977
Epoch 54/100
691/691 [==============================] - 2s 4ms/sample - loss: 0.0472 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6725 -
val_sparse_categorical_accuracy: 0.7977
Epoch 55/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0442 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6712 -
val_sparse_categorical_accuracy: 0.7977
Epoch 56/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0467 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6701 -
val_sparse_categorical_accuracy: 0.7977
Epoch 57/100
691/691 [==============================] - 3s 4ms/sample - loss: 0.0441 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6688 -
val_sparse_categorical_accuracy: 0.7977
Epoch 58/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0451 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6678 -
val_sparse_categorical_accuracy: 0.8035
```

```
Epoch 59/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0395 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6668 -
val_sparse_categorical_accuracy: 0.8035
Epoch 60/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0396 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6655 -
val_sparse_categorical_accuracy: 0.8035
Epoch 61/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0362 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6644 -
val_sparse_categorical_accuracy: 0.8035
Epoch 62/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0375 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6635 -
val_sparse_categorical_accuracy: 0.8035
Epoch 63/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0335 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6626 -
val_sparse_categorical_accuracy: 0.8035
Epoch 64/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0360 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6619 -
val_sparse_categorical_accuracy: 0.8035
Epoch 65/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0323 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6607 -
val_sparse_categorical_accuracy: 0.8092
Epoch 66/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0349 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6596 -
val_sparse_categorical_accuracy: 0.8092
Epoch 67/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0324 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6586 -
val_sparse_categorical_accuracy: 0.8092
Epoch 68/100
```

```
691/691 [==============================] - 2s 2ms/sample - loss: 0.0313 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6575 -
val_sparse_categorical_accuracy: 0.8092
Epoch 69/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0275 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6567 -
val_sparse_categorical_accuracy: 0.8092
Epoch 70/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0270 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6559 -
val_sparse_categorical_accuracy: 0.8092
Epoch 71/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0278 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6551 -
val_sparse_categorical_accuracy: 0.8092
Epoch 72/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0277 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6543 -
val_sparse_categorical_accuracy: 0.8092
Epoch 73/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0294 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6537 -
val_sparse_categorical_accuracy: 0.8092
Epoch 74/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0241 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6529 -
val_sparse_categorical_accuracy: 0.8092
Epoch 75/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0257 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6521 -
val_sparse_categorical_accuracy: 0.8092
Epoch 76/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0233 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6513 -
val_sparse_categorical_accuracy: 0.8150
Epoch 77/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0249 -
```

```
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6506 -
val_sparse_categorical_accuracy: 0.8150
Epoch 78/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0264 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6500 -
val_sparse_categorical_accuracy: 0.8150
Epoch 79/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0250 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6493 -
val_sparse_categorical_accuracy: 0.8150
Epoch 80/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0233 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6486 -
val_sparse_categorical_accuracy: 0.8150
Epoch 81/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0228 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6480 -
val_sparse_categorical_accuracy: 0.8150
Epoch 82/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0229 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6481 -
val_sparse_categorical_accuracy: 0.8150
Epoch 83/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0219 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6476 -
val_sparse_categorical_accuracy: 0.8150
Epoch 84/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0226 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6470 -
val_sparse_categorical_accuracy: 0.8150
Epoch 85/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0218 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6471 -
val_sparse_categorical_accuracy: 0.8150
Epoch 86/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0205 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6466 -
```

```
val_sparse_categorical_accuracy: 0.8150
Epoch 87/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0203 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6460 -
val_sparse_categorical_accuracy: 0.8150
Epoch 88/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0196 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6456 -
val_sparse_categorical_accuracy: 0.8150
Epoch 89/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0184 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6452 -
val_sparse_categorical_accuracy: 0.8150
Epoch 90/100
691/691 [==============================] - 1s 2ms/sample - loss: 0.0189 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6447 -
val_sparse_categorical_accuracy: 0.8150
Epoch 91/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0177 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6442 -
val_sparse_categorical_accuracy: 0.8150
Epoch 92/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0185 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6437 -
val_sparse_categorical_accuracy: 0.8150
Epoch 93/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0209 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6430 -
val_sparse_categorical_accuracy: 0.8150
Epoch 94/100
691/691 [==============================] - 1s 2ms/sample - loss: 0.0155 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6425 -
val_sparse_categorical_accuracy: 0.8150
Epoch 95/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0191 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6419 -
val_sparse_categorical_accuracy: 0.8150
```

```
Epoch 96/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0167 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6414 -
val_sparse_categorical_accuracy: 0.8150
Epoch 97/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0176 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6409 -
val_sparse_categorical_accuracy: 0.8150
Epoch 98/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0194 -
sparse_categorical_accuracy: 0.9986 - val_loss: 0.6406 -
val_sparse_categorical_accuracy: 0.8150
Epoch 99/100
691/691 [==============================] - 2s 3ms/sample - loss: 0.0156 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6402 -
val_sparse_categorical_accuracy: 0.8150
Epoch 100/100
691/691 [==============================] - 2s 2ms/sample - loss: 0.0163 -
sparse_categorical_accuracy: 1.0000 - val_loss: 0.6397 -
val_sparse_categorical_accuracy: 0.8150
```

[39]:
```
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                multiple                  4276944
_________________________________________________________________
dropout_3 (Dropout)          multiple                  0
_________________________________________________________________
gru_1 (GRU)                  multiple                  4464
_________________________________________________________________
dropout_4 (Dropout)          multiple                  0
_________________________________________________________________
simple_rnn_1 (SimpleRNN)     multiple                  444
_________________________________________________________________
```

```
dropout_5 (Dropout)          multiple                  0
_________________________________________________________________
dense_1 (Dense)              multiple                  39
=================================================================
Total params: 4,281,891
Trainable params: 4,281,891
Non-trainable params: 0
_________________________________________________________________
```

[40]:

```
from sklearn.metrics import accuracy_score
pre_y = model.predict_classes(test_xx)
acc = accuracy_score(test_y, pre_y)
acc
```

[40]: 0.815028901734104

[ ]: