import pandas as pd

from keras.models import Sequential

from keras.layers import Dense, Activation, Flatten

from keras.layers.embeddings import Embedding

from keras.layers.recurrent import LSTM

from keras.layers.normalization import BatchNormalization

from keras.utils import np\_utils

from keras.callbacks import ModelCheckpoint

from keras.layers.advanced\_activations import PReLU

from keras.preprocessing import sequence, text

from keras.layers import SpatialDropout1D

from keras import metrics

train = pd.read\_csv('/Users/yeezhianliew/Desktop/train.csv')

test = pd.read\_csv('/Users/yeezhianliew/Desktop/test.csv')

y\_train = train.label.values

y\_test = test.label.values

tk = text.Tokenizer(num\_words=200000)

train.link\_name = train.link\_name.astype(str)

test.link\_name = test.link\_name.astype(str)

train.textdata = train.textdata.astype(str)

test.textdata = test.textdata.astype(str)

max\_len = 80

tk.fit\_on\_texts(list(train.link\_name.values) + list(train.textdata.values) + list(test.link\_name.values) + list(

test.textdata.values))

x\_train\_title = tk.texts\_to\_sequences(train.link\_name.values)

x\_train\_title = sequence.pad\_sequences(x\_train\_title, maxlen=max\_len)

x\_train\_textdata = tk.texts\_to\_sequences(train.textdata.values)

x\_train\_textdata = sequence.pad\_sequences(x\_train\_textdata, maxlen=max\_len)

x\_test\_title = tk.texts\_to\_sequences(test.link\_name.values)

x\_test\_title = sequence.pad\_sequences(x\_test\_title, maxlen=max\_len)

x\_test\_textdata = tk.texts\_to\_sequences(test.textdata.values)

x\_test\_textdata = sequence.pad\_sequences(x\_test\_textdata, maxlen=max\_len)

word\_index = tk.word\_index

ytrain\_enc = np\_utils.to\_categorical(y\_train)

model = Sequential()

model.add(Embedding(len(word\_index) + 1, 300, input\_length=80, dropout=0.2),)

model.add(LSTM(300, return\_sequences=True, dropout=0.2, recurrent\_dropout=0.2))

model.add(Dense(200))

model.add(PReLU())

model.add(SpatialDropout1D(0.2))

model.add(BatchNormalization())

model.add(Dense(200))

model.add(PReLU())

model.add(SpatialDropout1D(0.2))

model.add(BatchNormalization())

model.add(Flatten())

model.add(Dense(2))

model.add(Activation('softmax'))

model.compile(loss='categorical\_crossentropy', optimizer='adam', metrics=['acc' ])

checkpoint = ModelCheckpoint('/Users/yeezhianliew/Desktop/weights.h5', monitor='val\_acc', save\_best\_only=True, verbose=2)

model.fit(x\_train\_title, y=ytrain\_enc,

batch\_size=128, epochs=20, verbose=2, validation\_split=0.1,

shuffle=True, callbacks=[checkpoint])