```
#!/bin/python3.5
# Ostap Voynarovskiy
# CGML HW5
# October 9 2018
# Professo Curro

import keras
import numpy as np
import pandas as pd
from keras.models import Sequential
from keras.preprocessing.text import Tokenizer
from keras.layers import Embedding, Dense, Flatten, Conv1D, Dropout
from sklearn.model_selection import train_test_split
from keras.preprocessing.sequence import pad_sequences
from keras import regularizers
#import scikit
# Read Data from CSV files header = none to not ignore first row
train = pd.read_csv("./ag_news/train.csv",header=None)
test = pd.read_csv("./ag_news/test.csv",header=None)

#label the collumns
train.columns    = ['cat','title','description']
test.columns     = ['cat','title','description']

train['space'] = " "
train["text"] = train.title + train.space + train.description
test['space'] = " "
test["text"] = test.title + test.space + test.description

# split the data apart for val
train, val = train_test_split(train,test_size=.05)

# Create a Tokenizer instance
tokenizer = Tokenizer()
# Only  fit on text this for training data
tokenizer.fit_on_texts(train.text)

# Create sequenced data for the vocabs
trainSeq = tokenizer.texts_to_sequences(train.text)
valSeq   = tokenizer.texts_to_sequences(val.text)
testSeq  = tokenizer.texts_to_sequences(test.text)

# Pad the vocabs so that they are all the same len
lenPad=185
trainPad= pad_sequences(trainSeq,maxlen=lenPad)
valPad= pad_sequences(valSeq,maxlen=lenPad)
testPad= pad_sequences(testSeq,maxlen=lenPad)

# how long to make the embedding vector
embeddingVectorLen = 32
vocabLength = len(tokenizer.word_index)

# one hot encode the labels
train_cat = np.array(train.cat-1)
train_cat = train_cat.reshape(train_cat.shape[0],1)
train_cat = keras.utils.to_categorical(train_cat,num_classes=4)

val_cat = np.array(val.cat-1)
val_cat = val_cat.reshape(val_cat.shape[0],1)
val_cat = keras.utils.to_categorical(val_cat,num_classes=4)

test_cat = np.array(test.cat-1)
test_cat = test_cat.reshape(test_cat.shape[0],1)
test_cat = keras.utils.to_categorical(test_cat,num_classes=4)


model = Sequential()
model.add(Embedding(vocabLength+1, embeddingVectorLen, input_length=lenPad))
#model.add(Conv1D(8,32,padding = "same",kernel_regularizer=regularizers.l2()))
```



```
#model.add(Conv1D(128,32,padding = "same",dilation_rate=3,kernel_regularizer=reg
ularizers.l2()))

model.add(Dropout(.7))
model.add(Flatten())
model.add(Dense(4,activation="softmax"))

model.compile(loss=keras.losses.categorical_crossentropy,
        optimizer=keras.optimizers.Adam(),metrics=['accuracy'])

model.fit(trainPad, train_cat,
        batch_size=64,
        epochs = 2,
        verbose = 1,
        validation_data =(valPad,val_cat))

score = model.evaluate(testPad,test_cat,verbose= 1)
print("Test loss:",score[0])
print("Test accuracy:",score[1])
```