ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC BÁCH KHOA

KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH

KỸ THUẬT DỮ LIỆU

Bài tập lớn

Sử dụng CNN, LSTM và kết hợp để thực hiện sentiment analysis

Sinh viên thực hiện: Phạm Trần Đức Trọng - 2270519

Ho Chi Minh, 4-2023

Mục lục

1 Chuẩn bị dữ liệu

Trong bài tập lớn này tôi Sử dụng tập dữ liệu VLSP để thực hiện việc train và test cho các mô hình CNN, LSTM và kết hợp của 2 mô hình

2 Tiền xử lý dữ liệu

Thực hiện clean data đối với các loại data có những ký tự đặc biệt, emoji, xóa các từ bị duplicate, khoảng trắng...

```
def text_cleaner(text):
    # lower case text
    text = text.lower()
    # remove emojis
    text = re.sub(emoji_pattern, " ", text)
    # reduce repeated characters
    text = re.sub(r'([a-z]+?)\1+',r'\1', text)
    # remove multiple spaces
    text = re.sub(r'\s+', ' ', text)
    # make sure punctuation is in the right letter
    text = text_normalize(text)
    # remove punctuations
    text = text.translate(str.maketrans('', '', string.punctuation))
    return text
```

Listing 1: Text cleaner

Lấy ngẫu nhiên 5 câu trong tập dữ liệu để xem kết quả:

1. mua phê vãi hộp viên viên pin đồng hồ.
2. gờm thể xứng tầm Surface Pro Chất Full HD Thiết kế màn hình viền to thời Pin đối thiết dạng Tablet Pin

3. chí thua mấy Ultrabook Dell XPS Vaio T gọi Tablet lai Tablet h pin tối

4. Note hóng nè chấn Note

5. Ram màn hình độ phân giải sản phẩm Apple thiếp Ram chạy ok nâng sợ tốn tiền thiết lắm Cá nhân S ram G Iphone lắm Ram Iphone nâng chạy ISO Beta trơn tru Apple ko chạy đua cứng tiêng nâng Ram màn hình dung

3 LSTM

```
def lstm_model():
    x = embedding_layer(input_layer)
    x = Bidirectional(LSTM(128, return_sequences=True, dropout=0.2,
    recurrent_dropout=0.2))(x)
    x = Dropout(0.2)(x)
    x = Bidirectional(LSTM(128, return_sequences=True, dropout=0.2,
    recurrent_dropout=0.2))(x)
    x = Dropout(0.2)(x)
    x = Bidirectional(LSTM(128, return_sequences=True, dropout=0.2,
    recurrent_dropout=0.2))(x)
    x = Dropout(0.2)(x)
    x = GlobalMaxPool1D()(x)
    x = Dense(128, activation="relu")(x)
    x = Dropout(0.2)(x)
    x = Dense(3, activation="softmax")(x)
    model = Model(inputs=input_layer, outputs=x)
    model.compile(loss='binary_crossentropy', optimizer=adam, metrics=['accuracy'
    ])
    model.summary()
    return model
```

Listing 2: lstm model

4 CNN

```
def cnn_model():
    from keras.models import Model
    from keras.layers import *
    from keras.callbacks import EarlyStopping
    from tensorflow.keras.optimizers import Adam
    from keras.models import Model
    from keras import regularizers
    sequence_length = data.shape[1]
    filter_sizes = [3,4,5]
    num_filters = 100
    drop = 0.5

    inputs = Input(shape=(sequence_length,))
    embedding = embedding_layer(inputs)

    ################### LSTM ONLY ###############################
    reshape = Reshape((sequence_length,EMBEDDING_DIM))(embedding)

    ################## SINGLE LSTM ####################
    lstm_0 = LSTM(512)(reshape)

    # YOU WANNA ADD MORE LSTM LAYERS? UNCOMMENT THIS #
    lstm_2 = LSTM(1024, return_sequences=True)(reshape)
    lstm_1 = LSTM(512, return_sequences=True)(lstm_2)
    lstm_0 = LSTM(256)(lstm_1)

    ###########################################################


```

```
    ################### CRNN ###################################
    reshape = Reshape((sequence_length,EMBEDDING_DIM))(embedding)

    conv_0 = Conv1D(num_filters, (filter_sizes[0], ),padding="same",activation='
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)
    conv_1 = Conv1D(num_filters, (filter_sizes[1], ),padding="same",activation='
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)
    conv_2 = Conv1D(num_filters, (filter_sizes[2], ),padding="same",activation='
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)

    conv_0 = MaxPool1D(300)(conv_0)
    conv_1 = MaxPool1D(300)(conv_1)
    conv_2 = MaxPool1D(300)(conv_2)
    # Reshape output to match RNN dimension
    conv_0 = Reshape((-1, num_filters))(conv_0)
    conv_1 = Reshape((-1, num_filters))(conv_1)
    conv_2 = Reshape((-1, num_filters))(conv_2)

    concat = concatenate([conv_0, conv_1, conv_2])
    concat = Flatten()(concat)

    dropout = Dropout(drop)(concat)
    output = Dense(units=3, activation='softmax',kernel_regularizer=regularizers.
    l2(0.01))(dropout)
```

Listing 3: cnn model

5 Kết hợp CNN và LSTM

```
def lstm_cnn_model():
    from keras.models import Model
```

```
from keras.layers import *
from keras.callbacks import EarlyStopping
from tensorflow.keras.optimizers import Adam
from keras.models import Model
from keras import regularizers
sequence_length = data.shape[1]
filter_sizes = [3,4,5]
num_filters = 100
drop = 0.5

inputs = Input(shape=(sequence_length,))
embedding = embedding_layer(inputs)

################### LSTM ONLY ###################################
reshape = Reshape((sequence_length,EMBEDDING_DIM))(embedding)

################## SINGLE LSTM ####################
lstm_0 = LSTM(512)(reshape)

# YOU WANNA ADD MORE LSTM LAYERS? UNCOMMENT THIS #
lstm_2 = LSTM(1024, return_sequences=True)(reshape)
lstm_1 = LSTM(512, return_sequences=True)(lstm_2)
lstm_0 = LSTM(256)(lstm_1)

#################################################################


################### CRNN ##################################
reshape = Reshape((sequence_length,EMBEDDING_DIM))(embedding)

conv_0 = Conv1D(num_filters, (filter_sizes[0], ),padding="same",activation='
```

```
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)
34  conv_1 = Conv1D(num_filters, (filter_sizes[1], ),padding="same",activation='
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)
35  conv_2 = Conv1D(num_filters, (filter_sizes[2], ),padding="same",activation='
    relu',kernel_regularizer=regularizers.l2(0.01))(reshape)
36
37  conv_0 = MaxPool1D(300)(conv_0)
38  conv_1 = MaxPool1D(300)(conv_1)
39  conv_2 = MaxPool1D(300)(conv_2)
40  # Reshape output to match RNN dimension
41  conv_0 = Reshape((-1, num_filters))(conv_0)
42  conv_1 = Reshape((-1, num_filters))(conv_1)
43  conv_2 = Reshape((-1, num_filters))(conv_2)
44
45  concat = concatenate([conv_0, conv_1, conv_2])
46  concat = Flatten()(concat)
47
48  dropout = Dropout(drop)(concat)
49  output = Dense(units=3, activation='softmax',kernel_regularizer=regularizers.
    l2(0.01))(dropout)
```

Listing 4: lstm cnn model