ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

DEEP LEARNING TRONG KHOA HỌC DỮ LIỆU
LAB 6

ỨNG DỤNG MẠNG NEURAL HỒI QUY

Họ và tên : Lưu Quang Tiến Hoàng
MSSV : 20521342
Lớp : DS201.N11

Mục tiêu

- Ôn tập kiến thức cơ bản về mô hình mạng neural hồi quy (RNN).
- Biết cách xây dựng và sử dụng mô hình mạng neural hồi quy để giải quyết một số bài toán thường gặp trong lĩnh vực Xử lý ngôn ngữ tự nhiên.

1. BÀI TOÁN PHÂN TÍCH CẢM XÚC (SENTIMENT ANALYSIS)

Mô tả bài toán

Trong giáo dục, các tổ chức giáo dục sẽ thu thập các phản hồi của người học sau một khoá học hoặc môn học để làm căn cứ nâng cao chất lượng giảng dạy và đào tạo. Một trong các căn cứ đó là mức độ phản hồi tích cực hay tiêu cực của người học.

Giới thiệu bộ dữ liệu

- Bộ dữ liệu: **UIT-VSFC** [1].
- Số lượng dữ liệu: Khoảng hơn 16,000 câu.
- Số lượng nhãn: 3 nhãn, gồm tích cực (positive), tiêu cực (negative) và trung tính (neural).
- Link tải: https://drive.google.com/drive/folders/1xclbjHHK58zk2X6iqbvMPS2rcy9y9E0X.

Tổ chức dữ liệu

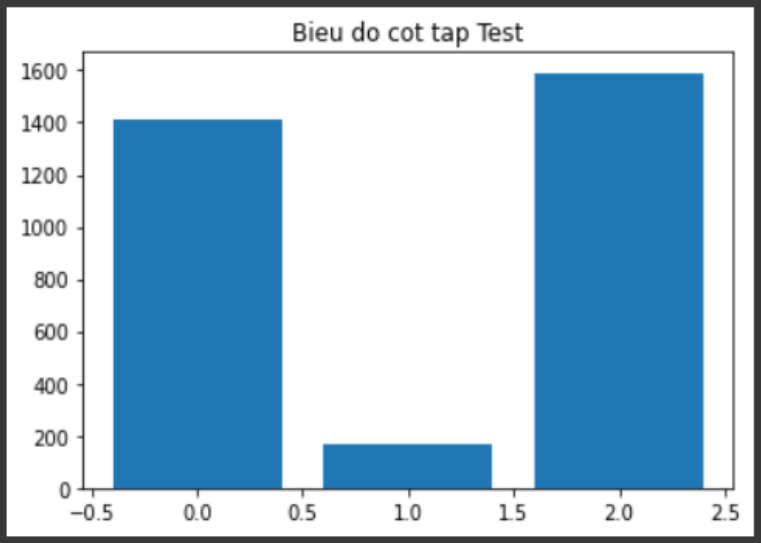
Dữ liệu được tổ chức dưới dạng các file .txt (đọc thêm file README để rõ hơn về cấu trúc và thông tin trong bộ dữ liệu).

Đọc dữ liệu

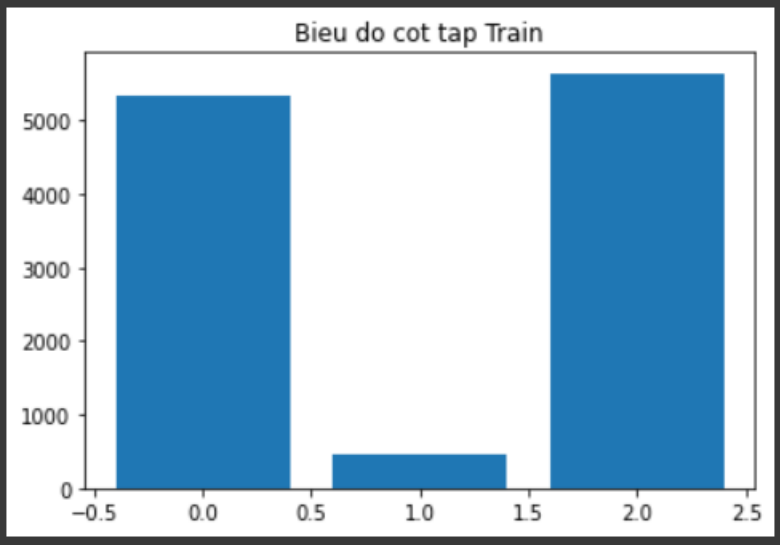
Đọc các tập dữ liệu train, dev, test từ file.

(?) Vẽ biểu đồ cột thống kê số lượng nhãn trong mỗi tập dữ liệu (mỗi tập dữ liệu vẽ 1 biểu đồ)?

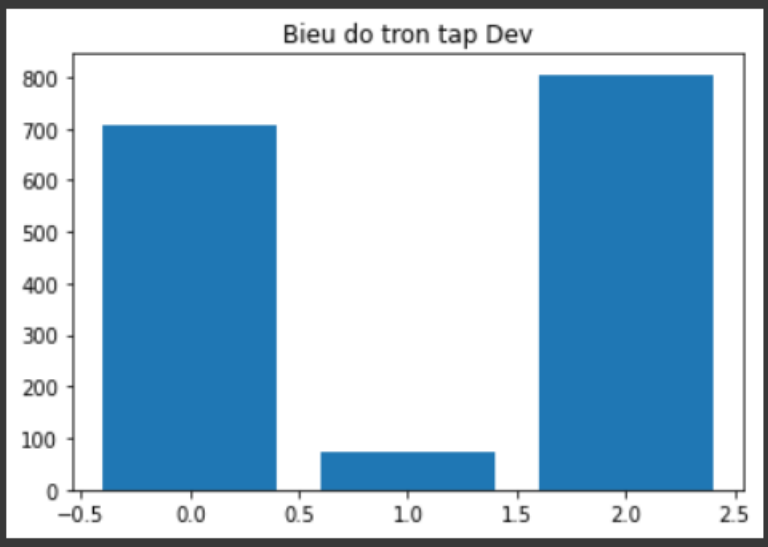
```
[20] plt.bar(list_label2.keys(),list_label2.values())
     plt.title("Bieu do cot tap Test")
     plt.show()
```




```
plt.bar(list_label1.keys(),list_label1.values())
plt.title("Bieu do cot tap Train")
plt.show()
```



```
plt.bar(list_label3.keys(), list_label3.values())
plt.title("Bieu do tron tap Dev")
plt.show()
```



Word embedding

Sử dụng các bộ **word embedding đã được huấn luyện sẵn** (pre-trained) thay vì phải đi xây dựng lại lớp Embedding dựa trên dữ liệu. Việc này sẽ giúp tiết kiệm thời gian huấn luyện và tăng khả năng nhận diện ngữ nghĩa trong câu do các bộ word embedding được huấn luyện sẵn đã được xây dựng dựa trên các tập dữ liệu rất lớn trước đó, nên khả năng nhận diện ngữ nghĩa sẽ rất cao và đa dạng.

Trong bài thực hành này, chúng ta sử dụng bộ PhoW2V của tác giả Nguyen và các cộng sự [2]. Link tải: https://github.com/datquocnguyen/PhoW2V.

Bộ word embedding này có 2 phiên bản: 100 chiều và 300 chiều, được huấn luyện trên mức độ **từ** và mức độ **tiếng**. Chúng ta sẽ sử dụng bộ 100 chiều được huấn luyện trên mức độ từ.

(?) Sử dụng lệnh summary để xem cấu trúc của mô hình đã xây dựng?

```
[112] model.summary()

Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_3 (Embedding)     (None, 100, 100)          158750800

 bidirectional_4 (Bidirectio  (None, 400)              481600
 nal)

 dense_3 (Dense)             (None, 3)                 1203

=================================================================
Total params: 159,233,603
Trainable params: 482,803
Non-trainable params: 158,750,800
_________________________________________________________________
```

(?) Huấn luyện mô hình với epochs=10 và batch_size=128?

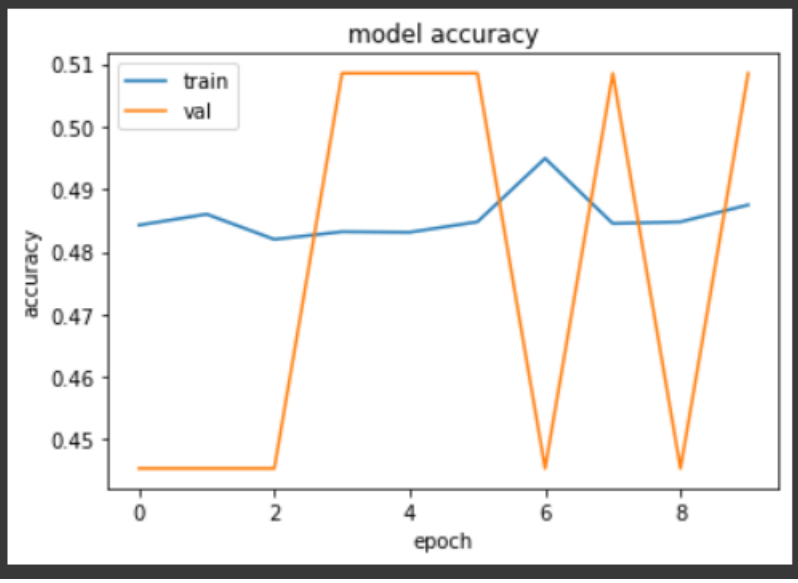
```
[114] history= model.fit(X_train_encoded, y_train_encoded, batch_size=128,epochs=10,validation_data=(X_dev_encoded, y_dev_encoded))

Epoch 1/10
90/90 [==============================] - 6s 33ms/step - loss: 0.9211 - accuracy: 0.4842 - val_loss: 0.8513 - val_accuracy: 0.4454
Epoch 2/10
90/90 [==============================] - 2s 23ms/step - loss: 0.8364 - accuracy: 0.4860 - val_loss: 0.8534 - val_accuracy: 0.4454
Epoch 3/10
90/90 [==============================] - 2s 23ms/step - loss: 0.8372 - accuracy: 0.4820 - val_loss: 0.8561 - val_accuracy: 0.4454
Epoch 4/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8362 - accuracy: 0.4832 - val_loss: 0.8464 - val_accuracy: 0.5085
Epoch 5/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8363 - accuracy: 0.4831 - val_loss: 0.8462 - val_accuracy: 0.5085
Epoch 6/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8354 - accuracy: 0.4848 - val_loss: 0.8463 - val_accuracy: 0.5085
Epoch 7/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8348 - accuracy: 0.4949 - val_loss: 0.8538 - val_accuracy: 0.4454
Epoch 8/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8373 - accuracy: 0.4845 - val_loss: 0.8475 - val_accuracy: 0.5085
Epoch 9/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8351 - accuracy: 0.4848 - val_loss: 0.8489 - val_accuracy: 0.4454
Epoch 10/10
90/90 [==============================] - 2s 24ms/step - loss: 0.8359 - accuracy: 0.4875 - val_loss: 0.8497 - val_accuracy: 0.5085
```

(?) Vẽ đồ thị học với Accuracy và Loss?

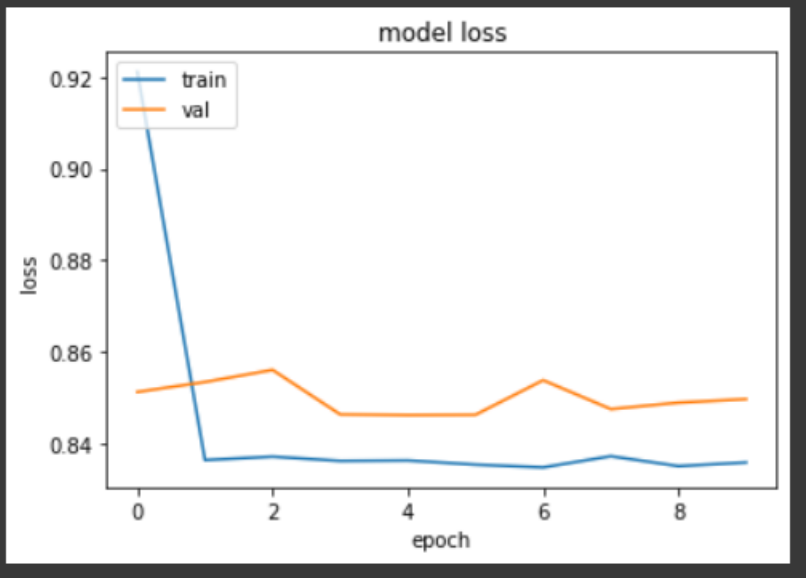
Đồ thị Accuracy

```
[117] plt.plot(history.history['accuracy'])
      plt.plot(history.history['val_accuracy'])
      plt.title('model accuracy')
      plt.ylabel('accuracy')
      plt.xlabel('epoch')
      plt.legend(['train','val'], loc = 'upper left')
      plt.show()
```



Đồ thị Loss

```
[118] plt.plot(history.history['loss'])
      plt.plot(history.history['val_loss'])
      plt.title('model loss')
      plt.ylabel('loss')
      plt.xlabel('epoch')
      plt.legend(['train','val'], loc = 'upper left')
      plt.show()
```



(?) Đánh giá độ chính xác của mô hình bằng độ đo Accuracy?

```
[115] y_pred = model.predict(X_test_encoded)
      y_pred_label = np.argmax(y_pred,axis=-1)

      99/99 [==============================] - 1s 7ms/step

[116] from sklearn.metrics import accuracy_score
      accuracy_score(y_test, y_pred_label)*100

      50.22109917877447
```

NX: Ta có thể thấy độ chính xác chưa cao chỉ đạt tầm 50.22%

2. BÀI TOÁN NHẬN DIỆN THỰC THỂ CÓ TÊN (NAMED ENTITIES RECOGNITION)

Mô tả bài toán

- Trong các đoạn văn bản, người ta có nhu cầu trích xuất ra các thông tin về các thực thể có tên như: tên người, tên tổ chức, tên địa danh.
- Đối với đại dịch COVID, các như cầu trích xuất ra các thông tin về người bệnh, độ tuổi, giới tính, triệu chứng,... là rất quan trọng, góp phần vào công tác phòng chống dịch bệnh COVID trong cộng đồng. Bài toán rút trích các thực thể liên quan đến COVID 19 như trên được gọi là bài toán nhận diện thực thể có tên (Named Entity Recognition - NER).

Giới thiệu bộ dữ liệu

- Bộ dữ liệu: **PhoNER COVID 19** [4].
- Số lượng dữ liệu: Khoảng 10,000 câu.
- Bộ dữ liệu này có 2 mức độ: tiếng (syllables) và từ (word).
- Link tải: https://github.com/VinAIResearch/PhoNER_COVID19.
- Bộ nhãn (tag):
  - PATIENT_ID: Mã bệnh nhân.
  - PERSON_NAME: Tên bệnh nhân hoặc người liên hệ với bệnh nhân.
  - AGE: Tuổi bệnh nhân hoặc người liên hệ với bệnh nhân.
  - GENDER: Giới tính bệnh nhân hoặc người liên hệ với bệnh nhân.
  - OCCUPUTATION: Nghề nghiệp bệnh nhân.
  - ORGANIZATION: Tổ chức liên quan đến bệnh nhân.
  - SYMTOM&DISEASE: Triệu chứng của bệnh nhân hoặc bệnh nền trước đó.
  - TRANSPORTATION: Phương tiện di chuyển của bệnh nhân.
  - DATE: Ngày xuất hiện trong văn bản.

Chúng ta vẫn sử dụng bộ word embedding đã huấn luyện sẵn PhoW2V [2] trên mức độ từ cho bộ dữ liệu này. Do bộ dữ liệu này đã chia mức độ sẵn nên ta không cần tách từ.

- **Đọc dữ liệu**

  Bộ dữ liệu này được lưu trữ theo cấu trúc của bộ CoNLL 2003 [5].

(?) Sử dụng lệnh summary để xem cấu trúc của mô hình đã xây dựng?

```
[128] model1.summary()

      Model: "sequential_4"
      _________________________________________________________________
       Layer (type)                Output Shape              Param #
      =================================================================
       embedding_4 (Embedding)     (None, 100, 100)          158751000

       bidirectional_5 (Bidirectio  (None, 100, 1024)        2510848
       nal)

      =================================================================
      Total params: 161,261,848
      Trainable params: 2,510,848
      Non-trainable params: 158,751,000
      _________________________________________________________________
```

(?) Huấn luyện mô hình với epochs=3 và batch_size=128?

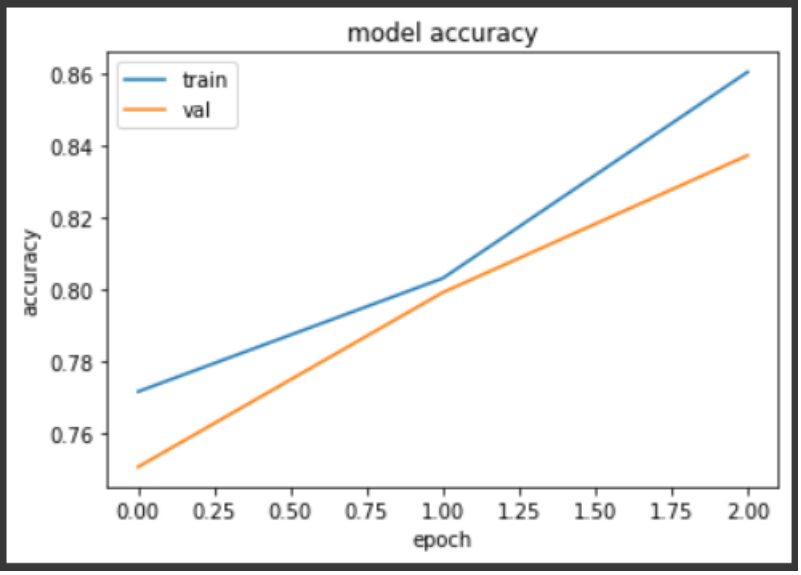
```
[131] history1= model1.fit(X_train, y_train, batch_size=128,epochs=3,validation_data=(X_dev, y_dev))

      Epoch 1/3
      40/40 [==============================] - 46s 1s/step - loss: 0.3010 - accuracy: 0.7714 - val_loss: 0.2920 - val_accuracy: 0.7504
      Epoch 2/3
      40/40 [==============================] - 40s 989ms/step - loss: 0.2064 - accuracy: 0.8030 - val_loss: 0.2061 - val_accuracy: 0.7991
      Epoch 3/3
      40/40 [==============================] - 41s 1s/step - loss: 0.1410 - accuracy: 0.8606 - val_loss: 0.1691 - val_accuracy: 0.8373
```

(?) Vẽ đồ thị học với Accuracy và Loss?

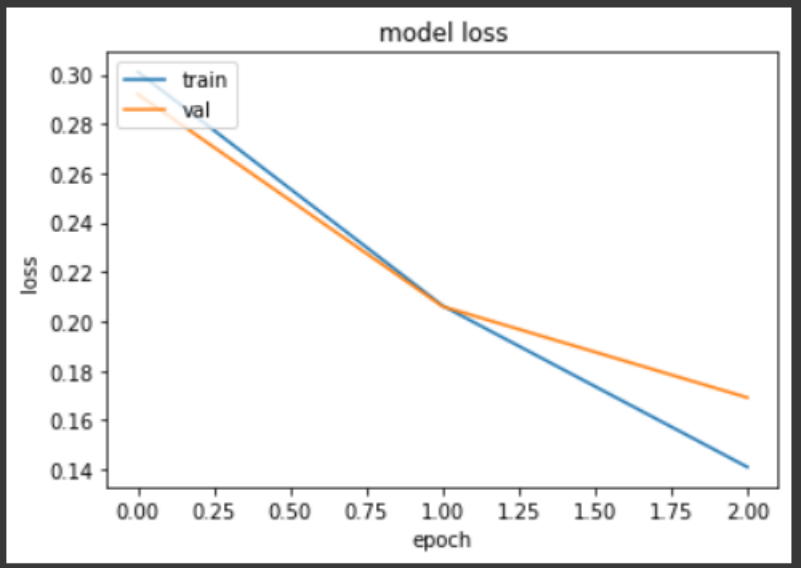
Đồ thị Accuracy

```
[132] plt.plot(history1.history['accuracy'])
      plt.plot(history1.history['val_accuracy'])
      plt.title('model accuracy')
      plt.ylabel('accuracy')
      plt.xlabel('epoch')
      plt.legend(['train','val'], loc = 'upper left')
      plt.show()
```



Đồ thị Loss

```
[133] plt.plot(history1.history['loss'])
      plt.plot(history1.history['val_loss'])
      plt.title('model loss')
      plt.ylabel('loss')
      plt.xlabel('epoch')
      plt.legend(['train','val'], loc = 'upper left')
      plt.show()
```



- **Đánh giá mô hình**
  Sử dụng độ đo đánh giá = tỉ lệ dự đoán đúng tên thực thể trên tổng số câu.

```
[134] y_pred1 = model1.predict(X_test_encoded)
      y_pred_label1 = np.argmax(y_pred1,axis=-1)

      99/99 [==============================] - 10s 92ms/step

[135] import numpy as np
      from sklearn.metrics import accuracy_score
      acc = []

      for i in range(0, len(y_test)):
       acc.append(accuracy_score(y_test[i], y_pred_label1[i]))

      np.mean(acc)*100

      91.62866666666667
```

NX: Ta thấy được rằng độ chính xác của mô hình rất cao đạt khoảng 91.63%