DS201 - Deep Learning trong Khoa học dữ liệu

Lab 1. LÀM QUEN VỚI MẠNG NEURAL CƠ BẢN

MSSV: 20521196

Họ tên: Nguyễn Mạnh Đức

Lớp: DS201.N11.1

(?) Mô hình mạng neural trong hình trên có những loại layer nào ?

⇨ Input layer, Hidden layer, Output layer

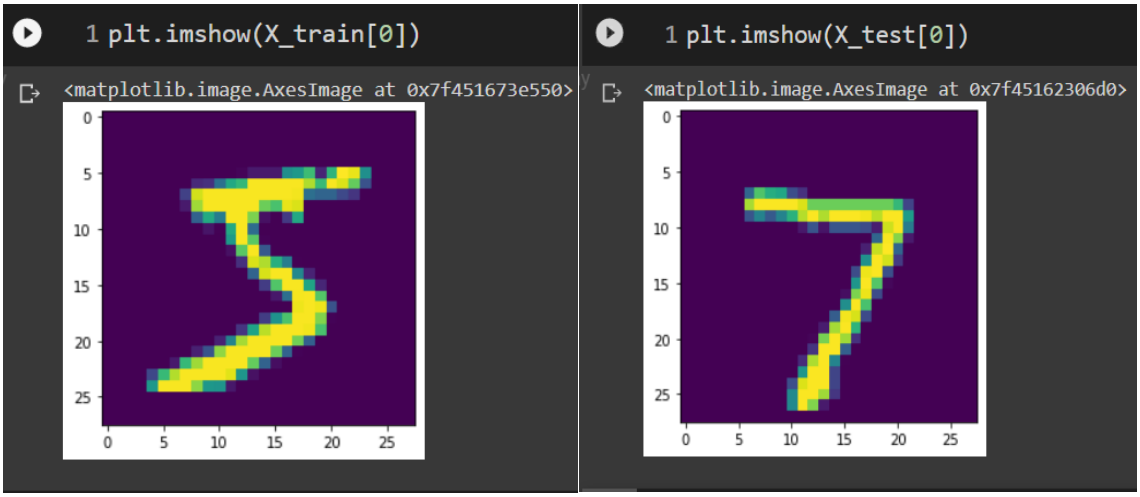
1. LOAD BỘ DỮ LIỆU

(?) MNIST là viết tắt của cụm từ nào ?

⇨ Modified National Institute of Standards and Technology

2. CHUẨN BỊ DỮ LIỆU

(?) Sử dụng thư viện Matplotlib để trực quan hóa ảnh train[0] và ảnh test[0] ?



(?) Hãy khảo sát bộ dữ liệu MNIST do thư viện Keras cung cấp:

```
[5]  1 X_train.shape, X_test.shape

     ((60000, 28, 28), (10000, 28, 28))
```

- Xác định số lượng ảnh của tập train và tập test ?

⇨ Tập train: 60.000 ảnh Tập test: 10.000 ảnh

- Tính tỉ lệ train : test ?

  ⇨ Train : Test là 60.000:10.000 = 6:1

- Xác định kích thước của mỗi ảnh trong tập train và tập test ?

  ⇨ Kích thước mỗi ảnh: 28 x 28

- Xác định số phần tử ảnh của mỗi ảnh trong tập train và tập test ?

  ⇨ Số phần từ ảnh của mỗi ảnh: 784

(?) Kiểu dữ liệu set có những tính chất gì ?

⇨ Các phần tử của tập hợp được đặt trong cặp ngoặc nhọn {} các phần tử phân cách bởi dấu phẩy " , "
⇨ Mỗi phần tử chỉ xuất hiện 1 lần cho dù nhập nhiều lần
⇨ Kiểu tập hợp set cũng có các phép lấy hợp, lấy giao như trong toán học
⇨ Để tạo 1 tập s rỗng, cần dùng hàm không có thông số s = set()

(?) Xác định số lượng nhãn và liệt kê các nhãn có trong tập train ?

```
1 set(y_train)
```

```
{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}
```

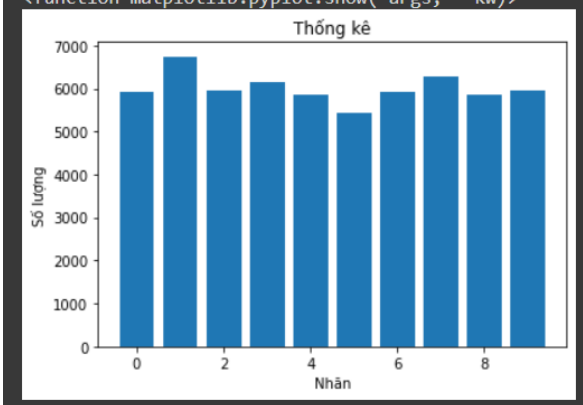
⇨ 10 nhãn. Gồm 0, 1, 2, 3, 4, 5, 6, 7, 8, 9

(?) Sử dụng thư viện Matplotlib để vẽ các biểu đồ sau:

- Biểu đồ cột thể hiện sự phân bố các nhãn có trong tập train ?

```
1 plt.bar(list_labels.keys(), list_labels.values())
2 plt.xlabel("Nhãn")
3 plt.ylabel("Số lượng")
4 plt.title("Thống kê")
5 plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```



● Biểu đồ tròn thể hiện tỉ lệ các nhãn có trong tập train ?

```
plt.pie(x = list_labels.values(), labels = list_labels.keys(),
        startangle = 90)
plt.title("Thống kê")
plt.show
```

```
<function matplotlib.pyplot.show(*args, **kw)>
```

(?) Sau khi thực hiện thao tác reshape trên tập train:

● Mỗi điểm dữ liệu là một vector có số chiều là bao nhiêu ?

⇨ 784

● Các số -1 và 784 có ý nghĩa gì ?

⇨ -1: Lấy tất cả (có thể thay thành 60.000) ; 784: Vì mỗi bức ảnh là 1 mảng 28x28, đưa về dạng vector 1x784

● Viết ra dạng của input X = … ?

⇨ Là 1 mảng 2 chiều có 60.000 hàng và 784 cột

(?) Điền từ hoặc số thích hợp vào chỗ trống trong những câu dưới đây:

● Đầu ra của dữ liệu sẽ có giá trị trong khoảng 0 → 9 (tương ứng với (1) … chữ số viết tay).

⇨ Hình ảnh

● Mỗi điểm dữ liệu sẽ được phân loại vào một trong (2) … lớp tương ứng.

⇨ 10

● Như vậy, đây là một bài toán phân lớp đa lớp ((3)... classification).

⇨ Multiple

(?) Sau khi thực hiện thao tác trên:

● Xác định kích thước của y_train ? Các số liệu này tương ứng với những đại lượng nào?

```
1 y_train.shape
(60000, 10)
```

⇨ 60.000 hình ảnh, 10 là số lượng lớp

● Viết ra vector tương ứng y_train[5] ?

```
1 y_train[5]
array([0., 0., 1., 0., 0., 0., 0., 0., 0., 0.], dtype=float32)
```

● Các giá trị "1" và "0" trong mỗi vector có ý nghĩa gì ?

⇨ Số thứ tự có giá trị 1 chính là nhãn, còn lại sẽ là 0

3. XÂY DỰNG MẠNG NEURAL BẰNG KERAS

(?) Kể tên và ghi công thức của một số hàm kích hoạt mà bạn biết ?

⇨ Hàm Sigmoid: $\frac{1}{1+e^{-x}}$

⇨ Tanh: $\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$

⇨ ReLU: $f(x) = \max(0, x)$

(?) Xác định kích thước đầu ra (output_shape) của lớp vừa được thêm vào mô hình ?

⇨ Số thứ tự có giá trị 1 chính là nhãn, còn lại sẽ là 0

(?) Tại sao số units ở lớp này lại là 10 ?

⇨ Bộ dữ liệu có tổng cộng 10 nhãn

(?) Tổng xác suất của các lớp bằng bao nhiêu ?

⇨

(?) Hai layers trong mô hình trên tương ứng với những loại layer nào trong mô hình mạng neural tổng quát ?

⇨ Input layer và Output layer

(?) Quan sát kết quả thực thi câu lệnh, cho biết số lượng tham số của mỗi layer và tổng số lượng tham số (total params) ?

⇨ Dense_6: 615.440 Dense_7: 7850 Total params: 623.390

4. HUẤN LUYỆN MÔ HÌNH

(?) Vì sao cần set giá trị learning_rate nhỏ ?

⇨ Độ lớn của learning rate sẽ ảnh hưởng trực tiếp tới tốc độ hội tụ của hàm loss tới điểm cực trị toàn cục. Trường hợp Learning rate quá lớn, thuật toán sẽ học nhanh, thuật toán bị dao động xung quanh hoặc thậm chí nhảy qua điểm cực tiểu.

(?) Viết công thức tính loss Cross Entropy cho bài toán phân lớp nhị phân "cat vs. non-cat" với hai nhãn là y = 1 (cat) và y = 0 (non-cat) (Chú thích các đại lượng trong công thức) ?

⇨ $Loss\ function: L(\hat{y}, y) = -(y.\log(\hat{y}) + (1-y).\log(1-\hat{y}))$

⇨ $Cross\ Entropy\ Cost\ function: \frac{1}{m}\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$

y: Nhãn (có thể nhận y = 1(cat) hoặc y = 0 (non-cat))

$\hat{y}$: Giá trị dự đoán từ mô hình

m: Số lượng dữ liệu

(?) Tính thời gian huấn luyện mô hình ?

```
CPU times: user 17.5 s, sys: 1.68 s, total: 19.2 s
Wall time: 20.9 s
<keras.callbacks.History at 0x7f447838ffd0>
```

5. ĐÁNH GIÁ MÔ HÌNH

(?) Kết quả dự đoán của mô hình sẽ là một ma trận có kích thước (a, b).

● Hãy xác định các giá trị a, b ?

⇨ Trên tập test: a = 10.000, b = 10

● Nêu ý nghĩa của các giá trị a, b ?

⇨ a = 10.000: Có 10.000 dự đoán tương ứng với 10.000 bức ảnh dữ liệu tập test

⇨ b = 10: Có 10 nhãn nên mỗi dự đoán 1 bức ảnh sẽ là vector 1x10

(?) Lớp được chọn sẽ có xác suất cao nhất hay thấp nhất ?

⇨ Cao nhất

(?) Dựa vào kết quả thực thi:

● Viết ra "vector xác suất dự đoán" của ảnh có chỉ số [50] và cho biết kết quả dự đoán ảnh này thuộc lớp nào ?

```
1 y_pred[50]

array([ -1677.2532, -46206.086 , -13379.435 ,  -4038.6448, -39152.746 ,
        -26890.268 , -27486.594 ,  35958.215 , -21782.193 ,  -1176.1626],
      dtype=float32)
```

● So sánh nhãn dự đoán (y_pred_label) và nhãn đúng (y_test) của ảnh này?

```
1 y_pred_label[50] == y_test[50]
```

```
False
```

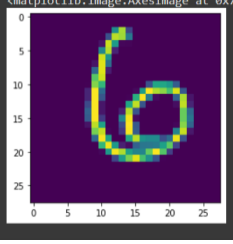
● Xác định nhãn phân loại của các ảnh có chỉ số [100], [1001],[4123] trong tập test ? Trực quan hóa các ảnh này để kiểm nghiệm lại kết quả dự đoán ?

```
1 y_pred_label[100], y_pred_label[1001], y_pred_label[4123]
```

```
(7, 7, 7)
```

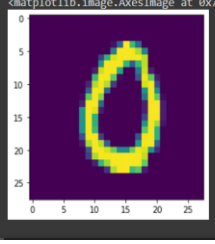
```
1 plt.imshow(X_test[100])
```

```
<matplotlib.image.AxesImage at 0x7f447a55f5d0>
```



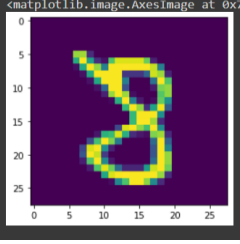
```
1 plt.imshow(X_test[1001])
```

```
<matplotlib.image.AxesImage at 0x7f447a562750>
```



```
1 plt.imshow(X_test[4123])
```

```
<matplotlib.image.AxesImage at 0x7f44784e15d0>
```



(?) Độ chính xác của mô hình là bao nhiêu ?

```
1 from sklearn.metrics import accuracy_score
2 accuracy_score(y_test, y_pred_label)*100
```

```
15.8
```

⇨ 15.8%

6. LƯU MÔ HÌNH

(?) Việc lưu lại mô hình có tác dụng gì ?

⇨ Không cần tốn thời gian đào tạo lại mô hình từ trước vì các mô hình học sâu thường tốn rất nhiều thời gian.

BÀI TẬP

BÀI TẬP 1 Xây dựng lại mô hình mạng neural, trong đó sử dụng hàm kích hoạt Sigmoid cho cả hai lớp, và xem kết quả.

```
model_1 = Sequential()
model_1.add(Dense(784, input_shape=(784, ),activation='sigmoid'))
model_1.add(Dense(10, input_shape=(10, ), activation='sigmoid'))
optimizer = Adam(learning_rate=0.01)
loss1 = BinaryCrossentropy()
model_1.compile(optimizer=optimizer, loss=loss1, metrics=['accuracy'])
model_1.fit(X_train, y_train, batch_size=128, epochs=10)

y_pred_1 = model_1.predict(X_test)
y_pred_label_1 = np.argmax(y_pred_1, axis=-1)
print("")
print("")
print("Kết quả: ")
accuracy_score(y_test, y_pred_label_1)*100
```

KẾT QUẢ: 87.57%

```
Kết quả:
87.57000000000001
```

BÀI TẬP 2 Xây dựng lại mô hình mạng neural, trong đó sử dụng hàm kích hoạt ReLU cho lớp thứ nhất và Sigmoid cho lớp đầu ra, và xem kết quả.

```
[50] model_2 = Sequential()
     model_2.add(Dense(784, input_shape=(784, ),activation='relu'))
     model_2.add(Dense(10, input_shape=(10, ), activation='sigmoid'))
     optimizer = Adam(learning_rate=0.01)
     loss1 = BinaryCrossentropy()
     model_2.compile(optimizer=optimizer, loss=loss1, metrics=['accuracy'])
     model_2.fit(X_train, y_train, batch_size=128, epochs=10)

     y_pred_2 = model_2.predict(X_test)
     y_pred_label_2 = np.argmax(y_pred_2, axis=-1)
     print("")
     print("")
     print("Kết quả: ")
     accuracy_score(y_test, y_pred_label_2)*100
```

KẾT QUẢ: 88.53%

```
Kết quả:
88.53
```

BÀI TẬP 3

- **Hãy thực hiện lại mô hình ở Bài tập 2 trên dữ liệu X_train và X_test đã được chuẩn hoá.**

● **So sánh độ chính xác của mô hình trong hai trường hợp: không chuẩn hóa và có chuẩn hóa ?**

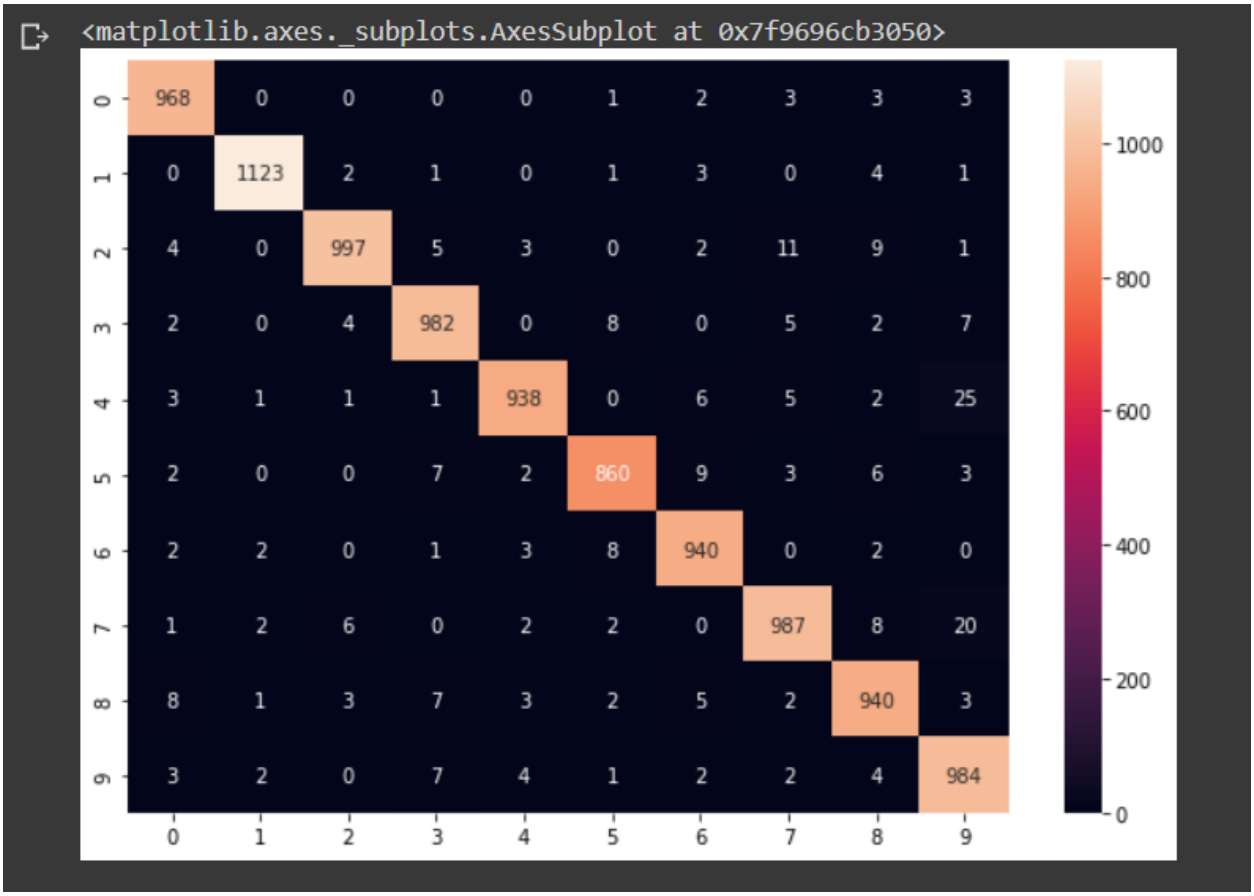
```
# Sử dụng model_2 cho dữ liệu đã chuẩn hóa
model_2_new = Sequential()
model_2_new.add(Dense(784, input_shape=(784, ),activation='relu'))
model_2_new.add(Dense(10, input_shape=(10, ), activation='sigmoid'))
optimizer = Adam(learning_rate=0.01)
loss1 = BinaryCrossentropy()
model_2_new.compile(optimizer=optimizer, loss=loss1, metrics=['accuracy'])
model_2_new.fit(X_train_new, y_train, batch_size=128, epochs=10)

y_pred_2_new = model_2_new.predict(X_test_new)
y_pred_label_2_new = np.argmax(y_pred_2_new, axis=-1)
print("")
print("")
print("Kết quả: ")
accuracy_score(y_test, y_pred_label_2_new)*100

```

KẾT QUẢ: 97.19%

```
Kết quả:
97.19
```

So sánh: Sau khi chuẩn hóa dữ liệu, sử dụng cùng 1 mô hình của Bài tập 2, thì kết quả dự đoán của mô hình Bài tập 2 khi sử dụng dữ liệu chuẩn hóa để train mô hình tốt hơn so với dùng dữ liệu chưa chuẩn hóa để train mô hình (khả năng dự đoán đã được cải thiện từ 88.53% lên thanh 97.19%)

BÀI TẬP 4 In ra ma trận nhầm lẫn (confusion matrix) của mô hình ở Bài tập 3 và nêu nhận xét.



BÀI TẬP MỞ RỘNG

BÀI TẬP 5* Tìm hiểu bộ dữ liệu small CIFAR10 và xây dựng mô hình mạng neural đơn giản gồm hai lớp cho bộ dữ liệu này.

Bộ dữ liệu small CIFAR10 là tập con của bộ dữ liệu Tiny Images. Gồm 60.000 hình ảnh màu kích thước 32x32. Các hình ảnh được gán nhãn với 1 trong 10 loại . Mỗi lớp gồm 6000 hình ảnh, chia 6000 ảnh thành 5000 ảnh của tập train, 1000 ảnh tập test.

```
[ ]  1 from keras.layers import Dropout
     2 model_CIFAR10 = Sequential()
     3 model_CIFAR10.add(Dense(3072, input_shape=(X_train_CIFAR10.shape[1],), activation='sigmoid'))
     4 model_CIFAR10.add(Dense(10, input_shape=(10, ), activation='sigmoid'))
     5 optimizer = Adam(learning_rate=0.01)
     6 model_CIFAR10.add(Dropout(0.02))
     7 loss1 = BinaryCrossentropy()
     8 model_CIFAR10.compile(optimizer=optimizer, loss=loss1, metrics=['accuracy'])
     9 model_CIFAR10.fit(X_train_CIFAR10, y_train_CIFAR10, batch_size=64, epochs=10)
    10
    11 y_pred_CIFAR10 = model_CIFAR10.predict(X_test_CIFAR10)
    12 y_pred_label_CIFAR10 = np.argmax(y_pred_CIFAR10, axis=-1)
    13 print("")
    14 print("")
    15 print("Kết quả: ")
    16 accuracy_score(y_test_CIFAR10, y_pred_label_CIFAR10)*100
    17
```

Kết quả:

```
Kết quả:
10.0
```

BÀI TẬP 6

- Thiết kế và xây dựng mô hình mạng neural có số lượng và đặc điểm các lớp tùy thích.

```
 1 model_3 = Sequential()
 2
 3 model_3.add(Dense(784, input_shape=(784, ),activation='relu'))
 4 model_3.add(Dense(784, input_shape=(784, ),activation='relu'))
 5 model_3.add(Dense(784, input_shape=(784, ),activation='relu'))
 6 model_3.add(Dense(10, input_shape=(10, ), activation='sigmoid'))
 7 optimizer = Adam(learning_rate=0.002)
 8 model_CIFAR10.add(Dropout(0.02))
 9 loss1 = BinaryCrossentropy()
10 model_3.compile(optimizer=optimizer, loss=loss1, metrics=['accuracy'])
11 model_3.fit(X_train, y_train, batch_size=64, epochs=20)
12
13 y_pred_3 = model_3.predict(X_test)
14 y_pred_label_3 = np.argmax(y_pred_3, axis=-1)
15 print("")
16 print("")
17 print("Kết quả: ")
18 accuracy_score(y_test, y_pred_label_3)*100
```

- Áp dụng mô hình này trên tập dữ liệu MNIST do thư viện Keras cung cấp và tính độ chính xác dự đoán của mô hình.

```
Kết quả:
97.38
```