เอกสารอธิบายการทดลองที่ 3 ตอนที่ 2

การทดลองสร้าง สอน และ ทดสอบโมเดล เพื่อดูค่าความแม่นยำเบื้องต้นของแต่ละโมเดล (CNN, LSTM)

ตอนที่ 2 : การทดลองสร้าง สอน และ ทดสอบโมเดล เพื่อดูค่าความแม่นยำเบื้องต้นของแต่ละโมเดล (CNN, LSTM)

2.1 กำหนดโครงสร้างโมเดล (model architecture) ที่ต้องการนำมาทดสอบ

- กำหนดโครงสร้างโมเดล CNN โดยกำหนดโครงสร้างเป็น conv2d(32), MaxPooling2D(2, 2), conv2d(64), conv2d(128) , Flatten, Dense(128) และ Dense(5) ซึ่ง activation function เป็น ReLU ยกเว้น output layer เป็น sigmoid

```
model = Sequential()

model.add(Conv2D(32, kernel_size=(3, 3), activation='relu', input_shape=(X_train
.shape[1:]), padding='same'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(BatchNormalization())
model.add(Dropout(0.25))

model.add(Conv2D(64, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())

model.add(Conv2D(128, kernel_size=(3, 3), activation='relu', padding='same'))
model.add(BatchNormalization())

model.add(Flatten())
model.add(Dense(128 , activation='relu' ))
model.add(Dense(y_train.shape[1], activation='sigmoid'))
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 3, 4, 32)          320
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 1, 2, 32)          0
_________________________________________________________________
batch_normalization (BatchNo (None, 1, 2, 32)          128
_________________________________________________________________
dropout (Dropout)            (None, 1, 2, 32)          0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 1, 2, 64)          18496
_________________________________________________________________
batch_normalization_1 (Batch (None, 1, 2, 64)          256
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 1, 2, 128)         73856
_________________________________________________________________
batch_normalization_2 (Batch (None, 1, 2, 128)         512
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 128)               32896
_________________________________________________________________
dense_1 (Dense)              (None, 5)                 645
=================================================================
Total params: 127,109
Trainable params: 126,661
Non-trainable params: 448
_________________________________________________________________
```

รูปโครงสร้างโมเดล CNN

- กำหนดโครงสร้างโมเดล LSTM โดยกำหนดโครงสร้างเป็น LSTM(32) และ Dense(5) โดย output layer ใช้ activation function เป็น sigmoid

```
model = Sequential()
model.add(LSTM(32, input_shape=(X_2d_train.shape[1:]) ))
model.add(Dense(y_train.shape[1], activation='sigmoid'))
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 32)                4736
_________________________________________________________________
dense_3 (Dense)              (None, 5)                 165
=================================================================
Total params: 4,901
Trainable params: 4,901
Non-trainable params: 0
```

รูปโครงสร้างโมเดล LSTM

2.2 กำหนด Optimizer Parameters

- โมเดล CNN ใช้ optimizer เป็น Adam มี learning rate เท่ากับ 0.001 loss เป็น categorical_crossentropy

```
optimizer = Adam(learning_rate=0.001, beta_1=0.9, beta_2=0.999, amsgrad=False)
model.compile( loss='categorical_crossentropy', optimizer=optimizer, metrics=['a
ccuracy'])
```

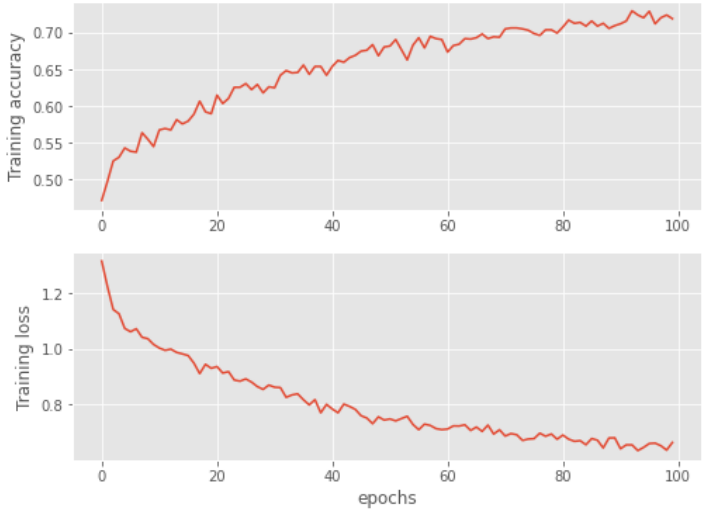
- โมเดล LSTM ใช้ optimizer เป็น Adam มี learning rate เท่ากับ 0.003 loss เป็น categorical_crossentropy

```
optimizer = Adam(learning_rate=0.003, beta_1=0.9, beta_2=0.999, amsgrad=False)
model.compile( loss='categorical_crossentropy', optimizer=optimizer, metrics=['a
ccuracy'])
```

2.3 ทำการสอนโมเดล (Training model)

- สอนโมเดล CNN ด้วยข้อมูล X_train และ y_train ใช้ batch size เท่ากับ 16 epochs เท่ากับ 100 โดยมีชุดข้อมูล X_valid และ y_valid ในการตรวจสอบ

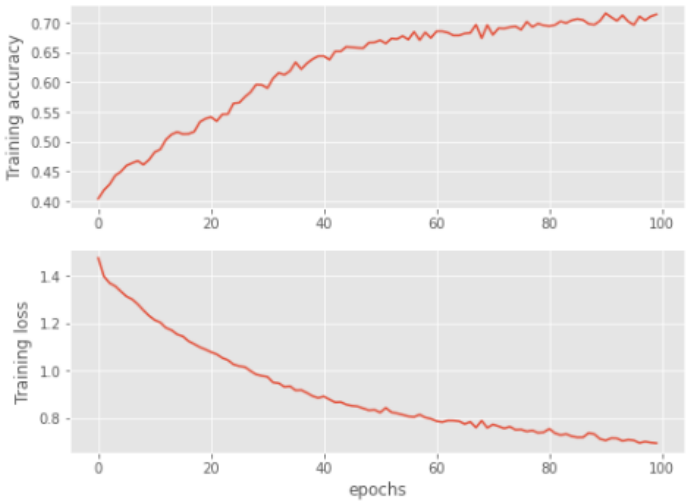
```
history = model.fit(X_train, y_train, batch_size=16, validation_data=(X_valid, y
_valid), epochs=100, verbose=1)
```



รูปการณ์สอนโมเดล CNN

- สอนโมเดล LSTM ด้วยข้อมูล X_2d_train และ y_2d_train ใช้ batch size เท่ากับ 16 epochs เท่ากับ 100 โดยมีชุดข้อมูล X_2d_valid และ y_2d_valid ในการตรวจสอบ

```
history = model.fit( X_2d_train, y_train, batch_size=16, validation_data=(X_2d_v
alid, y_valid), epochs=100)
```



รูปการณ์สอนโมเดล LSTM

2.4 ทำการ predict ค่าจากโมเดลทั้ง 2 แบบ (CNN, LSTM) ด้วยข้อมูล X_test, y_test ที่แบ่งไว้

- ทำนายค่าจากโมเดล CNN ด้วยข้อมูล X_test, y_test

```
y_prediction = model.predict(X_test)
y_pred_single = [np.argmax(p) for p in y_prediction]
y_test_single=[np.argmax(p) for p in y_test]
```

- ทำนายค่าจากโมเดล LSTM ด้วยข้อมูล X_2d_test, y_2d_test

```
y_prediction = model.predict(X_2d_test)
y_pred_single = [np.argmax(p) for p in y_prediction]
y_test_single = [np.argmax(p) for p in y_test]
```

2.5 และ 2.6 คำนวณค่าตัววัดประสิทธิภาพของการทำนายจากโมเดล CNN และ LSTM โดยวัดค่า classification report/ confusion matrix และแสดงรูปภาพของ classification_report และ confusion_matrix

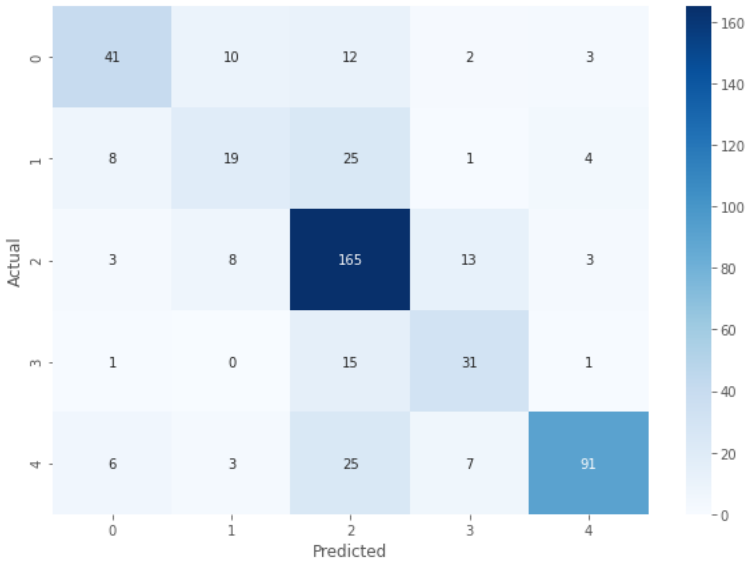
- คำนวณค่าตัววัดประสิทธิภาพของการทำนายจากโมเดล CNN ด้วยฟังชัน classification_report จาก sklearn เพื่อดู precision, recall, f1-score, accuracy เป็นต้น และแสดง confusion matrix โดยพบว่าโมเดลมีความแม่นยำร้อยละ 70 จากการทำนายได้ดังนี้

```
print(classification_report(y_test_single, y_pred_single))

conf_mat = confusion_matrix(y_test_single, y_pred_single)
plt.figure(figsize = (10, 7))
ax = sns.heatmap(conf_mat, annot=True, fmt="d", xticklabels='0 1 2 3 4'.split(),
 yticklabels='0 1 2 3 4'.split(), cmap="Blues")
bottom, top = ax.get_ylim()
plt.ylabel("Actual")
plt.xlabel("Predicted")
plt.show()
```

	precision	recall	f1-score	support
0	0.69	0.60	0.65	68
1	0.47	0.33	0.39	57
2	0.68	0.86	0.76	192
3	0.57	0.65	0.61	48
4	0.89	0.69	0.78	132
accuracy			0.70	497
macro avg	0.66	0.63	0.64	497
weighted avg	0.71	0.70	0.69	497

รูปการวัดประสิทธิโมเดล CNN ด้วย classification_report function



รูปการวัดประสิทธิโมเดล CNN ด้วย confusion matrix

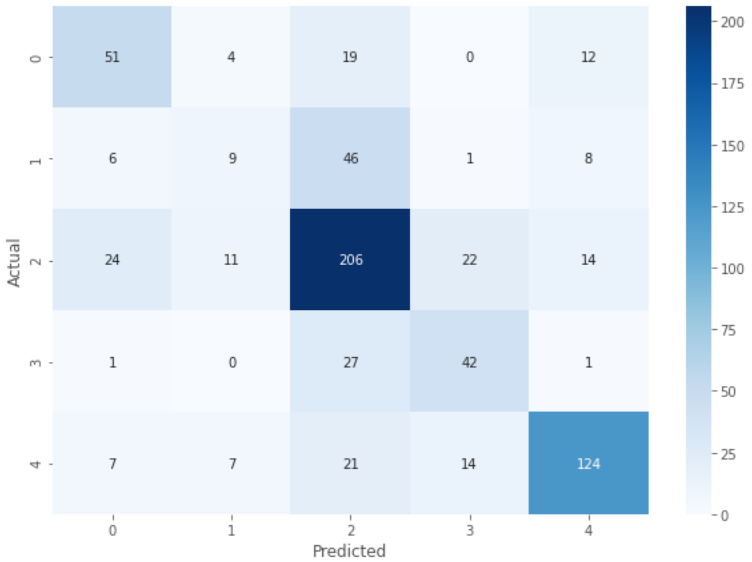
- คำนวณค่าตัววัดประสิทธิภาพของการทำนายจากโมเดล LSTM ด้วยฟังชัน classification_report จาก sklearn เพื่อดู precision, recall, f1-score, accuracy เป็นต้น และแสดง confusion matrix โดยพบว่าโมเดลมีความแม่นยำร้อยละ 74 จากการทำนายได้ดังนี้

```
print(classification_report(y_test_single, y_pred_single))

conf_mat = confusion_matrix(y_test_single, y_pred_single)
plt.figure(figsize = (10, 7))
ax = sns.heatmap(conf_mat, annot=True, fmt="d", xticklabels='0 1 2 3 4'.split(),
 yticklabels='0 1 2 3 4'.split(), cmap="Blues")
bottom, top = ax.get_ylim()
plt.ylabel("Actual")
plt.xlabel("Predicted")
plt.show()
```

	precision	recall	f1-score	support
0	0.57	0.59	0.58	86
1	0.29	0.13	0.18	70
2	0.65	0.74	0.69	277
3	0.53	0.59	0.56	71
4	0.78	0.72	0.75	173
accuracy			0.64	677
macro avg	0.56	0.55	0.55	677
weighted avg	0.62	0.64	0.62	677

รูปการวัดประสิทธิโมเดล LSTM ด้วย classification_report function



รูปการวัดประสิทธิโมเดล LSTM ด้วย confusion matrix