**1. ระบุ dataset ที่ใช้ พร้อมวิเคราะห์ความแตกต่างปริมาณข้อมูลสำหรับแต่ละคลาสคำตอบ ว่าแตกต่างกันอย่างไร และอาจส่งผลกระทบต่อการสอนโมเดลหรือไม่ \***

ใช้ dataset ชุดที่ 1 โดยใช้รหัส 46343, 781756 และ 1066528 โดยคลาสคำตอบ(labeled\_sleep) ประกอบด้วย :

คลาส 2 จำนวน 930 sample

คลาส 5 จำนวน 658 sample

คลาส 0 จำนวน 353 sample

คลาส 1 จำนวน 305 sample

คลาส 3 จำนวน 241 sample

ส่งผลกระทบต่อโมเดลเนื่องจากคลาสแต่ละคลาสมีจำนวนไม่เท่ากัน อาจทำให้โมเดลทำนายคลาสที่มีจำนวนน้อยกว่าไม่ดีเท่าที่ควรได้ อาจแก้โดยการ bias คลาส เป็นต้น

**2. ถ้ามีบางคลาสคำตอบ มีข้อมูลในปริมาณน้อยเกินไป แล้วต้องการเพิ่มปริมาณข้อมูลสามารถทำได้ ด้วยวิธีใดบ้าง ยกตัวอย่าง \***

1. ถ้าหา dataset สามารถหาเพิ่มได้ ควรหาคลาสที่มีน้อยแล้วเพิ่มใน dataset ที่มี เช่น ในกรณีข้อ 1 พบว่า คลาส 3 มีจำนวนน้อยสุดเมื่อเทียบกับคลาสอื่น

2. เพิ่มจำนวนคลาสที่น้อยโดยการคัดลอก instances เช่น คัดลอกคลาส 3 ให้มีจำนวนมากขึ้น

3. ลบคลาสที่มีจำนวนมากๆ เพื่อทำให้คลาสเหลือเท่าๆ กัน เช่น ลบคลาส 2 และ 5 ออกเพื่อให้มีจำนวนเท่าๆกับคลาสอื่น

**3. ระบุ input\_shape ที่ใส่ให้กับ CNN และ LSTM \***

CNN :

- Train = (1589, 3, 4, 1)

- Valid = (398, 3, 4, 1)

- Test = (497, 3, 4, 1)

LSTM :

- Train = (4768, 4, 1)

- Valid = (1193, 4, 1)

-Test = (1491, 4, 1)

**4. ค่า epoch, batch\_size ที่แตกต่าง มีผลอย่างไรต่อการ train \***

ถ้า batch\_size มีค่ามากเกินไปอาจส่งผลให้โมเดลไม่สามารถปรับพารามิเตอร์ได้ดีเท่าที่ควรหรือโมเดลยังไม่สามารถจำ pattern ได้เท่าที่ควรโดยจะทำให้การรันโมเดลมีระยะเวลาที่เร็วขึ้น แต่ถ้า batch\_size มีค่าน้อยเกินไปจะทำให้การรันโมเดลมีระยะเวลาที่นานขึ้น

ถ้า epoch มีค่ามากจะทำให้การรันโมเดลมีระยะเวลาที่นานมากขึ้นและอาจทำให้ดมเดลเกิดการ overfiting ได้ แต่ถ้ามีค่าน้อยเกินไป อาจทำ underfit ได้

**5. วิเคราะห์ผลลัพธ์ เมื่อปรับเปลี่ยน ชนิด optimizer, จำนวน layer, activation@output layer \***

- ทดลองเปลี่ยน optimizer จาก

Adam(lr=0.001, beta\_1=0.9, beta\_2=0.999, amsgrad=False) เป็น SGD(lr=0.001, decay=1e-6, momentum=0.8, nesterov=True)

- เปลี่ยน activation function ที่ output layer เป็น relu

- เพิ่มจำนวน Conv2D layer ของโมเดล CNN

model.add(Conv2D(128, kernel\_size=(3, 3), activation='relu', padding='same'))

จากการเปลี่ยนค่าตามข้างต้นพบว่า โมเดล CNN มีค่าความแม่นยำอยู่ที่ร้อยละ 39 ซึ่งน้อยกว่าในการทำการทดลอง โดยทำนายเพียงคลาส 0 และ 3 เท่านั้น และโมเดล LSTM มีค่าความแม่นยำอยู่ที่ร้อยละ 12 โดยทำนายเพียงคลาส 1 เท่านั้น อาจเนื่องจากการเปลี่ยน activation function ที่ output layer เป็น relu เพราะคลาสมีค่าระหว่าง 0 และ 1 เท่านั้น หรืออาจเพราะเปลี่ยน optimizer จาก Adam เป็น SGD ซึ่ง Adam เป็น optimizer ที่สามารถสามารถแก้ปัญหา decaying ของ gradients ได้

**6. แสดงค่าพารามิเตอร์ที่ดีที่สุดสำหรับ CNN และ LSTM ที่ทำการทดลอง \***

CNN : (3 Conv2D(32, 64, 128), 1 MaxPooling2D(2\*2) )

- optimizer = Adam(learning\_rate=0.001, beta\_1=0.9, beta\_2=0.999, amsgrad=False)

- batch\_size = 16

- epoch = 100

- loss = categorical\_crossentropy

LSTM : (LSTM(128))

- optimizer = Adam(learning\_rate=0.003, beta\_1=0.9, beta\_2=0.999, amsgrad=False)

- batch\_size = 16

- epoch = 100

- loss = categorical\_crossentropy

**7. วิเคราะห์ผลความสามารถในการแบ่งแยกคลาสผลลัพธ์แต่ละคลาสสำหรับ CNN, LSTM ว่าแบ่งแยกได้ดีหรือไม่ แตกต่างกันอย่างไร \***

โมเดล CNN มีความแม่นยำของโมเดลอยู่ที่ร้อยละ 69 พบว่าคลาส 2 และ 4 โมเดลสามารถทำนายได้แม่นยำมากที่สุด โดยโมเดลทำนายคลาส 1 และ 3 ยังไม่ดีเท่าที่ควร

โมเดล LSTM มีความแม่นยำของโมเดลอยู่ที่ร้อยละ 73 พบว่าคลาส 0 2 3 และ 4 โมเดลสามารถทำนายได้แม่นยำมากที่สุด โดยโมเดลทำนายคลาส 1 ยังไม่ดีเท่าที่ควร

จากโมเดลทั้ง 2 พบว่าโมเดล LSTM มีความแม่นยำมากกว่า CNN อาจเนื่องจากเป็น recurrent network ที่มีการป้อนขอ้มูลกลับหรือเก็บสถานะก่อนหน้า ทำให้โมเดลมีความแม่นยำมากกว่าโมเดล CNN ได้

**8. ค่า validation\_loss สุดท้ายในการทำ gridsearch สำหรับ CNN, LSTM มีค่าเป็นอย่างไร สามารถบอกได้หรือไม่ว่า โมเดลยืดหยุ่นใช้งานดีพอหรือยัง \***

จากการทดลองโดยใช้ Talos พบว่าจะทำการคืนค่าพารามิเตอร์ที่ดีที่สุดโดยดูจาก validation accuracy และ validation loss จากนั้นจะนำค่าพารามิเตอร์ดังกล่าวเข้าโมเดลเพื่อทำการเทรน ซึ่งได้ค่า validation loss ดังนี้

CNN = 0.818297

LSTM = 0.773707

พบว่าโมเดลอาจไม่ยืดหยุ่นเท่าที่ควรเนื่องจากผู้ทำได้ใช้ validation accuracy ตัดสินใจในการเลือกโมเดลเป็นหลัก ซึ่งอาจทำให้โมเดลมีความผิดพลาดได้