

**EGCO486: Image Processing**

**Final Project**

RE-AL or FA-KE

**Reported by**

Nipatsa Chainiwattana 6513114

Puthipong Yomabut 6513134

Phattaradanai Sornsawang 6513172

Patiharn Kamenkit 6513170

**Submitted to**

Asst. Prof. Dr. Narit Hnoohom

COMPUTER ENGINEERING INTERNATIONAL PROGRAM

MAHIDOL UNIVERSITY INTERNATIONAL COLLEGE

2024

**Background**

จากปัจจุบันเทคโนโลยี Generative AI มีบทบาทสำคัญในหลายด้านโดยเฉพาะในการสร้างภาพที่มีความ เหมือนจริงสูง ทำให้ยากต่อการแยกแยะระหว่างภาพวาดที่ถูกสร้างขึ้นด้วย Generative AI และภาพที่เป็น ของจริง คณะผู้จัดทำจึงได้จัดทำโปรเจคนี้ขึ้นเพื่อแก้ปัญหาการแอบอ้างใช้ Generative AI แทนภาพจริง โดย RE-AL or FA-KE เป็นเว็บแอพพลิเคชั่นสำหรับจำแนกภาพวาดจริงกับภาพวาดจาก Generative AI โดยใช้หลักการทาง Image Processing และ Convolutional Neural Network มาประยุกต์ใช้ โดยในโปรเจคนี้ได้พัฒนาโมเดล Convolutional Neural Networks (CNN) และโมเดลที่ใช้เทคนิค Transfer Learning จากโมเดล ResNet50, Moblienet V2 และ VGG19 แล้ว Deploy บนเว็บแอพพลิเคชั่นที่พัฒนาด้วย Streamlit เพื่อใช้แสดงผลลัพธ์และ ทำนายรูปภาพที่อัพโหลดว่าเป็นภาพวาดจริงหรือภาพวาดจาก Generative AI คณะผู้จัดทำขอขอบพระคุณ ผศ.ดร. นริศ หนูหอม ผู้ให้ความรู้ และแนวทางในการศึกษา สุดท้ายนี้ทางคณะผู้จัดทำหวังว่ารายงานฉบับนี้จะให้ ความรู้และประโยชน์ไม่มากก็น้อยแก่ผู้อ่านทุกท่าน โดยหากมีข้อผิดพลาดประการใดผู้จัดทำจะขอน้อมรับไว้และ ขออภัยมา ณ ที่นี้

คณะผู้จัดทำ

**Contents**

Objective 4

Scope 5

Method 6

Experiment results 20

Conclusion 38

Discussion and future work 39

References 40

**Objective**

1. เพื่อสามารถจำแนกภาพวาดที่สร้างโดยมนุษย์ (Real Art) และภาพวาดที่สร้างโดย AI (AI-Generated) โดยใช้โมเดล Deep Learning
2. แสดงผลบน Web Application ผ่าน Streamlit platform ซึ่งเป็นเครื่องมือที่ใช้การสร้างและแชร์ Web Application เพื่อให้ user สามารถใช้งานได้ง่าย
3. เพื่อนำความรู้ด้าน Image Processing เเละ Deep Learning Processing มาประยุกต์ใช้ในการแก้ ปัญหาของวงการศิลปะ โดยเฉพาะในกรณีที่เกี่ยวกับการแอบอ้างหรือปลอมเเปลงผลงานและในเรื่องของ ข้อจำกัดทางลิขสิทธิ์จากภาพที่สร้างจาก AI

**Scope**

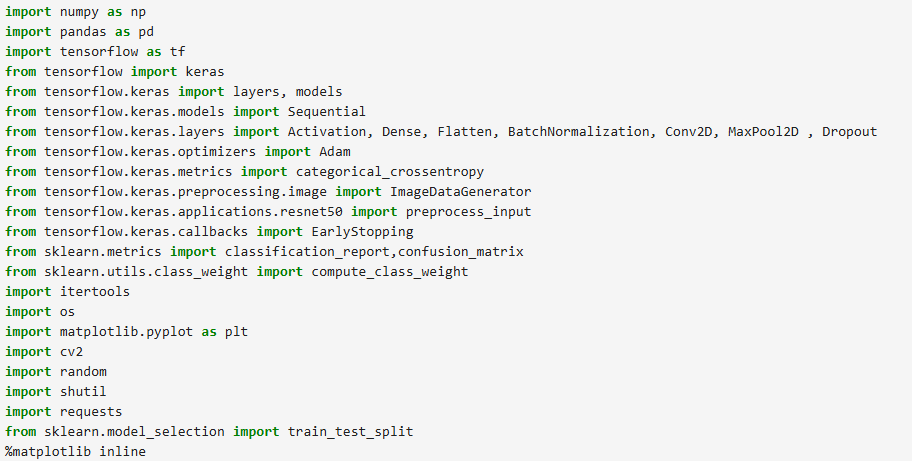
โปรเจคนี้เน้นศึกษาในการพัฒนาเว็บแอพพลิเคชั่นและโมเดลที่ใช้ในการจำแนกภาพระหว่างภาพวาดจริง และภาพวาดจาก Generative AI Dataset โดยพัฒนาโมเดลด้วย Dataset จาก Kaggle : AI and Human Art Classification โดยได้วางแผนพัฒนาโมเดล 5 แบบ ดังนี้ :

1. Basic CNN Classifier
2. ResNet50 Classifier
3. ResNet50 Fine-Tuned Classifier
4. MobileNetV2 Classifier
5. VGG19 Classifier

**Method**

**ขั้นตอนการพัฒนาโมเดลที่ 1 (Basic CNN Classifier)**

1. Import Library ที่ใช้งาน



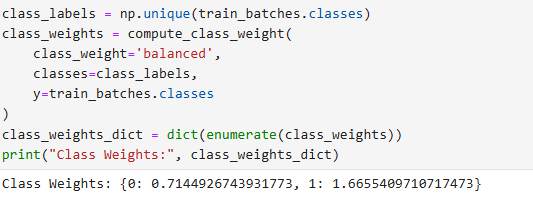
1. กำหนด path และสร้างฟังก์ชั่นสำหรับแบ่งชุดข้อมูลเป็น train, validation และ test



1. ใช้ ImageDataGenerator ในการเตรียมข้อมูลภาพด้วยฟังก์ชั่น preprocess\_input จาก ResNet50 โดยกำหนดขนาดภาพเป็น 224 x 224 และทำ Data Augmentation เพื่อให้โมเดลได้เรียนรู้ภาพใน ลักษณะหลายๆแบบ โดยแบ่งข้อมูลเป็น train\_batches, valid\_batches, test\_batches โดย กำหนดขนาดแบตช์ (batch size) ละ 16 ภาพ



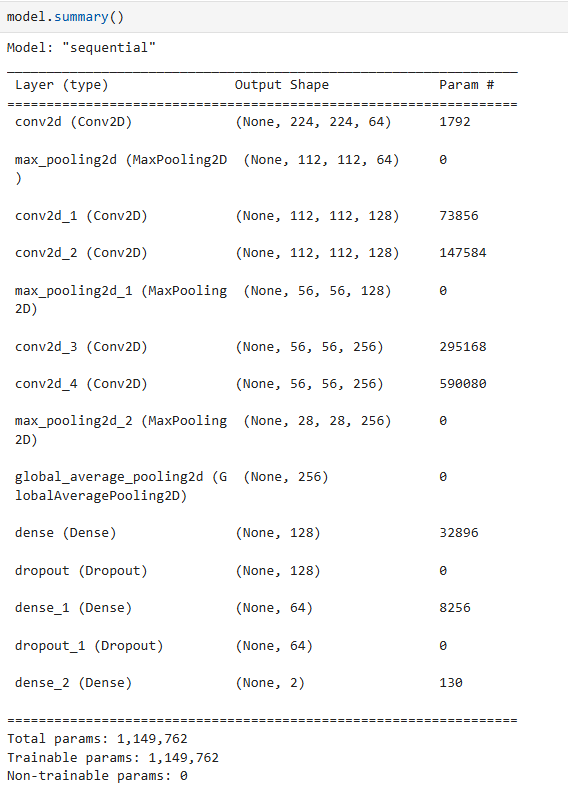
1. คำนวน weight ของแต่ละ class เพื่อแก้ปัญหา class imbalance จากการที่จำนวนรูปของทั้ง 2 class ไม่เท่ากัน



1. สร้างโมเดล CNN ที่มี layer ดังนี้ โดยเลือกใช้ output layer ที่มี activation เป็น softmax เพื่อแสดงค่าความน่าจะเป็นของทั้ง 2 คลาส

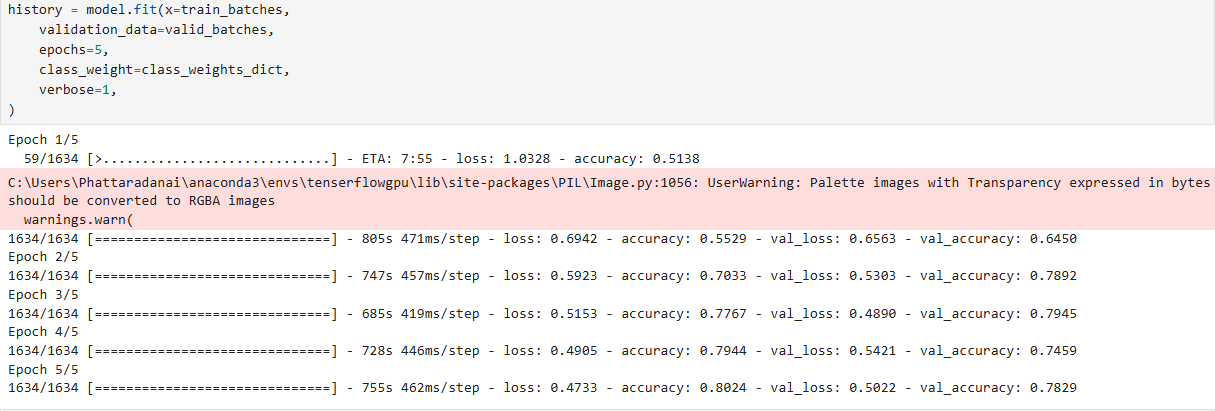
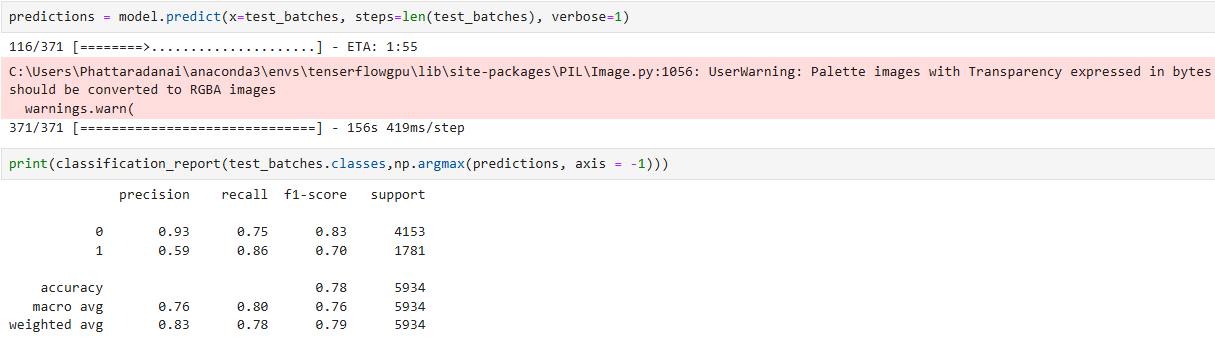


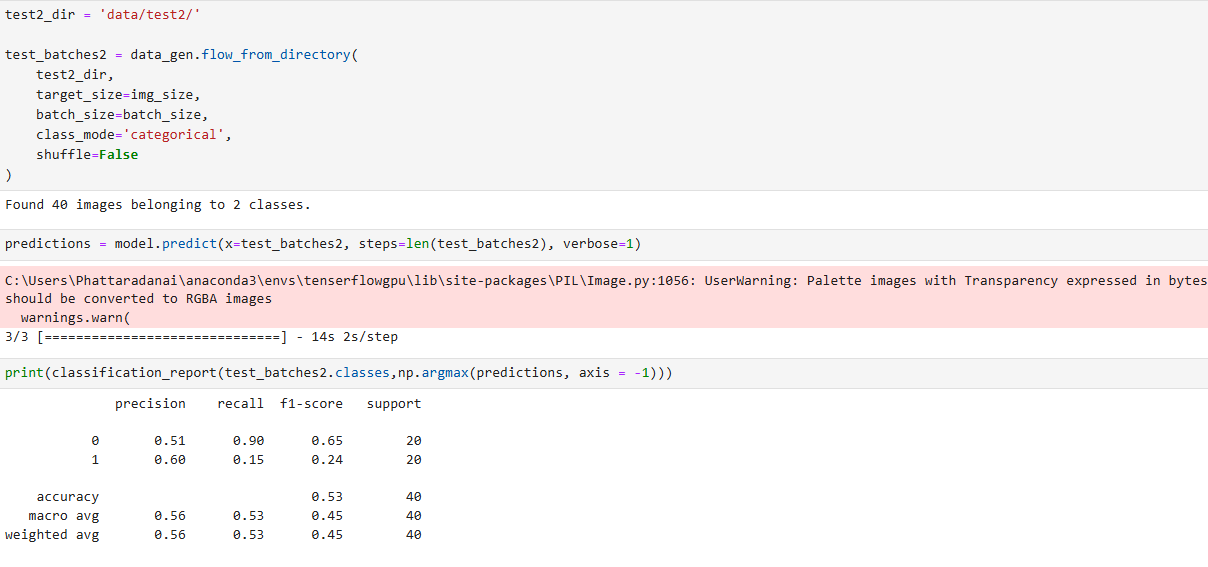
1. ใช้คำสั่ง model.summary() เพื่อตรวจสอบจำนวนเลเยอร์และพารามิเตอร์ ได้ผลดังนี้



1. Compile โมเดลโดยใช้ Optimizer เป็น adam และ Loss function เป็น categorical\_crossentropy และเพิ่ม metrics accuracy สำหรับการวัดผล



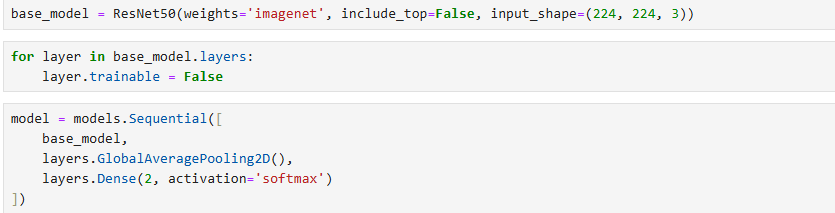
1. เทรนโมเดลด้วยคำสั่ง model.fit โดยทดลองทั้งแบบ 5 epoch และ 10 epoch เพื่อเปรียบเทียบ ผลลัพธ์ที่ได้
2. นำ model ไป predict กับข้อมูลชุด test แล้วตรวจสอบความแม่นยำที่ได้ด้วย classification\_report
3. นำ model ไป predict กับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตเพื่อตรวจสอบความแม่นยำของโมเดล เมื่อใช้กับภาพนอก Dataset แล้วประเมินด้วย classification\_report

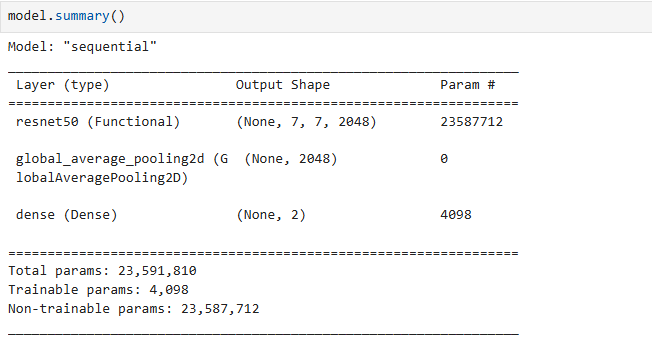


1. เซฟไฟล์โมเดล เป็นไฟล์ .h5 เพื่อนำไป deploy บนเว็บ



**ขั้นตอนการพัฒนาโมเดลที่ 2 (ResNet50 Classifier)**

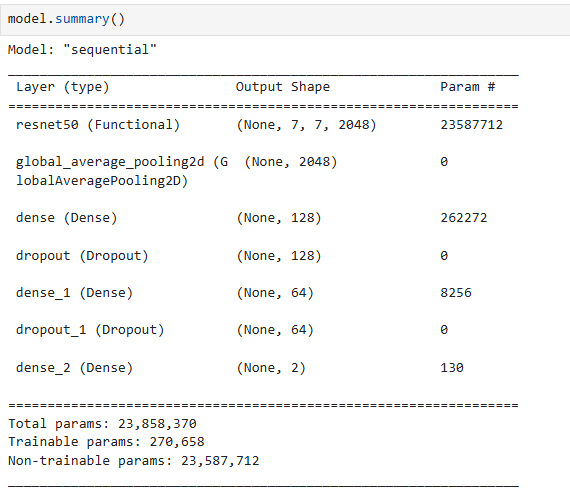
ใช้กระบวนการเดียวกับการพัฒนาโมเดลแรก แต่เปลี่ยนในส่วนขั้นตอนการสร้างโมเดล ด้วยการ import ResNet50 มาเป็น base\_model แล้วทำการ Freeze layer ไว้ แล้วเพิ่ม GlobalAveragePooling2D layer กับ output layer 



**ขั้นตอนการพัฒนาโมเดลที่ 3 (ResNet50 Fine-Tuned Classifier)**

ใช้กระบวนการเดียวกับการพัฒนาโมเดลที่ 2 แต่เพิ่ม layer Dense กับ Dropout ก่อน output layer เพื่อช่วยให้โมเดลทำนายได้ดีขึ้นและลด overfit





**ขั้นตอนการพัฒนาโมเดลที่ 4 (MobilenetV2 Fine-Tuned Classifier)**

ใช้กระบวนการเดียวกับการพัฒนาโมเดลที่ 3 แต่ใช้ base model เป็น MobileNetV2 และใช้ preprocess\_input จาก mobilenet\_v2





**ขั้นตอนการพัฒนาโมเดลที่ 5 (VGG19 Fine-Tuned Classifier)**

ใช้กระบวนการเดียวกับการพัฒนาโมเดลที่ 3 แต่ใช้ base model เป็น VGG19 และใช้ preprocess\_input จาก vgg19

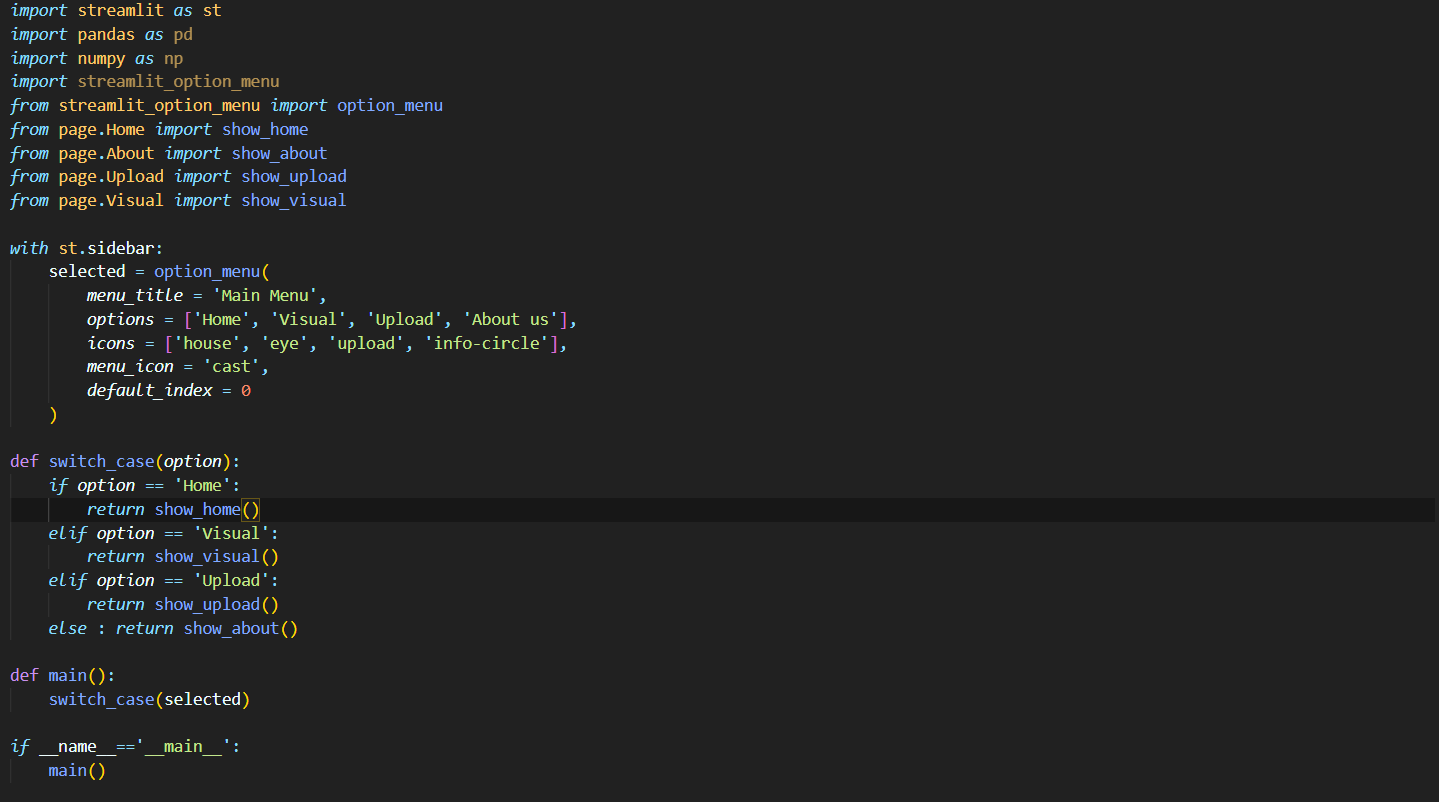




ขั้นตอนการพัฒนาเว็บแอพพลิเคชั่น

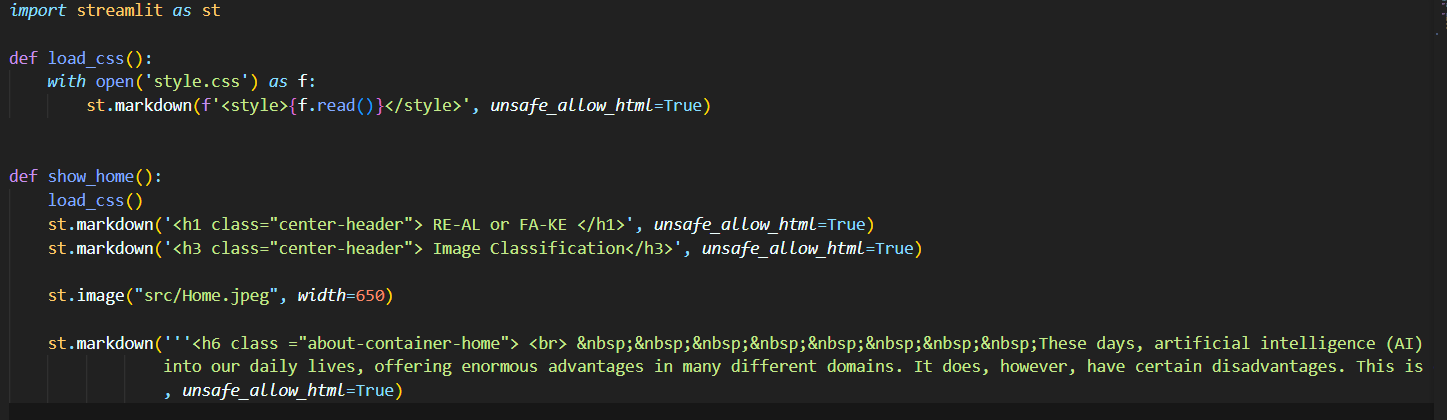
**app.py**

เป็นส่วนที่รวมหน้า page ต่างๆของ Web เข้าไว้ด้วยกัน

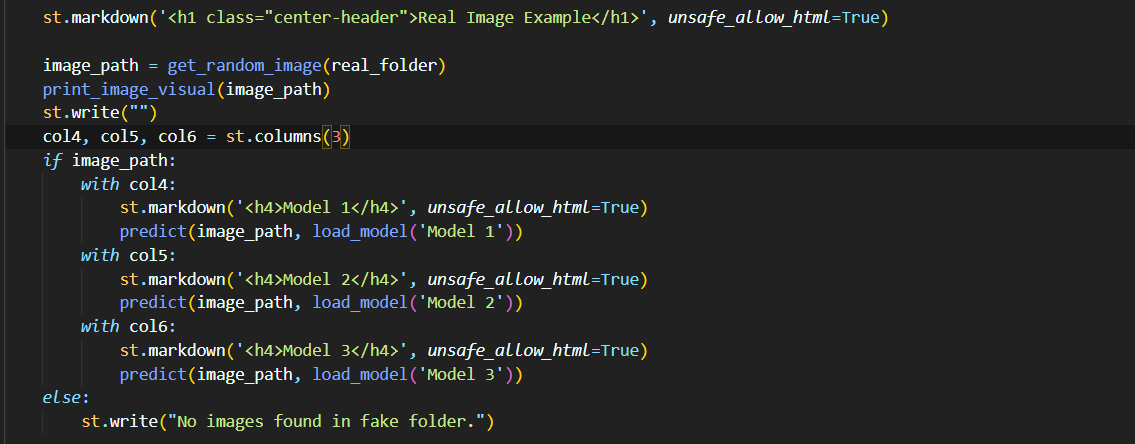
****

**Home.py**

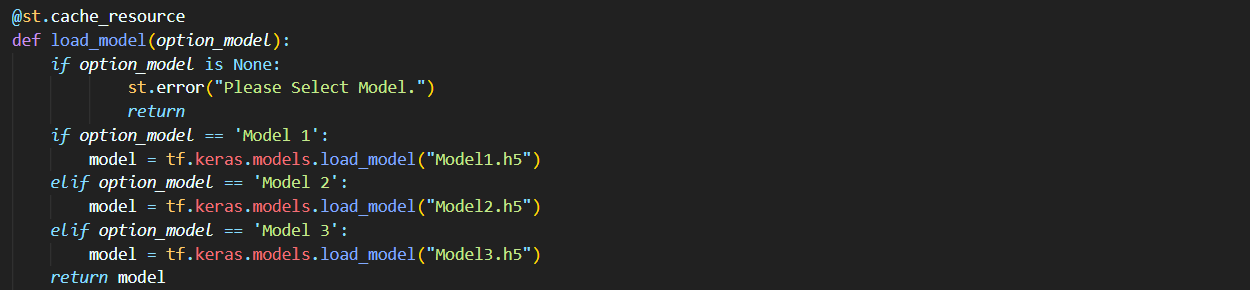
เป็นหน้า Web หลักที่อธิบายเกี่ยวกับ Project นี้

****

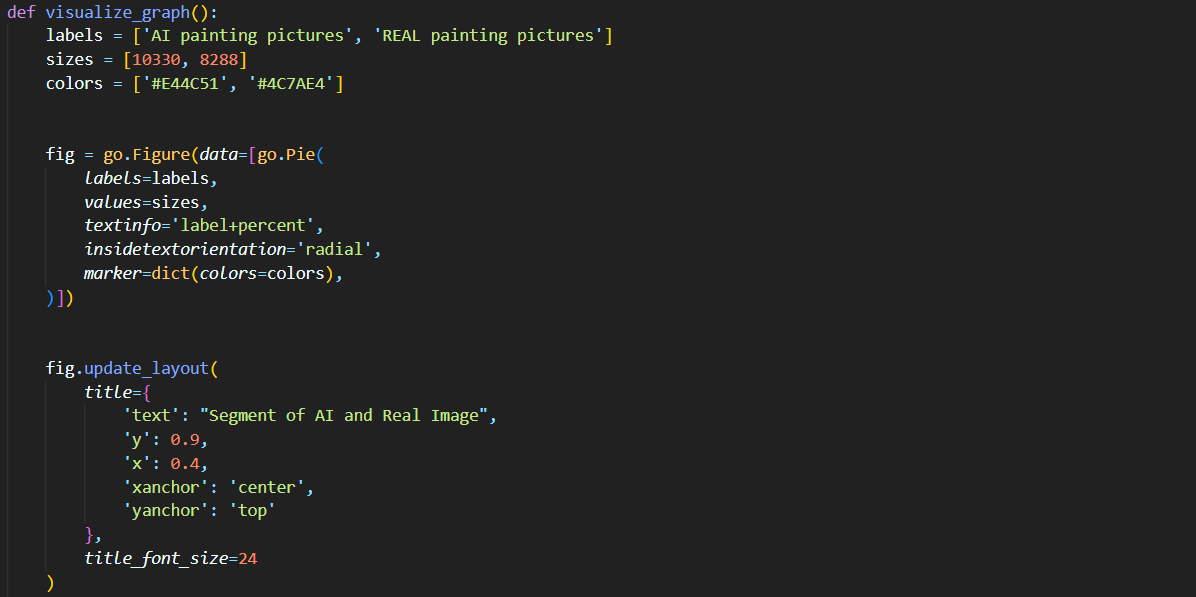
**Visual.py**

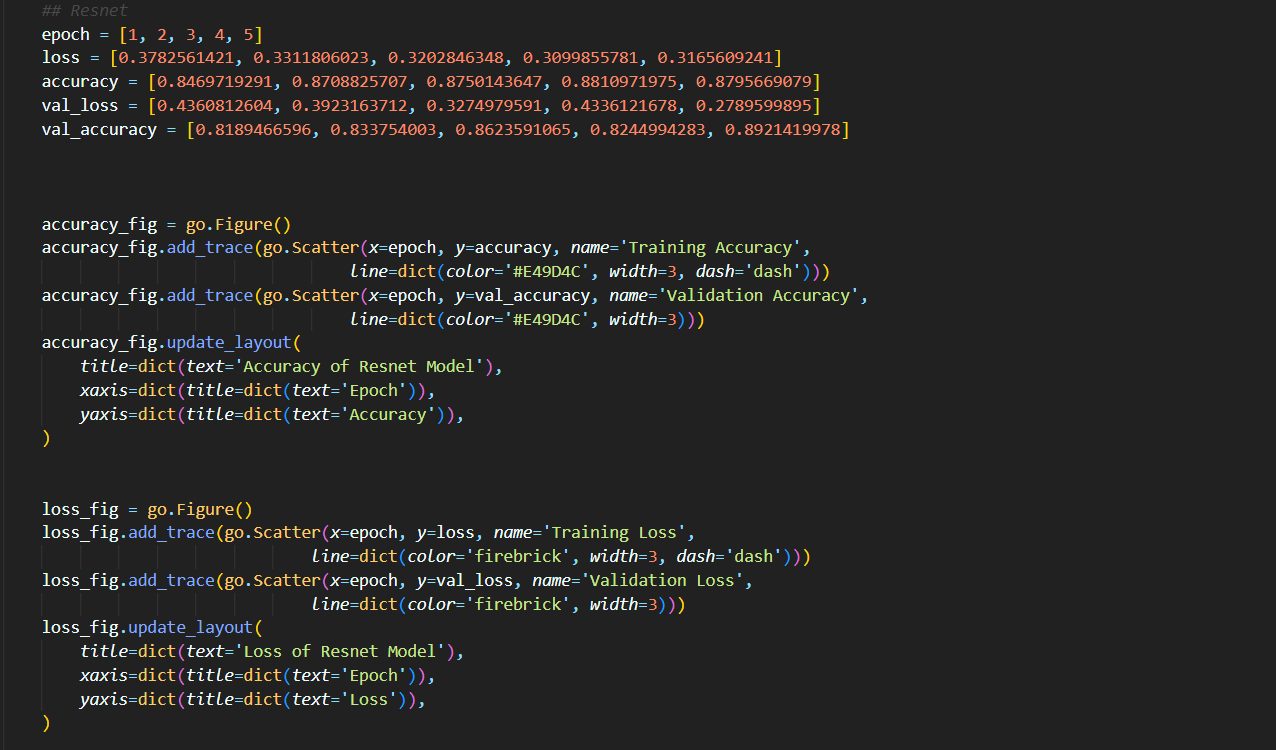
เป็นหน้าที่สุ่มรูปภาพที่ใช้ train model มาเข้า Model และ โชว์ผลลัพธ์จากการ predict

load\_model() : เป็น function ที่เรียกใช้ Model ต่างๆ

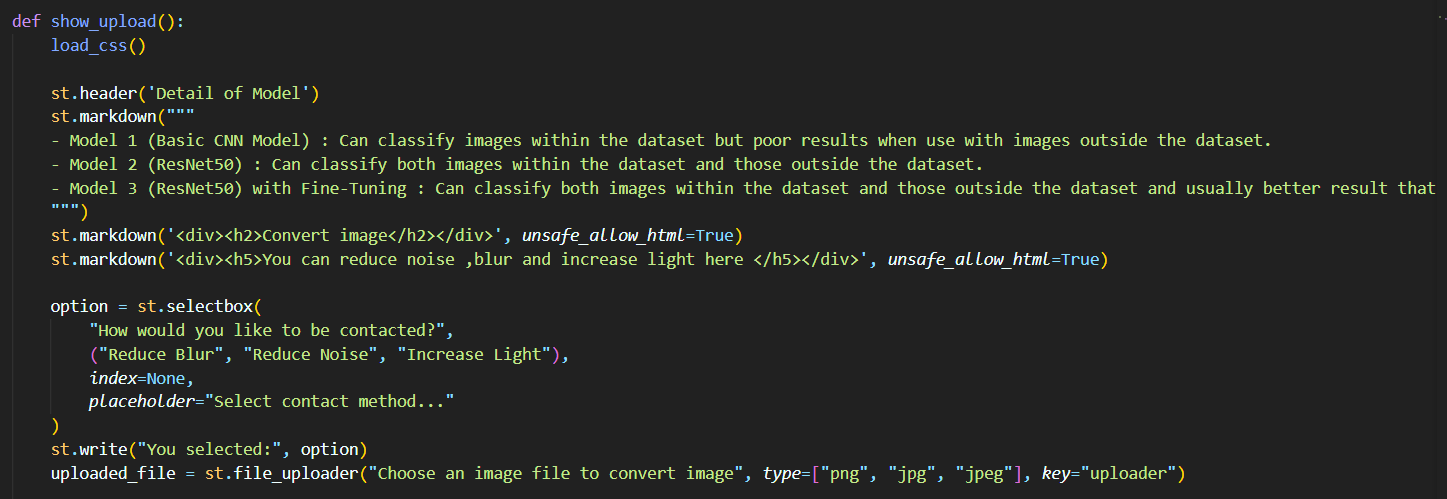


visualize\_graph() : เป็น function ที่แสดงกราฟต่างๆ เช่น จำนวนรูปที่ใช้ train model และ ค่า Accuracy และ Loss ในการ train model แต่ละรอบ





**Upload.py เป็นหน้าที่ให้ User นำภาพมาเข้า Model ได้**

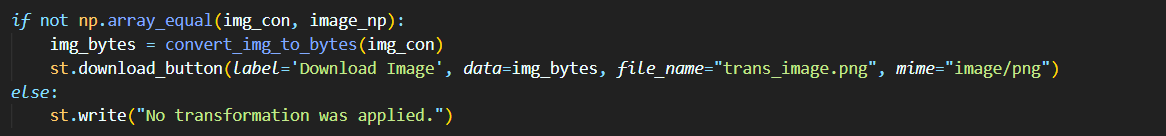


renoise : เป็น function ที่เอาไว้ใช้ลด noise ในภาพ

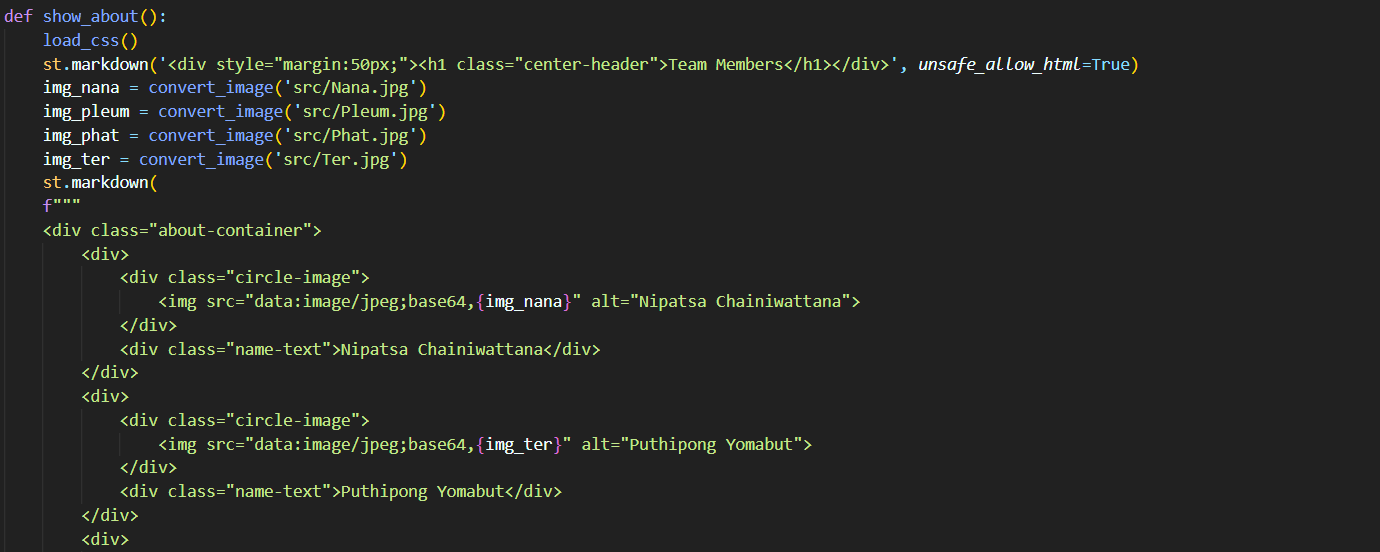
reblur : เป็น function ที่เอาไว้ใช้ลดความ blur ในภาพ

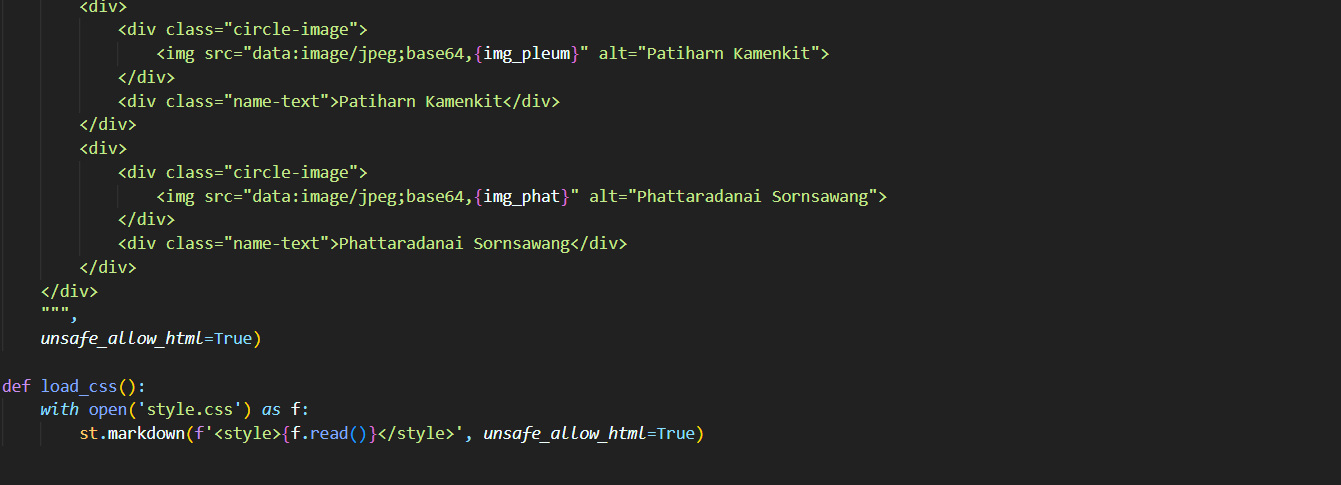
increaselight : เป็น function ที่เอาไว้ใช้เพิ่มความสว่างให้ภาพ



convert\_img\_to bytes : เป็น fucntion ที่แปลงการแสดงภาพแบบ NumPy เป็น bytes เพื่อสามารถให้ User ดาวน์โหลดภาพที่ผ่านการ ลด noise , ลดเบลอ หรือ เพิ่มแสง ได้

**About.py**

เป็นหน้าที่ใช้แสดงรายชื่อผู้จัดทำ Project

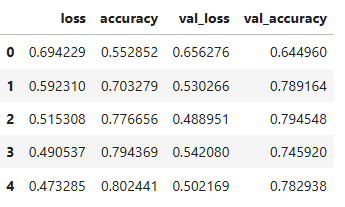


**Experiment results**

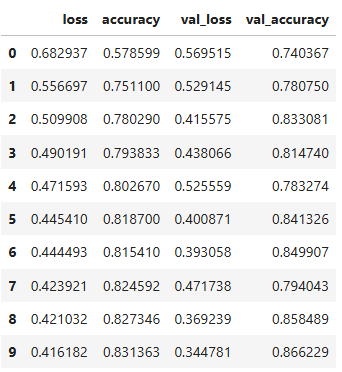
1. **Basic CNN Classifier**

* ได้ค่า loss, accuracy, val\_loss, val\_accuracy ดังนี้

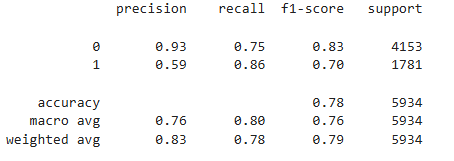
ผลลัพธ์จากการใช้ 5 Epoch

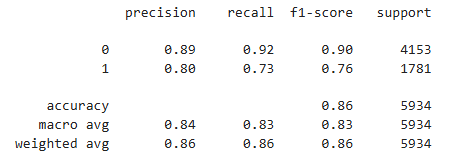


ผลลัพธ์จากการใช้ 10 Epoch



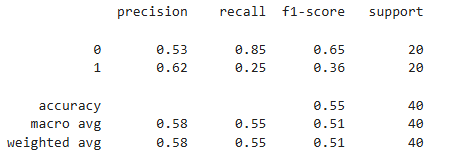
* ได้ค่า classification report เมื่อทดสอบกับข้อมูลชุดทดสอบดังนี้

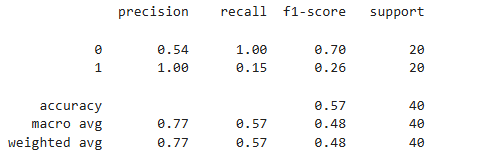
ผลลัพธ์จากการใช้ 5 epoch

ผลลัพธ์จากการใช้ 10 Epoch

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตดังนี้

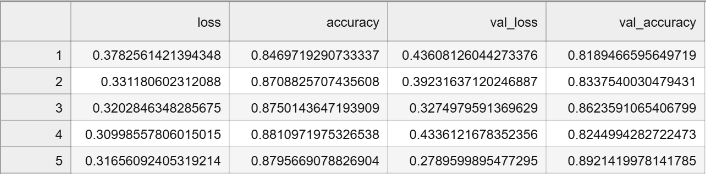
ผลลัพธ์จากการใช้ 5 Epoch

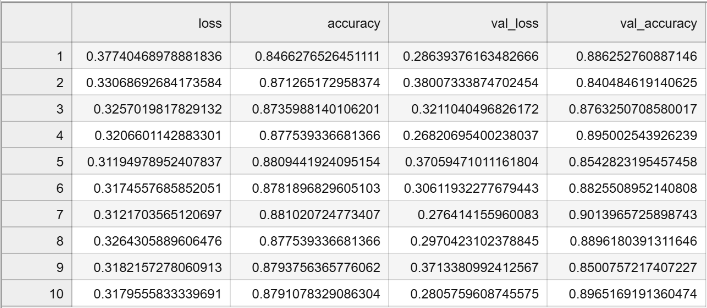


ผลลัพธ์จากการใช้ 10 Epoch

1. ResNet50 Classifier

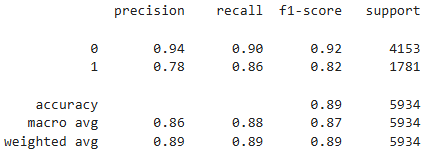
* ได้ค่า loss, accuracy, val\_loss, val\_accuracy ดังนี้

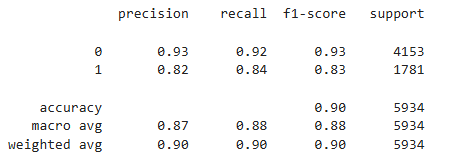
ผลลัพธ์จากการใช้ 5 Epoch

ผลลัพธ์จากการใช้ 10 Epoch

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลชุดทดสอบดังนี้

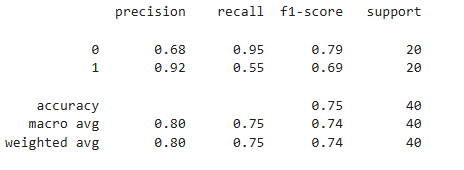
ผลลัพธ์จากการใช้ 5 Epoch



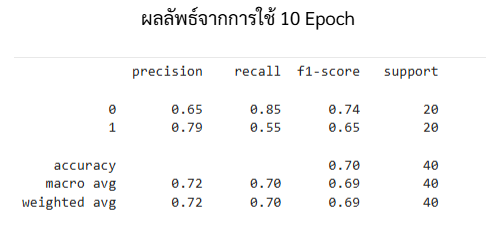
ผลลัพธ์จากการใช้ 10 Epoch

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

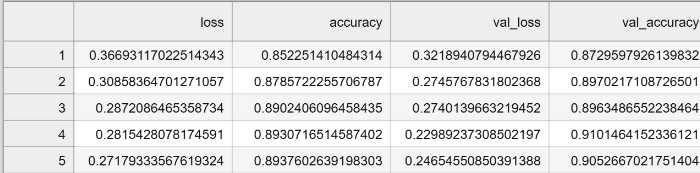
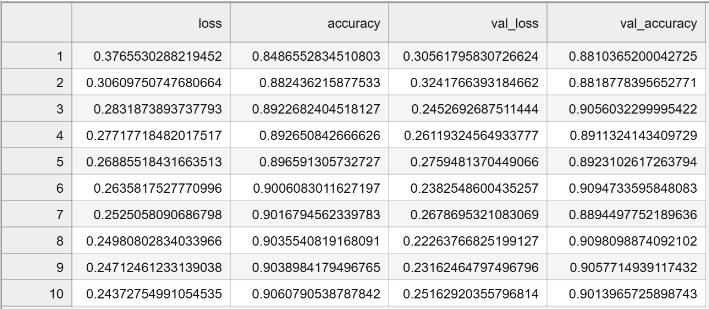


ผลลัพธ์จากการใช้ 10 Epoch



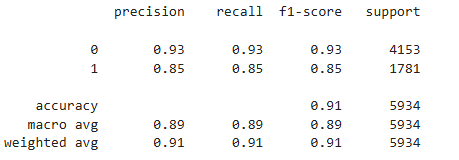
1. ResNet50 Fine-Tuned Classifier

* ได้ค่า loss, accuracy, val\_loss, val\_accuracy ดังนี้

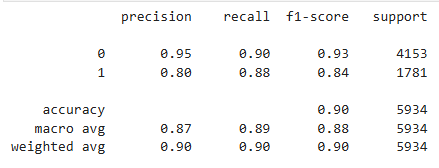
ผลลัพธ์จากการใช้ 5 Epochผลลัพธ์จากการใช้ 10 Epoch

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลชุดทดสอบดังนี้

ผลลัพธ์จากการใช้ 5 Epoch



ผลลัพธ์จากการใช้ 10 Epoch

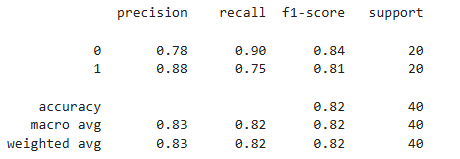


* ได้ค่า classification report เมื่อทดสอบกับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตดังนี้

ผลลัพธ์จากการใช้ 5 Epoch



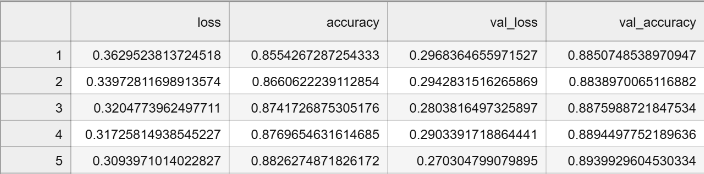
ผลลัพธ์จากการใช้ 10 Epoch



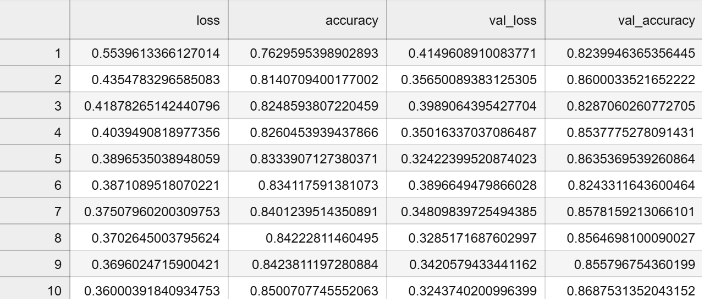
1. MoblienetV2 Fine-Tuned Classifier

* ได้ค่า loss, accuracy, val\_loss, val\_accuracy ดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

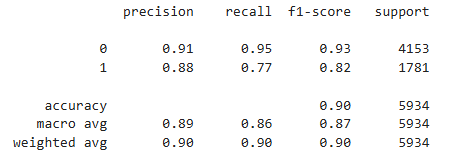
****

ผลลัพธ์จากการใช้ 10 Epoch

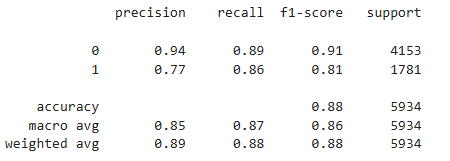
****

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลชุดทดสอบดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

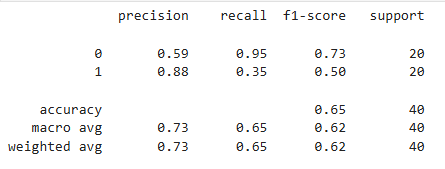
****

ผลลัพธ์จากการใช้ 10 Epoch

****

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

****

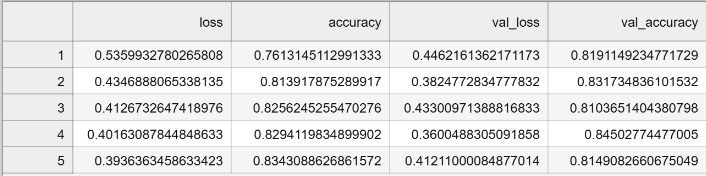
ผลลัพธ์จากการใช้ 10 Epoch



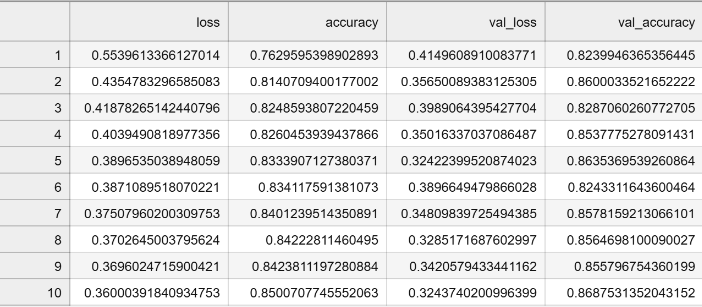
1. VGG19 Fine-Tuned Classifier

* ได้ค่า loss, accuracy, val\_loss, val\_accuracy ดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

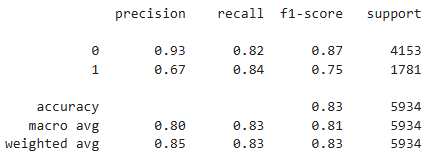
****

ผลลัพธ์จากการใช้ 10 Epoch

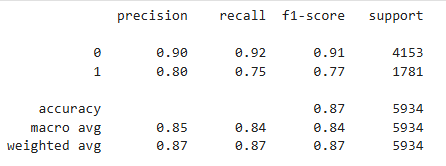
****

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลชุดทดสอบดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

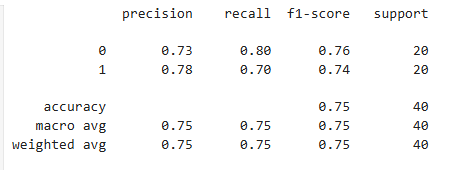
****

ผลลัพธ์จากการใช้ 10 Epoch

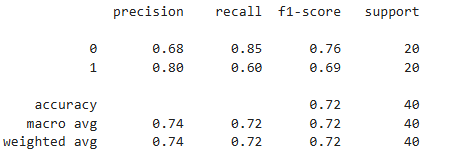
****

* ได้ค่า classification report เมื่อทดสอบกับข้อมูลภาพที่รวบรวมจากอินเตอร์เน็ตดังนี้

ผลลัพธ์จากการใช้ 5 Epoch

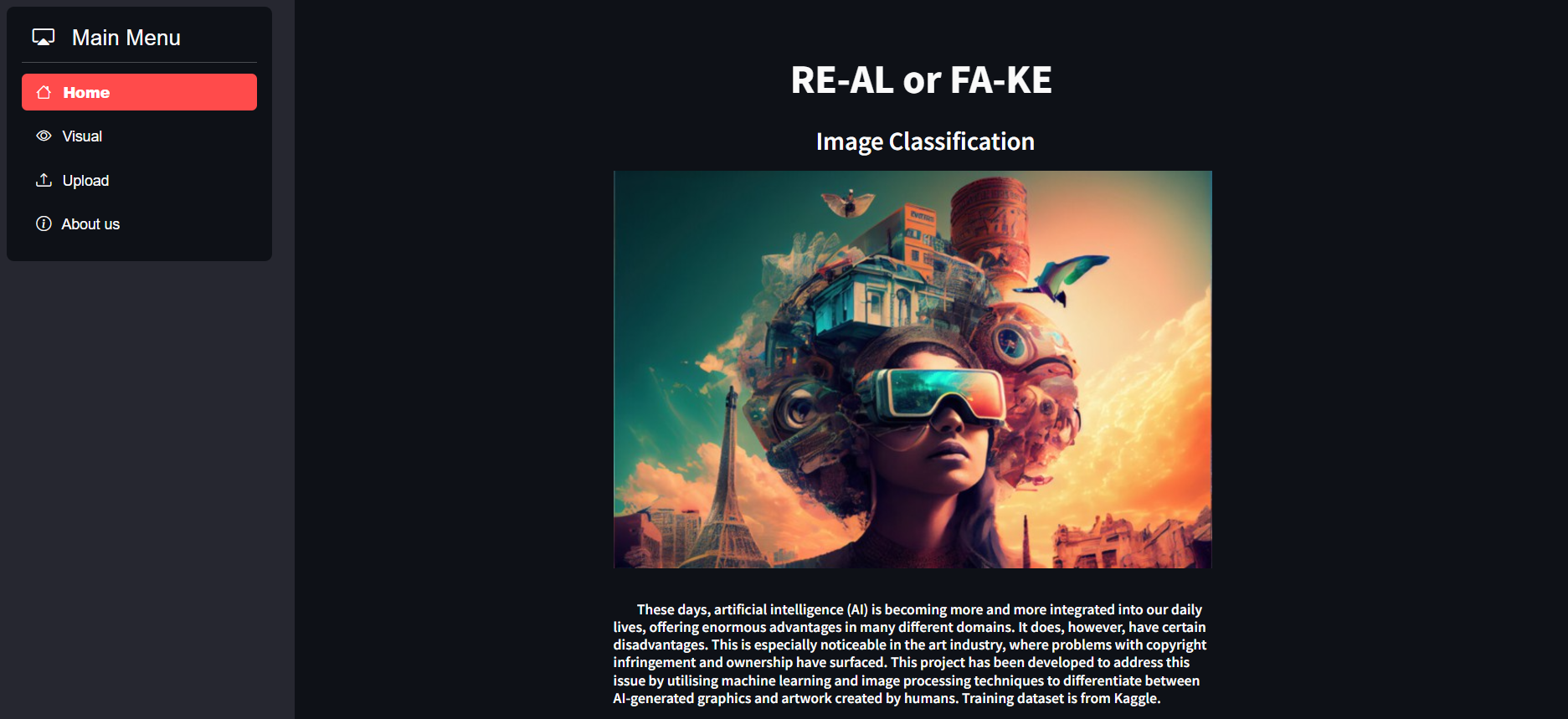
****

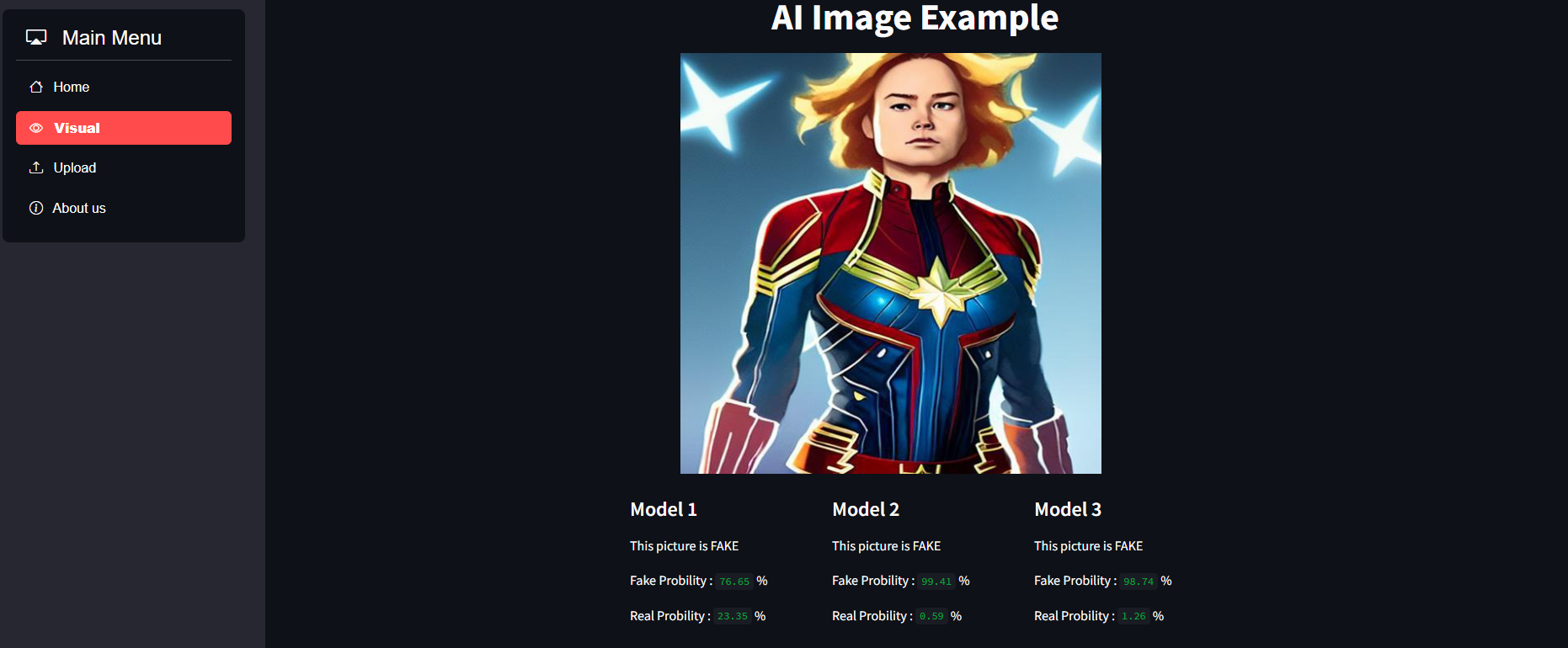
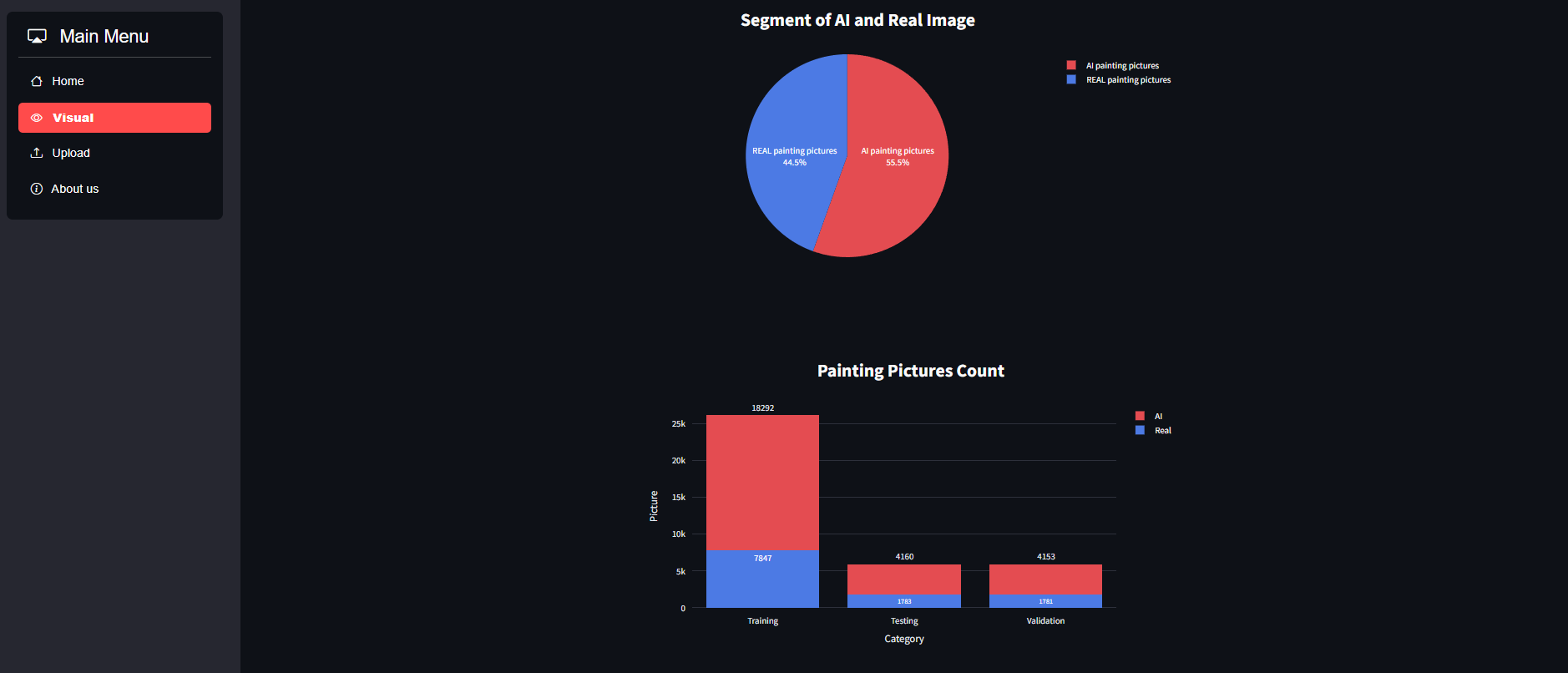
ผลลัพธ์จากการใช้ 10 Epoch

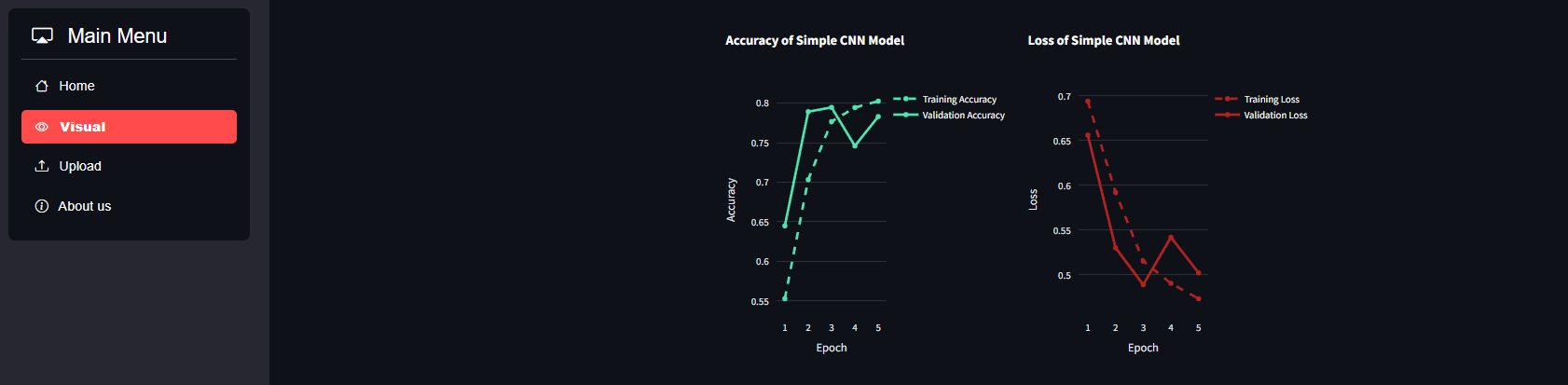
****

**Web-Application Section**

ได้เว็บแอพพลิเคชั่นที่แสดงผล 5 เมนูย่อยดังนี้

1. Home

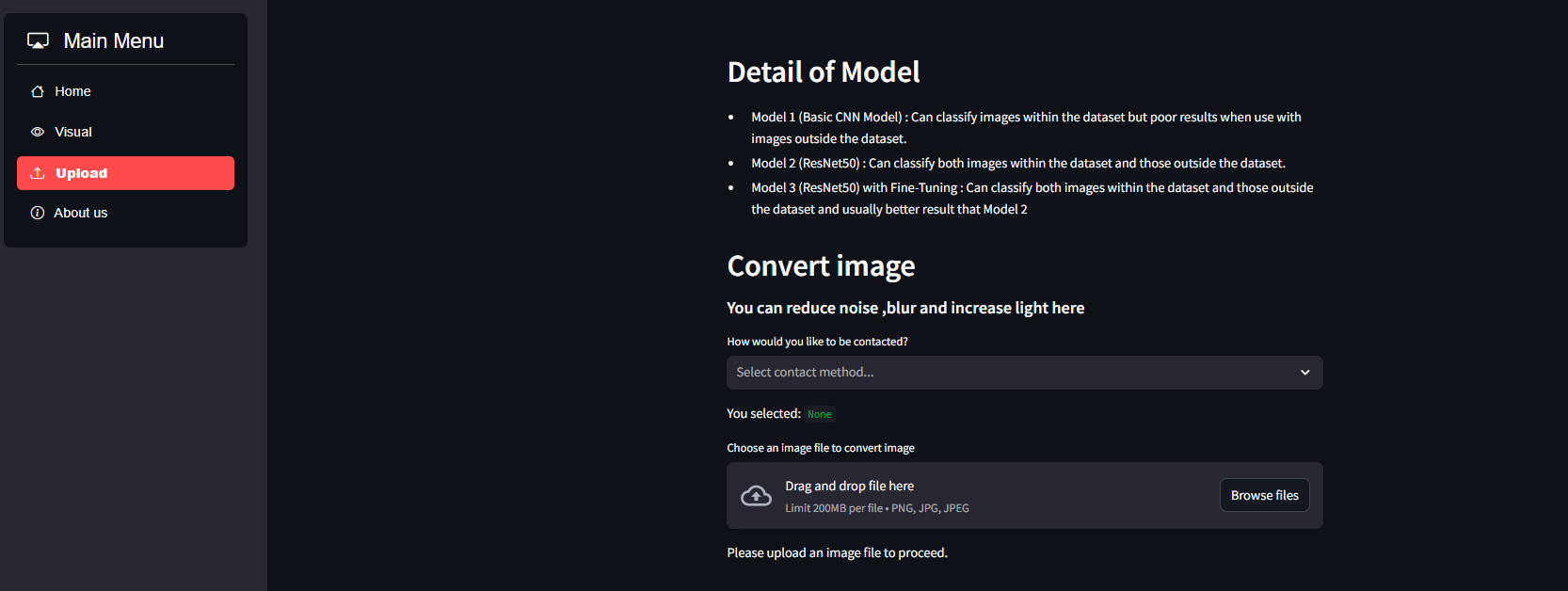
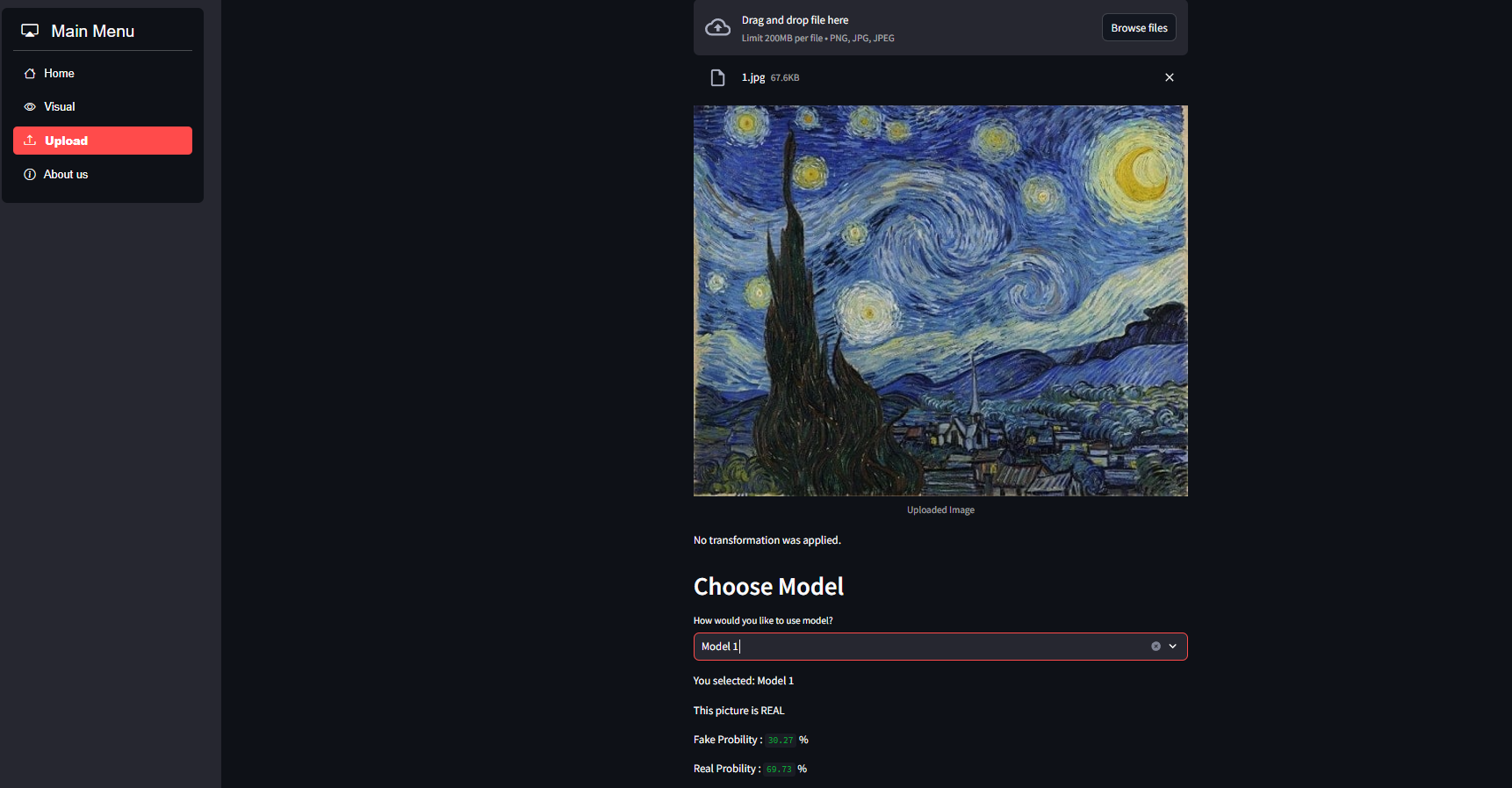
2. Visual

หมายเหตุ

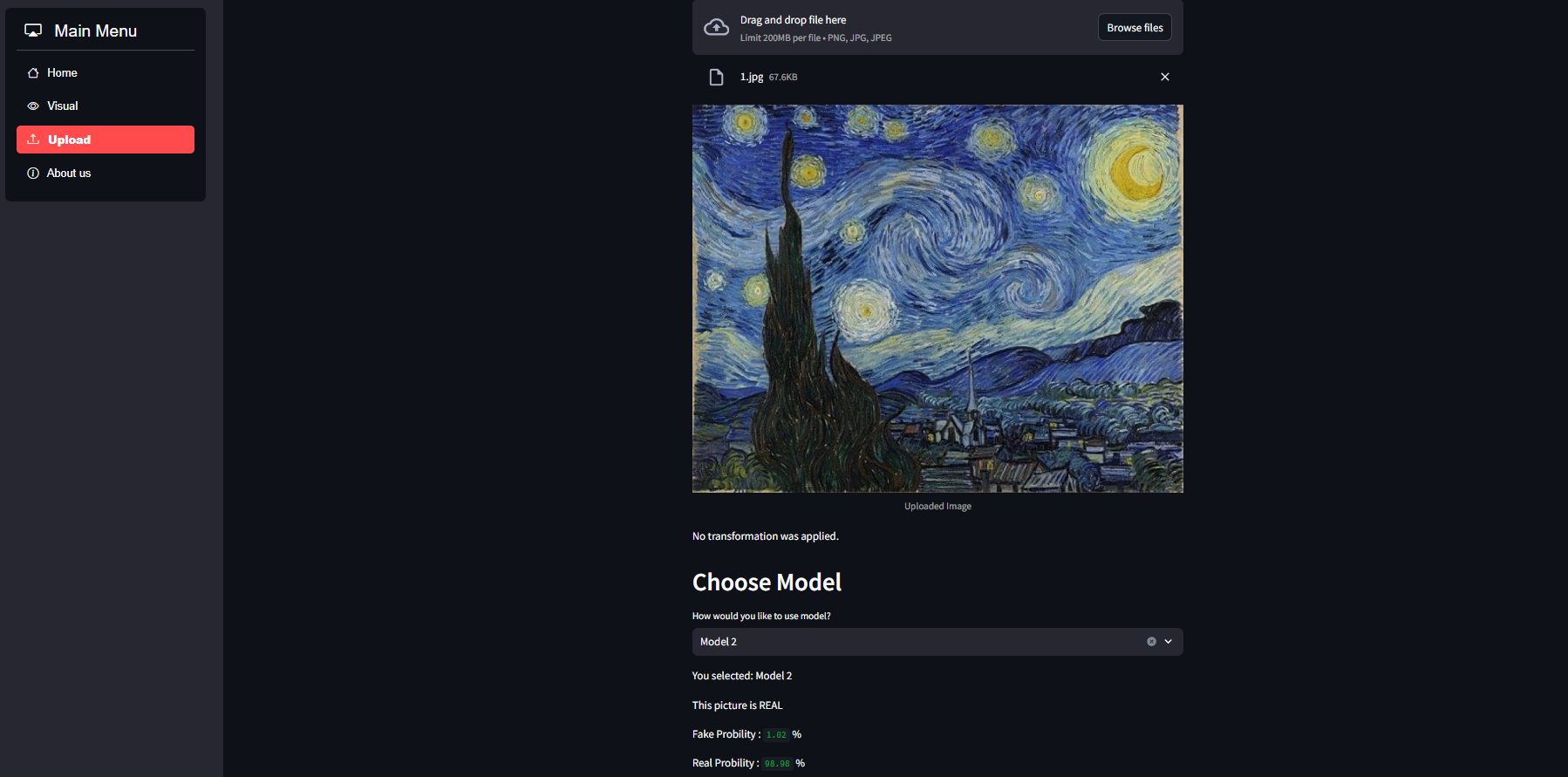
model1: Basic CNN Classifier

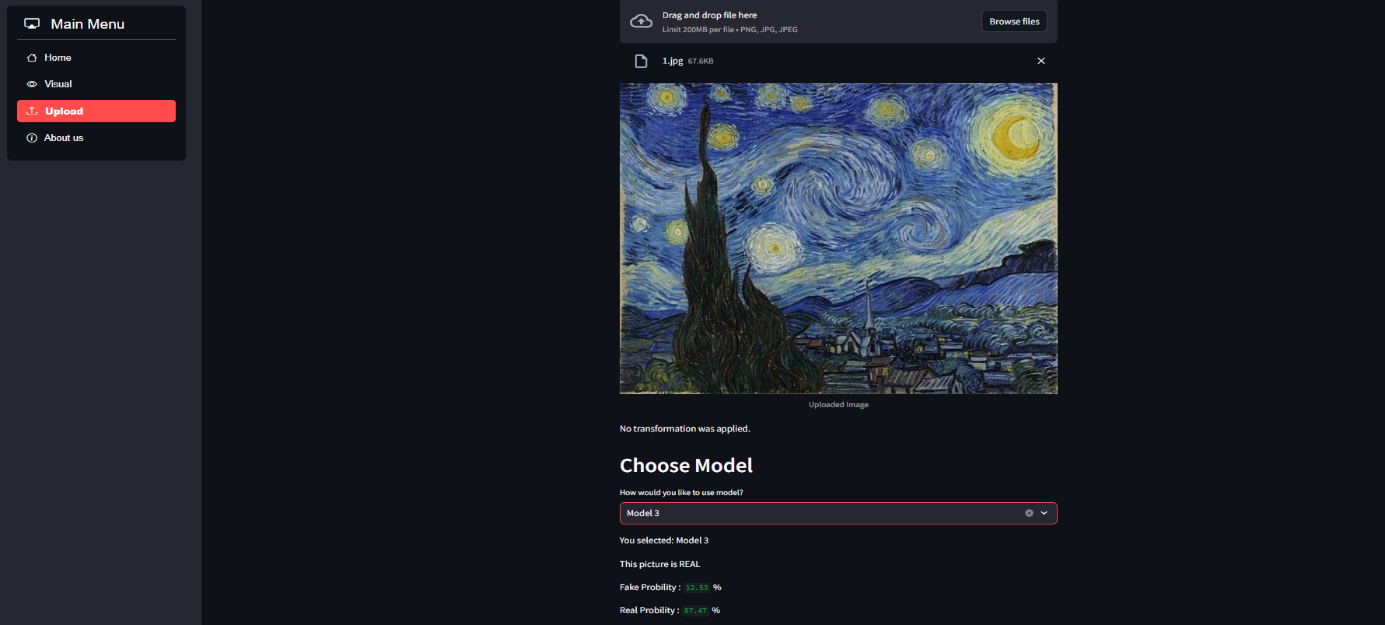
model2 : ResNet50 Classifier

model3 ResNet50 Fine-Tuned Classifier

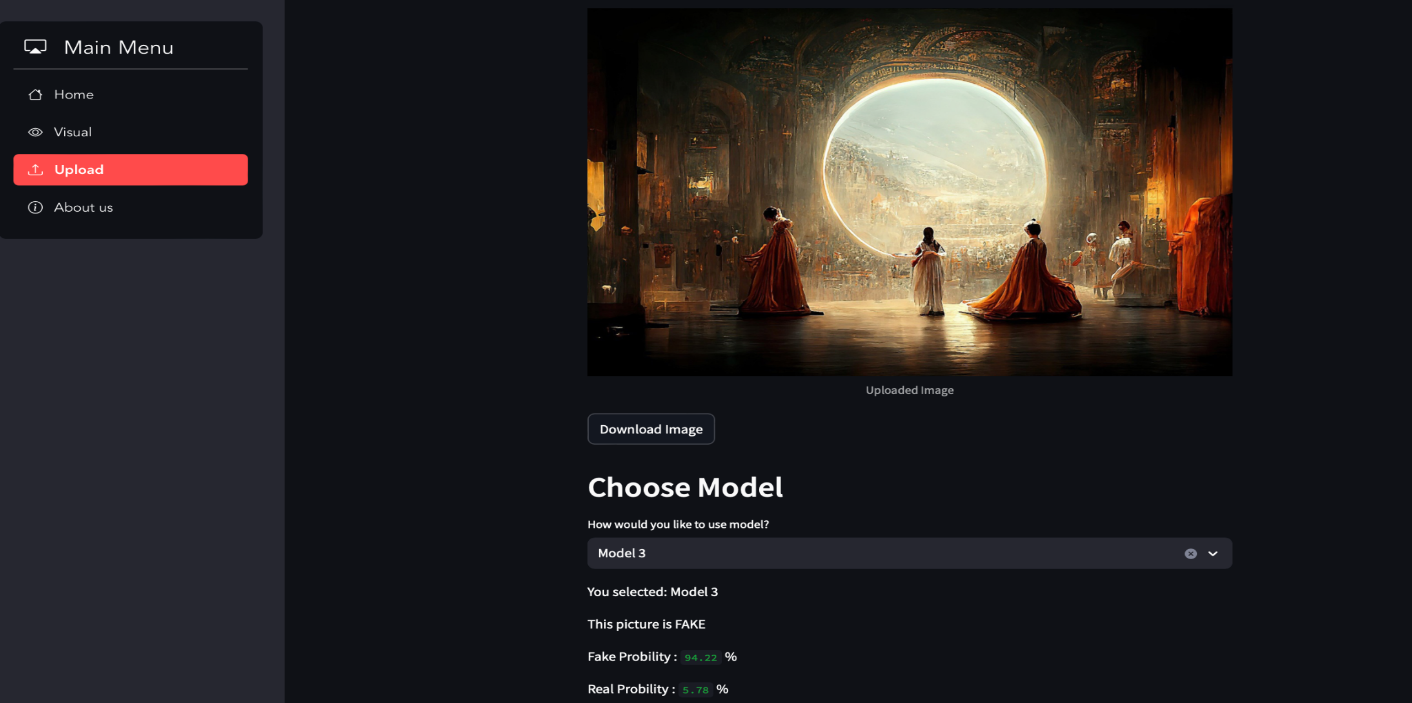
3.Upload

ภาพจากอินเตอร์เน็ตที่เป็นภาพจริง โดยเลือกใช้ model 1

ภาพจากอินเตอร์เน็ตที่เป็นภาพจริง โดยเลือกใช้ model 2

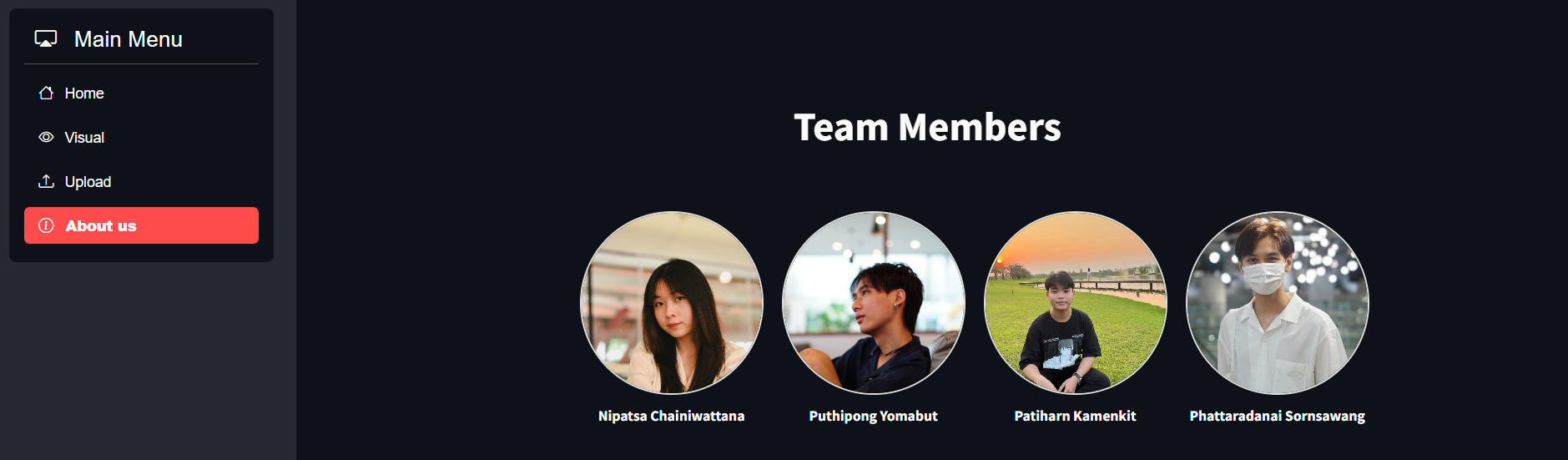


ภาพจากอินเตอร์เน็ตที่เป็นภาพจริง โดยเลือกใช้ model 3



ภาพจากอินเตอร์เน็ตที่เป็นภาพ AI โดยเลือกใช้ model 3

ภาพจริงถ่ายจากกล้องโทรศัพท์ที่ ATT 19 โดย Reduce Noise เเละเลือกใช้ model 3

4. About us

**Conclusion**

จากการทดลองพบว่าในโมเดลตัวที่ 1 (Basic CNN Classifier) และ 4 (MobileNetV2 Classifier) เมื่อฝึกฝนด้วยจำนวน Epoch ที่ 10 จะให้ความแม่นยำสูงกว่าการฝึกฝนด้วยจำนวน 5 Epoch เมื่อทดสอบกับชุดข้อมูลใน Dataset และ ภาพที่ไม่อยู่ใน Dataset แต่ในโมเดลตัวที่ 2 (ResNet50 Classifier) โมเดลตัวที่ 3 (ResNet50 Fine-Tuned Classifier) และโมเดลตัวที่ 5 (VGG19 Fine-Tuned Classifier) การฝึกฝนด้วยจำนวน Epoch ที่ 10 จะให้ความแม่นยำสูงกว่าการฝึกฝนด้วยจำนวน 5 Epoch เมื่อทดสอบกับชุดข้อมูลใน Dataset แต่เมื่อทดสอบกับภาพที่ไม่อยู่ใน Dataset พบว่าความแม่นยำในการ ทำนายคลาสของภาพจริงลดลงเมื่อฝึกฝนด้วยจำนวน Epoch ที่มากขึ้น เนื่องจากทั้ง 2 คลาสมีลักษณะ ของภาพที่คล้ายกันมากซึ่งส่งผลให้โมเดลจดจำลักษณะเฉพาะของภาพใน Dataset มามากเกินไปส่งผล ให้เมื่อเจอภาพที่ไม่เคยเห็นมาก่อนโมเดลจะไม่สามารถทำนายได้ โดยโมเดลที่ทำนายได้แม่นยำที่สุดคือ โมเดลตัวที่ 3 (ResNet50 Fine-Tuned Classifier) รองลงมาคือโมเดลตัวที่ 5 (VGG19 Classifier) ตาม ด้วยโมเดลตัวที่ 2 (ResNet50 Classifier) โมเดลตัวที่ 4 (MobileNetV2 Classifier) และโมเดลตัวที่ 1 (Basic CNN Classifier) ตามลำดับ โดยผู้จัดทำได้เลือกใช้โมเดลตัวที่ 1 (Basic CNN Classifier) ที่ผ่าน การฝึกฝนด้วยจำนวน 5 Epoch กับโมเดลตัวที่ 2 (ResNet50 Classifier) และโมเดลตัวที่ 3 (ResNet50 Fine-Tuned Classifier) ที่ผ่านการฝึกฝนด้วยจำนวน 10 Epoch สำหรับการใช้งานจริงในเว็บ แอปพลิเคชันที่ได้พัฒนาขึ้น

**Discussion and future work**

เนื่องจากข้อจำกัดด้านทรัพยากรคอมพิวเตอร์ที่ยังไม่สามารถใช้งานในการเทรนโมเดลด้วย Dataset ขนาดใหญ่ได้ โมเดลปัจจุบันจึงสามารถจำแนกได้เฉพาะภาพวาดจริงที่มาจากมนุษย์และภาพวาดที่สร้างโดย Generative AI เท่านั้น โดยโปรเจคนี้ยังสามารถพัฒนาให้มีความแม่นยำในการจำแนกสูงขึ้นกว่าที่เป็นอยู่ในปัจจุบันหากมีความ พร้อมของอุปกรณ์มากขึ้น ทางผู้จัดทำได้วางแผนที่จะพัฒนาโมเดลให้สามารถจำแนกภาพได้หลากหลายประเภท มากขึ้น โดยการเพิ่มจำนวนภาพใน Dataset ซึ่งอาจรวมถึงการจำแนกภาพถ่ายต่างๆและภาพวาดที่มาจากมนุษย์ และจาก Generative AI รวมถึงมีการ Preprocess มากขึ้นเพื่อความแม่นยำ นอกจากนี้ยังมีแผนที่จะพัฒนาโมเดล ให้สามารถใช้งานได้ในรูปแบบเว็บแอปพลิเคชันหรือแอปพลิเคชันที่สามารถให้ผู้ใช้บริการเข้ามาใช้งานได้

**References**

1. Zhu, M., Chen, H., Huang, M., Li, W., Hu, H., Hu, J., & Wang, Y. (2023, December 12). *GENDET: Towards good generalizations for AI-Generated Image Detection*. arXiv.org. https://arxiv.org/abs/2312.08880
2. Géron, A. (แปลโดย วิโรจน์ อัศวรังสี). (2024). *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow: แนวคิด เครื่องมือ และเทคนิคสำหรับสร้างระบบอัจฉริยะ*. กรุงเทพฯ: Core Function.
3. Nogibjj. (n.d.). *GitHub - nogibjj/Detecting-AI-Generated-Fake-Images*. GitHub. <https://github.com/nogibjj/Detecting-AI-Generated-Fake-Images>