การประยุกต์ใช้การประเมินภาพวาดมนุษย์โดยใช้ เทคนิคการเรียนรู้เชิงลึก

AN APPLICATION OF EVALUATION OF HUMAN SKETCHES USING DEEP LEARNING TECHNIQUE

ศรายุทธ ธิบดี

บัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒ

2563

การประยุกต์ใช้การประเมินภาพวาดมนุษย์โดยใช้ เทคนิคการเรียนรู้เชิงลึก

ศรายุทธ ธิบดี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ
คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒ
ปีการศึกษา 2563
ลิขสิทธิ์ของมหาวิทยาลัยศรีนครินทรวิโรฒ

AN APPLICATION OF EVALUATION OF HUMAN SKETCHES USING DEEP LEARNING TECHNIQUE

SARAYUT THIBHODEE

A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of MASTER OF SCIENCE

(Information Technology)

Faculty of Science, Srinakharinwirot University

2020

Copyright of Srinakharinwirot University

ปริญญานิพนธ์

เรื่อง

การประยุกต์ใช้การประเมินภาพวาดมนุษย์โดยใช้ เทคนิคการเรียนรู้เชิงลึก

ของ

ศรายุทธ ธิบดี

ได้รับอนุมัติจากบัณฑิตวิทยาลัยให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร

ปริญญาวิทยาศาสตรมหาบัณฑิต สาขาวิชาเทคโนโลยีสารสนเทศ

ของมหาวิทยาลัยศรีนครินทรวิโรฒ

(รองศาสตราจารย์ นายแพทย์ฉัตรชัย เอกปัญญาสกุล)

คณบดีบัณฑิตวิทยาลัย

คณะกรรมการสอบปากเปล่าปริญญานิพนธ์

..... ที่ปรึกษาหลัก

(ผู้ช่วยศาสตราจารย์ ดร.วราภรณ์ วิยานนท์)

..... ประธาน

(ผู้ช่วยศาสตราจารย์ ดร.อัครา ประโยชน์)

..... กรรมการ

(อาจารย์ ดร.ศิริสรรพ เหล่าหะเกียรติ)

ชื่อเรื่อง	การประยุกต์ใช้การประเมินภาพวาดมนุษย์โดยใช้ เทคนิคการเรียนรู้เชิงลึก
ผู้วิจัย	ศรายุทธ ธิบดี
ปริญญา	วิทยาศาสตรมหาบัณฑิต
ปีการศึกษา	2563
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ ดร. วราภรณ์ วิยานนท์

งานวิจัยนี้เป็นการศึกษาการประเมินการร่างภาพคนเต็มตัวโดยใช้หลักการประมาณท่าทางของมนุษย์ด้วยไลบรารีโอเพ่นโพสซึ่งเป็นไลบรารีสำหรับค้นหาจุดสำคัญ จำนวน 18 จุดตามโครงสร้างร่างกายมนุษย์ ชุดข้อมูลที่ใช้ในงานวิจัยนี้นำมาจากการร่างภาพของนักศึกษา จำนวน 22 คน ซึ่งวาดคนละ 3 แบบ จุดสำคัญที่ตรวจจับได้บนภาพที่ร่างนั้นนำมากำหนดคุณลักษณะเพิ่มติมได้แก่ ระยะห่างระหว่างจุดสำคัญซึ่งเป็นตัวแทนของความยาวของโครงสร้างร่างกายแต่ละส่วนและมุมองศาตามโครงสร้างร่างกายมนุษย์ ซึ่งสกัดข้อมูลได้จำนวน 26 คุณลักษณะ คุณลักษณะเหล่านี้นำไปสร้างแบบจำลองโดยใช้โครงข่ายประสาทเทียมเพื่อจำแนกระดับการร่างภาพโดยจำแนกออกเป็น 3 กลุ่ม ได้แก่ ระดับดี ระดับปานกลาง และระดับแย่ ผลการประเมินประสิทธิภาพของแบบจำลองพบว่าแบบจำลองมีความแม่นยำ 67.33% ค่าความเที่ยงตรง 70.00% ค่าการระลึก 66.67% และค่า F1 65.00%

คำสำคัญ : โครงข่ายประสาทเทียม, การประมาณค่าท่าทางของมนุษย์, โครงสร้างกระดูกของมนุษย์, การวาดภาพโครงร่าง

Title	AN APPLICATION OF EVALUATION OF HUMAN SKETCHES USING DEEP LEARNING TECHNIQUE
Author	SARAYUT THIBHODEE
Degree	MASTER OF SCIENCE
Academic Year	2020
Thesis Advisor	Assistant Professor Waraporn Viyanon , Ph.D.

This research is a study of the evaluation of full-body sketches and the principle of human pose estimation using the OpenPose library, a method to detect 18 keypoints on the human structure. The dataset used in this research was drawing sketches of 22 first-year students, each of whom had three drawings of three models. The detected keypoints were calculated to determine the angle and distance between keypoints, which had 26 features. These features were modeled using ANN for predicting the grades of drawings classified as good, moderate, and poor. The model performance evaluation results showed that it had an accuracy of 67.33%, a precision of 70.00%, a recall value of 66.67%, and an F1 value of 65.00%.

Keyword : Artificial Neutral Network, human pose estimation, human skeleton, drawing sketches

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้สำเร็จลุล่วงได้ด้วยความอนุเคราะห์จาก ผศ.ดร.วราภรณ์ วิยานนท์ อาจารย์ที่ปรึกษา ที่ได้ให้ความเมตตากรุณาเป็นที่ปรึกษาและให้ความช่วยเหลือชี้แนะแนวทางในสิ่งที่เป็นประโยชน์ต่อการศึกษาและการทำปริญญานิพนธ์นี้ด้วยความเอาใจใส่ตลอดมา ผู้วิจัยขอกราบขอบพระคุณเป็นอย่างสูงไว้ ณ โอกาสนี้

ขอกราบขอบพระคุณคณะกรรมการสอบปริญญานิพนธ์ ที่ได้ให้คำแนะนำและข้อเสนอแนะสำหรับการปรับปรุงปริญญานิพนธ์ และการใช้เทคนิคการเรียนรู้ของเครื่องเป็นเครื่องมือในการวิเคราะห์และแก้ไขปัญหาทางข้อมูลต่อไป

ขอกราบขอบพระคุณคณาจารย์และกรรมการบริหารหลักสูตรสาขาวิทยาการข้อมูล คณะวิทยาศาสตร์ มหาวิทยาลัยศรีนครินทรวิโรฒทุกท่าน ที่ได้กรุณาประสิทธิประสาทความรู้ต่าง ๆ ให้แก่ผู้วิจัย ตลอดจนให้ความช่วยเหลือในการทำวิจัยครั้งนี้

ขอขอบคุณทุนอุดหนุนการวิจัยในการนำเสนอผลงานจากบัณฑิตวิทยาลัย มหาวิทยาลัยศรีนครินทรวิโรฒที่ทำให้งานวิจัยสำเร็จลุล่วงไปได้ด้วยดี

ขอขอบคุณพี่ ๆ และเพื่อน ๆ สาขาวิชาวิทยาการข้อมูล รวมถึงบุคคลอีกหลายท่านที่ไม่ได้กล่าวนามไว้ ณ ที่นี้ที่ได้ให้ความช่วยเหลือและเป็นกำลังใจให้กับผู้วิจัยมาโดยตลอด

สุดท้ายนี้ ผู้วิจัยขอน้อมรำลึกถึงคุณของบิดามารดาและครูอาจารย์ ที่อบรมสั่งสอนให้ความรู้เป็นกำลังใจและให้การสนับสนุนผู้วิจัยด้วยดีตลอดมา

ศรายุทธ ธิบดี

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญ

ศิลปินที่ทำงานในสายอาชีพที่เกี่ยวกับ ศิลปะการเล่าเรื่อง ภาพยนตร์ การ์ตูน เกมและงานภาพประกอบนั้น ต้องมีทักษะในการวาดภาพที่ดี ซึ่งในการเรียนรู้การวาดภาพคนเต็มตัวอาจจะเป็นเรื่องยาก เพราะต้องศึกษาเกี่ยวกับสัดส่วนร่างกายมนุษย์ โดยมีตำแหน่งกระดูกเป็นส่วนประกอบหลัก [2] ซึ่งในการวาดภาพคนเต็มตัว (figure) เป็นภาพที่แสดงท่าทางของคนแบบเต็มตัวเพื่อแสดงตำแหน่งกระดูก และโครงสร้างของร่างกายคนเป็นหลัก โดยไม่เน้นการแสดงออกทางสีหน้าและอารมณ์ของต้นแบบ

ผู้เรียนในหลักสูตรหรือสายอาชีพที่เกี่ยวกับศิลปะ มีความจำเป็นในการฝึกฝนการวาดภาพคนเต็มตัวให้มีสัดส่วนที่ถูกต้องและตรวจสอบผลงานของตัวเองอยู่เสมอ ในการประยุกต์ใช้การประมาณท่าทางของมนุษย์ (human pose estimation) เป็นเครื่องมือของคอมพิวเตอร์วิทัศน์ (computer vision) ในการแปลงข้อต่อของมนุษย์จากภาพหรือวิดีโอ มาแสดงเป็นโครงสร้างกระดูก ซึ่งเป็นการตรวจจับท่าทางของบุคคลในภาพอัตโนมัติ [3] ซึ่งสามารถพัฒนาศักยภาพและความสามารถในทักษะที่จำเป็นของผู้เรียนในหลักสูตรหรือสายอาชีพที่เกี่ยวกับศิลปะได้แม่นยำ

อย่างไรก็ตามปัญหาในการฝึกวาดภาพคนเต็มตัวนั้น จำเป็นต้องมีทักษะพื้นฐานในการวัดสัดส่วน ก่อนที่จะเริ่มวาดภาพคนเต็มตัวต้องทำความรู้จักสัดส่วนร่างกายแต่ละส่วน เพื่อให้ผู้เรียนสามารถวาดภาพคนเต็มตัวได้อย่างถูกต้อง สิ่งนี้นำไปสู่ความจำเป็นที่ผู้วิจัยหาแนวทางแก้ไขปัญหาของการวาดภาพคนเต็มตัวไม่ถูกต้อง โดยใช้เทคโนโลยีปัญญาประดิษฐ์ เพื่อมาระบุตำแหน่งข้อต่อและโครงสร้างของร่างกายมนุษย์ และนำมาหาสัดส่วนองศาของโครงสร้างภาพวาดคนเต็มตัว เพื่อประเมินการวาดภาพคนเต็มตัว ซึ่งช่วยผู้สอนจำแนกระดับการวาดภาพ และ ผู้เรียนสามารถปรับปรุงการวาดภาพคนเต็มตัว

งานวิจัยนี้ ได้พัฒนาแบบจำลองการประเมินภาพวาดคนเต็มตัว โดยใช้วิธี ANN classification เพื่อจำแนกระดับการวาดภาพคนเต็มตัว โดยอาศัยเทคนิค human pose estimation ร่วมกับ Convolutional Neural Networks เพื่อหาตำแหน่ง keypoints ของภาพวาดคนเต็มตัวตามโครงสร้างของร่างกายมนุษย์ โดยใช้ไลบรารี OpenPose ในการตรวจจับ keypoints ซึ่งคาดว่าจะเป็นประโยชน์สำหรับนักศึกษาที่ฝึกวาดภาพคนเต็มตัว และใช้แบบจำลองการจำแนกระดับการวาดภาพคนเต็มตัวนี้เปรียบเทียบผลงานภาพวาดคนเต็มตัวของตนเองว่า อยู่ในระดับใด ระหว่าง ดี ปานกลาง และแย่

1.2 ความมุ่งหมายของงานวิจัย

งานวิจัยนี้ผู้วิจัยได้กำหนดความมุ่งหมายไว้ดังนี้

1. เพื่อกำหนดคุณลักษณะสำคัญของภาพวาดคนเต็มตัวสำหรับใช้สร้างแบบจำลองเพื่อประเมินการวาดภาพคนเต็มตัว

2. เพื่อสร้างแบบจำลองประเมินการวาดภาพคนเต็มตัวของแต่ละกลุ่ม ว่าภาพวาดของนักศึกษาอยู่ในกลุ่มใด

1.3 ความสำคัญของการวิจัย

ในปัจจุบันการทำงานในสายอาชีพที่เกี่ยวกับศิลปะ ต้องมีทักษะในการวาดภาพ โดยการเรียนรู้การวาดภาพคนเต็มตัวเป็นส่วนหนึ่งของงานด้านศิลปะ ซึ่งการวาดภาพคนเต็มตัวนั้นต้องศึกษาเกี่ยวกับสัดส่วนพื้นฐานของร่างกายมนุษย์ โดยมีส่วนประกอบทั้งหมด 7 ส่วน ได้แก่ ศีรษะ แขนทั้ง 2 ข้าง กระดูกซี่โครง กระดูกเชิงกราน และขาทั้ง 2 ข้าง เพื่อประกอบเป็นโครงสร้างของร่างกายมนุษย์

อย่างไรก็ตามปัญหาที่เกิดขึ้นจากการวาดภาพคนเต็มตัวนั้น มีด้วยกันหลายประการ ได้แก่ การวาดขายาวเกินไป วาดแขนสั้นไป วาดโครงสร้างไม่สัมพันธ์กับกล้ามเนื้อ วาดเทียบระยะระหว่างศีรษะกับลำตัวและลำตัวกับเท้าไม่สัมพันธ์กัน เป็นต้น ด้วยปัญหาเหล่านี้จึงจำเป็นต้องมีทักษะพื้นฐานการวาดคนเต็มตัว โดยเริ่มจากศึกษาโครงสร้างร่างกายมนุษย์ และทำความรู้จักสัดส่วนร่างกายแต่ละส่วนว่ามีส่วนประกอบอะไรบ้าง เพื่อให้สามารถฝึกฝนการวาดภาพคนเต็มตัวได้อย่างถูกต้อง

แนวทางในการแก้ไขปัญหาเริ่มจากการศึกษาสัดส่วนร่างกายมนุษย์ และทำการฝึกฝนวาดภาพคนเต็มตัวให้มีสัดส่วนที่ถูกต้องและสามารถตรวจสอบการวาดภาพได้ โดยใช้การประยุกต์เครื่องมือคอมพิวเตอร์วิทัศน์ ในการประมาณท่าทางของมนุษย์ จากการแปลงข้อต่อมนุษย์ที่คอมพิวเตอร์สามารถประมวลผลภาพ โดยนำภาพวาดคนเต็มตัว ใช้ไลบรารี OpenPose ในการสกัดหาตำแหน่งจุดสำคัญ ทั้ง 18 จุด ได้แก่ จมูก คอไหล่ขวา ข้อศอกขวา ข้อมือขวา ไหล่ซ้าย ข้อศอกซ้าย ข้อมือซ้าย เอวขวา เข่าขวา ข้อเท้าขวา เอวซ้าย เข่าซ้าย ข้อเท้าซ้าย ตาขวา ตาซ้าย หูขวา หูซ้าย ออกมาเป็นจุดสำคัญที่อยู่ตามตำแหน่งร่างกายมนุษย์แต่ละส่วนและแสดงเป็นโครงสร้างกระดูกบนภาพวาดตามโครงสร้างร่างกายมนุษย์ หลังจากได้ข้อมูลของจุดสำคัญในภาพวาด ขั้นตอนต่อไปเป็นใช้เครื่องมือในการจำแนกภาพวาดคนเต็มตัว ซึ่งการจำแนกภาพวาดคนเต็มตัว เริ่มจากให้ผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัว ตรวจผลงานนักศึกษาโดยแบ่งตามกลุ่มคะแนน หลังจากแบ่งคะแนนตามกลุ่มเรียบร้อย นำมาสกัดคุณลักษณะเพื่อใช้ในการสร้าง

แบบจำลอง โดยผู้วิจัยแบ่งคุณลักษณะตามสัดส่วนทั้ง 7 ส่วนซึ่งรูปแบบการสกัดคุณลักษณะได้แก่ มุมจากจุดสำคัญ และระยะห่างระหว่างจุดสำคัญ ซึ่งรวมทั้งสิ้น 26 คุณลักษณะ จากนั้นใช้เทคนิค Artificial Neural Network (ANN) ในการสร้างแบบจำลองการทำนายผลจากกลุ่มภาพวาดคนเต็มตัว โดยใช้ข้อมูลภาพวาดคนเต็มตัวในการ training แบบจำลอง ซึ่งสามารถจำแนกภาพวาดได้ 3 กลุ่ม ได้แก่ ดี ปานกลาง และแย่ และผลลัพธ์ที่ออกมา สามารถจำแนกตามกลุ่มที่ระบุไว้พร้อมกับค่าเปอร์เซ็นต์ของคะแนนความเชื่อมั่น (Confidence Score) เพื่อระบุภาพวาดที่นักศึกษาวาดนั้นอยู่กลุ่มภาพวาดใด

งานวิจัยนี้ได้พัฒนาแบบจำลองการประเมินภาพวาดคนเต็มตัว โดยศึกษาเทคนิคการวาดภาพคนเต็มตัว ไลบรารี OpenPose และการจำแนกด้วย ANN Classification โดยคาดหวังว่าแบบจำลองการจำแนกระดับการวาดภาพคนเต็มตัวนี้ เพื่อเปรียบเทียบภาพวาดของตนเองว่าอยู่ในระดับใดระหว่าง ดี ปานกลาง และแย่

1.4 ขอบเขตของการวิจัย

ประชากรที่ใช้ในการวิจัย

ข้อมูลที่ใช้ในการวิจัยเป็นผลงานการวาดภาพคนเต็มตัวของ รายวิชา 802106 : DRAWING FOR INFORMATION TECHNOLOGY II ซึ่งเป็นนักศึกษาคณะเทคโนโลยีสารสนเทศและการสื่อสาร มหาวิทยาลัยศิลปากร สาขาเทคโนโลยีสารสนเทศเพื่อการออกแบบชั้นปีที่ 1 เอกแอนิเมชัน ปีการศึกษา 2/2562

กลุ่มตัวอย่างที่ใช้ในการวิจัย

1. ตัวแปรอิสระ แบ่งเป็นดังนี้

1.1 ภาพวาดที่นักศึกษาวาด จำนวน 66 ภาพ

1.2 ภาพนายแบบหรือนางแบบ จำนวน 3 คน

1.3 ระยะห่างระหว่าง นายแบบ กับ ภาพวาดของนักศึกษา

1.4 ตำแหน่งองศาระหว่าง ตำแหน่งที่นายแบบยืน กับ ตำแหน่งที่นักศึกษายืนวาดรูปต้องอยู่ด้านหน้าตรงของนายแบบ และมุมองศาออกจากตำแหน่งที่นายแบบยืนข้างละ 30 องศา รวมเป็น 60 องศา

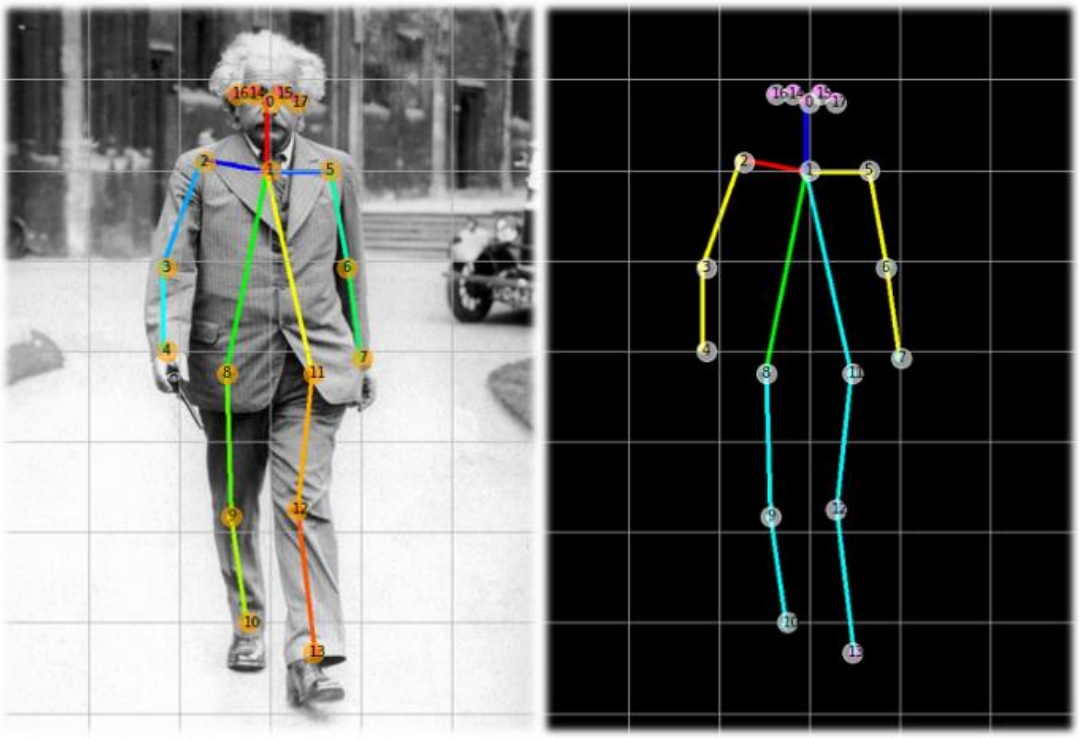
2. ตัวแปรตาม

2.1 ตำแหน่งของจุดสำคัญ (keypoints) ของ human pose estimation

2.2 คุณลักษณะ (feature) ของมุมและระยะห่างระหว่างจุดสำคัญที่เปรียบเทียบแต่ละส่วนของภาพวาดคนเต็มตัว

1.5 กรอบแนวคิดในงานวิจัย

งานวิจัยนี้ชิ้นนี้ใช้เทคนิคโครงข่ายประสาทเทียมเชิงลึก(deep learning) ในการประมวลผลภาพที่ทำงานบนโปรแกรมคอมพิวเตอร์ โดยระบบจะนำรูปภาพที่ถ่าย มาประมวลผลภาพโดยใช้เทคนิคโครงข่ายประสาทแบบคอนโวลูชัน (Convolutional Neural Network : CNN) เพื่อใช้ในการแยกคุณลักษณะของรูปภาพ (feature) ซึ่งเกี่ยวข้องกับรูปภาพ โดยหลักการทำงานของ human pose estimation ใช้ไลบรารี OpenPose โดยพัฒนาจาก มหาวิทยาลัย Carnegie Mellon University (CMU) [4] นำมาใช้ในด้านการประมาณท่าทาง (pose estimation) ซึ่งสามารถตรวจหาจุดสำคัญบนภาพได้หลายรูปแบบ เช่น ร่างกาย เท้า ใบหน้า และมือ ซึ่งสามารถระบุตำแหน่งข้อต่อของร่างกายมนุษย์ ซึ่งตำแหน่งจุดสำคัญที่ระบุในภาพสามารถระบุชิ้นส่วนร่างกายได้ดังนี้ P0 = nose, P1=neck, P2=right shoulder, P3=right elbow, P4=right wrist, P5=left shoulder, P6=left elbow, P7=left wrist, P8=right hip, P9=right knee, P10=right ankle, P11=left hip, P12=left knee, P13=left ankle, P14=right eye, P15=left eye, P16=right ear, P17=left ear นอกเหนือจากข้อต่อทั้ง 18 จุด ซึ่งเป็นผลลัพธ์ที่ได้จากการประมวลผลรูปภาพ โดยเริ่มจากการอินพุตรูปภาพ และเอาต์พุตที่ได้จะระบุตำแหน่งจุดสำคัญทั้ง 18 จุดในรูปภาพ ดังภาพประกอบ 1 และใช้เทคนิค Artificial Neural Network (ANN) ในการสร้างแบบจำลองทำนายผลซึ่งรูปแบบของการสกัดหาคุณลักษณะในการจำแนกแต่ละกลุ่ม คือการใช้คุณลักษณะมุมของจุดสำคัญและคุณลักษณะระยะห่างระหว่างจุดสำคัญ เพื่อใช้ในการทำนายผลของการการแบ่งกลุ่มการประเมินผลการวาดภาพทั้ง 3 กลุ่ม ได้แก่ ดี ปานกลาง และแย่เมื่อได้ผลลัพธ์เป็นที่เรียบร้อย ผู้วิจัยต้องวัดประสิทธิภาพของแบบจำลองว่ามีความเหมาะสมกับผลลัพธ์ที่ได้อย่างไร



ภาพประกอบ 1 ผลลัพธ์ของ human pose estimation ในการระบุจุดสำคัญทั้ง 18 จุด

ที่มา : Marcelo Rovai, (Aug 4,2020). Realtime Multiple Person 2D Pose Estimation using TensorFlow2.x สืบค้นจาก https://towardsdatascience.com/realtime-multiple-person-2d-pose-estimation-using-tensorflow2-x-93e4c156d45f

1.6 สมมุติฐานในการวิจัย

1. แบบจำลองที่ได้จากงานวิจัยนี้เป็นเครื่องมือในการแบ่งกลุ่มการวาดภาพของแต่ละกลุ่มการวาดรูปภาพคนเต็มตัวให้กับนักศึกษาทราบ

2. ไลบรารี OpenPose เป็นเครื่องมือในการหาจุดสำคัญที่มีประสิทธิภาพดีที่สุด

3. การสกัดคุณลักษณะจากจุดสำคัญนอกเหนือจากมุมและระยะห่างระหว่างจุดสำคัญอาจมีวิธีที่สามารถสกัดคุณลักษณะรูปแบบอื่นได้

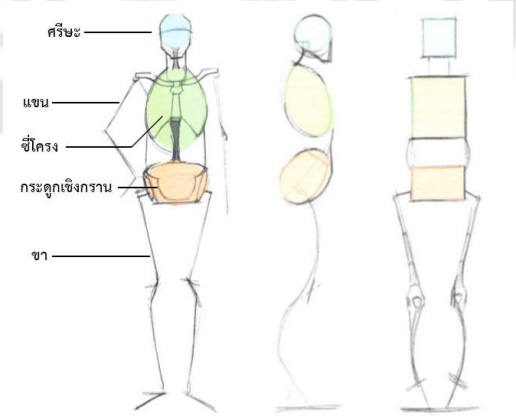
บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

ในงานวิจัยครั้งนี้ ผู้วิจัยได้ศึกษาเอกสารและงานวิจัยที่เกี่ยวข้อง และได้นำเสนอตามหัวข้อต่อไปนี้

1. สัดส่วนร่างกายมนุษย์
2. การเรียนรู้เชิงลึก
3. การประมาณท่าทางของมนุษย์
4. การตรวจจับจุดสำคัญ
5. ไลบรารี MoblieNets
6. การสร้างแบบจำลองด้วย ANN
7. งานวิจัยที่เกี่ยวข้อง

2.1 **สัดส่วนร่างกายมนุษย์**

การวาดภาพคนเต็มตัว จำเป็นต้องเข้าใจโครงสร้างของมนุษย์ โดยโครงสร้างของร่างกายมนุษย์ออกเป็น 7 ส่วน ได้แก่ ศีรษะ กระดูกซี่โครง แขน 2 ข้าง กระดูกเชิงกราน และขา 2 ข้าง ดังภาพประกอบ 2 ซึ่งแต่ละส่วนที่กล่าวมาเป็นองค์ประกอบสำคัญ ในการวาดภาพคนเต็มตัว [5]



ภาพประกอบ 2 ส่วนประกอบของร่างกาย

ที่มา : Hampton, M. (2009). Figure Drawing: Design and Invention (2nd edition). United States: Michael Hampton.

นิยามของสัดส่วน คือ ขนาดสัมพัทธ์ของร่างกายที่มีความสัมพันธ์ระหว่างกัน โดยใช้กระดูกสันหลังเป็นเกณฑ์พื้นฐานในการสังเกตความถูกต้องของสัดส่วน โดยสัดส่วนของมนุษย์ขณะยืนตรง แบ่งออกเป็น 4 ส่วน ได้แก่ A) จากศีรษะถึงบ่า B) จากบ่าถึงก้น C) จากก้นถึงน่อง D) จากน่องถึงเท้า [6] ตามลำดับ ดังภาพประกอบ 3 เป็นพื้นฐานการแบ่งแต่ละส่วนของร่างกายมนุษย์

ภาพประกอบ 3 ท่าทางยืนและสัดส่วน A) จากศีรษะถึงหลัง B) จากคอถึงสะโพก C) จากสะโพกถึงเข่า D) จากเข่าถึงเท้า

ที่มา : Hart, C. (2014). Figure It Out! Human Proportions: Draw the Head and Figure Right Every Time (Illustrated edition). New York: Drawing with Christopher Hart.

2.2 การเรียนรู้เชิงลึก

การเรียนรู้เชิงลึกเป็นสาขาหนึ่งของ machine learning โดยการเรียนรู้เชิงลึกนั้นถูกใช้อย่างกว้างขวาง เช่น การตรวจวัตถุ (object recognition) การจดจำรูปแบบ (pattern recognition) การประมวลผลภาษาธรรมชาติ (natural language processing) และการประมวลผลรูปภาพ (image processing)

นักวิจัยด้าน machine learning ได้เสนอสถาปัตยกรรมการเรียนรู้หลายแบบบนหลักการของการเรียนรู้เชิงลึก ได้แก่ deep artificial neural networks, convolutional neural networks, deep belief networks และ recurrent neural network ซึ่งมีการนำมาใช้งานอย่างแพร่หลาย

ในทางคอมพิวเตอร์วิทัศน์ (computer vision) หลายด้าน เช่น การจดจำเสียงพูด การประมวลผลภาษาธรรมชาติ การจดจำเสียง [7] ซึ่งในงานวิจัยจะพูดถึง 2 หัวข้อได้แก่ นิวรอนเน็ตเวิร์ค และนิวรอนเน็ตเวิร์คคอนโวลูชัน

2.2.1 **นิวรอนเน็ตเวิร์ค**

นิวรอนเน็ตเวิร์ค (neural networks) เป็นระบบที่พัฒนาขึ้นเสมือนกับการทำงานของระบบสมอง ซึ่งมีกระบวนการเรียนรู้ของมนุษย์ เพื่อให้เรียนรู้รูปแบบการจดจำแบบเดียวกับการทำงานของสมองมนุษย์ ซึ่งกระบวนการทำงานนำข้อมูลเข้าไปยังโครงข่ายประสาทเทียม จะนำไปชั้นที่ซ่อน (hidden layer) เพื่อทำการประมวลผลออกมาเป็นผลลัพธ์ของข้อมูล

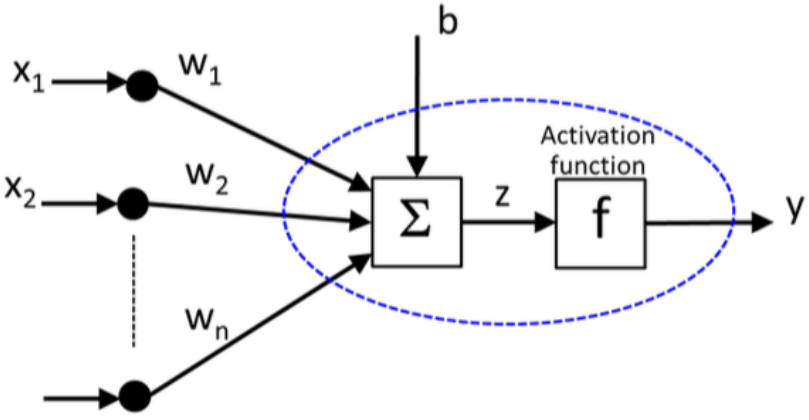
โครงข่ายประสาทเทียมมีหน่วยการทำงานที่เล็กที่สุดเรียกว่า นิวรอน (neural) มีโครงสร้างดังภาพประกอบ 4 ทำหน้าที่แยกผลที่ได้จากแต่ละกลุ่มออกจากกัน และมีฟังก์ชันการทำงาน ดังแสดงในสมการที่ (1)

$$f(x) = \sum_{i=1}^{n} W_i X_i + b \qquad (1)$$

โดย W แทนค่าน้ำหนัก

b แทนค่าความเอนเอียง

n แทนจำนวนของข้อมูลนำเข้า

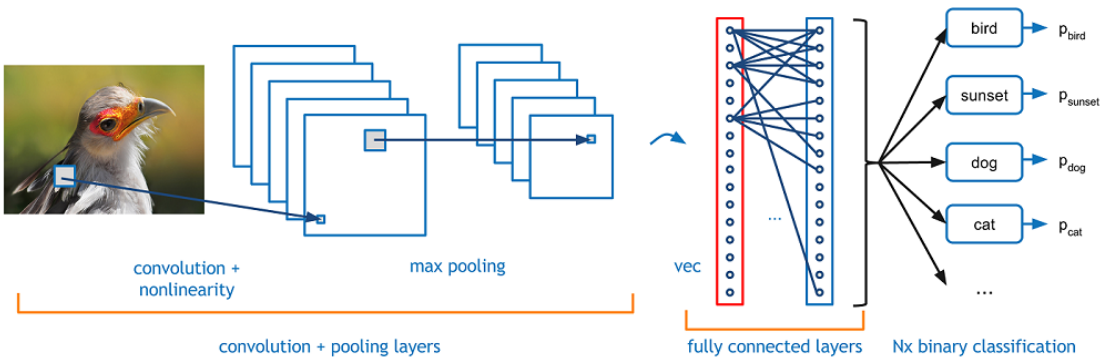


ภาพประกอบ 4 โครงสร้างของนิวรอนเน็ตเวิร์ค

ที่มา savannah logan, Nov 28, 2017. Understanding the Structure of Neural Networks สืบค้นจาก https://becominghuman.ai/understanding-the-structure-of-neural-networks-1fa5bd17fef0

2.2.2 นิวรอนเน็ตเวิร์คคอนโวลูชัน

นิวรอนเน็ตเวิร์คคอนโวลูชัน (convolutional neural network) เป็นโครงข่ายประสาทเทียมเชิงลึกประเภทหนึ่งมีจุดเริ่มต้นมาจากงานวิจัยทางด้านการรับรู้การจดจำภาพ โดยมักจะใช้ข้อมูลรับเข้าจากรูปภาพแปลงเป็นเมทริกซ์แบบ 2 มิติหรือ 3 มิติในขณะที่โครงข่ายประสาทเทียมแบบธรรมดาเหมาะกับข้อมูลนำเข้า 1 มิติซึ่งโครงสร้างของโครงข่ายประสาทเทียมคอนโวลูชันมีได้หลายรูปแบบ [7] ดังภาพประกอบ 5



ภาพประกอบ 5 โครงข่ายประสาทเทียมคอนโวลูชัน

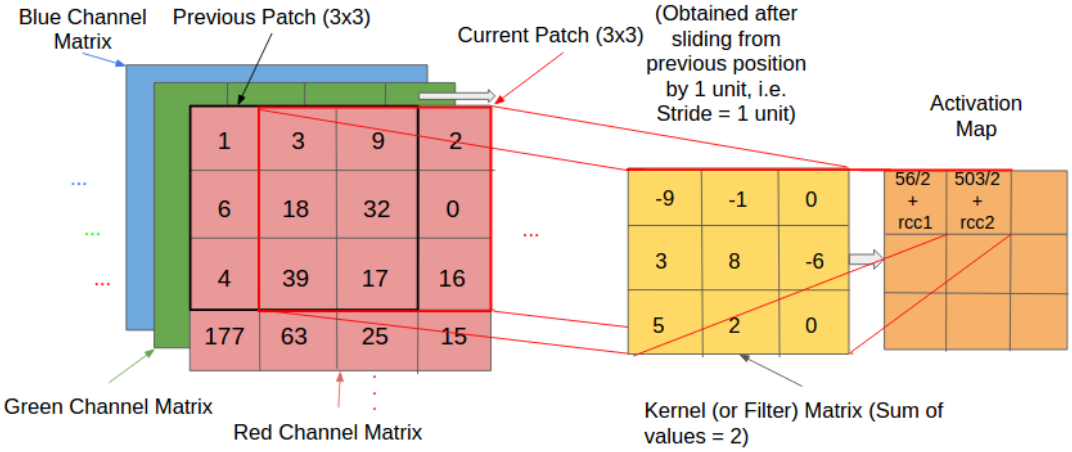
ที่มา dshahid380, Feb 25, 2019. Convolutional Neural Network สืบค้นจาก https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529

โครงสร้างของโครงข่ายประสาทเทียมคอนโวลูชัน นั้นถูกออกแบบมาเพื่อการเพิ่มความสามารถในการสกัดคุณลักษณะสำหรับการจำแนก (classification) ที่เกี่ยวข้องกับรูปภาพ อีกทั้งในปัจจุบันถูกนำแนวคิดไปใช้ต่อยอดมากมาย แต่ยังต้องพึ่งมนุษย์ในการออกแบบสถาปัตยกรรมของ neural network โดยประกอบด้วย 4 ส่วน ได้แก่ ชั้นคอนโวลูชัน ชั้นพูลลิ่ง ชนิดของการทำคอนโวลูชัน และชั้นการเชื่อมโยงเต็มรูปแบบ

1.ชั้นคอนโวลูชัน

ชั้นคอนโวลูชัน (convolutional layer) เป็นชั้นที่ทำหน้าที่หาคุณลักษณะจากกลุ่มของข้อมูลรูปภาพ ซึ่งการทำงานของ convolutional layer เป็นการคำนวณเพื่อหาชั้นของผลลัพธ์

ซึ่งเรียกว่า feature map ด้วยการนำพื้นที่ส่วนย่อยรูปภาพ (sub-region) ไปคำนวณแบบ dot product กับเคอร์นอล (kernel) โดย kernel ที่นำมาคำนวณต้องมีขนาดเล็กกว่ารูปภาพ ซึ่งการคำนวณของ convolutional layer [1] ดังภาพประกอบ 6



ภาพประกอบ 6 ชั้นคอนโวลูชัน (convolution layer)

ที่มา Abhineet Saxena, June 29, 2016. Convolutional Neural Networks (CNNs): An Illustrated Explanation สืบค้นจาก https://blog.xrds.acm.org/wp-content/uploads/2016/06/Figure_2.png

จากภาพประกอบ 6 แสดงให้เห็นลักษณะสีที่เป็นตารางค่าเมทริกซ์ของช่องค่าสีทั้ง 3 สี ได้แก่ เมทริกซ์ช่องสีน้ำเงิน เมทริกซ์ช่องสีเขียว และเมทริกซ์ช่องสีแดง โดยผ่านตัวกรองเมทริกซ์ที่เรียกว่า เคอร์เนล โดยจากภาพจะมีตัวกรองขนาด 3 x 3 เพื่อนำมาคำนวณให้ภาพมีขนาดเล็กลงกว่ารูปภาพเดิม

2.ชั้นพูลลิ่ง

ชั้นพูลลิ่ง (pooling layer) มีจุดประสงค์เพื่อทำการลดขนาดของข้อมูลที่ผ่านการคอนโวลูชันมาแล้วนิยมนำมาทำต่อกับชั้นคอนโวลูชัน แต่ก็อาจไม่จำเป็นต้องนำมาต่อกันเสมอไป ซึ่งจะขึ้นอยู่กับการออกแบบสถาปัตยกรรม [1] ซึ่งการพูลลิ่งที่เป็นที่นิยมมีด้วยกัน 2 วิธี คือ การพูลลิ่งแบบหาค่ามากที่สุด และการพูลลิ่งแบบหาค่าเฉลี่ย

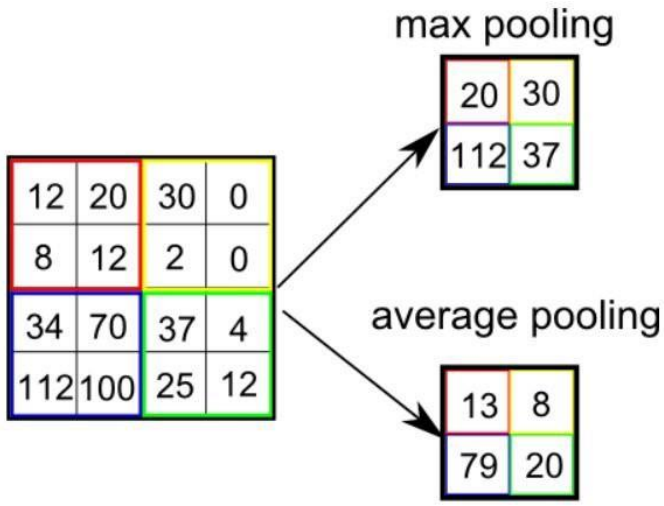
2.1 การทำพูลลิ่งแบบหาค่ามากที่สุด

การทำพูลลิ่งแบบหาค่ามากที่สุด (max pooling) เป็นวิธีที่นิยมแพร่หลายมากในงานวิจัยด้านคอนโวลูชันแนลโครงข่ายประสาทเทียมในปัจจุบัน โดยหลักการทำงานเมื่อ

ภาพถูกแปลงค่าเป็นเมทริกซ์ ที่กำหนดขนาดกว้าง * ยาว ชั้นพูลลิ่งจะทำการหาค่าเมทริกซ์ที่มากที่สุด เพื่อแปลงเป็นค่าเมทริกซ์ใหม่ ดังภาพประกอบ 7

2.2 การทำพูลลิ่งแบบหาค่าเฉลี่ย

การทำพูลลิงแบบหาค่าเฉลี่ย (average pooling) เป็นวิธีการพูลลิ่งเหมือนกับการพูลลิ่งแบบหาค่ามากที่สุด แต่ผลลัพธ์ที่ได้จากการพูลลิ่งจะเป็นค่าเฉลี่ย ซึ่งแสดงให้เห็นถึงการเปรียบเทียบกันของการทำพูลลิ่งทั้งสองแบบ ดังภาพประกอบ 7



ภาพประกอบ 7 ชั้นพูลลิ่ง (pooling layer)

ที่มา Sumit Saha, Dec 16, 2018. A Comprehensive Guide to Convolutional Neural Networks — the ELI5 way สืบค้นจาก https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

จากภาพประกอบ 7 แสดงให้เห็นการทำพูลลิ่งแบบหาค่ามากที่สุด (max pooling) ใช้หลักการทำงานเมื่อภาพถูกแปลงค่าเป็นเมทริกซ์ที่กำหนดขนาดกว้าง * ยาวชั้นพูลลิ่งจะทำการหาค่าเมทริกซ์ที่มากที่สุด ส่วนการทำพูลลิงแบบหาค่าเฉลี่ย (average pooling) ใช้หลักการทำงานเมื่อภาพถูกแปลงค่าเป็นเมทริกซ์ที่กำหนดขนาดกว้าง * ยาวชั้นพูลลิ่งจะทำการหาค่าเฉลี่ยภายในเมทริกซ์และแสดงผลออกมาเป็นรูปแบบเมทริกซ์ที่มีค่าเฉลี่ย

3. ชนิดของการทำคอนโวลูชัน

ชนิดของการทำคอนโวลูชัน (convolution type) เป็นการทำคอนโวลูชันด้วยกันหลายชนิด ได้แก่ narrow convolution, wide convolution, stride size, number of filters ซึ่งใน

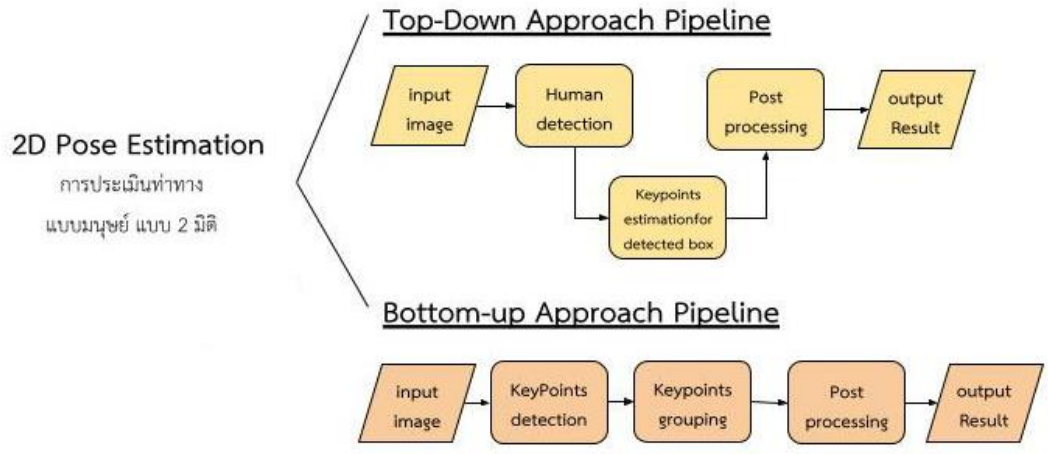
การทำคอนโวลูชันนี้เป็นตัวกรองที่นำมาทำการดอทเมทริกซ์เพื่อรับค่าเข้า และส่งผลลัพธ์ของการทำคอนโวลูชันที่มีข้อมูลรับเข้าขนาด N * N กับตัวกรองขนาด M * M จะได้เมทริกซ์ขนาด (N-M+1) * (N-M+1) [1]

4. ชั้นการเชื่อมโยงเต็มรูปแบบ (fully connected layer)

ชั้นการเชื่อมโยงเต็มรูปแบบ (fully connected layer) โดยชั้นการเชื่อมโยงเต็มรูปแบบ เป็นขั้นตอนสุดท้ายของนิวรอนเน็ตเวิร์คคอนโวลูชัน ซึ่งประกอบด้วยชั้นคอนโวลูชันและชั้นพูลลิ่ง คือการเชื่อมโยงเต็มรูปแบบในขั้นตอนนี้ซึ่งมีส่วนต่างในชั้นย่อยที่มีนิวรอนอยู่ส่วนหนึ่ง ซึ่งนิวรอนแต่ละนิวรอนจะมีเส้นโยงกับนิวรอนทุกตัวในขั้นตอนก่อนหน้านี้และนิวนอลทุกตัวในขั้นตอนถัดไป ทำให้การคำนวณแบบส่งไปส่วนด้านหน้า (feed forward) และการกระจายแบบย้อนกลับ (backpropagation) สามารถทำได้ปกติ [1]

2.3 การประมาณท่าทางของมนุษย์

การประมาณท่าทางของมนุษย์ (human pose estimation) [8] สามารถปรับปรุงให้มีประสิทธิภาพที่ดี โดยการเรียนรู้เชิงลึก (deep learning) และการประเมินท่าทางแบบมนุษย์แบบ 2 มิติ นั้นสามารถแบ่งได้ 2 รูปแบบ คือ top-down และ bottom-up โดยแต่ละรูปแบบจะเริ่มจากการนำรูปภาพเข้าหาจุดสำคัญ เชื่อมจุดสำคัญเป็นโครงสร้าง และแสดงผลลัพธ์ ดังภาพประกอบ 8

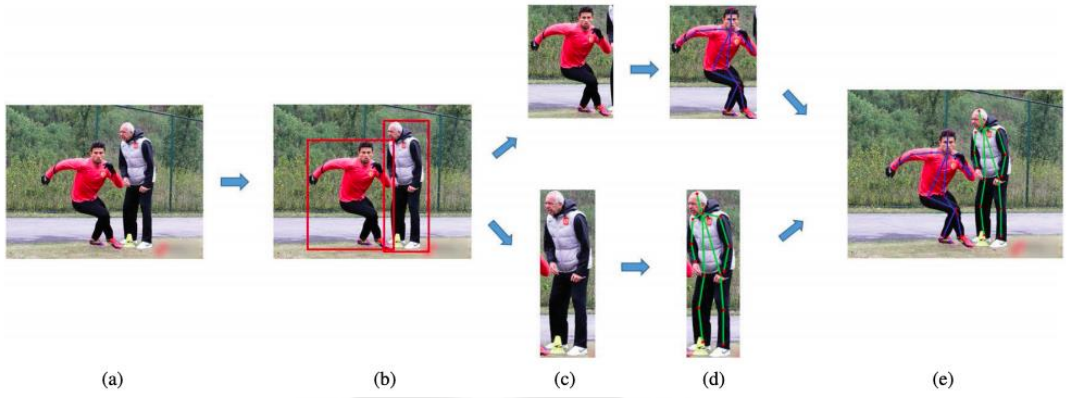


ภาพประกอบ 8 การทำงานแต่ละแบบของการประเมินท่าทางแบบมนุษย์แบบ 2 มิติ

ในการจำแนกท่าทางของมนุษย์ในช่วงหลายปีที่ผ่านมามีการพัฒนาการประเมินท่าทางแบบ 2 มิติ ซึ่งสามารถพัฒนาการจำแนกท่าโพสแบบคนเดียวและหลายคนพร้อมกันได้ [9] ซึ่งในปัจจุบันอัลกอริทึมที่ใช้ในการประมาณท่าทางของมนุษย์ (human pose estimation) ที่รู้จักกันอย่างแพร่หลายมีด้วยกันอยู่ 2 วิธีการ คือ การจำแนกอัลกอริทึมแบบบนลงล่าง (top-down) และการจำแนกอัลกอริทึมแบบล่างขึ้นบน (bottom-up) [10]

2.3.1 TOP-DOWN

การประเมินท่าทางของมนุษย์แบบ top-down เมื่อนำภาพเข้าไปในกระบวนการดังกล่าว ดังภาพประกอบ (9a) จะมีการตรวจจับหาบุคคลภายในภาพ ดังภาพประกอบ (9b) แล้วทำการหาขอบเขตรอบบุคคลที่เรียกว่า bounding box แล้วทำการตัดบางส่วนของภาพ (crop) เพื่อทำการหาจุดสำคัญ แต่ละจุดของภาพแต่ละบุคคล ดังภาพประกอบ (9c) เมื่อสามารถหาท่าทางของบุคคล ดังภาพประกอบ (9d) ของภาพได้แล้วจึงนำมารวมกับภาพที่มีบุคคล [8] ดังภาพประกอบ (9e) เป็นอันจบกระบวนการทำงานแบบ top-down



ภาพประกอบ 9 กระบวนการทำงาน top-down โดย a) ภาพอินพุต b) ตีกรอบบุคคลในภาพ c) ตัดภาพแต่ละบุคคล d) จุด keypoints ของ pose และสร้างโครงกระดูกที่เชื่อมกันระหว่าง keypoints e) รวมภาพ

ที่มา : Dang, Q., Yin, J., Wang, B., & Zheng, W. (2019). Deep learning based 2D human pose estimation: A survey. Tsinghua Science and Technology, 24(6), 663-676.

2.3.2 BOTTOM-UP

การประมาณท่าทางของมนุษย์ แบบ bottom-up เมื่อนำภาพเข้ากระบวนการ ดังภาพประกอบ (10a) จะมีการตรวจหาตำแหน่งจุดสำคัญแต่ละจุดที่อยู่ภายในภาพ ดังภาพประกอบ

(10b) และทำการประมวลผลของจุดสำคัญของแต่ละบุคคล และสร้างโครงกระดูกที่เชื่อมกันระหว่างท่าทางของแต่ละบุคคล [8] ดังภาพประกอบ (10c) เป็นอันจบกระบวนการทำงานแบบ bottom-up

ภาพประกอบ 10 กระบวนการทำงาน bottom-up โดย a) ภาพอินพุต b) หาจุดสำคัญแต่ละจุดในภาพ c) สร้างกระดูกที่เชื่อมกันระหว่างจุดสำคัญของแต่ละบุคคลในภาพ

ที่มา : Dang, Q., Yin, J., Wang, B., & Zheng, W. (2019). Deep learning based 2D human pose estimation: A survey. Tsinghua Science and Technology, 24(6), 663-676.

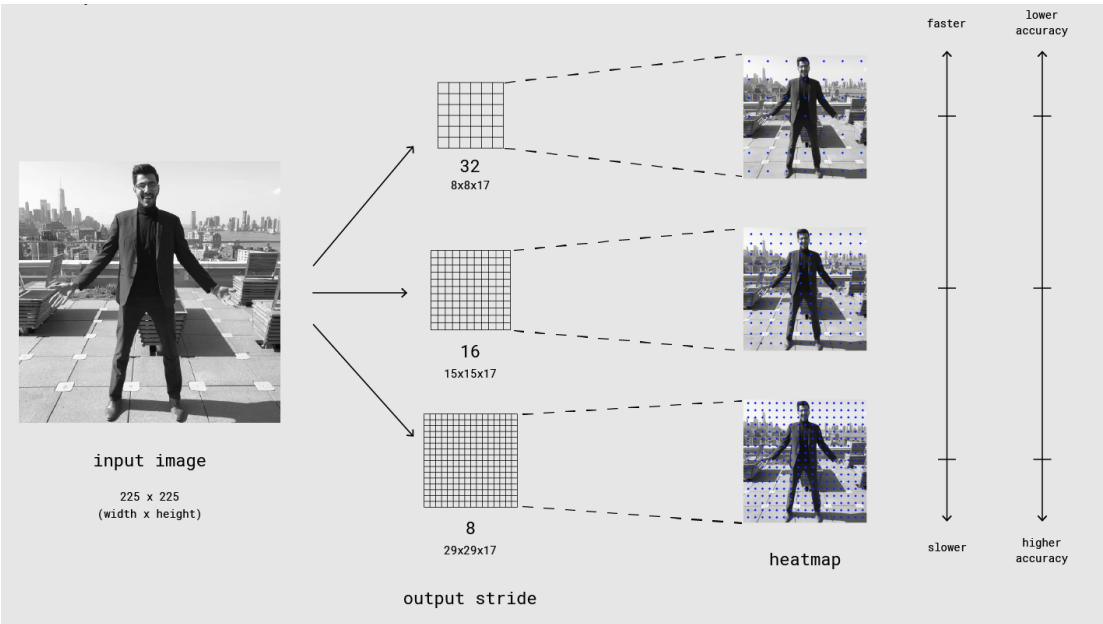
2.4 การตรวจจับจุดสำคัญ

การตรวจจับจุดสำคัญ (keypoint detection) มีรูปแบบในการตรวจจับข้อต่อต่าง ๆของร่างกายมนุษย์ ซึ่งขั้นตอนที่จำเป็นต้องเลือกใช้ไลบรารีซึ่งมีให้เลือกใช้หลากหลายแต่ผู้วิจัยเลือกเปรียบเทียบระหว่าง PoseNet และ OpenPose ในการแสดงถึงข้อต่อร่างกายที่สามารถตรวจจับจุดสำคัญได้ดังนี้

2.4.1 PoseNet

PoseNets [11] เป็นการประมาณท่าทางของมนุษย์ (human pose estimation) โดยนำมาใช้ใน tensorflow นอกจากนี้โครงสร้างโครงข่ายการทำงาน สามารถประมวลผลแบบเรียลไทม์ ซึ่งการประมาณท่าทาง (pose estimation) เป็นเทคนิคการมองเห็นด้วยคอมพิวเตอร์ที่ตรวจจับรูปร่างของมนุษย์ในภาพและวิดีโอ ตามกระบวนการการทำงานต่อมาเพื่อให้สามารถระบุจุดสำคัญได้ ซึ่งมีค่าคะแนนความเชื่อมั่น (confidence score) ระหว่าง 0.0 ถึง 1.0 ซึ่งค่า 1.0 เป็นคะแนนสูงสุด โดยประสิทธิภาพของ tf-pose-estimation จะแตกต่างกันไปตามอุปกรณ์และ stride (heatmaps and offset vectors) ซึ่งแบบจำลองจะใช้ขนาดภาพที่คงที่ซึ่งหมายความว่าสามารถทำนายตำแหน่งภาพต้นฉบับโดยไม่คำนึงว่าภาพจะถูกลดขนาดหรือไม่นั้นหมายความว่า tf-pose-estimation สามารถกำหนดค่าให้มีความแม่นยำตาม stride ซึ่งเป็นตัวกำหนดว่าจะลดขนาด

เอาต์พุตเมื่อเทียบกับขนาดภาพที่ป้อนเข้าไป ซึ่งมันมีผลต่อขนาดของเลเยอร์และรูปแบบผลลัพธ์ ยิ่ง stride มากเท่าใดความละเอียดของเลเยอร์ในโครงข่ายและเอาต์พุต ก็จะยิ่งน้อยลง ในการดำเนินการนี้ stride ของเอาต์พุตสามารถมีค่า 8 16 หรือ 32 ใน stride ของเอาต์พุต ของ 32 จะส่งผลให้ประสิทธิภาพที่เร็วที่สุดแต่มีความแม่นยำต่ำที่สุด ในขณะที่ stride ของเอาต์พุต ของ 8 จะให้ความแม่นยำสูงสุด แต่ประสิทธิภาพที่ช้าที่สุด โดยแนะนำให้เริ่มต้นด้วย stride 16 ดังภาพประกอบ 11



ภาพประกอบ 11 การทำงาน PoseNet ซึ่งมี stride 8, 16 หรือ 32

ที่มา Tensorflow, (2021). Pose estimation สืบค้นจาก https://www.tensorflow.org/lite/examples/pose_estimation/overview

จากกระบวนการที่กล่าวมานั้น รูปแบบของ PoseNet เหมาะสำหรับแอปพลิเคชันมือถือและการฝังเข้าไปในแอปพลิเคชัน โดยมีจุดเด่นเรื่องขนาดของโมเดลที่เล็กความซับซ้อนของกระประมวลผลมีค่อนข้างน้อย [12]

2.4.2 OpenPose

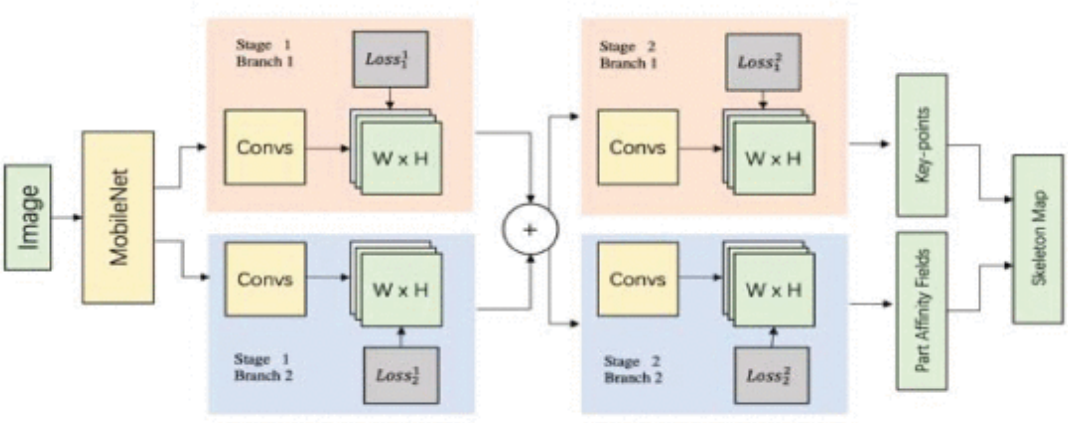
OpenPose เป็นการประมาณท่าทาง (pose estimation) เพื่อตรวจจับหาจุดสำคัญแบบหลายคน โดยพัฒนามาจาก มหาวิทยาลัย Carnegie Mellon University (CMU) [13] โดย OpenPose ใช้สถาปัตยกรรมแบบ CNN สามารถระบุจุดสำคัญบนใบหน้า มือและเท้า พร้อมทั้ง

ระบุข้อต่อของร่างกายมนุษย์จากภาพได้ ซึ่งจุดสำคัญที่หาออกมาได้สามารถระบุชิ้นส่วนร่างกายได้แก่ ตา หู อก จมูก ข้อศอก ไหล่ เข่า ข้อมือ ข้อเท้า และสะโพก ซึ่งผลลัพธ์ที่ได้จากการประมวลผลโดยการอินพุตภาพวาด จะได้จุดสำคัญทั้ง 18 จุด ตามตารางที่ 1 [14]

ตาราง 1 จุดสำคัญและชื่อตำแหน่งของจุดสำคัญ

จุดสำคัญ	ชื่อตำแหน่งของจุดสำคัญ
P0	Nose
P1	Neck
P2	Right Shoulder
P3	Right Elbow
P4	Right Wrist
P5	Left Shoulder
P6	Left Elbow
P7	Left Wrist
P8	Right Hip
P9	Right Knee
P10	Right Ankle
P11	Left Hip
P12	Left Knee
P13	Left Ankle
P14	Right Eye
P15	Left Eye
P16	Right Ear
P17	Left Ear

โมเดลของ OpenPose [13] ทำงานผ่าน MoblieNet [15] มีจุดประสงค์ของการรันโมเดลให้มีความเร็วและใช้ข้อมูล training จากชุดข้อมูล COCO datasets [16] โดยแบบจำลองการประมาณท่าทางมาจากโครงสร้างของสถาปัตยกรรมการประเมินท่าทาง ดังภาพประกอบ 12



ภาพประกอบ 12 การทำงานของ pose estimation

ที่มา : Chen, X., Zhou, Z., Ying, Y., & Qi, D. (2019, July 2019). Real-time Human Segmentation using Pose Skeleton Map. Paper presented at the 2019 Chinese Control Conference (CCC).

ด้วยสถาปัตยกรรมที่นำมาใช้ มี 2 branches และ 2 stage ซึ่งระบบจะป้อนรูปภาพขนาด W * H โดยรูปภาพจะทำงานผ่าน MoblieNet เพื่อดึงคุณลักษณะ (feature) โดย stage 1 จะแยกออกเป็น 2 branches ดังนี้

1. branches 1 กรอบสีส้ม จะใช้คุณลักษณะทำนายจุดสำคัญที่หามาได้จากรูปภาพแล้วดูว่าจุดสำคัญที่หามาได้นั้น มีค่าความเชื่อมั่นว่าเป็นจุดข้อต่อของร่างกายมนุษย์อยู่ตำแหน่งไหนบ้าง

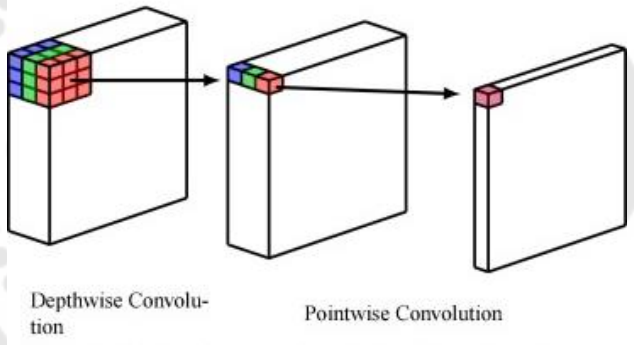
2. branches 2 กรอบสีน้ำเงิน จะใช้คุณลักษณะทำนายหาคู่จุดสำคัญที่มีความสัมพันธ์ระหว่างข้อต่อ ที่หามาได้หาความเชื่อมั่นของโครงกระดูกแต่ละส่วนของร่างกายมนุษย์ว่าอยู่ตำแหน่งไหนบ้าง

เมื่อได้ผลจาก stage 1 จะได้ค่าของแต่ละจุดสำคัญและค่าของโครงกระดูกแต่ละส่วนระหว่างจุดสำคัญนำมารวมกันเพื่อใช้ใน stage 2 และก็ทำซ้ำเหมือน stage 1 สุดท้าย จะได้

ค่าตำแหน่งจุดสำคัญแต่ละจุด และค่าโครงกระดูกแต่ละส่วน เมื่อได้ทั้ง 2 ส่วนนี้ก็จะนำมา map กับภาพเอาต์พุตที่ได้ ดังภาพประกอบ 12

2.5 ไลบรารี MobileNets

MobileNets [15] เป็นรูปแบบการประมาณท่าทาง (pose estimation) เพื่อตรวจจับหาหาจุดสำคัญซึ่งทำงานได้อย่างมีประสิทธิผลตามรูปแบบ convolutions โดย depthwise convolutions แบ่งได้ convolutions ที่แตกต่างกันแบ่งออกเป็น 2 อย่างคือ depthwise convolution และ pointwise convolution นอกจากนี้ทั้งสอง hyperparameters ใหม่จะถูกนำมาใช้ใน MobileNets ซึ่งช่วยให้ประมวลผลได้เร็วมากแต่ต้องแลกกับความถูกต้องที่ลดลง แต่ในปัจจุบันมี MobileNetV1 ที่มาจาก Google ที่ใช้ depthwise separable convolution [17] ใช้ลดขนาดและความซับซ้อนของแบบจำลอง ดังภาพประกอบ 13



ภาพประกอบ 13 โครงข่ายประสาทเทียมคอนโวลูชัน 2 แบบ depthwise และ pointwise

ที่มา : Kc, K., Yin, Z., Wu, M., & Wu, Z. (2019). Depthwise separable convolution architectures for plant disease classification. Computers and Electronics in Agriculture, 165, 104948.

2.6 การสร้างแบบจำลองด้วย ANN

การสร้างแบบจำลองด้วย ANN คือ ชุดของโครงข่ายระหว่างอินพุตและเอาต์พุตโดยมี ค่าน้ำหนัก(weight) เชื่อมระหว่าง 2 ค่านี้ โดยประกอบด้วย ชั้นอินพุต 1 ชั้น เลเยอร์กลางอย่างน้อย 1 ชั้น และชั้นเอาต์พุต 1 ชั้น โดยโครงข่ายประสาทเทียมจะเรียนรู้โดยการปรับค่าน้ำหนัก ของการเชื่อมต่อ โดยจะมีการอัพเดทค่าน้ำหนักวนซ้ำ เพื่อประสิทธิภาพของโครงข่ายจะดีขึ้น โดยพื้นฐานของการเชื่อมต่อ ANN สามารถแบ่งออกเป็น 2 ประเภท คือ feed-forward network และ recurrent

network ซึ่งโครงข่ายประสาทเทียมแบบ feed-forward network จะเชื่อมต่อหน่วยทีไปในทิศทางเดียวกัน แต่โครงข่ายประสาทเทียมแบบ recurrent network จะเชื่อมต่อหน่วยที่สามารถเกิดวนซ้ำในระหว่างการฝึกอบรม (training) ค่าน้ำหนักที่เชื่อมต่อไปในแต่ละโหนดจะถูกปรับให้เหมาะสมจนกว่าโครงข่ายจะมีความแม่นยำถึงระดับที่กำหนด ซึ่งมีข้อดีที่สามารถเรียนรู้ทักษะของโครงข่ายได้ดีโดย artificial neural network สามารถใช้งานได้หลากหลายรูปแบบไม่ว่าจะเป็นการจดจำรูปแบบ การแพทย์ แอปพลิเคชันทางธุรกิจ การรู้จำเสียง เป็นต้น

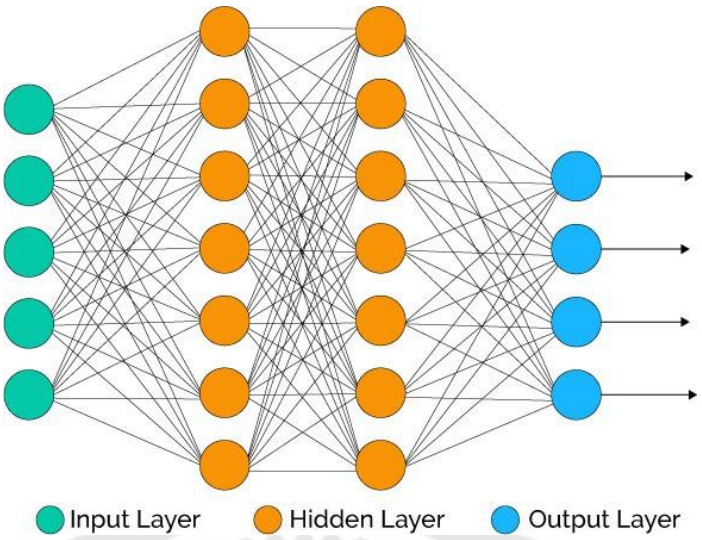
ANN เป็นระบบการปรับตัวที่ซับซ้อนซึ่งสามารถเปลี่ยนโครงสร้างภายในโดยอิงตามข้อมูลที่ส่งผ่าน โดยการปรับค่าน้ำหนักของการเชื่อมต่อ ซึ่งการเชื่อมต่อแต่ละครั้งมีค่าน้ำหนักที่เกี่ยวข้อง โดยค่าน้ำหนัก คือ ตัวเลขที่ควบคุมระหว่าง 2 โหนด ซึ่งมีการปรับค่าน้ำหนัก เพื่อปรับปรุงผลลัพธ์ โดยวิธีการเรียนรู้ที่นิยม มีดังนี้

1. Supervised learning เป็นเทคนิคที่เกี่ยวข้องกับการฝึกอบรมที่มีความฉลาดกว่าโครงข่าย

2. Unsupervised learning เป็นเทคนิคที่จะใช้เมื่อไม่มีชุดตัวอย่างที่มีคำตอบ

3. Reinforcement learning เป็นการตัดสินใจบนพื้นฐานของข้อเสนอแนะจากสภาพแวดล้อม

รูปแบบที่จะนำค่า feature vector มาใช้สำหรับ ANN (Artificial Neural Network) [18] ดังภาพประกอบ 14 ซึ่งสามารถที่จะจำแนกประเภทในการจัดหมวดหมู่ด้วยชุดข้อมูล ซึ่งชุดข้อมูลแบ่งออกเป็น 2 ส่วน คือ training phase และ test phase โดยที่ training phase จะใช้สำหรับการเรียนรู้โครงข่าย (learning of network) และ test phase ใช้สำหรับวัดความแม่นยำของการจำแนกประเภท ในการแบ่งชุดข้อมูลจะทำการสุ่มค่า โดยค่า parameters ส่วนใหญ่จะมี input node output node และ hidden layer ซึ่งจะถูกกำหนดด้วยจำนวนค่าคงที่



ภาพประกอบ 14 โครงข่ายประสาทเทียม

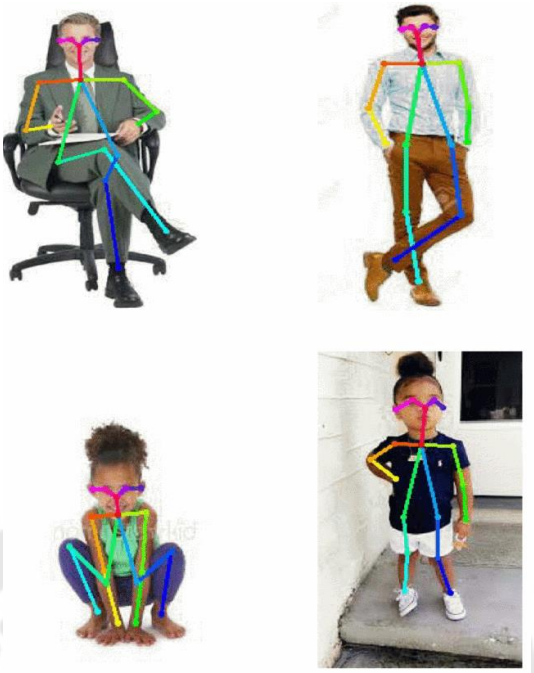
ที่มา Conor Mc., Dec 22, 2017. Machine learning fundamentals (II): Neural networks สืบค้นจาก https://towardsdatascience.com/machine-learning-fundamentals-ii-neural-networks-f1e7b2cb3eef

2.7 งานวิจัยที่เกี่ยวข้อง

ในส่วนของหัวข้องานวิจัยที่เกี่ยวข้อง Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh.,2016 [4] ได้มีการพัฒนาไลบรารีที่ชื่อว่า OpenPose [19] โดยระบบจะประมวลผลแบบเรียลไทม์ สำหรับการตรวจจับจุดสำคัญของร่างกายมนุษย์ มือ เท้า และใบหน้า ซึ่งในส่วนของไลบรารีนี้ มีการใช้งานกันอย่างแพร่หลายในปัจจุบัน [20], [21], [14], [13], [12] นอกจากนี้ OpenPose ยังบรรจุอยู่ในไลบรารีของ OpenCV [22] โดยขอยกตัวอย่างงานวิจัยที่เกี่ยวข้องนำเสนอ 3 หัวข้อดังนี้

2.7.1 บทความวิจัยเรื่อง Human posture classification using skeleton information โดย [21]

การทำงานเริ่มจากนำรูปภาพจากอินเทอร์เน็ตจำนวน 146 รูปซึ่งเป็นรูปคนยืนหรือนั่งบนเก้าอี้ โดยใช้ไลบรารี OpenPose ในการแยกคุณสมบัติสำหรับการจัดประเภทท่ายืนและนั่ง โดยระบบจะใช้รูปอินพุตและสร้างเอาต์พุตที่มีตำแหน่งของจุดสำคัญทั้ง 18 จุด ซึ่งเอาต์พุตที่ออกมาจะเป็นโครงร่างกระดูกมนุษย์ ดังภาพประกอบ 15



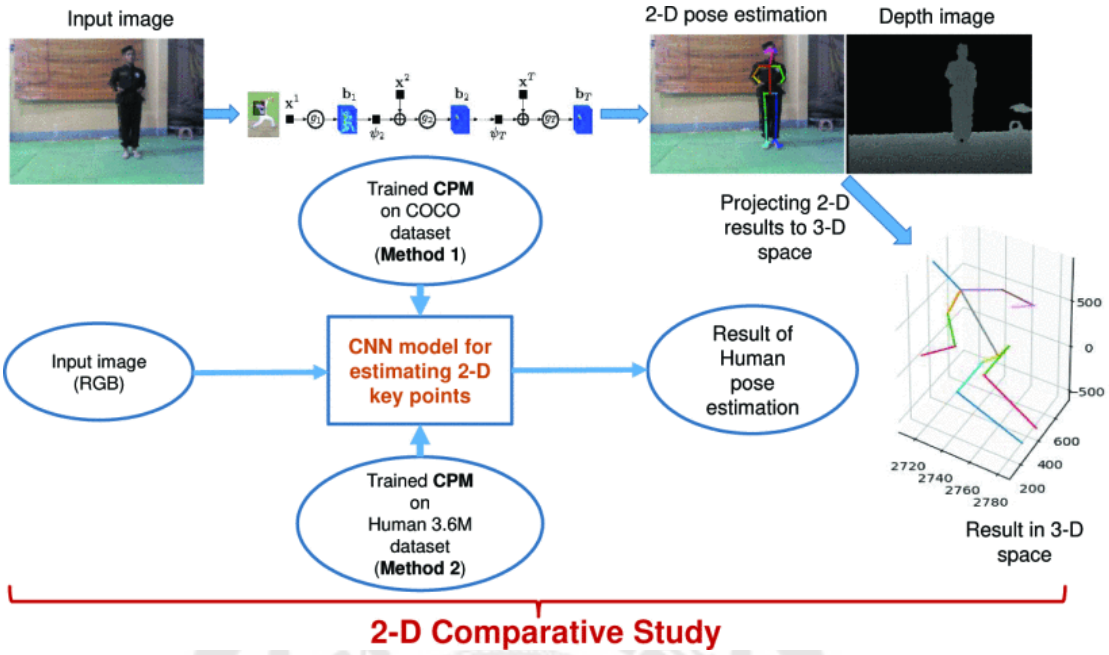
ภาพประกอบ 15 ตำแหน่งข้อต่อที่ได้รับจากการใช้ไลบรารี OpenPose

ที่มา : Ghazal, S., & Khan, U. S. (2018, March 2018). Human posture classification using skeleton information. Paper presented at the 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET).

การตรวจจับท่าทางของคนในภาพนิ่งโดยอาศัย OpenPose โดยการสกัดคุณลักษณะองศาของมุมและระยะห่างระหว่างจุดสำคัญ โดยมีด้วยกัน 10 คุณลักษณะ ซึ่งนำไปใช้จำแนกระหว่างท่านั่งและท่ายืน โดยการทำงานของอัลกอริทึมการจำแนกท่านั่งท่ายืนมีค่าเฉลี่ยความแม่นยำอยู่ที่ 95%

2.7.2 บทความวิจัยเรื่อง OpenPose's Evaluation in The Video Traditional Martial Arts Presentation ในส่วนของ [23]

ในบทความนี้เสนอให้ใช้ Convolutional Neural Network (CNN) ในการประมาณจุดสำคัญและข้อต่อของศิลปะการป้องกันตัวในแบบ 2 มิติ แล้วนำผลลัพธ์ที่ได้ทำเป็นรูปแบบ 3 มิติ โดยใช้การวัดจากจุดสำคัญ สำหรับการประเมินการประมาณท่าทาง ใช้โมเดล CNN คือ CPM (Convolutional Pose Machine) โดยเปรียบเทียบชุดข้อมูลที่ใช้ในการฝึกอบรมสำหรับ CPM คือ ชุดข้อมูล MSCOCO Keypoints Challenge และ Human3.6m ดังภาพประกอบ 16



ภาพประกอบ 16 การเปรียบเทียบการประเมินท่าทางมนุษย์ 2 มิติ กับผลลัพธ์ 3 มิติ

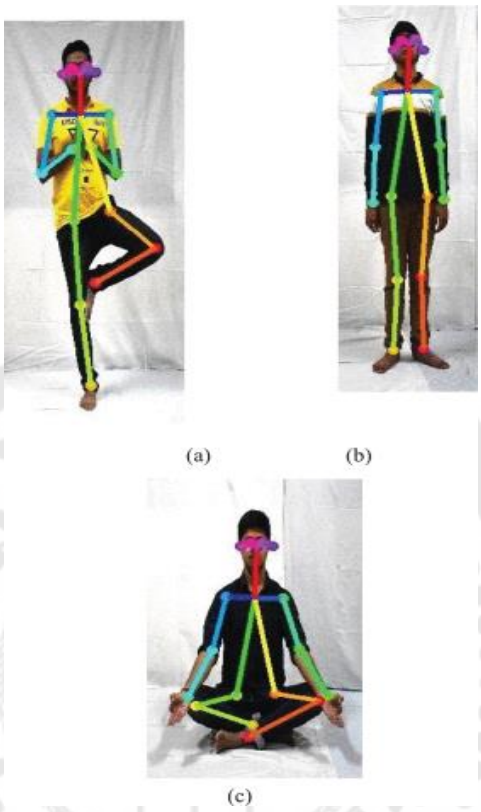
ที่มา : Le, V., Nguyen, T., Tran, N., & Pham, T. (2019, Sep. 2019). OpenPose's Evaluation in The Video Traditional Martial Arts Presentation. Paper presented at the 2019 19th International Symposium on Communications and Information Technologies (ISCIT).

การนำ OpenPose มาประเมินท่าทางจาก VDO ศิลปะการป้องกันตัว จากภาพ 3 มิติ โดยการประเมินท่าทางใช้การวัด ความยาวของข้อต่อ มุมของข้อต่อ และระยะเบี่ยงเบนจากจุดสำคัญโดยผลลัพธ์ที่ได้รับการประเมินมาจากชุดข้อมูลศิลปะการต่อสู้ การเต้นรำ และกีฬา

2.7.3 บทความวิจัยเรื่อง Implementation of Machine Learning Technique for Identification of Yoga Poses.และงานวิจัยของ [14]

ในบทความนี้ได้ใช้เทคนิคการตรวจจับท่าทางที่สามารถระบุท่าทางของโยคะ โดยงานวิจัยนี้ไม่มีชุดข้อมูล สำหรับการตรวจจับท่าทาง ซึ่งเทคนิคการตรวจจับท่าทางโยคะ สร้างจากชุดข้อมูลโยคะจำนวน 5,500 ภาพ โดยจำแนกท่าโยคะได้ 10 ท่าโพสโยคะ โดยอัลกอริทึม tf-pose estimation ทำการวาดโครงกระดูกของร่างกายมนุษย์ ดังภาพประกอบ 17 จากนั้นทำการสกัดหาคุณสมบัติที่เกิดจากมุมที่มาจากข้อต่อ เพื่อใช้สำหรับโมเดลการเรียนรู้ของเครื่อง โดยชุดข้อมูลจะถูกแบ่งชุดสำหรับฝึกอบรม 80% และชุดข้อมูลสำหรับการทดสอบ 20% โดยใช้ machine learning

model ในการจำแนกการประเมินท่าทาง โดยใช้ Random Forest Classifier โดยมีความแม่นยำ 99.04%



ภาพประกอบ 17 โครงร่างกระดูกที่สร้างขึ้นโดยใช้ tf-pose estimation

ที่มา : Agrawal, Y., Shah, Y., & Sharma, A. (2020, April 2020). Implementation of Machine Learning Technique for Identification of Yoga Poses. Paper presented at the 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT).

บทที่ 3
วิธีดำเนินการวิจัย

ในบทนี้จะกล่าวด้วยวิธีดำเนินการวิจัยโดยจะกล่าวเป็นลำดับขั้นตอนดังนี้

1. การกำหนดประชากรและการสุ่มกลุ่มตัวอย่าง
2. การสร้างเครื่องมือที่ใช้ในงานวิจัย
3. การเก็บรวบรวมข้อมูล
4. เกณฑ์การให้คะแนนของผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัว
5. อัลกอริทึมและแบบจำลองของการทำนาย

3.1 การกำหนดประชากรและการสุ่มกลุ่มตัวอย่าง

ประชากร

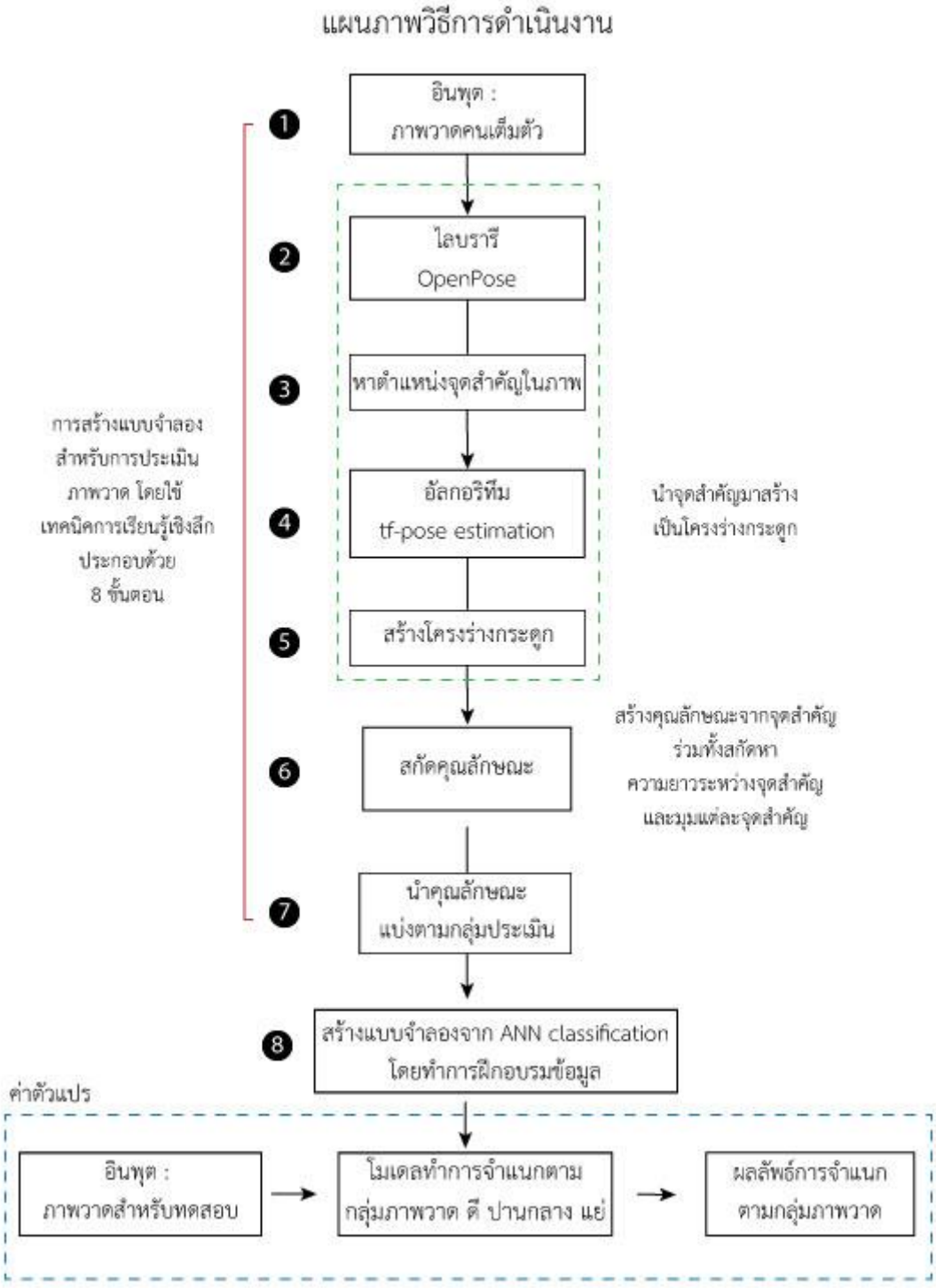
ข้อมูลที่ถูกเก็บรวบรวมจากผลงานการวาดรูปคนเต็มตัวของนักศึกษาชั้นปีที่ 1 เอกแอนิเมชัน โดยหนึ่งห้องเรียนมีจำนวนนักศึกษา 22 คนต่อห้องเรียน และมีนายแบบสำหรับเป็นแบบจำนวน 3 คน เพื่อใช้เป็นข้อมูลกลุ่มภาพวาด ซึ่งรวบรวมผลงานได้จำนวน 66 ผลงาน

การเลือกกลุ่มตัวอย่าง

ข้อมูลภาพวาดจะถูกแบ่งออกเป็น 2 ชุด คือ ชุดสำหรับสร้างแบบจำลอง (train dataset) และชุดสำหรับทดสอบแบบจำลอง (test dataset)

3.2 การสร้างเครื่องมือที่ใช้ในงานวิจัย

การสร้างเครื่องมือที่ใช้ในงานวิจัย มีกระบวนการทำงานของการประเมินท่าทางประกอบด้วย 8 ขั้นตอน หลังจากผ่านกระบวนการสกัดข้อมูลภาพวาด แล้วท้ายสุดจะเป็นกระบวนการจำแนกตามกลุ่มภาพวาด ดังภาพประกอบ 18



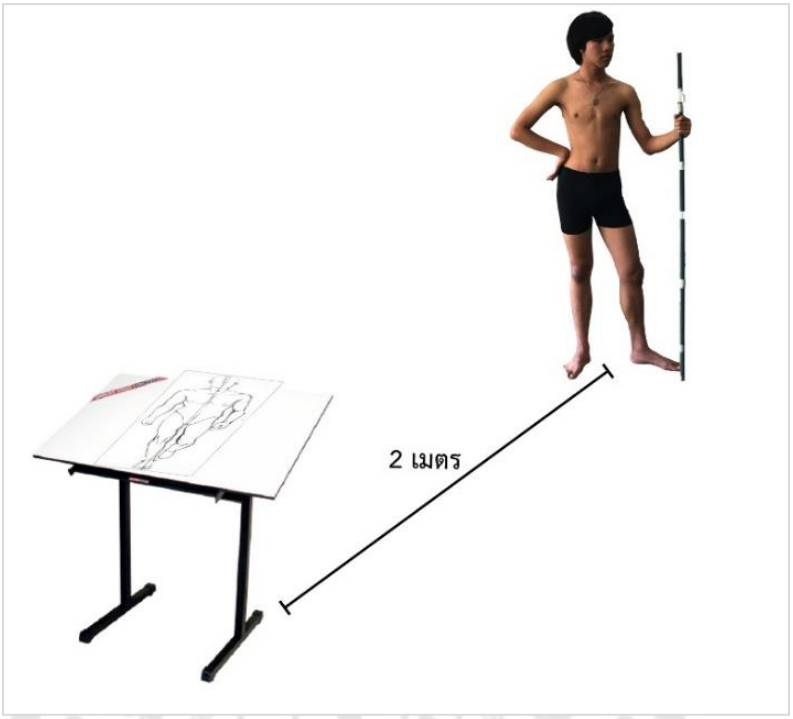
ภาพประกอบ 18 กระบวนการทำงานของระบบ

จากภาพประกอบ 18 แสดงถึงกระบวนการทำงานของแบบจำลอง human pose estimation และแบบจำลอง ANN โดยเริ่มตั้งแต่ขั้นตอนแรกที่นำข้อมูลภาพวาดใช้ไลบรารี OpenPose ในการหาจุดสำคัญทั้ง 18 จุดสำคัญ เพื่อสร้างโครงร่างกระดูก จากอัลกอริทึม tf-pose estimation อันเป็นการเสร็จกระบวนการ human pose estimation

หลังจากนั้นนำค่าตำแหน่งจุดสำคัญที่อยู่บนภาพวาด นำมาสกัดคุณลักษณะซึ่งสกัดค่าคุณลักษณะองศาของจุดสำคัญและคุณลักษณะระยะห่างระหว่างจุดสำคัญ โดยรวบรวมได้ 26 คุณลักษณะ ต่อไปใช้กระบวนการ ANN สร้างแบบจำลองโดยแบ่งกลุ่มภาพวาด คือ ดี ปานกลาง แย่ และทำการแบ่งข้อมูลออกเป็น train และtest แล้วนำข้อมูล train ใส่เข้าไปในแบบจำลองการประเมิน ให้แบบจำลองเรียนรู้ข้อมูล และสุดท้ายนำข้อมูลที่ได้จากการทำนายมาเทียบกับข้อมูล test เพื่อประเมินความแม่นยำของแบบจำลอง

3.3 การเก็บรวบรวมข้อมูล

งานวิจัยนี้ได้รับความอนุเคราะห์จากอาจารย์มานพ เอี่ยมสะอาด เจ้าของรายวิชา 802106 : DRAWING FOR INFORMATION TECHNOLOGY II เพื่อการเก็บข้อมูลจากนักศึกษาชั้นปีที่ 1 เอกแอนิเมชัน จำนวน 1 ห้องเรียน ซึ่งในแต่ละสัปดาห์นักศึกษาจะต้องวาดภาพคนเต็มตัวอย่างน้อย คนละหนึ่งภาพ ขนาด A2 เพื่อเป็นการฝึกวาดและมองนายแบบเพื่อวาดภาพคนเต็มตัว ซึ่งมุมมองที่จะทำการเก็บรวบรวมข้อมูลภาพนั้นจะเป็นมุมมองด้านหน้า ซึ่งมีระยะห่างจากนายแบบ 2 เมตรโดยประมาณ ดังภาพประกอบ 19 โดยผู้วิจัยจะเก็บมุมมองด้านหน้าโดยนายแบบคนที่ 1 หันซ้ายไม่เกิน 30 องศา นายแบบคนที่ 2 หันขวาไม่เกิน 30 องศา และนายแบบคนสุดท้ายหันหน้าตรงกับผู้วาดภาพ

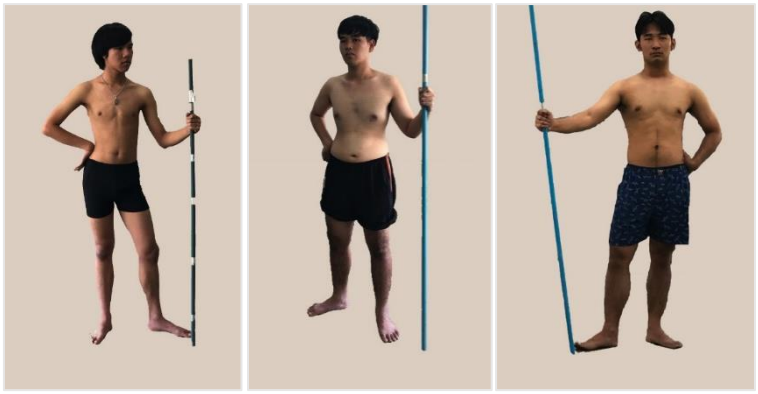


ภาพประกอบ 19 ตำแหน่งการยืนของนายแบบและตำแหน่งที่นักศึกษาวาดภาพ

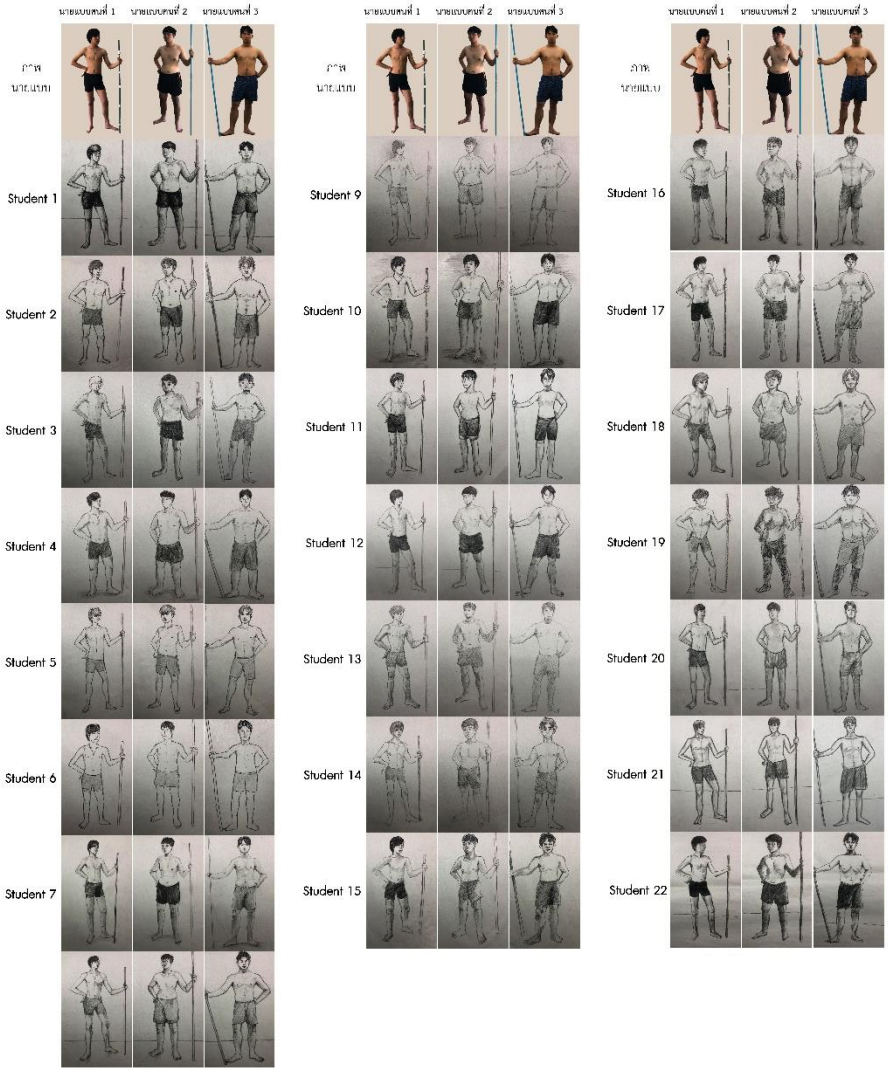
ในการวิจัยนี้ได้นำข้อมูลภาพวาดคนเต็มตัวจากนักศึกษาจำนวน 22 คน คนละ 3 แบบ รวม 66 ภาพ ซึ่งข้อมูลภาพวาดมีการเก็บรวบรวมอยู่ 2 กลุ่ม คือ

1. กลุ่มข้อมูลภาพนายแบบ จำนวน 3 คน ดังภาพประกอบ 20 มีรายละเอียดดังนี้ นายแบบ 3 คน ลักษณะการยืนหันซ้าย 1 คน หันขวา 1 คน และหน้าตรง 1 คน

2. ชุดข้อมูลภาพวาดนักศึกษา จำนวน 22 คน คนละ 3 ภาพ ดังภาพประกอบ 21 มีรายละเอียดดังนี้ ชื่อ – นามสกุล รหัสนักศึกษา วาดภาพนายแบบคนที่ ได้แบ่งกลุ่มภาพวาด ดี ปานกลาง และแย่ ตามคะแนนที่ผู้เชี่ยวชาญการวาดภาพคนเต็มตัว



ภาพประกอบ 20 ภาพนายแบบทั้งหมด 3 คน



ภาพประกอบ 21 ผลงานภาพวาดของนักศึกษาทั้ง 22 คน

3.4 เกณฑ์การให้คะแนนของผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัว

การดำเนินงานผู้วิจัยได้เตรียมภาพวาดคนเต็มตัวให้ผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัวให้คะแนนแต่ละส่วน ซึ่งเกณฑ์ในการตรวจให้คะแนนภาพวาดคนเต็มตัวมีสัดส่วนคะแนนตามตาราง 2

ตาราง 2 เกณฑ์การให้คะแนนของผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัว

โครงสร้างและสัดส่วน	น้ำหนักแสงเงา	รายละเอียดและเส้น	รวม
50%	30%	20%	100%

จะเห็นได้ว่าคะแนนในส่วน โครงสร้างและสัดส่วน มีผลต่อการวาดภาพคนเต็มตัวถึง 50% ของภาพวาด ดังนั้นผู้วิจัยเลือกใช้คะแนนเฉพาะส่วนโครงสร้างและสัดส่วนมาใช้ในการเปรียบเทียบ โดยนำมาปรับให้เป็น 100% เพื่อนำมาเทียบเป็นเกรดซึ่งสามารถระบุคะแนนในส่วนโครงสร้างและสัดส่วนได้ตามตาราง 3

ตาราง 3 ผู้เชี่ยวชาญให้คะแนนเฉพาะส่วนโครงสร้างและสัดส่วนเต็ม 100 คะแนน

นักศึกษาคนที่	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 1	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 2	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 3
คนที่ 01	54	70	57
คนที่ 02	50	65	38
คนที่ 03	40	20	20
คนที่ 04	44	30	30
คนที่ 05	40	45	30
คนที่ 06	42	40	22
คนที่ 07	42	67	70

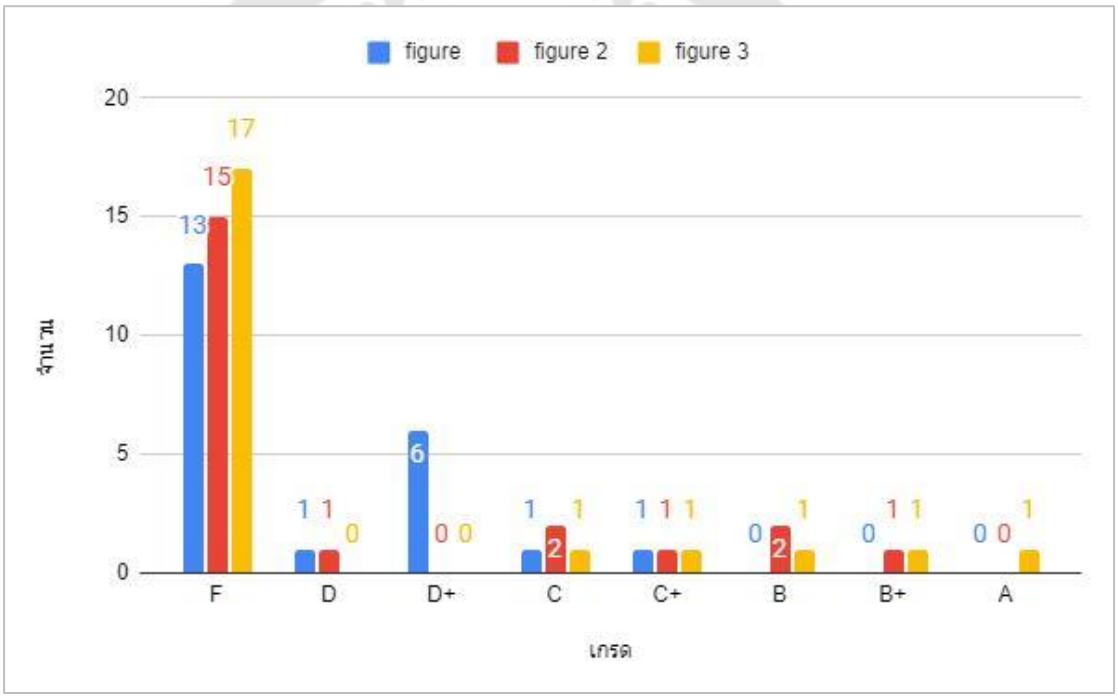
นักศึกษาคนที่	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 1	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 2	คะแนนเฉพาะโครงสร้างและสัดส่วน figure 3
คนที่ 08	44	60	65
คนที่ 09	52	42	40
คนที่ 10	44	40	40
คนที่ 11	50	42	25
คนที่ 12	58	55	75
คนที่ 13	50	55	60
คนที่ 14	52	30	30
คนที่ 15	38	35	27
คนที่ 16	40	40	34
คนที่ 17	60	40	40
คนที่ 18	38	22	32
คนที่ 19	40	20	20
คนที่ 20	44	30	21
คนที่ 21	46	31	21
คนที่ 22	42	28	30

จากตารางที่ 3 สามารถนำคะแนนที่นักศึกษาวาดภาพมาอ้างอิงตามเกรด ในตารางที่ 4 เพื่อใช้ในการแบ่งคะแนนตามกลุ่ม ซึ่งมีทั้งหมด 8 กลุ่ม (เกรด : A-F) โดยเทียบเป็น class ได้ทั้งหมด 8 class (class : A-F) ดังนี้

ตาราง 4 ช่วงคะแนนของเกรด

เกรด	A	B+	B	C+	C	D+	D	F
ช่วง	100 ถึง	74 ถึง	69 ถึง	64 ถึง	59 ถึง	54 ถึง	49 ถึง	44 ถึง
คะแนน	75	70	65	60	55	50	45	0

จากคะแนนที่นักศึกษาวาดภาพคนเต็มตัว คะแนนเฉพาะส่วนของโครงสร้างและสัดส่วนจากนายแบบทั้ง 3 คน สามารถนำมาเทียบคะแนนเป็นเกรด A-F ซึ่งจะได้จำนวนชิ้นงานของแต่ละเกรด ดังภาพประกอบ 22



ภาพประกอบ 22 จำนวนนักศึกษาที่ได้เกรดแต่ละภาพวาดตามรูปแบบ histogram

จากนักศึกษาจำนวน 22 คน วาดภาพคนละ 3 แบบ รวม 66 ภาพ จะเห็นได้ว่าช่วงเกรด A B+ B C+ C D+ มีจำนวนรวมเฉลี่ย 5 ถึง 8 คน เมื่อเทียบกับเกรด D F ซึ่งมีจำนวนรวมเฉลี่ย 14 ถึง17 คน ต่อ 1 ภาพนายแบบ โดยแต่ละกลุ่มสามารถแบ่งตามกลุ่มภาพวาดที่ผ่านการตรวจจากผู้เชี่ยวชาญแต่เนื่องด้วยจำนวนของภาพวาดมีไม่ครบทุกกลุ่มเกรด ทำให้การแยกข้อมูลของแต่ละกลุ่มเกรดเพื่อนำไปใช้ในการทดสอบมีจำนวนค่อนข้างน้อย ดังนั้นผู้วิจัยได้จัดกลุ่มใหม่ เป็น 3 กลุ่ม

และแบ่งตามจำนวนภาพวาดแต่ละกลุ่ม ได้แก่ ดี ปานกลาง และแย่ (class : good moderate poor) ตามตาราง 5

1. กลุ่มนักศึกษาที่วาดภาพดี อยู่ในช่วงเกรด A ถึง B
2. กลุ่มนักศึกษาที่วาดภาพปานกลาง อยู่ในช่วงเกรด C+ ถึง C
3. กลุ่มนักศึกษาที่วาดภาพแย่ อยู่ในช่วงเกรด D+ D และ F

ตาราง 5 จำนวนภาพวาดที่จำแนกในแต่ละกลุ่ม

กลุ่มการจำแนก	**จำนวนภาพวาดแต่ละกลุ่มของ figure 1**	**จำนวนภาพวาดแต่ละกลุ่มของ figure 2**	**จำนวนภาพวาดแต่ละกลุ่มของ figure 3**
ดี	8	9	5
ปานกลาง	8	6	7
แย่	6	7	10
รวม	22	22	22

3.5 อัลกอริทึมและแบบจำลองของการทำนาย

งานวิจัยนี้ได้ใช้อัลกอริทึม OpenPose และ tf-pose estimation ในการหาจุดสำคัญและสร้างเป็นโครงร่างกระดูก ซึ่งข้อมูลใน dataset เป็นภาพวาดที่ถูกแบ่งตามกลุ่ม ดี ปานกลาง แย่ : ซึ่งเป็นไปตามกระบวนการและใช้การจำแนกเครื่องมือด้วย ANN เพื่อสร้างแบบจำลอง โดยแบบจำลองได้ผ่านการฝึกอบรม และนำภาพวาดมาทดสอบตามขั้นตอนดังนี้

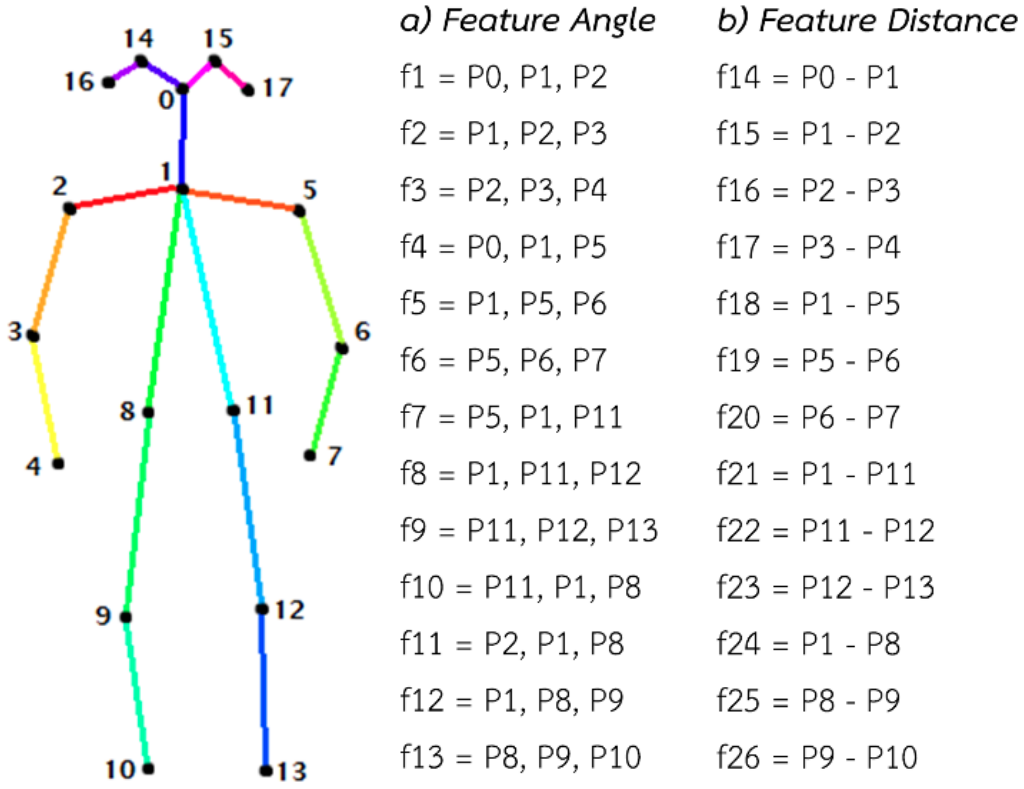
3.5.1 การเตรียมภาพวาด

การเตรียมภาพวาดคนเต็มตัว ที่ผ่านการให้คะแนนจากผู้เชี่ยวชาญด้านการวาดภาพคนเต็มตัวโดยแบ่งภาพวาดของแต่ละต้นแบบออกเป็น 3 กลุ่ม

1. กลุ่มภาพวาดดี
2. กลุ่มภาพวาดปานกลาง
3. กลุ่มภาพวาดแย่

3.5.2 ไลบรารี OpenPose

ไลบรารี OpenPose จะสกัดตำแหน่งจุดสำคัญเพื่อระบุตำแหน่งจุดสำคัญแต่ละจุดที่อยู่ในภาพวาดคนเต็มตัว ซึ่งมีทั้งหมด 18 จุด โดยเริ่มจากตำแหน่งที่ 0 คือ บริเวณจมูกและไล่ลำดับไปจนถึงตำแหน่งที่ 17 คือ บริเวณตาซ้าย ดังภาพประกอบ 23



ภาพประกอบ 23 ตำแหน่งจุดสำคัญทั้ง 18 จุด a) คุณลักษณะของมุมจากจุดสำคัญ b) ระยะห่างระหว่างจุดสำคัญ

ที่มา Ale Solano, Nov 20, 2017. Human pose estimation using OpenPose with TensorFlow (Part 2) สืบค้นจาก https://arvrjourney.com/human-pose-estimation-using-openpose-with-tensorflow-part-2-e78ab9104fc8

จากภาพประกอบ 23 แสดงให้เห็นว่าตำแหน่งจุดสำคัญทั้ง 18 จุดนั้นอยู่ตำแหน่งใดบ้าง หลังจากที่ได้ตำแหน่งจุดสำคัญมาทั้ง 18 จุดจะนำจุดสำคัญที่ได้มาหาคุณลักษณะมุมโดยใช้จุดสำคัญอย่างน้อย 3 จุดในการหามุมแต่ละค่าคุณลักษณะดังภาพประกอบ (23a) และในการหา

คุณลักษณะของระยะห่างจำเป็นต้องใช้จุดสำคัญ 2 จุดในการหาระยะห่างแต่ละค่าคุณลักษณะดังภาพประกอบ (23b)

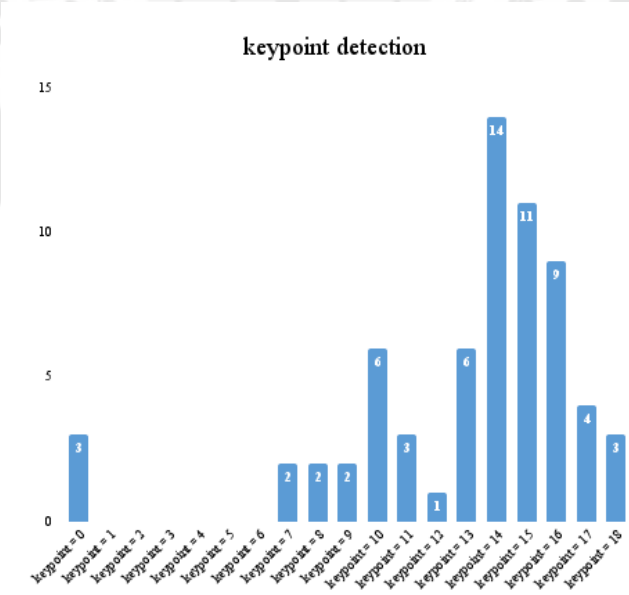
3.5.3 การสกัดจุดสำคัญ

สกัดจุดสำคัญ (keypoints) ตามโครงสร้างกระดูกจากท่าทางของมนุษย์มี 18 จุดสำคัญ โดยจะมีคุณลักษณะแบบเวกเตอร์ ที่มีจุดสำคัญเท่ากับ N จำนวนสามารถแสดงด้วยเวกเตอร์ 2 มิติซึ่งประกอบด้วยพิกัด x และ y ของจุดสำคัญแสดงในสมการที่ (2)

$$F = [x_1, y_1, x_2, y_2, x_3, y_3, \dots, x_N, y_N] \quad (2)$$

จากการแสดงคุณลักษณะจะเห็นได้ว่าข้อมูลจากสมการที่ (2) สามารถใช้เป็นข้อมูลสำหรับหาคุณลักษณะต่อไป

จากการทดสอบหาจุดสำคัญของแต่ละภาพวาดพบว่าจากภาพวาดทั้งหมด 66 ภาพ จำนวนภาพวาดที่ OpenPose สามารถตรวจจับได้นั้น จะเห็นได้ว่าช่วงค่าเฉลี่ยของจุดสำคัญ คือ จำนวนจุดสำคัญที่มากกว่าและเท่ากับ keypoint = 14 ขึ้นไปจนถึง keypoint = 18 มีจำนวนภาพวาดทั้งสิ้น 41 คิดเป็น 62.62% ของภาพวาดทั้งหมด ดังภาพประกอบ 24



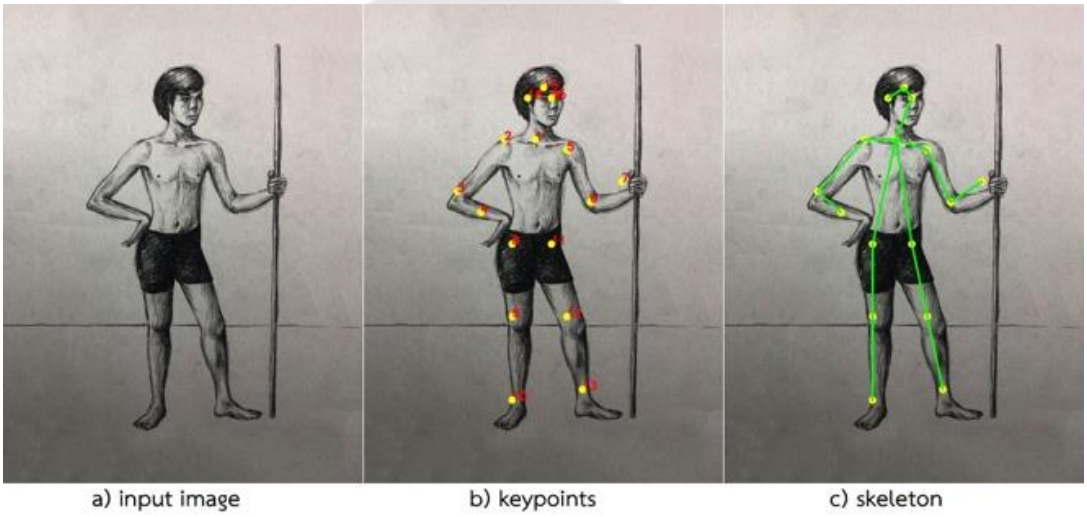
ภาพประกอบ 24 จำนวนภาพวาดที่สามารถตรวจจับจุดสำคัญได้

3.5.4 **อัลกอริทึม** tf-pose-estimation

อัลกอริทึม tf-pose-estimation [11] ใช้ในการสร้างกระดูกของบุคคลที่แสดงท่าทางแต่ละภาพโดยมีข้อต่อแต่ละส่วนของร่างกายเชื่อมกับโครงกระดูก [14] โดยอัลกอริทึมทำงานในการบอกตำแหน่งจุดสำคัญแต่ละจุด

3.5.5 **โครงร่างกระดูก**

โครงร่างกระดูก (skeleton feature) เมื่อภาพวาดสามารถระบุตำแหน่งจุดสำคัญบนภาพวาดครบทั้ง 18 จุดไลบรารี OpenPose จะทำการหาความสัมพันธ์ระหว่างจุดสำคัญ แล้วทำการลากเส้นเชื่อมเป็นโครงร่างกระดูกดังภาพประกอบ (25c)



ภาพประกอบ 25 a) นำเข้ารูปภาพ b) จุดสำคัญ c) โครงสร้างกระดูก

3.5.6 **การสกัดคุณลักษณะ**

การสกัดคุณลักษณะ (feature extraction) หลังจากการนำโครงร่างกระดูก นำมาคำนวณหา 2 คุณลักษณะ ได้แก่ มุมของจุดสำคัญ ดังภาพประกอบ (23a) และ ระยะห่างระหว่างจุดสำคัญ ดังภาพประกอบ (23b) ในส่วนคุณลักษณะมุมจากจุดสำคัญ ดังสามารถแสดงโดยสมการที่ (3) และสมการที่ (4)

$$\theta_i = arctan\left(\frac{y_i}{x_i}\right) \quad (3)$$

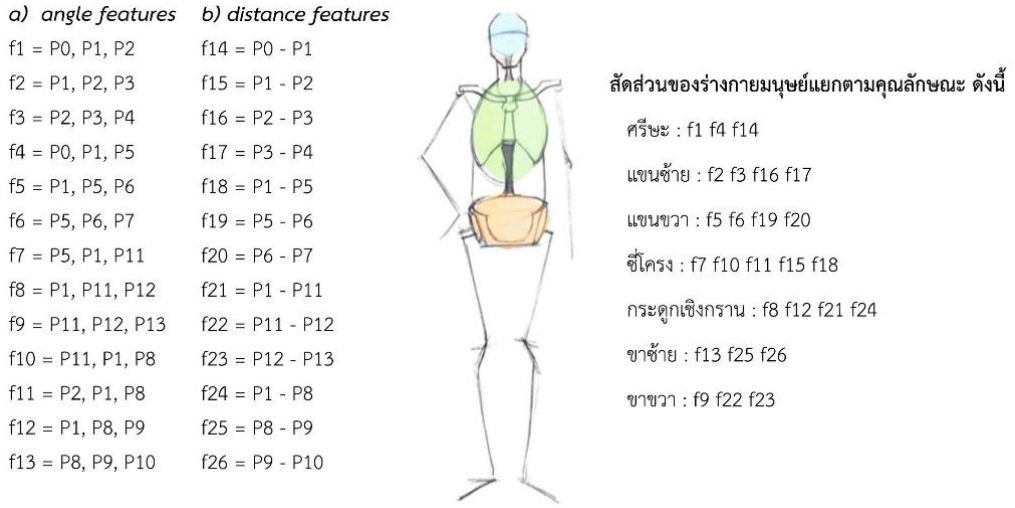
$$F_\theta = \left[\theta_{A,}\theta_{B,}\theta_{C,}\ldots,\theta_H\right] \quad (4)$$

ในแต่ละมุม θ_i คำนวณโดยพิจารณาจากฟังก์ชัน arctan2 [24] ที่เชื่อมต่อระหว่างจุดสำคัญ ด้วยเวกเกตอร์คุณลักษณะที่ประกอบด้วยมุม ที่แสดงในสมการที่ (4) และในส่วนคุณลักษณะการวัดระยะระหว่างจุดสำคัญใช้เครื่องมือ euclidean distance [25] ซึ่งเป็นมาตราวัดระยะพื้นฐานใช้สำหรับการหาระยะทางระหว่างจุด 2 จุด ซึ่งสามารถคำนวณได้ตามสมการที่ (5) และสมการที่ (6)

$$d_1(p,q) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2} \quad (5)$$

$$F_d = \left[d_A, d_B, d_C, \ldots, d_H\right] \quad (6)$$

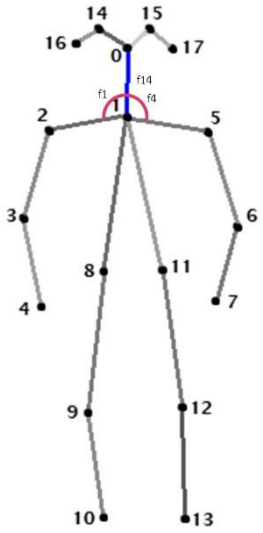
เมื่อทำการสกัดคุณลักษณะตามสมการด้านบนได้ทั้งหมด 26 คุณลักษณะตามตำแหน่งที่ระบุไว้ใน ดังภาพประกอบ 26 จากนั้นคุณลักษณะที่ได้จากจุดสำคัญสามารถระบุค่าคุณลักษณะตามสัดส่วนร่างกายมนุษย์ ดังภาพประกอบ 26



ภาพประกอบ 26 คุณลักษณะตามสัดส่วนร่างกายมนุษย์

จากบทที่ 2 เรื่องสัดส่วนร่างกายมนุษย์สามารถแบ่งร่างกายมนุษย์ได้ทั้งหมด 7 ส่วน จากการอ้างอิงดังกล่าวทำให้ผู้วิจัยกำหนดคุณลักษณะตามสัดส่วนได้ดังนี้

1. ส่วนศีรษะ ประกอบด้วย คุณลักษณะ มุม(f1) มุม(f4) และระยะห่าง(f14) ดังภาพประกอบ 27



ส่วนที่ 1 ศีรษะ

ประกอบด้วยคุณลักษณะดังนี้

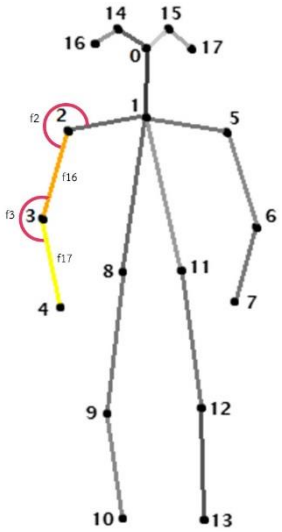
f1 = P0, P1, P2f1

f4 = P0, P1, P5

f14 = P0 - P1

ภาพประกอบ 27 คุณลักษณะตามสัดส่วนของศีรษะ

2. ส่วนแขนซ้าย ประกอบด้วย คุณลักษณะ มุม(f2) มุม(f3) ระยะห่าง(f16) และระยะห่าง(f17) ดังภาพประกอบ 28



ส่วนที่ 2 แขนซ้าย

ประกอบด้วยคุณลักษณะดังนี้

f2 = P1, P2, P3

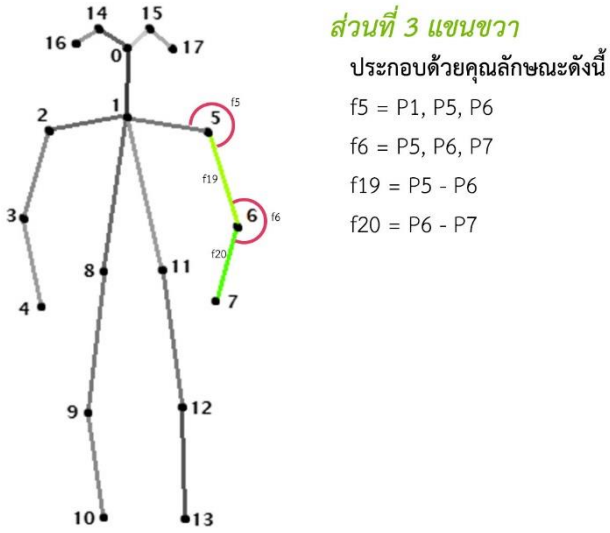
f3 = P2, P3, P4

f16 = P2 - P3

f17 = P3 - P4

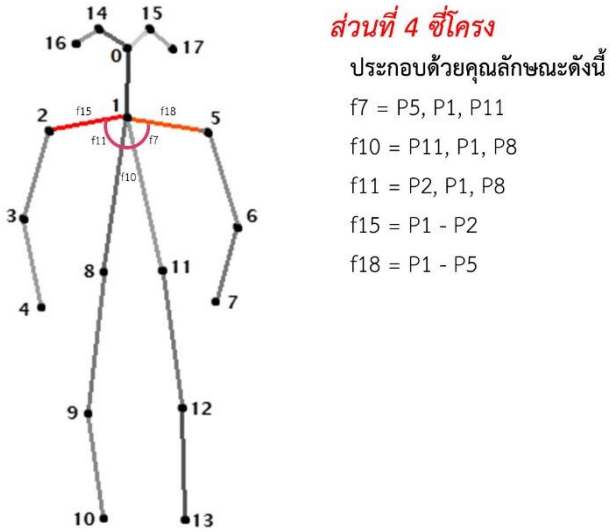
ภาพประกอบ 28 คุณลักษณะตามสัดส่วนของแขนซ้าย

3. ส่วนแขนขวา ประกอบด้วย คุณลักษณะ มุม(f5) มุม(f6) ระยะห่าง(f19) และระยะห่าง(f20) ดังภาพประกอบ 29



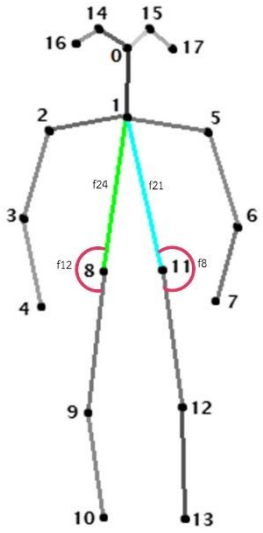
ภาพประกอบ 29 คุณลักษณะตามสัดส่วนของแขนขวา

4. ส่วนกระดูกซี่โครง ประกอบด้วย คุณลักษณะ มุม(f7) มุม(f10) มุม(f11) ระยะห่าง (f15) และระยะห่าง(f18) ดังภาพประกอบ 30



ภาพประกอบ 30 คุณลักษณะตามสัดส่วนของซี่โครง

5. ส่วนกระดูกเชิงกราน ประกอบด้วย คุณลักษณะ มุม(f8) มุม(f12) ระยะห่าง(f21) และระยะห่าง(f24) ดังภาพประกอบ 31



ส่วนที่ 5 กระดูกเชิงกราน

ประกอบด้วยคุณลักษณะดังนี้

f8 = P1, P11, P12

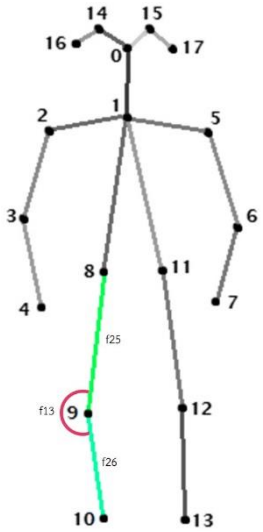
f12 = P1, P8, P9

f21 = P1 - P11

f24 = P1 - P8

ภาพประกอบ 31 คุณลักษณะตามสัดส่วนของกระดูกเชิงกราน

6. ส่วนขาซ้าย ประกอบด้วย คุณลักษณะ มุม(f13) ระยะห่าง(f25) และระยะห่าง (f26) ดังภาพประกอบ 32



ส่วนที่ 6 ขาซ้าย

ประกอบด้วยคุณลักษณะดังนี้

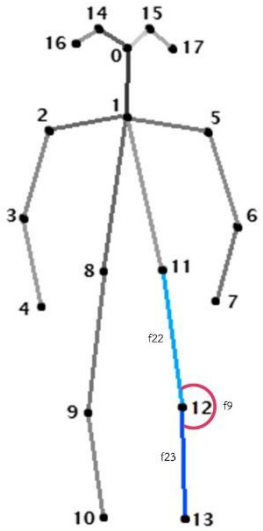
f13 = P8, P9, P10

f25 = P8 - P9

f26 = P9 - P10

ภาพประกอบ 32 คุณลักษณะตามสัดส่วนของขาซ้าย

7. ส่วนขาขวา ประกอบด้วย คุณลักษณะ มุม(f9) ระยะห่าง(f22) และระยะห่าง(f23) ดังภาพประกอบ 33



ส่วนที่ 7 ขาขวา

ประกอบด้วยคุณลักษณะดังนี้

f9 = P11, P12, P13

f22 = P11 - P12

f23 = P12 - P13

ภาพประกอบ 33 คุณลักษณะตามสัดส่วนของขาขวา

แต่เนื่องด้วยตำแหน่งจุดสำคัญที่ใช้ในการประเมินท่าทางผู้วิจัยเห็นว่าตำแหน่งจุดสำคัญที่ P14(ตาขวา) P15(ตาซ้าย) P16(หูขวา) P17(หูซ้าย) ในบางครั้งการยืนของนายแบบการหันหน้าอาจจะส่งผลให้ได้จุดสำคัญไม่ครบ ดังนั้นจึงไม่นำมาคิดคุณลักษณะบริเวณส่วนดวงตาและหู เนื่องจากในงานวิจัยนี้จะใช้การประเมินเฉพาะส่วนโครงสร้างและสัดส่วนเท่านั้น

3.5.7 การนำคุณลักษณะมาใช้ในการจำแนก

งานวิจัยนี้ใช้การนำคุณลักษณะมาใช้จำแนกกลุ่ม (feature vector for classification) ทั้ง 26 คุณลักษณะ ซึ่งในแต่ละคุณลักษณะจะประกอบไปด้วยค่าคุณลักษณะมุม (feature angle) ตั้งแต่ f1-f13 ดังภาพประกอบ (23a) และค่าคุณลักษณะระยะห่าง (feature distance) ตั้งแต่ f14-f26 ดังภาพประกอบ (23b) ในการสกัดคุณลักษณะทั้ง 26 คุณลักษณะของนายแบบแต่ละคน ดังแสดงตามตารางที่ 6-8

ตาราง 6 ค่าที่สกัดคุณลักษณะจากมุม และระยะจากจุดสำคัญตามกลุ่มของนายแบบคนที่ 1

นายแบบ	นักศึกษา	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23	f24	f25	f26	กลุ่ม	รายละเอียดกลุ่ม
figure01	stu03	252.1	141.6	25.3	120.3	275.1	125.5	33.4	283.3	344.6	173.4	273.3	4.9	336.8	71.5	55.0	74.0	175.6	153.0	182.2	188.0	87.3	15.2	44.4	576.9	356.7	170.8	1	แย่
figure01	stu15	249.1	148.4	350.4	63.6	251.1	260.8	349.4	5.5	262.3	17.7	277.1	175.4	12.1	103.8	62.0	76.3	53.2	62.3	106.5	175.3	336.8	447.8	22.6	378.9	160.2	359.7	1	แย่
figure01	stu18	246.9	134.1	27.8	110.4	136.5	338.4	339.7	187.7	359.3	18.5	273.9	174.6	12.1	112.0	69.0	94.8	218.8	165.5	59.0	247.3	377.2	150.1	639.5	377.9	154.1	336.8	1	แย่
figure01	stu04	249.9	138.1	21.5	106.5	286.8	136.2	345.4	2.4	251.1	14.6	282.0	12.9	332.2	110.2	73.8	97.3	225.1	144.9	218.5	171.0	524.0	641.5	23.7	555.0	363.6	145.9	2	ปานกลาง
figure01	stu07	256.4	121.5	104.6	66.3	243.6	4.7	343.3	182.9	190.7	21.6	271.7	169.4	181.5	102.5	55.2	78.6	6.4	58.2	113.5	89.4	200.6	180.8	154.0	207.5	171.4	183.1	2	ปานกลาง
figure01	stu08	251.4	136.1	20.3	105.7	286.4	20.8	42.4	59.3	222.0	148.6	256.4	172.5	11.2	90.8	58.0	106.8	215.0	141.7	197.6	72.7	108.2	49.2	421.2	310.7	61.1	463.0	2	ปานกลาง
figure01	stu10	249.1	357.3	238.0	171.0	170.4	181.3	359.7	232.0	7.1	157.3	258.7	175.7	192.7	103.8	69.0	128.1	82.7	174.7	177.1	177.3	111.2	24.6	68.9	168.3	148.5	217.3	2	ปานกลาง
figure01	stu20	259.2	129.5	27.3	111.7	282.1	129.8	348.3	2.7	273.5	10.4	268.7	7.0	338.3	113.6	62.3	89.1	187.9	129.6	195.3	194.0	539.1	663.2	18.0	560.5	360.9	180.0	2	ปานกลาง
figure01	stu21	257.1	127.8	142.2	66.7	252.4	258.7	349.7	200.0	349.9	20.0	270.0	179.0	9.9	94.4	66.0	107.6	6.0	66.1	116.8	189.0	370.4	200.0	667.3	399.0	177.0	381.6	2	ปานกลาง
figure01	stu01	248.2	146.8	19.5	114.0	283.6	122.9	348.0	2.9	277.7	15.2	280.5	7.1	331.7	106.0	58.3	90.3	201.8	143.4	200.2	177.3	540.1	656.9	29.0	543.8	343.2	189.8	3	ดี
figure01	stu11	248.8	136.7	113.5	69.2	247.2	258.4	351.7	185.7	357.9	12.4	281.9	174.8	10.9	100.2	51.4	96.5	96.2	54.3	98.5	192.9	374.0	148.1	628.3	366.5	166.0	349.3	3	ดี
figure01	stu13	250.1	357.6	247.2	172.9	165.9	182.6	358.5	244.2	354.0	162.3	262.8	170.8	190.2	96.8	58.0	120.1	112.7	211.3	177.0	171.0	103.8	22.8	63.3	207.6	166.8	178.0	3	ดี
figure01	stu17	258.9	126.4	24.8	91.1	227.5	261.6	343.6	191.8	354.6	16.8	268.0	171.8	12.5	101.2	62.3	92.8	202.5	111.3	69.4	208.9	381.1	166.0	649.6	417.8	137.4	369.5	3	ดี

จากตาราง 7 โดยแถวที่ 1 ระบุชื่อนายแบบคนใด แถวที่ 2 ระบุถึงผลงานภาพวาดของนักศึกษา แถวที่ 3-15 เป็นคุณลักษณะของ f1-f13 โดยค่าเป็นองศา แถวที่ 16-28 เป็นคุณลักษณะของ f14-f26 โดยค่าระยะห่างมีหน่วยเป็นพิกเซล และแถวที่ 29-30 เป็นการระบุภาพวาดแต่ละกลุ่ม ดี ปานกลาง และแย่

ตาราง 7 ค่าที่สกัดคุณลักษณะจากมุม และระยะจากจุดสำคัญตามกลุ่มของนายแบบคนที่ 2

นายแบบ	นักศึกษา	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23	f24	f25	f26	กลุ่ม	รายละเอียดกลุ่ม
figure02	stu18	277.2	121.1	114.4	85.1	245.2	166.3	340.3	180.4	0.8	24.1	278.7	163.4	182.1	125.2	72.2	90.4	59.2	77.8	88.2	24.8	221.4	150.8	510.2	246.0	103.2	131.1	1	แย่
figure02	stu20	274.4	128.4	112.5	94.6	244.6	74.4	340.1	191.6	181.4	27.7	288.7	168.8	175.2	110.6	62.3	94.0	63.2	65.3	100.0	31.8	188.0	217.3	154.1	193.1	206.1	142.2	1	แย่
figure02	stu22	277.3	131.7	33.4	146.7	270.2	151.4	350.0	357.5	269.6	15.8	287.7	12.7	336.4	101.2	74.0	74.6	195.2	124.2	225.6	127.7	569.7	683.1	33.5	554.0	366.6	160.2	1	แย่
figure02	stu10	298.0	122.6	46.1	122.0	289.0	146.7	353.3	357.3	264.3	11.3	264.9	9.9	337.6	101.2	56.1	111.0	22.0	64.3	207.2	217.0	533.1	644.9	25.7	505.0	326.9	167.0	2	ปานกลาง
figure02	stu15	285.6	119.2	95.8	95.4	244.2	268.2	26.1	272.6	344.1	204.0	274.5	167.2	19.7	116.9	65.3	97.1	43.3	70.3	105.8	202.8	130.0	29.6	49.6	364.3	154.2	315.4	2	ปานกลาง
figure02	stu21	276.9	119.3	120.6	91.6	248.9	68.1	342.3	189.3	181.6	24.7	279.9	171.1	177.8	91.7	59.0	104.3	79.8	65.3	108.2	37.5	212.0	200.9	166.4	214.2	166.0	148.0	2	ปานกลาง
figure02	stu01	293.5	132.1	29.2	143.1	296.1	130.4	24.0	292.8	16.0	203.9	277.8	11.8	144.0	117.8	73.0	70.1	196.4	151.8	221.6	131.7	130.1	51.4	474.7	559.2	357.7	350.3	3	ดี
figure02	stu02	271.3	109.5	138.9	91.2	306.0	151.1	31.1	259.7	0.5	173.8	259.8	190.4	182.5	115.5	69.2	83.0	32.2	80.2	228.1	160.2	127.3	19.0	38.9	205.4	160.4	160.0	3	ดี
figure02	stu08	281.8	122.2	117.4	93.9	234.4	279.0	348.0	188.6	353.4	18.3	274.1	169.2	17.3	108.9	65.2	92.2	82.3	81.2	103.1	227.6	388.6	188.0	696.5	364.0	177.1	336.4	3	ดี
figure02	stu11	284.2	124.7	118.3	94.4	253.2	67.6	339.6	185.7	180.4	23.4	284.4	165.6	175.7	118.6	62.0	75.5	64.0	71.0	89.4	61.0	190.4	188.3	137.2	205.4	166.0	148.4	3	ดี
figure02	stu12	291.2	126.6	113.0	105.4	301.5	159.1	22.1	277.2	195.3	188.5	256.1	187.1	176.0	110.5	58.0	92.2	87.0	59.3	228.6	131.5	123.3	22.8	36.0	212.2	160.1	192.4	3	ดี
figure02	stu13	279.4	118.6	128.1	94.7	251.8	253.0	350.6	358.3	282.2	16.4	280.8	171.3	17.4	105.3	70.3	99.4	78.3	65.9	102.9	227.0	400.7	519.8	22.0	361.9	160.2	319.2	3	ดี

ตาราง 8 ค่าที่สกัดคุณลักษณะจากมุม และระยะจากจุดสำคัญตามกลุ่มของนายแบบคนที่ 3

นายแบบ	นักศึกษา	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15	f16	f17	f18	f19	f20	f21	f22	f23	f24	f25	f26	กลุ่ม	รายละเอียดกลุ่ม
figure03	stu04	275.5	0.2	290.4	190.5	169.5	18.0	356.5	83.9	4.9	166.2	256.8	187.7	354.1	86.0	62.3	120.6	247.3	424.0	165.0	365.9	103.6	48.0	70.3	431.9	143.0	671.6	1	แย่
figure03	stu11	264.7	5.3	302.8	189.8	171.1	15.2	357.8	58.9	19.6	168.3	270.4	188.5	353.3	108.0	65.3	135.0	210.5	365.3	194.0	368.4	125.9	55.9	73.2	389.4	172.4	680.0	1	แย่
figure03	stu14	259.8	131.4	29.9	110.8	2.2	102.2	30.0	67.7	11.4	185.1	290.9	30.9	129.6	125.0	77.9	94.9	209.8	105.8	12.1	213.0	149.1	64.3	88.2	350.7	155.7	369.1	1	แย่
figure03	stu15	259.3	5.1	249.0	194.6	180.2	175.8	356.5	75.0	359.1	160.6	270.3	197.8	349.8	86.3	64.3	130.0	106.3	184.8	226.6	184.0	103.3	60.0	85.7	200.8	222.3	517.2	1	แย่
figure03	stu02	257.6	144.4	26.3	122.8	281.3	148.4	20.4	284.9	29.2	193.1	273.2	359.9	269.0	97.0	78.9	92.8	231.6	147.6	231.3	151.6	104.9	83.7	541.3	266.5	382.1	33.5	2	ปานกลาง
figure03	stu10	264.5	102.4	63.5	90.0	298.8	165.3	26.9	289.1	27.0	193.4	265.4	0.4	279.8	80.0	62.3	91.7	14.9	66.0	228.3	147.5	92.7	77.2	505.7	226.6	324.7	29.0	2	ปานกลาง
figure03	stu17	268.6	108.3	44.5	197.5	190.5	164.6	343.7	285.7	338.9	184.8	269.1	201.6	169.4	93.1	63.0	98.0	178.4	207.5	199.2	217.5	107.0	26.0	47.2	232.2	238.1	143.0	2	ปานกลาง
figure03	stu18	273.2	86.5	54.3	120.2	298.2	140.8	38.3	64.1	16.5	184.4	279.9	16.0	141.7	126.2	80.0	114.2	197.6	176.2	240.0	160.2	143.7	66.1	88.2	550.3	342.6	379.7	2	ปานกลาง
figure03	stu22	263.6	8.9	231.0	191.9	182.2	173.8	346.9	280.0	349.3	179.6	268.5	191.3	184.9	91.1	70.9	132.0	74.7	226.0	198.1	188.9	109.7	33.0	59.0	237.9	205.0	149.8	2	ปานกลาง
figure03	stu01	265.9	2.0	298.6	192.4	175.4	13.8	355.8	72.3	10.7	167.8	268.9	190.8	351.7	103.0	70.2	135.1	235.7	391.1	213.0	404.4	114.9	59.0	80.0	400.6	183.9	696.8	3	ดี
figure03	stu07	265.1	141.0	16.4	99.3	228.1	251.0	341.6	189.9	186.1	24.2	291.2	176.2	177.9	97.3	69.9	92.8	55.2	66.3	127.2	76.3	209.7	183.1	206.5	209.7	173.8	206.2	3	ดี
figure03	stu08	259.3	4.1	238.2	191.3	180.6	166.0	347.9	279.5	349.3	177.1	270.2	184.8	188.7	91.0	77.9	154.7	112.8	196.6	173.6	211.5	109.5	29.0	59.0	204.8	166.5	206.0	3	ดี
figure03	stu12	271.7	135.4	25.9	126.8	288.0	145.0	347.1	359.1	287.1	17.9	284.7	12.4	336.4	103.0	58.0	96.9	196.4	147.9	210.0	186.3	599.9	720.8	26.7	601.8	416.8	177.3	3	ดี

3.5.8 การจำแนกกลุ่มด้วย ANN

การจำแนกกลุ่มด้วย ANN เป็นการจำแนกกลุ่มโดยนำค่า feature vector เพื่อใช้สำหรับ ANN (Artificial Neural Network) โดยสามารถที่จะจำแนกประเภทในการจัดหมวดหมู่ด้วยชุดข้อมูลจากภาพวาด 66 ภาพ OpenPose สามารถตรวจจับจุดสำคัญจากการทดสอบจากภาพวาดพบว่ามีภาพวาดจำนวน 41 ภาพที่สามารถตรวจพบจุดสำคัญได้มากกว่า 14 จุดคิดเป็น 62.12% จากภาพวาดทั้งหมด และด้วยชุดข้อมูลนี้ใช้สำหรับการ training ข้อมูลซึ่งจะเหลือภาพวาดที่ใช้ training จำนวน 32 ภาพและภาพที่ใช้ในการทดสอบประสิทธิภาพจำนวน 9 ภาพ โดยแบ่งข้อมูลเป็น train และ test ดังตาราง 9

ตาราง 9 การแบ่งจำนวนภาพวาดแต่ละกลุ่ม

นายแบบ	กลุ่มจำแนก	จำนวนภาพที่ใช้ training	จำนวนภาพที่ใช้ testing
Figure 1	ดี	3	1
Figure 1	ปานกลาง	6	1
Figure 1	แย่	3	1
	รวม	12	3
Figure 2	ดี	5	1
Figure 2	ปานกลาง	3	1
Figure 2	แย่	2	1
	รวม	10	3
Figure 3	ดี	3	1
Figure 3	ปานกลาง	3	1
Figure 3	แย่	4	1
	รวม	10	3

ค่าตัวแปรที่ใช้ในการกำหนดค่า ANN parameter

ค่าตัวแปรที่ใช้ในการกำหนดค่า ANN parameter นั้นผู้วิจัยได้กำหนดตัวอย่างพารามิเตอร์ที่ใช้ในการเรียนรู้ ดังตาราง 10

ตาราง 10 การกำหนดค่าพารามิเตอร์สำหรับใช้ในการทดสอบภาพวาด

ชื่อพารามิเตอร์	ค่าตัวแปรเริ่มต้น
Input_node	26
Hidden_node	38
Output_node	3

โดยในส่วนของพารามิเตอร์เป็นค่าเริ่มตัวของ ANN parameter ซึ่งประกอบดังนี้

1. Input nodes เป็นเวกเตอร์คุณลักษณะที่ประกอบด้วย 26 คุณลักษณะดังแสดงในตาราง 10

2. Hidden node เป็นโหนดเลเยอร์ระหว่างโหนดอินพุตและโหนดเอาต์พุตที่มีอย่างน้อยหนึ่งเลเยอร์โดยมีจำนวน hidden node มากกว่า Input nodes > 20%

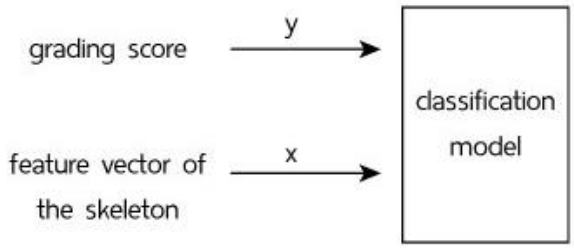
3. Output node เป็น class ที่แบ่งประเภทจำนวน 3 กลุ่ม ได้แก่ ดี ปานกลาง และแย่

หลังจากใช้ค่าพารามิเตอร์ที่ใช้ ต่อไปเป็นกระบวนการแบ่งข้อมูล train และ test ดังนี้

1. Training phase

44

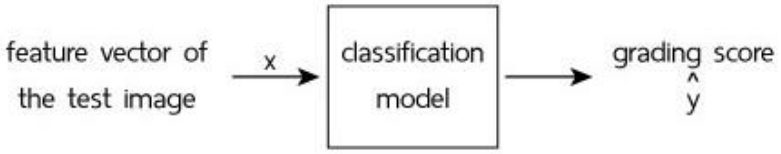
Training phase เป็นข้อมูลที่ใช้สำหรับการฝึกฝน เพื่อการจำแนกประเภทของภาพวาดคนเต็มตัวในส่วนนี้ได้ใช้วิธีการของ ANN โดยหลักการของมันก็คือ training phase และ test phase ซึ่งขั้นตอนในการ training โดยมีข้อมูลค่า x คือ feature vector ของโครงสร้างร่างกาย (โดย feature vector คือ keypoints ที่ได้มาจาก skeleton จากภาพวาด) และค่า y คือ ประเภทที่จะจำแนกกลุ่มภาพวาด (โดยมีด้วยกัน 3 ประเภท คือ กลุ่มภาพวาดดี กลุ่มภาพวาดปานกลาง และกลุ่มภาพวาดแย่) ดังภาพประกอบ 34



ภาพประกอบ 34 กระบวนการทำงาน training ของ ANN for classification model

2. Test phase

ในส่วนของ test phase จะนำข้อมูลใหม่ซึ่งเป็น feature vector ที่จะใช้ในการทดสอบ ซึ่งผลลัพธ์ที่ได้ คือ classification model มีการทำนายกลุ่มที่นักศึกษาวาดภาพได้ว่าอยู่ในกลุ่มใด ดังภาพประกอบ 35



ภาพประกอบ 35 กระบวนการ testing ของ ANN for classification model

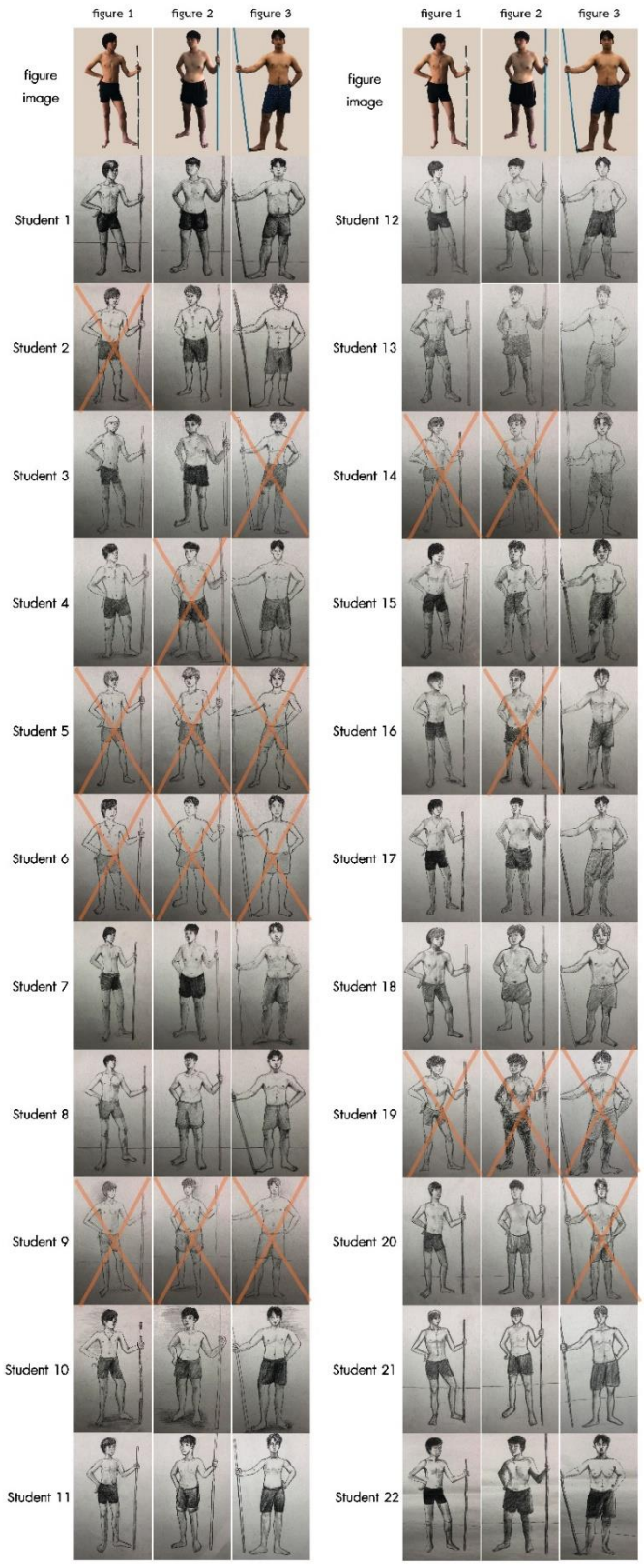
บทที่ 4
ผลการดำเนินการวิจัย

ในการวิจัยการทำนายภาพวาดโดยแบ่งภาพวาดตามกลุ่ม (class good moderate poor) แต่ละภาพนายแบบเรียบร้อยแล้ว จากนั้นทำการฝึกอบรมข้อมูล โดยสกัดคุณลักษณะ ทั้ง 26 คุณลักษณะมาสร้างแบบจำลองสำหรับฝึกอบรมข้อมูลด้วยข้อมูลภาพวาดเมื่อได้แบบจำลองสำหรับฝึกอบรมภาพวาด จากนั้นทำการใช้ภาพวาดทดสอบที่ไม่ผ่านการฝึกอบรมเข้าแบบจำลองให้โครงข่ายประสาทเทียมทำนายผลว่าภาพวาดชิ้นนี้อยู่ในกลุ่ม ดี ปานกลาง แย่ ของกลุ่มภาพวาดใด โดยผู้วิจัยได้ดำเนินการวิจัยโดยการศึกษาตามกระบวนการและขั้นตอนตลอดจนการวัดประสิทธิภาพ เพื่อให้บรรลุจุดประสงค์ของการวิจัยที่ได้กำหนดไว้ได้ดังนี้

1. การหาจุดสำคัญด้วย OpenPose
2. การสร้างแบบจำลอง ANN
3. การเปรียบเทียบการทำนายผลจากกลุ่มผลการประเมินภาพวาด
4. การวัดประสิทธิภาพของ ANN

4.1 การหาจุดสำคัญด้วย OpenPose

การวิจัยนี้ได้ผลการหาจุดสำคัญด้วย OpenPose โดยนำข้อมูลซึ่งเป็นกลุ่มข้อมูลซึ่งเป็นภาพวาดคนเต็มตัวจากนักศึกษาจำนวน 22 คน คนละ 3 แบบ รวม 66 ภาพ ดังภาพประกอบ 36 สังเกตว่าภาพประกอบ 36 บางภาพจะมีเครื่องหมายกากบาทแสดงให้เห็นว่า OpenPose ไม่สามารถตรวจจับจุดสำคัญได้ครบ 18 จุด



ภาพประกอบ 36 นายแบบและภาพวาดของนักศึกษา จำนวน 22 คน รวม 66 ภาพ

47

4.2 การสร้างแบบจำลอง ANN

ผลการสร้างแบบจำลอง ANN เมื่อทำการสร้างแบบจำลอง ANN นั้นการทดสอบด้วยภาพวาดปรากฏว่าภาพวาดคนเต็มตัว สามารถที่จะประเมินตามกลุ่ม ดี ปานกลาง และแย่ โดยเริ่มจากการทดสอบภาพวาดจากภาพนายแบบคนที่ 1 ถึง 3 ใช้ค่า sigmoid (1.0) สำหรับใช้สร้างแบบจำลอง และผลการทำนายจากตารางที่ 11

ตาราง 11 การทำนายแต่ละกลุ่มภาพวาด

ลำดับการทดสอบ	True_label	Predict_label
1	Poor	Poor
2	Moderate	Moderate
3	Good	Moderate
4	Poor	Good
5	Moderate	Moderate
6	Good	Good
7	Poor	Poor
8	Moderate	Moderate
9	Good	Moderate

จากตาราง 11 แสดงให้เห็นว่าการทดสอบภาพวาดมีทั้งหมด 9 ภาพโดยค่าที่ผู้เชี่ยวชาญประเมินในคอลัมน์ true_label และประเมินจากแบบจำลองในคอลัมน์ predict_label ซึ่งผลของการทำนายสามารถทำนายความแม่นยำได้ 67.33% สามารถทำนายถูก 6 ภาพจาก 9 ภาพและทำนายผิด 3 ภาพจาก 9 ภาพ

4.3 การเปรียบเทียบการทำนายผลจากกลุ่มผลการประเมินภาพวาด

การเปรียบเทียบการทำนายผลจากกลุ่มผลการประเมินภาพวาด เมื่อได้แบบจำลองจากการสกัดคุณลักษณะจากภาพวาดที่ใช้ในการทดสอบนำมาเข้าแบบจำลองการฝึกอบรม (model training) ผ่านกระบวนการ ANN เพื่อให้แบบจำลองทำนายข้อมูลจากภาพวาด โดยดึงคุณลักษณะ

48

ของแต่ละกลุ่ม ดี ปานกลาง แย่ ว่าภาพวาดชิ้นนี้อยู่ในกลุ่มใด โดยนำภาพวาดมาทดสอบ เพื่อทำนายกลุ่มและประเมินประสิทธิภาพออกเป็นค่าคะแนนความเชื่อมั่น (confidence score) ซึ่งค่าคะแนนความเชื่อมั่นมาจากค่าของ weight output ซึ่งเป็นตัวแปรที่จะเปลี่ยนแปลงค่าไประหว่างการ ฝึกอบรมเพื่อให้ fit กับข้อมูล เรียกว่า trainable parameter ซึ่งในทุก node ใน neural network จะมีตัวแปร weight นี้คอยปรับค่าของอินพุตโดยเริ่มต้นของการฝึกอบรมและสุ่มค่า weight แล้วทำการทำซ้ำลูปเดิมเพื่อให้ได้ค่าคะแนนความเชื่อมั่น ออกมาจากโมเดลที่ได้จาก ANN ที่ใช้ภาพวาดในการทดสอบสามารถที่ทำนายกลุ่มภาพวาดว่าอยู่ในกลุ่มใดออกมาเป็นเปอร์เซ็นต์ ดังภาพประกอบ 37

ภาพประกอบ 37 การทดสอบภาพวาดด้วยแบบจำลอง ANN

49

4.4 การวัดประสิทธิภาพของ ANN

งานวิจัยนี้ได้วัดประสิทธิภาพของ ANN โดยใช้เทคนิคโครงข่ายประสาทเทียมเชิงลึกในการประเมินกลุ่มจากภาพวาดโดยใช้ไลบรารี OpenPose ในการตรวจจับหาจุดสำคัญจากภาพวาดโดยจุดสำคัญแต่ละจุดจะเชื่อมต่อเป็นโครงสร้างร่างกายมนุษย์ ซึ่งผลจากการหาจุดสำคัญจะนำมาหามุมของแต่ละจุดสำคัญ และระยะห่างระหว่างจุดสำคัญ เพื่อใช้สำหรับคุณลักษณะ ทั้ง 26 คุณลักษณะ เพื่อนำไปสร้างแบบจำลองโดยใช้ ANN ในการจำแนกแต่ละ

ประเภทได้ทำการวัดประสิทธิภาพของ ANN จาก confusion matrix ซึ่งประสิทธิภาพของแบบจำลองที่ได้จากการประเมินค่าความแม่นยำอยู่ที่ 67.67% ดังตารางที่ 12

ตาราง 12 แบบจำลองและการเปรียบเทียบประสิทธิภาพกลุ่มตัวอย่างของชุดข้อมูลภาพวาด

กลุ่ม	precision	recall	f1-score	support
Poor	1.00	0.67	0.80	3
Moderate	0.60	1.00	0.75	3
Good	0.50	0.33	0.40	3
Accuracy			0.67	9
Macro avg	0.70	0.67	0.65	9
Weighted avg	0.70	0.67	0.65	9

จากผลการทดลองกลุ่มตัวอย่างชุดข้อมูล พบว่าแบบจำลองของกลุ่ม ได้ค่าความเที่ยงตรง (precision) คิดเป็น 70 เปอร์เซ็นต์ ได้ค่าการระลึก (recall) คิดเป็น 67 เปอร์เซ็นต์ ได้ค่า F1 (F1-Score) คิดเป็น 65 เปอร์เซ็นต์

บทที่ 5
สรุปผลการวิจัย อภิปรายผล และข้อเสนอแนะ

ในการวิจัยนี้การประเมินภาพวาดคนเต็มตัว โดยเริ่มจากหานายแบบในท่ายืนเต็มตัว แล้วให้นักศึกษาวาดภาพนายแบบ โดยการเก็บรวบรวมข้อมูลเพื่อใช้สำหรับการประมาณท่าทาง โดยใช้ไลบรารี OpenPose เพื่อหาจุดสำคัญแต่ละจุด จากตรวจจับโดยใช้ OpenPose จะตรวจพบจุดสำคัญประมาณ 62.12% (41/66) ของชุดข้อมูลภาพวาดทั้งหมด โดยหารูปแบบโครงสร้างกระดูกตามท่าทาง โดยจากจุดสำคัญที่ได้มาแต่ละจุดนั้นนำมาสกัดคุณลักษณะทั้งหมด 26 คุณลักษณะเพื่อใช้สำหรับการจำแนกออกเป็นกลุ่ม 3 กลุ่ม ดี ปานกลาง แย่ หลังจากการฝึกอบรมข้อมูลเพื่อใช้แบบจำลองสำหรับทดสอบภาพวาดโดยสามารถประเมินระดับการวาดภาพ

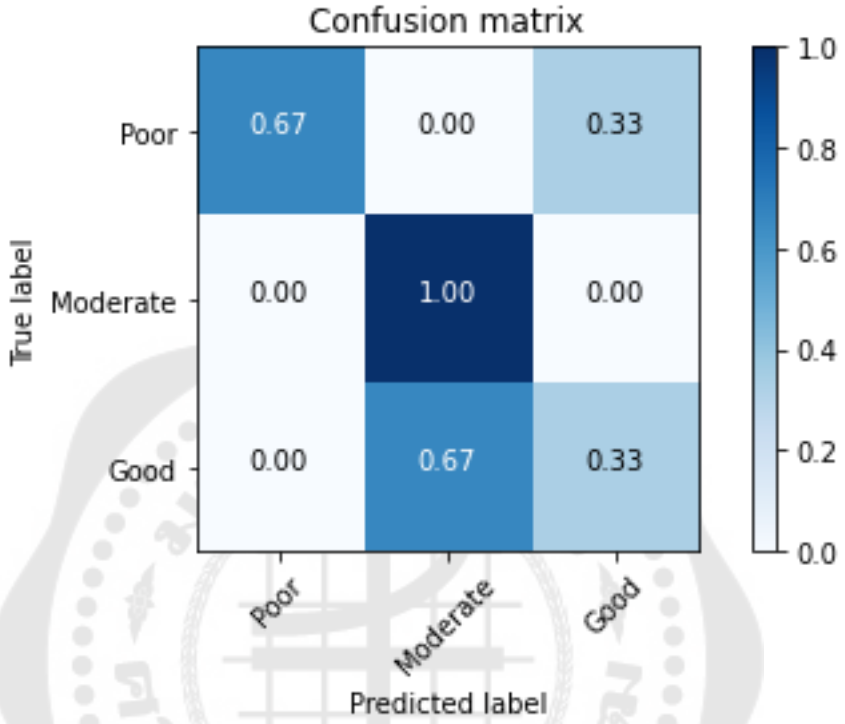
ผู้วิจัยได้วัดประสิทธิภาพแบบจำลองการประเมินระดับการวาดภาพโดยใช้เทคนิคโครงข่ายประสาทเทียมเชิงลึก และได้วัดประสิทธิภาพของแบบจำลองในการจำแนกการวาดภาพเต็มตัว เพื่อนำไปเปรียบเทียบและสรุปผลโดยแบ่งหัวข้อในการสรุปผลได้ดังต่อไปนี้

1. อภิปรายผลการวิจัย
2. สรุปผลการวิจัย
3. ข้อเสนอแนะ

5.1 อภิปรายผลการวิจัย

งานวิจัยนี้นำเสนอวิธีการประเมินการวาดภาพคนเต็มตัวจากภาพถ่าย ด้วยการสกัดคุณลักษณะและเทคนิคโครงข่ายประสาทเทียมเชิงลึก โดยผ่านกระบวนการ 8 ขั้นตอน คือ 1) การนำเข้าข้อมูลรูปภาพ 2) ใช้ OpenPose หาตำแหน่งจุดสำคัญ 3) ตำแหน่งจุดสำคัญทั้ง 18 จุด 4) สร้างโครงร่างกระดูกจากข้อต่อของจุดสำคัญ 5) ร่างเส้นเชื่อมระหว่างจุดสำคัญเป็นโครงร่างกระดูก 6) การสกัดคุณลักษณะมุมและระยะห่าง 7) สกัดคุณลักษณะทั้ง 26 คุณลักษณะ 8) นำคุณลักษณะสร้างแบบจำลองจากการแบ่งกลุ่มภาพวาดแต่ละกลุ่ม ดี ปานกลาง แย่ จากภาพนายแบบ 3 คนจากเทคนิค ANN แล้วทำการฝึกอบรมแบบจำลอง จากนั้นนำภาพวาดสำหรับทดสอบการจำแนกภาพวาดว่าอยู่กลุ่มใด โดยตัวอย่างที่ใช้ในการศึกษาเป็นกลุ่มข้อมูลซึ่งเป็นภาพวาดคนเต็มตัวจากนักศึกษาจำนวน 22 คน คนละ 3 แบบ รวม 66 ภาพ จากผลการทดลองกลุ่มชุดข้อมูลพบว่าแบบจำลองของแต่ละกลุ่มได้ค่ากลุ่มดี (67.33 เปอร์เซ็นต์) กลุ่มปานกลาง (100.00 เปอร์เซ็นต์) กลุ่มแย่ (33.67 เปอร์เซ็นต์) และให้ค่าความแม่นยำ (accuracy) 67.33

เปอร์เซ็นต์ ค่าความเที่ยงตรง (precision) 70.00 เปอร์เซ็นต์ ค่าการระลึก (recall) 67.67 เปอร์เซ็นต์ ค่า F1 (F1-Score) 65.00 เปอร์เซ็นต์



ภาพประกอบ 38 Confusion Matrix ของแบบจำลองจากชุดข้อมูลภาพวาด

จากผลการคำนวณ confusion matrix เพื่อประเมินประสิทธิภาพของแบบจำลองจากชุดข้อมูลภาพวาด ดังภาพประกอบ 38 จากแบบจำลอง ANN สามารถจำแนกภาพวาดจากตัวอย่าง (test set) จำนวน 9 ภาพได้ถูกต้องจำนวน 6 ภาพคิดเป็นร้อยละ 66.67% และจำแนกผิดจำนวน 3 ภาพ คิดเป็นร้อยละ 33.33% โดยสรุป ข้อจำกัดของงานวิจัยดังนี้

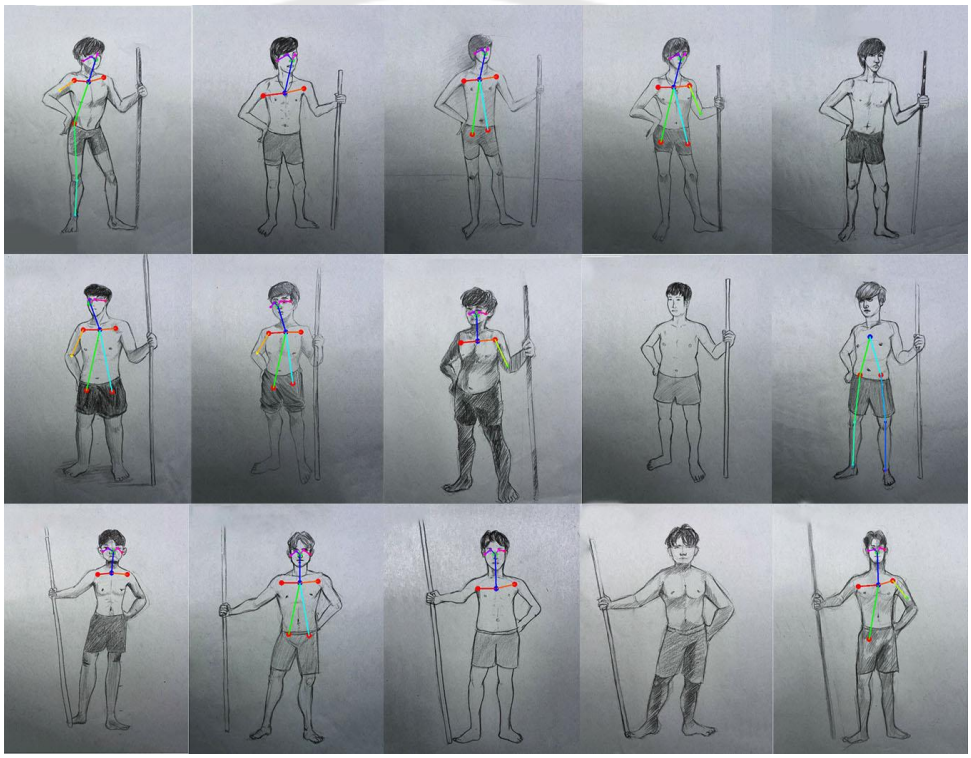
1. จำนวนภาพวาดที่ใช้ในการฝึกอบรมจากการเก็บข้อมูลคือกลุ่มนักศึกษาจำนวน 1 ห้องเรียนซึ่งมีนักศึกษาประมาณ 22 คน ซึ่งปริมาณที่ใช้ในการฝึกอบรมมีค่อนข้างน้อยสำหรับการทดลองในครั้งนี้

2. จากประเด็นข้อที่ 1 ทำให้การแบ่งคะแนนจากปกติ มีเกรด A B+ B C+ C D+ D F ซึ่งมาจากการตรวจของผู้เชี่ยวชาญทางด้านการวาดภาพ ทำให้ภาพวาดที่ทดลองมีไม่ครบทุกเกรดทำให้ผู้วิจัยต้องปรับให้เหลือ 3 กลุ่มคือ ดี ปานกลาง แย่เพื่อให้แบบจำลองสามารถจำแนกจากภาพวาดได้

3. การใช้เทคนิค OpenPose ในการตรวจจับหาจุดสำคัญในภาพวาดมีผลงานบ้างชิ้นที่ไม่สามารถตรวจจับหาจุดสำคัญได้ครบทั้ง 18 จุด ดังภาพประกอบ 39 เพราะฉะนั้นการหาวิธีหาจุดสำคัญที่เหมาะสมกับภาพวาดควรเป็นไปในลักษณะใด

4. จากการทดลองจะเห็นได้ว่าจุดสำคัญที่อยู่บริเวณศีรษะ เช่น หูทั้งสองข้าง ตาทั้งสองข้าง ไม่จำเป็นในการหาคุณลักษณะเนื่องจากผู้วิจัยวัดจากมุมและระยะลำตัว แขน ขา ก็เพียงพอแล้ว

5. ปัญหาการหาจุดสำคัญจากภาพวาดที่ไม่พบจุดสำคัญทั้งหมด 18 จุด ทำให้ไม่สามารถระบุมุมและระยะห่างของคุณลักษณะได้



ภาพประกอบ 39 ตัวอย่างภาพวาดที่ OpenPose ไม่สามารถหาจุดสำคัญได้ครบทั้ง 18 จุด

ดังนั้นสำหรับการจำแนกประเภทภาพวาดคนเต็มตัวจากลุ่ม ดี ปานกลาง แย่ จากภาพวาดคนเต็มตัว ด้วยแบบจำลอง ANN สามารถนำไปประยุกต์ใช้ในการประเมินการวาดภาพคนเต็มตัวในรายวิชาวาดเส้นได้

53

5.2 สรุปผลการวิจัย

การวาดภาพคนเต็มตัวเป็นการฝึกทักษะทางด้านศิลปะรูปแบบหนึ่งที่นักศึกษาสายงานแอนิเมชันจำเป็นต้องศึกษาและเรียนรู้ทักษะทางด้านนี้เพื่อนำไปต่อยอดกับผลงานวาดภาพ เพราะฉะนั้นการที่นักศึกษาสามารถฝึกวาดภาพคนเต็มตัว และสามารถประเมินการวาดภาพของนักศึกษาว่าอยู่ในกลุ่มใด ดังนั้นการวาดภาพคนเต็มตัวที่สามารถจำแนกกลุ่มภาพวาดได้นั้นสามารถทำให้นักศึกษาทราบถึงการประเมินจากผลงานตนเอง เพื่อนำไปพัฒนาทักษะในการวาดรูปเพิ่มเติมในครั้งต่อไป

ในการวิจัยนี้ผู้วิจัยได้นำข้อมูลจากการวาดภาพของนักศึกษามาใช้สร้างแบบจำลองด้วยเทคนิค ANN และเลือกคุณลักษณะด้วยวิธีมุมและระยะห่างโดยใช้เทคนิคโครงข่ายประสาทเทียมเชิงลึก เพื่อเป็นแนวทางในการพัฒนาการสร้างแบบจำลองการจำแนกกลุ่มภาพวาดคนเต็มตัวโดยวัดประสิทธิภาพของแบบจำลองตาม

จากการทดสอบการสกัดคุณลักษณะปรากฏว่าคุณลักษณะที่ใช้ในขั้นตอนวิธีการประมาณท่าทางโดยใช้ tf-pose-estimation มีประสิทธิภาพในการหาจุดสำคัญทั้ง 18 จุดกับภาพคนจริงได้ค่อนข้างครบทั้ง 18 จุดสำคัญแต่กลับกันถ้าภาพวาดเหมือนคนเต็มตัวจากการเรียนรู้ของเครื่องมีบางจุดที่คอมพิวเตอร์ไม่สามารถหาจุดสำคัญได้ ซึ่งแนวคิดในอนาคตชุดข้อมูลในการวาดภาพคนเต็มตัวควรมีเพิ่มเติมมากกว่านี้ เพื่อให้การเรียนรู้เชิงลึกของแบบจำลองมีประสิทธิภาพมากยิ่งขึ้น และควรหาแนวทางการศึกษาเพิ่มเติมกับรูปแบบของการหาจุดสำคัญจากภาพวาดให้ครบทั้ง 18 จุด เพื่อให้ผลออกมาเป็นที่น่าพอใจสำหรับการทดสอบในครั้งต่อไป

5.3 ข้อเสนอแนะ

54

1. การวาดภาพคนเต็มตัวหากมีการเก็บข้อมูลได้มากขึ้นจะเหมาะสมกับการสร้างแบบจำลองที่สร้างด้วย ANN

2. การสกัดคุณลักษณะของมุมและระยะห่างอาจต้องลองหาคุณลักษณะอื่นที่ช่วยในการพัฒนาการสร้างแบบจำลอง เช่น ความหนาแน่นบริเวณกล้ามเนื้อแต่ละส่วน

3. ท่าทางอิริยาบถของนายแบบ มีผลต่อการวาดภาพและการหาจุดสำคัญเพื่อใช้ในแบบจำลอง

4. การจำแนกของ ANN ยังไม่สามารถสรุปได้ว่าแบบจำลองที่ได้จาก ANN มีประสิทธิภาพเหมาะสมหรือไม่

บรรณานุกรม

[1] dshahid380. Convolutional neural network. 2019. Medium; [2019-02-26T16:54:11.097Z.

[2] Mellem J. How to draw people ebook by jeff mellem. 2018.

[3] Sigal L. Human pose estimation Ikeuchi K, editor. Computer vision: A reference guide. Boston, MA: Springer US; 2014 [2021-03-27 18:45:30]. 362-70.

[4] Cao Z, Simon T, Wei S-E, Sheikh Y. Realtime multi-person 2d pose estimation using part affinity fields. arXiv:161108050 [cs]. 2016.

[5] Hampton M. Figure drawing: Design and invention. 2nd edition. United States?: Michael Hampton; 2009.

[6] Hart C. Figure it out! Human proportions: Draw the head and figure right every time. Illustrated edition. New York: Drawing with Christopher Hart; 2014. 144.

[7] Haridas R. Convolutional neural networks: A comprehensive survey. 2019;14(3):10. en.

[8] Dang Q, Yin J, Wang B, Zheng W. Deep learning based 2d human pose estimation: A survey. Tsinghua Science and Technology. 2019;24(6):663-76.

[9] 'Ai guardman' - a machine learning application that uses pose estimation to detect shoplifters. Analytics Vidhya: 2018 [2018-06-27T06:22:59+00:00;2020-11-23 12:40:19].

[10] Ning G, Liu P, Fan X, Zhang C. A top-down approach to articulated human pose estimation and tracking. arXiv:190107680 [cs]. 2019.

[11] TensorFlow. Real-time human pose estimation in the browser with tensorflow.Js. 2018. Medium; [2018-09-27T22:19:17.079Z.

[12] Real-time human segmentation using pose skeleton map. 2019 Chinese Control Conference (CCC); 2019.

[13] Cao Z, Hidalgo G, Simon T, Wei S-E, Sheikh Y. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. arXiv:181208008 [cs]. 2019.

[14] Implementation of machine learning technique for identification of yoga poses. 2020 IEEE 9th International Conference on Communication Systems and Network Technologies (CSNT); 2020.

[15] Howard AG, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, et al. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv:170404861 [cs]. 2017.

[16] Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. Microsoft coco: Common objects in context. 2014. Cham: Springer International Publishing.

[17] Kc K, Yin Z, Wu M, Wu Z. Depthwise separable convolution architectures for plant disease classification. Computers and Electronics in Agriculture. 2019;165:104948. en.

[18] Bala R, Kumar DD. Classification using ann : A review. 2017 [2017.

[19] Cmu-perceptual-computing-lab/openpose. CMU-Perceptual-Computing-Lab; 2021.

[20] Exploring techniques to improve activity recognition using human pose skeletons. 2020 IEEE Winter Applications of Computer Vision Workshops (WACVW); 2020.

[21] Human posture classification using skeleton information. 2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET); 2018.

[22] Opencv. OpenCV; [

[23] Openpose's evaluation in the video traditional martial arts presentation. 2019 19th International Symposium on Communications and Information Technologies (ISCIT); 2019.

[24] Paulo. Programmingai: Calculating the bearing between two points. ProgrammingAI: 2012 [2012/03/13;2021-01-25 17:04:12].

[25] Paul R. Euclidean distance and normalization of a vector. 2020. Medium; [2020-12-23T02:38:04.360Z.

56

ภาคผนวก

55

รายละเอียดโค้ดที่ใช้ในการวิจัย(แบบจำลอง)

```
import argparse
import logging
import time
import cv2
import numpy as np
from tf_pose.estimator import TfPoseEstimator
from tf_pose.networks import get_graph_path, model_wh
import tkinter as tk
from PIL import ImageTk, Image
from tkinter import filedialog
from tqdm import tqdm
from os import path
from sklearn.model_selection import train_test_split
from skimage import feature
import pandas as pd
from model import NeuralNetwork
import os
from tkinter import messagebox
from sklearn import preprocessing

logger = logging.getLogger('TfPoseEstimator-WebCam')
logger.setLevel(logging.DEBUG)
ch = logging.StreamHandler()
ch.setLevel(logging.DEBUG)
formatter = logging.Formatter('[%(asctime)s] [%(name)s]
[%(levelname)s] %(message)s')
ch.setFormatter(formatter)
logger.addHandler(ch)
fps_time = 0
zero_count = 0
zero_count_prototype = 0

def str2bool(v):
    return v.lower() in ("yes", "true", "t", "1")

def Classify():

    mm_scaler = preprocessing.MinMaxScaler()
    norm_divide = 1000

### Drawing ###
    # Opendialog
    file = filedialog.askopenfilename(initialdir=os.getcwd(),
title="Select file", filetypes=(("jpeg files", "*.jpg"), ("all
files", "*.*")))
    image = cv2.imread(file, cv2.IMREAD_UNCHANGED)
    file_prototype = file
```

```
parser = argparse.ArgumentParser(description='tf-pose-estimation')
#parser.add_argument('--camera', type=int, default=1)

parser.add_argument('--resize', type=str, default='432x368',
                    help='if provided, resize images before they
are processed. default=0x0, Recommends : 432x368 or 656x368 or c
')
parser.add_argument('--resize-out-ratio', type=float, default=4.0,
                    help='if provided, resize heatmaps before they
are post-processed. default=1.0')

parser.add_argument('--model', type=str, default='mobilenet_thin',
                    help='cmu / mobilenet_thin /
mobilenet_v2_large / mobilenet_v2_small')
parser.add_argument('--show-process', type=bool, default=False,
                    help='for debug purpose, if enabled, speed for
inference is dropped.')

parser.add_argument('--tensorrt', type=str, default="False",
                    help='for tensorrt process.')
args = parser.parse_args()

logger.debug('initialization %s : %s' % (args.model,
get_graph_path(args.model)))
w, h = model_wh(args.resize)
if w > 0 and h > 0:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(w, h), trt_bool=str2bool(args.tensorrt))
else:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(432, 368), trt_bool=str2bool(args.tensorrt))

logger.info('cam image=%dx%d' % (image.shape[1], image.shape[1]))

logger.debug('image process+')
humans = e.inference(image, resize_to_default=(w > 0 and h > 0),
upsample_size=args.resize_out_ratio)

logger.debug('postprocess+')
[image, feature, key_point_length] =
TfPoseEstimator.draw_humans(image, humans, imgcopy=False)

if key_point_length >= 14:

    print(feature)

    X_test = np.zeros([1, len(feature)], dtype=float)
    for i in range(len(feature)):
        X_test[0, i] = feature[i] / norm_divide

    #X_test = mm_scaler.fit_transform(X_test)
    #X_test = X_test.T

    nn = NeuralNetwork()
    #nn.create(layer_sizes)
    nn.load_model('nn_model_drawing.xml')
    #result = nn.predict(X_test)
    ret, resp = nn.predict(X test)
```

```
print('ret', ret)
print('resp', resp)

Score = ''
if ret == 0:
    Score = 'poor'
elif ret == 1:
    Score = 'moderate'
elif ret == 2:
    Score = 'good'


img = cv2.resize(image, (800, 500),
interpolation=cv2.INTER_LINEAR)

cv2.putText(img, 'The result of ANN model prediction for score
range.', (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
cv2.putText(img, 'Class : ' + Score, (20, 60),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
resp = resp[1].tolist()

# Check accuracy
acc = resp[int(ret)] * 100
if acc > 100:
    acc = 100

cv2.putText(img, 'Confidence : ' + '{0:.2f} %'.format(acc), (20,
90), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)

# File name
file_name = os.path.basename(file)
index_of_dot = file_name.index('.')
file_name_without_extension = file_name[:index_of_dot]
cv2.putText(img, 'File name : ' + file_name_without_extension,
(20, 120), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)

# Show in tkinter label
dim = (500, 666)
img = cv2.resize(img, dim, interpolation=cv2.INTER_AREA)
prevImg = Image.fromarray(img)
imgtk = ImageTk.PhotoImage(image=prevImg)
panel.imgtk = imgtk
panel.configure(image=imgtk)

### Prototype ###

image = cv2.imread(file, cv2.IMREAD_UNCHANGED)
image_prototype = image

logger.debug('image process+')
humans = e.inference(image_prototype, resize_to_default=(w > 0 and
h > 0), upsample_size=args.resize_out_ratio)

logger.debug('postprocess+')
[image, feature, key_point_length] =
TfPoseEstimator.draw_humans(image_prototype, humans,
imgcopy=False)
```

58

```
if key_point_length >= 14:

    print(feature)

    X_test = np.zeros([1, len(feature)], dtype=float)
    for i in range(len(feature)):
        X_test[0, i] = feature[i] / norm_divide

    #X_test = mm_scaler.fit_transform(X_test)
    #X test = X test.T

    nn = NeuralNetwork()
    # nn.create(layer sizes)
    nn.load_model('nn_model_prototype.xml')
    # result = nn.predict(X_test)
    ret, resp = nn.predict(X_test)

    print("prototype predicted : {}".format(resp))

    # Show image
    files = os.listdir('Dataset_train_prototype/' +
str(int(ret)+1))
    img_files = list(filter(lambda x: '.jpg' in x, files))
    p_file = 'Dataset_train_prototype/' + str(int(ret)+1) + '/' +
img_files[1]
    image = cv2.imread(p_file, cv2.IMREAD_UNCHANGED)

    humans = e.inference(image, resize_to_default=(w > 0 and h >
0),
                         upsample_size=args.resize_out_ratio)

    [image, feature, key_point_length] =
TfPoseEstimator.draw_humans(image, humans, imgcopy=False)
    img = cv2.resize(image, (800, 500),
interpolation=cv2.INTER_LINEAR)

    Score = ''
    if ret == 0:
        Score = 'Figure 1'
    elif ret == 1:
        Score = 'Figure 2'
    elif ret == 2:
        Score = 'Figure 3'

    img = cv2.resize(image, (800, 500),
interpolation=cv2.INTER_LINEAR)

    cv2.putText(img, 'The result of ANN model prediction for model
figure.', (20, 30), cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
    cv2.putText(img, 'Class : ' + Score, (20, 60),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (255, 0, 0), 2)
    resp = resp[1].tolist()

    # Check accuracy
    acc = resp[int(ret)] * 100
    if acc > 100:
        acc = 100
```

```
            cv2.putText(img, 'Confidence : ' + '{0:.2f}
%'.format(acc), (20, 90), cv2.FONT_HERSHEY_SIMPLEX, 0.7,
                        (255, 0, 0), 2)

            # File name
            file_name = os.path.basename(p_file)
            index_of_dot = file_name.index('.')
            file_name_without_extension = file_name[:index_of_dot]
            cv2.putText(img, 'File name : ' +
file name without extension, (20, 120), cv2.FONT HERSHEY SIMPLEX,
0.7,
                        (255, 0, 0), 2)

            # Show in tkinter label
            dim = (500, 666)
            img = cv2.resize(img, dim,
interpolation=cv2.INTER_AREA)
            prevImg = Image.fromarray(img)
            imgtk = ImageTk.PhotoImage(image=prevImg)
            panel2.imgtk = imgtk
            panel2.configure(image=imgtk)


    else:
        messagebox.showerror("Error", "Can't find all key points.
Please try other image !")

def openpose(image, e, humans):


    logger.debug('postprocess+')
    [image, feature, key_point_length] =
TfPoseEstimator.draw_humans(image, humans, imgcopy=False)

    #cv2.imshow('tf-pose-estimation result', image)

    return feature, key_point_length

def Create_data_set():

    ### Drawing ###
    global zero_count, zero_count_prototype

    # Openpose
    parser = argparse.ArgumentParser(description='tf-pose-
estimation')
    #parser.add_argument('--camera', type=int, default=1)

    parser.add_argument('--resize', type=str, default='432x368',
                        help='if provided, resize images before
they are processed. default=0x0, Recommends : 432x368 or 656x368
or 1312x736 ')
    parser.add_argument('--resize-out-ratio', type=float,
default=4.0,
                        help='if provided, resize heatmaps before
they are post-processed. default=1.0')
```

```
parser.add_argument('--model', type=str, default='mobilenet_thin',
                    help='cmu / mobilenet_thin /
mobilenet_v2_large / mobilenet_v2_small')
parser.add_argument('--show-process', type=bool, default=False,
                    help='for debug purpose, if enabled, speed for
inference is dropped.')

parser.add_argument('--tensorrt', type=str, default="False",
                    help='for tensorrt process.')
args = parser.parse args()

logger.debug('initialization %s : %s' % (args.model,
get_graph_path(args.model)))
w, h = model_wh(args.resize)
if w > 0 and h > 0:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(w, h), trt_bool=str2bool(args.tensorrt))
else:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(432, 368), trt_bool=str2bool(args.tensorrt))

if path.exists("Feature_train.csv"):
    os.remove("Feature_train.csv")

# Feature extraction training
data_count_train = Files_count_train(1)
pbar = tqdm(total=data_count_train)
Feature_extraction_train = np.zeros([data_count_train, 27],
dtype=float);

# Extract feature all image
c = 0
for r, d, f in os.walk('Dataset_train'):
    for dir in d:
        for r2, d2, f2 in os.walk('Dataset_train/' + dir):
            for file in f2:
                if file.endswith(".jpg"):
                    full_path = os.path.join(r, dir, file)
                    image = cv2.imread(full_path)

                    # openpose
                    logger.debug('image process+')
                    humans = e.inference(image,
resize_to_default=(w > 0 and h > 0),
upsample_size=args.resize_out_ratio)

                    [F, key_point_length] = openpose(image, e,
humans)  # feature extraction

                    if key point length >= 14:
                        print('Have full key poinst :
{}'.format(full_path))
                        if F:
                            for i in range(len(F)):
                                Feature_extraction_train[c, i] =
F[i]
                            Feature_extraction_train[c, i + 1] =
dir  # Class
```

```
                            c += 1
                            pbar.update(1)
                        else:
                            zero_count += 1

# Write feature to csv
np.savetxt("Feature_train.csv", Feature_extraction_train,
fmt="%.5f", delimiter=",")

### Prototype ###

# Openpose
parser = argparse.ArgumentParser(description='tf-pose-estimation')
# parser.add_argument('--camera', type=int, default=1)

parser.add_argument('--resize', type=str, default='432x368',
                    help='if provided, resize images before they
are processed. default=0x0, Recommends : 432x368 or 656x368 or
1312x736 ')
parser.add_argument('--resize-out-ratio', type=float, default=4.0,
                    help='if provided, resize heatmaps before they
are post-processed. default=1.0')

parser.add_argument('--model', type=str, default='mobilenet_thin',
                    help='cmu / mobilenet_thin /
mobilenet_v2_large / mobilenet_v2_small')
parser.add_argument('--show-process', type=bool, default=False,
                    help='for debug purpose, if enabled, speed for
inference is dropped.')

parser.add_argument('--tensorrt', type=str, default="False",
                    help='for tensorrt process.')
args = parser.parse_args()

logger.debug('initialization %s : %s' % (args.model,
get_graph_path(args.model)))
w, h = model_wh(args.resize)
if w > 0 and h > 0:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(w, h), trt_bool=str2bool(args.tensorrt))
else:
    e = TfPoseEstimator(get_graph_path(args.model),
target_size=(432, 368), trt_bool=str2bool(args.tensorrt))

if path.exists("Feature_train_prototype.csv"):
    os.remove("Feature_train_prototype.csv")

# Feature extraction training
data_count_train_prototype = Files_count_train(2)
pbar = tqdm(total=data count train prototype)
Feature_extraction_train_prototype =
np.zeros([data_count_train_prototype, 27], dtype=float);
```

```
# Extract feature all image
c = 0
for r, d, f in os.walk('Dataset_train_prototype'):
    for dir in d:
        for r2, d2, f2 in os.walk('Dataset_train_prototype/' +
dir):
            for file in f2:
                if file.endswith(".jpg"):
                    full_path = os.path.join(r, dir, file)
                    image = cv2.imread(full path)

                    # openpose
                    logger.debug('image process+')
                    humans = e.inference(image,
resize_to_default=(w > 0 and h > 0),

upsample_size=args.resize_out_ratio)

                    [F, key_point_length] = openpose(image, e,
humans)  # feature extraction
                    #print("key_point_length :
{}".format(key_point_length))
                    if key_point_length >= 14:
                        if F:
                            for i in range(len(F)):

Feature_extraction_train_prototype[c, i] = F[i]
                            Feature_extraction_train_prototype[c,
i + 1] = dir  # Class
                            c += 1
                            pbar.update(1)
                        else:
                            zero_count_prototype += 1

# Write feature to csv
np.savetxt("Feature_train_prototype.csv",
Feature_extraction_train_prototype, fmt="%.5f", delimiter=",")


def ML_training():

    global zero_count, zero_count_prototype

    mm_scaler = preprocessing.MinMaxScaler()
    norm_divide = 1000
    #num_class = 14
```

```
### Drawing ###

# Training
df_train = pd.read_csv('Feature_train.csv', header=None)
Feature_train = df_train.values

[r, c] = Feature_train.shape

row = 0
for i in range(r):
    sum = 0
    for j in range(c):
        sum += Feature_train[i, j]
    if sum > 0:
        row += 1

X_train = np.zeros([row, 26], dtype=float);
y_train = np.zeros([row, 3], dtype=float);
for i in range(row):
    for j in range(c - 1):
        X_train[i, j] = Feature_train[i, j] / norm_divide

    target = np.zeros(shape=(1, 3), dtype=float)
    Class = Feature_train[i, 26]
    target[0, (int(Class) - 1)] = 1

    for k in range(3):
        y_train[i, k] = target[0, k]

#X_train = mm_scaler.fit_transform(X_train.T).T

# train test split
ANN_training(X_train, y_train, 'nn_model_drawing.xml', 'drawing')

### Prototype ###

# Training
df_train = pd.read_csv('Feature_train_prototype.csv', header=None)
Feature_train = df_train.values

[r, c] = Feature_train.shape

row = 0
for i in range(r):
    sum = 0
    for j in range(c):
        sum += Feature_train[i, j]
    if sum > 0:
        row += 1

X train = np.zeros([row, 26], dtype=float);
y_train = np.zeros([row, 3], dtype=float);
for i in range(row):
    for j in range(c - 1):
        X_train[i, j] = Feature_train[i, j] / norm_divide
```

```
        target = np.zeros(shape=(1, 3), dtype=float)
        Class = Feature_train[i, 26]
        target[0, (int(Class) - 1)] = 1

        for k in range(3):
            y_train[i, k] = target[0, k]

    #X_train = mm_scaler.fit_transform(X_train.T).T

    # train test split
    ANN_training(X_train, y_train, 'nn_model_prototype.xml',
'prototype')

def ANN_training(X_train, y_train, model_name, type):


    # Neural network training
    input_node = 26
    hidden_node = 38

    if type == 'prototype':
       output_node = 3
    else:
       output_node = 3

    epoch = 10000
    mse = 0.0001
    layer_sizes = [input_node, hidden_node, output_node]
    nn = NeuralNetwork(epoch, mse)
    nn.create(layer_sizes)
    #print(model_name)
    nn.train(X_train, y_train)

    # evaluate on train data
    train_accuracy = nn.evaluate(X_train, y_train)
    print("Train accuracy: ", "{0:.2f}%".format(train_accuracy *
100))

    # save model
    nn.save_model(model_name)

def Files_count_train(m):

    if m == 1:
        count = 0
        for r, d, f in os.walk('Dataset_train'):
            for dir in d:
                for r2, d2, f2 in os.walk('Dataset_train/' + dir):
                    for file in f2:
                        if file.endswith(".jpg"):
                            full_path = os.path.join(r, file)
                            count += 1
```

```
    else:
        count = 0
        for r, d, f in os.walk('Dataset_train_prototype'):
            for dir in d:
                for r2, d2, f2 in
os.walk('Dataset_train_prototype/' + dir):
                    for file in f2:
                        if file.endswith(".jpg"):
                            full_path = os.path.join(r, file)
                            count += 1

    return count

if __name__ == '__main__':

    # GUI
    global mainWindow, w, panel

    mainWindow = tk.Tk()
    mainWindow.title('Openpose estimation')
    mainWindow.resizable(width=False, height=False)

    window_height = 700
    window_width = 1200

    screen_width = mainWindow.winfo_screenwidth()
    screen_height = mainWindow.winfo_screenheight()

    x_cordinate = int((screen_width / 2) - (window_width / 2))
    y_cordinate = int((screen_height / 2) - (window_height / 2))

    mainWindow.geometry("{}x{}+{}+{}".format(window_width,
window_height, x_cordinate, y_cordinate))

    panel = tk.Label(mainWindow, compound=tk.CENTER,
anchor=tk.CENTER, bg="white")
    panel.place(x=10, y=10, bordermode="outside", height=666,
width=500)

    panel2 = tk.Label(mainWindow, compound=tk.CENTER,
anchor=tk.CENTER, bg="white")
    panel2.place(x=10 + 510, y=10, bordermode="outside",
height=666, width=500)

    button_webcam_start = tk.Button(mainWindow, text="Classify",
command=Classify, height=3, width=20)
    button_webcam_start.place(x=660 + 380, y=10, width=130,
height=40)

    button_webcam_stop = tk.Button(mainWindow, text="Create data
set", command=Create_data_set, height=3, width=20)
    button_webcam_stop.place(x=660 + 380, y=55, width=130,
height=40)

    button_webcam_stop = tk.Button(mainWindow, text="Training ML",
command=ML_training, height=3, width=20)
    button_webcam_stop.place(x=660 + 380, y=100, width=130,
height=40)
```

รายละเอียดโค้ดที่ใช้ในการวิจัย(มุมและระยะห่าง)

```
import logging
import math

import slidingwindow as sw

import cv2
import numpy as np
import tensorflow as tf
import time

from tf_pose import common
from tf_pose.common import CocoPart
from tf_pose.tensblur.smoother import Smoother
#import tensorflow.contrib.tensorrt as trt

try:
    from tf_pose.pafprocess import pafprocess
except ModuleNotFoundError as e:
    print(e)
    print('you need to build c++ library for pafprocess. See :
https://github.com/ildoonet/tf-pose-
estimation/tree/master/tf_pose/pafprocess')
    exit(-1)

logger = logging.getLogger('TfPoseEstimator')
logger.handlers.clear()
logger.setLevel(logging.INFO)
ch = logging.StreamHandler()
formatter = logging.Formatter('[%(asctime)s] [%(name)s]
[%(levelname)s] %(message)s')
ch.setFormatter(formatter)
logger.addHandler(ch)
logger.setLevel(logging.INFO)


def _round(v):
    return int(round(v))


def _include_part(part_list, part_idx):
    for part in part_list:
        if part_idx == part.part_idx:
            return True, part
    return False, None


class Human:
    """
    body_parts: list of BodyPart
    """
    __slots__ = ('body_parts', 'pairs', 'uidx_list', 'score')
```

```
def __init__(self, pairs):
    self.pairs = []
    self.uidx_list = set()
    self.body_parts = {}
    for pair in pairs:
        self.add_pair(pair)
    self.score = 0.0

@staticmethod
def  get uidx(part idx, idx):
    return '%d-%d' % (part_idx, idx)

def add_pair(self, pair):
    self.pairs.append(pair)
    self.body_parts[pair.part_idx1] =
BodyPart(Human._get_uidx(pair.part_idx1, pair.idx1),
                                                pair.part_idx1,
                                                pair.coord1[1],
pair.coord1[1], pair.score)
    self.body_parts[pair.part_idx2] =
BodyPart(Human._get_uidx(pair.part_idx2, pair.idx2),
                                                pair.part_idx2,
                                                pair.coord2[1],
pair.coord2[1], pair.score)
    self.uidx_list.add(Human._get_uidx(pair.part_idx1, pair.idx1))
    self.uidx_list.add(Human._get_uidx(pair.part_idx2, pair.idx2))

def is_connected(self, other):
    return len(self.uidx_list & other.uidx_list) > 0

def merge(self, other):
    for pair in other.pairs:
        self.add_pair(pair)

def part_count(self):
    return len(self.body_parts.keys())

def get_max_score(self):
    return max([x.score for _, x in self.body_parts.items()])

def get_face_box(self, img_w, img_h, mode=0):
    """
    Get Face box compared to img size (w, h)
    :param img_w:
    :param img_h:
    :param mode:
    :return:
    """
    # SEE : https://github.com/ildoonet/tf-pose-
estimation/blob/master/tf pose/common.py#L13
    _NOSE = CocoPart.Nose.value
    _NECK = CocoPart.Neck.value
    _REye = CocoPart.REye.value
    _LEye = CocoPart.LEye.value
    _REar = CocoPart.REar.value
    _LEar = CocoPart.LEar.value
```

```
_THRESHOLD_PART_CONFIDENCE = 0.2
parts = [part for idx, part in self.body_parts.items() if
part.score > _THRESHOLD_PART_CONFIDENCE]

is_nose, part_nose = _include_part(parts, _NOSE)
if not is_nose:
    return None

size = 0
is_neck, part_neck = _include_part(parts, _NECK)
if is_neck:
    size = max(size, img_h * (part_neck.y - part_nose.y) * 0.8)

is_reye, part_reye = _include_part(parts, _REye)
is_leye, part_leye = _include_part(parts, _LEye)
if is_reye and is_leye:
    size = max(size, img_w * (part_reye.x - part_leye.x) * 2.0)
    size = max(size,
               img_w * math.sqrt((part_reye.x - part_leye.x) ** 2
+ (part_reye.y - part_leye.y) ** 2) * 2.0)

if mode == 1:
    if not is_reye and not is_leye:
        return None

is_rear, part_rear = _include_part(parts, _REar)
is_lear, part_lear = _include_part(parts, _LEar)
if is_rear and is_lear:
    size = max(size, img_w * (part_rear.x - part_lear.x) * 1.6)

if size <= 0:
    return None

if not is_reye and is_leye:
    x = part_nose.x * img_w - (size // 3 * 2)
elif is_reye and not is_leye:
    x = part_nose.x * img_w - (size // 3)
else:  # is_reye and is_leye:
    x = part_nose.x * img_w - size // 2

x2 = x + size
if mode == 0:
    y = part_nose.y * img_h - size // 3
else:
    y = part_nose.y * img_h - _round(size / 2 * 1.2)
y2 = y + size

# fit into the image frame
x = max(0, x)
y = max(0, y)
x2 = min(img_w - x, x2 - x) + x
y2 = min(img_h - y, y2 - y) + y
```

```
    if _round(x2 - x) == 0.0 or _round(y2 - y) == 0.0:
        return None
    if mode == 0:
        return {"x": _round((x + x2) / 2),
                "y": _round((y + y2) / 2),
                "w": _round(x2 - x),
                "h": _round(y2 - y)}
    else:
        return {"x": _round(x),
                "y": _round(y),
                "w": _round(x2 - x),
                "h": _round(y2 - y)}

def get_upper_body_box(self, img_w, img_h):
    """
    Get Upper body box compared to img size (w, h)
    :param img_w:
    :param img_h:
    :return:
    """

    if not (img_w > 0 and img_h > 0):
        raise Exception("img size should be positive")

    _NOSE = CocoPart.Nose.value
    _NECK = CocoPart.Neck.value
    _RSHOULDER = CocoPart.RShoulder.value
    _LSHOULDER = CocoPart.LShoulder.value
    _THRESHOLD_PART_CONFIDENCE = 0.3
    parts = [part for idx, part in self.body_parts.items() if
part.score > _THRESHOLD_PART_CONFIDENCE]
    part_coords = [(img_w * part.x, img_h * part.y) for part in
parts if
                   part.part_idx in [0, 1, 2, 5, 8, 11, 14, 15,
16, 17]]

    if len(part_coords) < 5:
        return None

    # Initial Bounding Box
    x = min([part[0] for part in part_coords])
    y = min([part[1] for part in part_coords])
    x2 = max([part[0] for part in part_coords])
    y2 = max([part[1] for part in part_coords])

    # # ------ Adjust heuristically +
    # if face points are detcted, adjust y value

    is_nose, part_nose = _include_part(parts, _NOSE)
    is_neck, part_neck =  include_part(parts,  NECK)
    torso_height = 0
    if is_nose and is_neck:
        y -= (part_neck.y * img_h - y) * 0.8
        torso_height = max(0, (part_neck.y - part_nose.y) * img_h
* 2.5)
```

```
        #
        # # by using shoulder position, adjust width
        is_rshoulder, part_rshoulder = _include_part(parts,
_RSHOULDER)
        is_lshoulder, part_lshoulder = _include_part(parts,
_LSHOULDER)
        if is_rshoulder and is_lshoulder:
            half_w = x2 - x
            dx = half_w * 0.15
            x -= dx
            x2 += dx
        elif is_neck:
            if is_lshoulder and not is_rshoulder:
                half_w = abs(part_lshoulder.x - part_neck.x) *
img_w * 1.15
                x = min(part_neck.x * img_w - half_w, x)
                x2 = max(part_neck.x * img_w + half_w, x2)
            elif not is_lshoulder and is_rshoulder:
                half_w = abs(part_rshoulder.x - part_neck.x) *
img_w * 1.15
                x = min(part_neck.x * img_w - half_w, x)
                x2 = max(part_neck.x * img_w + half_w, x2)

        # ------ Adjust heuristically -

        # fit into the image frame
        x = max(0, x)
        y = max(0, y)
        x2 = min(img_w - x, x2 - x) + x
        y2 = min(img_h - y, y2 - y) + y

        if _round(x2 - x) == 0.0 or _round(y2 - y) == 0.0:
            return None
        return {"x": _round((x + x2) / 2),
                "y": _round((y + y2) / 2),
                "w": _round(x2 - x),
                "h": _round(y2 - y)}

    def __str__(self):
        return ' '.join([str(x) for x in
self.body_parts.values()])

    def __repr__(self):
        return self.__str__()


class BodyPart:
    """
    part_idx : part index(eg. 0 for nose)
    x, y: coordinate of body part
    score : confidence score
    """
```

```
    __slots__ = ('uidx', 'part_idx', 'x', 'y', 'score')

    def __init__(self, uidx, part_idx, x, y, score):
        self.uidx = uidx
        self.part_idx = part_idx
        self.x, self.y = x, y
        self.score = score

    def get_part_name(self):
        return CocoPart(self.part_idx)

    def __str__(self):
        return 'BodyPart:%d-(%.2f, %.2f) score=%.2f' %
(self.part_idx, self.x, self.y, self.score)

    def __repr__(self):
        return self.__str__()


class PoseEstimator:
    def __init__(self):
        pass

    @staticmethod
    def estimate_paf(peaks, heat_mat, paf_mat):
        pafprocess.process_paf(peaks, heat_mat, paf_mat)

        humans = []
        for human_id in range(pafprocess.get_num_humans()):
            human = Human([])
            is_added = False

            for part_idx in range(18):
                c_idx = int(pafprocess.get_part_cid(human_id,
part_idx))
                if c_idx < 0:
                    continue

                is_added = True
                human.body_parts[part_idx] = BodyPart(
                    '%d-%d' % (human_id, part_idx), part_idx,
                    float(pafprocess.get_part_x(c_idx)) /
heat_mat.shape[1],
                    float(pafprocess.get_part_y(c_idx)) /
heat_mat.shape[1],
                    pafprocess.get_part_score(c_idx)
                )

            if is_added:
                score = pafprocess.get_score(human_id)
                human.score = score
                humans.append(human)

        return humans
```

```
class TfPoseEstimator:
    # TODO : multi-scale

    def __init__(self, graph_path, target_size=(320, 240),
tf_config=None, trt_bool=False):
        self.target_size = target_size

        # load graph
        logger.info('loading graph from %s(default size=%dx%d)' %
(graph_path, target_size[1], target_size[1]))
        with tf.gfile.GFile(graph_path, 'rb') as f:
            graph_def = tf.GraphDef()
            graph_def.ParseFromString(f.read())

        if trt_bool is True:
            output_nodes = ["Openpose/concat_stage7"]
            graph_def = trt.create_inference_graph(
                graph_def,
                output_nodes,
                max_batch_size=1,
                max_workspace_size_bytes=1 << 20,
                precision_mode="FP16",
                # precision_mode="INT8",
                minimum_segment_size=3,
                is_dynamic_op=True,
                maximum_cached_engines=int(1e3),
                use_calibration=True,
            )

        self.graph = tf.get_default_graph()
        tf.import_graph_def(graph_def, name='TfPoseEstimator')
        self.persistent_sess = tf.Session(graph=self.graph,
config=tf_config)

        for ts in [n.name for n in
tf.get_default_graph().as_graph_def().node]:
            print(ts)

        self.tensor_image =
self.graph.get_tensor_by_name('TfPoseEstimator/image:0')
        self.tensor_output =
self.graph.get_tensor_by_name('TfPoseEstimator/Openpose/concat_sta
ge7:0')
        self.tensor_heatMat = self.tensor_output[:, :, :, :19]
        self.tensor_pafMat = self.tensor_output[:, :, :, 19:]
        self.upsample_size = tf.placeholder(dtype=tf.int32,
shape=(2,), name='upsample_size')
```

```
    self.tensor_heatMat_up =
tf.image.resize_area(self.tensor_output[:, :, :, :19],
self.upsample_size,

align_corners=False, name='upsample_heatmat')
    self.tensor_pafMat_up =
tf.image.resize_area(self.tensor_output[:, :, :, 19:],
self.upsample_size,

align_corners=False, name='upsample_pafmat')
    if trt_bool is True:
        smoother = Smoother({'data': self.tensor_heatMat_up}, 25,
3.0, 19)
    else:
        smoother = Smoother({'data': self.tensor_heatMat_up}, 25,
3.0)
    gaussian_heatMat = smoother.get_output()

    max_pooled_in_tensor = tf.nn.pool(gaussian_heatMat,
window_shape=(3, 3), pooling_type='MAX', padding='SAME')
    self.tensor_peaks = tf.where(tf.equal(gaussian_heatMat,
max_pooled_in_tensor), gaussian_heatMat,
                                 tf.zeros_like(gaussian_heatMat))

    self.heatMat = self.pafMat = None

    # warm-up
    self.persistent_sess.run(tf.variables_initializer(
        [v for v in tf.global_variables() if
         v.name.split(':')[0] in [x.decode('utf-8') for x in

self.persistent_sess.run(tf.report_uninitialized_variables())]
         ])
    )
    self.persistent_sess.run(
        [self.tensor_peaks, self.tensor_heatMat_up,
self.tensor_pafMat_up],
        feed_dict={
            self.tensor_image: [np.ndarray(shape=(target_size[1],
target_size[1], 3), dtype=np.float32)],
            self.upsample_size: [target_size[1], target_size[1]]
        }
    )
    self.persistent_sess.run(
        [self.tensor_peaks, self.tensor_heatMat_up,
self.tensor_pafMat_up],
        feed_dict={
            self.tensor_image: [np.ndarray(shape=(target_size[1],
target_size[1], 3), dtype=np.float32)],
            self.upsample_size: [target_size[1] // 2,
target_size[1] // 2]
        }
    )
```

74

```
        self.persistent_sess.run(
            [self.tensor_peaks, self.tensor_heatMat_up,
self.tensor_pafMat_up],
            feed_dict={
                self.tensor_image: [np.ndarray(shape=(target_size[1],
target_size[1], 3), dtype=np.float32)],
                self.upsample_size: [target_size[1] // 4,
target_size[1] // 4]
            }
        )

        # logs
        if self.tensor_image.dtype == tf.quint8:
            logger.info('quantization mode enabled.')

    def __del__(self):
        # self.persistent_sess.close()
        pass

    def get_flops(self):
        flops = tf.profiler.profile(self.graph,
options=tf.profiler.ProfileOptionBuilder.float_operation())
        return flops.total_float_ops

    @staticmethod
    def _quantize_img(npimg):
        npimg_q = npimg + 1.0
        npimg_q /= (2.0 / 2 ** 8)
        # npimg_q += 0.5
        npimg_q = npimg_q.astype(np.uint8)
        return npimg_q

    def angle_between(p1, p2):
        ang1 = np.arctan2(*p1[::-1])
        ang2 = np.arctan2(*p2[::-1])
        return np.rad2deg((ang1 - ang2) % (2 * np.pi))

    def angle3pt(a, b, c):
        """Counterclockwise angle in degrees by turning from a to c
around b
            Returns a float between 0.0 and 360.0"""
        ang = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))
        return ang + 360 if ang < 0 else ang

    @staticmethod
    def draw_humans(npimg, humans, imgcopy=False):

        feature = []
        position = np.zeros((18, 2), dtype=float)
        counter = 0

        if imgcopy:
            npimg = np.copy(npimg)
        image_h, image_w = npimg.shape[:2]
        centers = {}
        for human in humans:
```

```
# draw point
for i in range(common.CocoPart.Background.value):
    if i not in human.body_parts.keys():
        continue
    #print('human.body parts : {}'.format(len(human.body parts)))
    body_part = human.body_parts[i]
    center = (int(body_part.x * image_w + 0.5), int(body_part.y *
image_h + 0.5))
    centers[i] = center
    cv2.circle(npimg, center, 3, common.CocoColors[i],
thickness=3, lineType=8, shift=0)

    # Fall detection
    try:

        #if i <= 17:
        color = (0, 255, 0)
        cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
        position[counter, 0] = center[1]
        position[counter, 1] = center[1]
        counter += 1
        '''
        elif i == 1:
            color = (0, 0, 255)
            cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
            position[counter, 0] = center[1]
            position[counter, 1] = center[1]
            counter += 1
        elif i == 2:
            color = (255, 0, 0)
            cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
            position[counter, 0] = center[1]
            position[counter, 1] = center[1]
            counter += 1
        elif i == 5:
            color = (255, 0, 0)
            cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
            position[counter, 0] = center[1]
            position[counter, 1] = center[1]
            counter += 1
        elif i == 8:
            color = (255, 0, 0)
            cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
            position[counter, 0] = center[1]
            position[counter, 1] = center[1]
            counter += 1
        elif i == 11:
            color = (255, 0, 0)
            cv2.circle(npimg, center, 5, color, thickness=10,
lineType=8, shift=0)
            position[counter, 0] = center[1]
            position[counter, 1] = center[1]
            counter += 1
```

76

```
        #else:
            #cv2.circle(npimg, center, 3, common.CocoColors[i],
thickness=3, lineType=8, shift=0)
        '''

        body_part_1 = human.body_parts[1]
        center_1 = (int(body_part_1.x * image_w + 0.5),
int(body_part_1.y * image_h + 0.5))

        body_part_2 = human.body_parts[1]
        center_2 = (int(body_part_2.x * image_w + 0.5),
int(body_part_2.y * image_h + 0.5))

        changeInX = center_2[1] - center_1[1]
        changeInY = center_2[1] - center_1[1]
        Degree = math.degrees(math.atan2(changeInY, changeInX))
        org = (100, 100)
        font = cv2.FONT_HERSHEY_SIMPLEX
        fontScale = 2
        thickness = 2

        if Degree > 100 or Degree < 80:
            color = (0, 0, 255)
            #cv2.putText(npimg, 'Falling', org, font, fontScale,
color, thickness, cv2.LINE_AA)
        else:
            color = (0, 255, 0)
            #cv2.putText(npimg, 'Normal', org, font, fontScale,
color, thickness, cv2.LINE_AA)


    except Exception as e:
        logger.error('Failed to upload to ftp: ' + str(e))

# draw line
for pair_order, pair in enumerate(common.CocoPairsRender):
    if pair[1] not in human.body_parts.keys() or pair[1] not in
human.body_parts.keys():
        continue

    # npimg = cv2.line(npimg, centers[pair[0]], centers[pair[1]],
common.CocoColors[pair_order], 3)
    cv2.line(npimg, centers[pair[0]], centers[pair[1]],
common.CocoColors[pair_order], 3)
```

77

```
# Feature extraction radius

# Angle 01
a = [position[0, 0], position[0, 1]]
b = [position[1, 0], position[1, 1]]
c = [position[2, 0], position[2, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 02
a = [position[1, 0], position[1, 1]]
b = [position[2, 0], position[2, 1]]
c = [position[3, 0], position[3, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 03
a = [position[2, 0], position[2, 1]]
b = [position[3, 0], position[3, 1]]
c = [position[4, 0], position[4, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 04
a = [position[0, 0], position[0, 1]]
b = [position[1, 0], position[1, 1]]
c = [position[5, 0], position[5, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)
```

```
# Angle 05
a = [position[1, 0], position[1, 1]]
b = [position[5, 0], position[5, 1]]
c = [position[6, 0], position[6, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 06
a = [position[5, 0], position[5, 1]]
b = [position[6, 0], position[6, 1]]
c = [position[7, 0], position[7, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 07
a = [position[5, 0], position[5, 1]]
b = [position[11, 0], position[11, 1]]
c = [position[1, 0], position[1, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 08
a = [position[1, 0], position[1, 1]]
b = [position[11, 0], position[11, 1]]
c = [position[12, 0], position[12, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 09
a = [position[11, 0], position[11, 1]]
b = [position[12, 0], position[12, 1]]
c = [position[13, 0], position[13, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
```

```
if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 10
a = [position[11, 0], position[11, 1]]
b = [position[1, 0], position[1, 1]]
c = [position[8, 0], position[8, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 11
a = [position[2, 0], position[2, 1]]
b = [position[1, 0], position[1, 1]]
c = [position[8, 0], position[8, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 12
a = [position[1, 0], position[1, 1]]
b = [position[8, 0], position[8, 1]]
c = [position[9, 0], position[9, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)

# Angle 13
a = [position[8, 0], position[8, 1]]
b = [position[9, 0], position[9, 1]]
c = [position[10, 0], position[10, 1]]

angle = math.degrees(math.atan2(c[1] - b[1], c[1] - b[1]) -
math.atan2(a[1] - b[1], a[1] - b[1]))

if angle < 0:
    angle = 360 + angle

feature.append(angle)
```

```
# Feature extraction distance

# Distance 14
P1 = [position[0, 0], position[0, 1]]
P2 = [position[1, 0], position[1, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 15
P1 = [position[1, 0], position[1, 1]]
P2 = [position[2, 0], position[2, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 16
P1 = [position[2, 0], position[2, 1]]
P2 = [position[3, 0], position[3, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 17
P1 = [position[3, 0], position[3, 1]]
P2 = [position[4, 0], position[4, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 18
P1 = [position[1, 0], position[1, 1]]
P2 = [position[5, 0], position[5, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 19
P1 = [position[5, 0], position[5, 1]]
P2 = [position[6, 0], position[6, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 20
P1 = [position[6, 0], position[6, 1]]
P2 = [position[7, 0], position[7, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)
```

81

```
# Distance 21
P1 = [position[1, 0], position[1, 1]]
P2 = [position[11, 0], position[11, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 22
P1 = [position[11, 0], position[11, 1]]
P2 = [position[12, 0], position[12, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 23
P1 = [position[12, 0], position[12, 1]]
P2 = [position[13, 0], position[13, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 24
P1 = [position[1, 0], position[1, 1]]
P2 = [position[8, 0], position[8, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 25
P1 = [position[8, 0], position[8, 1]]
P2 = [position[9, 0], position[9, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)

# Distance 26
P1 = [position[9, 0], position[9, 1]]
P2 = [position[10, 0], position[10, 1]]

eDistance = math.sqrt(((P1[1] - P2[1]) ** 2) + ((P1[1] - P2[1]) **
2))
feature.append(eDistance)
```

```
    R = 0;
    if humans:
        R = len(human.body_parts)

    return npimg, feature, R

def _get_scaled_img(self, npimg, scale):
    get_base_scale = lambda s, w, h: max(self.target_size[1] /
float(h), self.target_size[1] / float(w)) * s
    img_h, img_w = npimg.shape[:2]

    if scale is None:
        if npimg.shape[:2] != (self.target_size[1],
self.target_size[1]):
            # resize
            npimg = cv2.resize(npimg, self.target_size,
interpolation=cv2.INTER_CUBIC)
        return [npimg], [(0.0, 0.0, 1.0, 1.0)]
    elif isinstance(scale, float):
        # scaling with center crop
        base_scale = get_base_scale(scale, img_w, img_h)
        npimg = cv2.resize(npimg, dsize=None, fx=base_scale,
fy=base_scale, interpolation=cv2.INTER_CUBIC)

        o_size_h, o_size_w = npimg.shape[:2]
        if npimg.shape[1] < self.target_size[1] or npimg.shape[1]
< self.target_size[1]:
            newimg = np.zeros(
                (max(self.target_size[1], npimg.shape[1]),
max(self.target_size[1], npimg.shape[1]), 3),
                dtype=np.uint8)
            newimg[:npimg.shape[0], :npimg.shape[1], :] = npimg
            npimg = newimg

        windows = sw.generate(npimg,
sw.DimOrder.HeightWidthChannel, self.target_size[1],
self.target_size[1], 0.2)

        rois = []
        ratios = []
        for window in windows:
            indices = window.indices()
            roi = npimg[indices]
            rois.append(roi)
            ratio_x, ratio_y = float(indices[1].start) / o_size_w,
float(indices[1].start) / o_size_h
            ratio_w, ratio_h = float(indices[1].stop -
indices[1].start) / o_size_w, float(
                indices[1].stop - indices[1].start) / o_size_h
            ratios.append((ratio_x, ratio_y, ratio_w, ratio_h))

return rois, ratios
```

83

```
elif isinstance(scale, tuple) and len(scale) == 2:
    # scaling with sliding window : (scale, step)
    base_scale = get_base_scale(scale[1], img_w, img_h)
    npimg = cv2.resize(npimg, dsize=None, fx=base_scale,
fy=base_scale, interpolation=cv2.INTER_CUBIC)
    o_size_h, o_size_w = npimg.shape[:2]
    if npimg.shape[1] < self.target_size[1] or npimg.shape[1] <
self.target_size[1]:
        newimg = np.zeros(
            (max(self.target size[1], npimg.shape[1]),
max(self.target_size[1], npimg.shape[1]), 3),
            dtype=np.uint8)
        newimg[:npimg.shape[0], :npimg.shape[1], :] = npimg
        npimg = newimg

    window_step = scale[1]

    windows = sw.generate(npimg, sw.DimOrder.HeightWidthChannel,
self.target_size[1], self.target_size[1],
                          window_step)

    rois = []
    ratios = []
    for window in windows:
        indices = window.indices()
        roi = npimg[indices]
        rois.append(roi)
        ratio_x, ratio_y = float(indices[1].start) / o_size_w,
float(indices[1].start) / o_size_h
        ratio_w, ratio_h = float(indices[1].stop -
indices[1].start) / o_size_w, float(
            indices[1].stop - indices[1].start) / o_size_h
        ratios.append((ratio_x, ratio_y, ratio_w, ratio_h))

    return rois, ratios
elif isinstance(scale, tuple) and len(scale) == 3:
    # scaling with ROI : (want_x, want_y, scale_ratio)
    base_scale = get_base_scale(scale[2], img_w, img_h)
    npimg = cv2.resize(npimg, dsize=None, fx=base_scale,
fy=base_scale, interpolation=cv2.INTER_CUBIC)
    ratio_w = self.target_size[1] / float(npimg.shape[1])
    ratio_h = self.target_size[1] / float(npimg.shape[1])

    want_x, want_y = scale[:2]
    ratio_x = want_x - ratio_w / 2.
    ratio_y = want_y - ratio_h / 2.
    ratio_x = max(ratio_x, 0.0)
    ratio_y = max(ratio_y, 0.0)
    if ratio_x + ratio_w > 1.0:
        ratio x = 1. - ratio w
    if ratio_y + ratio_h > 1.0:
        ratio_y = 1. - ratio_h

    roi = self._crop_roi(npimg, ratio_x, ratio_y)
    return [roi], [(ratio_x, ratio_y, ratio_w, ratio_h)]
```

84

```
def _crop_roi(self, npimg, ratio_x, ratio_y):
    target_w, target_h = self.target_size
    h, w = npimg.shape[:2]
    x = max(int(w * ratio_x - .5), 0)
    y = max(int(h * ratio_y - .5), 0)
    cropped = npimg[y:y + target_h, x:x + target_w]

    cropped_h, cropped_w = cropped.shape[:2]
    if cropped_w < target_w or cropped_h < target_h:
        npblank = np.zeros((self.target size[1],
self.target_size[1], 3), dtype=np.uint8)

        copy_x, copy_y = (target_w - cropped_w) // 2, (target_h -
cropped_h) // 2
        npblank[copy_y:copy_y + cropped_h, copy_x:copy_x +
cropped_w] = cropped
    else:
        return cropped

def inference(self, npimg, resize_to_default=True,
upsample_size=1.0):
    if npimg is None:
        raise Exception('The image is not valid. Please check your
image exists.')

    if resize_to_default:
        upsample_size = [int(self.target_size[1] / 8 *
upsample_size), int(self.target_size[1] / 8 * upsample_size)]
    else:
        upsample_size = [int(npimg.shape[0] / 8 * upsample_size),
int(npimg.shape[1] / 8 * upsample_size)]

    if self.tensor_image.dtype == tf.quint8:
        # quantize input image
        npimg = TfPoseEstimator._quantize_img(npimg)
        pass

    logger.debug('inference+ original shape=%dx%d' %
(npimg.shape[1], npimg.shape[1]))
    img = npimg
    if resize_to_default:
        img = self._get_scaled_img(npimg, None)[1]
    peaks, heatMat_up, pafMat_up = self.persistent_sess.run(
        [self.tensor_peaks, self.tensor_heatMat_up,
self.tensor_pafMat_up], feed_dict={
            self.tensor_image: [img], self.upsample_size:
upsample_size
        })
    peaks = peaks[1]
    self.heatMat = heatMat up[1]
    self.pafMat = pafMat_up[1]
    logger.debug('inference- heatMat=%dx%d pafMat=%dx%d' % (
        self.heatMat.shape[1], self.heatMat.shape[1],
self.pafMat.shape[1], self.pafMat.shape[1]))
```

```
        t = time.time()
        humans = PoseEstimator.estimate_paf(peaks, self.heatMat,
self.pafMat)
        logger.debug('estimate time=%.5f' % (time.time() - t))
        return humans


if __name__ == '__main__':
    import pickle

    f = open('./etcs/heatpaf1.pkl', 'rb')
    data = pickle.load(f)
    logger.info('size={}'.format(data['heatMat'].shape))
    f.close()

    t = time.time()
    humans = PoseEstimator.estimate_paf(data['peaks'],
data['heatMat'], data['pafMat'])
    dt = time.time() - t;
    t = time.time()
    logger.info('elapsed #humans=%d time=%.8f' % (len(humans),
dt))
```

ประวัติผู้เขียน

ชื่อ-สกุล	นายศรายุทธ ธิบดี
วัน เดือน ปี เกิด	19 พฤษภาคม 2532
สถานที่เกิด	ราชบุรี
วุฒิการศึกษา	พ.ศ. 2553 วิทยาศาสตรบัณฑิต สาขาเทคโนโลยีสารสนเทศเพื่อการออกแบบ จาก มหาวิทยาลัยศิลปากร พ.ศ. 2563 วิทยาศาสตรมหาบัณฑิต สาขาเทคโนโลยีสารสนเทศ จาก มหาวิทยาลัยศรีนครินทรวิโรฒ
ที่อยู่ปัจจุบัน	1 หมู่ 3 ต.สามพระยา อ.ชะอำ จ.เพชรบุรี 76120