ภาคผนวก

ก. การประชุมพิจารณาเครื่องมือการวิจัย

การประชุมร่วมระหว่างทีมนักวิจัย ประธานชุดวิชา 4 ชุดวิชา และผู้ทรงคุณวุฒิพิจารณาเครื่องมือการวิจัย 3 ท่าน โดยกำหนดการประชุมคณะวิจัยเพื่อพิจารณาเครื่องมือการวิจัยโดยผู้ทรงคุณวุฒิ เมื่อวันที่ 20 กันยายน 2565 เวลา 10.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams มีวัตถุประสงค์เพื่อพิจารณาเครื่องมือการวิจัยโดยผู้ทรงคุณวุฒิ โดยมีผู้เข้าร่วมประชุมดังรายนามต่อไปนี้

1. รองศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด หัวหน้าโครงการวิจัย
2. รองศาสตราจารย์ ดร.สุภาวดี ธีรธรรมากร ผู้ร่วมวิจัย
3. ผู้ช่วยศาสตราจารย์ ดร.พิมผกา ประเสริฐศิลป์ ผู้ร่วมวิจัย
4. ผู้ช่วยศาสตราจารย์ภิรมย์ คงเลิศ ผู้ร่วมวิจัย
5. นางสาวมาดีน่า อาโป ผู้ช่วยนักวิจัย
6. รองศาสตราจารย์ ดร.น้ำทิพย์ วิภาวิน ผู้ทรงคุณวุฒิ
7. ผู้ช่วยศาสตราจารย์ ดร.รัชฎา คงคะจันทร์ ผู้ทรงคุณวุฒิ
8. ดร.สรรพฤทธิ์ มฤคทัต ผู้ทรงคุณวุฒิ
9. ผู้ช่วยศาสตราจารย์ฐากร พฤกษวันประสุต ประธานชุดวิชา

โดยมีผู้ทรงคุณวุฒิพิจารณาเครื่องมือการวิจัย 3 ท่าน ได้แก่ รองศาสตราจารย์ ดร.น้ำทิพย์ วิภาวิน ผู้ช่วยศาสตราจารย์ ดร.รัชฎา คงคะจันทร์ และ ดร.สรรพฤทธิ์ มฤคทัต และประธานชุดวิชา 4 ชุดวิชา ได้แก่ ผู้ช่วยศาสตราจารย์ภิรมย์ คงเลิศ เป็นประธานชุดวิชา 96412 การบริหารโครงการด้านเทคโนโลยีสารสนเทศ ชุดวิชา 99419 ความมั่นคงปลอดภัยไซเบอร์ และ ชุดวิชา 99420 การโปรแกรมเว็บ และผู้ช่วยศาสตราจารย์ฐากร พฤกษวันประสุต ประธานชุดวิชา 96408 การจัดการระบบฐานข้อมูล

ผลการประชุมสรุปได้ดังนี้

การสนทนากลุ่ม เพื่อพิจารณาเครื่องมือการวิจัยโดยผู้ทรงคุณวุฒิ ครั้งที่ 3/2565 เมื่อวันที่ 20 กันยายน 2565 เวลา 10.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams โดยเป็นการนำเสนอรายละเอียดโครงการวิจัยเพื่อให้ที่ประชุมทราบ และพิจารณาเครื่องมือวิจัยที่ใช้ในโครงการวิจัยเรื่อง “การประเมินพฤติกรรมทุจริตระหว่างสอบออนไลน์ด้วยปัญญาประดิษฐ์บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ” เมื่อพิจารณาแล้ว คณะวิจัยจะต้องดำเนินการต่อไป ดังนี้

1. ศึกษาว่าพฤติกรรมใดบ้างที่เป็นลักษณะพฤติกรรมทุจริตในการสอบ
2. กำหนดว่า Feature ใดบ้างที่บ่งบอกว่าเป็นการทุจริต

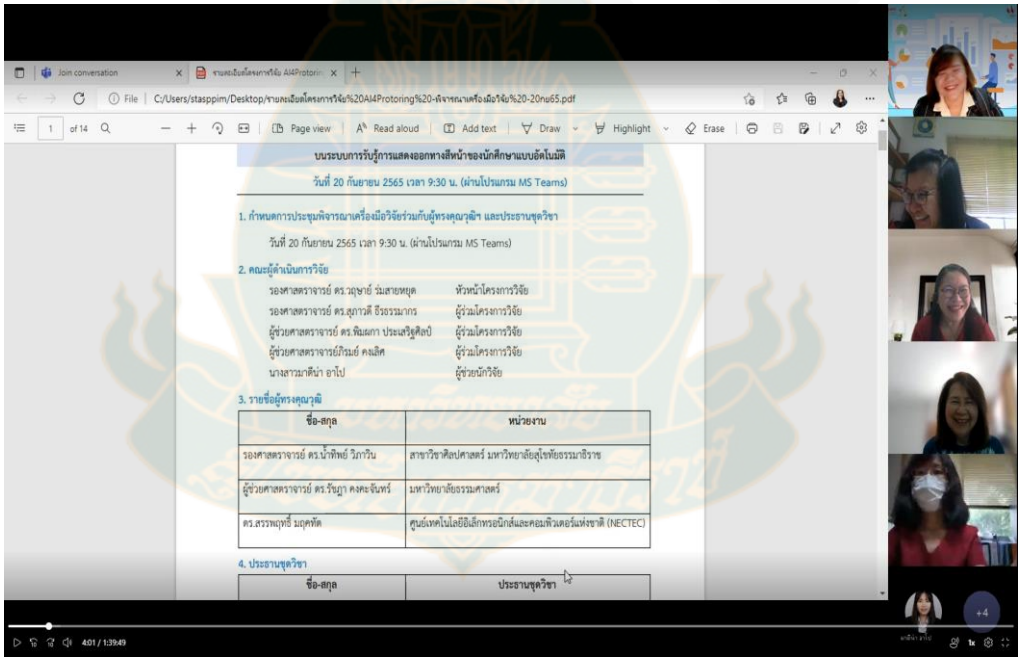
3. พิจารณาอัลกอริทึมที่ใช้ในการวิจัย ที่ได้เสนอไว้ คือ CNN และ MLP แต่ผู้ทรงคุณวุฒิเสนอแนะให้ใช้ Transformer แทน MLP เนื่องจากมีประสิทธิภาพดีกว่า

4. ศึกษากรณีทุจริตการสอบที่เคยเกิดขึ้นจริงจากสำนักทะเบียนฯ เพิ่มเติม เพื่อนำข้อมูลการทุจริตที่เคยเกิดขึ้นจริงมาวิเคราะห์และนำมาใช้ในการกำหนดขอบเขตการวิจัยใหม่อีกครั้ง เช่น ในการตรวจจับการทุจริตการสอบ อาจใช้วิธีการนำส่วนของวิดีโอมาตรวจสอบในภายหลัง (ไม่ใช่การตรวจจับแบบ real-time) และสามารถแจ้งเตือนผู้คุมสอบภายในช่วงเวลาที่ทำการสอบ 2-3 ชั่วโมงได้ เพื่อลดเวลาในการประมวลผลแบบ real-time

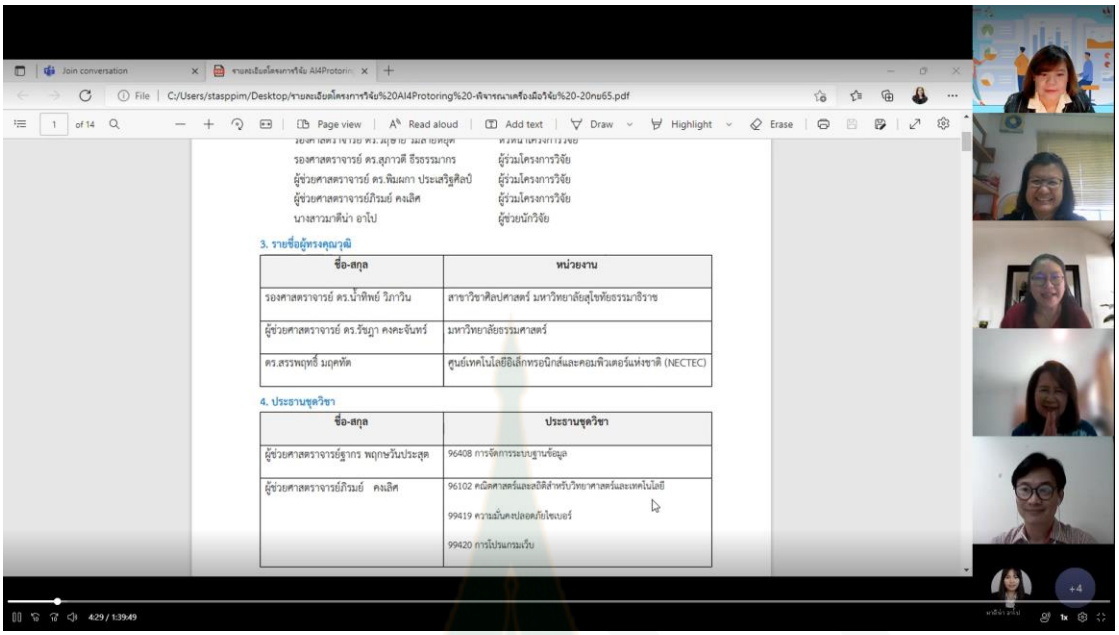
5. การใช้ emotion detection ในการกำหนดหรือประเมินการทุจริต แทนที่จะใช้ domain expert และใช้ emotion detection มาประเมิน/ตรวจจับการทุจริตในเบื้องต้นโดยพิจารณาลักษณะอารมณ์ในรูปแบบที่ส่อว่าเป็นลักษณะการทุจริต

6. การเสนอแพลตฟอร์ม/รูปแบบการจัดสอบในรูปแบบใหม่ที่มีการนำ AI มาประยุกต์ใช้เพื่อให้การสอบมีประสิทธิภาพ มีคุณภาพและมีความเชื่อมั่นได้มากขึ้น

7. ปรับชุดวิชาที่จะใช้ในการ train และการ test ให้เป็นชุดวิชาที่ใช้ลักษณะการท้าข้อสอบเป็นรูปแบบเดียวกัน



ก.1



ก.2

ภาพที่ ก.1-2 การประชุมคณะวิจัยครั้งที่ 3/2565 เมื่อวันที่ 20 กันยายน 2565 เวลา 10.30 น. ผ่านโปรแกรม Microsoft Teams

ข.การประชุมสนทนากลุ่ม ครั้งที่ 1

กำหนดการประชุมคณะวิจัยเพื่อสนทนากลุ่มครั้งที่ 1 ในประเด็นหลัก 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย เมื่อวันที่ 28 กันยายน 2565 เวลา 10.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams มีวัตถุประสงค์เพื่อสนทนากลุ่มในประเด็นหลัก 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย โดยมีผู้เข้าร่วมประชุมดังรายนามต่อไปนี้

1. รองศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด	หัวหน้าโครงการวิจัย
2. รองศาสตราจารย์ ดร.สุภาวดี ธีรธรรมากร	ผู้ร่วมวิจัย
3. ผู้ช่วยศาสตราจารย์ ดร.พิมผกา ประเสริฐศิลป์	ผู้ร่วมวิจัย
4. นางสาวมาดีน่า อาโป	ผู้ช่วยนักวิจัย
5. รองศาสตราจารย์ ดร.น้ำทิพย์ วิภาวิน	ผู้ทรงคุณวุฒิ
6. นายจักรี สวัสดิมงคล	ผู้ทรงคุณวุฒิ
7. ผู้ช่วยศาสตราจารย์ ดร.ชวาล คูร์พิพัฒน์	ผู้ทรงคุณวุฒิ
8. ผู้ช่วยศาสตราจารย์ ดร.ศรีสิทธิ์ เจียรบุตร	ผู้ทรงคุณวุฒิ
9. อาจารย์ ดร.ศรันย์ นาคณอม	ผู้ทรงคุณวุฒิ
10. อาจารย์ ดร.ธัญสินี เล่าส้ม	ผู้ทรงคุณวุฒิ
11. ผู้ช่วยศาสตราจารย์ฐากร พฤกษวันประสุต	ประธานชุดวิชา
12. นางรุจาภา รวยนิรันดร	เลขานุการประจำสาขาวิชาฯ
13. นางสมนึก สงวนตระกูล	เลขานุการประจำสาขาวิชาฯ
14. นายมนตรี ชะฎาแก้ว	เลขานุการประจำสาขาวิชาฯ
15. นายสมพล เพิงระนัย	เลขานุการประจำสาขาวิชาฯ
16. นางสาวนพวรรณ ชื่นอารมณ์	เลขานุการประจำสาขาวิชาฯ
17. นางสาวปัทมนันท์ อักษร	เลขานุการประจำสาขาวิชาฯ

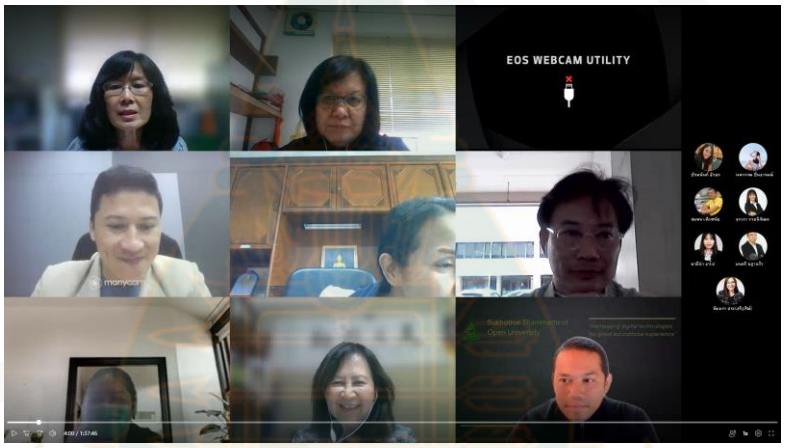
ผลการประชุมสรุปได้ดังนี้

การสนทนากลุ่มใน 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย ครั้งที่ 4/2565 เมื่อวันที่ 28 กันยายน 2565 เวลา 10.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams โดยเป็นการนำเสนอรายละเอียดโครงการวิจัยเพื่อให้ที่ประชุมทราบ และสนทนากลุ่มในประเด็นหลัก 3 ประเด็นเพื่อนำข้อมูลมาใช้ในโครงการวิจัยเรื่อง “การประเมินพฤติกรรมทุจริตระหว่างสอบออนไลน์ด้วยปัญญาประดิษฐ์บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ” เมื่อพิจารณาแล้ว คณะวิจัยจะต้องดำเนินการต่อ ดังนี้

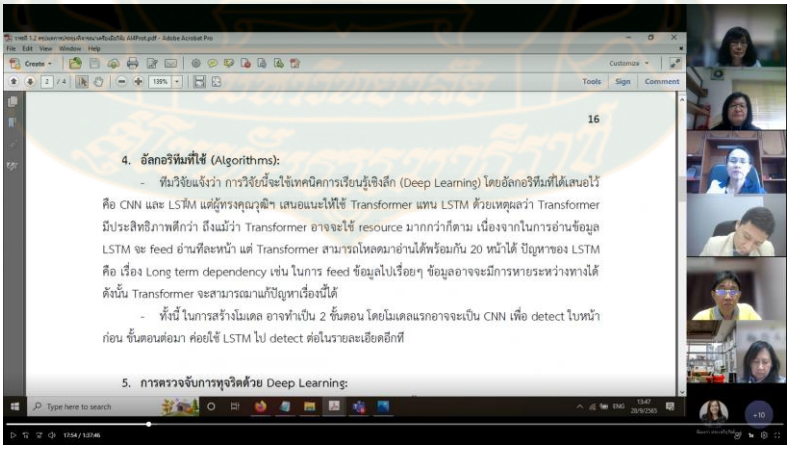
1) ในระหว่างการสอบออนไลน์ ลักษณะการแสดงออกบนใบหน้าของนักศึกษาแบบใด ที่แสดงพฤติกรรมที่นำไปสู่การทุจริตหรือไม่ทุจริต ได้แก่ ระดับของการส่อทุจริต การเหลือบซ้ายขวา องศาของหน้า จำนวนครั้งหรือความถี่ของลักษณะที่ไม่ปกติ กิจกรรมระหว่างสอบที่ไม่ทุจริตและทุจริต เช่น การหยิบแก้วน้ำดื่ม คนอื่นที่ไม่ใช่ผู้สอบ

2) ปัจจัยเสริมอะไร ที่สามารถใช้ในการจับใบหน้านักศึกษาระหว่างการสอบออนไลน์เพื่อการประเมินพฤติกรรมการทุจริตแบบอัตโนมัติได้ ได้แก่ การสร้างสภาพแวดล้อมในการสอบ จำนวนอุปกรณ์ที่ใช้ของนักศึกษาในการสอบ กล้อง (ตำแหน่งและจำนวน) เสียง จำนวนนักศึกษาในการคุมสอบแต่ละห้อง ประกาศการสอบที่ส่อทุจริตของมหาวิทยาลัย

3) อะไรคือปัญหาอุปสรรคที่ทำให้ไม่สามารถจับใบหน้านักศึกษาที่สอบออนไลน์เพื่อการวิเคราะห์การทุจริตแบบอัตโนมัติได้ ได้แก่ จำนวนอุปกรณ์ที่ใช้ของนักศึกษาในการสอบ การล็อกหน้าจอ สภาพแวดล้อมในการสอบ ความสามารถในการประมวลผล real time ดังภาพที่ ข.1-2



ข.1



ข.2

ภาพที่ ข.1-2 การประชุมคณะวิจัยครั้งที่ 4/2565 เมื่อวันที่ 28 กันยายน 2565 เวลา 10.30 น. ผ่านโปรแกรม Microsoft Teams

สรุปการสนทนากลุ่ม ครั้งที่ 1 วันที่ 28 กันยายน 2565 เวลา 13.30 – 15.30 น. โครงการวิจัย เรื่อง “การประเมินพฤติกรรมทุจริตระหว่างการสอบออนไลน์ด้วยปัญญาประดิษฐ์บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ”

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
1. ในระหว่างการสอบออนไลน์ ลักษณะการแสดงออกบนใบหน้าของนักศึกษาแบบใดที่แสดงพฤติกรรมที่นำไปสู่การทุจริตหรือไม่ทุจริต	**1.1.ลักษณะการแสดงออกบนใบหน้าที่ส่อทุจริต** 1.1.1 อาการสับส่ายของใบหน้า ลักษณะที่แสดงการไม่มีสมาธิ ไม่จดจ่ออยู่กับข้อสอบ หันไปมา การปิดบังใบหน้าหรือสายตา การขยี้ตาบ่อยๆ ถอด-ใส่แว่นตาบ่อยๆ เป็นต้น 1.1.2 ทิศทางการมองหรือ ล็อกแล็ก ของสายตาที่ไม่มองในตำแหน่งเดิม มีการกรอกตาไปมองรอบด้านอยู่เรื่อยๆ เปลี่ยนมุมโฟกัสบ่อยๆ ผ่านเว็บแคม โดยอาจจะมองไปทางด้านซ้าย ขวา บนล่าง จุดใดจุดหนึ่งนานๆเป็นต้น 1.1.3 การขยับปาก 1.1.4 ความถี่ของการละสายตาไปในจุดใดจุดหนึ่งมาหรือการชำเลืองสายตาไปมา เพื่อให้เกิดความมั่นใจว่าผู้สอบมีแนวโน้มที่จะทุจริตในการสอบการติดตามการจ้องมองสามารถทำได้ ใช้ตรวจจับการโกงข้อสอบออนไลน์ และระบุเจตนาของผู้สอบการติดตามดวงตาเนื่องจากผู้สอบจะแสร้งทำเป็นมองที่หน้าจอแต่พวกเขาจะขยับตาและเหลือบดูโน้ตของพวกเขา หรือโทรศัพท์มือถือ 1.1.5 องศาของใบหน้าในการนั่งทำข้อสอบ เช่น - กรณีใช้เครื่องคอมพิวเตอร์ Notebook องศาของใบหน้าจะกดต่ำลงเล็กน้อย - กรณีใช้เครื่องคอมพิวเตอร์ PC องศาของใบหน้าจะอยู่ในระดับตรงหรือแหงนขึ้นเล็กน้อย

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
	1.2 การแสดงออกบนใบหน้าอย่างเดียวอาจไม่สามารถบอกได้ว่าทุจริต 1.2.1 โปรแกรมที่ได้ช่วยการเตือนว่า ผู้สอบอาจทุจริตดูจากการแสดงออกทางใบหน้าอย่างเดียวได้หรือไม่ ซึ่งเป็นเรื่องยากที่จะตรวจสอบว่าผู้สอบโกงการสอบออนไลน์ด้วยวิธีการดูเพียงสายตา แต่อาจจะใช้เป็นส่วนประกอบกับพฤติกรรมอื่น ๆ เพื่อจับการทุจริต 1.2.2 ผู้วิจัยควรตรวจดูระเบียบ มสธ .ว่า การส่อทุจริตหรือทุจริต มีเกณฑ์อะไรบ้างใด เช่น ไม่อยู่ในห้องคนเดียว มีเสียงผู้อื่น มีการมองหนังสือ หากไม่มี หรือมีไม่ครอบคลุมควรให้ข้อเสนอแนะมสธที่เป็นประโยชน์ . 1.2.3 ควรมีลักษณะที่เป็นคะแนน และมีการบันทึกช่วงเวลาที่ส่อทุจริต เพื่อใช้ในการตรวจสอบภายหลังรวมทั้งควรมีการออกแบบโปรแกรมให้มีการแจ้งเตือนให้ผู้คุมทราบ
2. ปัจจัยเสริมอะไร ที่สามารถใช้ในการจับใบหน้านักศึกษาระหว่างการสอบออนไลน์เพื่อการประเมินพฤติกรรมการทุจริตแบบอัตโนมัติได้	**2.1 การสร้างสภาพแวดล้อมในการสอบเพื่อป้องกันการทุจริต** 2.1.1 กำหนดให้ผู้สอบเปิดกล้องเพื่อบันทึกการสอบไว้ช่วยให้สามารถนำมาใช้ประกอบการตรวจสอบการทุจริตในการสอบหลังสอบได้ 2.1.2 จัดที่นั่งสอบและมุมกล้องให้เห็นสภาพแวดล้อมในการสอบ หรือมีกล้องหลัง หรือกล้องที่สามารถเห็นได้รอบทิศทาง จะเป็นปัจจัยสนับสนุนทำให้สามารถรู้ได้ว่าการชำเลืองสายตาไปมาคือการส่อทุจริตหรือไม่เพื่อให้ผู้คุมสอบเห็นว่า ไม่มีสิ่งต้องห้ามในระยะใกล้เคียง

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
	2.1.3 เทคโนโลยีใหม่ๆ ในการวิเคราะห์อารมณ์ของผู้สอบ เช่น การหาอัตราการเต้นของหัวใจจากภาพ RGB 2.1.4 จำนวนนักศึกษาไม่มากเกินไป จะทำให้สามารถ Monitor ได้ในหน้าจอเดียว เนื่องจากปัจจุบัน 50 คน/1 ห้อง ทำให้การ Monitor จะต้องเลื่อน 2 หน้าจอ **2.2 ความยากที่จะควบคุมสภาพแวดล้อมให้เกิดอย่างเท่าเทียมกันที่บ้าน** เมื่อพิจารณาถึงความหลากหลายของสภาพแวดล้อม และการเข้าถึงทรัพยากรและอุปกรณ์ต่างๆ ของผู้สอบ เช่น Background ความสว่าง ความชัดเจนของกล้องสัญญาอินเตอร์เน็ตที่เสถียร ดังนั้น 2.2.1 ควรมีห้องสอบออนไลน์ที่ได้มาตรฐานหรือการวางกล้องและจำนวนกล้องที่เหมาะสม รวมถึงกำหนดองค์ประกอบของอุปกรณ์ที่ใช้ในการสอบออนไลน์ไว้ในระเบียบการสอบ 2.2.2 การกำหนดระเบียบการสอบและแนวปฏิบัติออนไลน์ ต้องสอดคล้องกัน 2.2.3 ช่วงเวลาขอการตรวจจับใบหน้าระหว่างการสอบ **2.3 การประมวลผลแบบ Real Time อาจใช้ Computing Power สูง** 2.3.1 ควรศึกษาความเป็นไปได้ว่า จะทำในลักษณะ Online หรือ Offline เพราะอาจมีค่าใช้จ่ายที่สูง 2.3.2 ควรทำการจำลองพฤติกรรมการทุจริตแบบเป็นระบบ เช่น กำหนดผู้ทุจริตแบบสุ่ม

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
	(กำหนดผู้ทุจริตล่วงหน้าแต่ระบบไม่ทราบว่าเป็นใคร) แล้วทำการตรวจสอบผลลัพธ์ความถูกต้องของระบบ รวมทั้งอาจพิจารณาเกณฑ์ทุจริตว่าตรงกับระเบียบของมหาวิทยาลัย
3. อะไรคือ ปัญหาอุปสรรคที่ทำให้ไม่สามารถจับ ใบหน้านักศึกษาที่สอบออนไลน์เพื่อการวิเคราะห์การทุจริตแบบอัตโนมัติได้	**3.1 ภาพการจับใบหน้า** ไม่สามารถบ่งบอกถึงการทุจริตได้ เป็นหลักฐานที่ไม่เพียงพอที่จะสรุปได้ว่าทุจริตน่าจะเป็นเพียงการเตือนว่า มีการส่อทุจริตเท่านั้น ดังนั้นการวิเคราะห์ภาพใบหน้าจึงไม่ใช่ปัญหาแต่จะเป็น ปัญหาอุปกรณ์และสภาพแวดล้อมไม่เอื้ออำนวยในการสอบ เช่น คุณภาพของกล้อง ภาพจากกล้องไม่ชัด อินเตอร์เน็ตช้า สภาพอากาศ ฝนตก ไฟฟ้าดับ เมื่อภาพจากกล้องไม่ชัดเจน ไม่เห็นใบหน้าในมุมที่ต้องการสภาพแสงมืดจนเกินไป เป็นต้น **3.2 จำนวนกรรมการคุมสอบน้อย**ทำให้จำนวนนักศึกษา แต่ละห้องสอบออนไลน์มาก จำนวนของใบหน้าที่ต้องตรวจจับต่อหนึ่งหน้าจออาจจะมากเกินกำลังความสามารถของการประมวลผล ดังนั้น อาจใช้ภาพจากหน้าจอของผู้คุมสอบเป็น input ของระบบตรวจจับใบหน้า (ตรวจจับเฉพาะหน้าคนที่อยู่บน Microsoft Teams หรือบน WebEX ที่ปรากฏบนหน้าจอเครื่องคอมฯ ของคนคุมสอบคนนั้นๆเท่านั้น) **3.3 Object ที่อยู่ใน frame** อาจทำให้การตรวจจับใบหน้าทำได้ยาก อาจต้องแจ้งนักศึกษาที่สอบออนไลน์ ให้ใช้ Background เป็นสีพื้นเพื่อให้การตรวจจับใบหน้ามีความแม่นยำมากขึ้น

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
	3.4 อื่นๆ 3.4.1 อาจไม่ใช่ Platform การสอบออนไลน์ที่เหมาะสม 3 4.2 การับรู้และความเข้าใจของผู้สอบในด้านจริยธรรม 3.4.3 การทำหน้าที่ของกรรมการคุมสอบหรือผู้ตรวจรวมทั้งความพร้อมของระบบ 3.4.4 ควรศึกษาความคุ้มค่าในเชิงเศรษฐศาสตร์และความเป็นไปได้ในทางปฏิบัติ รวมทั้งอาจให้ข้อเสนอแนะการแก้ไขที่เป็นประโยชน์ เช่น รูปแบบการสอนออนไลน์ ที่ไม่ต้องค านึงถึงการตรวจจับทุจริตซึ่งเป็นการแก้ไขที่สาเหตุ หรือแก้ไขที่รูปแบบข้อสอบ 3.4.5 การจับใบหน้าอาจมีความคลาดเคลื่อนหรือไม่หากผู้เข้าสอบมีพฤติกรรมส่อทุจริตโดยไม่ได้ทำการทุจริต จะทราบและป้องกันได้อย่างไร เช่น ผู้สอบมองหน้าต่างเพื่อเปลี่ยนอริยาบทหรือพักสายตา 3.4.6 สภาพแวดล้อมที่ไม่พร้อมอาจจะเป็นปัจจัยที่ทำให้การตรวจจับใบหน้าเกิดข้อผิดพลาด

ค.การประชุมประเมินการพิจารณาเครื่องมือการวิจัย

แบบประเมินความสอดคล้อง (IOC) ของแบบประเมินประสิทธิภาพของระบบ
“การประเมินพฤติกรรมทุจริตระหว่างการสอบออนไลน์ด้วยปัญญาประดิษฐ์
บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ”

วัตถุประสงค์การวิจัย

เพื่อประเมินผลประสิทธิภาพของระบบการประเมินพฤติกรรมทุจริตระหว่างการสอบออนไลน์ด้วยปัญญาประดิษฐ์

คำชี้แจง โปรดพิจารณาว่าแบบประเมินมีความเหมาะสมและสอดคล้องกับวัตถุประสงค์การวิจัยหรือไม่

โดยทำเครื่องหมาย ✓ ลงในช่องระดับความคิดเห็นของท่าน ตามเกณฑ์ที่กำหนดต่อไปนี้

+1 หมายถึง แน่ใจว่ารายการประเมินสอดคล้องกับวัตถุประสงค์การวิจัย

0 หมายถึง ไม่แน่ใจว่ารายการประเมินสอดคล้องกับวัตถุประสงค์การวิจัย

-1 หมายถึง แน่ใจว่ารายการประเมินไม่สอดคล้องกับวัตถุประสงค์การวิจัย

หากมีประเด็นข้อเสนอแนะหรือความคิดเห็นเพิ่มเติมในแต่ละรายการ โปรดเขียนข้อเสนอแนะไว้ท้ายรายการนั้น หรือหากมีข้อเสนอแนะโดยภาพรวม โปรดเขียนข้อเสนอแนะในตอนท้ายของแบบประเมิน

คณะผู้วิจัยขอขอบพระคุณในความกรุณาของท่านมา ณ โอกาสนี้

(รองศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด)
หัวหน้าคณะทำงานวิจัย

สรุปคะแนนความคิดเห็นจากผู้เชี่ยวชาญ (ผู้ทรงคุณวุฒิ) ทั้ง 3 ท่าน ดังรายละเอียดต่อไปนี้

รายการประเมิน	คะแนนความคิดเห็นผู้เชี่ยวชาญ			คะแนนเฉลี่ย
	ท่านที่ 1	ท่านที่ 2	ท่านที่ 3	
1. โปรแกรมที่พัฒนาสามารถรองรับการประเมินพฤติกรรมส่อทุจริตระหว่างการสอบออนไลน์ของนักศึกษาได้โดยอัตโนมัติ	1	1	0	0.67
2. ฟังก์ชันการแสดงผลการแจ้งเตือนพฤติกรรมส่อทุจริต ทำให้เข้าใจง่าย เช่น กรอบสีเขียว-ปกติ และกรอบสีแดง-ส่อทุจริต เป็นต้น	1	1	-1	0.33
3. การรายงานจำนวนครั้งของการแสดงออกทางสีหน้าที่ส่อทุจริต (count) มีความเหมาะสม	1	1	1	1.00
4. การติดตั้งโปรแกรมและการตรวจสอบสภาพแวดล้อม ของระบบ (system environment) เพื่อให้มีความพร้อมในการใช้งานโปรแกรม มีความชัดเจน	1	1	0	0.67
5. ฟังก์ชันการจัดการรายชื่อนักศึกษา ซึ่งรองรับทั้งรูปแบบการพิมพ์รายชื่อเองและรูปแบบการอัปโหลดไฟล์ชื่อนักศึกษา มีความเหมาะสม	1	1	1	1.00
6. การตั้งค่าการจัดวางตำแหน่งหน้าจอของห้องสอบเพื่อระบุขอบเขตจอภาพของนักศึกษาแต่ละคน เพื่อตรวจจับพฤติกรรมส่อทุจริตของนักศึกษา มีความเหมาะสม	1	1	1	1.00
7. จำนวนหน้าจอนักศึกษาในห้องสอบ ในการแสดงผลพร้อมกันมากที่สุด 25 หน้าจอ ที่สามารถตรวจจับพฤติกรรมส่อทุจริตได้ มีความเหมาะสม	1	1	1	1.00
8. การเลือกตั้งค่าร้อยละการจัดเก็บข้อมูลและการตรวจสอบการส่อทุจริต มีความเหมาะสม	1	1	0	0.67

9. รายงานการแสดงสถานะ (status) และความน่าจะเป็นในการส่อทุจริต (cheat probability) มีความเข้าใจง่าย	1	1	0	0.67
10. โดยภาพรวม โปรแกรมการประเมินพฤติกรรมทุจริตระหว่างการสอบออนไลน์ด้วยปัญญาประดิษฐ์ บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ มีประสิทธิภาพ	1	1	-1	0.33

สรุปข้อเสนอแนะจากผู้เชี่ยวชาญ (ผู้ทรงคุณวุฒิ) ทั้ง 3 ท่าน ดังรายละเอียดต่อไปนี้

- **รายการประเมินที่ 1** ควรเน้นความครอบคลุมของพฤติกรรมทุจริตที่โปรแกรมตรวจสอบได้ว่ามีอะไรบ้าง เช่น การกรอกสายตาซ้ายขวา
- **รายการประเมินที่ 2** ควรแยกประเมินทีละฟังก์ชัน
- **รายการประเมินที่ 3** ควรปรับคำว่า “การกระทำ” เป็น “การแสดงออกทางสีหน้า”
- **รายการประเมินที่ 4** ประเด็นที่สอบถามหมายถึง “คู่มือแนะนำการใช้งาน” หรือไม่ ควรสอบถามเรื่องความรวดเร็วในการติดตั้งโปรแกรม หรือใส่ภาพหน้าจอโปรแกรมเพิ่มเติมประกอบคำถาม
- **รายการประเมินที่ 5** ควรใช้ประเด็นเดียวว่า เหมาะสม หรือใช้งานง่าย
- **รายการประเมินที่ 6** ควรปรับคำว่า “การตรวจสอบพฤติกรรมทุจริตของนักศึกษาแต่ละคน” เป็นคำว่า “ของห้องสอบเพื่อระบุขอบเขตจอภาพของนักศึกษาแต่ละคนเพื่อตรวจจับพฤติกรรมทุจริตของนักศึกษา”
- **รายการประเมินที่ 7** ควรปรับเป็น ทำได้ครบ 25 จอภาพ
- **รายการประเมินที่ 8** ควรขยายความว่าจัดเก็บข้อมูลในเรื่องใด และควรแยกถามคนละประเด็นกับการตรวจสอบ
- **รายการประเมินที่ 9** ควรแยกประเด็นถามเรื่องรายงานการแสดงสถานะ (status) กับเรื่องความน่าจะเป็นในการส่อทุจริต (cheat probability) และควรเพิ่มการรายงานหมวดหมู่ที่เป็นประเภทของการส่อทุจริต
- **รายการประเมินที่ 10** ข้อ 1-9 เป็นการประเมินแล้ว ไม่จำเป็นต้องถามในภาพรวม
- **ข้อเสนอแนะอื่นๆ**
    1. ควรเพิ่มรายการการศึกษาคู่มือ วีดิทัศน์การใช้งาน เข้าใจง่าย
    2. ควรปรับคำว่า “พฤติกรรมทุจริต” เป็น “พฤติกรรมส่อทุจริต”

3. ควรลำดับข้อคำถาม ที่ถามในประเด็นใกล้เคียงกัน ควรอยู่ลำดับใกล้กัน เช่น ประเมินฟังก์ชัน ประเมิน report ควรลำดับต่อกันเป็นกลุ่มๆ
4. ควรให้โปรแกรมมีการจับเส้นกรอบของหน้าจอนักศึกษาแต่ละคนอัตโนมัติ (การวาง layout)

ง.การประชุมสนทนากลุ่ม ครั้งที่ 2

การประชุมสนทนากลุ่มครั้งที่ 2 ระหว่างทีมนักวิจัย ผู้ทรงคุณวุฒิ 3 ท่าน ประธานชุดวิชา 4 ชุดวิชา เจ้าหน้าที่ทีมเลขานุการกิจสาขาวิชาวิทยาศาสตร์และเทคโนโลยี ผู้ทรงคุณวุฒิจากสำนักคอมพิวเตอร์ สำนักทะเบียนและวัดผล สำนักเทคโนโลยีการศึกษา และผู้บริหารที่ดูแลนโยบายการสอบออนไลน์ กำหนดการประชุมคณะวิจัยเพื่อสนทนากลุ่มในประเด็นหลัก 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย เมื่อวันที่ 21 มิถุนายน 2566 เวลา 13.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams มีวัตถุประสงค์เพื่อสนทนากลุ่มในประเด็นหลัก 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย โดยมีผู้เข้าร่วมประชุมดังรายนามต่อไปนี้

18. รองศาสตราจารย์ ดร.วฤษาย์ ร่มสายหยุด	หัวหน้าโครงการวิจัย
19. รองศาสตราจารย์ ดร.สุภาวดี ธีรธรรมากร	ผู้ร่วมวิจัย
20. ผู้ช่วยศาสตราจารย์ ดร.พิมผกา ประเสริฐศิลป์	ผู้ร่วมวิจัย
21. นางสาวมาดีน่า อาโป	ผู้ช่วยนักวิจัย
22. รองศาสตราจารย์ ดร.น้ำทิพย์ วิภาวิน	ผู้ทรงคุณวุฒิ
23. นายจักรี สวัสดิมงคล	ผู้ทรงคุณวุฒิ
24. ผู้ช่วยศาสตราจารย์ ดร.ชวาล คูร์พิพัฒน์	ผู้ทรงคุณวุฒิ
25. ผู้ช่วยศาสตราจารย์ ดร.ศรีสิทธิ์ เจียรบุตร	ผู้ทรงคุณวุฒิ
26. อาจารย์ ดร.ศรันย์ นาคถนอม	ผู้ทรงคุณวุฒิ
27. อาจารย์ ดร.ธัญสินี เล่าส้ม	ผู้ทรงคุณวุฒิ
28. ผู้ช่วยศาสตราจารย์ฐากร พฤกษวันประสุต	ประธานชุดวิชา
29. นางรุจาภา รวยนิรันดร	เลขานุการประจำสาขาวิชาฯ
30. นางสมนึก สงวนตระกูล	เลขานุการประจำสาขาวิชาฯ
31. นายมนตรี ชะฎาแก้ว	เลขานุการประจำสาขาวิชาฯ
32. นายสมพล เพิงระนัย	เลขานุการประจำสาขาวิชาฯ
33. นางสาวนพวรรณ ชื่นอารมณ์	เลขานุการประจำสาขาวิชาฯ
34. นางสาวปัทมนันท์ อักษร	เลขานุการประจำสาขาวิชาฯ

ภาพที่ ง.1 กิจกรรม Focus group ครั้งที่ 2 ระหว่างทีมนักวิจัย ผู้ทรงคุณวุฒิ 3 ท่าน ประธานชุดวิชา 4 ชุดวิชา เจ้าหน้าที่ทีมเลขานุการกิจสาขาวิชาวิทยาศาสตร์และเทคโนโลยี ผู้ทรงคุณวุฒิจากสำนักคอมพิวเตอร์ สำนักทะเบียนและวัดผล สำนักเทคโนโลยีการศึกษา และผู้บริหารที่ดูแลนโยบายการสอบออนไลน์ กำหนดการประชุมคณะวิจัยเพื่อสนทนากลุ่มในประเด็นหลัก 3 ประเด็นและนำข้อมูลมาใช้ในโครงการวิจัย เมื่อวันที่ 21 มิถุนายน 2566 เวลา 13.30 น. โดยการประชุมออนไลน์ผ่านโปรแกรม Microsoft Teams

ผลการประชุมสรุปได้ดังนี้

ประเด็น	ข้อคิดเห็น/ข้อเสนอแนะ
1. จากโปรแกรมที่พัฒนาขึ้นมานี้ ท่านมีความคิดเห็นอย่างไรบ้าง	1.1 โปรแกรมที่แสดงดี มีการนำมาใช้ทดสอบทดลองจริงช่วยเตือนให้ผู้คุมสอบดูได้ง่ายขึ้น จัดเป็นผลิตภัณฑ์ที่มีสมบัติสำคัญน้อยที่สุดที่ดีหรือ MVP (Minimum Viable Product) ถือว่าผ่าน (สำหรับต้นแบบ) 1.2 การใช้งาน ควรมีปุ่ม Auto Calibration เพื่อให้สามารถปรับตั้งง่ายสำหรับผู้คุมสอบ

	(User Friendly) เพื่อลดปัญหาการปรับตั้ง (Set Up) ก่อนสอบ 1.3 ระบบ Notification สามารถระบุทุกหน้าและแสดง No/Percent ที่ Overlay หน้าจอพร้อมกันได้จะดี
2. ปัญหาและอุปสรรคที่น่ากังวลเมื่อนำโปรแกรมมาใช้จริง	2.1 ทรัพยากรที่ใช้ - การสอบแต่ละครั้งใช้เวลา 3 ชั่วโมง ต้องมีระบบรองรับการเก็บข้อมูลที่เพียงพอ - ความละเอียดของกล้องที่นักศึกษาใช้เพื่อให้สามารถจับภาพใบหน้าได้ชัดเจน 2.2 นักศึกษา - ควรมีการเตือนให้นักศึกษาได้ทราบด้วย เนื่องจาก นักศึกษาจะไม่เห็นการเตือนเป็นสีแดงเหมือนที่ผู้คุมสอบเห็น 2.3 โปรแกรม - จะสามารถเชื่อมโยงเพื่อเตือนให้นักศึกษาได้เห็นได้หรือไม่ ว่ามีพฤติกรรมการส่อทุจริตเพื่อการเตือน - การกำหนดกรอบแต่ละคนบนหน้าจอเป็นไปได้หรือไม่ที่จะสร้างกรอบเดียวและโปรแกรม generate เองตามจำนวนผู้เข้าสอบที่ปรากฏบนหน้าจอ - การไม่สมดุลของข้อมูลที่ใช้เพื่อจับใบหน้าอาจมีการส่งทุจริตที่น้อยกว่าความเป็นจริง

	อาจส่งผลต่อความถูกต้องของโปรแกรมในการใช้งานจริง - การรองรับภาพใบหน้าของนักศึกษาที่สลับไปมาใน Webex อาจไม่รองรับกับโปรแกรมที่ออกแบบมา 2.4 กฎระเบียบ - กฎระเบียบของ มสธ ที่มีการปรับเปลี่ยนเพื่อป้องกันการทุจริต เช่น การใช้กล้อง สองตัว อาจไม่รองรับการใช้โปรแกรม
3. ข้อเสนอแนะสำหรับการต่อยอดงานวิจัย	3.1 เชิงพาณิชย์ - ต่อยอดพัฒนาการตรวจจับพฤติกรรมผู้เข้าสอบ ทำเป็นโปรแกรมสำเร็จขายได้ หรือจัดทำเป็นศูนย์สอบด้วยโปรแกรม เช่น สอบตำรวจ 3.2 โปรแกรม - พัฒนาระบบเพื่อที่จะไม่ให้เกิดการทุจริต เนื่องจากสิ่งที่ต้องการ คือ ไม่ต้องการให้นักศึกษาถูกคาดโทษจะดีที่สุด (พยากรณ์เพื่อการยับยั้งการโกง) - การศึกษาเปรียบเทียบการทำงานระหว่างคนคุมกับ Machine คุม มีผลเป็นอย่างไร และการออกแบบ Supervised Learning ควรมีระบบ Feed back จากคน เพื่อให้ M/L มีการเรียนรู้ที่ฉลาดมากขึ้นเรื่อยๆ เมื่อข้อมูลมีการเพิ่มขึ้น พร้อมระบบควรออกแบบให้สามารถรองรับปรับปรุงพฤติกรรม

	- เก็บข้อมูลพฤติกรรมการส่อทุจริตเพื่อการปรับความถูกต้องแม่นยำของโปรแกรมให้มีประสิทธิภาพมากขึ้น 3.3 พฤติกรรมการส่อทุจริต - การเก็บข้อมูลการวิเคราะห์พฤติกรรมการส่อทุจริตเป็นระดับ เนื่องจากนักศึกษาจะมีการปรับตัวทุจริตมากขึ้น เช่น หน้าตรง เป็นระดับปกติ ระดับส่อทุจริตเป็นระดับ 1-10 โดยแบ่งเป็นช่วงของ อาทิ เตือน (1-3) เข้าข่ายทุจริต (4-6) หรือรับไม่ได้ (7-10) เป็นต้น และจัดทำเป็นคู่มือมาตรฐานข้อมูล

จ. การเขียนคำสั่งภาษาไพทอนสำหรับการพัฒนาระบบการประเมินพฤติกรรมทุจริตระหว่างการสอบออนไลน์ด้วยปัญญาประดิษฐ์บนระบบการรับรู้การแสดงออกทางสีหน้าของนักศึกษาแบบอัตโนมัติ

1. การเขียนคำสั่งด้วยภาษาไพทอนสำหรับการกำหนดตำแหน่งด้วย CNN

```
#import
import cv2
import mediapipe as mp
import cvzone
from cvzone.FaceMeshModule import FaceMeshDetector
import numpy as np
import time
import datetime
import utils,math
import matplotlib.pyplot as plt
import pandas as pd
import os
import mss
from sklearn.neural_network import MLPClassifier
from sklearn.metrics import accuracy_score
import sys

mp_face_mesh = mp.solutions.face_mesh
face_mesh = mp_face_mesh.FaceMesh(
    min_detection_confidence=0.5,
    min_tracking_confidence=0.5,
    # เพิ่มจำนวน detect หน้าสูงสุด
    max_num_faces=2,
)
mp_drawing = mp.solutions.drawing_utils
dawing_spec = mp_drawing.DrawingSpec(thickness=1, circle_radius=1)
count = 0
seconds = time.time()
```

```
ith_sample = 0
face_frame = 0
status = list('-'*20)

# variables
CEF_COUNTER =0
TOTAL_BLINKS =0
# constants
CLOSED_EYES_FRAME =3
mouthStatus = ''
# *----------------------------*
#screen size
# monitor = {"top": 40, "left": 40, "width": 500, "height": 500}
try:
    if len(sys.argv) == 1:
        monitor = {"top": 40, "left": 0, "width": 1000, "height": 1000}
    else:
        mornitor_t = int(sys.argv[-1].split(',')[1])
        mornitor_l = int(sys.argv[-1].split(',')[0])
        mornitor_w = int(sys.argv[-1].split(',')[2])
        mornitor_h = int(sys.argv[-1].split(',')[3])
        monitor = {"top": mornitor_t, "left": mornitor_l, "width": mornitor_w, "height":
mornitor_h}
except:
        monitor = {"top": 40, "left": 0, "width": 1000, "height": 1000}
LEFT_EYEBROW =[ 336, 296, 334, 293, 300, 276, 283, 282, 295, 285 ]
ratio_mount = 0
# right eyes indices
RIGHT_EYE=[ 33, 7, 163, 144, 145, 153, 154, 155, 133, 173, 157, 158, 159, 160, 161 , 246
]
RIGHT_EYEBROW=[ 70, 63, 105, 66, 107, 55, 65, 52, 53, 46 ]
```

```
#data list csv
#eye
RIGHT_IRIS = [474, 475, 476, 477]
LEFT_IRIS = [469, 470, 471, 472]
L_H_LEFT = [33]  # right eye right most landmark
L_H_RIGHT = [133]  # right eye left most landmark
R_H_LEFT = [362]  # left eye right most landmark
R_H_RIGHT = [263]  # left eye left most landmark
#mount
detector = FaceMeshDetector(maxFaces=1)
idList = [0,17,78,292]
countF = 0
countL = 0
countR = 0
countU = 0
countD = 0
#eye
lookL = 0
lookR = 0
lookC = 0
blink = 0
#mount
mount = 0
X_test = []
import csv
header = ['h_forward','h_left', 'h_right', 'h_up', 'h_down','L_left', 'L_Right', 'L_center',
'blink', 'mount']
data_train = [
    [countF, countL,  countR, countU,  countD, lookL, lookR, lookC, blink, mount],
]
def build_CSV():
```

```
    with open('data_train_test2.csv', 'w') as file:
        writer = csv.writer(file)
        writer.writerow(header)
        # Use writerows() not writerow()
        writer.writerows(data_train)


def landmarksDetection(img, landmark, draw=False):
        img_height, img_width= img.shape[:2]
        # list[(x,y), (x,y)....]
        mesh_coord = [(int(point.x * img_width), int(point.y * img_height)) for point in
landmark]
        if draw :
            [cv.circle(img, p, 2, (0,255,0), -1) for p in mesh_coord]


        # returning the list of tuples for each landmarks
        return mesh_coord
```

2. การเขียนคำสั่งด้วยภาษาไพทอนสำหรับการกำหนดการเปลี่ยนแปลงของวิดีโอตามช่วงเวลาด้วย MLP

ด้วยโครงข่ายการรับรู้หลายชั้น (Multi-Layer Perceptron: MLP) ทำหน้าที่หาความสัมพันธ์ของข้อมูลแต่ละลำดับของเฟรมภาพ โดยการนำภาพที่ประกอบด้วย features สำคัญซึ่งอยู่ในรูปของ feature vector จาก CNN มาคำนวณหาค่าพารามิเตอร์แล้วจากนั้นจึงนำไปคำนวณรวมกับ feature vector ของเฟรมถัดไป ทำให้การเปลี่ยนแปลงของ feature vector ของเฟรมที่ 1 ส่งผลต่อผลลัพธ์ของเฟรมที่ N

```
def euclaideanDistance(point, point1):
    x, y = point
    x1, y1 = point1
    distance = math.sqrt((x1 - x)**2 + (y1 - y)**2)
    return distance
```

```
# Blinking Ratio
def blinkRatio(img, landmarks, right_indices, left_indices):
    # Right eyes
    # horizontal line
    rh_right = landmarks[right_indices[0]]
    rh_left = landmarks[right_indices[8]]
    # vertical line
    rv_top = landmarks[right_indices[12]]
    rv_bottom = landmarks[right_indices[4]]
    # draw lines on right eyes
    # cv.line(img, rh_right, rh_left, utils.GREEN, 2)
    # cv.line(img, rv_top, rv_bottom, utils.WHITE, 2)

    # LEFT_EYE
    # horizontal line
    lh_right = landmarks[left_indices[0]]
    lh_left = landmarks[left_indices[8]]

    # vertical line
    lv_top = landmarks[left_indices[12]]
    lv_bottom = landmarks[left_indices[4]]

    rhDistance = euclaideanDistance(rh_right, rh_left)
    rvDistance = euclaideanDistance(rv_top, rv_bottom)

    lvDistance = euclaideanDistance(lv_top, lv_bottom)
    lhDistance = euclaideanDistance(lh_right, lh_left)

    reRatio = rhDistance/rvDistance
    leRatio = lhDistance/lvDistance
```

```
    ratio = (reRatio+leRatio)/2
    return ratio



# Eyes Extrctor function,
#Funcao que define distancias euclidianas dos pontos nos olhos
def euclidean_distance(point1, point2):
    x1, y1 =point1.ravel()
    x2, y2 =point2.ravel()
    distance = math.sqrt((x2-x1)**2 + (y2-y1)**2)
    return distance

#Funcao para encontra posicao da iris
def iris_position(iris_center, right_point, left_point):
    center_to_right_dist = euclidean_distance(iris_center, right_point)
    total_distance = euclidean_distance(right_point, left_point)
    ratio = center_to_right_dist/total_distance
    iris_position =""
    if ratio <= 0.42:
        iris_position="left"
        # cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,0,255),10)
    elif ratio > 0.42 and ratio <= 0.57:
        iris_position="center"
        # cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(255,0,0),10)
    else:
        iris_position = "right"
        # cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,0,255),10)
    return iris_position, ratio

build_CSV()
```

```
# landmark detection function
with mp_face_mesh.FaceMesh(max_num_faces=2, refine_landmarks=True, min_detection_confidence=0.5, min_tracking_confidence=0.5) as face_mesh:
    with mss.mss() as sct: #แคปหน้าจอ
        success = True
        print(mss.mss())
        while "Screen capturing":
            image = np.array(sct.grab(monitor))
            image.flags.writeable = False
            image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
            ith_sample += 1
            start = time.time()
            image, faces = detector.findFaceMesh(image,draw = False)

            image.flags.writeable = False

            # Get the result
            results = face_mesh.process(image)

            # To improve performance
            image.flags.writeable = True

            # Convert the color space from RGB to BGR
            image = cv2.cvtColor(image, cv2.COLOR_RGB2BGR)

            img_h, img_w, img_c = image.shape
            face_3d = []
            face_2d = []

            # ดูจำนวนคน
            # print('len(results.multi_face_landmarks)')
```

```
            cv2.putText(image, 'frame '+str(ith_sample), (20, 500), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 2)
            #cv2.putText(image, ' '.join(status), (20, 550), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 2)
            #mount//
            if faces:
                face = faces[0]
                for id in idList:
                    cv2.circle(image,face[id],5,(255,0,255),5)
                upDown,_ = detector.findDistance(face[idList[0]],face[idList[1]])
                leftRight,_ = detector.findDistance(face[idList[2]],face[idList[3]])

                ratio_mount = int((upDown/leftRight)*100)
                if ratio_mount > 50:
                    mouthStatus = 'open'
                    cv2.line(image,face[idList[0]],face[idList[1]],(0,0,255),3)
                    cv2.line(image,face[idList[2]],face[idList[3]],(0,0,255),3)
                else:
                    cv2.line(image,face[idList[0]],face[idList[1]],(0,255,0),3)
                    cv2.line(image,face[idList[2]],face[idList[3]],(0,255,0),3)
                    mouthStatus = 'close'

            #//mount
            if results.multi_face_landmarks:
                face_frame += 1
                cv2.putText(image,'face frame '+ str(face_frame), (20, 550), cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 2)
                mesh_points=np.array([np.multiply([p.x, p.y], [img_w, img_h]).astype(int) for p in results.multi_face_landmarks[0].landmark])
                (l_cx, l_cy), l_radius = cv2.minEnclosingCircle(mesh_points[LEFT_IRIS])
                (r_cx,r_cy), r_radius = cv2.minEnclosingCircle(mesh_points[RIGHT_IRIS])
```

```
            # transforma pontos centrais em array np
            center_left = np.array([l_cx, l_cy], dtype=np.int32)
            center_right = np.array([r_cx, r_cy], dtype=np.int32)
            #desenhe o círculo com base nos valores de retorno da minEnclosingCircle, através do CIRCLE que desenha a imagem do círculo com base no centro (x, y) e no raio
            cv2.circle(image, center_left, int(l_radius), (255, 0, 255), 1, cv2.LINE_AA)
            cv2.circle(image, center_right, int(r_radius), (255, 0, 255), 1, cv2.LINE_AA)
            #mostrar pontos nos cantos dos olhos
            cv2.circle(image, mesh_points[R_H_RIGHT][0], 3, (255, 255, 255), -1, cv2.LINE_AA)
            cv2.circle(image, mesh_points[R_H_LEFT][0], 3, (0, 255, 255), -1, cv2.LINE_AA)
            try:
                iris_pos, ratio = iris_position(center_right, mesh_points[R_H_RIGHT], mesh_points[R_H_LEFT][0])
            except:
                print("iris_pos error")
            #           cv2.putText(image, f"ref : {iris_pos} {ratio:.2f}",(50,50),cv2.FONT_HERSHEY_COMPLEX,0.7,(255,0,255),1)
            #eye//
            currentDate = str(datetime.datetime.now())
            for i,face_landmarks in enumerate(results.multi_face_landmarks):

                # เพิ่มกรอบหน้าขาว
                mp_drawing.draw_landmarks(
                    image=image,
                    landmark_list=face_landmarks,
                    connections=mp_face_mesh.FACEMESH_CONTOURS,
                    landmark_drawing_spec=drawing_spec,
                    connection_drawing_spec=drawing_spec)
```

```
                # ส่วนของการกระพริบตา
                mesh_coords = landmarksDetection(image, face_landmarks.landmark,
False)
                try:
                    ratio = blinkRatio(image, mesh_coords, RIGHT_EYE, LEFT_EYE)
                except:
                    print("An exception occurred")
                # cv2.putText(image, f'Ratio : {round(ratio,2)}', (20, 200),
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (0, 255, 0), 2)
                if ratio > 4.5: # ตรวจการกระพริบของตา
                    CEF_COUNTER +=1
                    # cv.putText(image, 'Blink', (200, 50), FONTS, 1.3, utils.PINK, 2)
                    utils.colorBackgroundText(image, f'Blink',
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (20, 100), 2, utils.YELLOW, pad_x=6, pad_y=6, )

                else:
                    if CEF_COUNTER>CLOSED_EYES_FRAME:
                        TOTAL_BLINKS +=1
                        CEF_COUNTER =0

                # cv.putText(image, f'Total Blinks: {TOTAL_BLINKS}', (100, 150), FONTS,
0.6, utils.GREEN, 2)
                # utils.colorBackgroundText(image, f'Total Blinks: {TOTAL_BLINKS}',
cv2.FONT_HERSHEY_SIMPLEX, 0.7, (20,150),2)
                # cv2.polylines(image, [np.array([mesh_coords[p] for p in LEFT_EYE ],
dtype=np.int32)], True, utils.RED, 1, cv2.LINE_AA) # map ตาซ้าย
                # cv2.polylines(image, [np.array([mesh_coords[p] for p in RIGHT_EYE ],
dtype=np.int32)], True, utils.GREEN, 1, cv2.LINE_AA) # map ตาขวา
                face_2d = []
                face_3d = []
```

```
                for idx, lm in enumerate(face_landmarks.landmark):

                    # เอาไว้หาตำแหน่งจุดของหน้า
                    # test_x, test_y = int(lm.x * img_w), int(lm.y * img_h)
                    # cv2.circle(image, (test_x, test_y), 2, (255, 0, 0), -1)
                    # cv2.putText(image, str(idx), (test_x, test_y), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1)

                    # เก็บตำแหน่งบนหน้าผาก
                    if idx == 10:
                        header_x,header_y = int(lm.x * img_w), (int(lm.y * img_h))
                        color_nose = (255, 0, 0)
                    # เก็บตำแหน่งจมูก
                    if idx == 33 or idx == 263 or idx == 1 or idx == 61 or idx == 291 or idx == 199:
                        # print('mosss',i)
                        if idx == 1:
                            nose_2d = (lm.x * img_w, lm.y * img_h)
                            nose_3d = (lm.x * img_w, lm.y * img_h, lm.z * 3000)
                        x, y = int(lm.x * img_w), int(lm.y * img_h)

                        # Get the 2D Coordinates
                        face_2d.append([x, y])

                        # Get the 3D Coordinates
                        face_3d.append([x, y, lm.z])

                # Convert it to the NumPy array
                face_2d = np.array(face_2d, dtype=np.float64)

                # Convert it to the NumPy array
```

```
                face_3d = np.array(face_3d, dtype=np.float64)

                # The camera matrix
                focal_length = 1 * img_w

                cam_matrix = np.array([[focal_length, 0, img_h / 2],
                                       [0, focal_length, img_w / 2],
                                       [0, 0, 1]])

                # The distortion parameters
                dist_matrix = np.zeros((4, 1), dtype=np.float64)

                # Solve PnP
                success, rot_vec, trans_vec = cv2.solvePnP(face_3d, face_2d,
cam_matrix, dist_matrix)

                # Get rotational matrix
                rmat, jac = cv2.Rodrigues(rot_vec)

                # Get angles
                angles, mtxR, mtxQ, Qx, Qy, Qz = cv2.RQDecomp3x3(rmat)

                # Get the y rotation degree
                x = angles[0] * 360
                y = angles[1] * 360
                z = angles[2] * 360

                start_point = (5, 220)
                end_point = (440, 440)
                color = (255, 0, 0)
                thickness = 2
```

```
#reset val
countF = 0
countL = 0
countR = 0
countU = 0
countD = 0
 lookL = 0
 lookR = 0
lookC = 0
 blink = 0
mount = 0
# See where the user's head tilting
if(ratio_mount > 50):
    mount = 1
if(ratio > 4.5):
    blink = 1
if(iris_pos == 'right'):
    lookR = 1
elif(iris_pos == 'left'):
    lookL = 1
else:
     lookC = 1

if angles[1] < -0.025:
    text = "Looking Right"
    str_status = "R"
    count = count+1
    countR = 1
    if(iris_pos == 'right'):
```

```
cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,0,255),10)
                    else:
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,  165,
255),10)
                    color_nose = (0, 0, 255)
                elif angles[1] > 0.025:
                    text = "Looking Left"
                    str_status = "L"
                    count = count+1
                    countL = 1
                    if(iris_pos == 'left'):

cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,0,255),10)
                    else:
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,  165,
255),10)
                    color_nose = (0, 0, 255)
                elif angles[0] < -0.025:
                    text = "Looking Down"
                    str_status = "D"
                     countD = 1
                    cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,      165,
255),10)
                    color_nose = (0, 0, 255)
                elif angles[0] > 0.025:
                    text = "Looking Up"
                    str_status = "U"
                    countU = 1

cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,255,0),10)
```

```
                    color_nose = (255, 0, 0)
                else:
                    text = "Forward"
                    count = count-1
                    countF = 1
                    str_status = "-"
                    if(iris_pos == 'right'):
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,  165,
255),10)
                    elif(iris_pos == 'left'):
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,  165,
255),10)
                    else:

cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,255,0),10)
                    color_nose = (255, 0, 0)
                #get data
                data_train.append([countF, countL,  countR, countU,  countD, lookL,
lookR, lookC, blink, mount])
                 if(len(results.multi_face_landmarks) >= 2):

cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,0,255),10)
                if(ratio > 4.5):
                    if(angles[0] < -0.025):
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,     0,
255),10)
                    else:
                        cv2.rectangle(image,(0,0),(image.shape[1],image.shape[0]),(0,  165,
255),10)
                # Display the nose direction
```

```
                nose_3d_projection, jacobian = cv2.projectPoints(nose_3d, rot_vec,
trans_vec, cam_matrix, dist_matrix)

                p1 = (int(nose_2d[0]), int(nose_2d[1]))
                p2 = (int(nose_2d[0] + y * 10), int(nose_2d[1] - x * 10))
                cv2.line(image, p1, p2, color_nose, 3)
                # Add the text on the image
                cv2.putText(image, text, (header_x,header_y),
cv2.FONT_HERSHEY_SIMPLEX, 1, color_nose, 2)
                cv2.putText(image, "x: " + str(np.round(angles[0], 2)),
(header_x,header_y-100), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
                cv2.putText(image, "y: " + str(np.round(angles[1], 2)),
(header_x,header_y-80), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
                cv2.putText(image, "z: " + str(np.round(z, 2)), (header_x,header_y-60),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
                # cv2.putText(image,str(mouthStatus),(header_x,header_y-
40),cv2.FONT_HERSHEY_COMPLEX,0.7,(255,0,255),1)
                cv2.putText(image, f"eye : {iris_pos} {ratio:.2f}",(header_x,header_y-
20),cv2.FONT_HERSHEY_COMPLEX,0.7,(255,0,255),1)
                cv2.putText(image, f"mouth : {mouthStatus}
{ratio_mount:.2f}",(header_x,header_y-
40),cv2.FONT_HERSHEY_COMPLEX,0.7,(255,0,255),1)
                #
cv2.putText(image,str(ratio),(header_x,header_y+20),cv2.FONT_HERSHEY_COMPLEX,0.
8,(255,0,255),1)
                # cv2.putText(image, text, (20, 50), cv2.FONT_HERSHEY_SIMPLEX, 2, (0,
255, 0), 2)
                # cv2.putText(image, "x: " + str(np.round(x, 2)), (500, 50),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
                # cv2.putText(image, "y: " + str(np.round(y, 2)), (500, 100),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
```

```
                # cv2.putText(image, "z: " + str(np.round(z, 2)), (500, 150),
cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)

                end = time.time()
                totalTime = end - start

                fps = 1 / totalTime

                #cv2.putText(image, f'FPS: {int(fps)}', (20, 450),
cv2.FONT_HERSHEY_SIMPLEX, 1.5, (0, 255, 0), 2)

            cv2.imshow('Head Pose Estimation1', image)

            if cv2.waitKey(5) & 0xFF == 27:
                print('break')
                build_CSV()
                break
        cv2.destroyAllWindows()
        # cap.release()
```

3. ทดสอบระบบแบบบูรณาการทั้งระบบ และปรับแก้ไขประสิทธิภาพ

การทดสอบระบบแบบบูรณาการทั้งระบบ และปรับแก้ไขประสิทธิภาพ ด้วยการวัดประสิทธิภาพของแบบจำลองมีการประเมินโดยใช้ค่า (Metric) มาตรฐานที่นิยมใช้กัน เช่น ค่าความถูกต้อง (Accuracy) เป็นต้น โดยค่าเหล่านี้คำนวณมาจากตัวเลขที่อยู่ในตาราง Confusion Matrix ซึ่งเป็นตารางที่ประกอบด้วยค่าของตัวเลขจำนวนเต็มที่แสดงการนับความถี่ของข้อมูลความเป็นจริงหรือสิ่งที่เกิดขึ้นจริง และผลลัพธ์ที่ได้จากการทำนายของแบบจำลองหรือสิ่งที่แบบจำลองคิด

```
import pandas as pd
import numpy as np
dataset = pd.read_csv('csv/data_train_face.csv')
dataset.head()
```

```
print(np.array(pd.concat([dataset["h_forward"][:5],          dataset["h_forward"][:5]],
keys=['s1', 's2'])))
print(len(dataset))
print(dataset)
X1 = [] #forward
X_train = []
#X1 = dataset["left"][:4]
#X2 = dataset["forward"][:4]
for i in range(len(dataset)):


    if(i>=25): #เริ่มเก็บตั้งแต่ตัวที่ 5
        X1 = pd.concat([dataset["h_forward"][i-24:i],
                        dataset["h_left"][i-24:i],
                        dataset["h_right"][i-24:i],
                        dataset["h_up"][i-24:i],
                        dataset["h_down"][i-24:i],
                        dataset["L_left"][i-24:i],
                        dataset["L_Right"][i-24:i],
                        dataset["L_center"][i-24:i],
                        dataset["blink"][i-24:i],
                        dataset["mount"][i-24:i]
                        ],
                       keys=['forward', #หันหน้า
                             'left',  #หันซ้าย
                             'right', #หันขวา
                             'up', #หันขึ้น
                             'down',#หันลง
                             'l', #มองซ้าย
                             'r', #มองขวา
                             'c', #มองตรงกลาง
                             'b', #กระพริบตา
```

```
                        'm' #ปาก
                    ])
        X_train.append(X1)



print(len(X1))

X_train = np.array(X_train)
print(len(X_train))

y = []
y_train = []
for i in range(len(dataset)):
    if(i>=25):
        y = dataset["answer"][i]
        y_train.append(y)
    # if(i > 1229):
    #     break

y_train = np.array(y_train)
print(len(y_train))
#print(np.array(y_train))

from sklearn.neural_network import MLPClassifier
model = MLPClassifier(alpha=1e-5,solver='adam',hidden_layer_sizes=(10),random_state=1,max_iter=30,verbose=False,activation='relu') #สร้าง Neuron
print(len(X_train))
print(len(y_train))
model.fit(X_train,y_train) #สั่ง train
from sklearn.metrics import precision_score, recall_score, f1_score
```

```
X, y = X_train, y_train
# X, y = X_test, y_test
predict_result = model.predict(X)

F1 = 2 * (precision_score(y, predict_result) * recall_score(y, predict_result)) /
(precision_score(y, predict_result) + recall_score(y, predict_result))
np.set_printoptions(threshold=np.inf)

print("MLP")
print('avg acc score:',model.score(X,y))
print('precision    :', precision_score(y, predict_result))
print('recall       :', recall_score(y, predict_result))
print('F1           :', f1_score(y, predict_result))
```