โครงงานเทคโนโลยีสารสนเทศเพื่อธุรกิจ

Senior Project in Information Technology for Business

ภาควิชาสถิติ คณะพาณิชยศาสตร์และการบัญชี

จุฬาลงกรณ์มหาวิทยาลัย

เรื่อง

66B17

การพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่อง

A Development of Machine-Generated Voice Classification Model

โดย

6342073626 พิชญะ ธีรารัตน์ตระกูล

อาจารย์ที่ปรึกษา

รศ.ดร. จันทร์เจ้า มงคลนาวิน

คณะกรรมการ

รศ.ดร. วรสิทธิ์ ชูชัยวัฒนา

ผศ.ดร. ภูริพันธ์ รุจิขจร

ปีการศึกษา 2566

บทคัดย่อ

โครงงานเทคโนโลยีสารสนเทศเพื่อธุรกิจ เรื่องการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่อง จัดทำขึ้นโดยมีวัตถุประสงค์เพื่อศึกษาหลักการทำงานของตัวแบบกับข้อมูลเสียงและเพื่อศึกษาหลักการและวิธีการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องเนื่องจากในปัจจุบันการพัฒนาปัญญาประดิษฐ์มีความก้าวหน้าเป็นอย่างมากจนปัญญาประดิษฐ์สามารถสร้างเสียงที่ใกล้เคียงกับเสียงของมนุษย์ทั้งเสียงและจังหวะการพูดการสร้างตัวแบบจำแนกเสียงสามารถป้องกันความเข้าใจผิดของมนุษย์ในการจำแนกเสียงเครื่องในงานประเภทต่างๆ

ในโครงงานนี้ผู้จัดทำได้นำชุดข้อมูลศึกษาและพัฒนาตัวแบบในการจำแนกเสียงที่สร้างขึ้นโดยเครื่องจากเสียงของมนุษย์ที่เก็บจากกลุ่มตัวอย่างและเสียงที่เครื่องสร้างจากเว็บไซต์ Botnoi Voice ประกอบด้วยกระบวนการเก็บข้อมูลเสียงสำหรับการพัฒนาตัวแบบ กระบวนการเตรียมข้อมูลเสียง (Data preprocessing) การสร้าง ตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องตลอดจนศึกษาภาษาไพธอน และไลบรารีที่เกี่ยวข้องกับการจำแนก ข้อมูลเสียง ผู้จัดทำได้ความรู้และความเข้าใจ เกี่ยวกับกระบวนการดังกล่าวจากการพัฒนาโครงงานนี้

กิตติกรรมประกาศ

โครงงานเทคโนโลยีสารสนเทศเพื่อธุรกิจ การพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่อง ฉบับนี้สามารถสำเร็จลุล่วงไปได้ด้วยดีเนื่องจากความกรุณาของบุคคลหลายท่าน ผู้ศึกษาขอขอบพระคุณทุกท่านที่ให้ความช่วยเหลือตลอดการทำโครงงานฉบับนี้

ขอขอบพระคุณ รองศาสตราจารย์ ดร.จันทร์เจ้า มงคลนาวิน อาจารย์ที่ปรึกษาโครงงานเป็นอย่างสูงที่กรุณาตอบรับเป็นที่ปรึกษาโครงงานฉบับนี้พร้อมทั้งให้คำปรึกษา คำแนะนำ ติดตามความก้าวหน้าของโครงงาน ตลอดจนช่วยแนะนำการแก้ไขปัญหาที่เกิดขึ้น และให้ความรู้เพิ่มเติมซึ่งเป็นประโยชน์ต่อการทำโครงงาน นอกจากนี้ยังให้คำแนะนำในเรื่องของการจัดทำรูปเล่มโครงงานและช่วยตรวจสอบแก้ไขให้โครงงานสมบูรณ์ ผู้จัดทำตระหนักและทราบซึ้งกับความทุ่มเทของอาจารย์ที่กรุณาสละเวลาคอยช่วยเหลือเสมอมา จึงขอกราบขอบพระคุณอาจารย์เป็นอย่างสูงไว้ ณ โอกาสนี้

ขอขอบคุณอาจารย์กรรมการทั้ง 2 ท่าน ได้แก่ ผู้ช่วยศาสตราจารย์ ดร.ภูริพันธ์ รุจิขจร และ รองศาสตราจารย์ ดร.วรสิทธิ์ ชูชัยวัฒนา ที่เสียสละเวลามาเป็นคณะกรรมการสอบโครงงานนี้รวมถึงให้คำแนะนำ ข้อชี้แนะ ความคิดเห็นและข้อติชม อันเป็นประโยชน์อย่างยิ่งต่อผู้จัดทำในการนำมาปรับปรุงแก้ไขโครงงานให้สมบูรณ์มากยิ่งขึ้น รวมถึงขอขอบพระคุณอาจารย์ทุก ๆ ท่านทั้งในและนอกภาควิชาสถิติที่ได้ให้ความรู้ตลอดระยะเวลาการเรียนมหาวิทยาลัย ที่สามารถนำมาประยุกต์ใช้ในการพัฒนาโครงงานให้ประสบความสำเร็จ

ขอขอบคุณเพื่อน ๆ ในภาควิชาสถิติโดยเฉพาะอย่างยิ่งสาขาเทคโนโลยีสารสนเทศเพื่อธุรกิจที่คอยให้คำแนะนำ และเป็นกำลังใจซึ่งกันและกัน และผลักดันให้โครงงานสำเร็จลุล่วงไปได้ครับ

พิชญะ ธีรารัตน์ตระกูล

สารบัญ

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1. ความสำคัญและที่มาของโครงการ

ที่มาและความสำคัญของโครงการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องและเสียงของมนุษย์ เกิดจากในปัจจุบัน เทคโนโลยีปัญญาประดิษฐ์ มีบทบาทสำคัญในหลาย ๆ ด้าน หนึ่งในนั้นคือการประมวลผลภาษาธรรมชาติ (Natural Language Processing: NLP) ซึ่งรวมไปถึงการวิเคราะห์เสียง (Speech Processing) ด้วยการพัฒนาเทคโนโลยีการจำแนกเสียง (Speech Recognition) มีความสำคัญอย่างยิ่งต่อการใช้งาน ปัญญาประดิษฐ์ หลายประเภท เช่น Virtual Assistant, Siri, Alexa, Google, ระบบแปลภาษาแบบเรียลไทม์ (Real-time Translation) และ ระบบถอดเสียง (Transcription)

การพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่อง และเสียงของมนุษย์มีความสำคัญอย่างยิ่งในด้านต่างๆ เช่น ด้านความปลอดภัยป้องกันการหลอกลวงโดย ใช้เสียงที่สร้างขึ้น โดยเครื่องการระบุเสียงที่สร้างขึ้นโดย ปัญญาประดิษฐ์มีความสำคัญอย่างยิ่งต่อการควบคุมเนื้อหาในปัจจุบัน เนื่องจากเทคโนโลยีด้านปัญญาประดิษฐ์มีความก้าวหน้าอย่างรวดเร็ว ทำให้มีความสามารถในการสร้างเสียงสังเคราะห์ที่มีความเหมือนจริงมากขึ้น การระบุและแยกแยะระหว่างเสียงมนุษย์จริงและเสียงเทียมจึงเป็นกลไกสำคัญในการป้องกันการแพร่กระจายของข้อมูลที่ผิดพลาดหรือเนื้อหาเทียม (deepfake) ซึ่งอาจก่อให้เกิดผลกระทบในวงกว้างต่อสังคม ดังนั้น การจำแนกเสียงจึงเป็นเครื่องมือที่ช่วยให้ผู้ควบคุมเนื้อหามีประสิทธิภาพในการกลั่นกรองและตรวจสอบข้อมูลที่มีความเสี่ยงสูง การควบคุมเนื้อหาผ่านการจำแนกเสียงจะช่วยให้เกิดความโปร่งใสและความเชื่อถือในข้อมูลที่แพร่หลาย ซึ่งจะเป็นประโยชน์ต่อผู้ใช้งานในการตัดสินใจและปฏิบัติตามอย่างถูกต้อง นอกจากนี้ ยังสามารถนำมาใช้ในการตรวจจับและป้องกันการใช้เสียงที่สร้างขึ้นโดยปัญญาประดิษฐ์เพื่อก่อให้เกิดการหลอกลวงหรือผลประโยชน์ที่ไม่เหมาะสม

แนวทางการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องและเสียงของมนุษย์สามารถทำได้ผ่านขั้นตอน การรวบรวมข้อมูลเสียงที่สร้างขึ้นโดยเครื่องและเสียงของมนุษย์ การพัฒนาตัวแบบ Deep Learning หรือ Generative Adversarial Networks (GANs) แต่ในโครงงานฉบับนี้จะใช้ตัวแบบ Deep Learning เป็นหลัก และ ประเมินผลโดยใช้ความแม่นยำ

1.2. วัตถุประสงค์ของโครงการ

1. เพื่อศึกษาหลักการทำงานกับข้อมูลเสียง
2. เพื่อศึกษาหลักการและวิธีการพัฒนาตัวแบบด้วยการใช้เทคนิค Convolution Neural Network
3. เพื่อพัฒนาตัวแบบในการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยเครื่อง

1.3. ขอบเขตของการศึกษา

ศึกษาและพัฒนาตัวแบบในการจำแนกเสียงของมนุษย์และเสียงที่สร้างโดยเครื่องจากบนข้อมูลที่ผู้จัดทำรวบรวมจากเว็ปไซต์ botnoi Voice และเสียงมนุษย์อ่านข้อความที่กำหนด โดยกำหนดรูปแบบของข้อมูลมีความยาวไม่เกิน 2 วินาที

1.4. วิธีการดำเนินงาน

1. ศึกษาบทความที่มีเนื้อหาเกี่ยวข้องกับหลักการทำงานและวิธีการพัฒนาตัวแบบจากข้อมูลเสียง
2. ศึกษาข้อมูลเสียงที่รวบรวมมา
3. ศึกษาบทเรียนและหาข้อมูลเพิ่มเติมเกี่ยวข้องกับตัวแบบ Convolution Neural Network
4. ศึกษาขั้นตอนในพัฒนาตัวแบบ Convolution Neural Network
5. เลือกเทคนิคพัฒนาตัวแบบที่เหมาะสมกับโครงงาน
6. พัฒนาตัวแบบตามเทคนิคที่เลือก
7. ปรับปรุงประสิทธิภาพของตัวแบบ
8. ทดลองใช้ตัวแบบกับข้อมูลเสียงชุดทดสอบ
9. สรุปผลการศึกษา ปัญหา ข้อเสนอแนะ และจัดทำรายงาน

1.5. ประโยชน์ของการศึกษา

1. ได้ศึกษาหลักการ วิธีการ และเครื่องมือในการสร้างตัวแบบจากข้อมูลเสียง
2. ได้ศึกษาหลักการ วิธีการ และเครื่องมือในการสร้างตัวแบบด้วยการใช้ตัวแบบ Convolution Neural Network
3. ได้พัฒนาทักษะในการสร้างตัวแบบจากข้อมูลเสียง
4. ได้พัฒนาทักษะในการสร้างตัวแบบด้วยเทคนิค Convolution Neural Network

1.6. นิยามศัพท์สำคัญ

1. โครงข่ายประสาทเทียม (Neural Network) หมายถึง Machine Learning ที่ได้แรงบันดาลใจมาจากโครงข่ายประสาทในสมองมนุษย์มนุษย์ซึ่งประกอบด้วยนิวรอน (Neurons)
   จำนวนมากที่เชื่อมโยงกันผ่าน โครงข่ายประสาท ดังนั้นโครงสร้าง (Topology)
   จึงประกอบไปด้วยโหนด (Nodes) ที่เชื่อมโยงกัน 3 ชั้น (layers) ได้แก่ 1. Input layer 2. Hidden layer 3. Output layer (จันทร์เจ้า มงคลนาวิน, 2564)

2. โครงข่ายประสาทเทียมแบบลึก (Deep Learning) หมายถึง การพัฒนาตัวแบบประสาทเทียมที่มีจำนวน Hidden Layers มากกว่า 1 ชั้น

3. โครงข่ายประสาทแบบคอนโวลูชัน (Convolutional Neural Network) หมายถึง โครงข่ายประสาทเทียมหนึ่งในกลุ่ม bio-inspired โดยที่ CNN จะจำลองการมองเห็นของมนุษย์ที่มองพื้นที่เป็นที่ย่อยๆ และนำกลุ่มของพื้นที่ย่อย ๆ มาผสานกันเพื่อดูว่าสิ่งที่เห็นอยู่คืออะไร CNN มีการดึงคุณลักษณะผ่านการคำนวณโดยใช้หลักการเดียวกัน กับ คอนโวลูชันเชิงพื้นที่ (Spatial Convolution) ในการทำงานด้าน Image Processing ซึ่งการคำนวณ นี้จะเริ่มจากการกำหนดค่าในตัวกรอง (filter) หรือ เคอร์เนล (kernel) ที่ช่วยดึงคุณลักษณะ ที่ใช้ในการรู้จำวัตถุออกมาใช้พัฒนาตัวแบบ (Natthawat Phongchit, 2561 : ออนไลน์)

4. แถบคลื่นแม่เหล็กไฟฟ้าที่เกิดจากการหักเหของแสง (Spectrogram) หมายถึง กราฟที่แสดงสเปกตรัมความถี่ของสัญญาณเสียง โดยแกนแนวนอนแสดงเวลา แกนแนวตั้งแสดงความถี่และความเข้มของสีแสดงความแรงของสัญญาณเสียง

บทที่ 2
แนวคิดและทฤษฎีที่เกี่ยวข้อง

การจำแนกเสียง (Audio Classification) เป็นสาขาหนึ่งของการประมวลผลสัญญาณดิจิทัล (Digital Signal Processing) ซึ่งมุ่งเน้นไปที่การวิเคราะห์และจำแนกคุณลักษณะของสัญญาณเสียงที่ได้รับเข้ามา ในกรณีของการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยปัญญาประดิษฐ์นั้น มีทฤษฎีและแนวคิดหลักที่เกี่ยวข้องมากมาย เช่น การประมวลผลสัญญาณเสียง การสกัดคุณลักษณะเสียง การเรียนรู้ของเครื่อง การประเมินประสิทธิภาพ การนำทฤษฎีและแนวคิดเหล่านี้มาประยุกต์ใช้ร่วมกัน ช่วยให้สามารถสร้างระบบจำแนกเสียงมนุษย์และเสียงที่สร้างโดยปัญญาประดิษฐ์ได้อย่างมีประสิทธิภาพ ซึ่งมีประโยชน์ในหลากหลายในการใช้งาน ทั้งด้านการรักษาความปลอดภัย หรือ การตรวจสอบความถูกต้อง

2.1. ลักษณะข้อมูลเสียง

2.1.1 ความหมายของเสียง

เสียง (Sound) คือ การถ่ายทอดพลังงานจากการสั่นสะเทือนของแหล่งกำเนิดเสียงผ่านโมเลกุลของตัวกลางไปยังผู้รับ โดยที่หูของนั้น สามารถรับรู้ถึงการสั่นสะเทือนของโมเลกุลเหล่านี้ได้และได้ทำการแปลผลลัพธ์ออกมาในรูปของเสียงต่างๆ ในตัวคลื่นเสียงจะมีคุณสมบัติทาง ฟิสิกส์ ได้แก่

1. แอมพลิจูด (Amplitude) คือ ความสูงของคลื่นเสียง เป็นสิ่งกำหนดความดัง-เบาของเสียงโดยคลื่น เสียงที่มีแอมพลิจูดมากจะมีความดังเสียงมากกว่าคลื่นเสียงที่มีแอมพลิจูดต่ำ
2. ความถี่ (Frequency) คือ จำนวนครั้งในการสั่นของอนุภาคต่อ 1 หน่วยเวลา ใช้กำหนดความแหลม-ทุ้ม โดยคลื่นเสียงที่มีความถี่สูงจะมีเสียงแหลมมากกว่าคลื่นเสียงที่มีความถี่ต่ำ
3. รูปแบบคลื่น (Waveform) คือ ลักษณะการเคลื่อนที่ของคลื่นเสียงจะแตกต่างกันตามต้นกำเนิดเสียง
4. ความเร็ว (Speed) คือ ความเร็วในการเคลื่อนที่ของเสียงผ่านตัวกลางจะมีความแตกต่างกันตามอุณหภูมิของ และความหนาแน่นของตัวกลาง

เทคโนโลยีมีบทบาทสำคัญในโลกของเสียง ในปัจจุบัน สามารถบันทึกเสียง ขยายเสียง และแปลงเสียงได้อย่างง่ายดายเทคโนโลยีใหม่ๆ เช่นปัญญาประดิษฐ์ยังช่วยให้สร้างเสียงที่สมจริง ตารางที่ 2.1 แสดงการเปรียบเทียบได้โดยไม่ต้องใช้เสียงจริงการเก็บ ข้อมูลเสียงของมนุษย์กับการเก็บข้อมูลที่สร้างโดยเครื่อง

คุณสมบัติ	ไฟล์เสียงจากมนุษย์	เสียงที่สร้างโดย ปัญญาประดิษฐ์
ความถี่	ขึ้นอยู่กับไมโครโฟน	กำหนดเองได้
ความละเอียด	16 บิต หรือ 24 บิต	กำหนดเองได้
รูปแบบไฟล์	.mp3, .wav, .m4a	หลากหลาย
เสียงรบกวน	มี	ไม่มี
เสียงสะท้อน	มี	ไม่มี
คุณภาพเสียง	ขึ้นอยู่กับไมโครโฟน	ควบคุมได้
โทนเสียง	ควบคุมไม่ได้	ควบคุมได้
น้ำเสียง	ควบคุมไม่ได้	ควบคุมได้
อารมณ์	ควบคุมไม่ได้	ควบคุมได้

ตารางที่ 2.1 ตารางเปรียบเทียบคุณสมบัติของไฟล์เสียงที่อัดจากมนุษย์ และเสียงที่เครื่องสร้าง

ไฟล์เสียงจากการเก็บข้อมูลเสียงของมนุษย์และเสียงที่สร้างโดยเครื่องต่างมีข้อดีและข้อเสียแตกต่างกันไฟล์เสียงจากมนุษย์สามารถที่จะเก็บข้อมูลได้สะดวกรวดเร็ว แต่คุณภาพเสียงอาจไม่ดีเนื่องจากอุปกรณ์ที่ใช้ในการเก็บข้อมูลเสียงด้วย เสียงที่สร้างโดยปัญญาประดิษฐ์สามารถควบคุมคุณสมบัติของข้อมูลเสียงได้หลากหลายแต่ต้องการเวลาและความเชี่ยวชาญในการใช้งานอีกทั้งยากที่จะหาผู้พัฒนาปัญญาประดิษฐ์หรือเครื่องที่จะสามารถสร้างเสียงภาษาไทยอย่างถูกต้องและเหมือนกับเสียงจริงของมนุษย์ได้น้อยราย

2.1.2 การประยุคต์การเรียนรู้ของเครื่องกับข้อมูลเสียง

การวิเคราะห์เสียงเป็นกระบวนการแปลง สำรวจ และตีความสัญญาณเสียงที่บันทึกโดยอุปกรณ์ดิจิทัล โดยมีจุดมุ่งหมายเพื่อทำความเข้าใจข้อมูลเสียง โดยใช้เทคโนโลยีที่หลากหลายรวมถึงอัลกอริธึมการเรียนรู้เชิงลึก การวิเคราะห์เสียงได้รับการยอมรับอย่างกว้างขวางในอุตสาหกรรมต่างๆ ตั้งแต่ความบันเทิง ความปลอดภัย สามารถแสดงลักษณะการใช้งานที่นิยมในปัจจุบัน ได้ดังต่อไปนี้

1. การรู้จำเสียง (Voice Recognition) หมายถึง

การรู้จำเสียงเป็นเทคโนโลยีที่ช่วยให้คอมพิวเตอร์สามารถเข้าใจและตอบสนองต่อเสียงพูดของมนุษย์ เทคโนโลยีนี้อยู่เบื้องหลังฟีเจอร์ต่างๆ เช่น Siri, Google Assistant, Alexa และระบบสั่งงานด้วยเสียงอื่นๆ Voice Recognition ทำงานโดยแปลงคลื่นเสียงเป็นสัญญาณดิจิทัล จากนั้นคอมพิวเตอร์จะวิเคราะห์สัญญาณเหล่านี้เพื่อแยกแยะคำพูด เสียงรบกวน โดยเฉพาะอย่างยิ่งในสภาพแวดล้อมที่การใช้มือไม่สะดวก เช่น ขณะขับรถ หรือทำอาหาร ในปัจจุบัน เทคโนโลยี Voice Recognition พัฒนาอย่างรวดเร็ว ในอนาคตอันใกล้ ระบบ Voice Recognition จะมีความแม่นยำ มีความสามารถหลากหลาย และใช้งานง่ายมากขึ้น

การใช้เทคโนโลยีนี้ช่วยให้ผู้ใช้สามารถสั่งการและควบคุมอุปกรณ์ดิจิทัลโดยไม่ต้องใช้มือนำไปสู่การเพิ่มความสะดวกและประสิทธิภาพในชีวิตประจำวัน นอกจากนี้ยังช่วยเพิ่มการเข้าถึงเทคโนโลยีสำหรับบุคคลที่มีความบกพร่องทางกายภาพ ซึ่งอาจมีข้อจำกัดในการใช้งานระบบคอมพิวเตอร์ด้วยวิธีการมาตรฐาน

2. Deepfake หมายถึง เทคโนโลยีที่ใช้ปัญญาประดิษฐ์ สร้างสื่อสังเคราะห์โดยเฉพาะวิดีโอและเสียงที่เหมือนจริงจนแยกไม่ออก Deepfake สามารถใช้สร้างเสียงที่เหมือนจริงโดยไม่ต้องใช้เสียงจริงของบุคคล เทคโนโลยี Deepfake สามารถสร้างเสียงที่เหมือนมนุษย์ จริงมาก มีการใช้งานหลากหลาย เช่น พากย์เสียงโฆษณา มีข้อดี เช่น สะดวก ประหยัด ควบคุมเสียงได้ ถึงแม้เทคโนโลยี Deepfake จะมีประเด็นด้าน จริยธรรมและความปลอดภัย แต่ในปัจจุบันยังไม่มีการออก กฎหมายมาควบคุมเท่าที่ควร

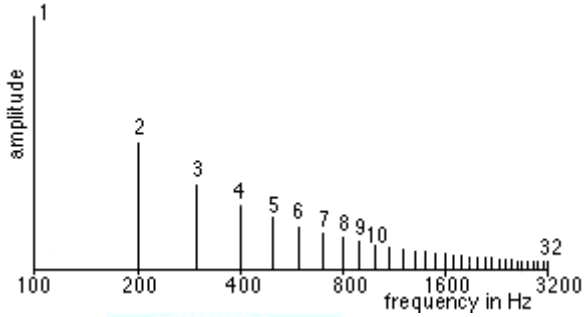
3. การจดจำเสียงสิ่งแวดล้อม (Environment Recognition) เป็นการศึกษาและมุ่งเน้นไปที่การระบุเสียงรอบตัวผู้ใช้ซึ่งให้ประโยชน์มากแก่อุตสาหกรรมยานยนต์และการผลิต และยังเป็นส่วนประกอบสำคัญในแอปพลิเคชัน IoT ในระบบของยานยนต์ เช่นการวิเคราะห์เสียงจะ "ฟัง" เหตุการณ์ภายในและภายนอกรถ ทำให้รถสามารถประมวลผลข้อมูลเพื่อเพิ่มความปลอดภัยของผู้ขับขี่ได้

การจำแนกเสียงสิ่งแวดล้อมใช้เพื่อระบุและจำแนกเสียงต่างๆในสิ่งแวดล้อมไม่ว่าจะเป็นเสียงจากรถยนต์ เสียงพูด หรือแม้กระทั่งเสียงธรรมชาติ โดยใช้ประโยชน์ในการตรวจสอบความปลอดภัย การดูแลสุขภาพ และการวิเคราะห์สิ่งแวดล้อม สามารถนำไปปรับปรุงความปลอดภัยในสถานที่ทำงาน และช่วยในการตรวจ สอบและป้องกันสถานการณ์ฉุกเฉิน เช่น การตรวจจับเสียงแก๊สรั่ว หรือการระบุเสียงของเครื่องจักรที่อาจ บ่งบอกถึงความเสี่ยงที่อาจเกิดขึ้น

2.1.3 Spectrogram

ก่อนที่จะอธิบายถึงสเปกโตรแกรม (Spectrogram) จะขออธิบายสเปกตรัม (Spectrum) ก่อนเป็นอันดับแรก

สเปกตรัม (Spectrum) คือชุดความถี่ที่รวมกันเพื่อสร้างสัญญาณ เช่น ภาพแสดงสเปกตรัมของเสียงพูดของมนุษย์หรือเสียงที่สร้างโดยเครื่องหนึ่งสเปกตรัมจะแสดงผลของความถี่ที่มีอยู่ในสัญญาณทั้งหมดพร้อมกับความดัง (Amplitude) ความถี่ต่ำสุด ในสัญญาณเรียกว่าความถี่พื้นฐาน (Fundamental Frequency) ความถี่ที่เป็นจำนวนเต็มเท่ากับความถี่พื้นฐานคือความถี่โฮร์มอนิก (Harmonics) ตัวอย่างเช่น
หากความถี่พื้นฐาน เป็น 200 เฮิรตซ์(Hz) ความถี่โฮร์ มอนิกของคือ 400 Hz, 600 Hz



รูปที่ 2.1 Spectrum ของเสียง

ที่มา: https://www.sfu.ca/sonic-studio-webdav/handbook/Spectrum.html

สเปกตรัมเป็นวิธีการแสดงสัญญาณเสียงที่เหมือนกันแต่อยู่ในรูปแบบอื่นๆ โดยแสดงความสูงตามความถี่ (Amplitude Against Frequency) และเนื่องจากแกน x แสดงช่วงของค่าความถี่ของสัญญาณ ณ เวลาใด ๆ ทำให้ Spectrum เป็นการมองคลื่นเสียงในโดเมนความถี่ (Frequency Domain) เป็นหลัก

สเปกโทรแกรม (Spectrogram) เป็นการแสดง สเปกตรัมของสัญญาณเสียงที่เปลี่ยนแปลงตามเวลา สเปกโทรแกรมแสดงสเปกตรัมของสัญญาณเสียงโดยใช้แนวแกน x เป็นเวลา (Time) และแนวแกน y เป็นความถี่ (Frequency) เหมือนทำการเก็บสเปกตรัมของสัญญาณเสียงซ้ำ ๆ ในช่วงเวลาที่แตกต่างกัน แล้วนำมาเชื่อมกันเป็นกราฟเดียว ซึ่งจะกล่าวได้ว่าเป็นเหมือนภาพถ่ายของสัญญาณเสียง ซึ่งสเปกโทรแกรมเกิดจากการคำนวณ Fourier Transform ในแต่ละช่วงเวลาย่อย ๆ ดังสมการนี้

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

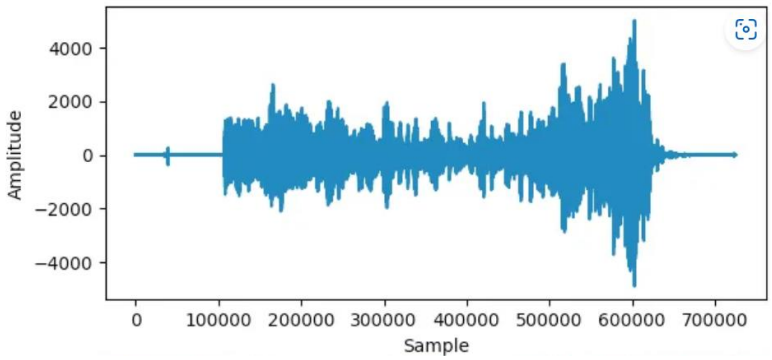
$$f(t) = \frac{1}{2\pi} \int_{-\infty}^{\infty} F(\omega) e^{i\omega t} d\omega$$

รูปที่ 2.2 สมการ Fourier Transform

ที่มา : https://proximacentauri360.wordpress.com/2012/08/15/fourier-transform/

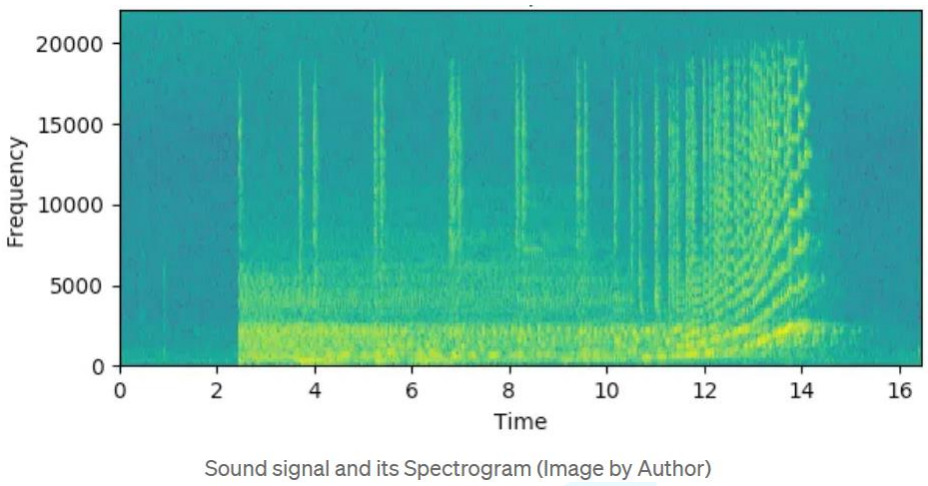
แล้วแสดงผลด้วยสีโดยสีที่แสดงจะสื่อความเข้มของสัญญาณในแต่ละช่วงความถี่ โดยสีที่สว่างแสดงถึงความเข้มของสัญญาณที่สูง และสีที่เข้มแสดงถึงความเข้มของสัญญาณที่ต่ำ

ดังตัวอย่างในภาพ 2.3 ซึ่งเป็นภาพ Plot Wave จะแสดงสัญญาณเสียงในโดเมนเวลาโดยแกน x เป็นเวลาและ แกน y เป็นค่าความเข้มของสัญญาณเสียง ซึ่งจะให้รู้สึกได้ว่าเสียงที่เป็นเสียงดังหรือเงียบอย่างไรในแต่ละ ช่วงเวลา แต่จะไม่สามารถให้เห็นข้อมูลเกี่ยวกับความถี่ที่มีอยู่ในสัญญาณเสียงได้มากนัก ต่างจากภาพ 2.4 ที่เป็นภาพของสเปกโตรแกรมโดยจะเห็นข้อมูลการแจกแจงของความถี่ได้ชัดเจนมากกว่า



รูปที่ 2.3 ตัวอย่าง Plot Wave ของเสียง

ที่มา : Audio Deep Learning Made Simple (Part 1): State-of-the-Art Techniques | by Ketan Doshi | Towards Data Science



รูปที่ 2.4 ตัวอย่าง Spectrogram

ที่มา : Audio Deep Learning Made Simple (Part 1): State-of-the-Art Techniques | by Ketan Doshi | Towards Data Science

2.1.4 Mel Spectrogram

ในสเปกโตรแกรมแบบเส้นตรง (Linear Spectrogram) ความถี่ถูกแสดงบนมาตราส่วนเส้นตรง โดยมีระยะห่างเท่ากันระหว่างบิน (Bin) ความถี่ (Frequency) นั่นหมายความว่าการเปลี่ยนแปลง 100 เฮิรตซ์จะถูกแสดงในลักษณะเดียวกันทั่วทั้งสเปกตรัมความถี่ ไม่ว่าจะเป็นจาก 100 เฮิรตซ์ไปยัง 200 เฮิรตซ์หรือจาก 10,000 เฮิรตซ์ไปยัง 10,100 เฮิรตซ์ สเปกโตรแกรมแบบเส้นตรงถูกสร้างขึ้นโดยตรงจากการแปลงฟูริเยร์ (Fourier Transform) ของสัญญาณ โดยพล็อตความถี่ (Hz) กับเวลา

ต่อมามีการคิดค้นหน่วยวัดของเสียงสำหรับการวัดความสูงต่อเสียง (Pitch) โดยมีชื่อเรียกว่าเมลสเกล (Mel Scale) โดยจากการที่แปลงเป็นความถี่เมลสเกลแสดงได้จากสมการ

$$\mathrm{Mel}(f) = 2595 \log\left(1 + \frac{f}{700}\right)$$

รูปที่ 2.5 สมการคำนวณค่า Mel scale

ที่มา: https://medium.com/prathena/the-dummys-guide-to-mfcc-aceab2450fd

ในทางกลับกัน เมลสเกลเป็นสเกลการรับรู้ของเสียงที่ผู้ฟังพิจารณาว่ามีระยะห่างเท่ากัน สเกลนี้ประมาณการว่าหูมนุษย์ตอบสนองใกล้เคียงกับสเกลความถี่เส้นตรงน้อยกว่าจึงทำให้ในสเปกโตรแกรมเมลบินและความถี่ถูกจัดเรียงตามเมล สเกลซึ่งหมายความว่ามีการให้ความละเอียดมากขึ้นแปรผันกับความถี่ต่ำกว่าเมื่อเทียบกับความถี่สูง เนื่องจากหูมนุษย์สามารถแยกแยะการเปลี่ยนแปลงเสียงที่เล็กน้อยได้ดีกว่าในความถี่ต่ำเมื่อเทียบกับช่วงความถี่สูง เมลสเกลจึงมีประโยชน์โดยเฉพาะสำหรับการประมวลผลเสียงและดนตรีและเสียงพูดของมนุษย์ เนื่องจากตรงกับการรับรู้ทางการได้ยินของมนุษย์มากกว่า

คำว่า"เมล"มาจากคำว่า"melody"เพื่อบ่งบอกวัตถุประสงค์ของสเกลซึ่งตรงกับการรับรู้ความถี่เสียงของหูมนุษย์ เมลสเกลเป็นการแปลงความถี่แบบไม่เชิงเส้น (Non-Linear Spectrogram) ที่ออกแบบมาเพื่อเลียนแบบการแก้ไขความถี่เสียงแบบลอการิทึมที่ใกล้เคียงกับการได้ยินของมนุษย์สเกลนี้รับรองว่าความแตกต่างในการรับรู้ความสูงของเสียงจะถูกจัดเรียงอย่างสม่ำเสมอทั่วทั้งสเปกตรัมความถี่ การคำนวณใช้สูตรที่แปลงเฮิรตซ์เป็นเมลและในทางกลับกัน เน้นคุณลักษณะการรับรู้ของเสียงมากกว่าการวัดทางกายภาพสเปกโตรแกรมแบบเมลสเกลมีข้อได้เปรียบหลายประการดังนี้

1. การจำลองระบบการได้ยินของมนุษย์ เมลสเกลจำลองการตอบสนองของระบบการได้ยินของมนุษย์ได้ดีกว่า ทำให้มีประสิทธิภาพมากขึ้นสำหรับงานที่เกี่ยวข้องกับการรู้จำเสียงพูดและเสียง
2. ความเกี่ยวข้องของคุณสมบัติ: ด้วยการโฟกัสความละเอียดมากขึ้นที่ความถี่ต่ำ ซึ่งเป็นที่ที่ข้อมูลเสียงพูดของมนุษย์ส่วนใหญ่อยู่สเปกโตรแกรมเมลสามารถจับคุณสมบัติที่เกี่ยวข้องได้มีประสิทธิภาพมากกว่าสเปกโตรแกรมแบบเส้นตรง
3. ประสิทธิภาพ:สเปกโตรแกรมเมลสามารถลดมิติของข้อมูล(เมื่อเปรียบเทียบกับสเปกโตรแกรมความถี่แบบเส้นตรง)โดยไม่สูญความสมบูรณ์ของข้อมูลซึ่งเหมาะกับการใช้เพื่อทำงานกับเสียงพูด

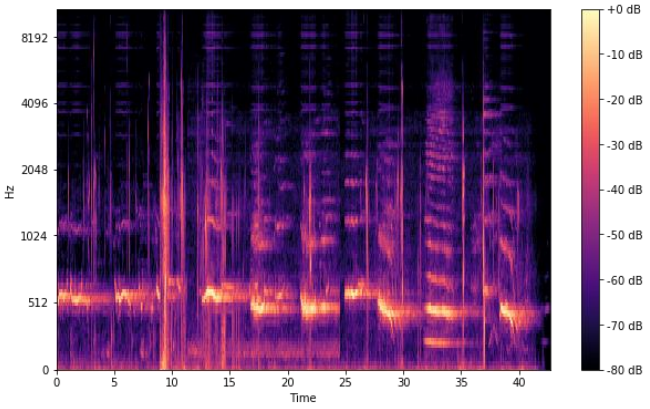
ค่าพารามิเตอร์ที่เกี่ยวข้องกับ Mel Spectrogram มีดังนี้

พารามิเตอร์กำหนดความถี่ของเสียงบน Spectrogram

1. fmin - ความถี่ต่ำสุดของเสียง
2. fmax - ความถี่สูงสุดที่กำหนดเพื่อควบคุมช่วงความถี่ที่ต้องการ
3. n_mels - จำนวนความถี่ (เช่น Mel bins) นี้คือความสูงของ Spectrogram

พารามิเตอร์กำหนดช่วงเวลาของเสียงบน Spectrogram

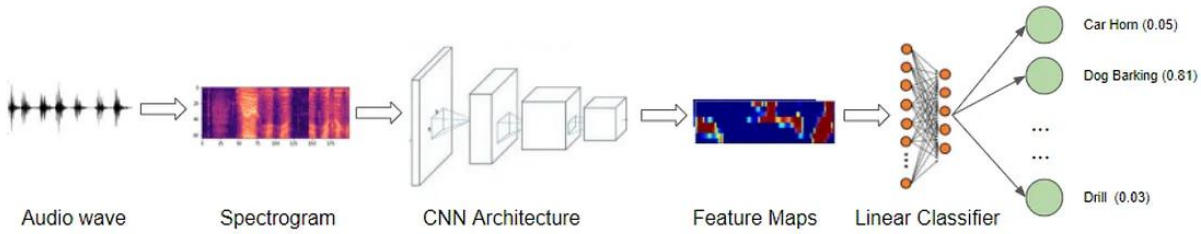
1. n_fft – ขนาดช่วงของครื่นเสียงที่นำข้าไปคิดในสมการ Fourier Transform
2. hop_length – ระยะห่างของการซ้อนทับระหว่างเฟรมของภาพ Spectrogram



รูปที่ 2.6 ตัวอย่างภาพ Spectrogram

ที่มา : Audio Deep Learning Made Simple (Part 1): State-of-the-Art Techniques | by Ketan Doshi | Towards Data Science

2.1.5 กระบวนการจัดการข้อมูลเสียงในการพัฒนา Machine Learning



รูปที่ 2.7 กระบวนการพัฒนาตัวแบบจำแนกเสียง

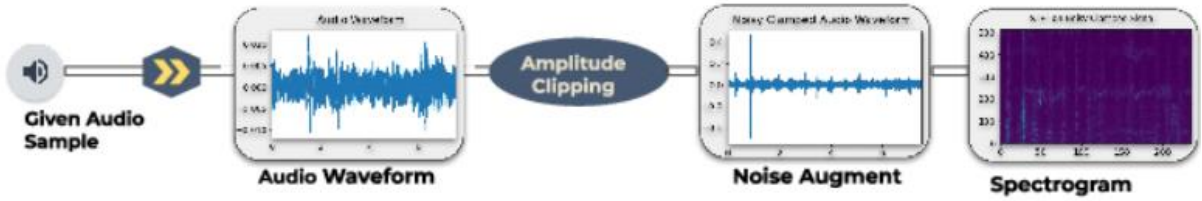
ที่มา : https://towardsdatascience.com/audio-deep-learning-made-simple-sound-classification-step-by-step-cebc936bbe5

ในการพัฒนาตัวแบบเพื่อการรู้จำเสียงของเครื่องจะมีกระบวนการทั้งหมด 3 ขั้นตอนหลักคล้ายเคียงกับการพัฒนาตัวแบบ การเรียนรู้ของเครื่องทั่วไป ได้แก่

1. การเก็บข้อมูล (Data Collection)
2. การเตรียมข้อมูล (Preprocess)
3. การพัฒนาตัวแบบ (Training Model)
4. การประเมินและปรับปรุงตัวแบบ (Evaluate)

การเก็บข้อมูล (Data Collection) เป็นการเก็บข้อมูลเพื่อนำไปใช้สำหรับพัฒนาตัวแบบกำหนดรูปแบบข้อมูลที่เก็บให้เหมาะสมตามหัวข้อของโครงการที่ศึกษา

การเตรียมข้อมูล (Preprocess) ประกอบไปด้วย การเตรียมคลาสข้อมูล, การเปลี่ยนรูปแบบเป็น Resample, การ Resize, การเพิ่มข้อมูลเสียง (Audio Augmentation) การแปลงข้อมูลเสียงจากคลื่นเสียงให้เป็น Spectrogram หรือ Mel Spectrogram และการทำ Spectrogram Augmentation



รูปที่ 2.8 ตัวอย่าง แผนผังการ Preprocess ข้อมูลเสียง จากข้อมูลคลื่นจนเป็นภาพ Spectrogram
ที่มา : https://www.researchgate.net/figure/Data-Pre-Processing-on-Raw-Audio_fig3_362324256

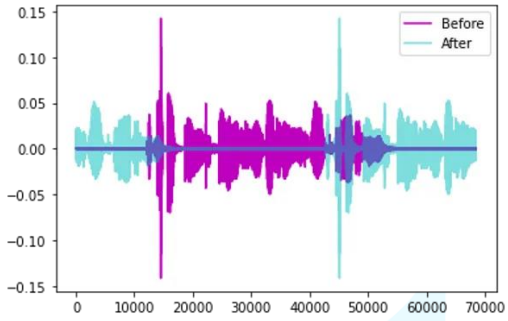
การเตรียมคลาส (Class) ข้อมูล เป็นการกำหนดคลาสข้อมูลที่ต้องการจำแนก ให้สอดคล้องกับชื่อไฟล์เสียง เช่น ชุดข้อมูลมีข้อมูลของเสียงมนุษย์และเสียงที่สร้างโดยเครื่องในการเตรียมคลาสข้อมูลจะกำหนดให้ไฟล์เสียงมนุษย์มีค่า (Label) เป็น 1 และเสียงที่สร้างโดยเครื่องมีค่าเป็น 0

ในกรณีทั่วไปในการพัฒนาตัวแบบจำแนกเสียงอาจมีการเพิ่มการสุ่มตัวอย่าง (Resample) การตัดต่อไฟล์เสียงที่มีSampling Rate (อัตราการสุ่ม ตัวอย่าง) ต่างกันไป บางไฟล์มีSampling Rate เท่ากับ 16000Hz ในขณะที่บางไฟล์อื่นมี Sampling Rate เท่ากับ 44100Hz สิ่งนี้จะทำให้ขนาดของอาร์เรย์ (Array) ที่เก็บข้อมูลเสียงต่างกันไป เพื่อให้ข้อมูลเข้ากันได้ จึงต้องทำการปรับ Sampling Rate ให้เหมือนกันทั้งหมด

การปรับขนาด (Resizing) จะปรับขนาดของเสียงให้มีความยาวเท่ากันโดยการเพิ่มเสียงเงียบ (Silence) ตอนเริ่มหรือตัดเสียงเงียบออก ในการศึกษาครั้งนี้จะกำหนดความยาวเป้าหมายให้เป็น 2 วินาที ซึ่งหมายความว่าตัวอย่างเสียงทั้งหมดจะถูกปรับขนาดให้มีความยาวเป็น 2 วินาที โดยการเพิ่มเสียงเงียบไปยังส่วนท้ายหรือตัดส่วนที่เงียบออกไปเพื่อทำให้ตัวอย่างทั้งหมดมีขนาดเดียวกันและสามารถประมวลผลโดยตัวแบบได้

การทำการเพิ่มข้อมูลเสียง (Audio Augmentation) **สามรถทำได้**หลายวิธี ตัวอย่างที่สามารถเข้าใจได้ง่าย 4 วิธีในการสร้างข้อมูลเสียงเพิ่มเติม คือ

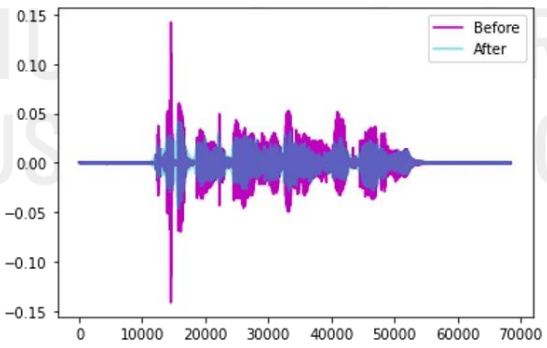
1. การเพิ่มความยาวของเสียงด้วยการแปลงเวลา (Time Shift) การแปลงเวลานี้จะถูกทำโดยการเลื่อนเสียงไปทางซ้ายหรือขวาตามจำนวนที่สุ่มได้



รูปที่ 2.9 ภาพคลื่นเสียงที่ผ่านการทำ Time Shift

ที่มา : https://towardsdatascience.com/audio-deep-learning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b52

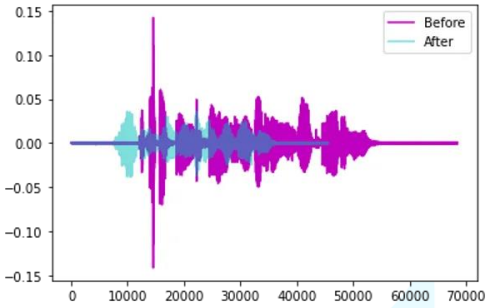
2. การเปลี่ยนแปลงความถี่ (Pitch Shift) การเปลี่ยนแปลงความถี่ของส่วนต่างๆ ในเสียงอย่างสุ่มโดยส่วนนี้อาจทำให้เสียงเปลี่ยนแปลงได้ตัวอย่างเช่นถ้า Pitch Shift เสียงขึ้น 2 ชั้นด้วยสัญญาณเสียงที่มีความถี่ 100 Hz จะกลายเป็น 200 Hz ซึ่งเป็นการเพิ่มความสูงของเสียง ให้เกิดขึ้น และถ้าลด Pitch 2 ชั้นจะกลายเป็น 50 Hz ซึ่งเป็นการลดความสูงของเสียงลง



รูปที่ 2.10 ภาพคลื่นเสียงที่ผ่านการทำ Pitch Shift

ที่มา : https://towardsdatascience.com/audio-deep-learning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b52

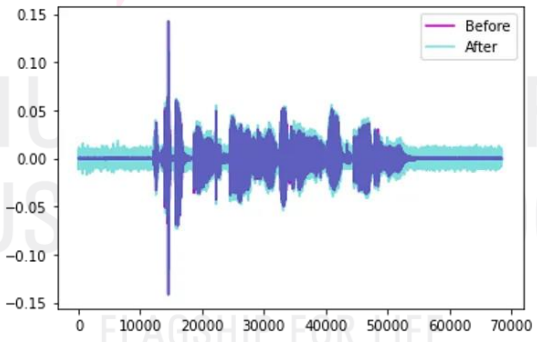
3. การเปลี่ยนความยาวเสียง (Time Stretch) เปลี่ยนความยาวเสียงเป็นสัดส่วนต่าง ๆ โดยการปรับให้เสียงช้าลงหรือเร็วขึ้น ซึ่งอาจจะเป็นประโยชน์ในกรณีที่ต้องการปรับความยาวของเสียงให้เข้ากับเวลาที่กำหนดไว้หรืออาจจะช่วยลดขนาดข้อมูลได้ในกรณีที่มีข้อมูลเสียงที่มีความยาวไม่เท่ากัน



รูปที่ 2.11 ภาพคลื่นเสียงที่ผ่านการทำ Pitch Shift

ที่มา : https://towardsdatascience.com/audio-deep-learning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b52

4. การเพิ่มเสียงรบกวน (Noise) การสร้างสัญญาณเสียงจำลองแบบสุ่มและเพิ่มลงในเสียงเดิม เช่น การสร้างเสียงรบกวนและบันทึกเสียงของวัตถุหรือสิ่งของที่กำลังถูกขยับไปมาในพื้นที่ใกล้เคียง การเพิ่ม เสียงรบกวนอาจช่วยเพิ่มความหลากหลายให้กับชุดข้อมูลเสียงในการฝึกตัวแบบ



รูปที่ 2.12 ภาพคลื่นเสียงที่ผ่านการเพิ่มเสียงรบกวน

ที่มา : https://towardsdatascience.com/audio-deep-learning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b52

การแปลงข้อมูลเสียงจากคลื่นเสียงให้เป็น Spectrogram หรือ Mel Spectrogram การสร้าง Spectrogram หรือ Mel Spectrogram ด้วยพารามิเตอร์ที่ต้องการ

การทำ Spectrogram Augmentation ทำการเพิ่มขนาดของข้อมูลได้อีกครั้ง โดยครั้งนี้จะทำการเพิ่มขนาดข้อมูลบน Mel Spectrogram แทนการเพิ่มข้อมูลเสียง (Audio Augmentation) โดยจะใช้เทคนิคที่เรียกว่า SpecAugment ซึ่งจะใช้วิธีการ ดังนี้

1. Frequency mask ลบช่วงความถี่ที่ต่อเนื่องกันออกไปโดยการเพิ่มแถบแนวนอนบน Spectrogram
2. Time mask คล้ายกับ frequency masks แต่ที่แตกต่างกันคือจะบล็อกช่วงของเวลาจาก Spectrogram โดยใช้แถบแนวตั้ง

จะสังเกตได้ว่าในข้อมูลเสียงจะมีการเพิ่มข้อมูล (Augmentation) สองจุด ได้แก่ บนข้อมูลเสียง และบน Spectrogram/Mel Spectrogram ทั้งนี้ผู้พัฒนาสามารถเลือกว่าจะทำการเพิ่มข้อมูลทั้งสองแห่งหรือแค่แห่งใดแห่งหนึ่งก็ได้

2.2 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolution Neural Network: CNN)

2.2.1 ความหมายและการประยุกต์ใช้งาน

โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network) เป็นรูปแบบหนึ่งของโครงข่ายประสาทเทียมแบบลึก (Deep Neural Network) โดยมีการจำลองพฤติกรรมการมองเห็นของมนุษย์ ซึ่งจะมองพื้นที่ส่วนย่อย แล้วนำภาพเล็กนั้นมาต่อกันเพื่อจำแนกเป็นภาพใหญ่ หรือก็คือการตรวจจับหรือสกัดลักษณะสำคัญ (Feature Extraction) ของภาพ เพื่อนำมาประมวลผลต่อโดยแบ่งการประมวลผลภาพออกเป็นหลายขั้นตอน ขั้นแรกจะเป็นการมองพื้นที่ส่วนย่อย จากนั้นจึงนำภาพเล็กๆ เหล่านั้นมาต่อกันเพื่อจำแนกเป็นภาพใหญ่ ซึ่งเป็นการตรวจจับหรือสกัดลักษณะสำคัญ (Feature Extraction) ของภาพ ก่อนที่จะนำมาประมวลผลต่อไป

โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network - CNN) เป็นโมเดลที่มีความโดดเด่นในการสกัดลักษณะสำคัญ (Feature Extraction) จากข้อมูลที่ไม่มีโครงสร้าง (Unstructured Data) โดยเฉพาะอย่างยิ่งข้อมูลประเภทรูปภาพ ซึ่งองค์ประกอบสำคัญของ CNN คือ การใช้ชั้นคอนโวลูชัน (Convolution Layer) เพื่อค้นหาลักษณะเฉพาะต่างๆ ในภาพ เช่น เส้นขอบ และการแยกแยะสี และใช้ชั้นพูลลิ่ง (Pooling Layer) เพื่อลดขนาดของภาพให้มีประสิทธิภาพในการจัดการข้อมูลมากขึ้น

คุณลักษณะที่โดดเด่นของ CNN ทำให้เป็นอัลกอริทึมที่ได้รับความนิยมและถูกนำไปพัฒนาต่อยอดเป็นสถาปัตยกรรม (Architecture) ที่มีความซับซ้อนมากขึ้น เช่น VGG16 ซึ่งมีชั้นคอนโวลูชันทั้งหมด 16 ชั้น และชั้นเชื่อมโยงแบบเต็มรูป (Fully Connected Layer) 3 ชั้น หรือ AlexNet ที่มีชั้นคอนโวลูชัน 5 ชั้น และชั้นพูลลิ่ง 3 ชั้น รวมถึงการใช้เทคนิคดรอปเอาท์ (Dropout)

โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolutional Neural Network - CNN) นิยมนำมาประยุกต์ใช้ในงานต่างๆ ดังนี้

1. การจำแนกภาพ (Image Classification)

   การจำแนกประเภทของภาพ (Image Classification) เป็นงานที่ต้องอาศัยความแม่นยำและประสิทธิภาพในระดับสูง เพื่อให้สามารถแยกแยะและจำแนกวัตถุหรือเนื้อหาในภาพได้อย่างถูกต้อง โครงข่ายประสาทเทียมแบบคอนโวลูชัน เป็นตัวแบบที่ถูกนำมาใช้ในงานจำแนกประเภทภาพเป็นอย่างดี โดย CNN จะใช้แนวคิดของชั้นคอนโวลูชัน (Convolutional Layers) และชั้นพูลลิ่ง (Pooling Layers) เพื่อสกัดลักษณะเฉพาะ (Features) ของภาพออกมา จากนั้นจึงใช้ชั้นเชื่อมโยงแบบเต็มรูป (Fully Connected Layers) ในการเรียนรู้และจำแนกภาพให้ถูกต้องตามประเภท เช่น จำแนกภาพเป็นสัตว์เลี้ยงลูกด้วยน้ำนม (Mammals) หรือ รถยนต์ (Cars)

   ด้วยความสามารถของ CNN ในการสกัดลักษณะเฉพาะของภาพอย่างมีประสิทธิภาพ ทำให้โมเดลนี้ถูกนำมาใช้อย่างแพร่หลายในงานจำแนกประเภทภาพ เพื่อให้ได้ผลลัพธ์ที่แม่นยำและน่าเชื่อถือ

2. การตรวจจับวัตถุในภาพ (Object Detection)

   การตรวจจับวัตถุ (Object Detection) ในภาพเป็นงานที่ต้องการการจำแนกและระบุตำแหน่งของวัตถุในภาพที่มีความซับซ้อน เช่น การจับตัดหมวกนักเตะ การจับมือในภาพ การตรวจจับใบหน้าในระบบรักษาความปลอดภัย หรือการตรวจจับวัตถุต่างๆ ในงานวิศวกรรมโยธา

3. การเข้ารหัสภาพ (Image Encoding)

   การเข้ารหัสภาพเป็นกระบวนการแปลงภาพเป็นรหัสเพื่อนำไปใช้ในการจัดเก็บและวิเคราะห์ข้อมูล โดย CNN จะถูกนำมาใช้ในการเข้ารหัสภาพเพื่อสกัดลักษณะเฉพาะภาพออกมาและแปลงเป็นรหัสเพื่อนำไปใช้งาน

4. การคาดการณ์ภาพ (Image Prediction)

การคาดการณ์ภาพเป็นกระบวนการทำนายภาพที่จะเกิดขึ้นในอนาคต โดย CNN จะถูกนำมาใช้ในการคาดการณ์ภาพเพื่อแยกแยะวัตถุหรือเนื้อหาในภาพว่าจะเกิดอะไรขึ้นในอนาคต

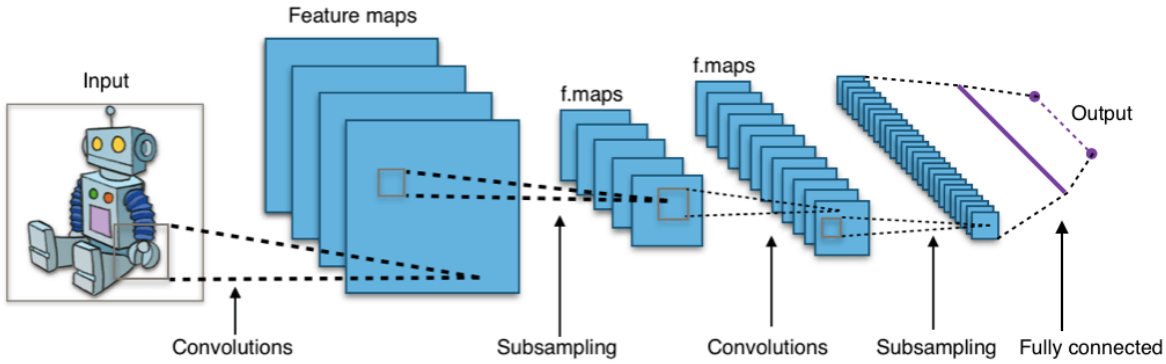
5. การสร้างตัวแบบภาษาธรรมชาติ (Natural Language Processing)

การสร้างตัวแบบภาษาธรรมชาติเป็นกระบวนการที่ใช้ในการวิเคราะห์และแปลงข้อความภาษาธรรมชาติเป็นรหัสเพื่อนำไปใช้งาน เช่น การแปลภาษา หรือ การตอบคำถาม

6. การวิเคราะห์เสียง (Speech Recognition)

การวิเคราะห์เสียงเป็นกระบวนการที่ใช้ในการแยกแยะเสียงและแปลงเป็นข้อความเพื่อนำไปใช้งาน เช่น การสั่งคำสั่งด้วยเสียง การตอบคำถามด้วยเสียง และการรับฟังโทรศัพท์อัตโนมัติ

2.2.2 กระบวนการทำงานของโครงข่ายประสาทเทียมแบบคอนโวลูชัน



รูปที่ 2.13 โครงข่ายประสาทเทียมแบบคอนโวลูชัน (Convolution Neural Network: CNN)
ที่มา : https://www.bualabs.com/archives/2461/what-is-convolutional-neural-network-cnn-convnet-mnist-deep-learning-convnet-ep-1/

แนวคิดของ CNN เป็นวิธีการที่ประสิทธิผลสูงพร้อมกับสามารถประยุคต์ใช้ได้หลากหลายด้าน แต่ด้านที่ซับซ้อนของ CNN คือระบบการคำนวณที่สอดคล้องกับ Concept ของ CNN เองและต้องมีหลักการทางคณิตศาสตร์รองรับ โดยการคำนวณตามแนวคิด CNN ใช้หลักการเดียวกันกับ คอนโวลูชันเชิงพื้นที่

(Spatial Convolution) ในการทำงานด้าน Image Processing การคำนวณนี้จะเริ่มจากการกำหนดค่าในตัวกรอง (filter) หรือ เคอร์เนล (kernel) ที่ช่วยดึงคุณลักษณะที่ใช้ในการรู้จำวัตถุออกโดยปกติตัวกรองหรือเคอร์เนลอันหนึ่งจะดึงคุณลักษณะที่สนใจออกมาได้หนึ่งอย่าง จึงจำเป็นต้องใช้เคอร์เนลหลายเตอร์เนลประมวลผลร่วมกัน เพื่อหาคุณลักษณะทางพื้นที่หลายอย่างประกอบกัน

CNN มี 3 กระบวนการทำงานหลัก ได้แก่

ขั้นตอนการคอนโวลูชัน (Convolution Stage) ขั้นตอนนี้วัตถุประสงค์เพื่อหาคุณลักษณะสำคัญที่เกี่ยวข้องกับภาพวิธีการนี้เรียกว่า คอนโวลูชัน (Convolution) และผลลัพธ์จะถูกเก็บไว้ในเมทริกซ์ชุดใหม่ที่เรียกว่า คอนโวลฟีเจอร์ (Convolved Feature) หรือผังคุณลักษณะ (Feature Map) เมื่อได้ผังคุณลักษณะจำนวนมากแล้วจึงจะเรียกชั้นนี้ว่า ชั้นคอนโวลูชัน (Convolution Layer) โดยจะมีองค์ประกอบดังนี้

1. ลักษณะของตัวกรอง

สำหรับตัวกรองของภาพดิจิทัลนั้นโดยปกติแล้วจะเป็นตารางสองมิติที่มีขนาดตามพื้นที่ย่อยตามที่พิจารณาสมมุติว่าถ้าต้องการหาเส้นตรงทะแยงสีขาว ตัวกรองของอาจจะอยู่ในลักษณะดังรูปนี้

1	-1	-1
-1	1	-1
-1	-1	1

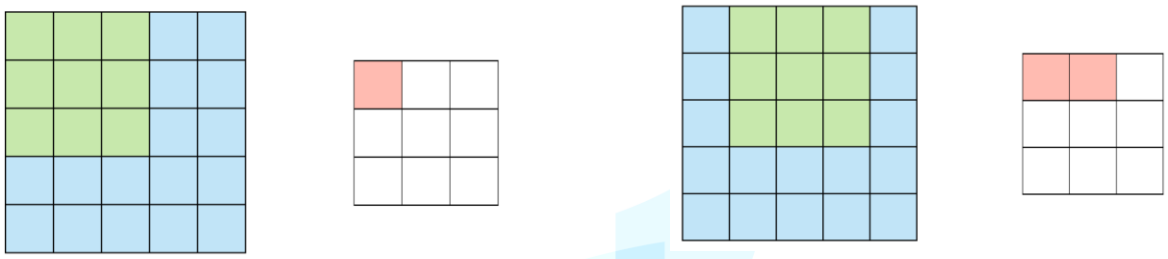
รูปที่ 2.14 ตัวอย่างตัวกรอง

ที่มา : https://medium.com/@natthawatphongchit/

ตำแหน่งตรงกลาง กรอบสีฟ้าคือ Anchor ตัวกรองจะถูก ทาบลงในพิกเซลแรกของภาพข้อมูลเข้า จากนั้นจะถูกเลื่อนไปทาบบนพิกเซลอื่นในภาพทีละพิกเซล จนครบทุกพิกเซลในภาพ อาจจะไม่ทาบตัวกรองบนพิกเซลที่อยู่ใกล้กรอบภาพเพราะตัวกรองจะล้น ออกนอกภาพเมื่อเลื่อนตัวกรองไปเรื่อย ๆ จนครบทุกพิกเซลที่สามารถเลื่อนได้ในภาพ สิ่งที่ได้นั้นจะเรียกว่าผังคุณลักษณะ (feature map)

2. Stride และ Padding

Stride เป็นตัวกำหนดว่าจะเลื่อนตัวกรองไปด้วย Step ละกี่ช่องตัวอย่างด้านล่างกำหนด Stride = 1 สามารถกำหนดค่าของ Stride ให้มากขึ้นก็ได้ถ้าต้องการให้การคำนวณหาคุณลักษณะมีพื้นที่ทับซ้อนกันน้อยลง แต่อย่างไรก็ตามการกำหนดค่าของ Stride ที่มากขึ้นจะทำให้ได้ผังคุณลักษณะ (feature map) ที่มีขนาดเล็กลง

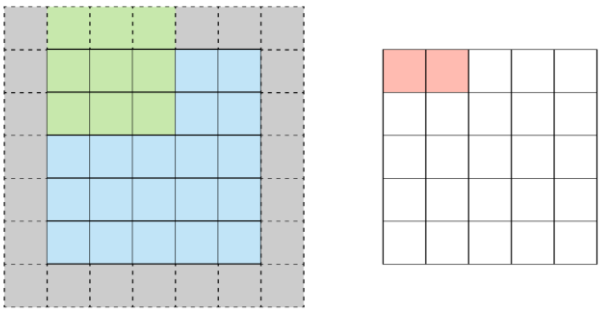


รูปที่ 2.15 ตัวอย่าง Stride

ที่มา : https://medium.com/@natthawatphongchit/

3. Padding

จากรูปด้านล่างพื้นที่สีเทารอบ Input พื้นที่เหล่านี้เป็นพื้นที่ที่เติมเข้าไป โดยจะเป็นเติม 0 หรือค่าอื่นเข้าไป เพื่อให้ในการทำ CNN นั้น Feature Map ที่ได้ยังคงมีขนาดเท่ากับ Input ในบางปัญหา Input ที่อยู่ ตามขอบภาพอาจมีความสำคัญที่ส่งผลต่อการตัดสินใจบางของตัวแบบจึงจำเป็นต้องเก็บคุณลักษณะตาม ขอบของรูปภาพไว้ด้วย



รูปที่ 2.16 ตัวอย่าง Padding

ที่มา : https://medium.com/@natthawatphongchit/

4. การพูลลิ่ง (Pooling Stage)

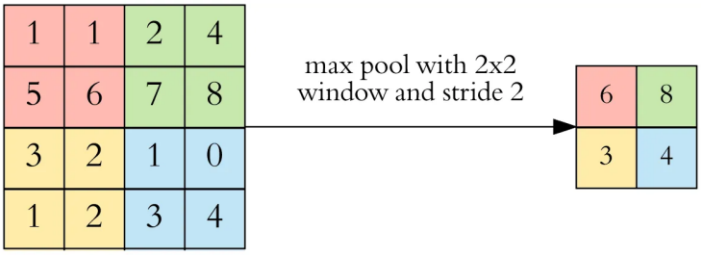
รูปที่ 2.17 ตัวอย่าง การมองภาพที่อยู่ไกลออกไป
ที่มา : https://medium.com/@natthawatphongchit/

จากภาพที่ จะเห็นว่าถึงแม้รูปภาพมีขนาดสเกลที่เล็กลง แต่ยังสามารถมองออกว่าคือเครื่องปั้นดินเผา แสดงว่า ให้เห็นถึงการจำแนกวัตถุชิ้นนี้ที่ความละเอียดต่ำลง มนุษย์จำแนกวัตถุโดยอาศัยทั้งการดูที่ราย ละเอียดเล็ก ๆ และการดูแบบคร่าว ๆ บนพื้นที่ใหญ่

โดยทั่วไปเป็นไปได้ยากมากที่จะอาศัยข้อมูลที่หยาบหรือละเอียดอย่างใดอย่างหนึ่งในการจำแนกวัตถุดังนั้นการฝึกเครื่องจึงจะเป็นต้องมีข้อมูลทั้งหยาบและละเอียดควบคู่กันไปทำให้ทราบว่าจำเป็นต้อง คำนวณภาพในหลายสเกลเพื่อความครบถ้วนของข้อมูล ปัญหาที่สำคัญคือจะทำให้การคำนวณอยู่ในรูป หลายสเกลได้อย่างไรหากใช้ตัวกรองขนาด 3x3 กำลังจะจัดการกับรายละเอียดเล็กๆ (ภาพใหญ่มีรายละ เอียดมาก จึงถือว่าเป็นสเกลละเอียด) แต่ด้วยตัวกรองขนาดเท่าเดิม หากทำกับภาพที่ขนาดเล็กลงแล้ว จะครอบคลุมพื้นที่วัตถุเดิมมากขึ้น ดังนั้นถ้าโครงข่ายควรจะต้องมีการย่อรูปประกอบด้วยก็จะสามารถเข้าถึงความสามารถด้านการวิเคราะห์ หลายความละเอียดได้

Pooling คือกระบวนการในการย่อรูปภาพแบบหนึ่งซึ่งนิยมในการทำให้ภาพมีความคมชัดน้อยลง ซึ่งมีสองประเภทหลักที่นิยมกันคือ max pooling และ mean pooling

Max Pooling เป็นตัวกรองแบบหนึ่งที่หาค่าสูงสุดในบริเวณที่ตัวกรองทาบอยู่มาเป็นผลลัพธ์ โดยจะเตรียมตัวกรองในลักษณะเดียวกับการทำ Feature Extraction ของ CNN มาทาบบนข้อมูล แล้วเลือกค่าที่สูงที่สุดบนตัวกรองนั้นมาเป็นผลลัพธ์ใหม่ และจะเลื่อนตัวกรองไปตาม Stride ที่กำหนดไว้ โดยขนาดตัวกรองของการทำ max pooling จะนิยมเรียกกันว่า pool size



รูปที่ 2.18 ตัวอย่าง Max Pooling

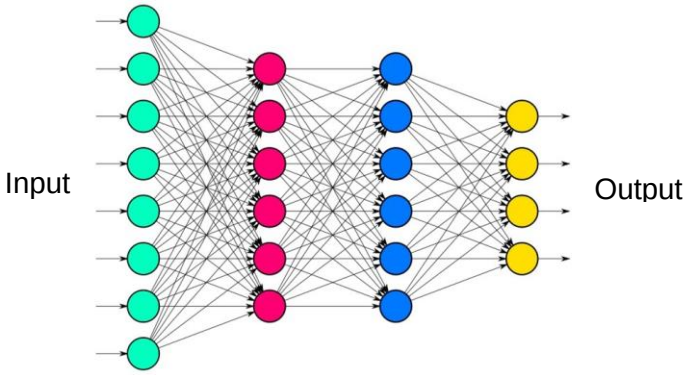
ที่มา : https://medium.com/@natthawatphongchit/

ขั้นตอนการเชื่อมโยงแบบสมบูรณ์ (Fully Connected Stage) ขั้นตอนการเชื่อมโยงแบบสมบูรณ์ (Fully Connected Stage) เป็นขั้นตอนที่ทำหน้าที่ในการจำแนก ประเภทของวัตถุ (Object) ซึ่งทุกโหนด ในชั้นการเชื่อมโยงแบบสมบูรณ์จะถูกเชื่อมกับชั้นคอนโวลูชัน และชั้นพูลลิ่งอย่างสมบูรณ์โดยกระบวนการเริ่ม จากการเปลี่ยนรูป (Re-shape) เมทริกซ์ที่ผ่านชั้นการพูลลิ่ง (Pooling Layer) ให้อยู่ในรูปคอลัมน์เดียวกัน เรียกกระบวนการนี้ว่า แฟลทเทนนิ่ง (Flattening) เพื่ออำนวยความสะดวก ในการส่งไปคำนวณในชั้นต่อไป ซึ่งเป็นชั้นโครงข่ายประสาทเทียม

รูปที่ 2.19 ภาพตัวอย่างการทำ Flattening

ที่มา : https://medium.com/geekculture/flattening-in-computer-vision-32ea85f2c9a3

หลังจากที่ข้อมูลผ่านกระบวนการแฟลทเทนนิ่ง (Flattening) แล้ว ผลลัพธ์ที่ได้จะถูกนำมาเข้าสู่ กระบวนการ เรียนรู้เชิงลึก (Deep Learning) สำหรับการเรียนรู้และแสดงผลลัพธ์สุดท้าย (Final Output) ออกมาในชั้นผลลัพธ์ (Output Layer)



รูปที่ 2.20 ตัวอย่าง Fully Connected Stage

ที่มา : https://medium.com/appengine-ai/dense-layers-in-artificial-intelligence-b2f79cc1534a

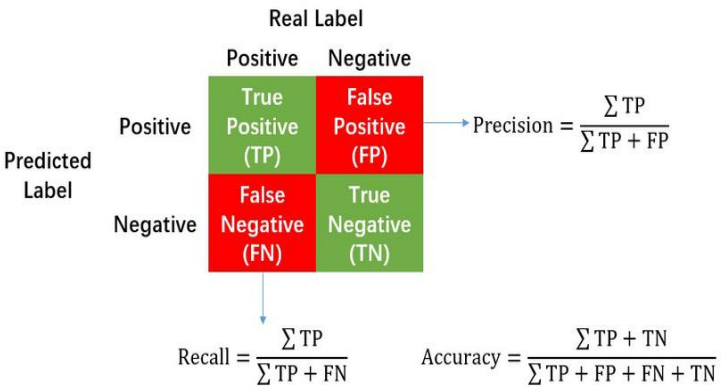
2.2.3 มาตรวัดประสิทธิภาพตัวแบบ (Model Evaluation Metric)

ตัวอย่างมาตรวัดประสิทธิภาพตัวแบบที่สำคัญ

1. Accuracy Score คือ ค่าเปรียบเทียบความถูกต้องระหว่างผลลัพธ์ที่ตัวแบบทำนายได้กับผลลัพธ์จริงซึ่งมักใช้ในการวัดประสิทธิภาพของตัวแบบที่ใช้ในงานปัญหาการจำแนก สูตร Accuracy คือ

   Accuracy Score = True Positive + True Negative / Number of Samples

   โดย True Positive คือ ตัวแบบทำนายว่าเป็นจริง และ ค่าจริงก็เป็นจริง True Negative คือ ตัวแบบทำนายว่าเป็นจริง แต่ค่าจริงเป็นเท็จ



รูปที่ 2.21 มาตรวัดที่สำคัญในการวัดประสิทธิภาพตัวแบบ

ที่มา : https://www.researchgate.net/figure/Calculation-of-Precision-Recall-and-Accuracy-in-the-confusion-matrix_fig3_336402347

2. Precision Score คือ ค่าที่วัดความแม่นยำในการทำนายว่าเมื่อตัวแบบทำนายว่าตัวอย่างในคลาสนั้นๆ เป็นบวก (positive) แล้ว จริงๆ แล้วเป็นบวกหรือไม่ (true positive) Precision score จะอยู่ในช่วง 0 ถึง 1 โดยค่าที่สูงกว่าแสดงว่าตัวแบบมีความแม่นยำในการทำนาย

3. Recall Score คือ ค่าที่วัดว่าตัวแบบสามารถจับคลุม positive คลาสได้มากเพียงใด โดยคิดจากสัดส่วนของ true positive ต่อจำนวน actual positive ทั้งหมด

4. Lift Score คือ ค่าบอกว่าตัวแบบที่สร้างขึ้นมามีประสิทธิภาพในการทำนายเป้าหมายมากเพียงใดเมื่อเทียบกับการทำนายแบบสุ่ม (random guess) โดย Lift Score จะมีค่าตั้งแต่ 0 ขึ้นไป โดยที่ 1 หมายถึงว่าตัวแบบสามารถทำนายได้ถูกต้องเท่ากับการทำนายแบบสุ่ม และค่าที่มากกว่า 1 หมายถึงว่าตัวแบบทำนายได้ดีกว่าการทำนายแบบสุ่ม ส่วนค่าน้อยกว่า 1 หมายถึงว่าตัวแบบทำนายได้แย่กว่าการทำนายแบบสุ่ม

2.3 เครื่องมือที่เกี่ยวข้องสำหรับการพัฒนาตัวแบบการเรียนรู้ของเครื่อง

เนื่องจากโครงงานนี้นั้นมีวัตถุประสงค์ในการจำแนกเสียงที่สร้างขึ้นโดยเครื่อง ผู้ศึกษาเลือกใช้ภาษาหลักเป็นภาษาไพธอน (Python) เวอร์ชัน 3.10.1 เนื่องจากภาษาไพธอน นั้นเป็นซอฟต์แวร์ Open Source และยังมีไลบรารีมากมายและครอบคลุมการทำโครงงาน เหมาะแก่การพัฒนาตัวแบบ พร้อมกับยังมีไลบรารีที่ทำงานการเตรียมข้อมูล (Data Preprocessing) ด้วย ผู้ศึกษาทำได้เลือกใช้เครื่องมือและไลบรารีต่อไปนี้ในการพัฒนาโครงการ

2.3.1 Jupyter Notebook

Jupyter Notebook เปรียบเสมือนสมุดบันทึกยุคใหม่ที่ผสมผสานโค้ดและผลลัพธ์เข้า ด้วยกัน เครื่องมือ open-source นี้ช่วยให้นักวิเคราะห์ข้อมูล นักพัฒนาซอฟต์แวร์ และนักเรียนทำงานได้อย่างมีประสิทธิภาพ ด้วยอินเทอร์เฟซที่เรียบง่าย Jupyter Notebook ช่วยให้นักพัฒนาเขียนโค้ดได้อย่างสะดวก รองรับภาษาโปรแกรมหลากหลาย เช่น Python, R, Julia, JavaScript, C++, Java ฯลฯ ผู้ใช้สามารถเขียนโค้ดทีละบรรทัดและเห็นผลลัพธ์แบบเรียลไทม์ ช่วยให้เรียนรู้ภาษาโปรแกรมใหม่หรือทดสอบโค้ดได้อย่างมีประสิทธิภาพ นอกจากนี้ Jupyter Notebook ยังเหมาะสำหรับการวิเคราะห์ข้อมูล วิเคราะห์ข้อมูลเชิงสถิติ และสร้าง visualization ได้สะดวก เครื่องมือนี้ยังรองรับการเขียนเอกสารประกอบ ช่วยให้นักพัฒนาสามารถเขียนข้อความ โค้ด และผลลัพธ์เข้าด้วยกันอีกทั้งแชร์เอกสารให้ใช้งานกับผู้อื่นได้

2.3.2 Visual Studio Code

Visual Studio Code คือโปรแกรมโอเพนซอร์สสำหรับการพัฒนาซอฟต์แวร์ที่มีความยืดหยุ่นสูง โดยมีรูปแบบการทำงานเป็นเครื่องมือแก้ไขโค้ดแบบพิมพ์ และคลิก (code editor) ซึ่งมีการแสดงผลของโค้ดเป็นรูปแบบที่เป็นไปในทิศทางของ IDE (Integrated Development Environment) รวมถึงมีฟีเจอร์สำหรับ debugging และการรันโค้ดต่าง ๆ Visual Studio Code พัฒนาโดย Microsoft และรองรับภาษาโปรแกรมหลายประเภท เช่น JavaScript, TypeScript, Python, Ruby, Go ฯลฯ และยังมีการติดตั้ง Extension ที่ช่วยให้ผู้ใช้สามารถเพิ่มฟีเจอร์หรือส่วนขยายเข้าไปในโปรแกรมได้ตาม ความต้องการ นอกจากนี้ Visual Studio Code ยังสามารถใช้งานบนหลายระบบปฏิบัติการได้ เช่น Windows, macOS, Linux ฯลฯ โดยมีการพัฒนาและอัพเดตโปรแกรม อยู่เป็นระยะๆ ซึ่งทำให้โปรแกรมเข้ากับเทคโนโลยีและสภาพแวดล้อมการพัฒนาโปรแกรมได้อย่างมีประสิทธิภาพและเหมาะสมต่อการใช้งานในปัจจุบัน

2.3.3 Tensorflow

เป็นไลบรารีโครงข่ายประสาทเทียมแบบลึก (Deep Learning Library) พัฒนาโดย Google ซึ่งมาจากคำว่า “Tensors” เป็นการรับข้อมูลนำเข้าที่เป็นอาเรย์หลายมิติและคำว่า “Flowchart” เป็นการเรียงลำดับของการประมวลผลข้อมูลที่ถูกป้อนเข้าไปจนกระทั่งออกมาเป็นผลลัพธ์ Tensorflow ถูกสร้างขึ้นมาเพื่อวัตถุประสงค์ในการพัฒนาโครงข่ายประสาทเทียมแบบลึก (Deep Learning) เป็นหลักโดยความสามารถของ Tensorflow นั้นสามารถจัดการได้ใน 3 ส่วนได้แก่ การสร้างตัวแบบ การฝึกประเมินผลตัวแบบ

2.3.4 Keras

Keras เป็นไลบรารีสำหรับ Deep Learning (Deep Learning Library) ไลบรารี Keras นั้นเป็น Highlevel interface ของไลบรารี Tensorflow โดย Keras เหมาะสำหรับการใช้งานเกี่ยวกับการสร้างโครงข่ายประสาทเทียม (Neural Network) เพราะรองรับทุกกระบวนการของการสร้างโครงข่ายประสาทเทียม สำหรับโครงงานนี้ผู้ศึกษาจะใช้ Keras ในการสร้างตัวแบบ CNN และ Hyperparameter Tuning ด้วย Bayesian Optimizer และ Hyper Band

2.3.5 Librosa

Librosa คือ ไลบรารีโอเพ่นซอร์สที่ใช้งานกับภาษา Python ถูกออกแบบมาเพื่อการประมวลผลเสียงและเพลงโดยเฉพาะ ไม่ว่าจะเป็นการโหลด บันทึก หรือแยกข้อมูลต่าง ๆ จากไฟล์เสียง Librosa มีจุดเด่นตรงที่ใช้งานง่ายเหมาะสำหรับทั้งผู้เริ่มต้นและผู้เชี่ยวชาญด้านการประมวลผลเสียง

ฟังก์ชันเด่นๆ ของ Librosa มีหลากหลาย เช่น การคำนวณสเปกตรัม (Spectrogram) ซึ่งเป็นกราฟิกแสดงความถี่และความแรงของสัญญาณเสียงตามช่วงเวลา เหมาะสำหรับการวิเคราะห์เสียงพูด หรือแยกแยะ

เสียงดนตรี นอกจากนี้ Librosa ยังสามารถคำนวณ Chromagram ซึ่งเป็นค่าที่แสดงความสัมพันธ์ของโน้ตดนตรี ผู้ศึกษาสามารถนำข้อมูลเหล่านี้ไปประยุกต์ใช้ได้อย่างหลากหลาย เช่น การจำแนกประเภทของเสียง (เสียงพูด เสียงดนตรี เสียงรบกวน) การแยกเสียงร้องออกจากเสียงดนตรี การสร้างระบบถอดเสียง (Transcription) หรือแม้กระทั่งการพัฒนาปัญญาประดิษฐ์ที่ใช้งานร่วมกับ เสียงด้วยความสามารถที่หลากหลายและใช้งานง่าย Librosa จึงเป็นเครื่องมือสำคัญสำหรับนักพัฒนา Python ที่ต้องการทำงานเกี่ยวกับเสียง ไม่ว่าจะเป็นนักวิจัย นักวิทยาศาสตร์ข้อมูล นักสร้างสรรค์เสียง หรือผู้ที่สนใจพัฒนาเทคโนโลยีด้านเสียง

2.3.6 Pandas

Pandas คือหนึ่งในไลบรารีที่ใช้สำหรับการวิเคราะห์และจัดการข้อมูลในภาษา Python โดยเฉพาะอย่างยิ่งข้อมูลแบบโครงสร้าง (structured data) เช่น ตาราง (table) หรือ ไฟล์ CSV ซึ่งมีลักษณะเหมือนกับไลบรารี Numpy แต่ Pandas จะมีฟังก์ชันที่มากกว่า และเหมาะสำหรับการจัดการข้อมูลที่มีความซับซ้อนมากขึ้น Pandas สามารถทำงานกับข้อมูลที่ไม่เรียงลำดับ มีค่าว่าง และมีการกำหนดชื่อให้กับคอลัมน์ได้อย่างสะดวก ซึ่งทำให้ผู้ใช้สามารถสร้างและจัดการข้อมูลในรูปแบบตารางได้ง่ายขึ้น นอกจากนี้ Pandas ยังมีฟังก์ชันสำหรับการรวม กรอง และการจัดกลุ่มข้อมูล รวมถึงฟังก์ชันการเชื่อมต่อข้อมูลจากหลายๆ แหล่งที่แตกต่างกันได้อย่างสะดวก Pandas เป็นไลบรารีที่นิยมใช้ในการวิเคราะห์ข้อมูลและเป็นเครื่องมือที่มีประสิทธิภาพสูงในการทำงานกับข้อมูลขนาดใหญ่ โดยเฉพาะอย่างยิ่งในการทำ Data Wrangling หรือการต่อกันของข้อมูลจากหลายแหล่ง ซึ่งสามารถเข้าถึงและใช้งานได้ง่ายจากการติดตั้งผ่าน pip หรือ conda ของ Python

2.3.7 Numpy

NumPy เป็นไลบรารี สำหรับการคำนวณทางคณิตศาสตร์และ วิทยาศาสตร์ข้อมูลในภาษา Python โดยมุ่งเน้นการทำงานกับข้อมูลแบบตัวเลขโดยเฉพาะอย่างยิ่งเมทริกซ์และอาเรย์หลายมิติทำให้สามารถดำเนินการคำนวณทางคณิตศาสตร์แบบเร็วขึ้นกว่าการใช้รูปแบบของภาษา Python และมีประสิทธิภาพสูงกว่าการใช้ list ใน Python ในการจัดการกับข้อมูลที่มีขนาดใหญ่ NumPy มีความสามารถในการดำเนินการทางคณิตศาสตร์ต่างๆ เช่น การบวก การลบ การคูณ การ หาร การยกกำลัง การแยกเวกเตอร์การหาค่าสูงสุดและต่ำสุด การคำนวณค่าเฉลี่ย การคำนวณค่าเบี่ยงเบนมาตรฐานและอื่นๆ นอกจากนี้ NumPy ยังมีฟังก์ชันสำหรับการสร้างอาเรย์ต่างๆ ได้ อย่างหลากหลาย เช่น zeros, ones, full, eye และอื่นๆ ทำให้การสร้างอาเรย์เป็นเรื่องง่ายและสะดวก NumPy เป็นไลบรารีที่สำคัญและถูกนำไปใช้งานในโครงการเทคโนโลยีต่างๆ อย่างเช่น Machine Learning, Data Science, Signal Processing, การคำนวณทางฟิสิกส์, การวิเคราะห์ และจัดการข้อมูลทางการเงิน และอื่นๆ นอกจากนี้ NumPy ยังสามารถใช้ร่วมกับไลบรารีอื่นๆ ที่เกี่ยวข้อง

2.3.8 Matplotlib

Matplotlib เป็นไลบรารีที่ใช้สำหรับการสร้างกราฟและ visualization ใน Python โดยเฉพาะอย่างยิ่งสำหรับการวิเคราะห์ข้อมูลและการแสดงผลข้อมูลทางสถิติโดย Matplotlib เป็นไลบรารีที่เป็น open-source และมีการพัฒนาโดยผู้ใช้ Python ทั่วโลก Matplotlib มีความสามารถในการสร้างกราฟหลายรูปแบบ เช่น กราฟเส้น,กราฟแท่ง, กราฟฮิสโตแกรม, กราฟเส้นผสม, และอื่นๆ นอกจากนี้ Matplotlib ยังมีฟังก์ชันในการกำหนดแกน x และ y ในกราฟ การกำหนดชื่อแกน การกำหนดหัวข้อ การกำหนดสี และอื่นๆ ทำให้Matplotlib เป็นไลบรารีที่ใช้งานได้หลากหลายและสามารถปรับแต่งได้ตามต้องการ

2.3.9 Seaborn

Seaborn เป็นไลบรารี (library) สำหรับการสร้างกราฟและภาพวาดใน Python ใช้งานง่ายและมีความสวยงามในการแสดงผลข้อมูล โดย Seaborn มุ่งเน้นไปที่การแสดงผลข้อมูลทางสถิติและง่ายต่อการใช้งาน มีฟังก์ชันการสร้างกราฟหลายรูปแบบ เช่น กราฟเส้น (line plots), กราฟแท่ง (bar plots), กราฟฮิสโตแกรม (histograms) แผนภาพการกระจาย (scatter plots) และอื่นๆ อีกทั้งมีความโดดเด่นในการสร้างกราฟแบบแผนภูมิความสัมพันธ์ (Relational plots) ที่สามารถแสดงผลข้อมูลที่มีความสัมพันธ์กันได้อย่างชัดเจน

2.3.10 Glob

glob library เป็นเครื่องมือที่มีประโยชน์สำหรับการค้นหาไฟล์และโฟลเดอร์ในภาษา Python เหมาะสำหรับการค้นหาไฟล์ตามรูปแบบที่กำหนด โดยรูปแบบนี้เรียกว่า glob pattern ซึ่ง glob pattern คล้ายกับ wildcard ที่ใช้ใน command line เช่น *.txt, data/*.csv

2.3.11 Scipy

SciPy เป็นไลบรารีที่มีประสิทธิภาพสำหรับการคำนวณทางวิทยาศาสตร์ เหมาะสำหรับนักวิทยาศาสตร์ วิศวกร นักพัฒนา และนักเรียนที่ต้องการใช้ Python สำหรับงานทางวิทยาศาสตร์ เช่น แก้ระบบสมการเชิงเส้น หาค่า Eigenvalues และ Eigenvectors หาพื้นที่ใต้กราฟ หาค่าเฉลี่ย ค่ามัธยฐาน ฟังก์ชันแกมมา ฟังก์ชันเบสเซิล กรองสัญญาณ แปลงภาพ แก้สมการเชิงอนุพันธ์ SciPy ทำงานร่วมกับ NumPy ซึ่งเป็นไลบรารีพื้นฐานสำหรับการจัดการข้อมูลเชิงตัวเลขใน Python

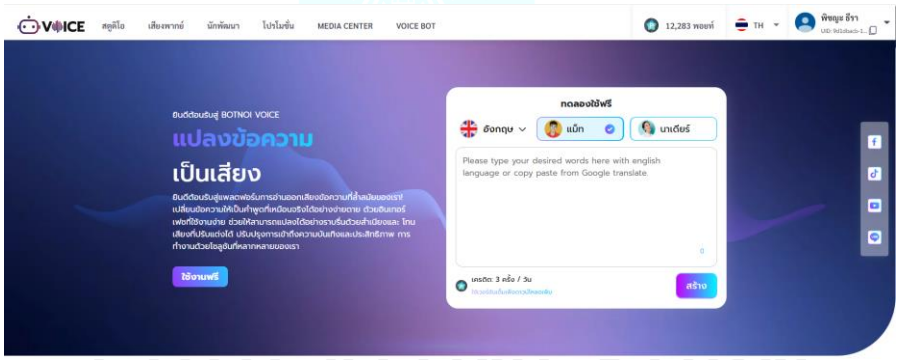
2.3.12 Shutil

Shutil เป็นไลบรารี Python สำหรับจัดการไฟล์และโฟลเดอร์ โดยมีฟังก์ชันที่คล้ายคลึงกับโมดูล os แต่ใช้งานง่ายกว่า เป็นไลบรารีที่มีประสิทธิภาพสำหรับการจัดการไฟล์เหมาะสำหรับนักพัฒนาที่ต้องการจัดการไฟล์และโฟลเดอร์ด้วยการเขียนโค้ดภาษาไพธอน

บทที่ 3
การออกแบบและพัฒนาตัวแบบ

3.1 ชุดข้อมูลและลักษณะของข้อมูล

ตัวโครงงานการศึกษาเรื่องการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องนั้นเป็นการศึกษาการพัฒนาตัวแบบเพื่อจำแนกเสียงพูดของมนุษย์ และเสียงเลียนแบบมนุษย์ที่สร้างโดยเครื่อง จึงสามารถที่จะแยกชุดข้อมูลเสียงสำหรับการพัฒนาตัวแบบออกได้เป็นสองชุด คือ ข้อมูลเสียงพูดของมนุษย์และข้อมูลเสียงที่ถูกสร้างโดยเครื่องซึ่งนำมาจากการสร้างของเทคโนโลยีปัญญาประดิษฐ์อ่านออกเสียงตามข้อความ (Text to Speech) จากเว็บไซต์ https://voice.botnoi.ai/ ซึ่งเว็บไซต์ Botnoi.voice มีเทคโนโลยีปัญญาประดิษฐ์อ่านออกเสียงตามข้อความสร้างเสียงภาษาไทยได้เหมือนเสียงอ่านข้อความของมนุษย์มากที่สุดในปัจจุบันในความเห็นของผู้ศึกษา



รูปที่ 3.1 เว็บไซต์ Botnoi Voice
ที่มา : https://voice.botnoi.ai/

การศึกษาการพัฒนาตัวแบบจำแนกเสียงที่สร้างขึ้นโดยเครื่องเป็นการศึกษาการพัฒนาตัวแบบจึงจำเป็นต้องออกแบบการพัฒนาตัวแบบเป็นรูปแบบการทดลอง ซึ่งการเก็บข้อมูลของโครงงานนี้ต้องควบคุมตัวแปลต่างๆ ที่อาจส่งผลต่อประสิทธิภาพของตัวแบบ ข้อมูลเสียงที่ผู้ศึกษาได้เลือกและกำหนดเพื่อเป็นข้อมูลสำหรับพัฒนาตัวแบบจำแนกเสียง กำหนดให้เสียงอ่านของมนุษย์จะเป็นเสียงอ่านข้อความชุดเดียวกันกับเสียงอ่านข้อความที่ถูกสร้างโดยเครื่อง มีการออกแบบองค์ประกอบของข้อมูลเสียงที่จะรวบรวมเพื่อควบคุมปัจจัยต่างๆที่อาจส่งผลต่อการพัฒนาตัวแบบ ดังนี้

1. ความยาวของข้อมูลเสียง :
   เนื่องจากข้อมูลเสียงที่ผู้ศึกษาต้องการรวบรวมเพื่อใช้เป็นข้อมูลสร้างตัวแบบต้องเป็นเสียงที่มีการอ่านข้อความที่กำหนดข้อความเดียวกัน โดยการเก็บข้อมูลเสียงอ่านข้อความของมนุษย์จำเป็นต้องนำชุดข้อความที่กำหนดให้กลุ่มทดลองอ่านและอัดเสียงซึ่งถ้าข้อความที่ให้อ่านสั้นเกินไปอาจส่งผลให้การแปลงภาพ Spectrogram จากเสียงไม่สมบูรณ์ได้ ผู้ศึกษาได้เลือกกำหนดความยาวระยะเวลาของข้อมูลเสียงที่ 2 วินาที

2. จำนวนรูปแบบของเสียง :
   เพื่อควบคุมให้เสียงอ่านข้อความของมนุษย์และที่สร้างโดยเครื่องพร้อมมีความหลากหลายของเนื้อเสียง ผู้ศึกษาจึงกำหนดให้เสียงแต่ละชุดเป็นเสียงของเพศชาย 4 เสียง และ เพศหญิง 4 เสียง ในทำนองเดียวกันเสียงที่สร้างโดยเครื่อง เป็นเสียงเพศชาย 4 เสียง และ เสียงเพศหญิง 4 เสียง

3. จำนวนข้อความต่อรูปแบบเสียง :
   เนื่องจากข้อจำกัดด้านทรัพยากรในการสร้างข้อมูลเสียงด้วยเครื่อง จำนวนที่ใช้ในการศึกษาคือ 50 ข้อความต่อเสียง 1 รูปแบบ

4. อายุของกลุ่มตัวอย่างอ่านข้อความ :
   เนื่องจากกลุ่มตัวอย่างที่ผู้ศึกษาเลือกสำหรับการอ่านข้อความเสียงโดยมนุษย์เป็นนิสิตมหาวิทยาลัยในช่วงอายุ 20 - 25 ปี

3.2 การเตรียมข้อมูล (Data Preprocessing)

1. **การเปลี่ยนรูปแบบไฟล์เสียงจาก .m4a ให้เป็น .wav**
   เริ่มต้นผู้ศึกษาได้เลือกไฟล์เสียงที่เก็บรวบรวมมาศึกษา พบข้อจำกัดของไลบรารีที่ไม่เหมาะกับการเตรียมข้อมูลในรูปแบบ .m4a จึงมีความจำเป็นที่ต้องแปลงรูปแบบไฟล์เสียงให้เป็น .wav ทั้งหมดเนื่องจากข้อมูลเสียงที่ถูกสร้างโดยเครื่องสามารถที่เลือกประเภทไฟล์ในขั้นตอนดาวน์โหลดข้อมูลเสียงได้เลย ผู้ศึกษาจึงไม่จำเป็นต้องเปลี่ยนรูปแบบไฟล์สามารภแปลงรูปแบบไฟล์

2. ตัดไฟล์เสียงที่ยาวกว่า 2 วินาที

เนื่องจากผู้ศึกษาต้องการควบคุมความยาวของข้อมูลเสียงให้มีความยาวที่เท่ากันจึงตัดข้อมูลเสียง ที่มีความยาวเกิน 2 วินาที ให้ข้อมูลเสียงนั้นเหลือเพียง 2 วินาที ตรวจสอบความ ถูกต้องสำเร็จแล้ว บันทึกทับกับไฟล์ข้อมูลเดิม โดยใช้คำสั่งต่อไปนี้

ต่อมาตรวจสอบความถูกต้องของเสียงที่ได้จากคำสั่งว่าไม่มีเสียงใดยาวกว่า 2 วินาที

3. เติมเสียงที่มีความยาวไม่ถึง 2 วินาที

เนื่องจากรูป Spectrogram ที่ผู้ศึกษาต้องการสร้างเป็นการสร้างรูปจากสเปกตรัมของสัญญาณเสียงที่เปลี่ยนแปลงตามเวลา ถ้าระยะเวลาของข้อมูลเสียงไม่เท่ากันจะส่งผลต่อการสร้างรูป Spectrogram ที่จะสร้างต่อไปผู้ศึกษาจึงเติมข้อมูลเสียงที่มีระยะเวลาสั้นกว่า 2 วินาที ด้วยเสียงว่างพร้อม ตรวจสอบความถูกต้องสำเร็จแล้วบันทึกทับกับไฟล์ข้อมูล

4. การสร้าง Melspectrogram

ใช้ library librosa ในการสร้าง MelSpectrogram โดยจะต้องมีการคำนวณค่า STFT ซึ่งเกิดจากการคำนวณค่าฟูริเยร์ทรานส์ฟอร์ม (Fourier transforms)
ก่อนและจึงจะสามารถแสดงผลในรูปแบบกราฟได้ทางผู้ศึกษายังได้ทดลองปรับค่าพารามิเตอร์ที่มีผลต่อการสร้าง MelSpectrogram ในหลายรูปแบบ เช่น

1. n_fft หมายถึง จำนวนตัวอย่างที่ใช้ในการทำฟูริเยร์ทรานส์ฟอร์ม (Fourier transforms) แต่ละครั้ง โดยพื้นฐานแล้วพารามิเตอร์นี้กำหนดขนาดของหน้าต่างที่ใช้สำหรับการทำฟูริเยร์ทรานส์ฟอร์มแต่ละครั้ง ถ้า n_fft มีค่ามากขึ้นจะให้ความละเอียดด้านความถี่ที่ดีขึ้นในสเปกโตรแกรมที่ได้ แต่อาจลดความละเอียดด้านเวลาลง

2. hop_length ทำหน้าที่กำหนดจำนวนตัวอย่างระหว่างเฟรม (Frame) ที่ต่อเนื่องกันในสเปกโตรแกรม พารามิเตอร์นี้ควบคุมการซ้อนทับระหว่างความต่อเนื่องของเฟรม ถ้าค่า hop_length น้อยจะทำให้มีเฟรมมากขึ้น และด้วยเหตุนี้จึงมีความละเอียดด้านเวลาที่สูงขึ้นในกราฟสเปกโตรแกรมที่สร้าง แต่ขนาดเฟรมที่เพิ่มขึ้นก็จะเพิ่มเวลาคำนวนด้วยเช่นกัน

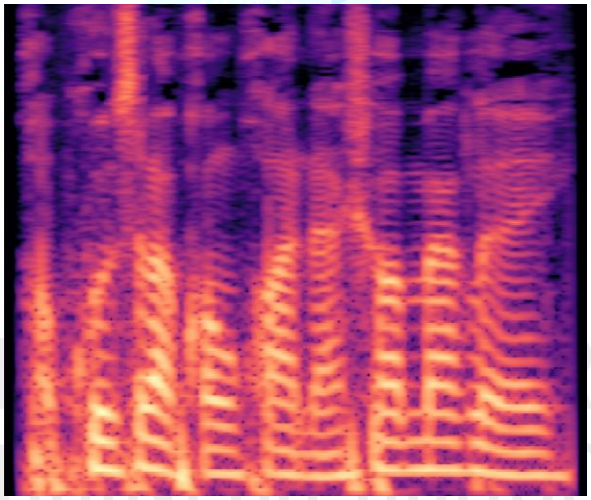
3. n_mels ทำหน้าที่กำหนดจำนวนช่องความถี่เมล (Number of Mel Frequency) ที่จะใช้ใน MelSpectrogram พารามิเตอร์นี้ควบคุมความละเอียดด้านความถี่ของ Spectrogram ถ้าค่า n_mels สูงขึ้นทำให้ความละเอียดด้านความถี่มากขึ้นแต่เพิ่มเวลาคำนวนด้วยเช่นกัน

4. fmax ระบุความถี่สูงสุดที่จะรวมอยู่ใน Spectrogram กำหนดขีดจำกัดสูงสุดสำหรับความถี่ที่แสดงในสเปกโตรแกรม โดยทั่วไปพารามิเตอร์นี้จะถูกตั้งค่าเป็นครึ่งหนึ่งของอัตราการสุ่มตัวอย่าง (sr/2) เพื่อหลีกเลี่ยงการเกิดสัญญาณหลอก

5. duration ระบุระยะเวลาของส่วนเสียงที่พิจารณา วัดเป็นวินาที ซึ่งกำหนดว่าจะประมวลผลส่วนใดของไฟล์เสียงเพื่อสร้างสเปกโตรแกรม

6. cmap พารามิเตอร์นี้กำหนดชุดสีของสเปกโตรแกรมตามความเหมาะสมของงาน

โดย Mel Spectrogram ที่สามารถสร้างได้จะถูกสร้างออกมาอยู่ในรูปแบบกราฟสเปกตรัมของ สัญญาณเสียงที่เปลี่ยนแปลงตามเวลา ผู้ศึกษาจึงต้องลบขอบที่ไม่เกี่ยวข้องกับ MelSpectrogram และอาจส่งผลต้อการพัฒนาตัวแบบออก และบันทึก MelSpectrogram ที่สร้างลงเครื่อง



รูปที่ 3.2 ตัวอย่าง MelSpectrogram

3.3 การพัฒนาตัวแบบ

1. เรียกใช้ library ในการพัฒนาตัวแบบ

2. การรวบรวมข้อมูล (Data Collection)

สร้างโฟลเดอร์แยกข้อมูล สำหรับเก็บข้อมูลโดยแยกชุดประเภทข้อมูลในแต่ละโฟลเดอร์ แยกเป็นเสียงอ่านของมนุษย์หรือเสียงที่สร้างโดยเครื่อง และ แยกตามการใช้งาน เช่น Train set, Validation set และ Test set (ถ้าไม่เคยมีโฟลเดอร์มาก่อนสามารถสร้างใหม่ในไดเรกทอรีเดิม

หลังจากสร้างโฟลเดอร์แยกข้อมูลเรียบร้อยแล้ว ผู้ศึกษาได้นำภาพ Mel Spectrogram ใส่เข้าไปในแต่ โฟลเดอร์แยกตามประเภทของข้อมูล โดยแบ่งข้อมูลตามประเภท Train set, Validation set และ Test set ในอัตตราส่วน 60 : 20 : 20 ข้อมูล

ถัดมาทำการตรวจสอบความครบถ้วนของข้อมูลในโฟลเดอร์ที่จัดเตียมไว้ โดยใช้คำสั่ง

ถัดมาทำการกำหนดค่าวัตถุ (Object) ของข้อมูลที่จะนำไปพัฒนาตัวแบบจากโฟลเดอร์ที่จัดเตรียมไว้แล้ว และทำ Label ของแต่ละชุดของข้อมูล โดยกำหนดให้ เสียงที่สร้างโดยเครื่องมีค่าเท่ากับ 0 และเสียงมนุษย์เท่ากับ 1 สามารถตรวจสอบการกำหนดค่าของวัตถุ

3. สร้างตัวแบบ Convolution Neural Network

กำหนด callback เพื่อสร้าง model check point และ early stopping โดยใช้คำสั่ง กำหนดโครงสร้าง Convolution Neural Network โดยตัวแบบแรกผู้ศึกษาได้ลองพิจรณาทดลอง กำหนดตัวแบบโดยความน่าจะเป็นที่ผู้ศึกษาคาดว่าเหมาะสม

ถัดมาทำการ compile model ซึ่งมีการเลือกใช้ optimizer เป็น Adam มี loss function เป็น binary crossentropy เพื่อให้เหมาะสมกับการจำแนกผลของข้อมูลเป็น 2 แบบ (BinaryClass) และใช้ metric เป็น accuracy

หลังจาก compile แล้วจะได้โครงสร้าง Convolution Neural Network ได้ผลดั่งรูป

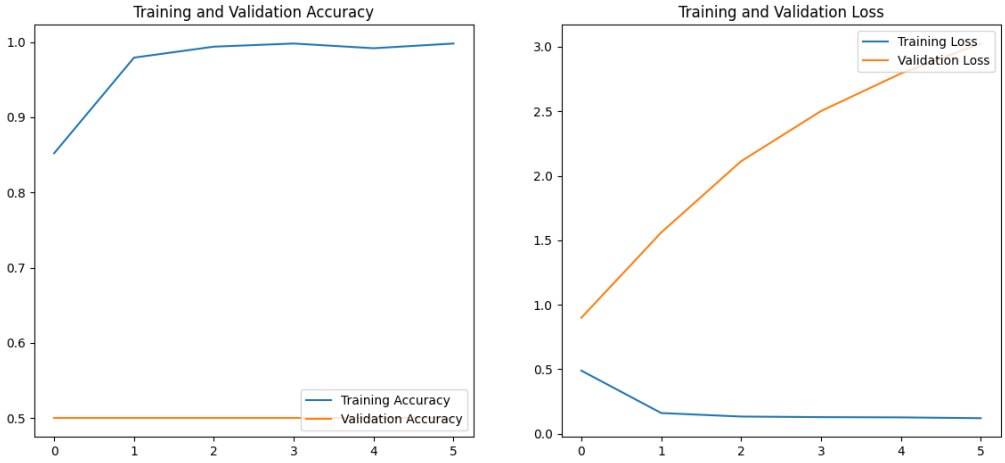
```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling (Rescaling)       (None, 224, 224, 3)       0

 conv2d (Conv2D)             (None, 224, 224, 32)      896

 batch_normalization (Batch  (None, 224, 224, 32)      128
 Normalization)

 dropout (Dropout)           (None, 224, 224, 32)      0

 conv2d_1 (Conv2D)           (None, 224, 224, 32)      9248

 max_pooling2d (MaxPooling2  (None, 112, 112, 32)      0
 D)

 batch_normalization_1 (Bat  (None, 112, 112, 32)      128
 chNormalization)

 dropout_1 (Dropout)         (None, 112, 112, 32)      0

 flatten (Flatten)           (None, 401408)            0

 batch_normalization_2 (Bat  (None, 401408)            1605632
 chNormalization)

 dense (Dense)               (None, 50)                20070450

 batch_normalization_3 (Bat  (None, 50)                200
 chNormalization)

 dropout_2 (Dropout)         (None, 50)                0

 dense_1 (Dense)             (None, 1)                 51

=================================================================
Total params: 21686733 (82.73 MB)
Trainable params: 20883689 (79.66 MB)
Non-trainable params: 803044 (3.06 MB)
_________________________________________________________________
```

รูปที่ 3.3 โครงสร้างของตัวแบบเริ่มต้น

ถัดมาทำการ fit model โดยกำหนดค่า epoch เป็น 10 และ และมี Callbacks เป็น early stopping, model checkpoint และ reduce learning rate

จากนั้นแสดงผลของการพัฒนาตัวแบบโดยใช้มาตรวัดประสิทธิภาพตัวแบบ Accuracy Score และ Loss ของ Training Set กับ Validation Set

เมื่อสร้างกราฟพบว่าได้ผลดังนี้



รูปที่ 3.4 ผลจากมาตรวัด Loss, Accuracy ทดลองบนตัวแบบเริ่มต้น

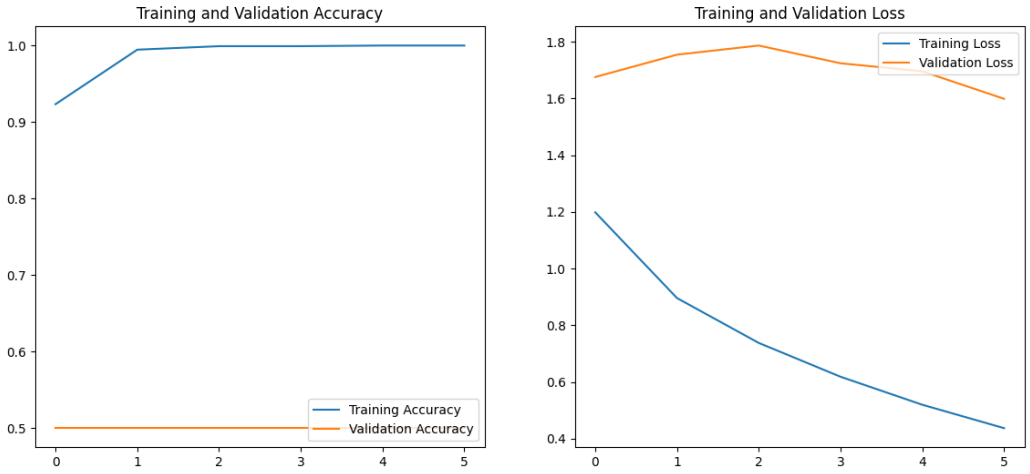
จากกราฟที่ได้จากตัวแบบแรกจะสามารถเห็นได้ถึง 2 ปัญหาหลัก คือ Validation Accuracy มีค่าเท่ากันทุก Epoch ที่ 0.5 ซึ่งแสดงให้เห็นถึงปัญหาในการจำแนกเสียงของตัวแบบ และจากกราฟ Loss ในepoch ต่างๆ เส้น Validation Loss และ Training Loss ที่ออกห่างจากกันใน epoch ที่มากขึ้นแสดงให้เห็นถึงปัญหา Overfitting ซึ่งผู้ศึกษาได้สันนิษฐานว่าข้อมูลเสียงของผู้ศึกษามีจำนวนที่น้อยเกินไป จึงดำเนินการเพิ่มจำนวนข้อมูลจากข้อมูลที่มีอยู่

4. การเพิ่มข้อมูลบนไฟล์เสียง (Data Augmentation)

ผู้ศึกษาเลือกที่จะทำการเพิ่มข้อมูลเสียง (Audio Augmentation) จากข้อมูล Training Set และ Validation Set จำนวน 4 เท่าจากข้อมูลเสียงเดิม โดยใช้เทคนิคในการ Augmentation 2 แบบ ได้แก่

1. การปรับความสูง-ต่ำของเสียง (Shift Pitch) โดยปรับให้เสียงสูงขึ้น 1 ขั้น และต่ำลง 1 ขั้นตามค่าเริ่มต้นของไลบรารี
2. การปรับความเร็ว-ช้าในเสียงพูด (Time Stretch) โดยปรับให้มีอัตราความเร็วช้าลงเป็น 90% ของความเร็วเดิม และ 110% ของความเร็วเดิม

ผลที่ได้ ข้อมูลเสียงก่อนกระบวนการการเพิ่มข้อมูล Augmentation ทั้งหมด 800 เสียง เพิ่มเป็น 4000 เสียง และหลังจากที่ผู้ศึกษาได้เพิ่มข้อมูลเสียงที่มีอยู่แล้ว ผู้ศึกษาได้ใช้ตัวแบบแรกกับข้อมูลชุดใหม่ที่เพิ่มเข้ามาพร้อมกับข้อมูลเดิมได้ผลดังนี้



รูปที่ 3.5 ผลจากมาตรวัด Loss, Accuracy ทดลองบนตัวแบบเริ่มต้นบนชุดข้อมูลที่ผ่าน Augmentation

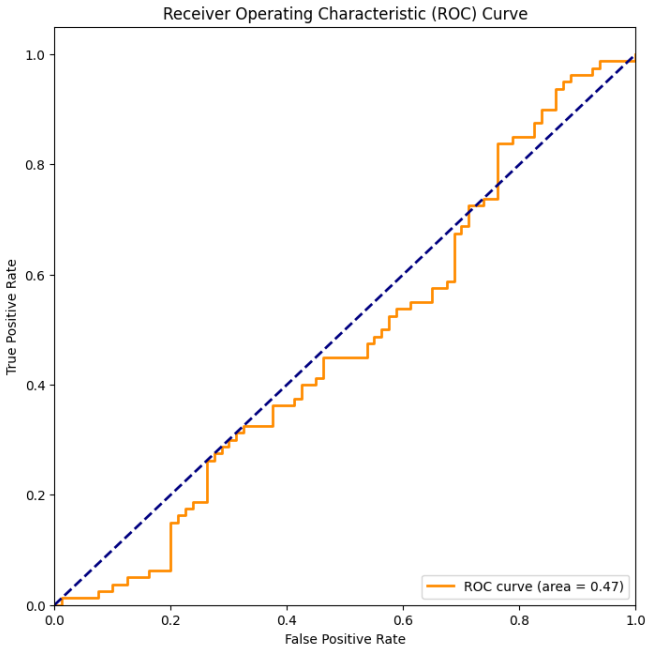
จากผลที่ได้ตามรูป 3.5 จะพบว่าตั่งแต่ epoch ที่ 1 จนถึง epochสุดท้าย ค่า Validation Accuracy ที่แสดงโดยใช้เส้นสีส้มมีค่าคงที่ตลอดเท่ากับ 0.5 ซึ่งแสดงว่าตัวแบบทำการทำนายค่าใดค่าหนึ่งตลอดการ ทำนายใน Validation Set เนื่องจากข้อมูลที่นำเข้ามาพัฒนาและทดสอบตัวแบบ เป็นข้อมูลที่สมดุล (Balance Data) คือทั้งสองชุดข้อมูลที่นำมาพัฒนาตัวแบบมีขนาดที่เท่ากันเมื่อตัวแบบทำนาย ค่าเป็นแบบใดแบบหนึ่งทั้งหมด แบบเดียวจึงทำให้ Accuracy Score ที่ได้จาก Validation Set มีค่าคงที่เท่ากับ 0.5 ตลอดทุก epoch ทั้งนี้แสดงให้เห็นถึงปัญหา Overfitting ผู้ศึกษาได้ตรวจสอบนำผลลัพธ์การทำนายของตัวแบบใน Validation Set

ได้ผลดังภาพ


```
3/3 [==============================] - 1s 191ms/step
Sample 1: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.9630241]
Sample 2: Predicted Class=0,   Actual Class=0,   Probability Scores=[0.7152182]
Sample 3: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.8221617]
Sample 4: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.967774]
Sample 5: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.9541432]
Sample 6: Predicted Class=0,   Actual Class=0,   Probability Scores=[0.61647403]
Sample 7: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.98013806]
Sample 8: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.96771187]
Sample 9: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.8893137]
Sample 10: Predicted Class=0,   Actual Class=1,   Probability Scores=[0.8896782]
```

รูปที่ 3.6 ผลจากการทำนายของตัวแบบใน Validation Set

ผู้ศึกษาได้ทดลองนำตัวแบบที่ได้ไปเขียนแผนภาพมาตรวัด ROC (Receiver Operating Characteristic) เพื่อยืนยันข้อสันนิฐานของผู้ศึกษาว่าตัวแบบที่พัฒนายังมีข้อบกพร่อง



รูปที่ 3.7 ผลจากมาตรวัด ROC ของการทำนายของตัวแบบใน Validation Set

จากผลลัพธ์ที่ได้จากมาตรวัด ROC พบว่าตัวแบบนี้ไม่สามารถแยกระหว่างคลาสข้อมูล (binary classes) ได้ดี โดยทั่วไปแล้วตัวแบบที่มีประสิทธิภาพที่ดีจะมีค่า AUC (Area Under the ROC Curve) สูงกว่า 0.5 อย่างมีนัยสำคัญ ค่า AUC ต่ำกว่า 0.5 บ่งบอกว่าตัวแบบทำงานได้แย่กว่าการเดาแบบสุ่ม ผู้ศึกษาจึงปรับปรุงการพัฒนาตัวแบบ

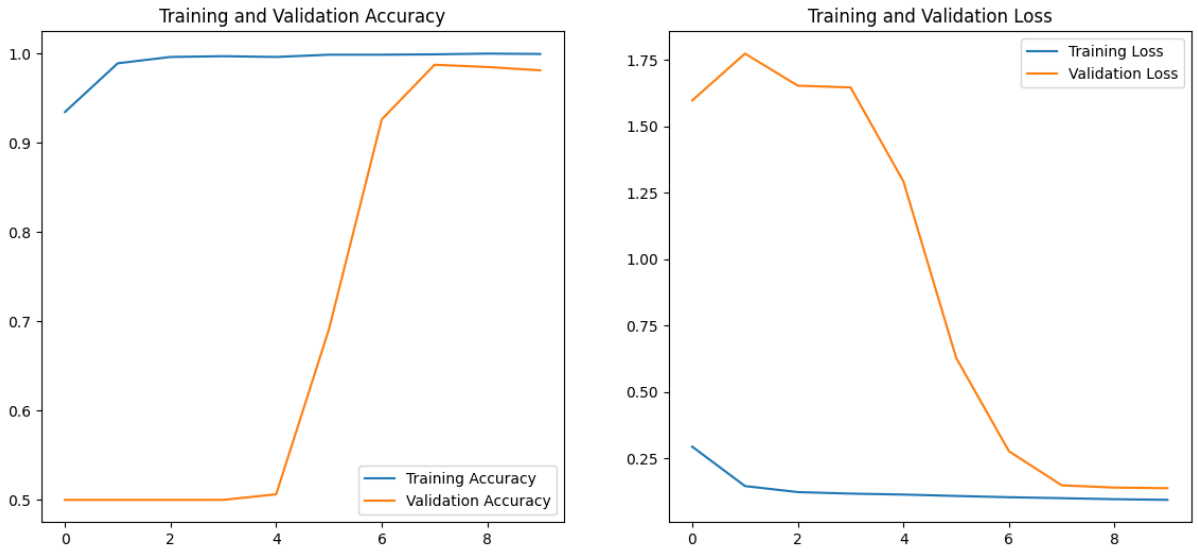
ผู้ศึกษาลดชั้นคอนโวลูชั่นลงเหลือ 1 ชั้น, ลดจำนวน Node ใน Dense Layer ลงครึ่งหนึ่ง จาก 100 Node เหลือ 50 Node และเพิ่มอัตราส่วน Node ที่จะถอกออกไปในระหว่างการเทรน (Dropout) ตัวแบบจาก 0.3 เป็น 0.6 เพื่อแก้ปัญหา Overfit โดยจะขอเรียกว่า ตัวแบบที่ 2 และสร้างโครงสร้างของ ตัวแบบที่ 2

จะได้โครงสร้างตัวแบบ Convolution Neural Network แบบที่ 2 ดังภาพ

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_2 (Rescaling)     (None, 224, 224, 3)       0

 conv2d_3 (Conv2D)           (None, 224, 224, 32)      896

 max_pooling2d_2 (MaxPoolin  (None, 112, 112, 32)      0
 g2D)

 batch_normalization_7 (Bat  (None, 112, 112, 32)      128
 chNormalization)

 dropout_5 (Dropout)         (None, 112, 112, 32)      0

 flatten_2 (Flatten)         (None, 401408)            0

 batch_normalization_8 (Bat  (None, 401408)            1605632
 chNormalization)

 dense_4 (Dense)             (None, 50)                20070450

 batch_normalization_9 (Bat  (None, 50)                200
 chNormalization)

 dropout_6 (Dropout)         (None, 50)                0

 dense_5 (Dense)             (None, 1)                 51

=================================================================
Total params: 21677357 (82.69 MB)
Trainable params: 20874377 (79.63 MB)
Non-trainable params: 802980 (3.06 MB)
_________________________________________________________________
```

รูปที่ 3.8 โครงสร้างของตัวแบบที่ 2

จากนั้นแสดงผลของตัวแบบโดยใช้มาตรวัดประสิทธิภาพตัวแบบ Accuracy Score และ Loss ของ Training Set กับ Validation Set เมื่อสร้างกราฟพบว่าได้ผลลัพธ์ดังภาพ



รูปที่ 3.9 ผลจากมาตรวัด Accuracy และ Loss ของตัวแบบที่ 2

ทดสอบตัวแบบที่ 2 ที่สร้างขึ้นมากับข้อมูลทดสอบ TestSet ได้ผลลัพท์ดังภาพ

25/25 [==============================] - 5s 173

รูปที่ 3.10 ผลการทำนายของตัวแบบที่ 2 บน TestSet

ผลการทดลองจากภาพ 3.11 ในครั้งนี้จะเห็นว่า Validation Accuracy มีการเพิ่มขึ้นตั้งแต่ epoch ที่ 4 เป็นต้นไป และ Accuracy Score บน Test Set มีค่าเท่ากับ 0.6125

3.4 การปรับไฮเปอร์พารามิเตอร์

จากผลที่ได้ในขั้นตอนการพัฒนาตัวแบบผู้ศึกษาตัดสินใจที่จะใช้โครงสร้างตัวแบบ 2 เป็นหลักในการจูน Parameter โดยการใช้ Bayesian Optimizer ซึ่งพัฒนาโดย keras เลือกช่วงการสุ่มค่า (Search Space) ดังนี้

1. Filters โดยผู้ศึกษาจะกำหนดช่วงของค่าที่สามารถเป็นไปได้ของ Filter ซึ่งกำหนดช่วงระหว่าง 32 ถึง 265 Filter และเพิ่มจำนวน Filter ถัดไปครั้งละ 32 อัน ค่าที่เป็นไปได้ของ Filter คือ 32, 64, 96, 128, 160, 192, 224 และ 256 Filter โดย Keras Tuner จะทำการสุ่มค่า Filter ตามช่วงที่กำหนด

2. Kernel_size โดยผู้ศึกษาจะกำหนดจำนวนจะกำหนดขนาด Kernel แบบสี่เหลี่ยมจัตุรัสตามค่าที่ถูกสุ่มได้ ซึ่งกำหนดช่วงระหว่าง 3 ถึง 5

3. Pool_size โดยผู้ศึกษาจะกำหนดช่วงของค่าที่สามารถเป็นไปได้ของ Pool Size ซึ่งกำหนดช่วงระหว่าง 2 ถึง 4 ค่าที่เป็นไปได้ของ Pool Size คือ 2, 3 และ 4 โดย Keras Tuner จะกำหนดขนาด Pool แบบสี่เหลี่ยมจัตุรัสตามค่าที่ถูกสุ่มได้ ตามช่วงที่กำหนด

4. Learning_rate โดยผู้ศึกษาจะกำหนดช่วงของค่าที่สามารถเป็นไปได้ของ Learning Rate ซึ่งกำหนดช่วงระหว่าง 1e-4 < x <1e-2 โดย Keras Tuner จะทำการสุ่มค่า learning rate ตามช่วงที่กำหนด

5. Dence_units โดยผู้ศึกษาจะกำหนดช่วงของค่าที่สามารถเป็นไปได้ของ Dence Units ซึ่งกำหนดช่วงระหว่าง 32 ถึง 128 Nodes และเพิ่มจำนวน Node ถัดไปครั้งละ 32 Node ค่าที่เป็นไปได้ของ Dence Units คือ 32, 64, 96, 128 Node โดย Keras Tuner จะทำการสุ่มค่า Dence Units ตามช่วงที่กำหนด

แล้วจึงสร้างตัวแบบจำลองโดยมีโครงสร้างเช่นเดียวกับตัวแบบที่ 2 ประกอบด้วยโครงสร้าง ดังนี้

1. Rescaling
2. Conv2D Layer
3. Maxpooling Layer
4. Flatten Layer
5. Fully Connected Layer

พร้อมกับ Search Space ขึ้นมา และ กำหนดค่า Tuner เป็น Tuner แบบ Bayesian Optimizer ซึ่งมีพารามิเตอร์ รับตัวแบบที่มี Search Space, Objective, จำนวนการทดลอง และจำนวนที่คำนวนต่อรอบ

จากนั้นใช้คำสั่ง tuner.search() เพื่อทำการค้นหา parameter ที่ดีที่สุดพร้อมหาค่าที่ดีที่สุดสำหรับตัวแบบที่ทำการทดลอง

ซึ่งผลที่ได้ออกมาเป็นตัวแบบจากการใช้ Bayesian Optimizer ดังนี้

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 rescaling_1 (Rescaling)     (None, 224, 224, 3)       0

 conv2d_1 (Conv2D)           (None, 222, 222, 256)     7168

 batch_normalization_1 (Bat  (None, 222, 222, 256)     1024
 chNormalization)

 max_pooling2d_1 (MaxPoolin  (None, 55, 55, 256)       0
 g2D)

 flatten_1 (Flatten)         (None, 774400)            0

 dense_2 (Dense)             (None, 128)               99123328

 dense_3 (Dense)             (None, 1)                 129

=================================================================
Total params: 99131649 (378.16 MB)
Trainable params: 99131137 (378.16 MB)
Non-trainable params: 512 (2.00 KB)
_________________________________________________________________
```

รูปที่ 3.11 โครงสร้างของตัวแบบที่ 2 หลังจากปรับพารามิเตอร์

ผลการทำนายจากตัวแบบที่มีโครงสร้างจากภาพ 3.11 ในครั้งนี้จะเห็นว่า Accuracy Score บน Test Set มีค่าเท่ากับ 0.8063 หรือ 80.63%

โดยพารามิเตอร์ที่ดีที่สุดจากการสุ่มเลือกและนำมาเป็นพารามิเตอร์สำหรับตัวแบบ มีค่าดังต่อไปนี้

1. conv_filters = 256
2. conv_kernel_size = 3
3. dense_units = 128
4. pool_size = 2
5. learning_rate: 0.0007022578158084675

บทที่ 4
ประสิทธิภาพของตัวแบบที่พัฒนา

4.1. ผลการจำแนกของตัวแบบบนผลข้อมูลชุดทดสอบของเสียงที่สร้างโดยเครื่องและเสียงของมนุษย์

ผลการจำแนกเสียงชุดทดสอบจากการตัวแบบที่พัฒนาตัวแบบสุดท้าย หรือ ตัวแบบที่ถูกปรับค่าพารามิเตอร์ได้ผลดังนี้

	precision	recall	f1-score	support
AI	0.9623	0.6375	0.7669	80
Human	0.7290	0.9750	0.8342	80
accuracy			0.8062	160
macro avg	0.8456	0.8062	0.8006	160
weighted avg	0.8456	0.8063	0.8006	160

รูปที่ 4.1 Classification Report บนผลข้อมูลชุด Test Set

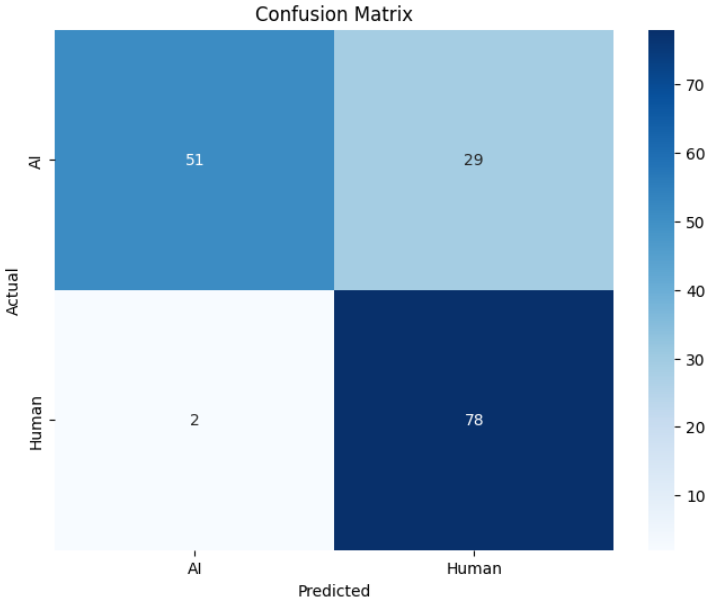
รูปที่ 4.1 แสดงประสิทธิภาพของตัวแบบการจำแนกเสียงซึ่งมีหน้าที่ในการแยกเสียงระหว่างเสียงมนุษย์กับเสียงที่ถูกสร้างขึ้นโดยเครื่อง โดยตัวแบบที่เลือกนำมาวัดประสิทธิภาพเป็นตัวแบบที่พัฒนาเป็นตัวแบบสุดท้าย หรือ ตัวแบบที่ได้รับการจูนพารามิเตอร์และมีความแม่นยำในทำนายผลที่ดีที่สุดบนข้อมูล Test set มาตรวัดสำหรับความแม่นยำ (Precision) สำหรับการจำแนกเสียงที่สร้างโดยเครื่องเท่ากับ 0.9623 และมีความแม่นยำในการทำนายเสียงมนุษย์มีค่าเท่ากับ 0.7290

จากมาตรวัด Recall ค่า 0.6375 ซึ่งถึง 63.75% ของเสียงที่สร้างโดยเครื่องทั้งหมดถูกตัวแบบระบุได้อย่างถูกต้อง และตัวแบบสามารถระบุเสียงของมนุษย์ได้ดีกว่า ด้วย Recall เท่ากับ 0.9750

ข้อมูลใน Test Set ประกอบด้วยข้อมูลเสียงมนุษย์และข้อมูลเสียงที่สร้างโดยเครื่องจำนวนประเภทละ 80 ตัวอย่าง ในภาพรวมตัวแบบจำแนกเสียงได้ถูกต้อง

อย่างไรก็ตาม การที่ตัวแบบสามารถระบุเสียงที่สร้างโดยเครื่องได้ดีกว่า (ค่า Recall ของเสียงมนุษย์สูงกว่า) เมื่อเปรียบเทียบกับเสียงมนุษย์ โดยเสียงเครื่องถูกจำแนกเป็นเสียงมนุษย์ ซึ่งอาจนำไปสู่ความท้าทายในการประยุกต์ใช้จริงในการแยกแยะระหว่างเสียงมนุษย์กับเสียงเครื่อง

แม้ว่าตัวแบบจะทำงานได้อย่างมีประสิทธิผล แต่ยังมีพื้นที่สำหรับการปรับปรุงเพิ่มเติม โดยเฉพาะในการเพิ่มความแม่นยำสำหรับการตรวจจับเสียงที่สร้างโดยเครื่อง การพัฒนาในอนาคต อาจปรับปรุงโดยให้ชุดข้อมูลการฝึกหลากหลายมากขึ้น การปรับปรุงวิธีการสกัดคุณลักษณะ หรือการสำรวจโครงสร้างตัวแบบขั้นสูงเพื่อปรับปรุงความแม่นยำ โดยไม่ทำให้ Recall ลดลง



รูปที่ 4.2 Confusion Matrix บนผลข้อมูลชุด Test Set

Confusion Matrix จากรูป 4.2 บนข้อมูล Test Set แสดงประสิทธิภาพของตัวแบบการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยเครื่อง ตัวแบบสามารถจำแนกเสียงเป็นเสียงของมนุษย์ได้ 107 เสียง โดยสามารถจำแนกได้ถูกต้อง 78 เสียง และจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์ 29 เสียง ซึ่งสามารถคำนวนความแม่นยำของตัวแบบจำแนกเสียงของมนุษย์ ตัวแบบสามารถจำแนกเสียงของมนุษย์ได้ถูกเป็นจำนวนมากกว่าเมื่อเปรียบเทียบกับการจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์

การจำแนกเสียงที่สร้างโดยเครื่องของตัวแบบ สามารถจำแนกเสียงที่สร้างโดยเครื่องได้ทั้งหมด 53 เสียงโดยสามารถจำแนกเสียงเป็นเสียงที่สร้างโดยเครื่องได้ถูกต้อง 51 เสียง และเสียงมนุษย์ที่ตัวแบบทำนายเป็นเสียงที่สร้างโดยเครื่อง 2 เสียง ซึ่งแสดงให้เห็นถึงความสามารถของตัวแบบในการทำนายเสียงเครื่องได้อย่างถูกต้อง

โดยสรุปแล้วจะสามารถสังเกตเห็นได้ว่าผลของการจำแนกตัวอย่างเสียงชุดทดสอบบน Test Set สามารถจำแนกเสียงของมนุษย์ได้ถูกต้องมากที่สุด นอกจากนี้ เมื่อพิจารณาถึงความสามารถการจำแนกเสียงของตัวแบบจะพบว่า ตัวแบบมีความสามารถในการจำแนกเสียงที่เป็นเสียงของมนุษย์ได้มีความถูกต้องมากกว่าเสียงที่สร้างโดยเครื่อง ซึ่งสาเหตุอาจมาจากจังหวะการพูด ของมนุษย์อาจมีจังหวะการออกเสียงที่

หลากหลาย ขึ้นอยู่กับปัจจัยของการออกเสียงในแต่ละผู้พูด เช่น อารมณ์ของผู้พูดในขณะพูด ลักษณะของเสียงวรรณยุคในภาษาไทย เป็นต้น ซึ่งเสียงที่สร้างโดยเครื่องอาจจะยังไม่สามารถสร้างให้มีความใกล้เคียงได้

4.2. ผลการจำแนกของตัวแบบบนผลข้อมูลชุดทดสอบของเสียงที่สร้างโดยเครื่องและเสียงของมนุษย์เพศชาย

ผลการจำแนกเสียงชุดทดสอบจากการตัวแบบที่พัฒนาตัวแบบสุดท้าย หรือ ตัวแบบที่ถูกปรับค่าพารามิเตอร์บนชุดข้อมูลทดสอบที่มีแต่ข้อมูลเสียงของเพศชายได้ผลดังนี้

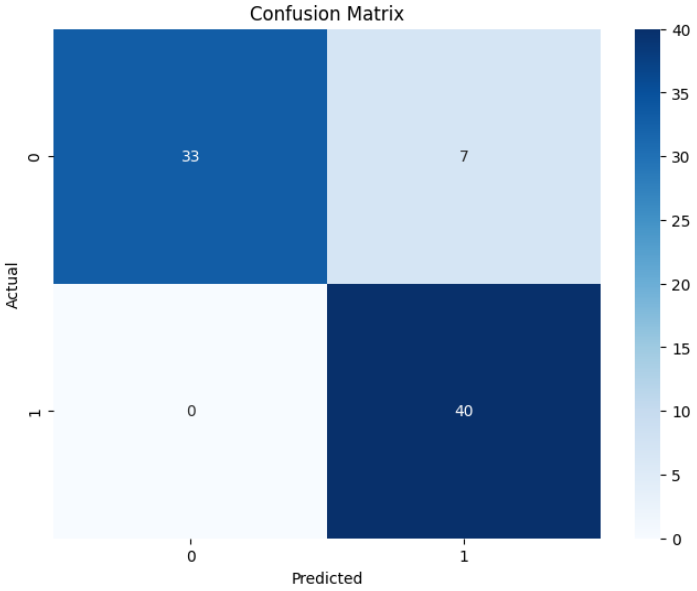
	precision	recall	f1-score	support
AI	1.0000	0.8250	0.9041	40
Human	0.8511	1.0000	0.9195	40
accuracy			0.9125	80
macro avg	0.9255	0.9125	0.9118	80
weighted avg	0.9255	0.9125	0.9118	80

รูปที่ 4.3 Classification Report บนผลข้อมูลชุด Test Set เพศชาย

จากผลการจำแนกเสียงของเสียงชุดทดสอบเพศชายในภาพที่ 4.3 พบว่าตัวแบบสามารถจำแนกเสียงโดยมี Accuracy Score โดยรวมเท่ากับ 0.9125 หรือ 91.25% และมีค่าความแม่นยำที่ได้จากมาตรวัดสำหรับความแม่นยำ (Precision) สำหรับการจำแนกเสียงมนุษย์อยู่ที่ 85.11% และมีความแม่นยำในการทำนายเสียงที่ถูกสร้างขึ้นโดยเครื่องเท่ากับ 100% ซึ่งตัวแบบสามารถจำแนกเสียงที่สร้างโดยเครื่องเพศชายในชุดทดสอบได้ถูกต้อง โดยที่ไม่มีความผิดพลาดในการจำแนกเสียงของมนุษย์เป็นเสียงที่สร้างโดยเครื่องเลย

จากมาตรวัด Recall ของการจำแนกเสียงของมนุษย์เพศชายมีค่าเท่ากับ 1.000 ซึ่งคิดเป็น 100% ของเสียงมนุษย์ทั้งหมดถูกตัวแบบระบุได้ถูกต้อง เนื่องจากตัวแบบไม่มีความผิดพลาดในการจำแนกเสียงมนุษย์เป็นเสียงที่สร้างโดยเครื่อง ในทางตรงกันข้ามตัวแบบสามารถระบุเสียงที่สร้าง โดยเครื่องได้ด้วยค่า Recall เท่ากับ 0.8250 หรือ 82.5%

ข้อมูลใน Test Set เพศชายชุดนี้ ประกอบด้วยข้อมูลเสียงมนุษย์และข้อมูลเสียงที่สร้างโดยเครื่อง เฉพาะเพศชาย จำนวนประเภทละ 40 ตัวอย่าง และในภาพรวมของการจำแนกเสียงตัวแบบสามารถจำแนกเสียงได้ถูกต้อง



รูปที่ 4.4 Confusion Matrix บนผลข้อมูลชุด Test Set เพศชาย

Confusion Matrix จากรูป 4.4 บนข้อมูล Test Set เพศชาย แสดงประสิทธิภาพของตัวแบบการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยเครื่อง ตัวแบบสามารถจำแนกเสียงเป็นเสียงของมนุษย์ได้ 47 เสียง โดยสามารถจำแนกได้ถูกต้อง 40 เสียง และจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์ 7 เสียง ตัวแบบสามารถจำแนกเสียงของมนุษย์ได้ถูกเป็นจำนวนมากกว่าเมื่อเปรียบเทียบกับการจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์ และ ตัวแบบสามารถจำแนกเสียงของมนุษย์ได้ถูกต้องทั้งหมดทุกเสียงในข้อมูลชุดทดสอบเพศชาย

การจำแนกเสียงที่สร้างโดยเครื่องของตัวแบบ สามารถจำแนกเสียงที่สร้างโดยเครื่องได้ทั้งหมด 33 เสียงโดยสามารถจำแนกเสียงเป็นเสียงที่สร้างโดยเครื่องได้ถูกต้องทั้งหมด ซึ่งแสดงให้เห็นถึงความสามารถของตัวแบบในการทำนายเสียงเครื่องได้อย่างถูกต้อง

โดยสรุปสามารถสังเกตเห็นได้ว่าตัวแบบสามารถจำแนกเสียงของมนุษย์ได้ถูกต้องทั้งหมดในข้อมูลชุดทดสอบ และไม่มีการจำแนกเสียงที่ผิดพลาดในการจำแนกเสียงของมนุษย์เป็นเสียงที่สร้างโดยเครื่องเลย

4.3. ผลการจำแนกของตัวแบบบนผลข้อมูลชุดทดสอบของเสียงที่สร้างโดยเครื่องและเสียงของมนุษย์เพศหญิง

ผลการจำแนกเสียงชุดทดสอบจากการตัวแบบที่พัฒนาตัวแบบสุดท้าย หรือ ตัวแบบที่ถูกปรับค่าพารามิเตอร์บนชุดข้อมูลทดสอบที่มีแต่ข้อมูลเสียงของเพศหญิงได้ผลดังนี้

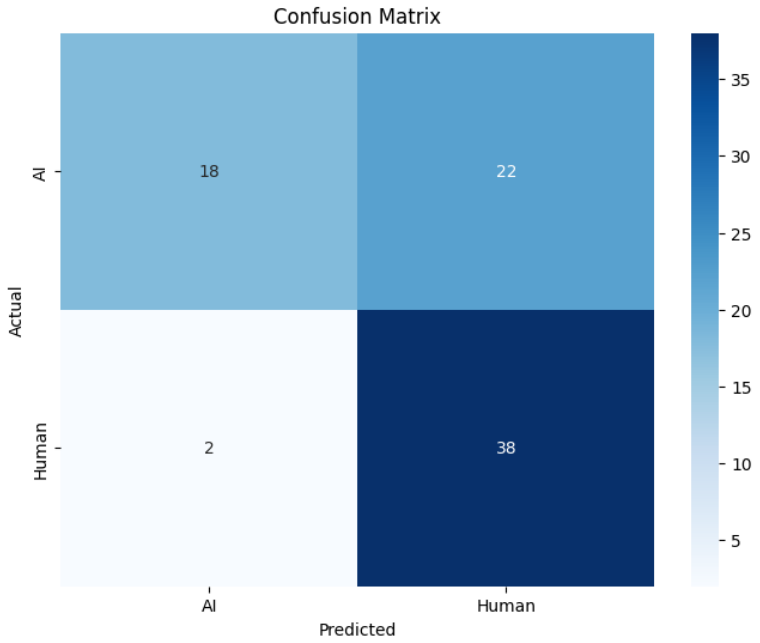
	precision	recall	f1-score	support
AI	0.9000	0.4500	0.6000	40
Human	0.6333	0.9500	0.7600	40
accuracy			0.7000	80
macro avg	0.7667	0.7000	0.6800	80
weighted avg	0.7667	0.7000	0.6800	80

รูปที่ 4.5 Classification Report บนผลข้อมูลชุด Test Set เพศหญิง

จากผลการจำแนกเสียงของเสียงชุดทดสอบเพศหญิงในภาพที่ 4.5 พบว่าตัวแบบสามารถจำแนกเสียงโดยมี Accuracy Score โดยรวมเท่ากับ 0.7000 หรือ 70% เมื่อพิจารณาแยกตามคลาสจะพบว่า คลาสที่มีค่าความแม่นยำที่ได้จากมาตรวัดสูงที่สุดคือคลาสของข้อมูลเสียงที่สร้างโดยเครื่อง ความแม่นยำ สำหรับการจำแนกเสียงที่สร้างโดยเครื่องมีค่าเท่ากับ 0.9000 และมีความแม่นยำในการ ทำนายเสียงของมนุษย์เท่ากับ 0.6333 ซึ่งตัวแบบมีความแม่นยำในการจำแนกเสียงเพศหญิงที่สร้างโดยเครื่องในชุดทดสอบมากกว่าเสียงมนุษย์

จากมาตรวัด Recall ของการจำแนกเสียงของมนุษย์เพศหญิงมีค่าเท่ากับ 0.9500 ซึ่งคิดเป็น 95% ของเสียงมนุษย์ทั้งหมดถูกตัวแบบระบุได้ถูกต้อง และตัวแบบสามารถระบุเสียงเพศหญิงที่สร้างโดยเครื่องได้ด้วยค่า Recall เท่ากับ 0.4500

ข้อมูลใน Test Set เพศหญิงชุดนี้ ประกอบด้วยข้อมูลเสียงมนุษย์และข้อมูลเสียงที่สร้างโดยเครื่องเฉพาะเพศหญิง จำนวนประเภทละ 40 ตัวอย่าง และในภาพรวมของการจำแนกเสียงตัวแบบสามารถจำแนกเสียงได้ถูกต้อง



รูปที่ 4.6 Confusion Matrix บนผลข้อมูลชุด Test Set เพศหญิง

Confusion Matrix จากรูป 4.6 บนข้อมูล Test Set เพศหญิง แสดงประสิทธิภาพของตัวแบบการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยเครื่อง ตัวแบบสามารถจำแนกเสียงเป็นเสียงของมนุษย์ได้ 60 เสียง โดยสามารถจำแนกได้ถูกต้อง 38 เสียง และจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์ 22 เสียง ตัวแบบสามารถจำแนกเสียงของมนุษย์เพศหญิงได้ถูกเป็นจำนวนมากกว่าเมื่อเปรียบเทียบกับการจำแนกเสียงที่สร้างโดยเครื่องผิดพลาดเป็นเสียงของมนุษย์เพศหญิง

การจำแนกเสียงที่สร้างโดยเครื่องของตัวแบบ สามารถจำแนกเสียงที่สร้างโดยเครื่องได้ทั้งหมด 20 เสียงโดยสามารถจำแนกเสียงที่สร้างโดยเครื่องได้ถูกต้อง 18 เสียง และเสียงมนุษย์ที่ตัวแบบทำนายเป็นเสียงที่สร้างโดยเครื่อง 2 เสียง ซึ่งแสดงให้เห็นถึงความสามารถของตัวแบบในการทำนายเสียงเครื่องได้อย่างถูกต้อง

โดยสรุปแล้วจะเห็นได้ว่าเพศเป็นอีกหนึ่งตัวแปรที่มีผลกับการจำแนกเสียงของตัวแบบ โดยประเด็นที่เหมือนกันในทั้งสองเพศ คือ ความแม่นยำในการจำแนกเสียงที่สร้างโดยเครื่อง ที่ตัวแบบสามารถจำแนกได้ดีกว่าเสียงของมนุษย์ และสิ่งที่แตกต่างกันคือ เพศชายมีค่า Accuracy Score และ Recall โดยรวมสูงกว่าเพศหญิง เนื่องจากตัวแบบสามารถจำแนกเสียงบนข้อมูลเสียงเพศชายได้ถูกต้องมากกว่า และ จากมาตรวัดต่าง ๆ สามารถสังเกตได้ว่าตัวแบบมีความสามารถโดยรวมในการจำแนกข้อมูลเสียงในข้อมูลชุดทดสอบเพศชายสูงกว่าเพศหญิง แต่เนื่องจากความเหมือนและความแตกต่างของทั้งสอบเพศในการศึกษานี้อาจยังถูกต้องและชัดเจนมากเพียงพอ จากข้อมูลที่นำมาพัฒนาและทดสอบตัวแบบในการศึกษานี้มีจำนวนที่น้อยมากจึงทำให้เมื่อมีความแตกต่างเพียงเล็กน้อยของการจำแนกเสียงของตัวแบบ การวัดค่าประสิทธิภาพก็จะเปลี่ยนแปลงไปมาก

ประสิทธิภาพของตัวแบบมีค่าความแม่นยำโดยรวมอยู่ที่ 0.8062 ซึ่งทำให้ตัวแบบนี้สามารถนำไปต่อยอดเพื่อพัฒนา หรือ นำไปใช้งานในระบบงานอื่น โดยอาจนำไปพัฒนาต่อยอดเป็น ระบบการคัดกรองเสียงจากในการตรวจสอบความถูกต้องของข้อมูลเสียงของผู้ใช้ในการเข้าถึงบริการออนไลน์หรือใช้ในการตรวจสอบการเข้าถึงระบบโดยมนุษย์

บทที่ 5

สรุปผลการศึกษาและข้อเสนอแนะ

5.1 สรุปผลการศึกษาและการนำไปใช้

5.1.1 สรุปผลการศึกษา

จากการศึกษาและดำเนินโครงงานนี้ผู้จัดทำได้บรรลุวัตถุประสงค์ที่ระบุไว้ข้างต้น ซึ่งสามารถสรุปผลการศึกษาตามวัตถุประสงค์แต่ละข้อได้ดังนี้

1. เพื่อศึกษาหลักการทำงานของตัวแบบกับข้อมูลเสียง

   ในการศึกษาเพื่อบรรลุวัตถุประสงค์ในข้อนี้ผู้จัดทำได้ศึกษาวิธีการ, งานวิจัยและเครื่องมือที่เกี่ยวข้องกับการทำงานของตัวแบบกับข้อมูลเสียง เพื่อให้ทราบข้อมูลเกี่ยวกับวิธีการ เครื่องมือและไลบรารี (Library) ที่สำคัญอันสามารถนำมาประยุกต์ใช้ในการศึกษาและดำเนินโครงงานนี้ได้ประกอบด้วย ความเข้าใจเกี่ยวกับข้อมูลเสียงซึ่งผู้ศึกษาได้ศึกษานิยามและองค์ประกอบของคลื่นเสียง วิธีการออกแบบและรวบรวมข้อมูลสำหรับการพัฒนาตัวแบบ ซึ่งต้องมองเห็นภาพรวมของการพัฒนาตัวแบบจึงสามารถออกแบบการเก็บข้อมูล กระบวนการเตรียมข้อมูลประเภทเสียงให้สามารถนำมาใช้ ในกระบวนการพัฒนาตัวแบบได้อย่างมีประสิทธิภาพ (Audio Preprocessing) สำหรับข้อมูลเสียงจะต้องมีการควบคุมความยาวของข้อมูลเสียง จำนวนข้อมูลเสียงและความหลากหลายของข้อมูลเสียง การตัดทอนเสียงให้มีระยะเวลาที่สม่ำเสมอกัน หรือการเพิ่มระยะเวลาของข้อมูลเสียงเพื่อให้ได้ระยะความยาวของเสียงที่ต้องการ จากนั้นจึงทำการสร้างข้อมูลเสียง เพิ่มเป็นจำนวน 400% จากข้อมูลเดิม (Audio Augmentation) ด้วยวิธีการปรับความสูง-ต่ำของเสียง (Shift Pitch) และการปรับความเร็ว-ช้าในเสียงพูด (Time Stretch)

2. เพื่อศึกษาหลักการและวิธีการพัฒนาตัวแบบด้วยการใช้ตัวแบบ Convolution Neural Network

   ในการศึกษาเพื่อบรรลุวัตถุประสงค์ในข้อนี้ผู้จัดทำได้ศึกษาวิธีการ, งานวิจัยและเครื่องมือที่เกี่ยวข้อง กับหลักการทำงานและวิธีการพัฒนาตัวแบบ Convolution Neural Network เพื่อให้เข้าใจเกี่ยวกับขั้นตอนในการพัฒนาตัวแบบ Convolution Neural Network, เทคนิคโครงข่ายประสาทเทียมแบบ Convolution Neural Network ที่นิยมใช้กับข้อมูลเสียง

3. เพื่อพัฒนาตัวแบบในการจำแนกเสียงมนุษย์และเสียงที่สร้างขึ้นโดยเครื่อง

การศึกษาและพัฒนาตัวแบบในโครงงานนี้ผู้จัดทำได้ใช้ภาษาไพธอน (Python) สำหรับการเขียนคำสั่ง วิเคราะห์ข้อมูลและพัฒนาตัวแบบ โดยได้ศึกษาและนำไพธอนไลบรารีต่างๆ ที่ช่วยในกระบวนการเตรียมข้อมูล สำหรับการประมวลภาษาธรรมชาติไปจนถึงการพัฒนาตัวแบบ ประกอบด้วยไพธอนไลบรารีที่สำคัญ ได้แก่ ไลบรารี Librosa ที่ใช้ในการจัดการกับข้อมูลเสียง เช่น การสร้าง Mel spectrogram, การปรับ Sampling Rate รวมถึงการทำ Audio Augmentation แบบต่างๆ และไลบรารีใช้ในการสร้างโครงข่ายประสาทเทียมแบบลึก (Deep Learning) อย่าง Keras และ Tensorflow

ผู้จัดทำได้พัฒนาตัวแบบการจำแนกเสียงที่สร้างขึ้นโดยเครื่องจากชุดข้อมูลที่ผู้ศึกษาเก็บเป็นภาษาไทย โดยการแบ่งข้อมูลออกเป็น 3 ชุด ได้แก่ Training Set, Validation Set, Test Set แล้วทำการพัฒนาตัว แบบโดย Training Set ซึ่งประกอบด้วยข้อมูลเสียงที่สร้างโดยเครื่องและเสียงอ่านของมนุษย์ ด้วยโครง สร้างตัวแบบคอนโวลูชั่น (Convolution Neural Network) แล้วปรับจูนด้วย Baysain Optimizer ผลของการทดสอบบน Test Set ได้ค่า Accuracy 0.7362 นอกจากนี้เมื่อพิจารณาถึงพฤติกรรมการทำนาย ตัวแบบจะพบว่าตัวแบบมีความสามารถในการทำนายข้อมูลที่เป็นข้อมูลเสียงของมนุษย์ได้มีความถูก ต้องมากกว่าเสียงที่สร้างโดยเครื่อง

5.1.2 การนำไปใช้

1. สามารถนำองค์ความรู้และวิธีการไปประยุกต์ใช้สำหรับการต่อยอดธุรกิจและการศึกษาส่วน บุคคล โดยนำองค์ความรู้นี้มาเป็นแนวทางหนึ่งในการวิเคราะห์ข้อมูลหรือพัฒนาตัวแบบ ด้วยข้อมูลเสียง

2. สามารถนำแนวทางไปศึกษาต่อยอดเพื่อประยุกต์ในการและการตรวจจับการฉ้อโกง ในธุรกิจธนาคาร การเงิน และ การสื่อสารที่มีความปลอดภัยการแยกแยะระหว่าง เสียงของมนุษย์และเสียงที่สร้างขึ้นโดยเครื่องสามารถเสริมสร้างมาตรการความปลอดภัย โดยป้องกันการปลอมเสียงและการโจมตีด้วยเสียงลึกลับ ตรวจสอบคำสั่งเสียงหรือขั้นตอนการตรวจสอบสิทธิ์ทำโดยบุคคลจริง องค์กรสามารถลดความเสี่ยงของการฉ้อโกง ได้อย่างมีนัยสำคัญ

3. การบังคับใช้ลิขสิทธิ์และสิทธิ์ในทรัพย์สินทางปัญญา สำหรับผู้สร้าง บริษัทสื่อ และผู้ถือสิทธิ์ลิขสิทธิ์ การแยกแยะระหว่างเสียงมนุษย์และเสียงที่สร้างขึ้นโดย เครื่อง มีความสำคัญอย่างยิ่งในการป้องกันทรัพย์สินทางปัญญา ซึ่งเทคโนโลยีนี้สามารถช่วย

ในการระบุการใช้ปัญญาประดิษฐ์ อย่างไม่ได้รับอนุญาตเพื่อทำซ้ำเสียงของบุคคลหรือสร้างเนื้อหาใหม่ที่ละเมิดลิขสิทธิ์ของผู้อื่นโดยไม่ได้รับอนุญาต

5.2 ปัญหาและวิธีการแก้ไข

1. Environment ของอุปกรณ์ผู้ศึกษามีความไม่สอดคล้องกับไลบรารีจึงต้องทำการติดตั้งไลบรารีหรือแพคเกจ เสริมอื่นๆให้เหมาะกับการทำโครงงาน

2. มีข้อจำกัดในด้านอุปกรณ์การทำโครงงานของผู้ศึกษา โดยเมื่อมีการคำนวณในขั้นตอนการสร้างตัวแบบ หรือ ในขั้นตอนการ Tuning Hyperparameter อุปกรณ์ของผู้ศึกษาอาจไม่เหมาะสมสำหรับการทำงานที่ต้องการความสามารถในการคำนวนสูง และปริมาณของข้อมูลที่สามารถเก็บเอาไว้ในอุปกรณ์มีจำกัด ส่งผลต่อการเก็บตัวอย่างของข้อมูลเสียงที่อาจเก็บได้น้อยลง วิธีการแก้ไขคือการนำการประมวลผลและที่เก็บข้อมูลไปดำเนินการบน Cloud หรือ Virtual Machine

3. มีปัญหาในส่วนการเก็บข้อมูลเสียงที่สร้างขึ้นโดยเครื่องและเสียงที่อ่านข้อความของมนุษย์เนื่องจากคุณภาพของเสียงที่บันทึกจากเครื่องมือบันทึกเสียงที่ผู้ศึกษามีปัจจุบัน สามารถบันทึกเสียงที่ถูกจำกัดคุณภาพไว้เพียง 16000Hz ซึ่งเสียงที่สร้าง โดยเครื่องมีคุณภาพ 22500Hz จากความแตกต่างของคุณภาพเสียงที่มีอยู่ทำให้ผลลัพธ์จากการสร้าง Spectrogram มีความแตกต่างกัน จึงอาจจะมีผลระหว่างการเรียนรู้รวมถึงผลลัพธ์การเรียนรู้ของตัวเครื่อง มีแนวทางแก้ไขคือ ทดลองลดคุณภาพของเสียงที่สร้างโดยเครื่องจาก 22500Hz ให้มีคุณภาพเสียงเพียง 16000Hz เพื่อให้คุณภาพเท่ากับเสียงอ่านของมนุษย์ แต่จะเสียคุณภาพของเสียงไป ในโครงงานนี้ จึงเลือกใช้เสียงของเครื่องและมนุษย์ที่มีความแตกต่างกัน

4. ปัญหาในเรื่องความหลากหลายของแหล่งที่มาของเสียงที่เครื่องสร้าง จากการทดลองสร้างเสียงจากแหล่งข้อมูลเสียง ที่สร้างโดยเครื่องที่แตกต่างกันพบว่า มีแหล่งที่มาของเสียงจำนวนน้อยที่คุณภาพของเสียงภาษาไทยที่สร้างมีความคล้ายกับเสียงที่อ่านโดยมนุษย์

5. มีปัญหาในส่วนการเก็บข้อมูลเสียงที่ถูกสร้างขึ้นโดยเครื่อง เนื่องจากระบบที่สามารถสร้างเสียงภาษาไทยที่มีคุณภาพมีน้อย อีกทั้งมีค่าใช้จ่ายในการใช้บริการ ผู้จัดทำแก้ปัญหาด้วยการหาแหล่งข้อมูลที่ไม่มีค่าใช้จ่ายมาทดแทน เพื่อจะลดค่าใช้จ่าย

5.3 ข้อเสนอแนะ

1. ในโครงงานนี้มีข้อจำกัดในเรื่องของข้อมูลที่ใช้สำหรับพัฒนาตัวแบบ ซึ่งมีผลต่อความสามารถของตัวแบบที่พัฒนาเนื่องจากข้อจำกัดของข้อมูล ข้อจำกัดของข้อมูลของโครงงานมีหลายประเด็นที่สามารถจะปรับปรุงได้ เช่น ความถี่ของเสียงที่เครื่องสร้างกับข้อมูลเสียงของมนุษย์ที่ไม่เท่ากัน ซึ่งส่งผลต่อการสร้าง Spectrogram ถ้าสามารถหาอุปกรณ์ที่ดีที่สามารถเก็บข้อมูลเพื่อได้ คุณภาพของข้อมูลที่ดีกว่า ก็จะได้ผลลัพท์ของการพัมนาตัวแบบที่ดีกว่า หรือความหลากหลายของอายุของกลุ่มตัวอย่างเสียง เพื่อความหลากหลายและง่ายต่อการนำตัวแบบที่พัฒนาไปใช้

2. มีปัญหาในการจำแนกเสียงของตัวแบบ สามารถจะปรับปรุงได้โดยการปรับ threshold ในการจำแนกว่าเสียงคนหรือเสียงเครื่อง ไม่จำเป็นต้องตั่งค่า threshold ไว้เท่ากับ 0.5 ตลอดเพราะเมื่อตัวแบบทำนายผิดส่วนมากจะจำแนกเสียงมนุษย์ว่าเป็นเสียงเครื่อง สามารถปรับ threshold ลงเพื่อถ้าหาค่า threshold ให้เหมาะสมและสามารถได้ผลลัพธ์ในการจำแนกตัวแบบที่ดีกว่า

บรรณานุกรม

1. mlearnere. Learning from Audio: The Mel Scale, Mel Spectrograms, and Mel Frequency Cepstral Coefficients [Online]. 2021. Available from: https://towardsdatascience.com/learningfrom-audio-the-mel-scale-mel-spectrograms-and-mel-frequency-cepstral-coefficientsf5752b6324a8 [2023, December]

2. Ketan Doshi. Audio Deep Learning Made Simple (Part 1): State-of-the-Art Techniques (2021). [Online]. 2021 Available from: https://towardsdatascience.com/audio-deep-learning-madesimple-part-1-state-of-the-art-techniques-da1d3dff2504 [2023, December]

3. Ketan Doshi. Audio Deep Learning Made Simple (Part 2): Why Mel Spectrograms perform better (2021). [Online]. 2021 Available from : https://towardsdatascience.com/audio-deep-learningmade-simple-part-2-why-mel-spectrograms-perform-better-aad889a93505 [2023, December]

4. Ketan Doshi. Audio Deep Learning Made Simple (Part 3): Data Preparation and Augmentation (2564). [Online]. 2021 Available from : https://towardsdatascience.com/audio-deeplearning-made-simple-part-3-data-preparation-and-augmentation-24c6e1f6b52 [2023, December]

5. Ketan Doshi. Audio Deep Learning Made Simple (Part 4): Audio Deep Learning Made Simple: Sound Classification, Step-by-Step (2021). [Online]. 2021 Available from : https://towardsdatascience.com/audio-deep-learning-made-simple-sound-classification-stepby-step-cebc936bbe5 [2023, December]

6. ภณฐานีย์. (2565). การจำแนกกลุ่มอายุจากเสียงผู้พูดด้วยการใช้โครงข่ายประสาทเทียมแบบลึก (รายงานผลการวิจัย).กรุงเทพฯ: จุฬาลงกรณ์มหาวิทยาลัย.

7. Natthawat Phongchit. Convolutional Neural Network (CNN) คืออะไร [ออนไลน์]. 2558. แหล่งที่มา: https://medium.com/@natthawatphongchit/มาลองดูวิธีการคิดของ-cnn-กันe3f5d73eebaa [2567, มกราคม]

8. Wilfried Schaffner. What is speech recognition and How does speech recognition work [Online]. 2021. Available from : https://www.techtarget.com/searchcustomerexperience/definition/speech-recognition [2024, January]

9. Gaffar Shakhadri. Guide to Audio Classification Using Deep Learning. [Online]. 2023 Available from : https://www.analyticsvidhya.com/blog/2022/04/guide-to-audio-classification-using-deep-learning/ [2023, December]

10. David Oluyle. Audio Classification using Deep Learning and TensorFlow: A Step-by-Step Guide. [Online]. 2023 Available from : https://medium.com/@oluyaled/audio-classification-using-deep-learning-and-tensorflow-a-step-by-step-guide-5327467ee9ab [2023, November]

11. seth814. Audio-Classification (Kapre Version). [Online]. 2018 Available from : https://github.com/seth814/Audio-Classification [2024, February]

12. Olarik Surinta. โครงข่ายประสาทเทียมแบบคอนโวลูชัน [ออนไลน์]. 2563. แหล่งที่มา: https://medium.com/olarik/โครงข่ายประสาทเทียมแบบคอนโวลูชัน [2567, มกราคม]

13. JalFazy Shaikh. Getting Started with Audio Data Analysis using Deep Learning. [Online]. 2021 Available from : https://www.analyticsvidhya.com/blog/2017/08/audio-voice-processing-deep-learning/ [2023, December]

14. Rendyk. Tuning the Hyperparameters and Layers of Neural Network Deep Learning. [Online]. 2024 Available from : https://www.analyticsvidhya.com/blog/2021/05/tuning-the-hyperparameters-and-layers-of-neural-network-deep-learning/ [2024, February

ภาคผนวก

ตัวอย่างคำสั่งเปลี่ยนชนิตของข้อมูลเสียงจาก .m4a เป็น .wav

```
def convertMp4ToWav(inputFile, outputFile):
    Convertor = [
        "ffmpeg",
        "-i",
        inputFile,
        "-vn",
        "-acodec",
        "pcm_s16le",
        "-ar",
        "22500",
        outputFile,
    ]
    try:
        subprocess.run(Convertor, check=True)
        print("Successfully converted!!")
    except subprocess.CalledProcessError as e:
        print("Conversion failed!")

for i, filename in enumerate(os.listdir(".")):
    actualFilename = filename[:-4]
    if filename.endswith(".m4a"):
        convertMp4ToWav(filename, actualFilename + ".wav")
```

ตัวอย่างคำสั่งตัดไฟล์เสียงที่ยาวกว่า 2 วินาที ออกจากข้อมูลเสียง

```
import os
from pydub import AudioSegment
import librosa

os.makedirs(output_dir, exist_ok=True)  # Create the directory if it
doesn't exist

for filename in os.listdir(directory):
    if filename.endswith(".wav"):  # Check for supported audio formats
        full_path = os.path.join(directory, filename)
```

```
        sound = AudioSegment.from_file(full_path, format="wav")

        two_seconds = sound[:2000]

        output_filename = f"{filename}"
        output_path = os.path.join(output_dir, output_filename)

        two_seconds.export(output_path, format="wav")

        print(f"Trimmed {filename} and saved as {output_filename}")
```

ตัวอย่างคำสั่งตรวจสอบความยาวของสียงหลังจากตัดเสียงที่ยาวเกินกว่าที่ต้องการออก

```
for filename in os.listdir(directory):
    if filename.endswith(".wav"):
        audio_file_path = os.path.join(file_path, filename)
        try:
            y, sr = librosa.load(audio_file_path)

            duration = librosa.get_duration(y=y, sr=sr)

            print(f"The duration of the audio file {filename} is:
{duration:.2f} seconds")

        except Exception as e:
            print(f"Error processing {filename}: {e}")
```

ตัวอย่างคำสั่งเติมข้อมูลเสียงที่มีความยาวของเสียงไม่ถึง 2 วินาที ด้วยเสียงว่าง และตรวจสอบความถูกต้อง

```
for filename in os.listdir(directory):
    if filename.endswith(".wav"):
        full_path = os.path.join(directory, filename)

        sound = AudioSegment.from_file(full_path)

        print(f"Original duration of {filename}: {len(sound)}
milliseconds")
```

```
        if len(sound) < desired_duration:
            duration_to_add = desired_duration - len(sound)

            silence = AudioSegment.silent(duration=duration_to_add)
            elongated_sound = sound + silence

            elongated_sound.export(full_path, format="wav")

            print(f"Elongated {filename} and saved in place")
        else:
            print(f"{filename} is already long to 2 seconds.")

for filename in os.listdir(directory):
    if filename.endswith(".wav"):
        file_path = os.path.join(directory, filename)
        try:
            y, sr = librosa.load(file_path)

            duration = librosa.get_duration(y=y, sr=sr)

            print(f"The duration of the audio file {filename} is:
{duration:.2f} seconds")

        except Exception as e:
            print(f"Error processing {filename}: {e}")
```

ตัวอย่างคำสั่งในการสร้าง Mel Spectrogram ตามข้อมูลเสียง โดยกำหนดค่าพารามิเตอร์เพื่อให้ได้คุณภาพของภาพสูงที่สุด

```
def audio_to_spectrogram(audio_path, output_folder):

    y, sr = librosa.load(audio_path, sr=None, duration=2.0)

    n_fft = 1024
    hop_length = 64

    # Generate a Mel-scaled spectrogram
    S = librosa.feature.melspectrogram(y=y, sr=sr, n_fft=n_fft,
hop_length=hop_length, n_mels=256, fmax=sr/2)
```

```
    S_DB = librosa.power_to_db(S, ref=np.max)

    plt.figure(figsize=(12, 8))
    librosa.display.specshow(S_DB, sr=sr, hop_length=hop_length,
x_axis='time', y_axis='mel', fmax=sr/2)
    plt.axis('off')  # Removes the axis to have only the spectrogram

    plt.tight_layout(pad=0)

    base_filename = os.path.splitext(os.path.basename(audio_path))[0]
    output_filename = f"{base_filename}.png"
    output_path = os.path.join(output_folder, output_filename)

    plt.savefig(output_path, dpi=150, bbox_inches='tight', pad_inches=0)
    plt.close()
```

ตัวอย่างคำสั่งเรียก Library สำหรับในการพัฒนาตัวแบบ

```
from matplotlib import pyplot as plt
import tensorflow as tf
import os

from PIL import Image
from tensorflow import keras
from tensorflow.keras import Sequential, Input, Model
from tensorflow.keras.layers import Dense, Flatten, Softmax, Dropout,
Conv2D, MaxPooling2D, BatchNormalization, Activation, AveragePooling2D
from tensorflow.keras.layers import RandomFlip, RandomRotation, RandomZoom,
Rescaling, RandomWidth, RandomHeight, Lambda
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint,
ReduceLROnPlateau, LearningRateScheduler
from tensorflow.keras.utils import image_dataset_from_directory
from tensorflow.keras.optimizers import RMSprop, Adam, SGD
from tensorflow.keras.regularizers import l2
from sklearn.metrics import roc_curve, auc
from tensorflow.keras.layers import Dense
from tensorflow.keras.models import Sequential
```

ตัวอย่างคำสั่งสร้างแฟ้มข้อมูลสำหรับเก็บ Spectrogram แยกประเภท

```
import shutil
os.mkdir(baseDir)

trainAIDir = os.path.join(baseDir, 'train')
os.mkdir(trainDir)

validationDir = os.path.join(baseDir, 'validation')
os.mkdir(validationDir)

testDir = os.path.join(baseDir, 'test')
os.mkdir(testDir)

trainHumanDir = os.path.join(trainDir, 'Human')
os.mkdir(trainHumanDir)

trainDir = os.path.join(trainDir, 'AI')
os.mkdir(trainAiDir)

validationHumanDir = os.path.join(validationDir, 'Human')
os.mkdir(validationHumanDir)

validationAIDir = os.path.join(validationDir, 'AI')
os.mkdir(validationAIDir)

testHumanDir = os.path.join(testDir, 'Human')
os.mkdir(testHumanDir)

testAIDir = os.path.join(testDir, 'AI')
os.mkdir(testAIDir)

print("Finished create file to receive all data !!")
```

ตัวอย่างคำสั่งนำภาพ Mel Spectrogram ใส่เข้าไปในแต่ โฟลเดอร์แยกตามประเภทของข้อมูล โดยแบ่งข้อมูลตามประเภท Train set, Validation set และ Test set

```
fnames = ['AI{}.png'.format(i+1) for i in range(1200)]
for fname in fnames:
  src = os.path.join(AIDataDir, fname)
  dst = os.path.join(trainAIDir, fname)
  shutil.copyfile(src, dst)

fnames = ['AI{}.png'.format(i+1) for i in range(1200, 1600)]
for fname in fnames:
  src = os.path.join(AIDataDir, fname)
  dst = os.path.join(validationAIDir, fname)
  shutil.copyfile(src, dst)

fnames = ['AI{}.png'.format(i+1) for i in range(1600, 2000)]
for fname in fnames:
  src = os.path.join(AIDataDir, fname)
  dst = os.path.join(testAIDir, fname)
  shutil.copyfile(src, dst)

fnames = ['Human{}.png'.format(i+1) for i in range(1200)]
for fname in fnames:
  src = os.path.join(HumanDataDir, fname)
  dst = os.path.join(trainHumanDir, fname)
  shutil.copyfile(src, dst)

fnames = ['Human{}.png'.format(i+1) for i in range(1200, 1600)]
for fname in fnames:
  src = os.path.join(HumanDataDir, fname)
  dst = os.path.join(validationHumanDir, fname)
  shutil.copyfile(src, dst)

fnames = ['Human{}.png'.format(i+1) for i in range(1600, 2000)]
for fname in fnames:
  src = os.path.join(HumanDataDir, fname)
  dst = os.path.join(testHumanDir, fname)
  shutil.copyfile(src, dst)
```

ตัวอย่างคำสั่ง การตรวจสอบความครบถ้วนของข้อมูลในโฟลเดอร์ที่จัดเตียมไว้

```
print('total training AI voice images:', len(os.listdir(trainAIDir)))
print('total training Human voice images:', len(os.listdir(trainHumanDir)))
print('total validation AI voice images:',
len(os.listdir(validationAIDir)))
print('total validation Human voice images:',
len(os.listdir(validationHumanDir)))
print('total test AI voice images:', len(os.listdir(testAIDir)))
print('total test Human voice images:', len(os.listdir(testHumanDir)))
```

ตัวอย่างคำสั่งการกำหนดค่าวัตถุ (Object) ของข้อมูลที่จะนำไปพัฒนาตัวแบบจากโฟลเดอร์ที่จัดเตียมไว้แล้ว และทำคำนิยามแต่ละชุดของข้อมูล

```
trainSet = image_dataset_from_directory(
    trainDir,
    image_size = (224,224),
    batch_size = 32
)
valSet = image_dataset_from_directory(
    validationDir,
    image_size = (224,224),
    batch_size = 32
)
testSet = image_dataset_from_directory(
    testDir,
    image_size = (224,224),
    batch_size = 32
)

classLabel = trainSet.class_names

for label, className in enumerate(classLabel):
    print("class: {} --> {}".format(className, label))
```

ตัวอย่างคำสั่งกำหนดตัวแปลเก็บ Callback function

```
early_stopping_cb = EarlyStopping(monitor='val_accuracy', patience=5,
restore_best_weights=True)
model_checkpoint_cb1 = ModelCheckpoint('model_scratch/woaugment',
monitor='val_loss',save_best_only=True, save_weights_only=True)
model_checkpoint_cb2 = ModelCheckpoint('model_scratch/waugment',
monitor='val_loss',save_best_only=True, save_weights_only=True)
model_checkpoint_cb3 =
ModelCheckpoint('model_feature_extraction/woaugment',
monitor='val_loss',save_best_only=True, save_weights_only=True)
model_checkpoint_cb4 = ModelCheckpoint('model_feature_extraction/waugment',
monitor='val_loss',save_best_only=True, save_weights_only=True)
model_checkpoint_cb5 = ModelCheckpoint('model_fine_tuning/woaugment',
monitor='val_loss',save_best_only=True,save_weights_only=True)
model_checkpoint_cb6 = ModelCheckpoint('model_fine_tuning/waugment',
monitor='val_loss',save_best_only=True,save_weights_only=True)
reduce_lr_cb = ReduceLROnPlateau(min_lr=0.001)
```

ตัวอย่างคำสั่งสำหรับวางโครงสร้างของตัวแบบแรก

```
def build_model_scratch():
  model = Sequential()
  model.add(Input((224,224,3)))
  model.add(Rescaling(1./255))
  model.add(Conv2D(filters=32,kernel_size=3,padding='same',activation='relu
'))
  model.add(BatchNormalization())
  model.add(Dropout(0.3))

  model.add(Conv2D(filters=32,kernel_size=3,padding='same',activation='relu
'))
  model.add(MaxPooling2D(pool_size=2))
  model.add(BatchNormalization())
  model.add(Dropout(0.3))

  model.add(Conv2D(filters=64,kernel_size=3,padding='same',activation='relu
'))
  model.add(MaxPooling2D(pool_size=2))
  model.add(BatchNormalization())
  model.add(Dropout(0.3))
```

```
  model.add(Flatten())
  model.add(BatchNormalization())
  model.add(Dense(100, activation='relu', kernel_regularizer=l2(0.01)))
  model.add(BatchNormalization())
  model.add(Dropout(0.3))
  model.add(Dense(1, activation='sigmoid'))

  return model
```

ตัวอย่างคำสั่งสำหรับเริ่มปฏิบัติการพัฒนาตัวแบบ

```
model = build_model_scratch()
model.compile(optimizer=Adam(learning_rate=0.0001,
      beta_1=0.9, beta_2=0.999),
      loss='binary_crossentropy',
      metrics=['accuracy'])
```

ตัวอย่างคำสั่งตรวจสอบโครงสร้างของตัวแบบ

```
model.summary()
```

ตัวอย่างคำสั่งเริ่มการพัฒนาตัวแบบพร้อมเก็บข้อมูลการพัฒนาและใส่ Callback เผื่อการพัฒนาไม่สามารถดำเนินต่อได้ระหว่างการพัฒนา อีกทั้งช่วยย่นระยะการพัฒนาได้

```
history = model.fit(trainSet,
epochs=10,
validation_data=valSet,
verbose=1,
callbacks=[early_stopping_cb,
model_checkpoint_cb2,reduce_lr_cb])
```

ตัวอย่างคำสั่งแสดงผลของการพัฒนาตัวแบบโดยใช้มาตรวัดประสิทธิภาพตัวแบบ Accuracy Score และ Loss ของ Training Set กับ Validation Set

```
def plot_training_history(history):
    acc = history.history['accuracy']
    val_acc = history.history['val_accuracy']
    loss = history.history['loss']
    val_loss = history.history['val_loss']
    epochs_range = range(len(acc))

    plt.figure(figsize=(14, 6))
    plt.subplot(1, 2, 1)
    plt.plot(epochs_range, acc, label='Training Accuracy')
    plt.plot(epochs_range, val_acc, label='Validation Accuracy')
    plt.legend(loc='lower right')
    plt.title('Training and Validation Accuracy')

    plt.subplot(1, 2, 2)
    plt.plot(epochs_range, loss, label='Training Loss')
    plt.plot(epochs_range, val_loss, label='Validation Loss')
    plt.legend(loc='upper right')
    plt.title('Training and Validation Loss')
    plt.show()

plot_training_history(history)
```

ตัวอย่างคำสั่งการทำ Data Augmentation บนข้อมูลสียง

```
modifications = [
    (0.9, -2),
    (0.9, 2),
    (1.1, -2),
    (1.1, 2),
]

def modify_audio(audio_path, time_stretch_factor,
pitch_shift_semitones,modification_id): # [1, 0]
    y, sr = librosa.load(audio_path)

    y_stretched = librosa.effects.time_stretch(y, rate=time_stretch_factor)

    # Apply pitch shifting
    y_shifted = librosa.effects.pitch_shift(y_stretched, sr=sr,
n_steps=pitch_shift_semitones)

    base_filename = os.path.splitext(os.path.basename(audio_path))[0]

    output_filename = f"{base_filename}M{modification_id}.wav"
    output_path = os.path.join(os.path.dirname(audio_path),
output_filename)

    sf.write(output_filename, y_shifted, sr)
```

ตัวอย่างคำสั่งแสดงการทำนายของตัวแบบใน Validation Set

```
predictions = model.predict(valSet)
acClass = []
for x, labels in valSet:
    acClass.extend(labels)

for i in range(len(predictions)):
    predicted_class = np.argmax(predictions[i])
    actual_class = acClass[i]
    probability_scores = predictions[i]

    print(f"Sample {i + 1}: Predicted Class={predicted_class},
      Actual Class={actual_class},
      Probability Scores={probability_scores}")
```

ตัวอย่างคำสั่งการสร้างกราฟ ROC และหาค่า AUC หรือพื้นที่ใต้กราฟ ROC

```
fpr, tpr, _ = roc_curve(acClass, predictions)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8, 8))
plt.plot(fpr, tpr, color='darkorange', lw=2, label='ROC curve (area =
{:.2f})'.format(roc_auc))
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend(loc="lower right")
plt.show()
```

ตัวอย่างคำสั่งการสร้างโครงสร้างตัวแบบที่ 2

```
def build_model_scratch2():
  model = Sequential()
  model.add(Input((224,224,3)))
  model.add(Rescaling(1./255))
  model.add(Conv2D(filters=32,kernel_size=3,padding='same',activation='relu
', kernel_regularizer=l2(0.001)))
  model.add(MaxPooling2D(pool_size=2))
  model.add(BatchNormalization())
  model.add(Dcasropout(0.6))

  model.add(Flatten())
  model.add(BatchNormalization())
  model.add(Dense(50, activation='relu', kernel_regularizer=l2(0.001))) # -
-> change to LeakyReLU
  model.add(BatchNormalization())
  model.add(Dropout(0.6))
  model.add(Dense(1, activation='sigmoid'))

  return model
```

ตัวอย่างคำสั่งวัดผลของตัวแบบกับข้อมูลชุดทดสอบ

```
# test set
loss, acc = model.evaluate(testSet)
print('test_acc:', acc)
```

ตัวอย่างคำสั่งการสร้างโครงสร้างสำหรับพัฒนาตัวแบบโดยสร้าง Search Space สำหรับเป็นกรอบการสุ่มค่า

```
def build_model_scratch_RandomSearch2(hp):
    model = Sequential()
    model.add(Rescaling(1./255))
    model.add(Conv2D(
            filters=hp.Int('conv_filters', min_value=32, max_value=256,
step=32),
            kernel_size=hp.Choice('conv_kernel_size', values=[3, 5]),
            activation='relu',
            input_shape=(224, 224, 3),
            kernel_regularizer=l2(0.001)))
    model.add(MaxPooling2D(pool_size=hp.Choice('pool_size', values =[2,
4])))
    model.add(BatchNormalization())
    model.add(Dropout(0.6))
    model.add(Flatten())
    model.add(Dense(units=hp.Int('dense_units', min_value=32,
max_value=128, step=32), activation='relu', kernel_regularizer=l2(0.001)))
    model.add(Dropout(0.6))
    model.add(Dense(1, activation='sigmoid'))

    model.compile(optimizer=Adam(hp.Float('learning_rate', min_value=1e-4,
max_value=1e-2,sampling='LOG')),loss='binary_crossentropy',
metrics=['accuracy'])

    return model
```

ตัวอย่างคำสั่งการค้นหาพารามิเตอร์ที่ดีที่สุดพร้อมหาค่าที่ดีที่สุดสำหรับตัวแบบที่ ทำการทดลอง

```
tuner = BayesianOptimization(
    build_model_scratch_RandomSearch,
    objective='val_accuracy',
    max_trials=10,
    executions_per_trial=3,
)
tuner.search(trainSet, epochs=10, validation_data=valSet)
best_hps = tuner.get_best_hyperparameters()[0]
best_model = build_model_scratch_RandomSearch2(best_hps)
best_model.build((None, 224, 224, 3))
best_model.summary()
```