บทที่ 3
วิธีดำเนินงานวิจัย

การดำเนินการวิจัยดังต่อไปนี้จะมุ่งเน้นในการศึกษาวิธีการจำแนกภาพถ่ายของปอดที่ได้จากการเอกซเรย์ (X-Ray) ออกเป็น 3 ประเภท คือ ปอดมีการติดเชื้อโควิด-19 , ปอดมีสภาวะปอดอักเสบ (Pneumonia) และปอดไม่มีความเสียหาย (ปกติ)

3.1 การรวบรวมข้อมูล

ทำการการเก็บข้อมูลตัวอย่างของภาพเอกซ์เรย์ปอด โดยมีจำนวนของภาพที่ใช้ทั้งหมด (data set) 678 ภาพ แบ่งเป็นชุดข้อมูลฝึก (training set) 539 ภาพ และชุดข้อมูลทดสอบ (testing set) 139 ภาพ โดยแบ่งออกเป็น 3 classes โดยแยกตามสภาวะความเสียหายของปอด คือ 1.ปอดมีการติดเชื้อโควิด-19 2. ปอดมีสภาวะปอดอักเสบ (Pneumonia) และ 3. ปอดไม่มีความเสียหาย (ปกติ)

3.2 กรอบการดำเนินงาน

3.2.1 การเตรียมข้อมูล (Data Preparation)

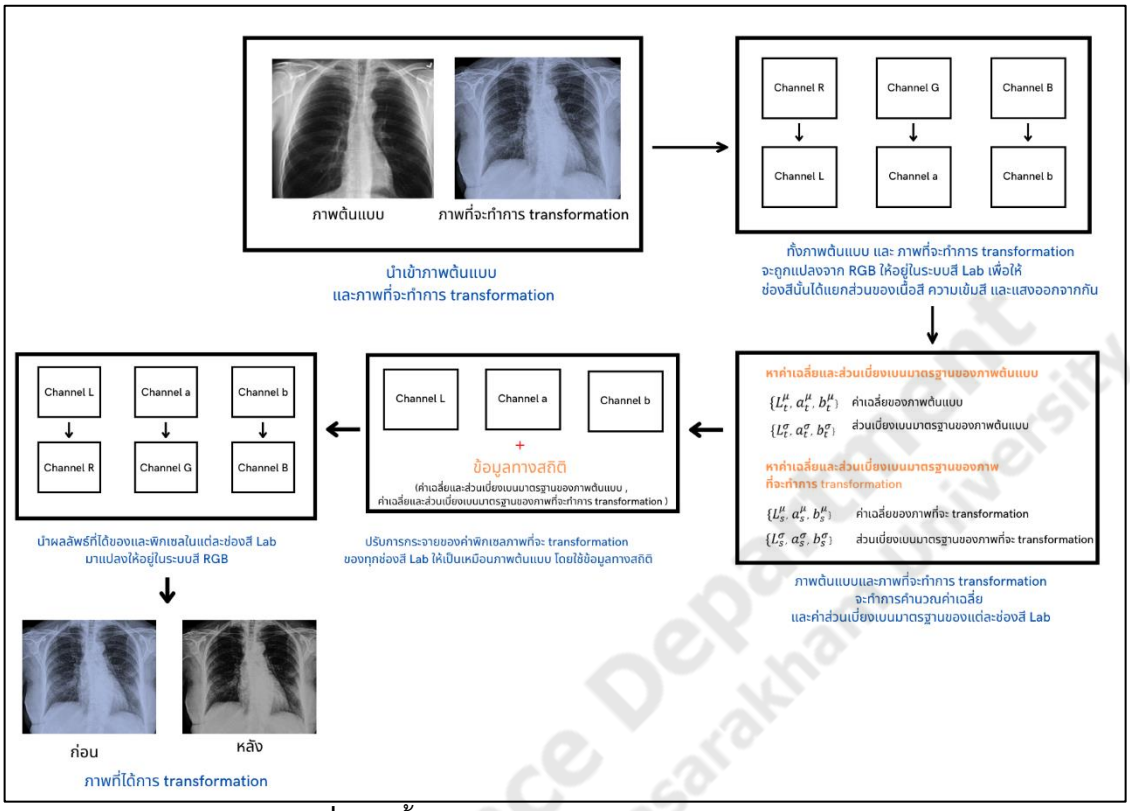
Image Normalization

ทำการ Image Normalization เนื่องจากข้อมูลดิบที่เราได้รับมานั้นมีความหลากหลาย ทั้งชนิดข้อมูล รูปแบบข้อมูล และ Scale ช่วงของข้อมูล

สำหรับอัลกอริทึม Machine Learning หลาย ๆ ตัว ไม่สามารถรับข้อมูลหลากหลาย Scale แบบนี้ได้โดยตรง จำเป็นที่เราต้องทำ Normalization ก่อนที่เราจะป้อนข้อมูลให้กับโมเดล อัลกอริทึมถึงจะสามารถทำงานได้

การทรานสฟอร์มภาพ (Image transformation)

การทรานฟอร์มภาพเป็นอีกกระบวนการทางด้านการประมวลผลภาพหนึ่งที่มีการนำมาปรับปรุง คุณภาพ ของภาพเพื่อให้การประมวลผลในขั้นตอนถัดไปนั้นง่ายขึ้น หรือ ลดความหลากหลายของภาพเพื่อให้การประมวลผลสามารถดำเนินการได้อย่างมีประสิทธิภาพ ในหัวข้อนี้จะอธิบาย หลักการและวิธีการการทรานสฟอร์มภาพด้วยการปรับภาพให้อยู่ในมาตรฐานหรือรูปแบบเดียวกัน เพื่อให้ง่ายต่อการประมวลผล



ภาพประกอบที่ 3.1 ขั้นตอนในการทรานฟอร์ภาพจากคุณสมบัติของภาพ

ขั้นตอนในการทรานฟอร์ภาพจากคุณสมบัติของภาพ มีดัง ต่อไปนี้ คือ

1. ทำการเลือกภาพต้นแบบ (T)
2. ทำการเปลี่ยนระบบสีภาพทั้งภาพต้นแบบ และ ภาพที่จะประมวลผล หรือ ภาพนำเข้า (S)
3. คำนวณค่าสถิติของทั้ง 2 ภาพ
4. ทำการ ปรับการกระจายของค่าพิกเซลของภาพประมวลผลให้เป็นเหมือนภาพต้นแบบ โดยใช้ข้อมูลทางสถิติ
5. ทำการแปลงภาพประมวลผลกลับไปยังระบบสีเดิม

เพื่อให้เห็นภาพถึงขั้นตอนใน การประมวลผลการทรานสฟอร์มภาพ จึงขอกำหนดสัญลักษณ์ที่จะใช้สำหรับอธิบายขั้นตอนการประมวลผล ดัง**ตารางที่ 3.1**

ตารางที่ 3.1 ตารางแสดงสัญลักษณ์สำหรับอธิบายการประมวลผลการทรานสฟอร์มภาพ

สัญลักษณ์	คำอธิบาย
$\{L_s^{\mu}, a_s^{\mu}, b_s^{\mu}\}$	ค่าเฉลี่ยของภาพ
$\{L_s^{\sigma}, a_s^{\sigma}, b_s^{\sigma}\}$	ส่วนเบี่ยงเบนมาตรฐานของภาพ
$\{L_t^{\mu}, a_t^{\mu}, b_t^{\mu}\}$	ค่าเฉลี่ยของภาพต้นแบบ
$\{L_t^{\sigma}, a_t^{\sigma}, b_t^{\sigma}\}$	ส่วนเบี่ยงเบนมาตรฐานของภาพต้นแบบ

โดย L ใช้กำหนดค่าความสว่าง (Lightness) มีค่าตั้งแต่ 0 - 100

L = 0 สีที่ได้จะมืดเป็นสีดำ

L = 100 สีที่ได้จะสว่างเป็นสีขาว

a ใช้กำหนดสีแดง หรือสีเขียว มีค่าตั้งแต่ (-128 to 128)

a เป็น + วัตถุมีสีออกแดง

a เป็น - วัตถุมีสีออกเขียว

b ใช้กำหนดสีเหลือง หรือสีน้ำเงิน มีค่าตั้งแต่ (-128 to 128)

b เป็น + วัตถุมีสีออกเหลือง

b เป็น - วัตถุมีสีออกน้ำเงิน

ภาพจะถูกแปลงจาก RGB ให้อยู่ในระบบสี Lab เพื่อให้ ช่องสีนั้นได้แยกส่วนของเนื้อสี ความเข้มสี และแสงออกจากกัน จากนั้นทั้งภาพต้นแบบและภาพประมวลผลจะทำการคำนวณค่าเฉลี่ย และค่าส่วนเบี่ยงเบนมาตรฐานของแต่ละช่องสี (ซึ่งจะทำการ คำนวณแยก)

10	30	52
20	18	56
41	21	58

R

30	36	29
53	65	42
59	13	51

G

40	12	51
62	37	59
19	12	25

B

ภาพประกอบที่ 3.2 ค่าสี RGB ของภาพต้นแบบ

10.01	13.08	11.18
20.04	23.79	19.24
22.54	4.55	21.48

L

-5.44	-6.41	-8.80
-9.25	-22.27	9.46
-13.47	4.47	0.10

a

-9.40	14.03	9.85
-9.52	12.28	-8.33
21.76	3.90	17.57

b

ภาพประกอบที่ 3.3 ค่าสี Lab ของภาพต้นแบบ

หาค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานของภาพต้นแบบ

สูตรการหาค่าส่วนเบี่ยงเบนมาตรฐาน

$$\{L_t^{\sigma}, a_t^{\sigma}, b_t^{\sigma}\} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - L_t^{\mu}, a_t^{\mu}, b_t^{\mu}\right)^2} \quad (7)$$

$\{L_t^{\sigma}, a_t^{\sigma}, b_t^{\sigma}\}$ คือส่วนเบี่ยงเบนมาตรฐานที่ต้องการหา

n คือจำนวนของข้อมูลทั้งหมด

X_i คือข้อมูลแต่ละค่าตัวในพิกเซล

$\{L_t^{\mu}, a_t^{\mu}, b_t^{\mu}\}$ คือค่าเฉลี่ยของภาพต้นแบบ

ขั้นตอนที่ 1 คำนวณหาค่าเฉลี่ยของภาพต้นแบบ

ค่าเฉลี่ยของช่องสี L

$$L_t^{\mu} = \frac{(10.01 + 13.08 + 11.18 + 19.04 + 20.24 + 23.79 + 22.54 + 4.55 + 21.48)}{9}$$

$$L_t^{\mu} = \frac{145.91}{9}$$

$$L_t^{\mu} = 16.21$$

ค่าเฉลี่ยของช่องสี a

$$a_t^{\mu} = \frac{(-5.44 + (-6.41) + (-8.8) + 9.46 + (-22.27) + (-10.08) + (-13.47) + 4.47 + 0.1)}{9}$$

$$a_t^{\mu} = \frac{-52.44}{9}$$

$$a_t^{\mu} = -5.82$$

ค่าเฉลี่ยของช่องสี b

$$b_t^{\mu} = \frac{(-9.4 + 14.03 + 9.85 + -8.33 + 17.57 + -3.9 + 21.76 + -9.52 + 12.28)}{9}$$

$$b_t^{\mu} = \frac{44.34}{9}$$

ค่าเฉลี่ยของช่องสี b (ต่อ)

$$b_t^{\mu} = 4.92$$

ขั้นตอนที่ 2 คำนวณหาส่วนเบี่ยงเบนมาตรฐานของภาพต้นแบบ

L_t^{σ}

$$L_t^{\sigma} = \sqrt{\frac{(10.01-16.21)^2+(13.08-16.21)^2+(11.18-16.21)^2+(19.04-16.21)^2+(20.24-16.21)^2+(23.79-16.21)^2+(22.54-16.21)^2+(4.55-16.21)^2+(21.48-16.21)^2}{(n-1)}}$$

$$L_t^{\sigma} = \sqrt{\frac{38.46+9.81+25.32+7.99+16.22+57.42+40.04+136+27.74}{(9-1)}}$$

$$L_t^{\sigma} = \sqrt{\frac{359.04}{(9-1)}}$$

$$L_t^{\sigma} = 6.70$$

a_t^{σ}

$$a_t^{\sigma} = \sqrt{\frac{\Sigma}{(n-1)}}$$

$$= \sqrt{\frac{(-5.44-(-5.82))^2+(-6.41-(-5.82))^2+(-8.8-(-5.82))^2+(9.46-(-5.82))^2+(-22.27-(-5.82))^2+(-10.08-(-5.82))^2+(-13.47-(-5.82))^2+(4.47-(-5.82))^2+(0.1-(-5.82))^2}{(n-1)}}$$

$$a_t^{\sigma} = \sqrt{\frac{0.14+0.34+8.84+233.68+270.38+18.09+58.42+106.02+35.12}{(9-1)}}$$

$$a_t^{\sigma} = \sqrt{\frac{731.05}{(9-1)}}$$

$$a_t^{\sigma} = 9.56$$

b_t^{σ}

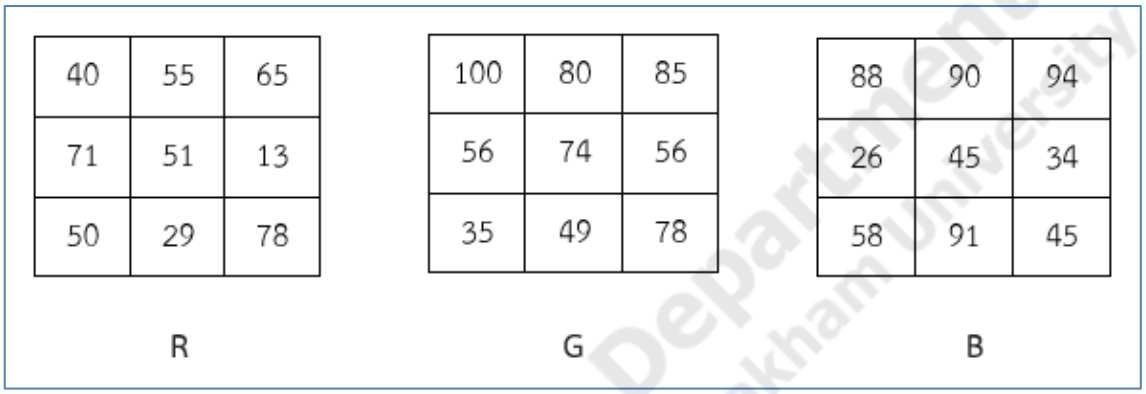
$$b_t^{\sigma} = \sqrt{\frac{\Sigma}{(n-1)}}$$

$$= \sqrt{\frac{(-9.4-4.92)^2+(14.03-4.92)^2+(9.85-4.92)^2+(-8.33-4.92)^2+(17.57-4.92)^2+(-3.9-4.92)^2+(21.76-4.92)^2+(-9.52-4.92)^2+(12.28-4.92)^2}{(n-1)}}$$

$$= \sqrt{\frac{205.25 + 82.87 + 24.23 + 175.73 + 159.85 + 77.91 + 283.36 + 208.7 + 54.07}{(9-1)}}$$

$$= \sqrt{\frac{1272}{(9-1)}}$$

$$b_t^{\sigma} = 12.61$$



40	55	65
71	51	13
50	29	78

R

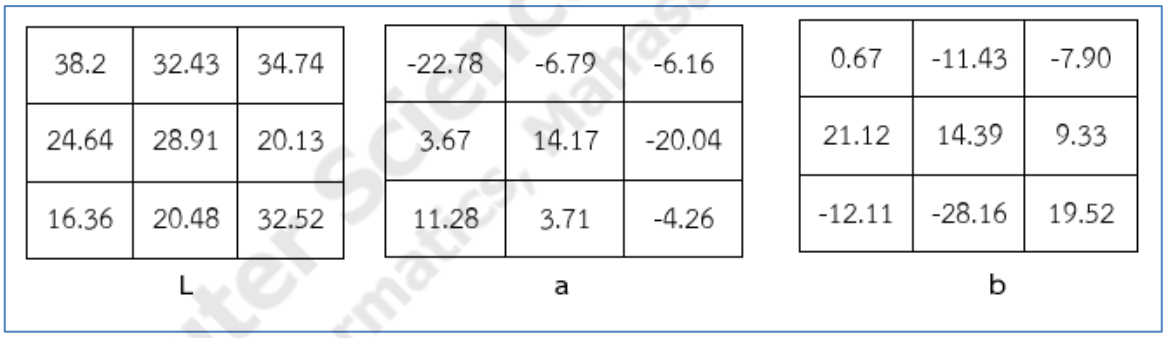
100	80	85
56	74	56
35	49	78

G

88	90	94
26	45	34
58	91	45

B

ภาพประกอบที่ 3.4 ค่าสี RGB ของภาพนำเข้า



38.2	32.43	34.74
24.64	28.91	20.13
16.36	20.48	32.52

L

-22.78	-6.79	-6.16
3.67	14.17	-20.04
11.28	3.71	-4.26

a

0.67	-11.43	-7.90
21.12	14.39	9.33
-12.11	-28.16	19.52

b

ภาพประกอบที่ 3.5 ค่าสี Lab ของภาพนำเข้า

จากนั้นหาค่าค่าเฉลี่ยและส่วนเบี่ยงเบนมาตรฐานของภาพนำเข้า

สูตรการหาค่าส่วนเบี่ยงเบนมาตรฐาน

$$\{L_s^{\sigma}, a_s^{\sigma}, b_s^{\sigma}\} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - \{L_s^{\mu}, a_s^{\mu}, b_s^{\mu}\}\right)^2}, \quad (8)$$

$\{L_s^{\sigma}, a_s^{\sigma}, b_s^{\sigma}\}$ คือส่วนเบี่ยงเบนมาตรฐานที่ต้องการหา

n คือจำนวนของข้อมูลทั้งหมด

X_i คือข้อมูลแต่ละค่าตัวในพิกเซล

$\{L_s^{\mu}, a_s^{\mu}, b_s^{\mu}\}$ คือค่าเฉลี่ยของภาพนำเข้า

ขั้นตอนที่ 1 คำนวณหาค่าเฉลี่ยของภาพนำเข้า

ค่าเฉลี่ยของช่องสี L

$$L_s^{\mu} = \frac{(38.2 + 32.43 + 34.74 + 24.64 + 28.91 + 20.13 + 16.36 + 20.48 + 32.52)}{9}$$

$$L_s^{\mu} = \frac{248.41}{9}$$

$$L_s^{\mu} = 27.6$$

ค่าเฉลี่ยของช่องสี a

$$a_s^{\mu} = \frac{(-22.78 + (-6.79) + (-6.16) + 3.67 + 14.17 + (-20.04) + 11.28 + 3.71 + (-4.26))}{9}$$

$$a_s^{\mu} = \frac{-27.2}{9}$$

$$a_s^{\mu} = -3.02$$

ค่าเฉลี่ยของช่องสี b

$$a_s^{\mu} = \frac{(0.67 + (-11.43) + (-7.9) + 21.12 + 14.39 + 9.33 + (-12.11) + (-28.16) + 19.52)}{9}$$

$$a_s^{\mu} = \frac{5.43}{9}$$

$$a_s^{\mu} = 0.6$$

ขั้นตอนที่ 2 คำนวณหาส่วนเบี่ยงเบนมาตรฐาน

คำนวณค่า L_s^{σ}

$$L_s^{\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - L_s^{\mu}\right)^2}$$

$$L_s^{\sigma} = \sqrt{\frac{(10.6)2 + (4.83)2 + (7.14)2 + (-2.96)2 + (1.31)2 + (-7.47)2 + (-11.24)2 + (-7.12)2 + (4.92)2}{(9-1)}}$$

$$L_s^{\sigma} = \sqrt{\frac{112.33 + 23.31 + 50.96 + 8.76 + 1.71 + 55.81 + 126.36 + 50.71 + 24.19}{8}}$$

$$L_s^{\sigma} = \sqrt{\frac{454.18}{8}}$$

$$L_s^{\sigma} = 7.53$$

คำนวณค่า a_s^{σ}

$$a_s^{\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - a_s^{\mu}\right)^2}$$

$$a_s^{\sigma} = \sqrt{\frac{\begin{matrix}(-22.78-(-3.02))2 + (-6.79-(-3.02))2 + (-6.16-(-3.02))2 + \\ (3.67-(-3.02))2 + \\ (14.17-(-3.02))2 + (-20.04-(-3.02))2 + (11.28-(-3.02))2 + \\ (3.71-(-3.02))2 + \\ (-4.26-(-3.02))2\end{matrix}}{(9-1)}}$$

$$a_s^{\sigma} = \sqrt{\frac{\begin{matrix}(-19.76)2 + (-3.77)2 + (-3.14)2 + (6.69)2 + (17.19)2 + \\ (-17.02)2 + (14.3)2 + (6.73)2 + (-1.24)2\end{matrix}}{(9-1)}}$$

$$a_s^{\sigma} = \sqrt{\frac{\begin{matrix}390.36 + 14.19 + 9.84 + 44.78 + 295.57 + \\ 289.6 + 204.55 + 45.32 + 1.53\end{matrix}}{8}}$$

$$a_s^{\sigma} = \sqrt{\frac{1295.78}{(9-1)}}$$

$$a_s^{\sigma} = 12.73$$

คำนวณค่า b_s^{σ}

$$b_s^{\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}\left(x_i - b_s^{\mu}\right)^2}$$

$$b_s^{\sigma} = \sqrt{\frac{\begin{matrix}(0.67-0.6)2 + (-11.43-0.6)2 + (-7.9-0.6)2 + (21.12-0.6)2 \\ + (14.39-0.6)2 + (9.33-0.6)2 + (-12.11-0.6)2 + (-28.16-0.6)2 + \\ (19.52-0.6)2\end{matrix}}{(9-1)}}$$

$$b_s^{\sigma} = \sqrt{\frac{\begin{matrix}(0.07)2 + (-12.03)2 + (-8.5)2 + (20.52)2 + (13.79)2 + \\ (8.73)2 + (-12.71)2 + (-28.76)2 + (18.92)2\end{matrix}}{(9-1)}}$$

$$b_s^{\sigma} = \sqrt{\frac{0 + 144.8 + 72.3 + 420.93 + 190.07 + 76.15 + 161.62 + 827.32 + 357.84}{(9-1)}}$$

$$b_s^{\sigma} = \sqrt{\frac{2251.07}{8}}$$

$$b_s^{\sigma} = 16.77$$

จากนั้นพิกเซลทุกพิกเซลในภาพประมวลผล ของทุกช่องสี จะทำการปรับเพื่อหาระยะห่างจากค่ากลาง โดยการนำค่าพิกเซลลบออกจากค่าเฉลี่ย ดังสมการที่ 9

$$\begin{aligned} L_s^f &= L_s - L_s^{\mu} \\ a_s^f &= a_s - a_s^{\mu} \\ b_s^f &= b_s - b_s^{\mu} \end{aligned} \qquad (9)$$

$L_s^f =$

38.2 - 27.6	32.43 -27.6	34.74 - 27.6
24.64 - 27.6	28.91 - 27.6	20.13 - 27.6
16.36 - 27.6	20.48 - 27.6	32.52 - 27.6

$L_s^f =$

10.6	4.83	7.14
-2.96	1.31	-7.47
-11.24	-7.12	4.92

ภาพประกอบที่ 3.6 การหาระยะห่างจากค่ากลางของช่องสี L

$a_s^f =$

-22.78 – (-3.02)	-6.79 – (-3.02)	-6.16 – (-3.02)
3.67-3.02	14.17-3.02	-20.04 - (-3.02)
11.28-(-3.02)	3.71-3.02	-4.26-(-3.02)

$a_s^f =$

-19.76	-3.77	-3.14
0.65	11.15	-17.02
14.3	0.69	-1.24

ภาพประกอบที่ 3.7 การหาระยะห่างจากค่ากลางของช่องสี a

$b_s^f =$

0.67-0.6	-11.43-(-0.6)	-7.90-(-0.6)
21.12-0.6	14.39-0.6	9.33-0.6
-12.11-(-0.6)	-28.16-(-0.6)	19.52-0.6

$b_s^f =$

0.07	-10.83	-7.3
20.25	13.79	8.73
-11.51	-27.56	18.92

ภาพประกอบที่ 3.8 การหาระยะห่างจากค่ากลางของช่องสี b

จากนั้นทำการทรานฟอร์มค่าโดยการนำค่าระยะทางจากค่ากลางที่ได้ของแต่ละพิกเซล นำมาคูณกับ ค่าน้ำหนัก ซึ่งค่าน้ำหนักนี้คำนวณจากสัดส่วนของค่าส่วนเบี่ยงเบนมาตรฐานของภาพต้นแบบ และ ภาพประมวล ดังสมการที่ (10)

$$L'_s = \frac{L_t^\sigma}{L_S^\sigma} * L_s^f$$
$$a'_s = \frac{a_t^\sigma}{a_S^\sigma} * a_s^f \quad (10)$$
$$b'_s = \frac{b_t^\sigma}{b_S^\sigma} * b_s^f$$

$L'_s = \frac{L_t^\sigma}{L_S^\sigma} * L_s^f$, L_t^σ = 6.70 , L_S^σ = 7.53

$L'_s =$

$\frac{6.70}{7.53}$ * 10.6	$\frac{6.70}{7.53}$ * 4.83	$\frac{6.70}{7.53}$ *7.14
$\frac{6.70}{7.53}$ *-2.96	$\frac{6.70}{7.53}$ *1.31	$\frac{6.70}{7.53}$ *-7.47
$\frac{6.70}{7.53}$ *-11.24	$\frac{6.70}{7.53}$ *-7.12	$\frac{6.70}{7.53}$ *4.92

$L'_s =$

9.43	4.29	6.35
-2.63	1.16	-6.64
-10	-6.33	4.37

ภาพประกอบที่ 3.9 ทำการทรานฟอร์มช่องสี L

$a'_s = \frac{a_t^\sigma}{a_S^\sigma} * a_s^f$, a_t^σ = 9.56 , a_S^σ = 12.73

$a'_s =$

$\frac{9.56}{12.73}$ * (-19.76)	$\frac{9.56}{12.73}$ * (-3.77)	$\frac{9.56}{12.73}$ * (-3.14)
$\frac{9.56}{12.73}$ *0.65	$\frac{9.56}{12.73}$ *11.15	$\frac{9.56}{12.73}$ * (-17.02)
$\frac{9.56}{12.73}$ *14.3	$\frac{9.56}{12.73}$ *0.69	$\frac{9.56}{12.73}$ * (-1.24)

$a'_s =$

-14.83	-2.83	-2.35
0.49	8.37	-12.78
10.74	0.52	-0.93

ภาพประกอบที่ 3.10 ทำการทรานฟอร์มช่องสี a

$b'_s = \frac{b^{\sigma}_t}{b^{\sigma}_S} * b^{f}_s$, b^{σ}_t = 12.61 , b^{σ}_s = 16.77

$b'_s =$

$\frac{12.61}{16.77} * 0.07$	$\frac{12.61}{16.77} * (-10.83)$	$\frac{12.61}{16.77} * (-7.3)$
$\frac{12.61}{16.77} * 20.25$	$\frac{12.61}{16.77} * 13.79$	$\frac{12.61}{16.77} * 8.73$
$\frac{12.61}{16.77} * (-11.51)$	$\frac{12.61}{16.77} * (-27.56)$	$\frac{12.61}{16.77} * 18.92$

$b'_s =$

0.05	-8.14	-5.49
15.22	10.37	6.56
-8.65	-20.72	14.23

ภาพประกอบที่ 3.11 ทำการทรานฟอร์มช่องสี b

เมื่อทำการปรับค่าพิกเซลใหม่ให้กับภาพประมวลผลแล้ว ขั้นตอนต่อไปคือการปรับค่าพิกเซล ให้กลับคืนสู่ค่าเดิมด้วยการบวกค่าเฉลี่ยของแต่ละช่องสีให้กับพิกเซล แต่ในการบวกค่าเฉลี่ยคืน นั้น ในขั้นตอนนี้จะได้ค่าเฉลี่ยของแต่ละช่องสีจากภาพต้นแบบ ดังสมการที่ (11)

$$L''_s = L'_s + L^{\mu}_t$$
$$a''_s = a'_s + a^{\mu}_t \quad (11)$$
$$b''_s = b'_s + b^{\mu}_t$$

$L''_s =$

9.43 + 16.21	4.29+ 16.21	6.35 + 16.21
-2.63 + 16.21	1.16 + 16.21	-6.64+ 16.21
-10 + 16.21	-6.33 + 16.21	4.37 + 16.21

$L''_s =$

25.64	20.5	22.56
13.58	17.37	9.57
6.21	9.88	20.58

ภาพประกอบที่ 3.12 การปรับค่าพิกเซลของช่องสี L ให้กลับคืนสู่ค่าเดิม

$a_s'' = a_s' + a_t^{\mu}$, a_t^{μ} = (-5.82)

$a_s'' =$

-14.83 + (-5.82)	-2.83 + (-5.82)	-2.35 + (-5.82)
0.49 + (-5.82)	8.37 + (-5.82)	-12.78 + (-5.82)
10.74 + (-5.82)	0.52 + (-5.82)	-0.93 + (-5.82)

$a_s'' =$

-20.65	-8.65	-8.17
-5.33	2.55	-18.6
4.92	-5.3	-6.75

ภาพประกอบที่ 3.13 การปรับค่าพิกเซลขอช่องี a ให้กลับคืนสู่ค่าเดิม

$b_s'' = b_s' + b_t^{\mu}$, b_t^{μ} = 4.92 คำนวณได้จาก**ภาพประกอบที่ 3.14**

$b_s'' =$

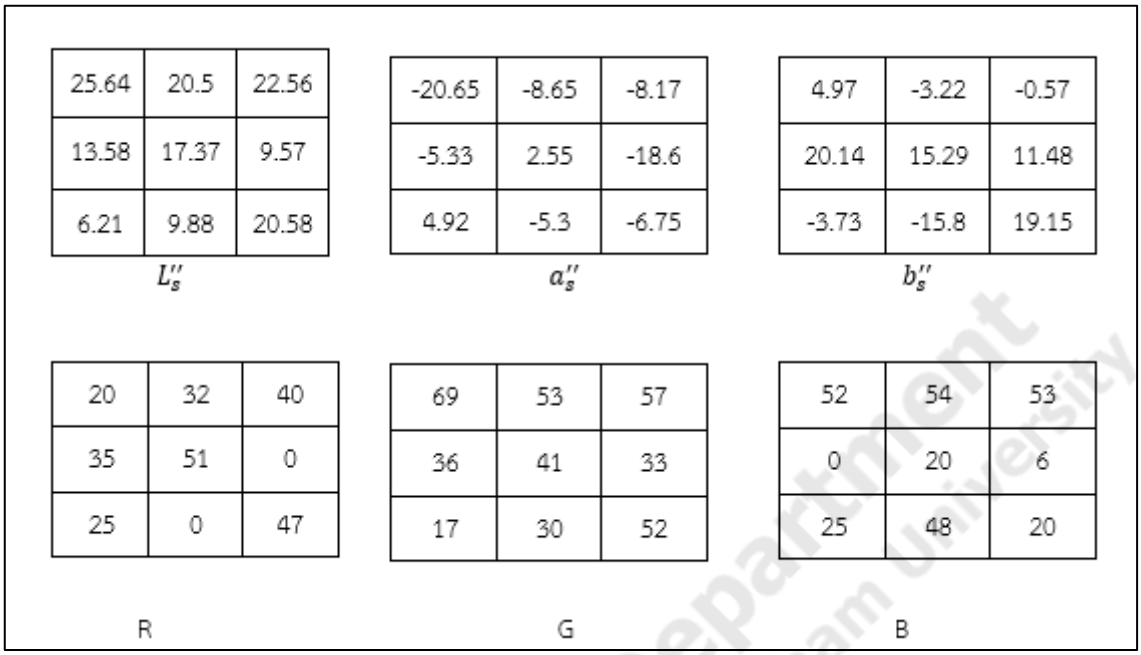
0.05 + 4.92	-8.14+ 4.92	-5.49+ 4.92
15.22 + 4.92	10.37 + 4.92	6.56+ 4.92
-8.65 + 4.92	-20.72+ 4.92	14.23+ 4.92

$b_s'' =$

4.97	-3.22	-0.57
20.14	15.29	11.48
-3.73	-15.8	19.15

ภาพประกอบที่ 3.14 การปรับค่าพิกเซลขอช่องี b ให้กลับคืนสู่ค่าเดิม

นำผลลัพธ์ที่ได้ของและพิกเซลในแต่ละช่องสี $(\{ L_s'' , a_s'', b_s''\})$ มาแปลงให้อยู่ในระบบสี RGB ดัง**ภาพประกอบที่ 3.15**



ภาพประกอบที่ 3.15 แปลงค่าสี Lab ให้เป็น RGB

ซึ่งจะทำได้ภาพผลลัพธ์และตัวอย่างของการทรานฟอร์มภาพนั้นแสดงดัง**ภาพประกอบที่ 3.16**

ภาพประกอบที่ 3.16 ตัวอย่างของการทรานสฟอร์มภาพ

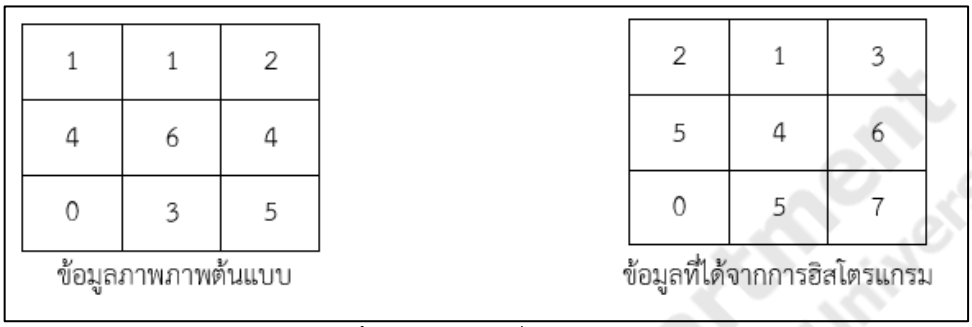
จากภาพจะเห็นได้ว่าภาพประมวลผลหรือภาพนำเข้านั้น (b) เมื่อมีการทรานฟอร์มภาพ แล้วจะมีโทนสีหรือ look and feel ของภาพที่คล้ายกับภาพต้นแบบ

3.2.2 การทำให้ภาพคมชัด (Contrast)

การเพิ่มคอนทราสต์ของภาพ (contrast enhancement) เป็นอีกกระบวนการหนึ่งที่สามารถทำได้ด้วยการดำเนินการกับฮิสโตรแกรมของภาพ ซึ่งคอนทราสต์ของภาพนั้นจะทำให้เราสามารถ เห็นความแตกต่างระหว่างค่าสีที่มีความต่างกันได้ชัดเจนขึ้น โดยทั่วไปการเพิ่มคอนทราสต์ของภาพ นั้นจะดำเนินกับภาพที่มีค่าสีในภาพนั้นใกล้เคียงกัน และในบางครั้งการที่มีคอนทราสต์ที่ไม่เหมาะ สมจะให้เราไม่สามารถเห็นรายละเอียดบางส่วนในภาพอย่างชัดเจน

ฮิสโตรแกรมอีควอไลซ์เซชัน (histogram equalization)

เป็นเทคนิคที่ปรับการกระจายของ ค่าสีในภาพให้มีลักษณะที่เท่า ๆ กันในทุกระดับค่าสี สำหรับภาพระดับเทาการทำฮิสโตรแกรมอี ควอไลซ์เซชันสามารถดำเนินการโดยอัลกอริทึมที่ 3 ขั้นตอนแรกคือการคำนวณค่าฮิสโตรแกรม ของภาพ และนำฮิสโตรแกรมที่ได้ไปประมวลผลเพื่อหาฮิสโตรแกรมสะสมของภาพ (AH) ดัง**ภาพประกอบที่ 3.17**



ภาพประกอบที่ 3.17 ข้อมูลที่ได้จากการฮิสโตรแกรม

ตารางที่ 3.2 การหาฮิสโตรแกรมสะสมของภาพ (AH)

ระดับค่าสี **Level (L)**	**ความถี่** **Histogram**	**ความถี่สะสม** **AH**
0	1	1
1	1	2
2	1	3
3	1	4
4	1	5
5	2	7
6	1	8
7	1	9

Algorithm การสร้างฮิสโตรแกรมสะสมของภาพระดับเทา

1: คำนวณฮิสโตรแกรมของภาพ

2: คำนวณฮิสโตรแกรมสะสมของภาพ

3: คำนวณหาความน่าจะเป็นของความถี่สะสมของทุกระดับ

4: ทำการสร้างตารางระดับค่าสี

จากนั้นจะทำการหาค่าที่ได้จากฟังก์ชันแจกแจงฮิสโตรแกรมสะสมของแต่ละระดับค่าสี (Cumulative Distributive Function : CDF) หรือความน่าจะเป็นของฮิสโตรแกรมสะสมของแต่ละระดับค่าสี ดัง**ตารางที่ 3.3** คำนวณได้จาก

$$CDF(l) = \frac{AH(l)}{N} \qquad (12)$$

โดย l คือระดับสี และ N คือจำนวนพิกเซลทั้งหมดในภาพ

ตารางที่ 3.3 การหาค่าที่ได้จากฟังก์ชันแจกแจงฮิสโตรแกรมสะสมของแต่ละระดับค่าสี (CDF)

ระดับค่าสี Level (L)	ความถี่ Histogram	ความถี่สะสม AH	ความน่าจะเป็น CDF
0	1	1	$\frac{AH(l)}{N} = \frac{1}{9} = 0.1$
1	1	2	$\frac{AH(l)}{N} = \frac{2}{9} = 0.22$
2	1	3	$\frac{AH(l)}{N} = \frac{3}{9} = 0.33$
3	1	4	$\frac{AH(l)}{N} = \frac{4}{9} = 0.44$
4	1	5	$\frac{AH(l)}{N} = \frac{5}{9} = 0.56$
5	2	7	$\frac{AH(l)}{N} = \frac{7}{9} = 0.78$
6	1	8	$\frac{AH(l)}{N} = \frac{8}{9} = 0.89$
7	1	9	$\frac{AH(l)}{N} = \frac{9}{9} = 1$

จากนั้นจะคำนวณระดับค่าสีใหม่ (L') ซึ่งคำนวณได้โดยสมการที่ (13)

$$L' = [CDF(L) * (L_n - 1) \quad (13)$$

เมื่อ L_n คือจำนวนระดับค่าสี ตัวอย่างของผลการคำนวณแสดงดัง

ตารางที่ 3.4 การคำนวณระดับค่าสีใหม่ (L')

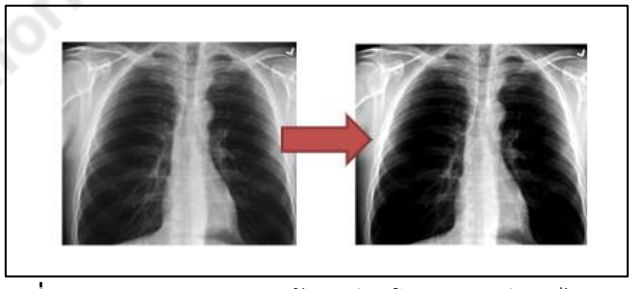
ระดับค่าสี Level (L)	ความถี่ Histogram	ความถี่สะสม AH	ความน่าจะเป็น CDF	ระดับค่าสีใหม่ L´
0	1	1	$\frac{AH(l)}{N} = \frac{1}{9} = 0.1$	$L' = [CDF(L) * (L_n - 1)]$ $= 0.1 * (8 - 1)$ $= 0$
2	1	3	$\frac{AH(l)}{N} = \frac{3}{9} = 0.33$	$L' = [CDF(L) * (L_n - 1)]$ $= 0.33 * (8 - 1)$ $= 2$
3	1	4	$\frac{AH(l)}{N} = \frac{4}{9} = 0.44$	$L' = [CDF(L) * (L_n - 1)]$ $= 0.44 * (8 - 1)$ $= 3$
4	1	5	$\frac{AH(l)}{N} = \frac{5}{9} = 0.56$	$L' = [CDF(L) * (L_n - 1)$ $= 0.56 * (8 - 1)$ $= 3$

ตารางที่ 3.4 การคำนวณระดับค่าสีใหม่ (L')

ระดับค่าสี Level (L)	ความถี่ Histogram	ความถี่สะสม AH	ความน่าจะเป็น CDF	ระดับค่าสีใหม่ L´
5	2	7	$\frac{AH(l)}{N} = \frac{7}{9} = 0.78$	$L' = [CDF(L) * (L_n - 1)$ $= 0.78 * (8 - 1)$ $= 5$
1	1	2	$\frac{AH(l)}{N} = \frac{2}{9} = 0.22$	$L' = [CDF(L) * (L_n - 1)]$ $= 0.22 * (8 - 1)$ $= 1$
6	1	8	$\frac{AH(l)}{N} = \frac{8}{9} = 0.89$	$L' = [CDF(L) * (L_n - 1)$ $= 0.89 * (8 - 1)$ $= 6$
7	1	9	$\frac{AH(l)}{N} = \frac{9}{9} = 1$	$L' = [CDF(L) * (L_n - 1)$ $= 1 * (8 - 1)$ $= 7$

จากตัวอย่างกำหนดใหระดับคาสีมีทั้งหมด 8 ระดับ (L = 8) และความถี่ของแต่ละระดับแสดงดังคอลัมนที่ 2 ของตาราง คอลัมนที่ 3 แสดงฮิสโตรแกรมสะสมของแต่ละระดับคาสี และคอลัมนที่ 4 แสดงคา CDF และ คอลัมนที่ 5 แสดงคาสีใหม่ของแต่ละระดับคาสี ซึ่งคำนวณจากสมการที่ 1.3 คาสีใหม่นี้เองจะทำใหการกระจายของพิกเซลในแต่ละดับตางจากเดิม ตัวอย่างเชนระดับคาสี L = 4 จะถูกเปลี่ยนใหเป็นระดับคาสี 3 ดังนั้นพิกเซลใดๆ ที่อยู่ในภาพที่มีคาสีเป็น 4 จะถูกเปลี่ยนใหเป็นคาสี 3 นั่นเอง สำหรับภาพระดับเทาที่มี L = 256

ภาพตัวอย่างของฮิสโตรแกรมอีควอไลซเซชันแสดงดัง**ภาพประกอบที่ 3.18** จากผลลัพธ์ที่ได้จะเห็นได้วาคอนทราสตของภาพนั้นเพิ่มมากขึ้น



ภาพประกอบที่ 3.18 การ contrast ด้วยวิธีฮิสโตรแกรมอีควอไลซเซชัน

3.2.3 Deblur

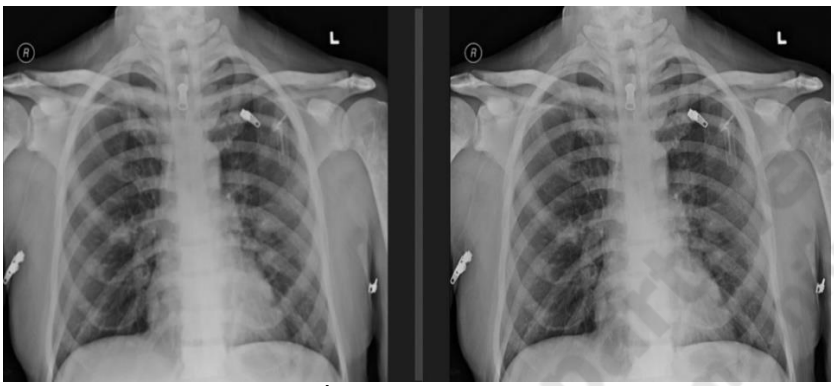
เป็นอีกกระบวนการทางด้านการประมวลผลภาพหนึ่งที่มีการนำมาปรับปรุง คุณภาพ ของภาพ เพื่อให้การประมวลผลในขั้นตอนถัดไปนั้นง่ายขึ้น หรือ ลดเบลอของภาพเพื่อให้การประมวลผลสามารถดำเนินการได้อย่างมีประสิทธิภาพ ในหัวข้อนี้จะอธิบาย หลักการและวิธีการการดีเบลอภาพด้วยการปรับให้มีความคมชัดมากขึ้น เพื่อให้ง่ายต่อการประมวลผล ขั้นตอนในดีเบลอภาพที่ใช้มีดัง ต่อไปนี้ คือ

1. เลือกภาพต้นแบบ

2.หาค่า threshold ของภาพ ด้วย Otsu อัลกอริทึม

3. หาเส้นของภาพด้วย canny age

4. ปรับค่าให้มีความคมชัดมากยิ่งขึ้นด้วยฟังก์ชั่น sharpening Kernel

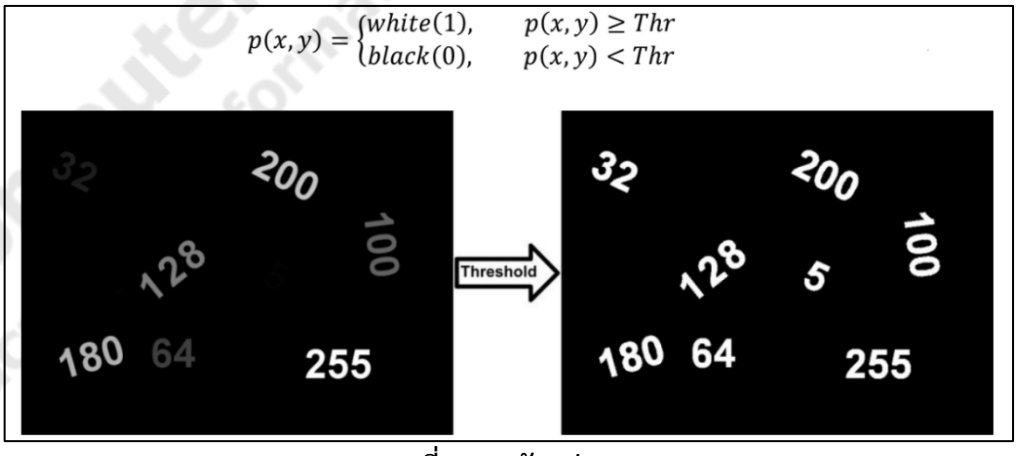


ภาพประกอบที่ 3.19 ภาพผลลัพธ์การ deblur

การหาค่า threshold เป็นการเลือกจุดตัดที่เหมาะสมในการแบ่งส่วนของภาพ ซึ่งจะบ่างเป็นการคัดเลือกจุดตัดแบบภาพรวม (Global thresholding) และการคัดเลือกจุดตัดแบบกลุ่มย่อย(Local thresholding) ดังต่อไปนี้

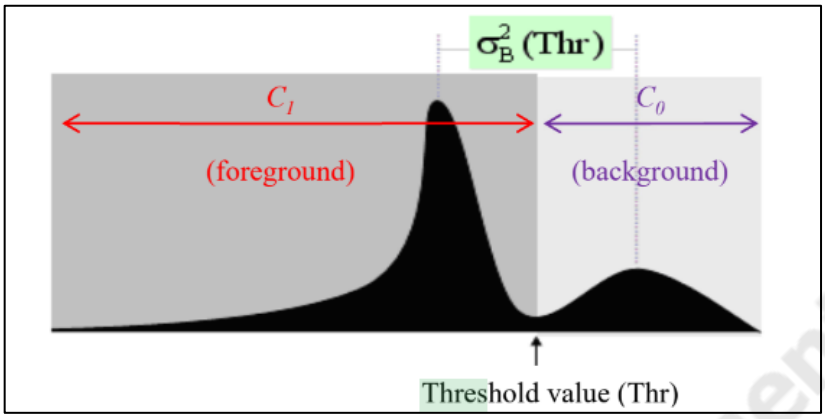
Global Thresholding เป็นการหาจุดตัดจากภาพรวมทั้งหมดของภาพ แบ่งเป็นวิธีหลักๆ 2 วิธี ได้แก่ Single thresholding และ Double thresholding

Single thresholding เป็นการพิจารณาจากฮิสโตแกรมของระดับความเข้มของภาพ เพื่อหาจุดตัดที่เหมาะสม วิธีนี้ใช้ได้ดีกับการแยกภาพที่วัตถุกับพื้นหลังแยกกันอย่างชัดเจน



ภาพประกอบที่ 3.20 ตัวอย่าง threshold

Otsu Algorithm มีหลักการทางสถิติในการหาจุดตัด ดังนี้



ภาพประกอบที่ 3.21 ภาพแสดงจุดตัด Threshold ของภาพด้วย Otsu algorithm

1. คำนวณหาฮิสโตแกรมของภาพจากภาพระดับเทา และ Normalize ให้อยู่ในรูปของการแจกแจงความน่าจะเป็น ดังสมการ

$$p_1 = \frac{n_i}{N}, p_i \geq 0, \sum_{i=0}^{L-1} p_i = 1 \quad (14)$$

โดยที่

n_i = จำนวนจุดตัดภาพที่ระดับความเข้ม i

N = จำนวนจุดภาพทั้งหมด

L = คือระดับตวามเข้มของภาพระดับเทา 0,1,2, … , $L-1$

2. แบ่งกลุ่มของฮิสโตแกรมของภาพเป็น 2 กลุ่ม (Class) คือ C_0 (Object /Foreground) และ C_1 (Background)

C_0 แทนด้วยกลุ่มระดับความเข้ม (0, … , $k-1$)

C_1 แทนด้วยกลุ่มระดับความเข้ม (k, … , $L-1$)

$$\omega_1 = P(C_1) = \sum_{i=0}^{k-1} p_i = \omega(k), \quad (15)$$

$$\omega_1 = P(C_1) = \sum_{i=k}^{L-1} p_i = \omega(k) \quad (16)$$

3. หาค่าเฉลี่ยของแต่ละกลุ่มดังนี้

$$\mu_0 = \sum_{i=0}^{k-1} i \cdot P(i|C_0) = \sum_{i=0}^{k-1} i \cdot \frac{p_i}{\omega_0} = \frac{\mu(k)}{\omega(k))}, \quad (17)$$

$$\mu_1 = \sum_{i=K}^{L-1} i \cdot P(i|C_1) = \sum_{i=k}^{L-1} i \cdot \frac{p_i}{\omega_1} = \frac{\mu_T - \mu(k)}{1-\omega(k)} \quad (18)$$

ซึ่งสมการที่ (19)

$$\omega(k) = \sum_{i=0}^{k-1} i \cdot p_i, \quad \mu_t = \mu(L) = \sum_{i=0}^{L-1} i \cdot p_i, \tag{19}$$

ดังนั้นจะเห็นได้ว่าสมการที่ (20)

$$\omega_0\mu_0 + \omega_1\mu_1 = \mu_T, \ \omega_0 + \omega_1 = 1 \tag{20}$$

4. หาความแปรปรวน (Variance) ของแต่ละกลุ่มดังนี้

$$\sigma_0^2 = \sum_{i=0}^{k-1}(i - \mu_0)^2 p(i|c_0) = \sum_{i=0}^{k-1}(i-\mu_0)^2 \cdot \frac{p_i}{\omega_0} \tag{21}$$

$$\sigma_1^2 = \sum_{i=k}^{L-1}(i - \mu_1)^2 p(i|c_1) = \sum_{i=0}^{L-1}(i-\mu_1)^2 \cdot \frac{p_i}{\omega_i} \tag{22}$$

จากนั้นเพื่อหาจุดตัดที่ดีที่สุดจะใช้หลักการ Discriminant analysis ซึ่งเป็นตัววัดที่ดีที่สุดดังนี้

$$\mu = \frac{\sigma_B^2}{\sigma_T^2} \tag{23}$$

5. หา Optimal threshold k* ซึ่งจากสมการที่ผ่านมาจะเห็นได้ว่าสมการที่ (24)

$$n(k) = \frac{\sigma_B^2}{\sigma_T^2}, \ \sigma_B^2(k) = \frac{[\mu_T\omega(k) - \mu(k)]^2}{\omega(k)[1\omega(k)]} \tag{24}$$

ดังนั้นจะได้ Optimal threshold ดังสมการที่ (25)

$$\sigma_B^2(k*) = \max_{0\le k<L-1} \sigma_B^2(k) = \min_{0\le k<L-1} n(k) \tag{25}$$

ซึ่ง Thr = k* นั่นเอง

Double Thresholding สำหรับกรณีที่ต้องการหาค่า Threshold 2 ค่า สามารถทำได้ ดังนี้

$$p(x,y) = \begin{cases} white(1), & T_1 \le p(x,y) \le T_2 \\ black(0), & otherwise \end{cases} \tag{26}$$

ดังนั้นเราสามารถประยุกต์ Otsu algorithm ได้โดยสมมติว่ามีค่า Threshold 2 ค่า นั่นคือมีกลุ่มข้อมูล 3 กลุ่ม คือ C_0, C_1, C_2 ที่จำแนกจากกันได้ ดังนั้น

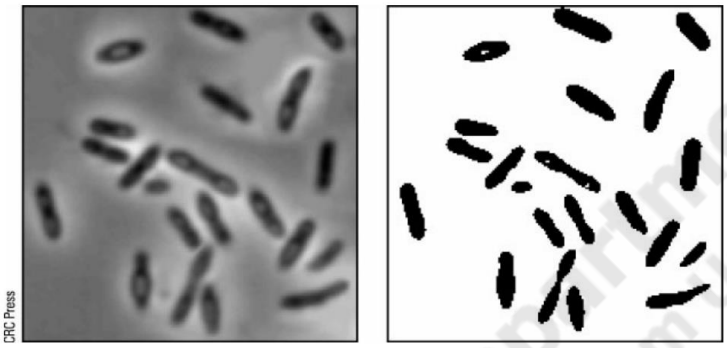
C_0 แทนด้วยกลุ่มระดับความเข้ม (0, ... $k_0 - 1$),

C_1 แทนด้วยกลุ่มระดับความเข้ม (k_0, ... $k_1 - 1$)

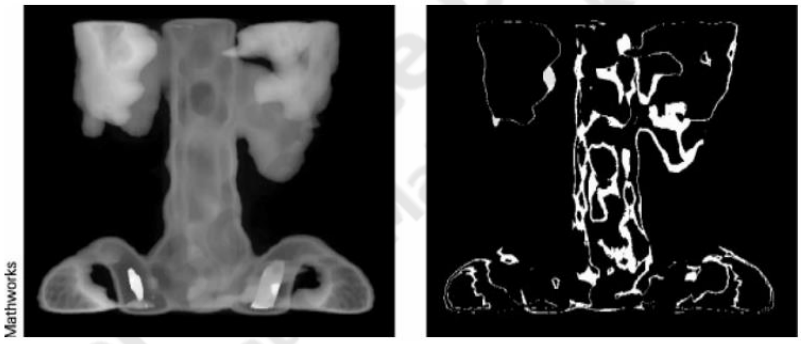
C_2 แทนด้วยกลุ่มระดับความเข้ม $(k_1, \ldots L-1)$

ดังนั้นเราจะหาได้จากสมการที่ (27)

$$\sigma_B^2(k_0^*, k_1^*) = \max_{0 \le k_0, k_1 < L-1} \sigma_B^2(k_0, k_1) \quad (27)$$



ภาพประกอบที่ 3.22 แสดงผลลัพธ์จาก Single Thresholding



ภาพประกอบที่ 3.23 แสดงผลลัพธ์จาก Double Thresholding

คำสั่งการหาค่า Threshold ใน OpenCV

ปกติการหาค่า Threshold เป็นการสร้างภาพสองระดับ(Binary Image) ซึ่งจะให้ผลลัพธ์ของระดับความเข้มของภาพมีเพียง 2 ระดับ คือ สีดำ(0) และสีขาว(1 หรืออาจจะเป็นค่าใดๆ ที่มากที่สุดของระบบสีนั้นๆ)

reVal , dst = cv2.threshold(source, thresholdValue, maxValue, thresholdingTechnique)

Parameters :

source : ภาพนำเข้า

thresholdValue : ค่าจุดแบ่ง(Threshold) ระดับสีสองระดับ ซึ่งเป็นการกำหนดค่าคงที่

maxVal : ค่าระดับความเข้มสูงสสุดที่จะกำหนดให้กับจุดภาพ

thresholdingTechnique : วิธีการหาค่า Threshold หรือจุดแบ่งสีของระดับ

- cv2.THRESH_BINARY :

$$des(x,y) = \begin{cases} maxValue & if\ \ src(x,y) > threshold \\ 0\ (black) & otherwise \end{cases}$$

- cv2.THRESH_BINARY_INV : เป็น inverse ของ cv2.THRESH_BINARY

$$des(x,y) = \begin{cases} 0\ (black) & otherwise \\ maxValue & if\ \ src(x,y) > threshold \end{cases}$$

- cv2.THRESH_TRUNC :

$$des(x,y) = \begin{cases} threshold & if\ \ src(x,y) > threshold \\ src(x,y) & otherwise \end{cases}$$

- cv2.THRESH_TOZERO :

$$des(x,y) = \begin{cases} src(x,y) & if\ \ src(x,y) > threshold \\ 0\ (black) & otherwise \end{cases}$$

- cv2.THRESH_TOZERO_INV : เป็น inverse ของ cv2.THRESH_TOZERO

$$des(x,y) = \begin{cases} src(x,y) & if\ \ src(x,y) > threshold \\ 0\ (black) & otherwise \end{cases}$$

- cv2.THRESH_OTSU :

เป็นการหาค่า Threshold ด้วยวิธีการ Otsu Algorithm ซึ่งนำมาใช้ผสมกับ 5 วิธีข้างต้น เพื่อให้ได้ค่า Threshold โดยอัตโนมัติ

Return :

retVal : ค่า threshold

dst : ภาพผลลัพธ์สองระดับ

```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

image1 = cv2.imread('../house.tiff')
img = cv2.cvtColor(image1, cv2.COLOR_BGR2GRAY)

ret, thr1 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY)
ret, thr2 = cv2.threshold(img, 127, 255, cv2.THRESH_BINARY_INV)
ret, thr3 = cv2.threshold(img, 150, 255, cv2.THRESH_TRUNC)
ret, thr4 = cv2.threshold(img, 150, 255, cv2.THRESH_TOZERO)
ret, thr5 = cv2.threshold(img, 150, 255, cv2.THRESH_TOZERO_INV)
ret, thr6 = cv2.threshold(img, 0, 255, cv2.THRESH_OTSU)
print(ret)
ret, thr7 = cv2.threshold(img, 0, 255,
                          cv2.THRESH_TOZERO+cv2.THRESH_OTSU)

# De-allocate any associated memory usage
titles = ['Original Image','BINARY','BINARY_INV','TRUNC','TOZERO',
          'TOZERO_INV','Otsu','TOZERO+OTSU']
images = [img, thr1, thr2, thr3, thr4, thr5, thr6, thr7]

for i in xrange(6):
    plt.subplot(2,4,i+1),plt.imshow(images[i],'gray')
    plt.title(titles[i]),plt.xticks([]),plt.yticks([])

plt.show()
```

ภาพประกอบที่ 3.24 ตัวอย่างโปรแกรม Global Thresholding

Local (Adaptive) Thresholding

สำหรับ Local Threshold หรือเรียกว่า Adaptive Threshold ซึ่งเป็นการหาค่า Threshold ภาพในพื้นที่ย่อยๆ ของแต่ละจุดภาพ ดังนั้นค่า Threshold จะปรับเปลี่ยนไปตามแต่ละจุดภาพ

วิธีการหาค่าเฉลี่ย(Mean หรือ Average Filter) วิธีการหาค่าเฉลี่ยด้วยการปรับ Weight (Average Weight Filter หรือ Guassian Filter)หรือค่ามัธยฐาน(Median Filter) โดยมีอัลกอริทึมดังนี้

1. Convolution ภาพต้นฉบับด้วย Average Filter, Guassian Filter หรือ Median Filter แทนผลลัพธ์ด้วย $\boldsymbol{g(x,y)}$

2. เปรียบเทียบพิกเซล ทุกจุดบนภาพต้นฉบับ $f(x,y)$ กับ $g(x,y)$

$$dst(x,y) = \begin{cases} 1(white) & if\ \ f(x,y) > g(x,y) + c \\ 0\ (black) & otherwise \end{cases} \qquad (28)$$

โดยที่ c เป็นค่าคงที่

คำสั่งการหาค่า Adaptive Threshold ใน OpenCV

สำหรับ Adaptive Threshold ใน OpenCV มีอยู่ 2 วิธี คือ Mean_C และ Guassian_C

dst = cv.adaptiveThreshold(src, maxValue, adaptivMethod, ThresholdType,blockSize, C)

Parameters :

src : ภาพนำเข้า

maxVal : ค่าระดับความเข้มสูงสสุดที่จะกำหนดให้กับจุดภาพ

adaptiveMethod : ได้แก่ cv2.ADAPTIVE_THRESH_MEAN_C หรือ cv2.ADAPTIVE_THRESH_GAUSSIAN_C

thresholdType : วิธีการหาค่า threshold หรือจุดแบ่งระดับสองสี ได้แก่ cv2.THRESH_BINARY, cv2.THRESH_BINARY_INV,

thresholdType (ต่อ) : cv2.THRESH_BINARY,cv2.THRESH_TRUNC, cv2.THRESH_TOZERO, cv2.THRESH_TOZERO_INV

blockSize : ขนาดของ window ย่อย ปกติจะเป็นจำนวนคี่ เช่น 3,5,7,9

C : ค่าคงที่ที่ลบจาก mean หรือ guassian ที่คำนวณได้ในแต่ละจุดภาพ

Return :

dst : ภาพผลลัพธ์สองระดับ

ตารางที่ 3.5 ข้อดีและข้อเสียของวิธี Thresholding

วิธี	ข้อดี	ข้อเสีย
Global Thresholding	- เป็นวิธีที่ง่าย - ทำงานได้เร็ว - เหมาะกับภาพที่ระดับสีมีการกระจายตัวสม่ำเสมอ (Uniform)	- กำจัดสัญญาณรบกวนได้ไม่ดี - ไม่เหมาะกับภาพที่ระดับสีมีการกระจายไม่สม่ำเสมอ - แบ่งระดับของสีได้ไม่มากนัก เหมาะกับภาพสีไม่ซับซ้อน
Local Thresholding	- เป็นวิธีที่ง่าย - ทำงานได้ช้ากว่า Global - เหมาะกับภาพที่ระดับสีมีการกระจายตัวไม่สม่ำเสมอ	- แบ่งระดับของสีได้ไม่มากนัก เหมาะกับภาพสีไม่ซับซ้อน

โดยรูปแบบในการทำ Pre-Processing นั้นจะทำการสลับทั้ง 3 วิธี เรียงลำดับวิธีใดทำก่อนหลังจะได้ทั้งหมด 6 รูปแบบ ได้แก่

1. Transformation -> Contrast -> Deblur
2. Transformation -> Deblur -> Contrast
3. Contrast -> Deblur -> Transformation
4. Contrast -> Transformation -> Deblur
5. Deblur -> Contrast -> Transformation
6. Deblur -> Transformation -> Contrast

หลังจากการทำ Pre-Processing ทั้ง 6 รูปแบบเสร็จสิ้น จะนำไปวัดประสิทธิภาพด้วย CNN โดยใช้ 3 เทคนิค ได้แก่ Inception-V3 , VGG-16 และ ResNet 50-V2

3.3 การวัดประสิทธิภาพ

การวัดประสิทธิภาพการทำงานของโปรแกรมเปรียบเทียบกับผลลัพธ์ที่ถูกต้องสามารถวัดได้โดยการหาค่าดังต่อไปนี้

Recall คือค่าที่บอกว่าโปรแกรมทำนายได้จริง ว่ามีอัตราส่วนเท่าใดของค่าจริงทั้งหมด สามารถคำนวณได้ดังนี้

$$Recall = \frac{TP}{(TP+FN)} = \frac{TP}{All\ Ground\ Truth} \quad (29)$$

Precision คือ ค่าที่โปรแกรมทำนายว่าจริงถูกต้อง สามารถคำนวณได้ดังนี้

$$Precision = \frac{TP}{(TP+FP)} = \frac{TP}{All\ predict} \quad (30)$$

Accuracy คือ ค่าที่โปรแกรมทำนายว่าจริง สามารถคำนวณได้ดังนี้

$$Accuracy = \frac{TP+TN}{P+N} \quad (31)$$

F1-score คือ ค่าเฉลี่ยฮาร์มอนิกของ Precision และ Recall สามารถคำนวณได้ดังนี้

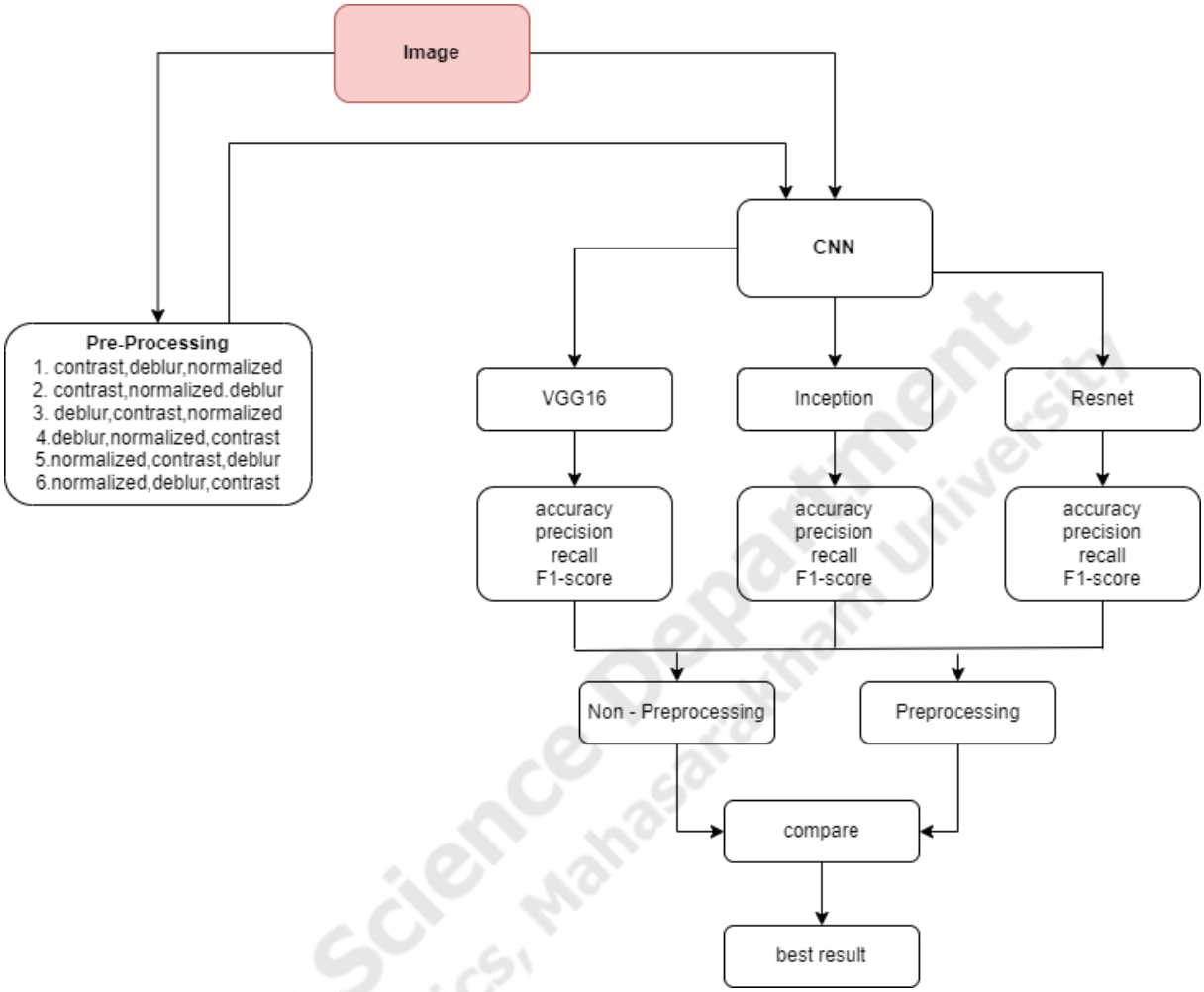
$$F1-score = 2 \times \frac{precision \times Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN} \quad (32)$$

โดยที่ True Positive (TP) คือ ค่าที่โปรแกรมทำนายว่าจริง และเป็นความจริง

True Negative (TN) คือ ค่าที่โปรแกรมทำนายว่าจริง แต่ไม่เป็นความจริง

False Positive (FP) คือ ค่าที่โปรแกรมทำนายว่าไม่จริง แต่เป็นความจริง

False Negative (FN) คือ ค่าที่โปรแกรมทำนายว่าไม่จริง และไม่เป็นความจริง



ภาพประกอบที่ 3.25 ภาพรวมระบบ

ในการจำแนกจำแนกสภาวะความเสียหายของปอดนั้นเราจะใช้วิธีการคือ จำแนกโดยการเรียนรู้เชิงลึก (Deep Learning) โดยใช้ 3 เทคนิค ได้แก่

- InceptionV3
- VGG16
- ResNet 50-V2

และจำแนกสภาวะความเสียหายของปอดเป็น 3 ประเภท คือ 1.ปอดมีการติดเชื้อโควิด-19 2. ปอดมีสภาวะปอดอักเสบ(Pneumonia) และ 3. ปอดไม่มีความเสียหาย (ปกติ) ในการวัดประสิทธิภาพนั้น สามารถทำได้โดยการแบ่งข้อมูล ดัง**ตารางที่ 3.6**

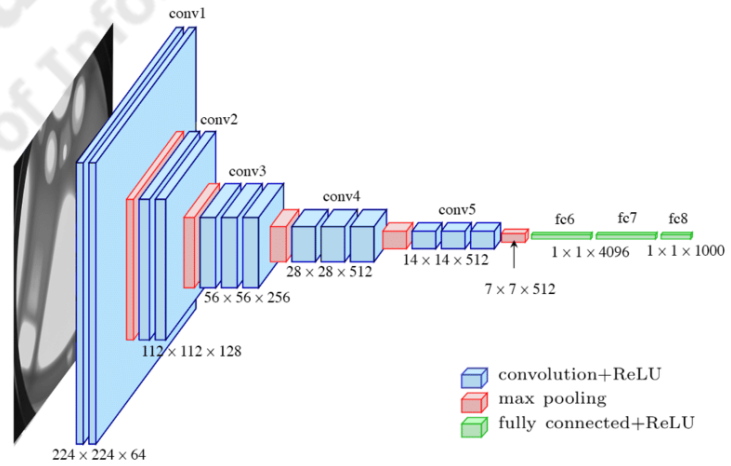
ตารางที่ 3.6 การแบ่งข้อมูล

Class	NO. Images		
	Training Data	Validation	Testing Data
ปอดไม่มีความเสียหาย(ปกติ)	184	35	46
ปอดมีการติดเชื้อโควิด-19	155	24	43
ปอดมีสภาวะปอดอักเสบ (Pneumonia)	200	25	50

เมื่อทำการแบ่งชุดข้อมูลเสร็จ จะทำการจำแนกข้อมูลภาพ (Image classification) ด้วย CNN (Convolutional Neural Network) ดังนี้

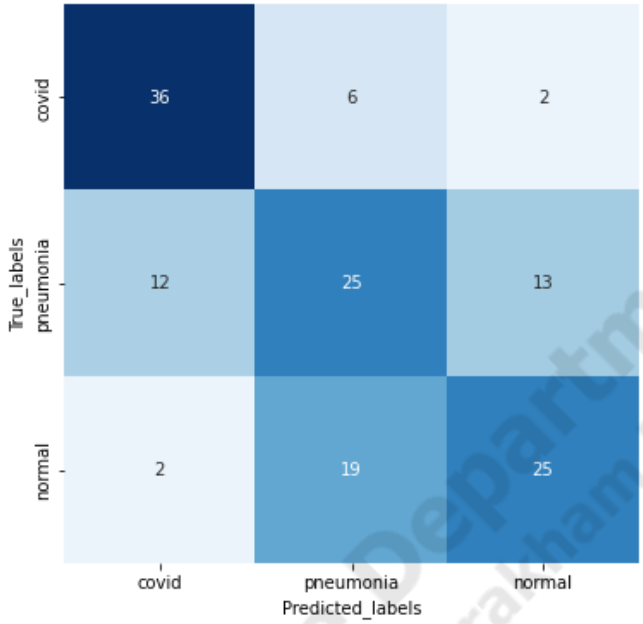
3.3.1 การจำแนกภาพเอกซ์เรย์ปอดโดยใช้ Model VGG16

โมเดล VGG ย่อมาจาก Visual Geometry Group ซึ่งเป็นกลุ่มนักวิจัยจาก Oxford ทำการพัฒนาสถาปัตยกรรมนี้ขึ้นมา และที่ได้รับความสนใจมากจากการแข่งขัน ILSVR ปี ค.ศ. 2014 และเป็นที่นิยมจนถึงปัจจุบัน โดยสิ่งที่เป็นจุดเด่นของ VGG16 คือการแทนที่ hyperparameter จำนวนมาก เน้นไปที่การออกแบบ เลเยอร์ conv2D 3x3 pixels, 1 stride และการใช้ same padding และ maxpooling 2x2 pixels, 2 stride แบบเดียวกันตลอดทั้งโครงสร้าง โดยชื่อของ VGG16 หมายถึงมี 16 ชั้นที่มีน้ำหนักเครือข่ายนี้เป็นเครือข่ายที่ใหญ่และมีพารามิเตอร์ประมาณ 138 ล้าน ดัง**ภาพประกอบที่ 3.26**



ภาพประกอบที่ 3.26 ภาพโครงสร้าง VGG16

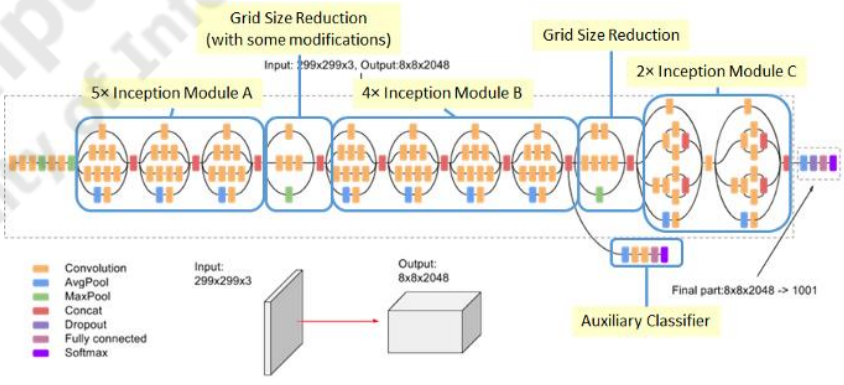
ผลที่ได้จากการจำแนกภาพโดยใช้ VGG16 ด้วยการฝึกโดยใช้ข้อมูลทั้งหมด 15 รอบ ได้ผลดังนี้



ภาพประกอบที่ 3.27 ตัวอย่างผลการทำนายโดยใช้ CNN VGG-16

3.3.2 การจำแนกภาพเอกซ์เรย์ปอดโดยใช้ Model inception V3

เป็นวิธีการที่ใช้ในการทำการจำแนกภาพ (Image classification model) ซึ่งเป็น CNN ที่มี ความซับซ้อนโดยตัว Inception V3 นี้สามารถ classify ได้มากถึง 1000 classes โดยในงานนี้เราจะนำ Inception V3 มาเพื่อใช้จำแนกภาพเอกซ์เรย์ปอดออกเป็น 3 ประเภท คือ 1.ปอดมีการติดเชื้อโควิด-19 2. ปอดมีสภาวะปอดอักเสบ(Pneumonia) และ 3. ปอดไม่มีความเสียหาย (ปกติ)



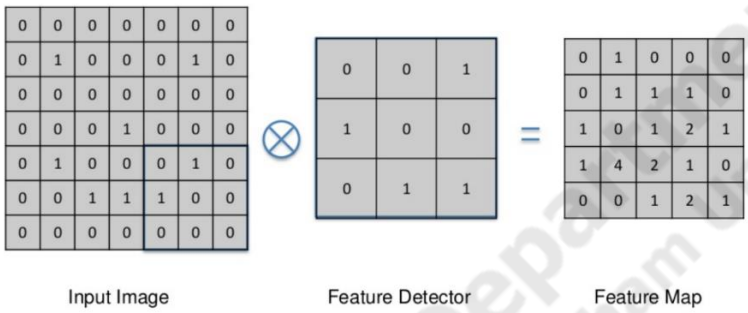
ภาพประกอบที่ 3.28 Inception-v3 Architecture

Inception V3 นั้นเป็น CNN รูปแบบหนึ่งโดย CNN (Convolutional Neural Network) นั้นคือ การวิเคราะห์รูปภาพที่มนุษย์มองเห็น โดยจะแบ่งรูปภาพออกเป็นพื้นที่ย่อย ๆ เป็น Pixel แต่ละอัน

เพื่อทำการวิเคราะห์ Metrix ของรูปภาพ โดยถ้าเป็นรูปภาพสีขาวดำ จะเป็น Metrix 2x2 แต่ถ้าเป็นภาพสีจะเป็น Metric 3x3 โดยจะมีการกำหนด โดยสถาปัตยกรรมจะเห็นได้จากภาพประกอบที่ 3.28

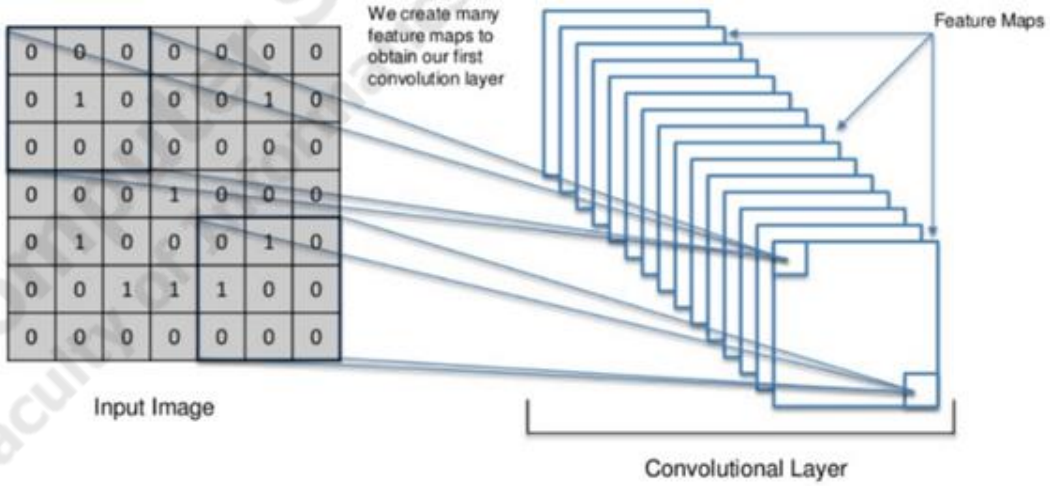
คอนโวลูชัน (Convolution)

คอนโวลูชันจะเป็นกระบวนการที่ให้สามารถสกัดเอาลักษณะเด่นของรูปถ่ายออกมา โดยใช้ การใช้ Filter ซึ่งจะมีการกำหนดขึ้นมาเพื่อให้ Filter นั้นเลื่อนไปตามภาพและทำการประมวลผลจนได้คุณลักษณะของภาพออกมาและเก็บไว้ในเมทริกซ์ใหม่ดัง**ภาพประกอบที่ 3.29**

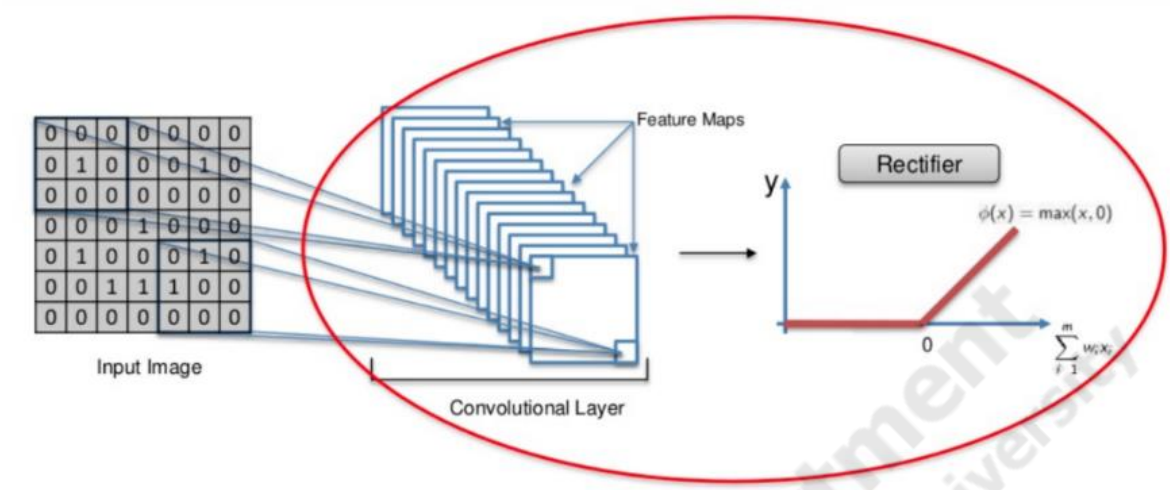


ภาพประกอบที่ 3.29 ตัวอย่างภาพรวมการทำ Convolution

คอนโวลูชันทำการคูณเมทริกซ์ ระหว่าง Input Image กับ Feature Detector ทำให้ได้ Feature Map เมื่อทำการคูณเมทริกซ์ทั้งหมดแล้วจะได้ Feature Map จำนวนมากโดยจะเรียกทั้งหมดว่า Convolutional Layer **ภาพประกอบที่ 3.30** และหลังจากได้ Convolutional Layer ก็ทำการ Rectifier โดยทำ ReLu Layer



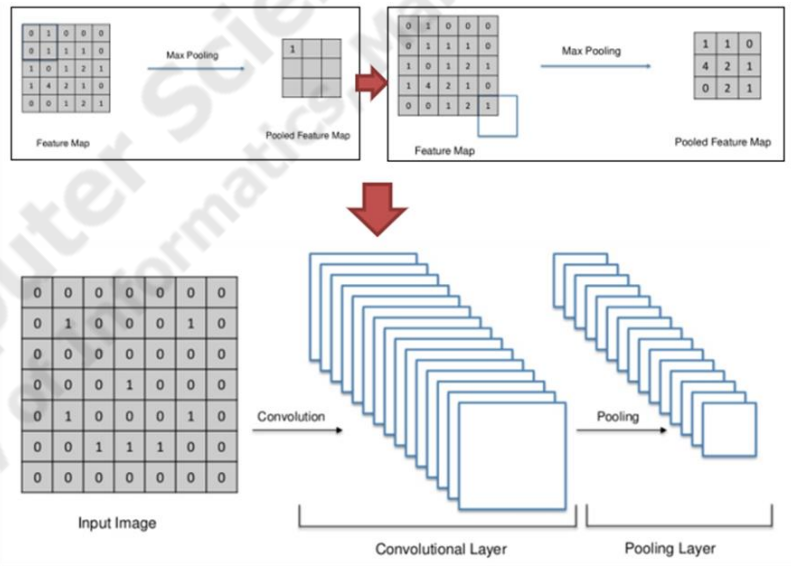
ภาพประกอบที่ 3.30 ทำการ Rectifier โดยทำ ReLu Layer



ภาพประกอบที่ 3.30 ทำการ Rectifier โดยทำ ReLu Layer (ต่อ)

Max Pooling

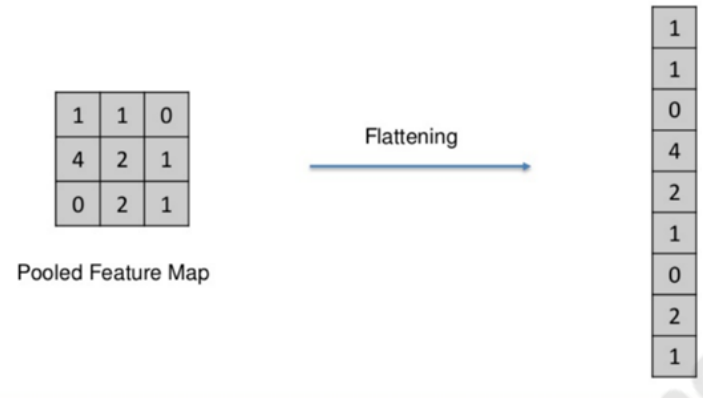
เป็นการลดขนาดของภาพเพื่อเพิ่มความเร็วในการประมวลผล โดยจะทำการกำหนด Filter ขึ้นมาและเลือกค่าที่มากที่สุดใน Filter นั้น และเก็บไว้ใน เมทริกซ์ซึ่งการทำวิธีการ Max pooling นั้น ถึงแม้ขนาดของภาพจะลดลงแต่ก็ยังคงรายละเอียดของภาพต้นฉบับไว้อยู่ ระบบจะทำการเลือกค่าที่มากที่สุดของแต่ละเมทริกซ์เพื่อให้ได้ Pooled Feature Map ดัง **ภาพประกอบที่ 3.31**



ภาพประกอบที่ 3.31 เลือกค่าที่มากที่สุดของแต่ละเมทริกซ์เพื่อให้ได้ Pooled Feature Map -

Flattening

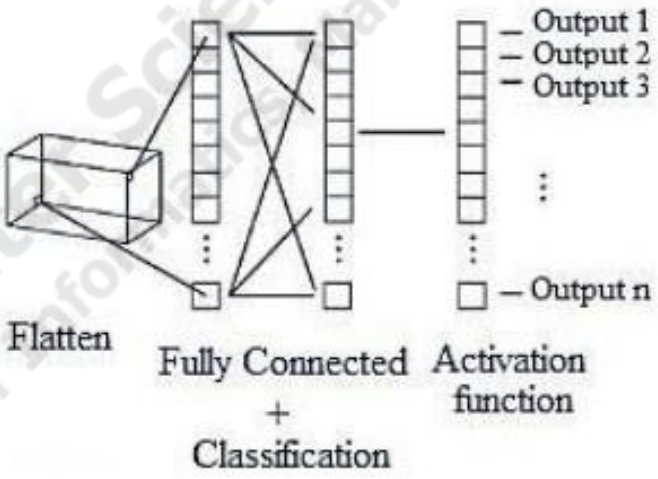
จะทำต่อจากการ Max Pooling เป็นการทำ Pooling Feature Map ที่ได้ทำเป็นคอลัมน์เดียวกัน หรือเป็นการทำให้ข้อมูลที่เป็นเมทริกซ์หลายมิติให้เป็นคอลัมน์เดียว เพื่อความสะดวกในการวิเคราะห์ข้อมูล ดัง**ภาพประกอบที่ 3.32**



ภาพประกอบที่ 3.32 การทำ Flattening

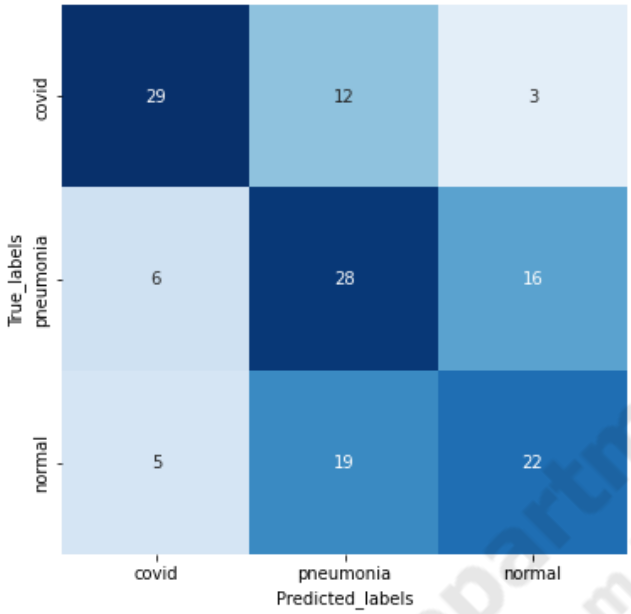
Fully connected layer

คือกระบวนการ Convolution, relu และการ pooling กระบวนการทั้งสามนี้เป็นกระบวนการที่ทำซ้ำได้หลายครั้งและในขั้นสุดท้ายมีการเชื่อมต่อกันของแต่ละชั้นอย่างสมบูรณ์ Fully connected layer ผลลัพธ์จากคอนโวลูชันและพูลลิ่ง นั้นให้ลักษณะเด่น (High-Level Features) ของรูปที่รับเข้ามา และชั้นสุดท้ายเพื่อนำลักษณะเด่นไปทำการคัดกรองรูปที่รับเข้าให้อยู่ในรูปของ classes ดัง **ภาพประกอบที่ 3.33**



ภาพประกอบที่ 3.33 การเชื่อมต่อกันของแต่ละชั้นอย่างสมบูรณ์ Full connected layer

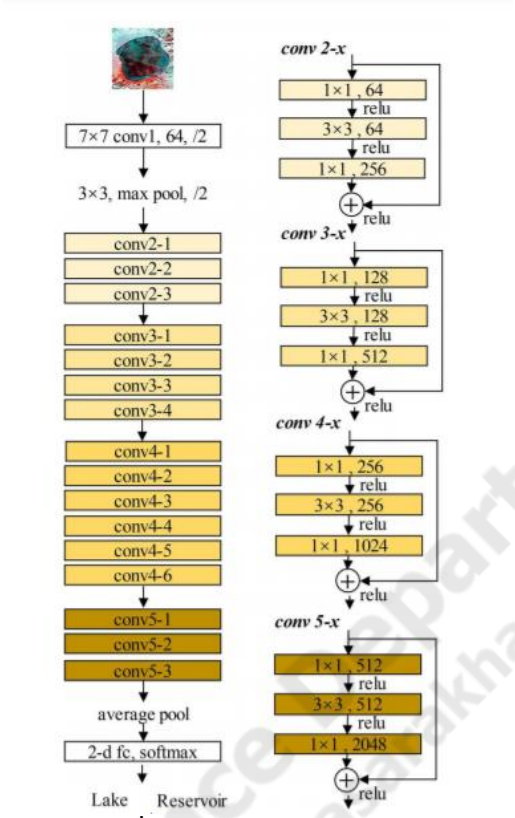
ผลที่ได้จากการจำแนกภาพโดยใช้ Inception ด้วยการฝึกโดยใช้ข้อมูลทั้งหมด 15 รอบ ได้ผลดังนี้



ภาพประกอบที่ 3.34 ตัวอย่างผลการทำนายโดยใช้ CNN Inception V3

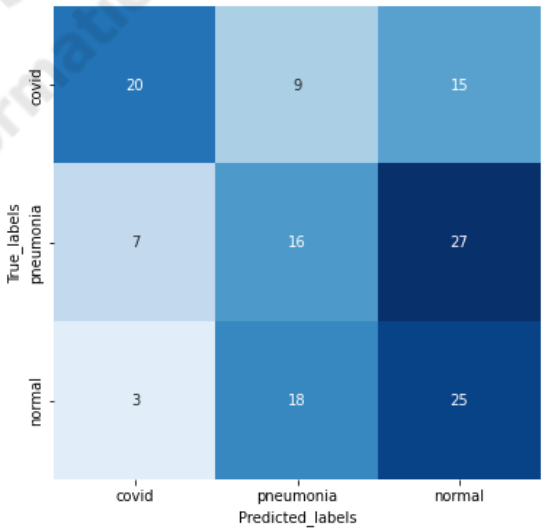
3.3.3 การจำแนกภาพเอกซ์เรย์ปอดโดยใช้ Model ResNet50-v2

ResNet50-v2 ResNet มาจาก Deep residual Network ถูกนำเสนอในงานวิจัย Deep residual learning for image recognition ซึ่งแก้ปัญหาเรื่อง vanishing gradient ซึ่งเกิดขึ้นกับโครงข่ายที่มี ความลึกค่อนข้างมากซึ่งมีจำนวนชั้นของ network ถึง 152 เลเยอร์ (8 เท่าของโมเดล VGG16) โดย ใช้เทคนิคการออกแบบ module ที่มีลักษณะทางลัดลงใน network ตัวโครงข่ายนี้ประกอบด้วยกัน 4 block โดยจำนวนที่มีพารามิเตอร์สำหรับฝึกทั้งหมดคือชั้นที่เราใช้เรียกชื่อ เช่น ResNet50 จะหมายถึงจำนวน 50 เลเยอร์ซึ่งจะอธิบายขนาดว่า [3, 4, 6, 3] ซึ่งคือ (3 + 4 + 6 + 3) x 3 = 48 ชั้น + 2 ชั้น = 50 ซึ่ง ResNet ที่นิยมใช้จะเป็น ResNet18, ResNet34, ResNet50, ResNet101 และ ResNet152 ซึ่งในงานวิจัยนี้ได้ใช้ ResNet50-v2 อ้างอิงงานวิจัย



ภาพประกอบที่ 3.35 ResNet50-v2 architecture

ผลที่ได้จากการจำแนกภาพโดยใช้ ResNet50-v2 ด้วยการฝึกโดยใช้ข้อมูลทั้งหมด 15 รอบ ได้ผลดัง**ภาพประกอบที่ 3.36**



ภาพประกอบที่ 3.36 ตัวอย่างผลการทำนายโดยใช้ CNN ResNet50-V2

3.4 การเปรียบเทียบเพื่อวัดประสิทธิภาพ

- การจำแนกโดยไม่ได้ทำ Pre - Processing

นำภาพที่ไม่ได้ทำการ Pre-Processing ใดๆ นำไปจำแนกโดย โดยใช้ CNN ทั้ง 3 รูปแบบ ได้แก่ InceptionV3 , VGG16 , ResNet 50-V2

- การจำแนกโดยการทำ Pre-Processing

นำภาพที่ทำการ Pre-Processing ทั้ง 6 วิธี ได้แก่

1. Transformation -> Contrast -> Deblur
2. Transformation -> Deblur -> Contrast
3. Contrast -> Deblur -> Transformation
4. Contrast -> Transformation -> Deblur
5. Deblur -> Contrast -> Transformation
6. Deblur -> Transformation -> Contrast

จากนั้น นำทั้ง 6 วิธีไปจำแนกโดย โดยใช้ CNN ทั้ง 3 รูปแบบ ได้แก่ InceptionV3 , VGG16 , ResNet 50-V2

ดังนั้น จะได้การจำแนกทั้งหมด 7 แบบ จากนั้น นำผลลัพธ์ การจำแนกโดยไม่ได้ทำ Pre – Processing และ การจำแนกโดยการทำ Pre - Processing ทั้ง 6 วิธี มาเปรียบเทียบกัน ว่าการทำ Pre-Processing ทั้ง 6 วิธี นั้น มีวิธีใด ที่ช่วยให้ CNN มีการจำแนกภาพได้มีประสิทธิภาพมากขึ้นหรือไม่ และวิธีใดเป็นวิธีที่ดีที่สุด โดยเปรียบเทียบด้วยค่าใน Confusion Matrix