# บทที่ 2

# ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

## 2.1 ทฤษฎีที่เกี่ยวข้อง

ในการรู้จำเอกสารจากภาพ จะประกอบด้วยขั้นตอนการทำงานหลักๆ 2 ส่วนได้แก่ การตรวจจับเอกสาร (Document Detection และการรู้จำข้อความบนแบบฟอร์มเอกสาร (Text Recognition) ดังนั้นทฤษฎีที่เกี่ยวข้องที่จะกล่าวถึงในบทนี้ ได้แก่ เทคนิคการประมวลผลเบื้องต้นกับภาพ และการรู้จำข้อความบนภาพเอกสาร

หลังจากอ่านไฟล์ภาพเข้ามาแล้วจะนำภาพไปทำ การแปลงภาพระดับเทา (Gray scale) จากนั้นจะเอาไปทำ การแปลงภาพ GaussianBlur และ เอาไปทำการแปลงภาพ Canny และ ทำการ Contour หาเฉพาะ วัตถุ ที่เป็น สี่เหลี่ยม แล้วทำการนำตำแหน่งที่ได้จาก ทำ Contour ไปทำการ Perspective Transformation และทำการ WarpPerspective เพื่อทำให้รูปที่ได้จากการทำ Perspective Transformation การมาตรงและทำการ resize เป็น width = 1357 height = 1920 เพื่อให้ภาพเหมาะกับการ

### 2.1.1 ภาษาที่เกี่ยวข้อง

ชุดตัวอักษรภาษาอังกฤษประกอบด้วยตัวพิมพ์ใหญ่ 26 ตัวอักษรและตัวพิมพ์เล็ก 26 ตัวอักษรตัวเลขอารบิกจำนวน 10 ตัวเลขชุดตัวอักษรภาษาไทย 44 พยัญชนะ 21 สระ (32 เสียงสระ) 4 วรรณยุกต์ (5 เสียง) 10 ตัวเลขไทยและ 2 ตัวแบ่งประโยค

| Type | Member |
|---|---|
| Consonants | a b c d e f g h i j k l m n o p q r s t u v w x y z A B C D E F G H I J K L M N O P Q R S T U V W X Y Z |
| Arabic digits | 0 1 2 3 4 5 6 7 8 9 |

**ภาพประกอบที่ 2.1** ชุดข้อมูลภาษาอังกฤษ

| Type | Member |
|---|---|
| Consonants | ก ข ฃ ค ฅ ฆ ง จ ฉ ช ซ ฌ ญ ฎ ฏ ฐ ฑ ฒ ณ ด ต ถ ท ธ น บ ป ผ ฝ พ ฟ ภ ม ย ร ล ว ศ ษ ส ห ฬ อ ฮ |
| Vowels | อะ อา อิ อี อึ อื อุ อู เอะ เอ แอะ แอ เอือ โอะ โอ เอาะ ออ เออะ เออ เอียะ เอีย เอือะ เอือ อัะ อัว ฮำ ไอ ไอ เอา ฤ ฤๅ ฦ ฦๅ |
| Thai Digits | ๐ ๑ ๒ ๓ ๔ ๕ ๖ ๗ ๘ ๙ |
| Tone Masks | อ่ อ้ อ๊ อ๋ |
| Special Symbols | ๆ ฯ |

**ภาพประกอบที่ 2.2** ชุดข้อมูลภาษาไทย

2.1.2 ขั้นตอนการ Pre-process

1) Pre-process คือการทำการเตรียมรูปภาพ ได้แก่

1.1 การเตรียมภาพเพื่อประมวลผลเบื้องต้น (Pre-Processing)

โดยปกติภาพทั่วไปหรือเอกสารโบราณที่นำเข้านั้นอาจจะมีสัญญาณรบกวน (Noise) เกิดขึ้นในเอกสาร ทำให้ภาพอาจไม่ชัด หรือมีริ้วรอยต่าง ๆ ซึ่งอาจจะส่งผลกับการประมวลผลในขั้นตอนการตรวจจับภาพ (Text detection) ดังนั้นขั้นตอนนี้จึงเป็นขั้นตอนการกำจัดสัญญาณรบกวน (Noise) ซึ่งเทคนิคที่ใช้ในการกำจัดสัญญาณรบกวน (Noise) ได้แก่การแปลงภาพระดับเทา (Gray scale) [1] ภาพระดับเทาคือ ภาพเกรย์สเกลหรือภาพระดับสีเทา คือภาพ ขาว-ดำ-เทา โดยจะมีระดับความเข้มจากการแปลงสีเท่าคือ 0–255 ( 8-bit ) - ภาพเกรย์สเกลเกิดจากการแปลงภาพสี RGB มาเป็น Grayscale โดยใช้สูตรทางคณิตศาสตร์ Gray = 0.299*R+0.587*G+0.114*B. ซึ่งระบบเราจะทำการเปลี่ยนภาพจาก RGB เป็นภาพระดับเทาจะทำให้แต่ละจุดภาพเหลือเพียงค่าความเข้มของสีที่มีค่าตั้งแต่ 0 ถึง 255 ซึ่งเทคนิคที่ใช้ในการกำจัดสัญญาณรบกวนนี้จะใช้การกรองภาพแบบเกาเซียน (Gaussian) มีลักษณะเป็นการแจกแจงสัญญาณรบกวนตามรูปแบบของฟังก์ชั่นความหนาแน่นของความน่าจะเป็นของเกาส์เซียน และมีรูปแบบของสัญญาณรบกวนมีลักษณะของระฆังคว่ำ

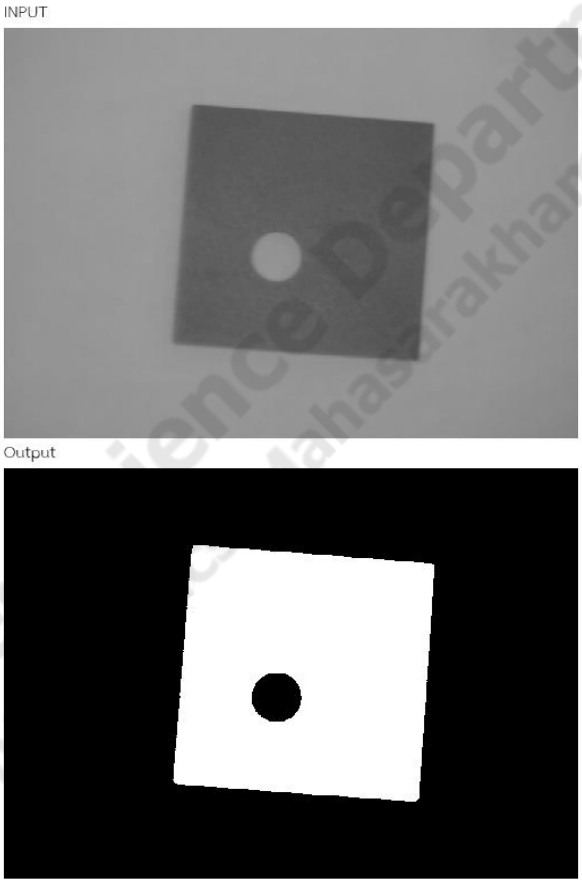

**ภาพประกอบที่ 2.3** ตัวอย่างไดเลชัน (Dilation)

การทำไดเลชัน (Dilation) [8] ใช้ในการเพิ่มขนาดของรูปร่างของภาพนำเข้า ดังนั้นผลลัพธ์เมื่อถูกดำเนินการจะทำให้วัตถุภายในภาพขยายขนาดใหญ่ขึ้น รวมทั้งเหมาะสมที่จะใช้ในการเชื่อมหรือขยายจุดภาพที่ขาดหายไป


**ภาพประกอบที่ 2.4** ตัวอย่างหลังทำอิโรชัน (Erosion)

การทำอิโรชัน (Erosion) [8] เป็นการกัดกร่อน หรือลดขนาดของวัตถุ ดังนั้นจึงสามารถประยุกต์ใช้ในการกำจัดขอบของวัตถุภายในภาพ และกำจัดข้อมูลขนาดเล็กๆออกจากภาพได้

การแปลงภาพสองระดับ (Thresholding) [2] การดำเนินการกำหนด Thresholding จะทำงานได้ดีกับภาพระดับสีเทาหรือภาพขาวดำ ผลลัพธ์จะเป็นภาพไบนารีที่แสดงถึงการแบ่งส่วนภาพ (image segmentation) ของภาพระดับเทา (Gray scale) ค่าพิกเซลแต่ละค่าจะถูกเปรียบเทียบกับค่าเกณฑ์หรือเรียกว่าค่าเธรดโชลด์ (Threshold) [2] ในการใช้งานอย่างง่าย การแบ่งส่วนจะถูกกำหนดโดยพารามิเตอร์เดียวที่เรียกว่าเกณฑ์ความเข้ม ในครั้งเดียว แต่ละพิกเซลในรูปภาพจะถูกเปรียบเทียบกับเกณฑ์ หากความเข้มของพิกเซลสูงกว่าเกณฑ์ พิกเซลจะถูกตั้งค่าเป็นสีขาวในเอาต์พุต หากน้อยกว่าเกณฑ์ จะเป็นสีดำ




**ภาพประกอบที่ 2.5** ตัวอย่างการแบ่งระดับ

2) การรู้จำลายมือ (Handwritten recognition) [17] ในกระบวนการรู้จำอักขระทั้งหมดนี้โดยทั่วไปใช้ Convolutional Neural Network หรือ CNN และสามารถพิจารณาข้อความเป็นลำดับของอักขระและสำหรับการแก้ปัญหาดังกล่าวเรามักใช้เครือข่าย Long Short-Term Memory หรือ LSTM เป็นรูปแบบที่นิยมของ RNN หรือ Recurrent Neural Networks

2.1.3 การรู้จำตัวอักษร (Optical Character Recognition หรือ OCR)

OCR ย่อมาจาก Optical Character Recognition (OCR) [15] ซึ่งเป็นกระบวณการของการแปลงสื่อสิ่งพิมพ์ เช่น กระดาษ นิตยสาร สัญญา หรือข้อมูลอะไรก็ตามที่อยู่ในรูปของเอกสารกระดาษ ให้กลายเป็นข้อความให้มีความฉลาดมากขึ้นกว่าการเป็นข้อความธรรมดา หรือสามารถบันทึกไปเป็นไฟล์ประมวลผลคำที่สามารถแก้ไขได้ง่ายและบันทึกเก็บไว้ได้ เทคโนโลยี OCR ช่วยเพิ่มประสิทธิภาพอย่างมากในการจัดเก็บข้อมูล แบ่งปันข้อมูลและแก้ไขข้อมูล โครงสร้างของระบบ OCR ประกอบไปด้วยขั้นตอนการทำงานหลัก 2 ขั้นตอน ได้แก่

1.การประมวลผลขั้นต้น (Pre-process) เช่น การปรับแต่งข้อมูล (Normalization) การกรองข้อมูลแทรกซ้อน (Noise Filtering) การตรวจจับวัตถุ (Object Detection) เป็นต้น ซึ่งวิธีการประมวลผลขั้นต้นได้กล่าวมาแล้วในหัวข้อ 2.1.1 และ 2.1.2

2. การรู้จำตัวอักษร (Character Recognition) เช่น วิธีทางโครงข่ายประสาทเทียม และการเรียนรู้เชิงลึก เป็นต้น

ปัจจุบันได้มีเครื่องมือที่ได้รับการพัฒนาทางด้าน OCR มาใช้กันอย่างแพร่หลาย รองรับหลากหลายภาษา เช่น Tesseract ดังจะได้กล่าวต่อไปนี้

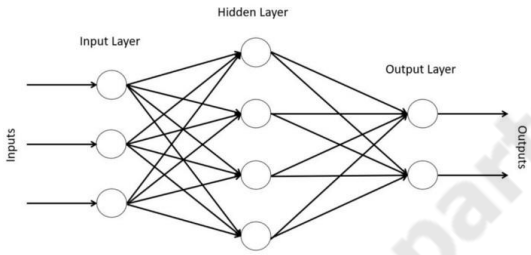
2.1.3.1 Tessreact

Tesseract [18] เป็นซอฟต์แวร์และไลบรารีที่ใช้ในการแปลงภาพข้อความที่มนุษย์เข้าใจให้ไปเป็นข้อความที่คอมพิวเตอร์เข้าใจ หรือเรียกอีกอย่างหนึ่งว่า OCR (Optical Character Recognition) ซึ่งเป็น Engine ใช้สำหรับรู้จำอักขระที่ถูกพัฒนาโดยบริษัท HP ระหว่างปี 1984-1985 เริ่มต้นจากโปรเจควิจัยในระดับปริญญาเอกในห้องปฏิบัติการณ์ของ HP โดยพัฒนาเพื่อนำไปทำเป็นเครื่องแสกนเนอร์ ซึ่งต่อมาในปี 2005 HP ได้เปิด Open Source โดยมี Google เป็นผู้สนับสนุน Tesseract ถือเป็นเครื่องมือ OCR ที่มีความแม่นยำสูงอีกชนิดหนึ่ง

สาเหตุที่ Tesseract ได้รับความนิยม เพราะเป็นซอฟต์แวร์เสรี (Free software) และมีประสิทธิภาพดี โดยการเรียกใช้งานผ่านทาง Command line ก็ได้ หรือนำไปเชื่อมต่อกับ API ของงานที่ทำได้ ซึ่งภาพที่ใช้เป็น Input ให้กับ Tesseract ต้องเป็นภาพที่มีการปรับแต่งมาให้เหมาะกับการอ่านข้อความคือหมุนมาค่อยข้างตรง มีการปรับแสงและสีให้อ่านได้ง่าย พื้นหลังสีขาวหรือสีอ่อน ตัวอักษรสีดำ Tesseract รุ่นที่รองรับภาษาไทยตั้งแต่รุ่นที่ 3 ขึ้นไป โดยในรุ่นที่ 4 สามารถใช้โมเดล Deep Learning แบบ LSTM ได้

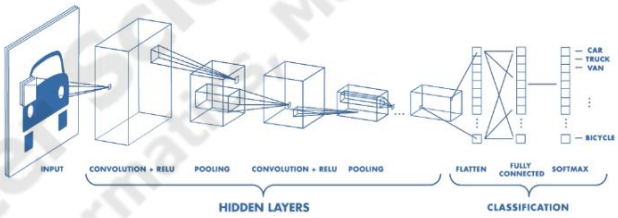
2.1.4 Convolution Neural Network

Convolutional Neural Network (CNN) [3] หรือ โครงข่ายประสาทแบบคอนโวลูชัน เป็นโครงข่ายประสาทเทียม แบบทั่วไปโดยจะประกอบไปด้วย ชั้นนำเข้า (Input Layer) ชั้นซ่อน (Hidden Layer) และชั้นผลลัพธ์ (Output Layer)




**ภาพประกอบที่ 2.6** Covolution Neural Network

ที่มา https://matteotor92.medium.com/convolutional-neural-network-theory-and-code-20bbe066fa48

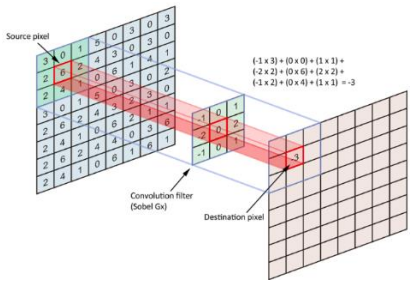



**ภาพประกอบที่ 2.7** Convolution Neural Network Layer

ที่มา https://matteotor92.medium.com/convolutional-neural-network-theory-and-code-20bbe066fa48

Convolution Neural Network ถูกสร้างขึ้นเพื่อแก้ปัญหานี้โดยทั่วไป CNN [3] จะใช้เวลาในการป้อนข้อมูลภาพและวิเคราะห์มันดังนั้นก็สามารถที่จะจำแนกประเภทวัตถุในปัจจุบันนั้น ซึ่งหมายความว่า CNN สามารถในการจับภาพข้อมูลเชิงพื้นที่ของภาพ ซึ่งจะประกอบไปด้วยชั้นคอนโวลูชันและชั้นพูล

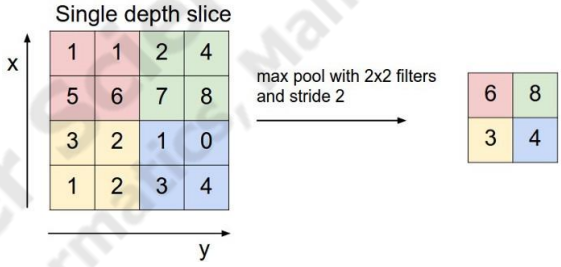
ชั้นคอนโวลูชัน (Convolution Layer) มีชื่อว่าตัวกรองหรือเคอร์เนลซึ่งมีเป้าหมายหลักในการนำฟิลเตอร์ไปใช้กับรูปภาพ การใช้เคอร์เนลกับเมทริกซ์อินพุตช่วยให้สามารถเน้นลักษณะของมันได้เช่นขอบวัตถุและอื่น ๆ และนี้คือการดำเนินการ Convolution




**ภาพประกอบที่ 2.8** Convolution Layer

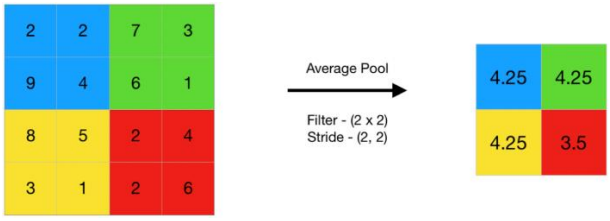
ที่มา : https://medium.com/botnoi-classroom/convolutional-neural-network-backbone-of-computer-vision-723ef3077219

ชั้นพูล ( Pooling layer ) เป็น Layer ที่ทำหน้าที่ในการปรับขนาดและปริมาณของข้อมูลตัวอย่าง(Sample) ให้ลดลงก่อนนำส่งเข้าสู่ Layer ถัดไปเพื่อให้สามารถวิเคราะห์และเก็บรายละเอียดของภาพได้อย่างครบถ้วน โดยในปัจจุบันมีอยู่สองรูปแบบคือ Max pooling, Average pooling โดยกระบวนการทั้งสองสามารถอธิบายได้ด้วยแผนภาพดังนี้




**ภาพประกอบที่ 2.9** Max Pooling

ที่มา : https://medium.com/@duanenielsen/deep-learning-cage-match-max-pooling-vs-convolutions-e42581387cb9

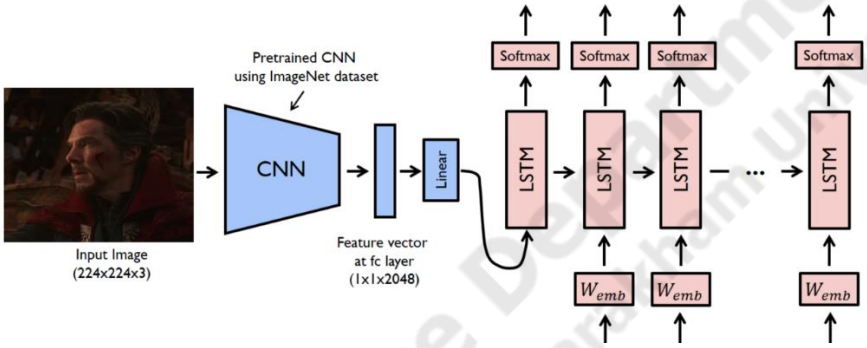



**ภาพประกอบที่ 2.10** Average Pooling

ที่มา : https://www.geeksforgeeks.org/cnn-introduction-to-pooling-layer/

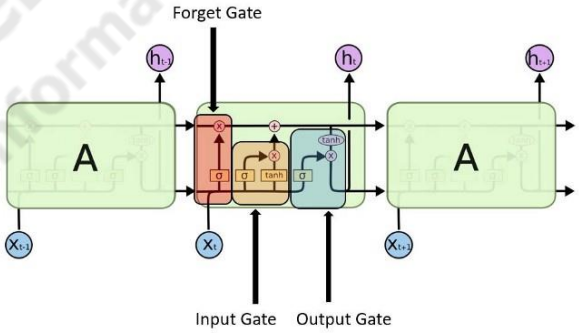
2.1.5 Long Short-Term Memory (LSTM)

เครือข่าย Long Short-Term Memory (LSTM) [18] เป็นเครือข่ายประสาทที่เกิดซ้ำซึ่งได้รับการแก้ไขซึ่งช่วยให้จดจำข้อมูลที่ผ่านมาในหน่วยความจำได้ง่ายขึ้น ปัญหาการไล่ระดับสีที่หายไปของ RNN ได้รับการแก้ไขแล้วที่นี่ LSTM เหมาะอย่างยิ่งในการจำแนกประมวลผลและคาดการณ์อนุกรมเวลาตามระยะเวลาที่ไม่ทราบระยะเวลา มันฝึกโมเดลโดยใช้การขยายพันธุ์ย้อนกลับ ในเครือข่าย LSTM มีสามประตู:




**ภาพประกอบที่ 2.11** Long Short-Term Momory (LSTM) ที่ 1

ที่มา : https://medium.com/analytics-vidhya/introduction-to-image-caption-generation-using-the-avengers-infinity-war-characters-6f14df09dbe5

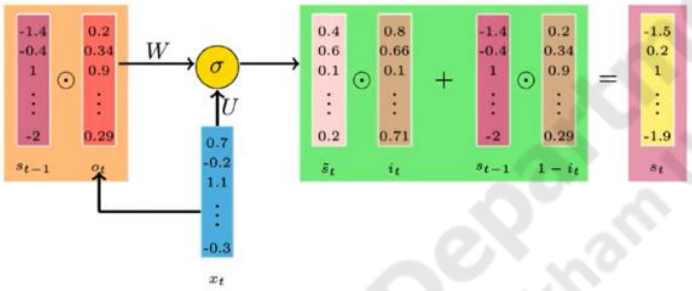



**ภาพประกอบที่ 2.12** Long Short-Term Momory (LSTM) ที่ 2

ที่มา : https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/

2.1.6 Gated Recurrent Units (GRU)

LSTM [18] สามารถมีได้หลายรูปแบบและ GRU [18] ก็เป็นหนึ่งในนั้น GRU [18] พยายามที่จะลดต้นทุนการคำนวณใน GRU โดยมีเอาท์พุตที่ควบคุมสัดส่วนของข้อมูลที่จะถูกส่งไปยังสถานะที่ซ่อนอยู่ถัดไปนอกจากนี้ยังมี Input Gates ที่มีการควบคุมการไหลของข้อมูลอินพุตปัจจุบันและไม่เหมือนกับ RNN ที่ไม่ใช้ Forget Gates




**ภาพประกอบที่ 2.13** Gated Recurrent Units (GRU)

ที่มา : https://towardsdatascience.com/long-short-term-memory-and-gated-recurrent-units-explained-eli5-way-eff3d44f50dd

เพื่อลดเวลาในการคำนวณเราลบ Gated และทิ้งข้อมูลที่ใช้

สมการที่ใช้สำหรับ GRU คือ:

Gates:

$$o_t = \sigma(W_o s_{t-1} + U_o x_t + b_o) \quad (2.1)$$

$$i_t = \sigma(W_i s_{t-1} + U_i x_t + b_i)$$

States:

$$\tilde{S}_t = \sigma(W(o_t * s_{t-1}) + U_{x_t} + b) \quad (2.2)$$

$$S_t = (1 * i_t) * s_{t-1} + t_i * \tilde{s_t})$$

ประเด็นสำคัญ

LSTM & GRU ได้รับการแนะนำเพื่อหลีกเลี่ยงความจำระยะสั้นของ RNN

LSTM ลืมโดยใช้ Forget Gates

LSTM จำโดยใช้ Input Gates

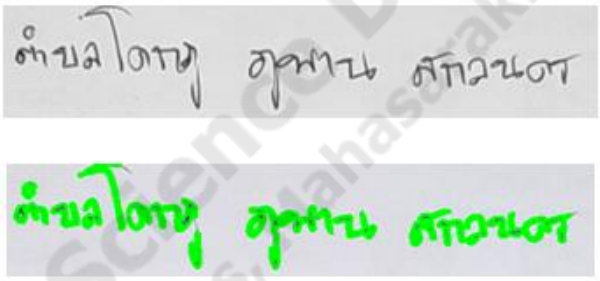
LSTM เก็บหน่วยความจำระยะยาวโดยใช้ Cell State

LSTM รวดเร็วแต่มีความแม่นยำน้อยกว่า GRU

2.1.7 การตรวจจับข้อความด้วยการ Contour

การ Contour [13] เป็นเหมือนการหาวัตถุสีดำจากพื้นหลังสีขาว โดยขั้นตอนการ Contour [13] นั้น จะหาพิกเซลสีดำและประกาศเป็น "จุดเริ่มต้น" ของพิกเซล ตำแหน่งของพิกเซล "เริ่มต้น" สามารถทำได้หลายวิธี ซึ่งหนึ่งในนั้นจะทำโดยเริ่มต้นที่มุมซ้ายด้างล่างของตาราง การสแกนพิกเซลจะสแกนจากด้านล่างขึ้นไป จากคอลัมน์ซ้ายสุดไปคอลัมน์ขวาสุดและจะดำเนินการต่อไปเรื่อยๆ จนกว่าจะพบพิกเซลสีดำ เมื่อพบพิกเซลสีดำก็จะกำหนดจุดที่พบปัจจุบันเป็นจุด "เริ่มต้น" ทั้งนี้ทั้งนั้นเราสามารถเลือกพิกเซลเริ่มต้นได้ตามความพึงพอใจ โดยจะมีข้อจำกัดในการเลือกพิกเซลเริ่มต้น ดังต่อไปนี้

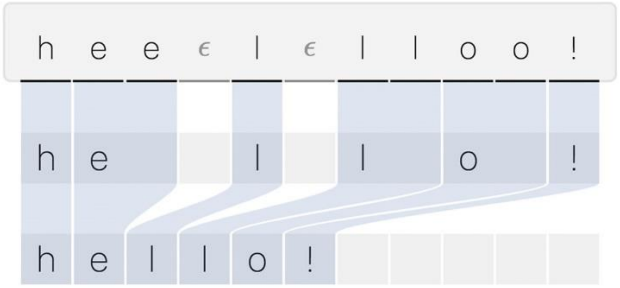
ข้อจำกัดที่สำคัญเกี่ยวกับทิศทางการเลือกพิกเซล "เริ่มต้น" สามารถเลือกจุดเริ่มต้นที่เป็นพิกเซลสีดำที่ตำแหน่งใดก็ได้ แต่มีข้อจำกัดอยู่ว่าในกรณีที่เลือกพิกเซลเพื่อกำหนดให้เป็นจุดเริ่มต้น พิกเซลที่ใกล้เคียงอยู่ทางด้านซ้ายของจุดเริ่มต้นต้องไม่ใช่พิกเซลสีดำ แต่ต้องเป็นพิกเซลสีขาว



**ภาพประกอบที่ 2.14** ตัวอย่างก่อนทำและหลังทำ Contour

2.1.8 Connection Temporal Classification (CTC)

เนื่องจากงานนี้จะเป็นการรู้จำตัวอักษรในการจัดลำดับก่อนหลังว่าอักษรใดมาก่อนและหลังจึงต้องใช้ CTC [21] ในการพิจารณารูปแบบการจัดลำดับของข้อมูลลายมือเขียนที่มีความต่อเนื่องกัน




**ภาพประกอบที่ 2.14** การทำงานของ CTC

ที่มา : https://distill.pub/2017/ctc/

โดยเงื่อนไขความน่าจะเป็นของตัวอักษรที่ติดกันสามารถหาได้ โดยผลรวมของความน่าจะเป็นที่รวมกันได้มากที่สุดของแต่ละ Time-Step ทำหน้าที่ ในการจัดเรียงตัวอักษรให้เป็นคำที่มีลักษณะที่สวยงามโดยตัดช่องว่างและการตัดอักขระที่มีลักษณะติดกันจะทำการตัดตัวที่ซ้ำออก โดยใช้เงื่อนไขความน่าจะเป็นของตัวอักษรที่ติดกันสามารถหาได้โดยหาผลรวมของความน่าจะเป็นที่รวมกันได้มากที่สุดของแต่ละ Time-Step จากสูตรดังต่อไปนี้

$$\sum_{(x,z)\in S} \log_{eP}(z|x) \quad (2,3)$$

โดยที่

X คือ Training Simple

S คือ Training Data

Z คือ Generated Sequence

## 2.2 งานวิจัยที่เกี่ยวข้อง

งานวิจัยของ ศุภรัตน์ อภิวงศ์โสภณ [1] การรู้จำตัวอักษรภาษาไทยที่เขียนด้วยลายมือในแบบฟอร์ม (Thai Character Recognition of Handwritten in Forms) เกี่ยวกับงานวิจัยนี้ รู้จำตัวอักษรภาษาไทยที่เขียนด้วยลายมือในแบบฟอร์มประวัติส่วนตัว ICHI.PRO [2] การรู้จำ OCR ด้วย Python และ Tesseract เกี่ยวกับงานวิจัยนี้ พื้นฐานสำหรับ OCR และ Tesseract Engine

งานวิจัยของ Pakapoom Mookdarsanit, Lawankorn Mookdarsanit [3] ThaiWritten Net: Thai Handwritten Script Recognition using Deep Neural Networks เกี่ยวกับงานวิจัยนี้ รู้จำตัวอักษรไทยด้วย Deep Neural Network กับ Deep Belief Network (DBN) สำหรับลายมือภาษาไทย นอกจากนี้ยังเปรียบเทียบ ThaiWrittenNet กับ machine learning แบบดั้งเดิมด้วยการสกัดคุณลักษณะที่สร้างขึ้นด้วยมือโดยชุดข้อมูลหลัก ของข้อมูลทั้งมี 9,282 ภาพลายมือภาษาไทยที่ประกอบไปด้วย 87 คลาส: 44 พยัญชนะ 18 สระ 5กำกับเสียง วรรณยุกต์ 4 ตัว เลขไทย 10 ตัว และสัญลักษณ์พิเศษ 6 ตัว ทำการฝึกอบรม 7,426 ภาพและการทดสอบภาพ 1,856 ภาพ จากการทดลองใช้ผลลัพธ์ ConvNet มีประสิทธิภาพดีกว่า machine learning แบบเดิมยิ่งไปกว่านั้น สามารถใช้ DBN เพื่อลดความซับซ้อนของเครือข่ายและทำให้ความแม่นยำสูงขึ้น

โครงงานปริญญานิพนธ์ของ นางสาวกมลทิพย์ เทศทอง [4] การประมวลผลลายมือเขียนเป็นตัวพิมพ์อัตโนมัติ เวอร์ชั่น 2 (Automatic Handwritten Recognition; Auto HWR v.2) ได้ทำการแปลงรูปภาพลายมือเขียนภาษาไทยออกมาเป็นตัวพิมพ์อัตโนมัติ โดยออกแบบสถาปัตยกรรมในการรู้จำตัวอักษรด้วยการเรียนรู้เชิงลึก (Deep Learning) โดยมีสถาปัตยกรรมการทำงานของ CNN ร่วมกับ

GRU และ CTC จะมี CNN จำนวน 5 ชั้น และ GRU แบบ Bidirectional GRU 3 ชั้น จากนั้นได้ทำการวัดประสิทธิภาพโดยใช้ Levenshtein edit distance และใช้ชุดข้อมูลเฉพาะลายมือเขียนทั้งสิ้น 10,792 รูปภาพ ข้อมูล Generate Font จำนวนทั้งหมด 6,432 ข้อความ จากผลการทดสอบพบกว่าในการใช้ตัวถอดรหัสแบบ Beam Search Decoding ร่วมกับ LM มีค่า Character Error Rate อยู่ที่ 2.53% และการใช้ Best Path Decoding มีค่า Character Error Rate อยู่ที่ 2.59%