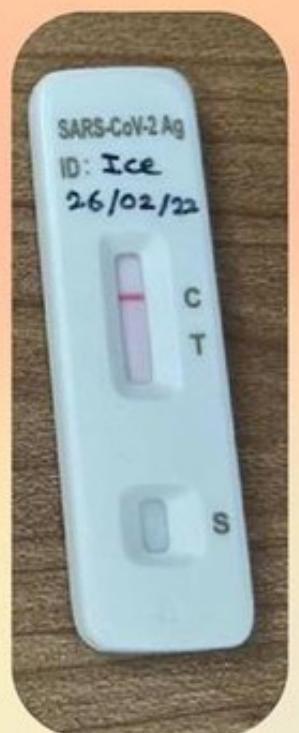
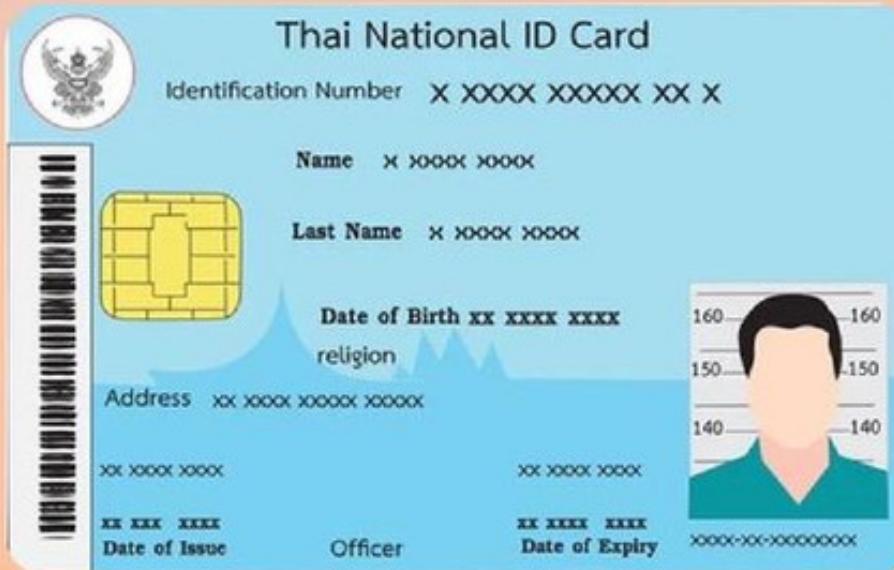


# ATK-OCR Classification (AOC)

Presented by Tanaanan Chalearnpan .M  
(Lucky Doves) AI Builders #2

ถ่ายผลตรวจ ATK คู่บัตรประชาชน



# What is AOC ?

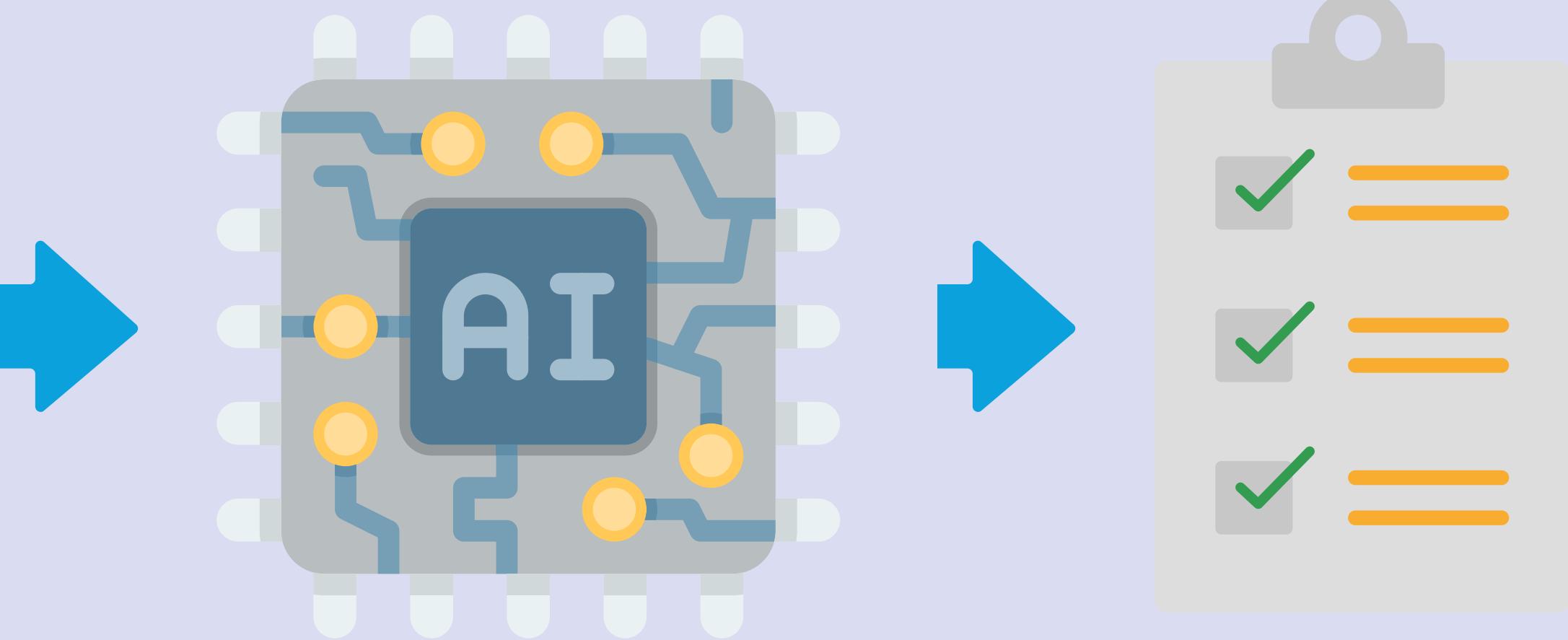


# What is AOC ?

ระบบที่สามารถคัดกรองผลตรวจเชื้อของ COVID-19 ได้ผ่าน กีตตรวจ ATK (Antigen Test Kit) ควบคู่กับบัตรประชาชน จากรูปภาพได้โดยอัตโนมัติ



Idcard + ATK

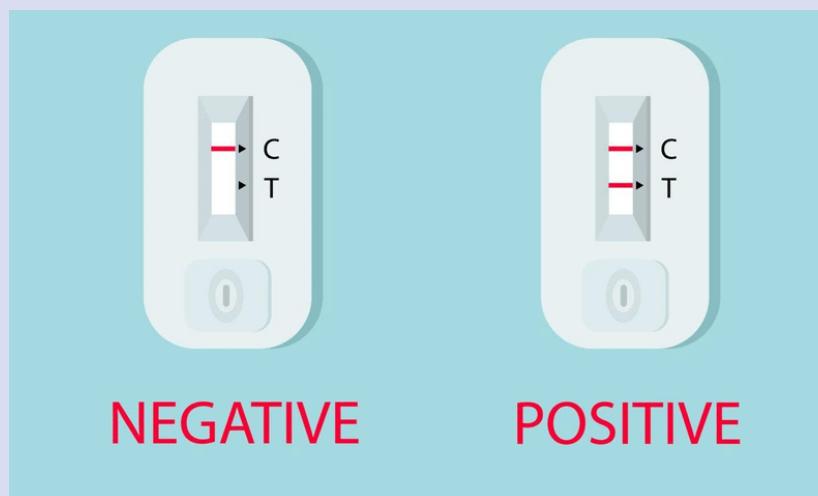


AOC model

ATK result  
Idnum  
name, lastname

# Inside AOC model ?

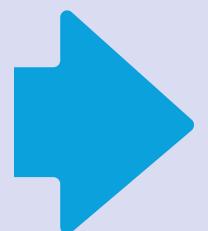
ระบบที่สามารถคัดกรองผลตรวจเชื้อของ COVID-19 ได้ผ่าน กีตตรวจ ATK (Antigen Test Kit) ควบคู่กับบัตรประชาชน จากรูปภาพได้โดยอัตโนมัติ



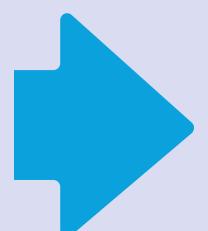
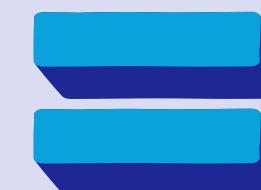
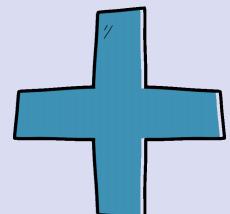
Antigen Test Kit



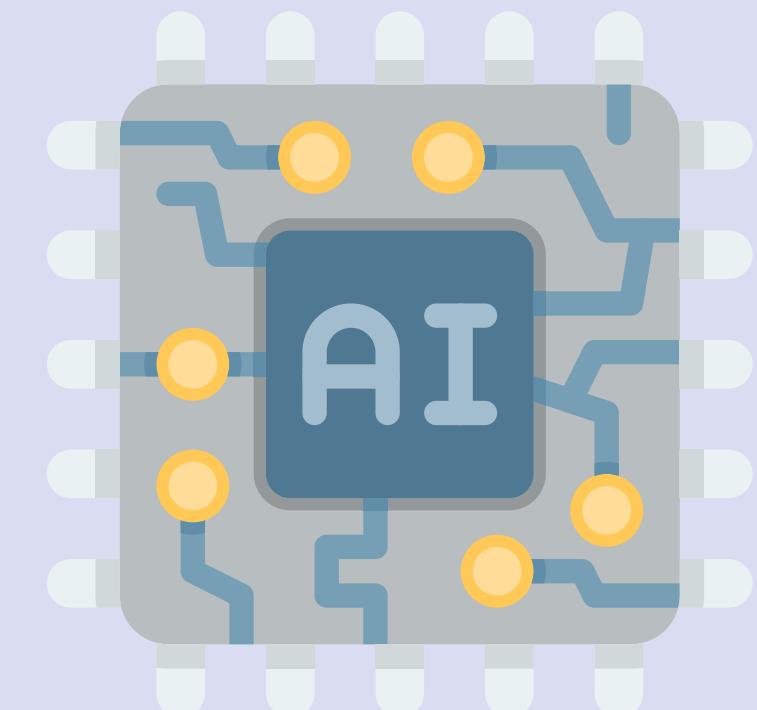
Identification Card



Object detection

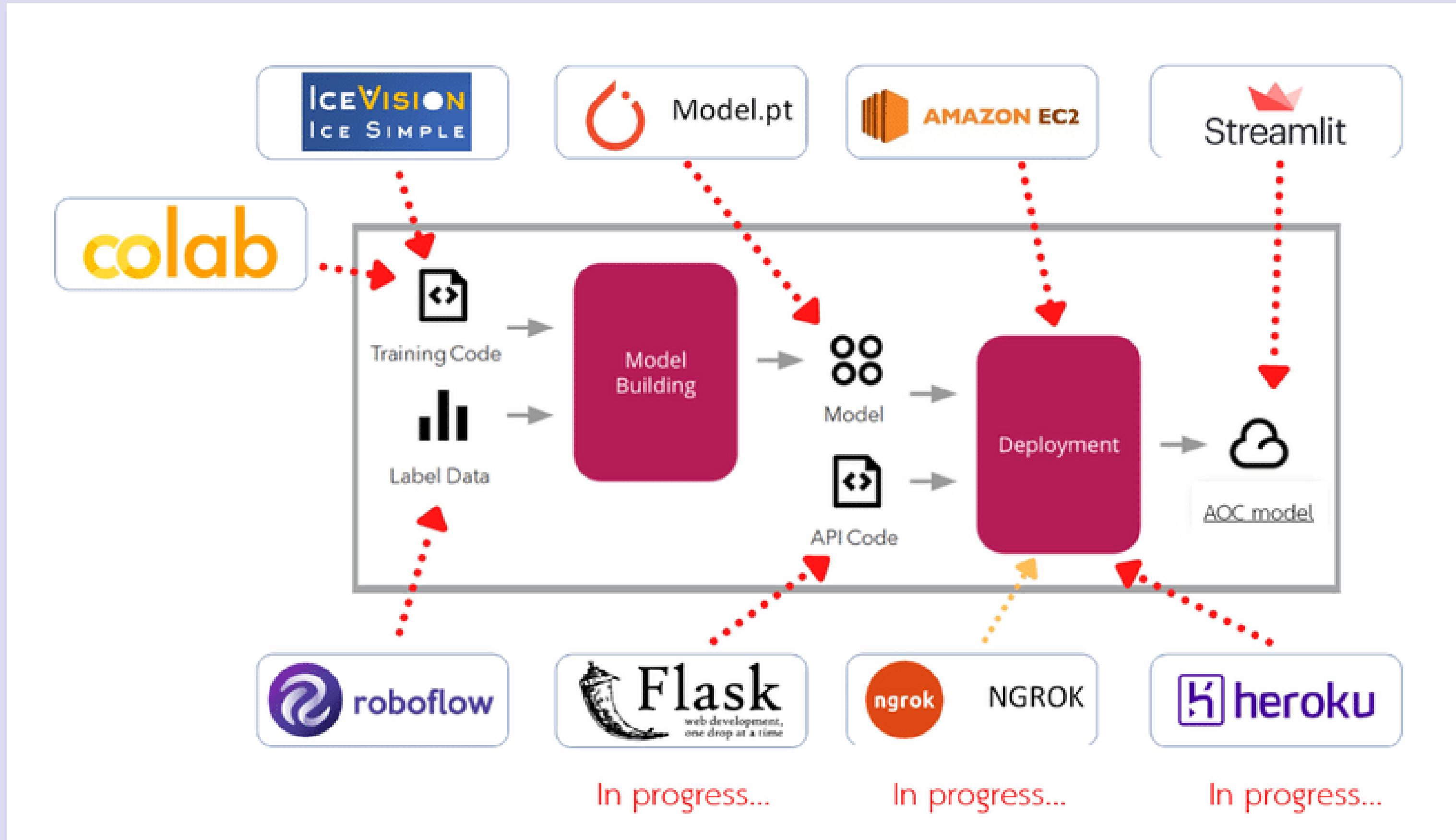


OCR + NLP + Edit distance  
(Rule-based system)

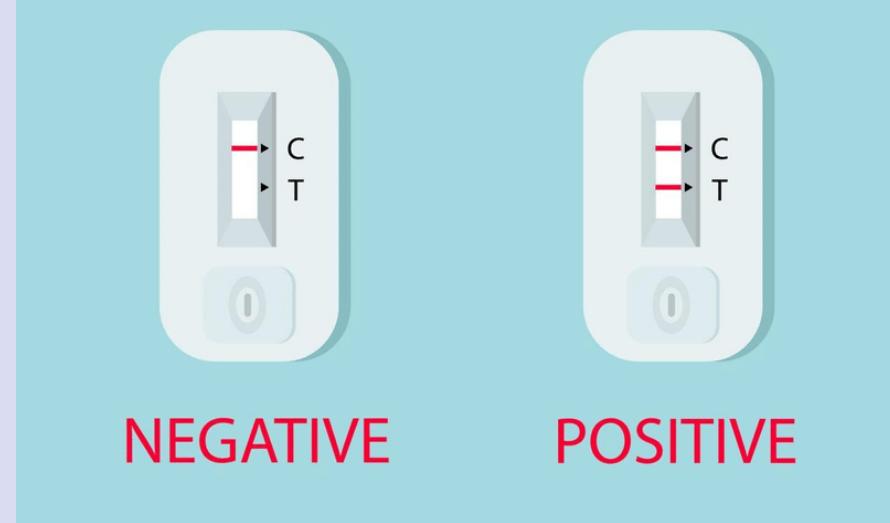


AOC model

# Pipeline



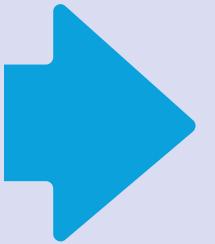
# Inside AOC model



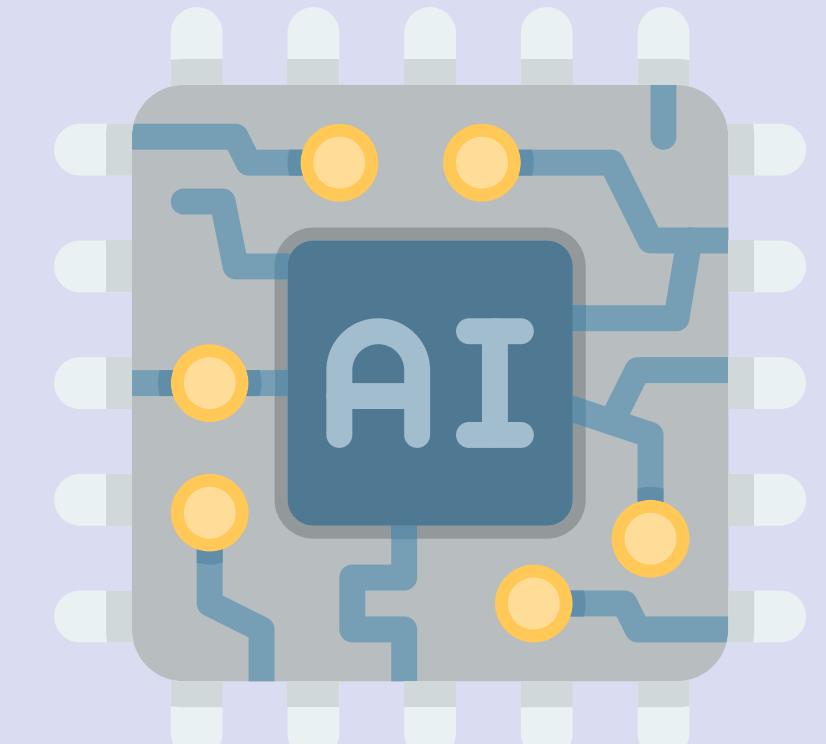
Antigen Test Kit



Identification Card



Object detection



OCR + NLP + Edit distance  
(Rule-based system)

AOC model

# Problem Statement



# Problem Statement (ໃກ້ຕົວ).

ss. ເປີດເທອນຕ້ອງກາຣທີ່ຈະບັນທຶກພລຕຣວຈໂຄວິດຂອງ  
ນັກເຮັຍນແຕ່ລະຫ້ອງ / ໂຮງເຮັຍນ ?



ບັນທຶກພຍັງໃຈ?.....

# Problem Statement

1. เปิด album Line ที่สร้างไว้ (1-3 วินาที)
2. ให้นักเรียนแบบภาพผลตรวจ ATK คู่กับ บัตรประชาชนลง album (5 - 10 วินาที)  
done ! (6- 15 วินาที)

ขึ้นตอนก็มีแค่นี้หนัง ?

# Problem Statement

ใช้แล้วครับ ขึ้นตอนของนักเรียนที่ต้องส่งมีแค่นี้...

ແຕ່ !

คนที่บันทึกข้อมูล และ เช็คความถูกต้องหายนะครับ (พมเงอ...)

# Problem Statement

หน้าที่ของคนควบรวม / ตรวจสอบความถูกต้อง (คร่าวๆ) คือ

1. เปิด album Line
2. หาภาพนักเรียนใน album และ เช็คว่าครบไหม (5 - 10 วินาที)
3. เปิดภาพ
4. บันทึกข้อมูล ผลตรวจจาก ATK , ชื่อ-นามสกุล, เลขบัตรประชาชน  
ลงใน excel หรือ กระดาษ (30 วินาที)

เวลาต่อคน ประมาณ 35 - 40 วินาที

# Problem Statement

สมมติว่าในห้องมีสมาชิกสัก 40 คน

- เปลี่ยต่อ 1 คน ประมาณ 35 - 40 วินาที
- ถ้า 40 คน จะใช้เวลาไปประมาณ 23 - 26 นาที

\*\*\*(พิมพ์ถูกหมด + พิมพ์ด้วยความเร็วต่อเนื่อง + พิมพ์ไม่พลาดเลย)

ชึ่งโคลนนานนนน.....



# Problem Statement

จะดีกว่าไหมครับ... ถ้ามี

ระบบที่ผู้ใช้งานใส่รูปภาพ แล้วสามารถสรุป ผลตรวจ ATK, ชื่อ-นามสกุล และ  
เลขบัตรประชาชน ได้เร็วกว่า และ แม่นยำกว่า พร้อมบันทึกผลเข้าสู่ระบบได้เลย....

นั่นก็คือ.. AOC นั่นเอง



# Metrics and baselines (การวัดผลเปรียบเทียบกับปัจจุหา)



# Metrics and baselines

## จากแบบสำรวจ (คบ)

โดยให้คน 10 คน กรอก ผลตรวจ ATK , ชื่อ-นามสกุล และ เลขบัตรประชาชน  
จำนวน 10 ภาพได้ผลสรุปมาดังนี้

- ATK accuracy : 94.0 %
- Idnum accuracy : 84.0 %
- name-lastname accuracy : 85 %
- Mean accuracy : 88.0 %
- Time per each : 26.87 second

### แบบสอบถามในการทดสอบ Human timing

ในการนำข้อมูลไปอ้างอิงในโครงการ AI Builders ในโปรเจค ATK-OCR Classification (AOC)

 mjsalyjoh@gmail.com (not shared) [Switch account](#) 

\* Required

#### วิธีในการทำแบบสอบถาม

ในแต่ละข้อจะมีภาพ บัตรประชาชน คู่กับ ผลตรวจ ATK โดยให้ผู้ที่ทำแบบสอบถามกรอก

- ผลตรวจโควิด ว่า (ติด / ไม่ติด)
- เลขบัตรประชาชน 13 หลัก (ใส่แค่ตัวเลขไม่ต้องเว้นวรรคหรือใส่ "-")
- ชื่อ นามสกุล (ภาษาอังกฤษ ใช้การเว้นวรรคในการแบ่ง ชื่อ นามสกุล)

\*\* ข้อ 3 จะใส่ชื่อต้นพิมพ์ใหญ่ หรือ เลิกก์ได้

What is your nickname \*

Your answer

# Metrics and baselines

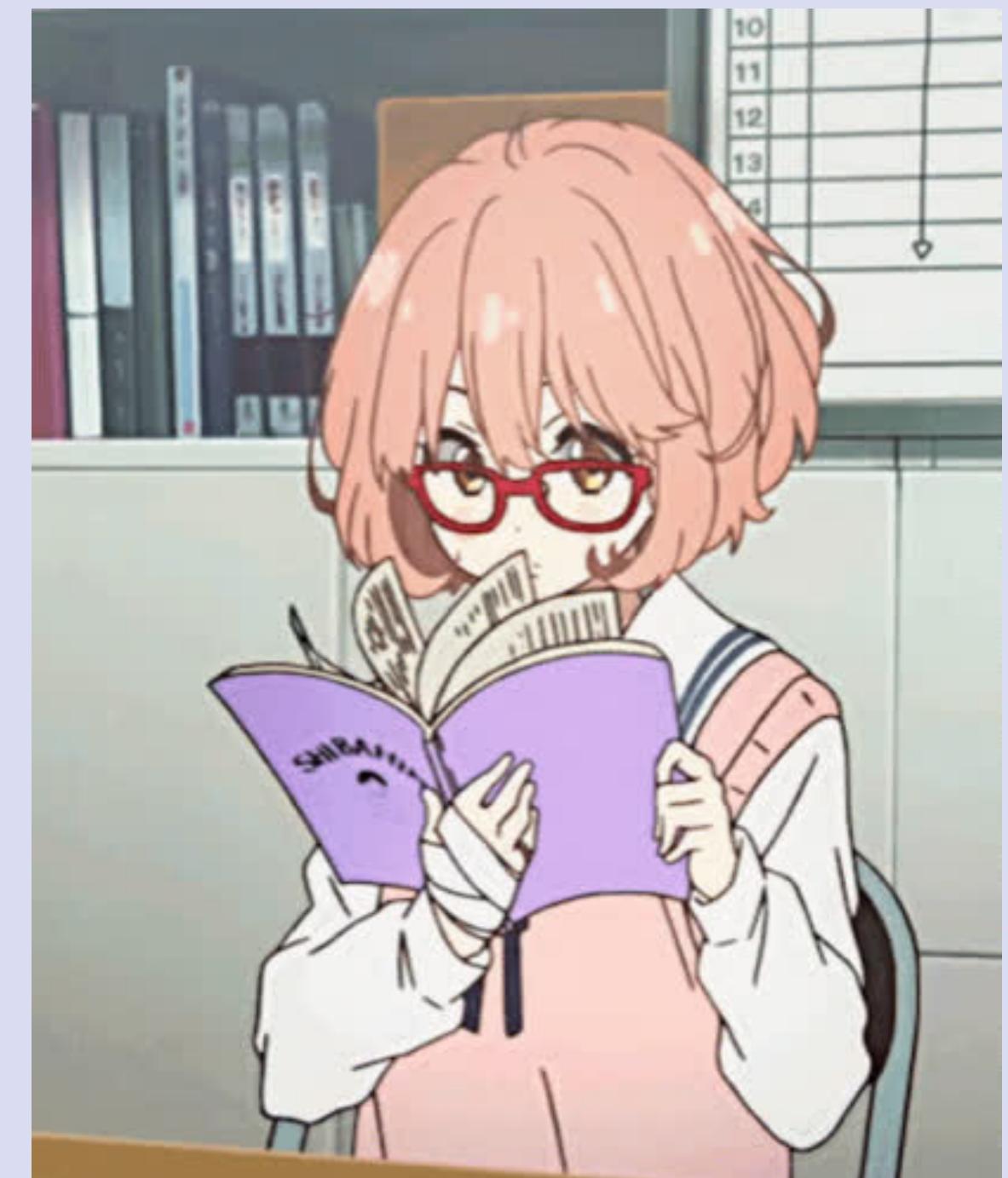
จากแบบสอบถามสามารถสรุปได้ว่า.....

model ที่เราทำนั้นควรที่จะได้ (Goal)

- ความเร็วที่น้อยกว่า 26.87 วินาที (baseline)
- Idnum or name-lastname accuracy มากราว 85 %
- Mean accuracy มากราว 88 %

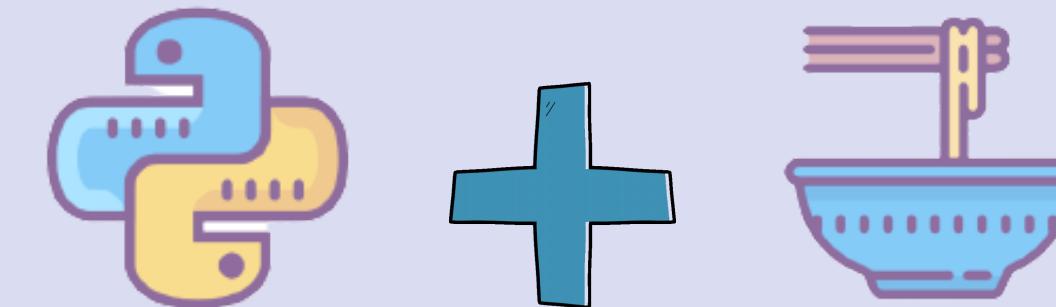
\*\*Pain point : ระยะเวลา + ความแม่นยำ\*\*

# Exploratory data analysis & Data collection, Data cleaning (การเก็บรวบรวมข้อมูล)



# Data collection

## Antigen Test Kit (ATK)



- Web scraping from stock image (BeautifulSoup)
- ขอทางโรงเรียน กับ ทางญาติๆ



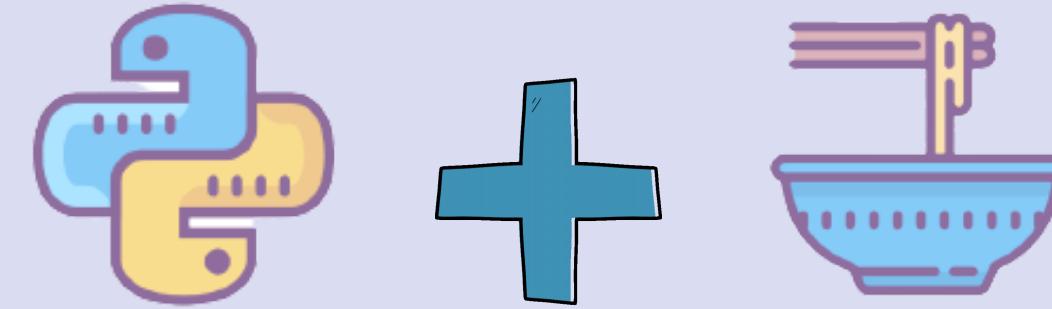
Stock image website

Web scraping

ATK image  
(dataset)

# Data analysis

## Antigen Test Kit (ATK)



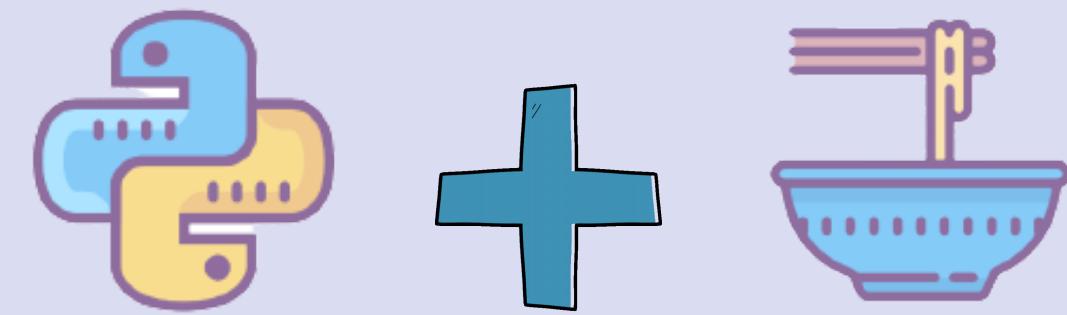
- ภาพที่ได้จากการ scraping มีบางส่วนที่ไม่สามารถใช้งานได้ เพราะ
  1. ไม่ใช่ภาพผลตรวจ ATK หรือ มีภาพ ATK หลายอันในภาพเดียว
  2. ภาพไม่ชัด หรือ อยู่ไกลเกินไป
  3. ภาพซ้ำกัน เพราะ scrape มาจากหลาย website



ตัวอย่างภาพที่ได้จากการ scraping ผ่าน website ต่างๆ ที่ไม่สามารถใช้เป็น dataset ได้

# Data cleaning

Antigen Test Kit (ATK)



- Label ឃោ + Coding (ខ្លឹមរាយឆ្នាំ)



= 2000 image

# Data collection

## Identification card (Idcard)

- ของญาติๆ กับของเพื่อนๆ
- \*\*\*(ใช้ในการวัดผลเท่านั้น ไม่ปล่อยเป็น public dataset)

# Data analysis

- ภาพที่บัตรมีรอยขีดข่วน ทำให้ตัว ชื่อ- นามสกุล หรือ  
เลขบัตรประชาชน มองเห็นได้ไม่ชัดเจน
- ภาพที่ถ่ายมาไม่ชัด และ มีแสงที่สว่างมากเกินไป หรือ มืดจนเกินไป

# Data cleaning\_(Image).

Identification card (Idcard)

- Label մօ (Back to basic....)

Very E Z desuuu.....



# Data cleaning\_(Text)

## 1. ใช้ EasyOCR ในการดึงข้อมูลจากบัตรประชาชน

```
Notebook OCR_result.txt ×
1 [ 'covid19', 'antigen', 'av', 'vigne.', '92879', 'thai', 'national', 'id', 'card', 'บ', '1', '2345', '67890', '123',
2 '4', 'wv', '6', 'a', 'vw', '6', 'coors', 'mt.', 'สหสันต์', 'ตานันทน์', 'name', 'master', 'tanaanan', 'last', 'name', 'chalermpan',
3 'สิริวันท์', '26', 'กม.', '2547', 'ssss', 'date', 'of', 'birth', '26', 'sep.', '2004', 'ก', '178', 'พากษา', 'พม', 'ເຊ', '໌', 'ທີ່ມູນ',
4 '7', 'ພູ້ທີ່', '2', 'ລ.ເພດວະອຸດິດ', '10', 'ລ.ຄວາມສັງ', 'a.', 'ພາກໃບ', 'q.', 'ໂຮງໝາຍ', '150', 'of', 'r', '31', 'ນ.ວ.', '2562', 'ເວລີ', '>',
5 'ນົມສ', '6', '25', 'ກນ.', '2570', 'lie', 'ຈົບອອນປະກາດ', 'o', 'ie', 'ໄສດຳ', 'ຈົບປະກາດຂອງອາຍ', '2', '31', 'ril.', '2019', 'ກົດ', 'ນູອຫຼາກ', 'an',
6 '25', 'sep.', '2027', 'date', 'of', 'issue', 'ເຈົ້າພະຈາກອອນລາຍ', 'date', 'of', 'ພາກ', 'k', '90110407311409', 'vie',
7 'l', 'mn', 'ee', 'ເວລີ', 'gages', 'ແກ້ວກະບົດ', 'ນ', 'ເວລີ', 'ເກົ່າ', 'ກອັນ', 'n', 'ເພີ່ມ', 'a', 'o', 'ເວລີ', 'ພົນຈະເພື່ອນບ',
8 | '9', 'a', 'ae' ]
```

```
import easyocr
#Easy OCR
# Doing OCR. Get bounding boxes.
reader = easyocr.Reader(['th','en'])
bounds = reader.readtext(name)
```

# Data cleaning\_(Text)

## 2. ใช้ PythaiNLP [Isthai]

แยกເອົາເລີພະສ່ວນທີ່ເປັນການອັງກຸບ

```
Notebook eng_name.txt ×
1 ['covid19', 'antigen', 'av', 'vigne.', '92879', 'thai', 'national', 'id', 'card', '1', '2345', '67890', '123', '4', 'wv',
2 '6', 'a', 'vw', '6', 'coors', 'mt.', 'name', 'master', 'tanaanan', 'last', 'name', 'chalermpan', '26', '၁၄၂၃', '2547',
3 'date', 'of', 'birth', '26', 'sep.', '2004', '၁၇၈', '၁၇၈', '၇', '၂', '၁၀', 'a.', 'q.', '၁၅၁၉', '150_၁', 'of', '31',
4 '2562', '>', '6', '25', '2570', 'lie', 'a', 'ie', '2', '31', '၂၁၉၁', 'an', '25', 'sep.', '2027', 'date', 'of',
5 'issue', 'date', 'of', '၁၉၀၁၀၄၀၇၃၁၁၄၀၉', 'vie', 'l', 'mn', 'ee', 'gages', '၁၃၃၅', '9', 'ae']
```

```
from pythainlp.util import isthai
```

# Data cleaning\_(Text)

## 3. ลบตัวอักษรที่ไม่พึงประสงค์ออกจากข้อความ (While loop)

```
Clean_syntax = ["""", "#", ".", "{", "}", "=", "/", "@", "#", "$", "--  
", "|", "%", "-", "(" , ")" , "¥", "[", "]" , ":", ";" ] # Symbol check for  
clean
```

Notebook Clean\_syntax.txt ×

```
1 ['covid19', 'antigen', 'av', 'vigne', '92879', 'thai', 'national', 'id', 'card',  
2 '1', '2345', '67890', '123', '4', 'wv', '6', 'a', 'vw', '6', 'coors', 'mt', 'name',  
3 'master', 'tanaanan', 'last', 'name', 'chalermpan', '26', 'กุย', '2547', 'date',  
4 'of', 'birth', '26', 'sep', '2004', '178', '7', '2', '10', 'a', 'q', 'กรุง',  
5 '150', 'of', '31', '2562', '>', '6', '25', '2570', 'lie', 'a', 'ie', '2',  
6 '31', 'ก1', '2019', 'an', '25', 'sep', '2027', 'date', 'of', 'issue', 'date',  
7 | 'of', '90110407311409', 'vie', 'l', 'mn', 'ee', 'gages', 'ເຂດ5', '9', 'ae' ]
```

# Modeling Validation and error analysis...

```
print("Hello World!")
```



**Modeling Validation and error  
analysis... (Identification Card)**

# Modeling\_analysis...



ส่วนที่ 1 : เลขบัตรประชาชน  
ส่วนที่ 2 : ชื่อ-นามสกุล

Image source : [https://aiforthai.in.th/service\\_cr.php](https://aiforthai.in.th/service_cr.php)

# Modeling\_analysis...

## เลขบัตรประชาชน

- ใช้ Rule-based python โดยจับหา pattern (เลข 13 หลัก)

\*\* ถ้า list ที่เจอเป็นตัวเลขหมวด แล้วครบ 13 หลัก ให้เป็น เลขบัตรประชาชน.

```
▶ idnum = ['1245678901234']
```

```
▶ idnum = ['1', '2345', '67890', '12', '4']  
▶ idnum = ['12345', '6789', '1', '11', '3']
```

# Modeling\_analysis...

## ชื่อ-นามสกุล

- ใช้ Rule-based python ในการหา pattern ตรวจจับ ชื่อ-นามสกุล

```
1                               2
name = [ 'name' , 'mr.' , 'tanaanan' , 'lastname' , 'chalearnpan' ]
```

โดยปกติตรงส่วนที่เป็นชื่อ นามสกุลที่ได้จาก EasyOCR + บัตรประชาชน เราจะได้รูปแบบที่มี 1. แท็กชื่อ (Name) กับ 2. แท็กนามสกุล (Lastname) คู่กัน

# Modeling analysis...

## ชื่อ-นามสกุล

- แบ่งได้เป็น 4 กรณี

```
1. name = [ 'name', 'mr.', 'tanaanan', 'lastname', 'chalearmpan' ]
```

```
2. name = [ 'name', 'mr.tanaanan', 'lastname', 'chalearmpan' ]
```

\*\* ตรวจจับง่าย เพราะ แท็กชื่อ (Name) กับ แท็กนามสกุล (lastname) สมบูรณ์

# Modeling\_analysis...

## ชื่อ-นามสกุล

- แบ่งได้เป็น 4 กรณี

3.

```
name = [ 'name', 'mL.', 'tanaanan', 'lastname', 'chalearmpang' ]  
name = [ 'name', 'mr.', 'tanaanan', 'lnsenawe', 'chalearmpang' ]
```

4.

```
name = [ 'nawe', 'mL.', 'tanaanan', 'lautname', 'chalearmpang' ]
```

\*\*ตรวจจับไม่ได้ เพราะ แท็กชื่อ กับ แท็กนามสกุล ขาดหาย และ ผิดเพี้ยนไป

# Modeling\_analysis...

## ชื่อ-นามสกุล

- จากรถที่ 3 กับ 4 เราจะใช้ Edit distance มาช่วยในการหา pattern

```
[6] !pip install editdistance
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: editdistance in /usr/local/lib/python3.7/dist-packages (0.5.3)

▶ import editdistance

def similar_check(word, tag_word):
    edit_val = editdistance.eval(word, tag_word)
    if (len(tag_word) == len(word)) and (edit_val <= 3):
        print(f'{word} equal {tag_word}; edit_val = {edit_val}')
    else:
        print(f'{word} not equal {tag_word} ; edit_val = {edit_val}')

similar_check('iautname', 'lastname')
'iautname' equal 'lastname'; edit_val = 2
```

# Validation analysis...

## ผลที่ได้จาก OCR บัตรประชาชน



```
[64] import time
    t1 = time.perf_counter()

    Get_Idcard_detail(file_path="/content/idcard_ex.jpg")

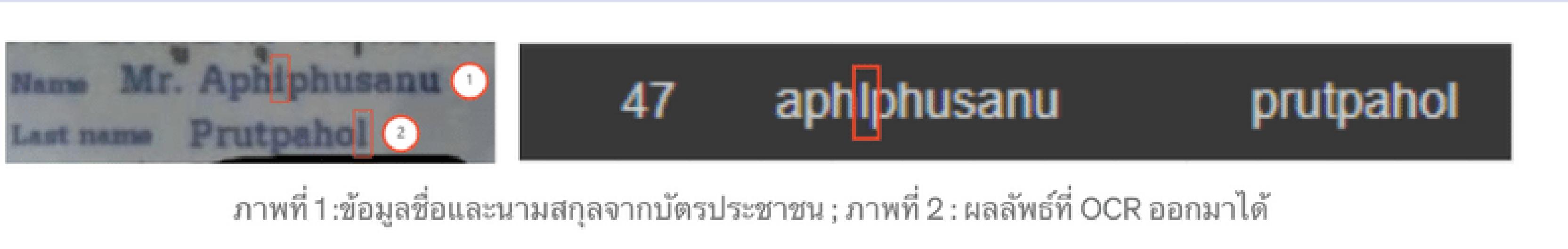
    t2 = time.perf_counter()
    print('time taken to run: {:.2f} sec'.format(t2-t1))

Process Completed!.....
{'id_num': '3411700830334'}
{'name': 'bunyang', 'lastname': 'lopez'}
time taken to run: 2.81 sec
```

\*\*ได้ระยะเวลาเฉลี่ยอยู่ที่ (2-5 วินาที) โดยได้ทิ้ง ชื่อ นามสกุล และ เลขบัตรประชาชน

# Error analysis...

## ข้อผิดพลาดที่พบจากการทำ OCR



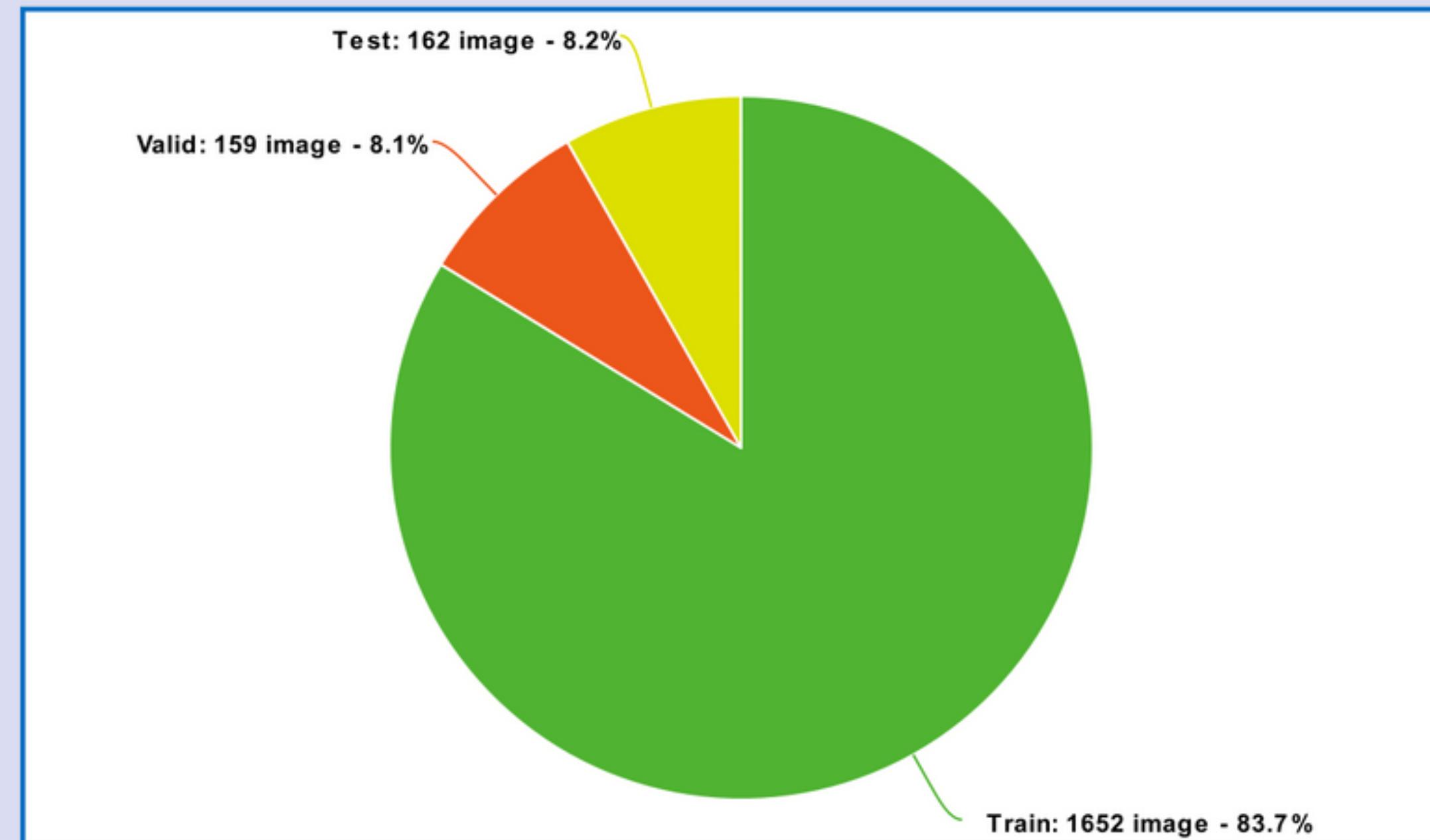
\*\*จากความสามารถสรุปได้ว่า model ยังมีความผิดพลาดในการแยกตัวอักษร i (ตัวอักษรไอเล็ก) กับ l (ตัวอักษรแอลเล็ก)

- ถ้ามีแสงที่สว่างมากเกินไป มืดเกินไป ไม่ชัด หรือ อยู่ไกลเกินไป อาจทำให้ผลการตรวจจับคลาดเคลื่อนเอ้าได้

**Modeling Validation and error  
analysis...  
(ATK : image classification)**

# Modeling\_analysis...

## Image classification with FastAI



- Train : Valid : Test split / 80 : 10 : 10 %

# Modeling\_analysis...

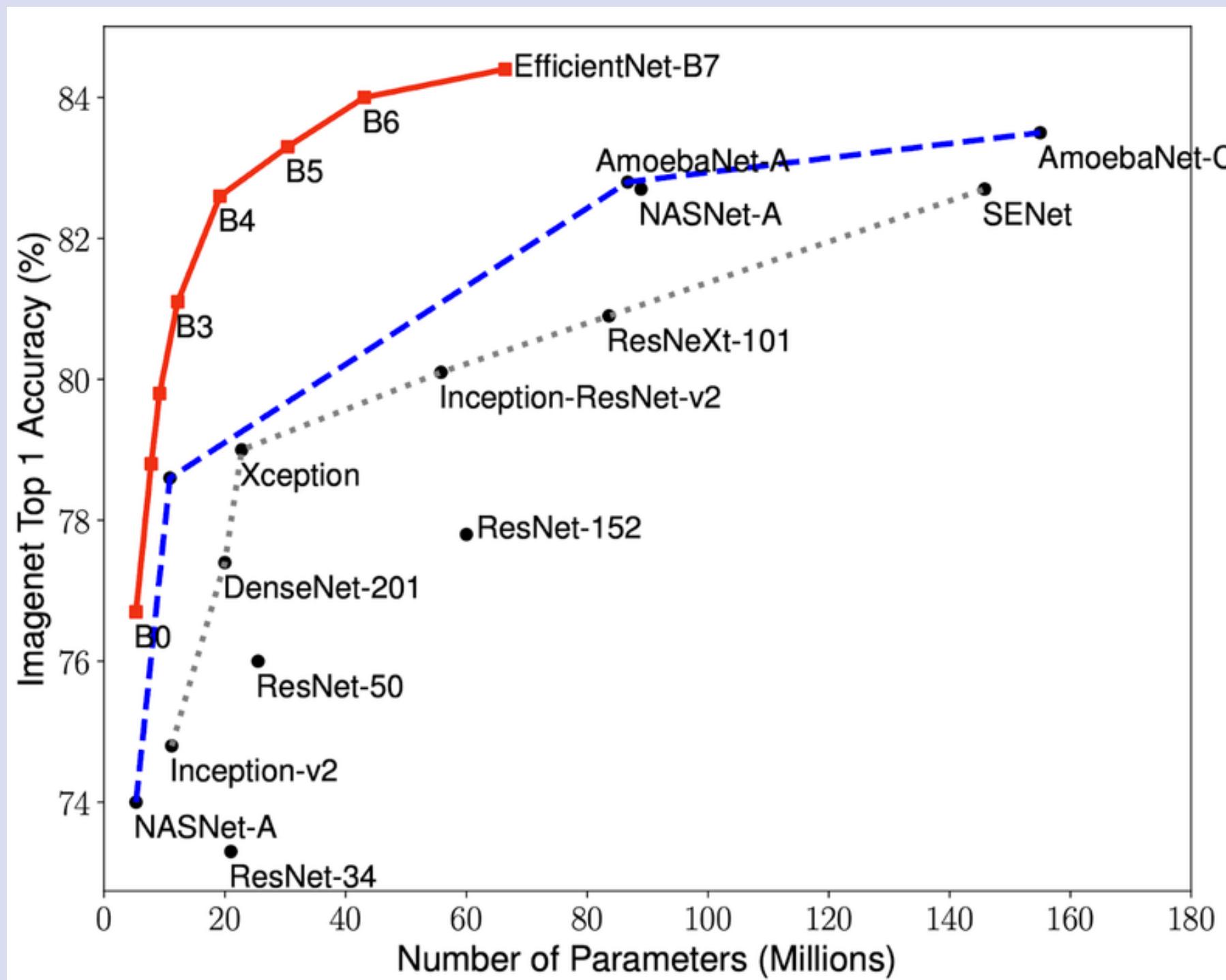
## Datablock

```
dblock = DataBlock(  
    blocks=(ImageBlock, CategoryBlock), #x - image ; y - single class  
    get_items=get_image_files, #get image from selected folder (path)  
; return list of pic  
    splitter=GrandparentSplitter(train_name =  
'Train', valid_name='Valid'), #use parent folder as train-valid split  
    get_y=parent_label, #use parent folder as label  
    item_tfms=Resize(512, method=ResizeMethod.Squish), # Resize image  
to same size using Squish  
    batch_tfms=aug_transforms(size=512, flip_vert=False,  
    pad_mode=PadMode.Reflection, max_lighting=0.2, p_lighting=0.75 )  
)
```

- จากที่ทดลองมาพบว่า ภาพขนาด  $512 \times 512$  จะได้ผลลัพธ์ที่ดีกว่าภาพที่ขนาดเล็กกว่านี้นั่นเพราะภาพเริ่มเบลอและมองไม่เห็นที่ตรวจ ATK ชัดเจน
- เพิ่มข้อมูลรูปภาพ (Data augmentation) โดยใช้ การปรับความเข้มแสง

# Modeling analysis...

Model มีมากมายแล้วเราจะเลือกใช้ตัวไหนดีล่ะ...



โดยเราเลือกใช้ Efficientnet\_b7 เนื่องจากที่  
ลองผิดลองถูกมา พบว่า Efficientnet\_b7  
ให้ค่า accuracy และ F1Score ได้ดีที่สุด

# Modeling\_analysis...

## Train model

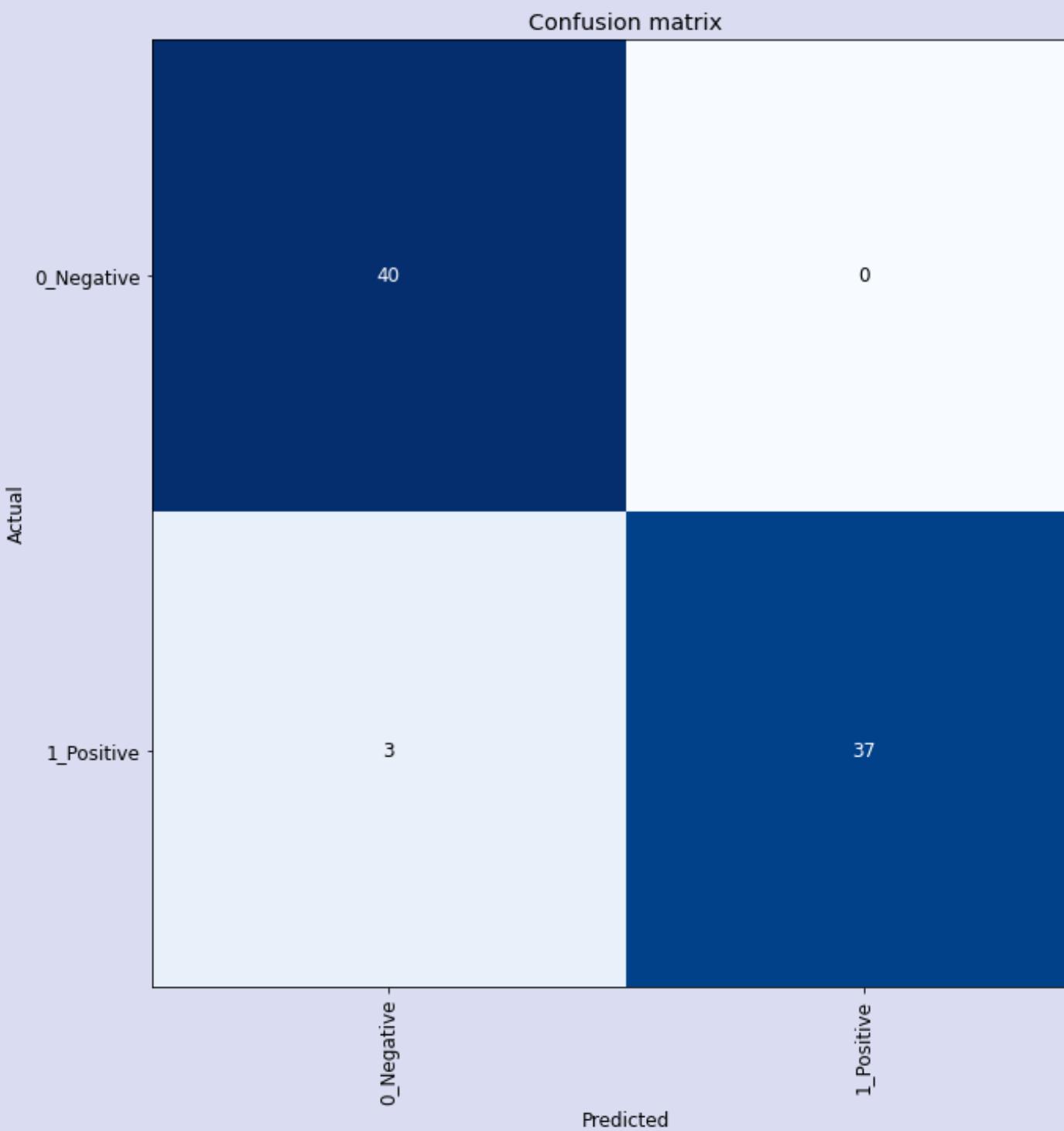
```
[ ] learn.fine_tune(epochs=20, freeze_epochs=1, base_lr=0.0001)
```

epoch	train_loss	valid_loss	accuracy	error_rate	precision_score	recall_score	f1_score	time
0	1.146837	0.706404	0.654088	0.345912	0.646341	0.670886	0.658385	05:43
30.00% [6/20 38:10<1:29:04]								
epoch	train_loss	valid_loss	accuracy	error_rate	precision_score	recall_score	f1_score	time
0	0.940056	0.523063	0.748428	0.251572	0.746835	0.746835	0.746835	06:21
1	1.007759	0.484331	0.786164	0.213836	0.792208	0.772152	0.782051	06:23
2	0.883855	0.382155	0.842767	0.157233	0.864865	0.810127	0.836601	06:21
3	0.782987	0.409482	0.805031	0.194969	0.807692	0.797468	0.802548	06:22

- เลือก model ด้วย Efficientnet b\_7 โดยใช้ transfer learning  
จำนวน 20 epoch

# Modeling\_analysis...

## ผลลัพธ์ที่ได้



```
interp.print_classification_report()
```

	precision	recall	f1-score	support
0_Negative	0.93	1.00	0.96	40
1_Positive	1.00	0.93	0.96	40
accuracy			0.96	80
macro avg	0.97	0.96	0.96	80
weighted avg	0.97	0.96	0.96	80

จากผลที่ได้มาเราจะเห็น model ทำผิดค่อนข้างน้อย (F1-score = 96 %) ผลที่เราได้มาค่อนข้างโอเคเลยจะ

# Validation analysis...

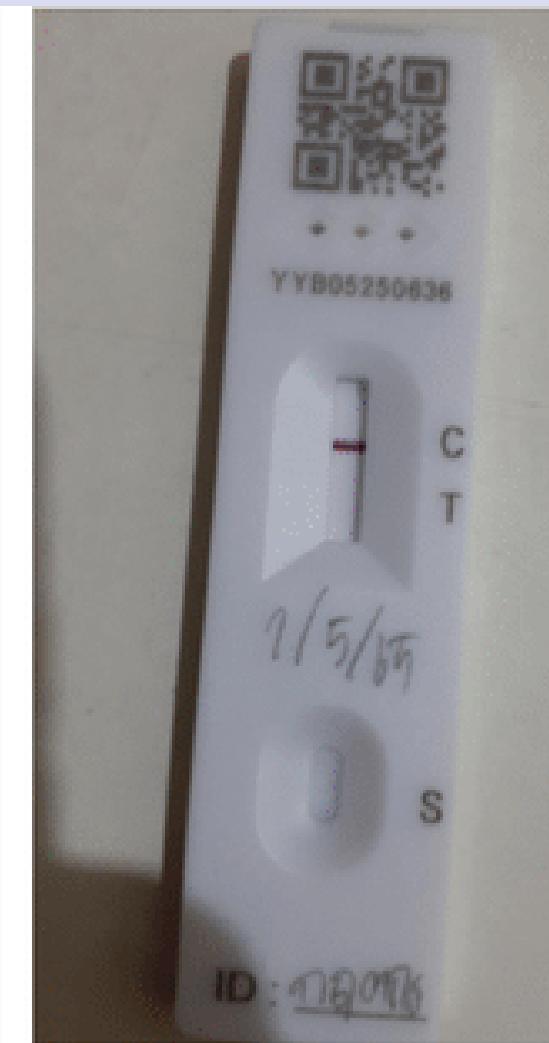
ทดลองใช้ model

```
[21] import time
     t1 = time.perf_counter()

     pred = learn_inf.predict(im_path)
     cls = int(pred[1]) # 0 = Negative, 1 = Positive
     print(pred, cls)

     t2 = time.perf_counter()
     print('time taken to run: {:.2f} sec'.format(t2-t1))

('0_Negative', TensorBase(0), TensorBase([0.9792, 0.0208])) 0
time taken to run: 3.56 sec
```



\*\*modelใช้เวลาในการตรวจจับ ATK ประมาณ (3-5 วินาที)  
สามารถตรวจจับได้อย่างแม่นยำ ถือว่าใช้ได้เลยทีเดียว...

# Validation analysis...

เช็คกับ test set อีกรอบเพื่อกันการ Overfitting

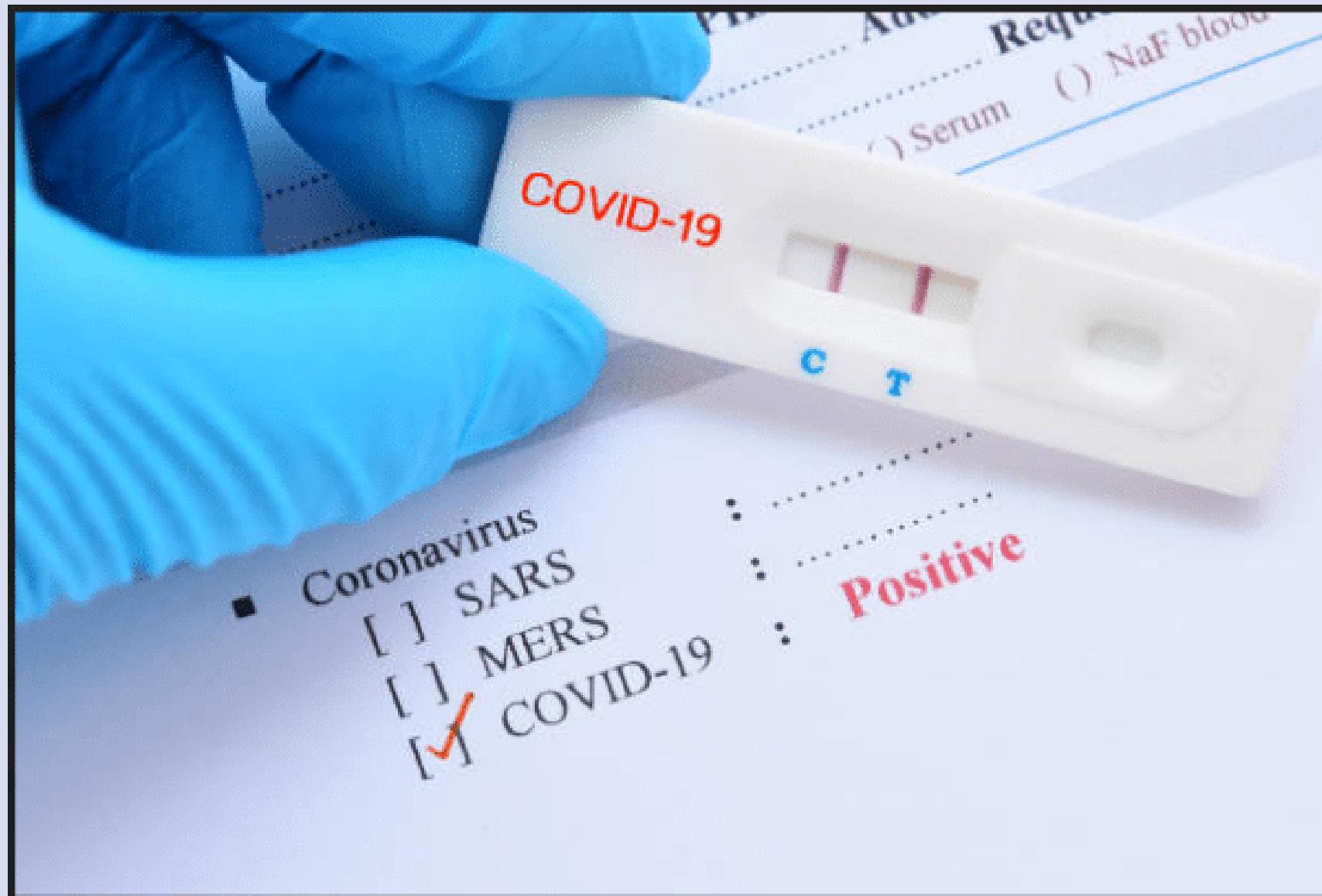
```
C Total image = 245  
Total loss = 6  
Pass = 239 ; Loss = 6  
accuracy = 97.55102040816327 % ; loss = 2.448979591836732 %
```

ผลที่ได้หลังจากการลองกับ test set

จาก test set ได้ความแม่นยำ ที่ 97.55 % ซึ่งมีประสิทธิภาพ และ ไม่ Overfitting

# Error analysis...

## ข้อผิดพลาดของ model



- จากภาพถ้าเรามองด้วยตาเปล่า จะรู้เลยว่าผลตรวจ ATK คือ Positive (ติดเชื้อ) แต่ปัญหาเกิดคือ.....

model ทำนายว่า เป็น Negative (ไม่ติดเชื้อ) !!! เพราะอันใด

# Error analysis...

ข้อผิดพลาดของ model

- แบ่งภาพออกเป็น 2 ส่วนแล้วให้โมเดลคำนายใหม่ พูดว่า

```
[6] im_path = '/content/atk_positive.PNG'
im = load_image(im_path).resize((224, 224))
im
```



```
[7] pred = learn_inf.predict(im_path)
cls = int(pred[1]) # 0 = Negative, 1 = Positive
pred, cls
```

```
(('1_Positive', TensorBase(1), TensorBase([0.0012, 0.9988])), 1)
```

```
[8] im_path = '/content/test.PNG'
im = load_image(im_path).resize((224, 224))
im
```



```
[9] pred = learn_inf.predict(im_path)
cls = int(pred[1]) # 0 = Negative, 1 = Positive
pred, cls
```

```
(('0_Negative', TensorBase(0), TensorBase([0.5335, 0.4665])), 0)
```

# Error analysis...

## ข้อผิดพลาดของ model

- ถ้า user ใส่ภาพอื่นที่ไม่ใช่ผลตรวจ ATK มาล่ะ

```
import time
t1 = time.perf_counter()

pred = learn_inf.predict(im_path)
cls = int(pred[1]) # 0 = Negative, 1 = Positive
print(pred, cls)

t2 = time.perf_counter()
print('time taken to run: {:.2f} sec'.format(t2-t1))

('0_Negative', TensorBase(0), TensorBase([0.8150, 0.1850])) 0
time taken to run: 2.49 sec
```



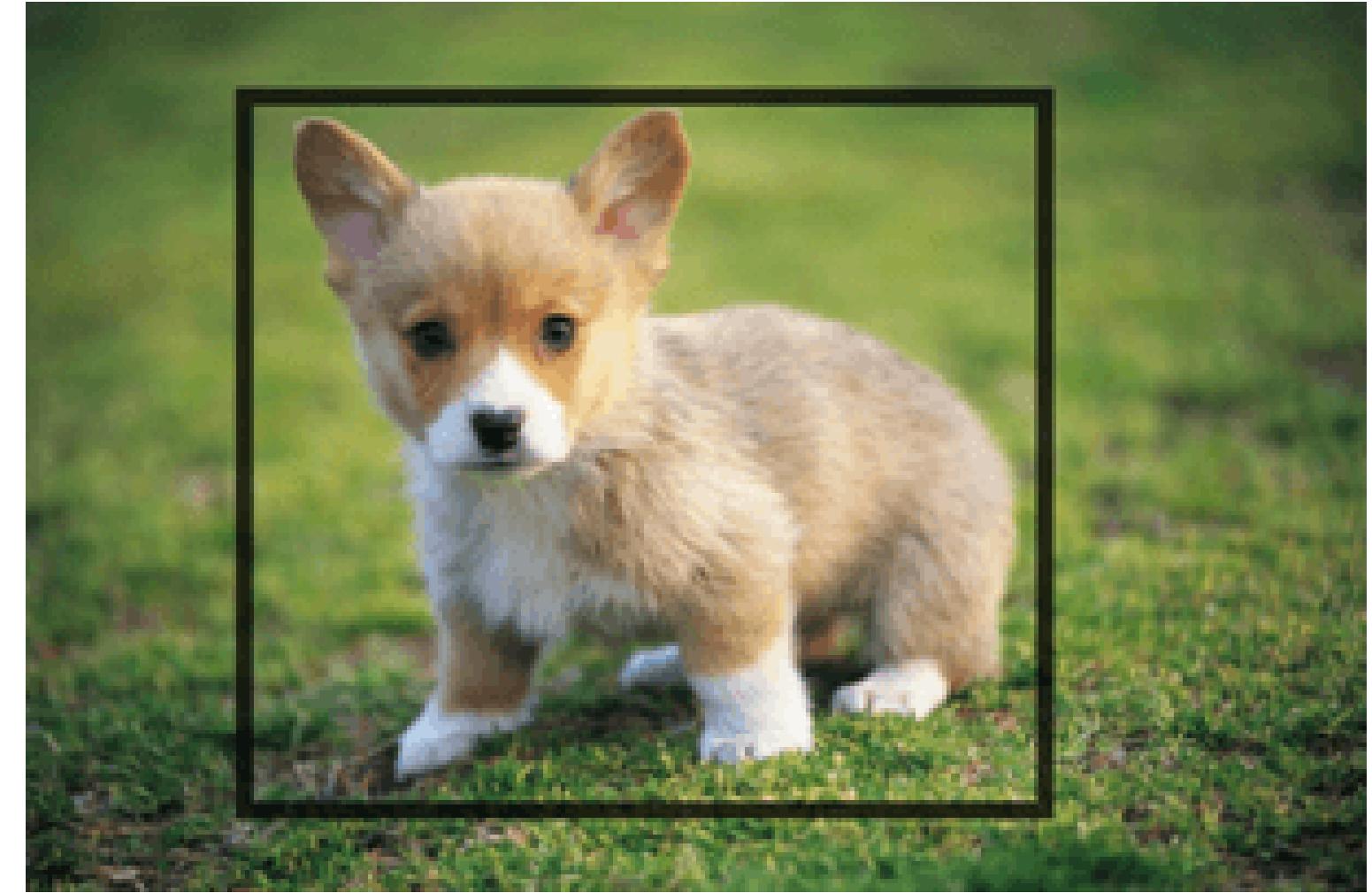
ภาพที่ 1: ผลที่ได้จากการ predict (Negative = 0.81 ; Positive = 0.18) ; ภาพที่ 2: ภาพใช้ในการทำนาย

# Error analysis...

แล้วแก้ปัญหานี้ยังไงดีล่ะ



Object Classification is the task of identifying that picture is a dog

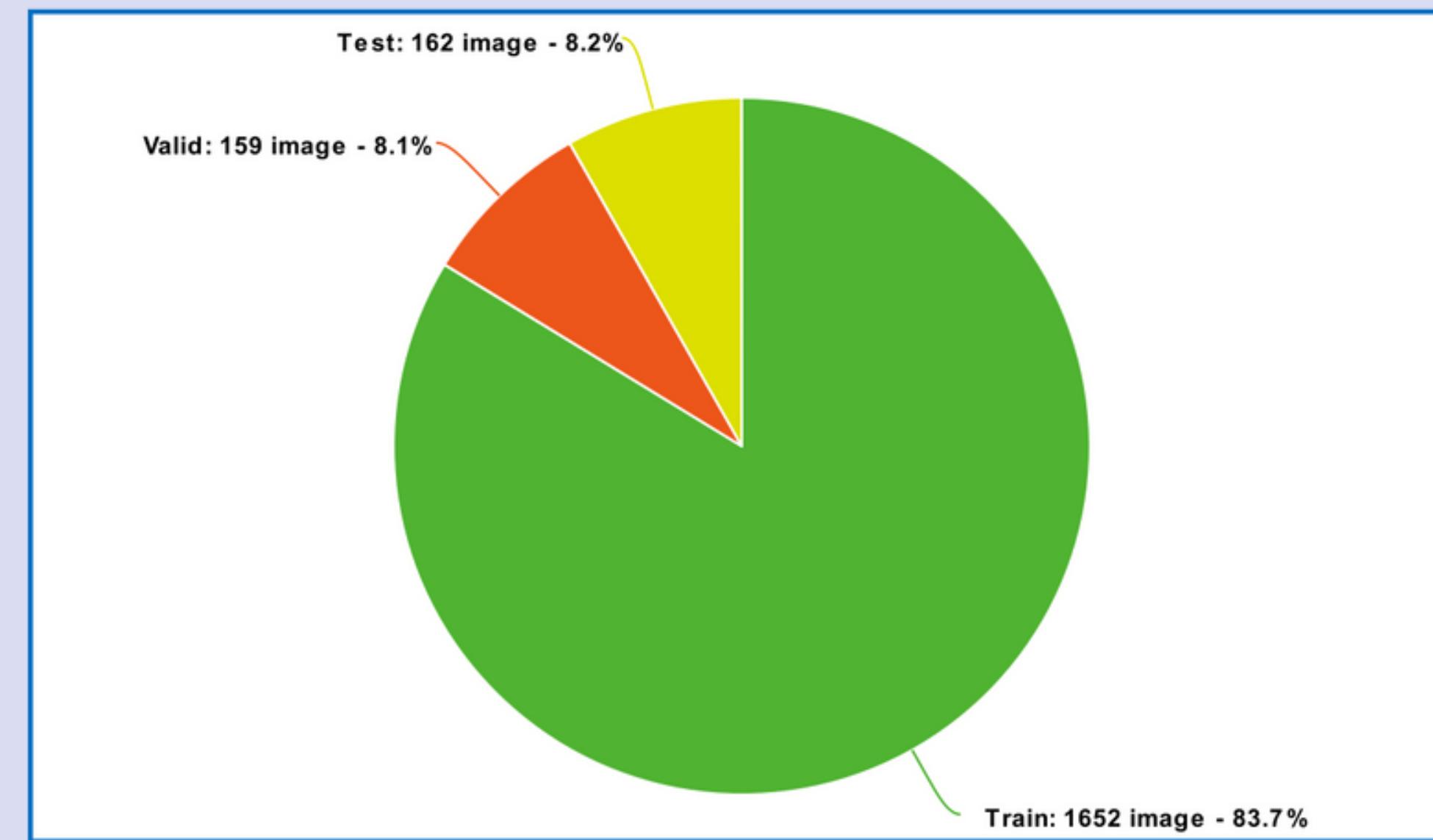


Object Localization involves the class label as well as a bounding box to show where the object is located.

**Modeling Validation and error  
analysis...  
(ATK : Object detection)**

# Modeling analysis...

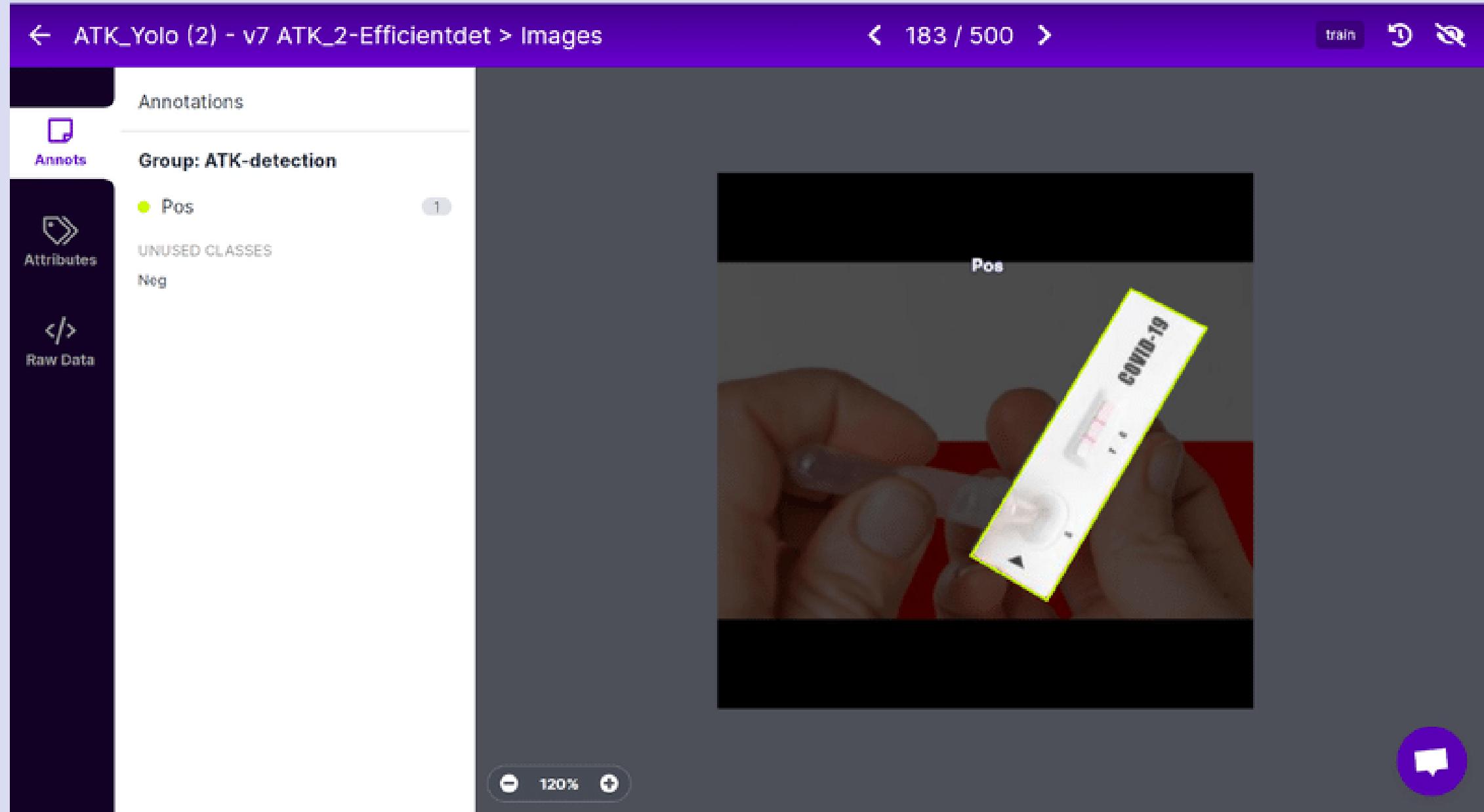
## Object detection with Icevision



- Train : Valid : Test split / 80 : 10 : 10 %

# Modeling analysis...

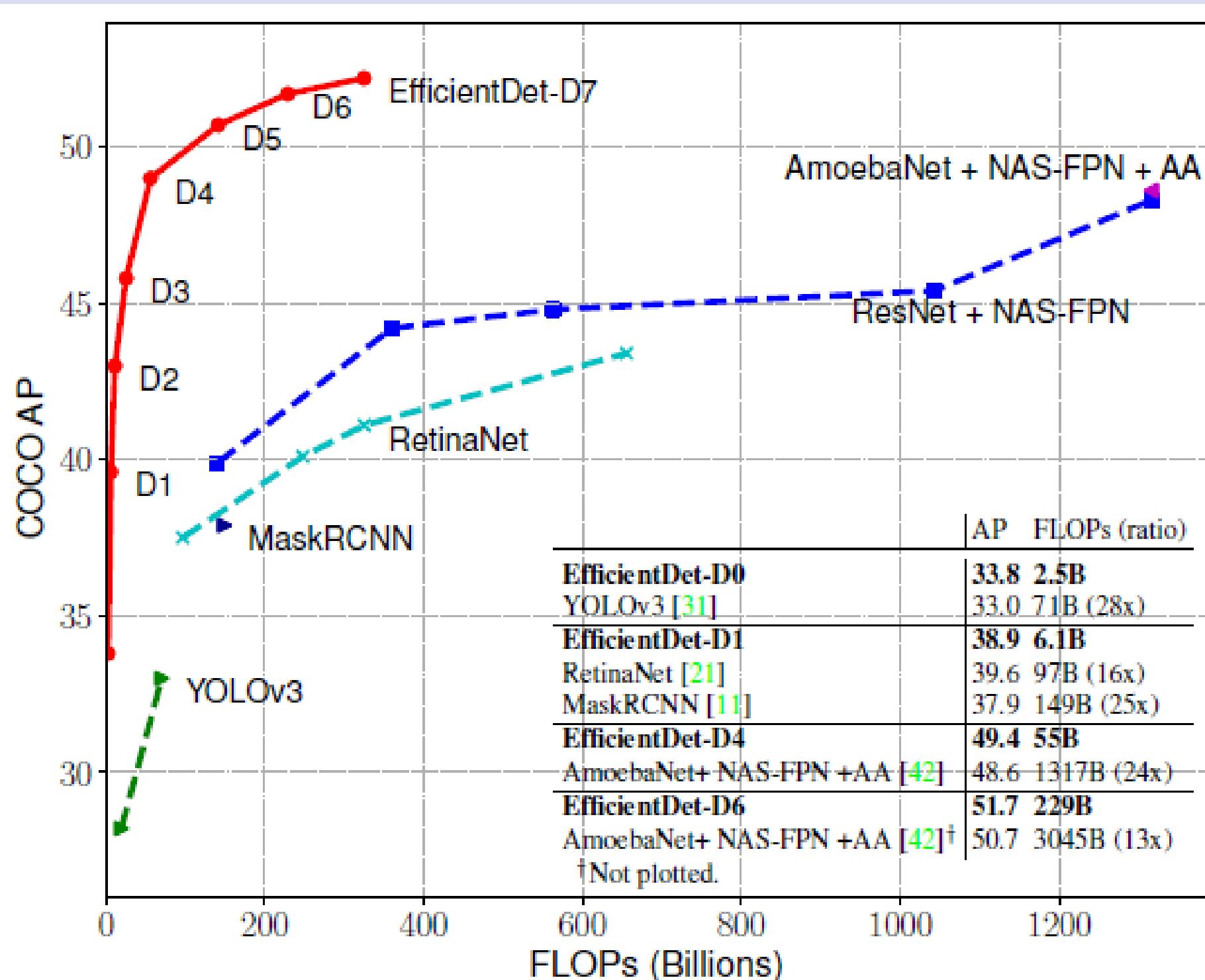
- ពេរឃម dataset សារុបា Object detection



Use roboflow for annotate image

# Modeling analysis...

Models มีหลากหลายเลือกใช้ตัวไหนดีล่ะ...



โดยเราเลือกใช้ Efficientdet\_d2 เนื่องจากที่  
ลองผิดลองถูกมา พบว่า Efficientdet\_d2  
ให้ค่า accuracy และ F1Score ได้ดีที่สุด  
(เท่าที่ CUDA เหลือ)

# Modeling analysis...

## Train model

```
learn.fine_tune(100, 10e-3, freeze_epochs=1)

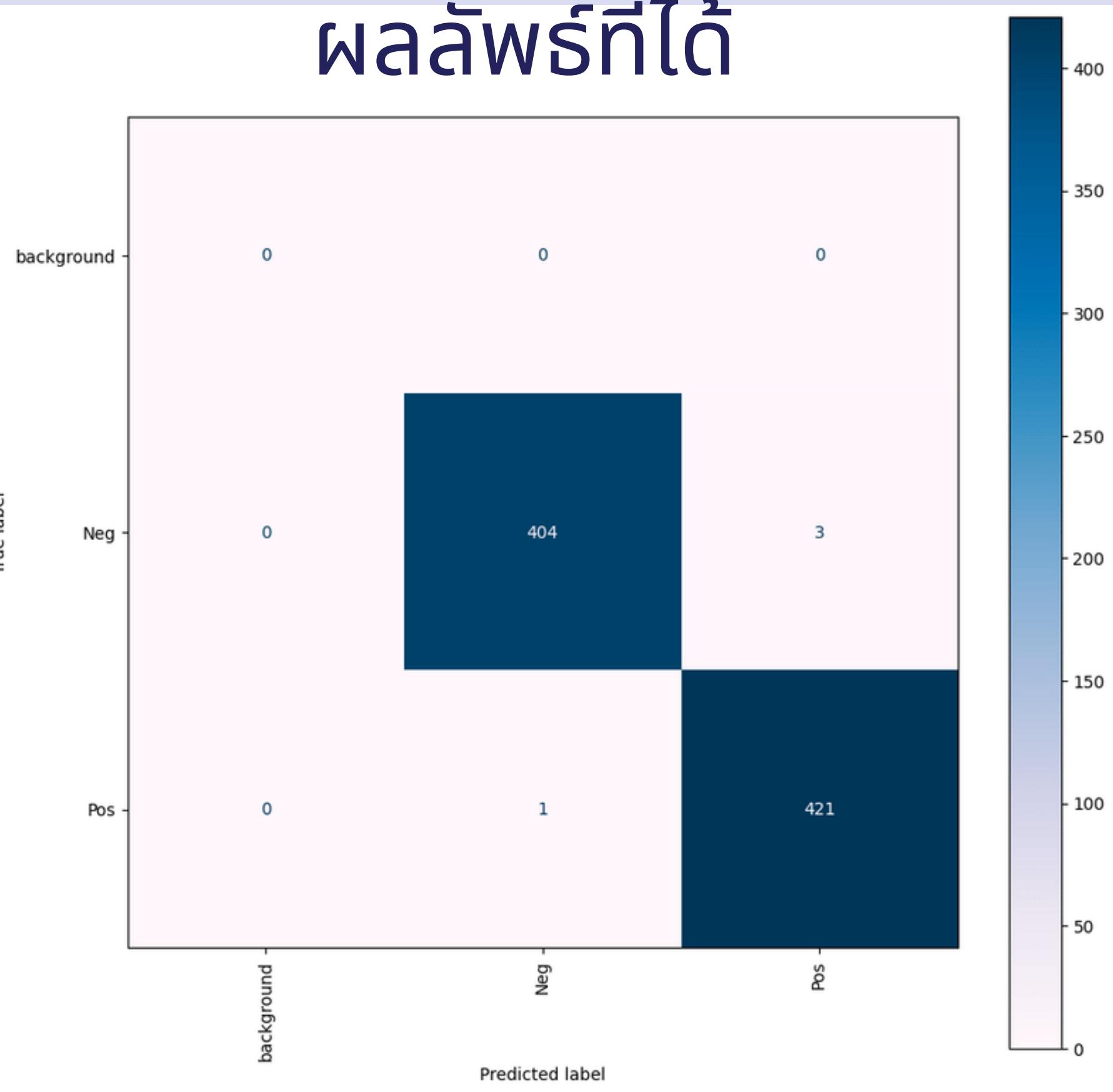
epoch train_loss valid_loss COCOMetric time
0    0.073071   0.070813   0.896608 01:55
/usr/local/lib/python3.7/dist-packages/effdet/bench.py:4
    indices_all = cls_topk_indices_all // num_classes
epoch train_loss valid_loss COCOMetric time
0    0.051661   0.057205   0.958570 02:21
1    0.049149   0.060535   0.957433 02:21
2    0.049422   0.063584   0.957773 02:20
```

93	0.036640	0.061577	0.974494	02:20
94	0.035953	0.062497	0.972912	02:20
95	0.035519	0.060736	0.975561	02:21
96	0.038861	0.061936	0.974340	02:20
97	0.040542	0.061649	0.974900	02:20
98	0.036378	0.061585	0.975220	02:20
99	0.038201	0.061993	0.974537	02:20

- เน้น model ด้วย Efficientdet d\_2 โดยใช้ transfer learning  
จำนวน 100 epoch (ได้ COCOMetric ที่ 97.45 %)

# Modeling analysis...

## ผลลัพธ์ที่ได้



- จาก Confusion matrix จะเห็นได้ว่า model ถ่ายผิดเพียงแค่ 4 ภาพ จากทั้งหมดกว่า 1000 ภาพ (99.51 %)



# Validation analysis...

# ໂຄສອນໃຫ້ງານ model

```
[21] img = PIL.Image.open('/content/51632986733_d9ac7937c9_a.jpg')

pred_dict = model_type.end2end_detect(img, valid_tfms, model, class_map=class_map, detection_threshold=0.7)

/usr/local/lib/python3.7/dist-packages/effdet/bench.py:45: UserWarning: __floordiv__ is deprecated, and its
indices_all = cls_topk_indices_all // num_classes

D import time
t1 = time.perf_counter()

try:
    labels, acc = pred_dict['detection']['labels'][0], pred_dict['detection']['scores'][0]
    print(labels, acc)
except IndexError:
    print("Not found Antigen test kit please take image again!")

t2 = time.perf_counter()
print("time taken to run: {:.2f} sec".format(t2-t1))

D Pos 0.9995173
time taken to run: 0.00 sec
```



ภาพที่ 1: ผลลัพธ์ที่ได้จากการทำ detection (Positive 99.5 %); ภาพที่ 2: ภาพที่ model ตรวจจับแบ่ง ATK ได้

ໂມເດລກໍານາຍຄ່ອນຂ້າງດີ ແລະ ແທບຈະຕອບຖັນທີ່ຮັງຈາກຜູ້ໃຊ້ງານກໍາການໃສ່ກາພ

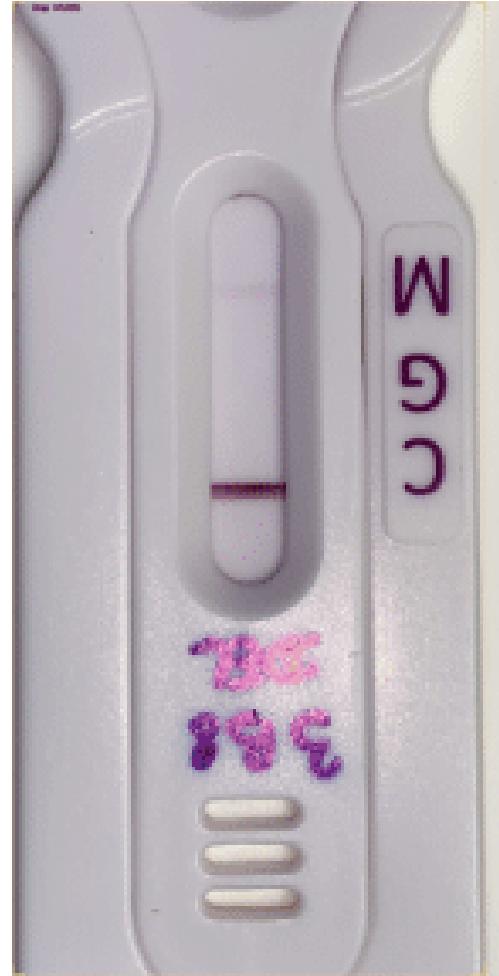
# Validation analysis...

เช็คกับ test set อีกรอบเพื่อ กันการ Overfitting

```
Total image = 233  
Total loss = 10 ( labels not found = 6, wrong = 4 )  
Pass = 223 ; Loss = 10  
accuracy = 95.70815450643777 % ; loss = 4.291845493562235 %
```

ได้ความแม่นยำ (Accuracy) ที่ 95.70 % ซึ่งถือว่าค่อนข้างมีประสิทธิภาพ และไม่ Overfitting นั่นเอง

# Error analysis...



```
[133]: import time
      t1 = time.perf_counter()

      try:
          labels, acc = pred_dict['detection']['labels'][0], pred_dict['detection']['scores'][0]
          print(labels, acc)
      except IndexError:
          print("Not found Antigen test kit please take image again!")
      except ValueError:# if image size to small resize it!
          print(f"Image size not support please resize it (divide by 128)")

      t2 = time.perf_counter()
      print('time taken to run: {:.2f} sec'.format(t2-t1))

      Neg 0.9435857
      time taken to run: 0.00 sec
```

ข้อผิดพลาดของ model ในกรณีที่บิดกีฬาตรวจมีความจำบากจนเกินไป (จีังจัง)

โอกาสเกิดขึ้นน้อยมาก แต่ก็เกิดขึ้นได้!



ข้อผิดพลาดของ model ในกรณีที่บีสิ่งของที่เป็นแท่งสีเหลืองรูปร่างคล้ายกับตัวตรวจ ATK

# Conclusion



# Any experiment before using FastAI and Icevision ?

-> Resnet 34, Resnet 50  
(accuracy = 50 %)

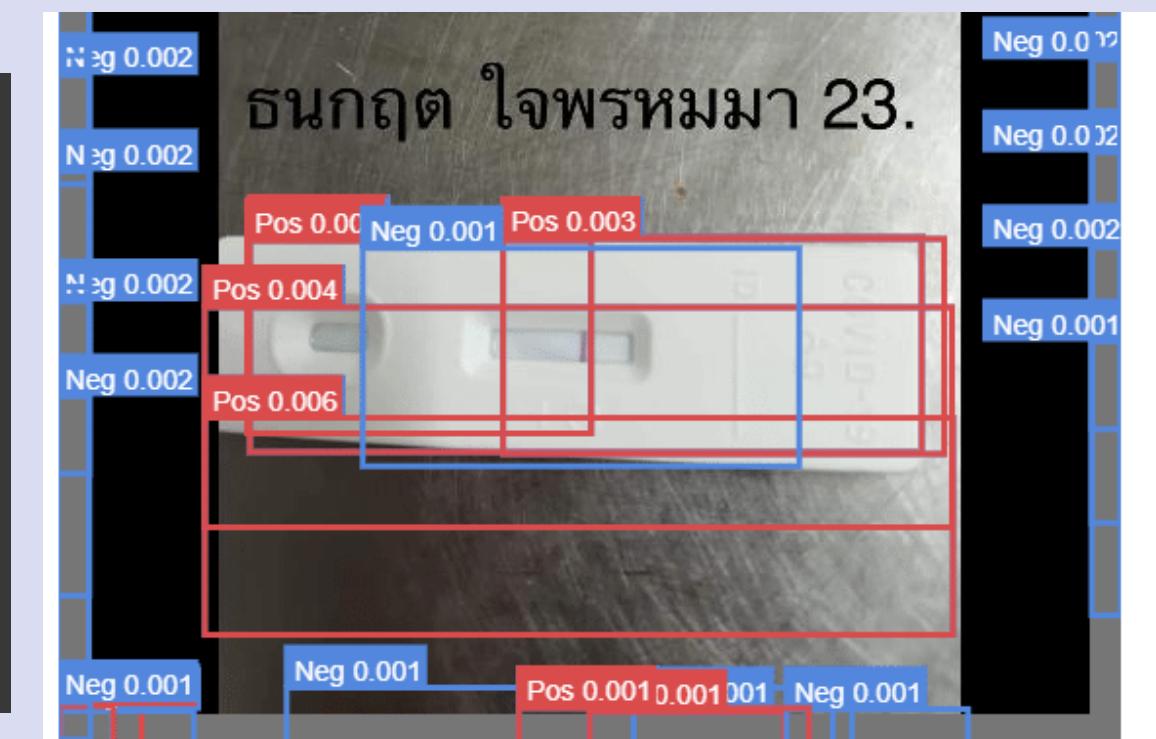
```
learn = vision_learner(dls, resnet50, metrics=accuracy)
learn.fine_tune(epochs=20, freeze_epochs=1, base_lr=2e-3)

Downloading: "https://download.pytorch.org/models/resnet50-  
100% 97.8M/97.8M [00:00<00:00, 97.8M/s]  
epoch train_loss valid_loss accuracy time  
0 0.932647 0.793536 0.666667 02:46  
epoch train_loss valid_loss accuracy time  
0 0.473135 1.384461 0.604520 00:42  
1 0.376337 1.107419 0.615819 00:42  
2 0.302250 1.233447 0.649718 00:42  
3 0.251572 2.718487 0.553672 00:42  
4 0.215731 1.645837 0.598870 00:42  
5 0.191812 2.392663 0.536723 00:42  
6 0.177738 3.135797 0.502825 00:42
```

-> YOLO V5 (S, M, L)  
(accuracy = 50 %)

469	0.008763	0.008577	0.480423	01:19
470	0.008682	0.008424	0.480892	01:20
471	0.008457	0.008801	0.482069	01:19
472	0.008548	0.008513	0.482153	01:19
473	0.008690	0.008467	0.482997	01:19
474	0.008630	0.008789	0.478305	01:19
475	0.008392	0.008500	0.481506	01:19
476	0.008525	0.008435	0.484974	01:19
477	0.008658	0.008608	0.482453	01:20

47.50% [19/40 00:19<00:21 0.0087]



# Image classification vs Object detection

## Compare on test set

```
↳ Total image = 245  
Total loss = 6  
Pass = 239 ; Loss = 6  
accuracy = 97.55102040816327 % ; loss = 2.448979591836732 %
```

```
Total image = 233  
Total loss = 10 ( labels not found = 6, wrong = 4 )  
Pass = 223 ; Loss = 10  
accuracy = 95.70815450643777 % ; loss = 4.291845493562235 %
```

### image classification (FastAI)

### Object detection (Icevision)

\*\*ถึงค่าความแม่นยำของ object detection จะได้น้อยกว่าการทำ image classification แต่เรา ก็สามารถมั่นใจได้ว่า ถ้ามีรูปภาพที่ไม่ใช่ที่ตรวจ ATK มา model ก็จะไม่ทำการเดาเมื่วนอน

เราจึงเลือกใช้ model ของ object detection

# (OCR + Object detection) vs baseline

	Based on	ATK accuracy %	Idnum accuracy %	name-lastname acc %	Accuracy %	Time per each (sec)
0	Baseline (human)	94.0 %	84.0 %	85.0 %	88.0 %	26.87 sec
1	AOC Model	100.0 %	100.0 %	82.76 %	94.25 %	4.84 sec
2	Comparison with baseline	+ 6.0 %	+ 16.0 %	- 2.24 %	+ 6.25 %	faster than 22.03 sec (81.99 %)

ตารางเปรียบเทียบความแม่นยำเฉลี่ยที่ model ทำได้เปรียบเทียบกับ baseline (คบ)



\*\* การวัดผลใช้ภาพที่มีความชัด และ ถ่ายถูกต้อง เมื่อันใน แบบสอบถาม (Google form) ในการวัดผล model

# Deployment



# EC2 (AWS)



Scan me OwO

Not Secure | 18.141.137.167

AI BUILDERS  
a program for kids who want to build good AI

X

Antigen test kit + Identification Card detector.

Select option mode:

ATK + Idcard Detect

ATK Detect

Idcard Detect

upload ATK + Idcard img here.. OwO

Drag and drop file here  
Limit 200MB per file • PNG, JPG, JPEG

Browse files

More image for test..

[Github img test set.](#)

Recomend / Issues report..

[Google form](#)

Waiting for image..

ด้วยผลตรวจ ATK คุ้มครองประชาชน

Thai National ID Card  
Identification Number: X XXXX XXXXX XX X  
Name: X XXXX XXXX

Test Date: ๒๕๖๔/๐๑/๒๒

<http://18.141.137.167/>

# Why not deploy on Streamlit Cloud ?

Buggggggggg !!!!

The screenshot shows a Streamlit app error page on the left and a terminal log on the right, both displaying the same error message: "AttributeError: partially initialized module 'cv2' has no attribute 'gapi\_wip\_gst\_GStreamerPipeline'".

**Error Message (Left):**

```
AttributeError: This app has encountered an error. The original error message is redacted to prevent data leaks. Full error details have been recorded in the logs (if you're on Streamlit Cloud, click on 'Manage app' in the lower right of your app).
```

**Traceback:**

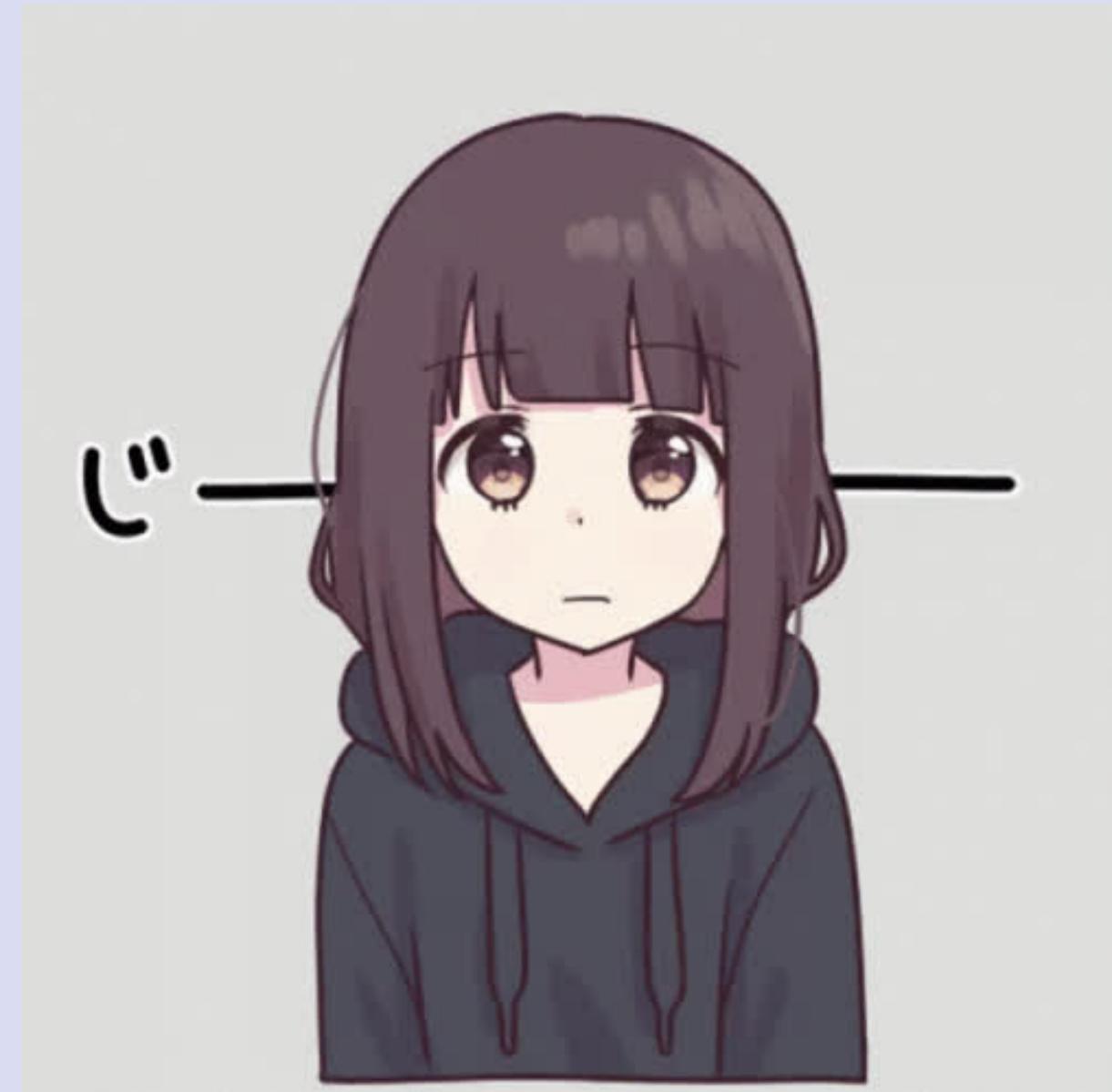
```
File "/home/appuser/venv/lib/python3.9/site-packages/streamlit/scriptrunner/_script_runner.py", line 10, in <module>
    exec(code, module.__dict__)
File "full.py", line 9, in <module>
    import easyocr as ocr #OCR
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/__init__.py", line 1, in <module>
    from .easyocr import Reader
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/easyocr.py", line 1, in <module>
    from .detection import get_detector, get_textbox
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/detection.py", line 7, in <module>
    import cv2
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 181, in <module>
    bootstrap()
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 175, in bootstrap
    if __load_extra_py_code_for_module("cv2", submodule, DEBUG):
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 28, in __load_extra_py_code_
    py_module = importlib.import_module(module_name)
File "/usr/local/lib/python3.9/importlib/__init__.py", line 127, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/gapi/__init__.py", line 290, in <module>
    cv.gapi.wip.GStreamerPipeline = cv.gapi_wip.gst_GStreamerPipeline
AttributeError: partially initialized module 'cv2' has no attribute 'gapi_wip_gst_GStreamerPipeline' (most recent call last):
  File "/home/appuser/venv/lib/python3.9/site-packages/streamlit/scriptrunner/_script_runner.py", line 564, in <module>
    exec(code, module.__dict__)
File "full.py", line 9, in <module>
    import easyocr as ocr #OCR
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/__init__.py", line 1, in <module>
    from .easyocr import Reader
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/easyocr.py", line 3, in <module>
    from .detection import get_detector, get_textbox
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/detection.py", line 7, in <module>
    import cv2
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 181, in <module>
    bootstrap()
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 175, in bootstrap
    if __load_extra_py_code_for_module("cv2", submodule, DEBUG):
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 28, in __load_extra_py_code_
    py_module = importlib.import_module(module_name)
File "/usr/local/lib/python3.9/importlib/__init__.py", line 127, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/gapi/__init__.py", line 290, in <module>
    cv.gapi.wip.GStreamerPipeline = cv.gapi_wip.gst_GStreamerPipeline
AttributeError: partially initialized module 'cv2' has no attribute 'gapi_wip_gst_GStreamerPipeline' (most recent call last):
```

**Terminal Log (Right):**

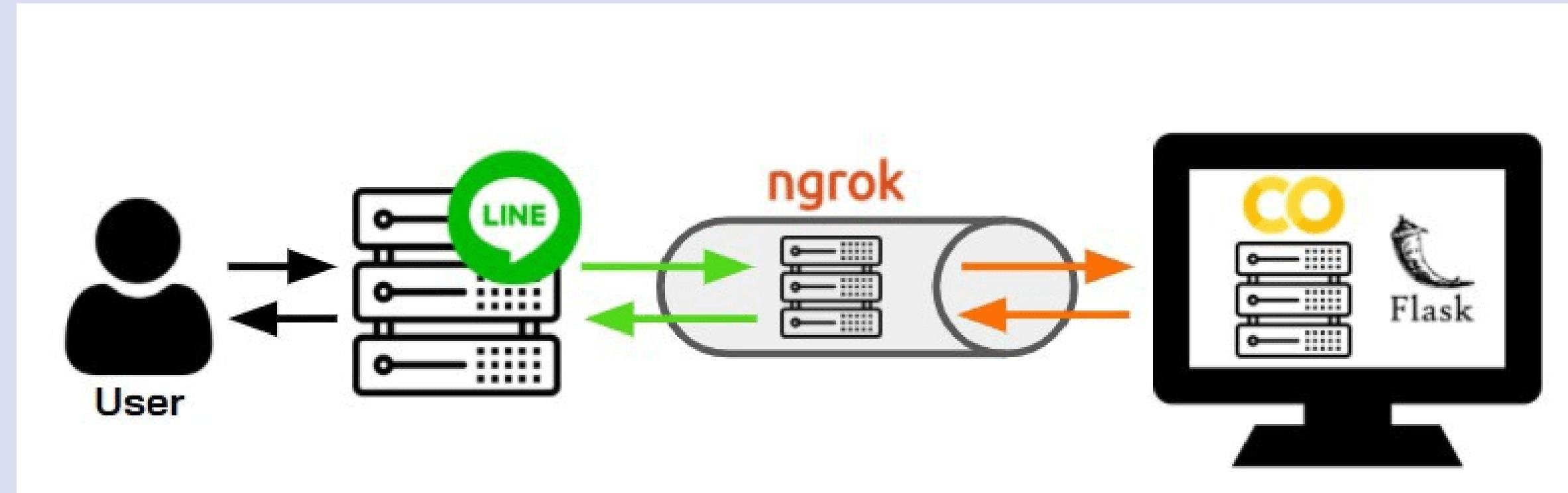
```
from .easyocr import Reader
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/easyocr.py", line 3, in <module>
    from .detection import get_detector, get_textbox
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/detection.py", line 7, in <module>
    import cv2
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 181, in <module>
    bootstrap()
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 175, in bootstrap
    if __load_extra_py_code_for_module("cv2", submodule, DEBUG):
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 28, in __load_extra_py_code_
    py_module = importlib.import_module(module_name)
File "/usr/local/lib/python3.9/importlib/__init__.py", line 127, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/gapi/__init__.py", line 290, in <module>
    cv.gapi.wip.GStreamerPipeline = cv.gapi_wip.gst_GStreamerPipeline
AttributeError: partially initialized module 'cv2' has no attribute 'gapi_wip_gst_GStreamerPipeline' (most recent call last):
  File "/home/appuser/venv/lib/python3.9/site-packages/streamlit/scriptrunner/_script_runner.py", line 564, in <module>
    exec(code, module.__dict__)
File "full.py", line 9, in <module>
    import easyocr as ocr #OCR
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/__init__.py", line 1, in <module>
    from .easyocr import Reader
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/easyocr.py", line 3, in <module>
    from .detection import get_detector, get_textbox
File "/home/appuser/venv/lib/python3.9/site-packages/easyocr/detection.py", line 7, in <module>
    import cv2
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 181, in <module>
    bootstrap()
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 175, in bootstrap
    if __load_extra_py_code_for_module("cv2", submodule, DEBUG):
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/__init__.py", line 28, in __load_extra_py_code_
    py_module = importlib.import_module(module_name)
File "/usr/local/lib/python3.9/importlib/__init__.py", line 127, in import_module
    return _bootstrap._gcd_import(name[level:], package, level)
File "/home/appuser/venv/lib/python3.9/site-packages/cv2/gapi/__init__.py", line 290, in <module>
    cv.gapi.wip.GStreamerPipeline = cv.gapi_wip.gst_GStreamerPipeline
AttributeError: partially initialized module 'cv2' has no attribute 'gapi_wip_gst_GStreamerPipeline' (most recent call last):
```

(ต้องนี่กำลัง debug เรื่อยๆครับ ถ้าได้จะมีการ update นะครับ OwO)

# Future plan



# Line api



for



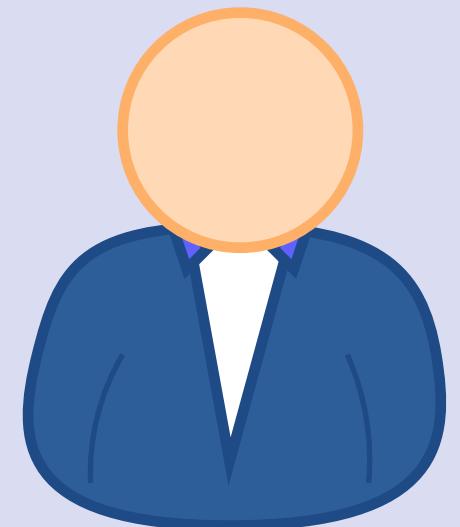
Institute



Organisation



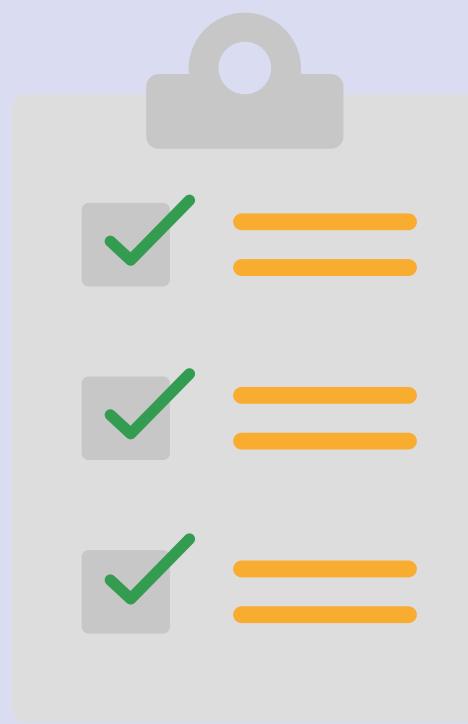
Family



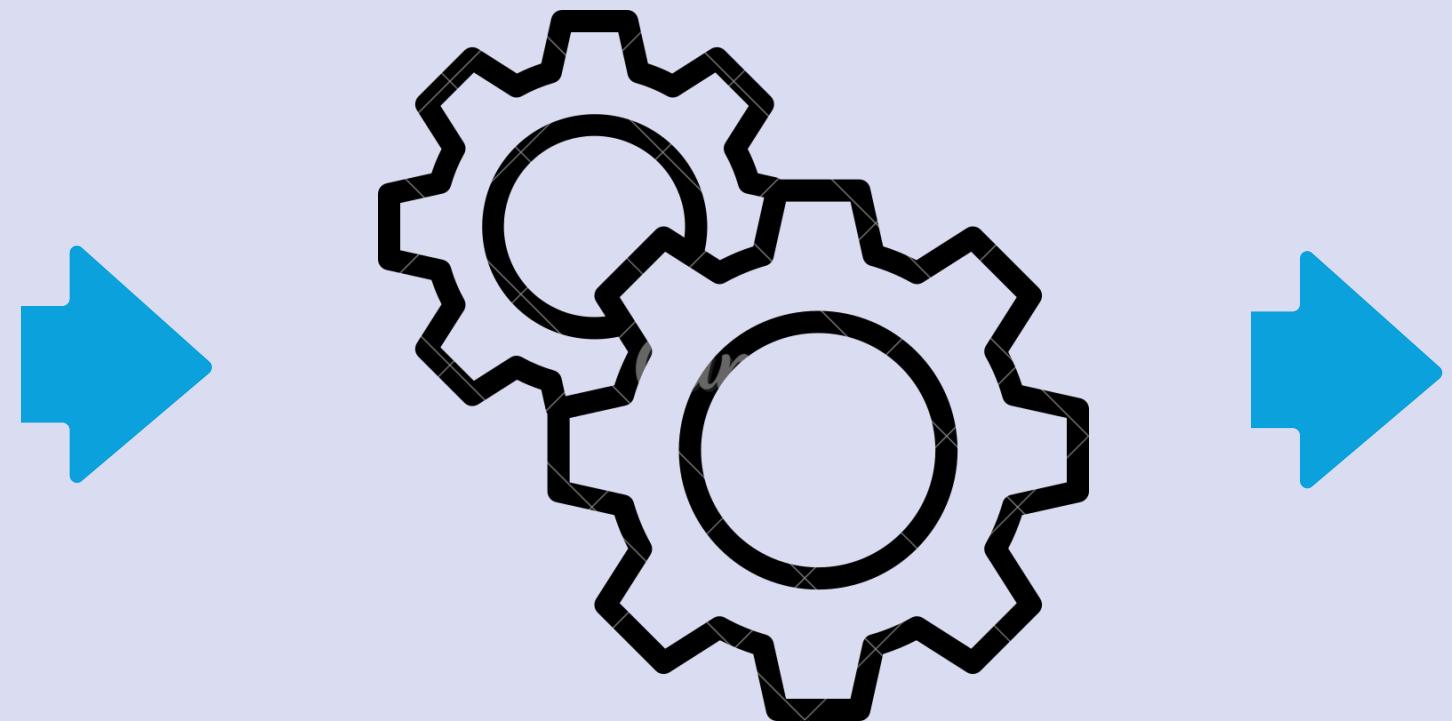
Person

# Debug name-lastname OCR

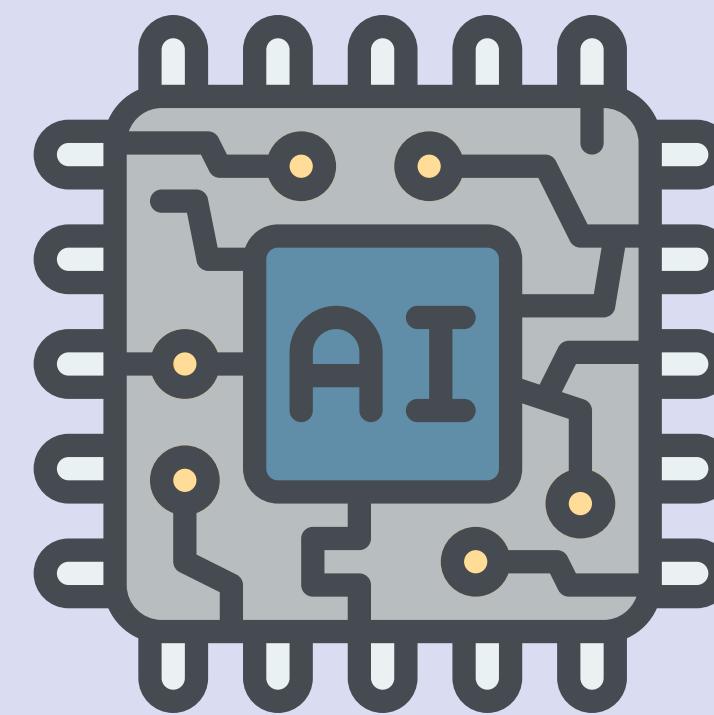
## Create new prediction model



Dataset  
(คนที่ซื้อเมื่อวันที่  $i, l$ )



Train model  
( $i, l$  prediction)

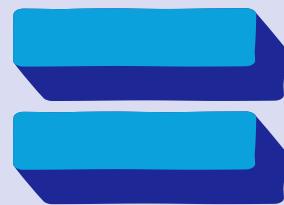
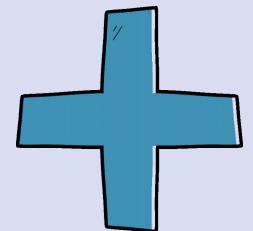
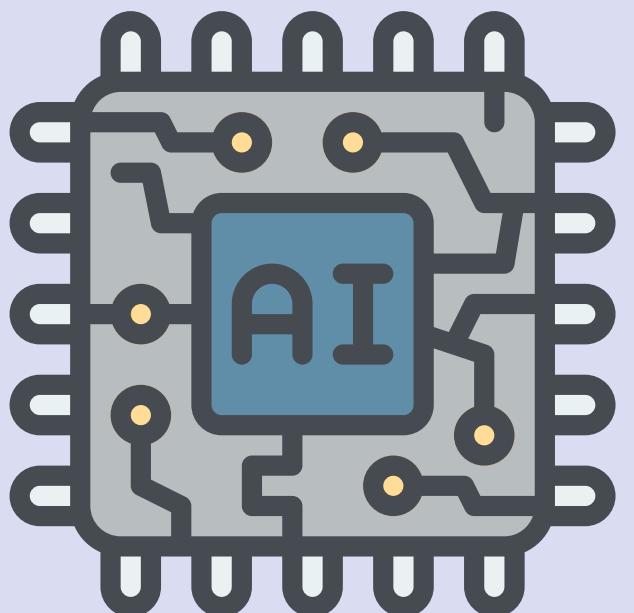


( $i, l$ ) name prediction  
model

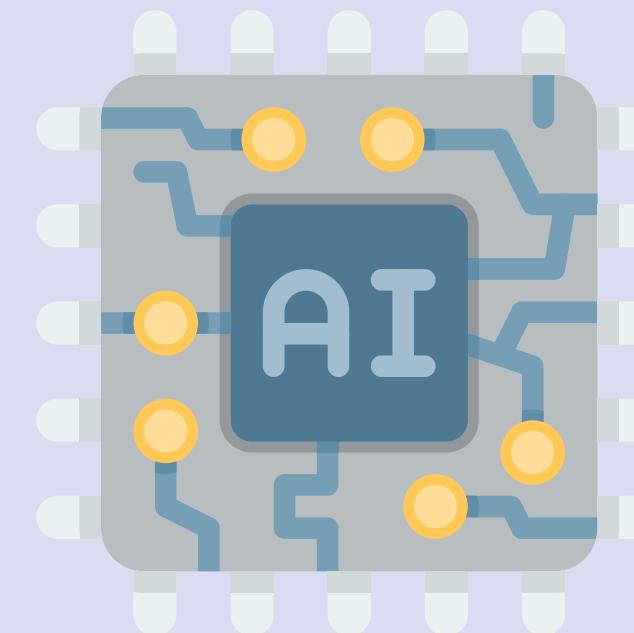
# Debug name-lastname OCR

Merge model

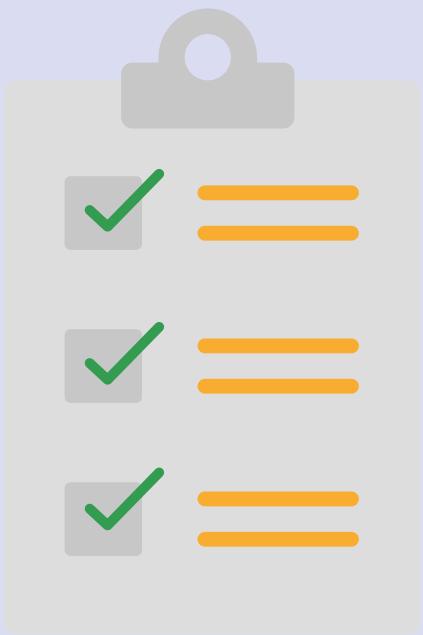
(i, l) name prediction  
model



Idcard + ATK

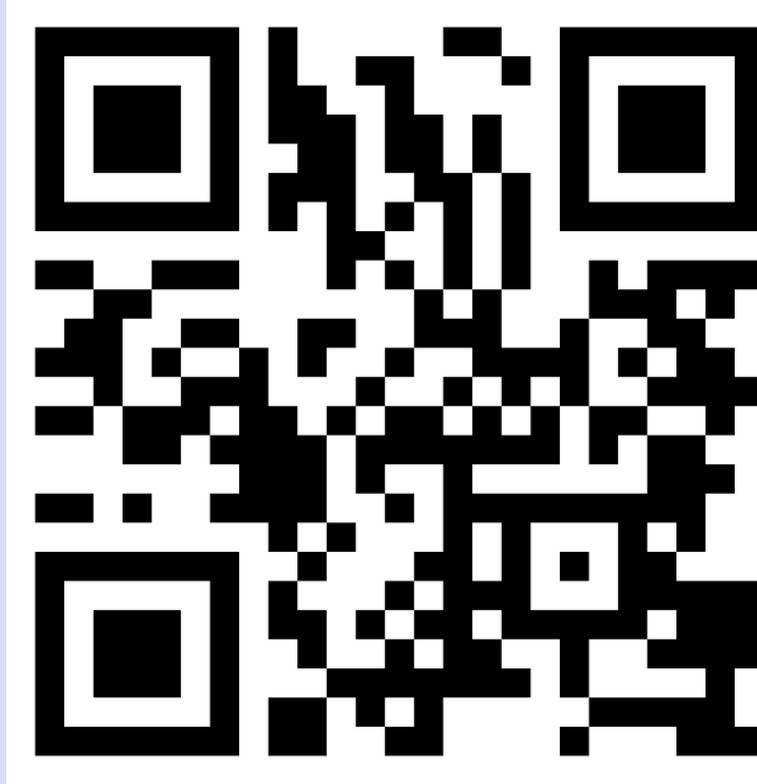
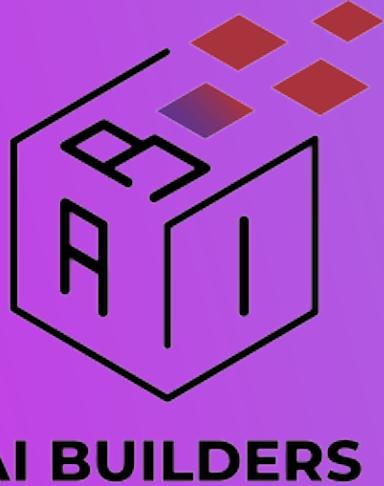


AOC model



ATK result  
Idnum  
name, lastname  
(more accurate)

# Thank you for your attention.. OwO



Web deploy (EC2)



Blog (Medium)



Github (Code)

Contact : mjsalyjoh@gmail.com



# Q & A Session



# Why use Rule-based system instead of Token Classification ?

- เคยลองใช้ NLTK, Bert, Huggingface ฯลฯ ชื่อ-นามสกุล ไทยที่กับศัพท์ token classification model ไม่สามารถแยกได้เลย ใช้ Rule-based system แทน

⚡ Hosted inference API ⓘ

TokenName Classification

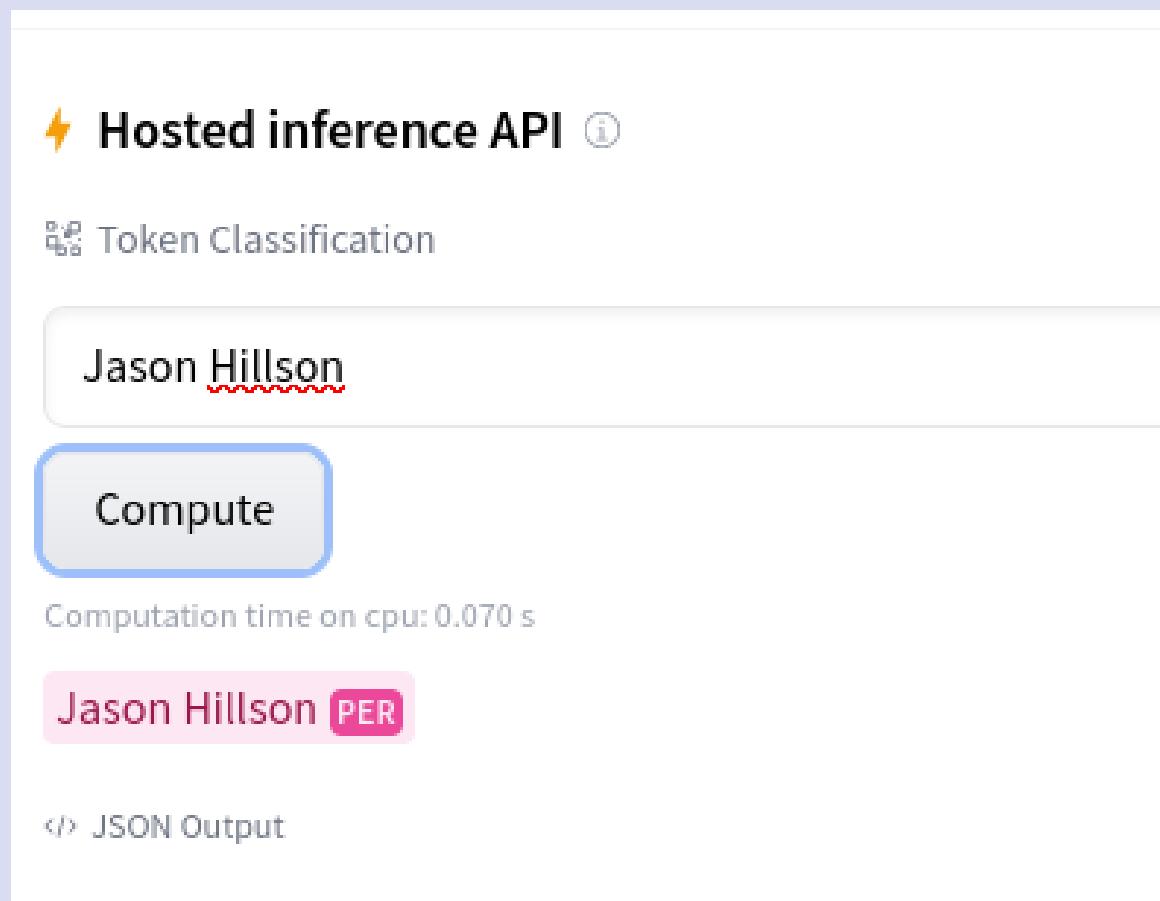
Jason Hillson

Compute

Computation time on cpu: 0.070 s

Jason Hillson PER

JSON Output



⚡ Hosted inference API ⓘ

TokenName Classification

Tanaanan Chalearnpan

Compute

Computation time on cpu: 13.500 s

Tanaanan Chalearnpan ORG

JSON Output

