Bank A Rai

Banknote Recognition for Visually Impaired People

Senior Project II
Semester 2/2017

Advisor
Dr. Kwankamol Nongpong

Project Committees
A. Tapanan Yeophantong
A. Chayapol Moemeng

Members

Wan

Chatchawan Yoojuie

Benz

Natthakul Boonmee

Top

Pathompong Kunapermsiri

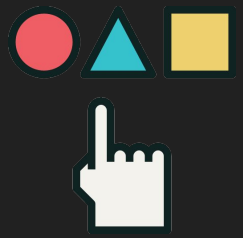
Problem

- Visually impaired take longer time than normal people to identify the banknote.
- Difficulties in spending money on a daily basis.
- The possibility of identify the wrong banknotes.

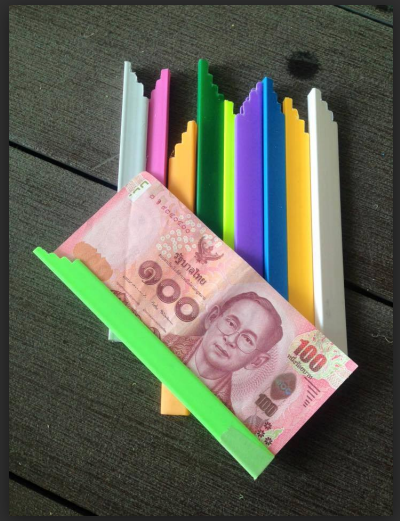


Existing Alternatives

How blind people identify the banknotes?



- Modified sliding bar report cover to distinguish money.
- Braille dots on the banknote

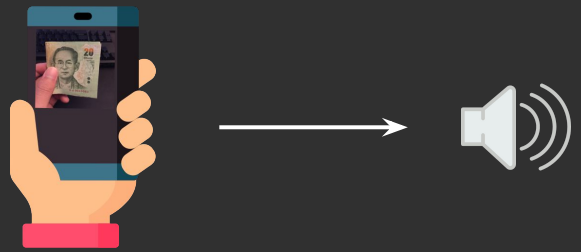


Solution

Mobile Application on Android

Basic flow of application:

1. Open an application, it will launch the back camera of device.
2. User holds the banknote in their hand or place on table.
3. Point the camera to the banknote.
4. If a banknote is detected, it will read out louds the denomination of banknote to user.



Objective

- Visually impaired people can help themselves.
- Ability to identify banknotes in real-time and accurate.

Scope

- Classify only Thai banknotes Type 16.
- Detect a single banknote per frame.
- Cannot detect counterfeit.

Type of Thai banknotes

Thai Banknote Type 16

Survey & Interview

Survey & Interview

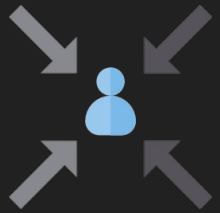
Interview 3 manager of visually impaired club. 3 of them said, most of blind people use smartphone. There are 2 main group.

1. People who just had visually impaired problem and hardly try to recognise the banknote.
2. People who have visually impaired since they were born, can easily recognise the banknote by touching.

Interview 1 MSCS student who has visually impaired problem.

- He can recognise banknote easily, but he also prefer to use application.

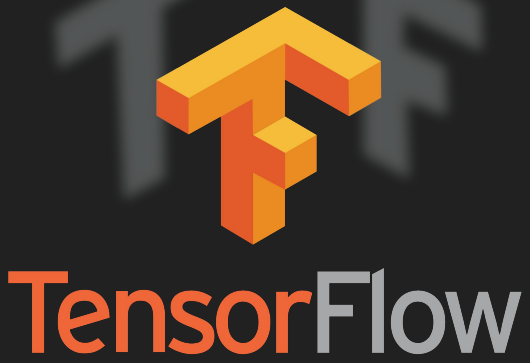
Target User



- People who just have visually impaired problem.
- Visually impaired people who usually have nurse or someone to take care of them.
- Visually impaired people whose their job doesn’t involve of touching money often.

Tools

Android SDK



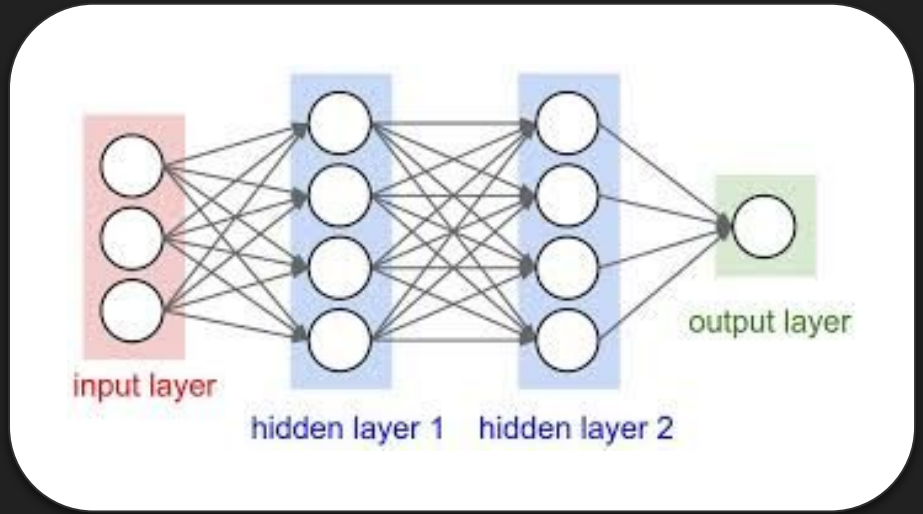
TensorFlow

Keras

Literature Review

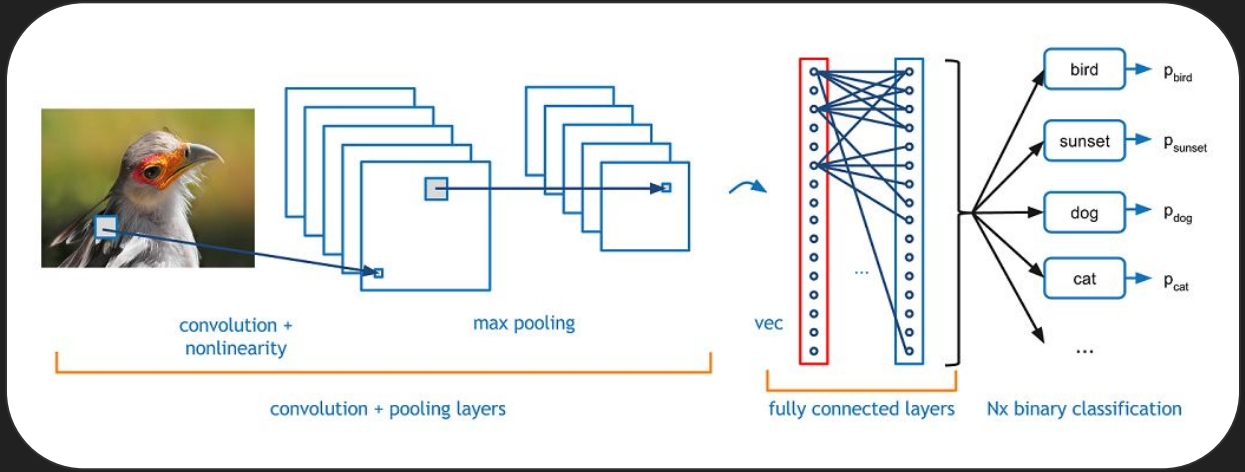
Artificial Neural Network

- Network of connected layer of nodes
- Input layer is image
- Output layer is prediction
- Hidden layer will extract the feature from image



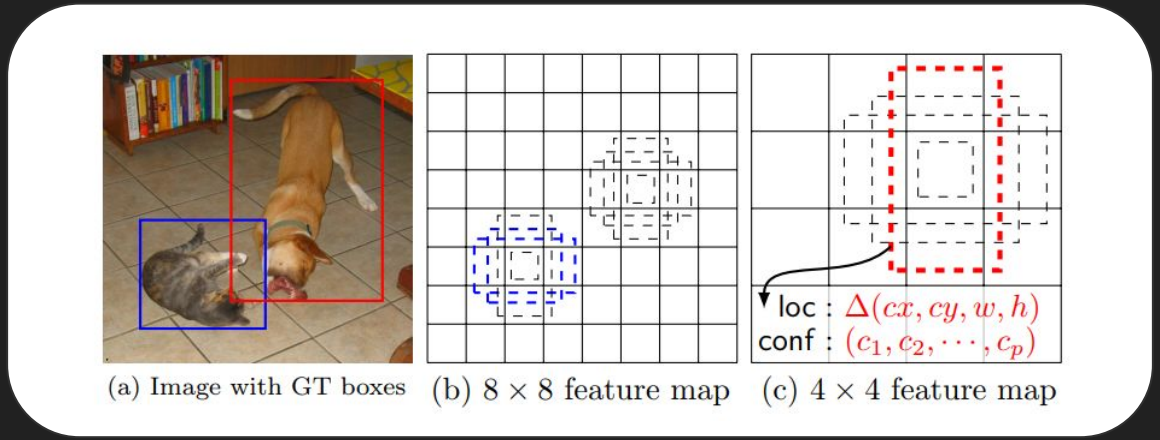
Convolutional Neural Network

- Type of ANN that explicitly design for process images
- Pooling and Convolution layer will extract feature and Fully-connected will use these feature to predict class



Single Shot MultiBox Detector

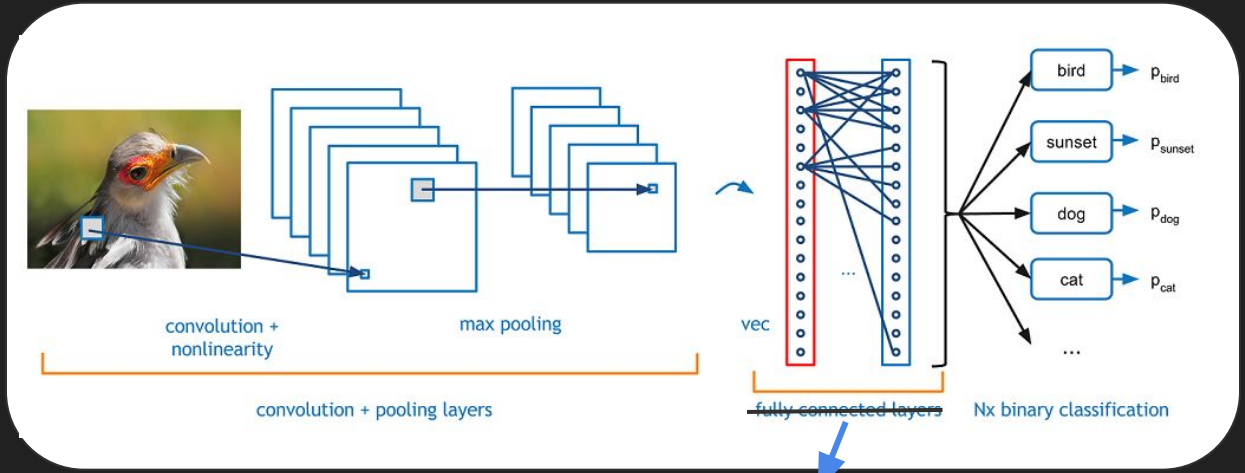
- Type of CNN that each image in training dataset contain ground truth of the classes
- Has been proved that its has more accuracy that CNN



(a) Image with GT boxes (b) 8×8 feature map (c) 4×4 feature map

Inception-v2

- A micro-CNN which capable of “filtering” the feature value in image
- Can improve performance during training and reduce overfitting problem
- Replace the Fully-connected layer

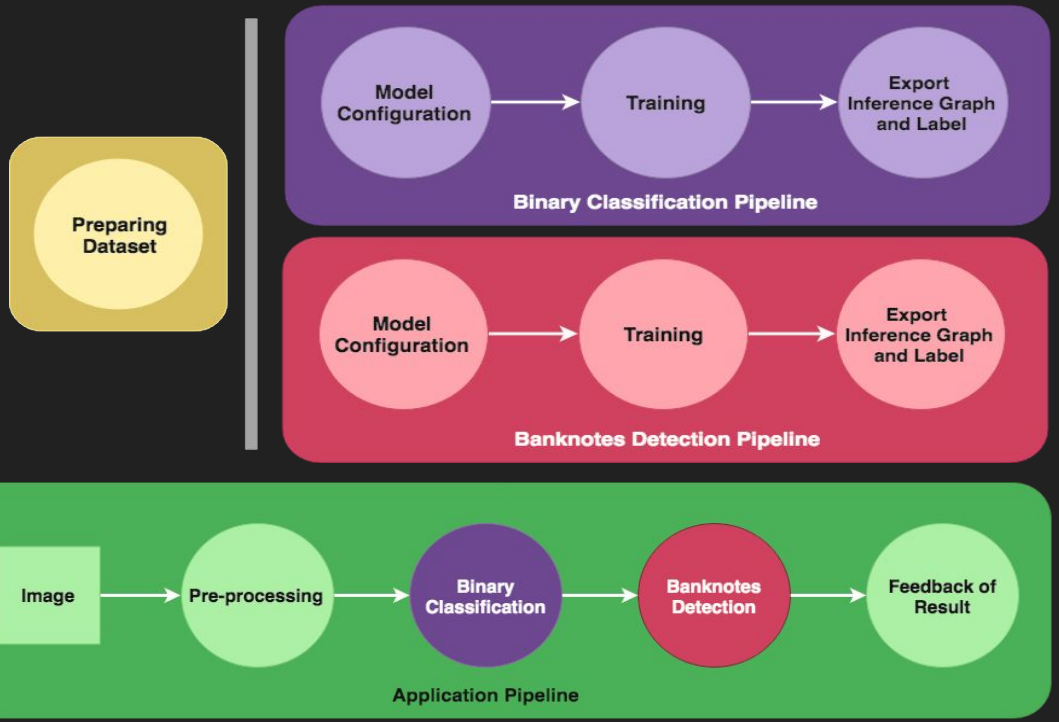


Framework

Framework

The design framework can be divided into two parts :-

1. Preparing the dataset & Model training pipeline



2. Application Pipeline

Preparing Dataset

Preparing Dataset

Collecting Dataset :

- 5 classes of Thai banknotes
- Use mobile camera to collect the banknote images
- Collect the negative images

Preparing Dataset

Pre-processing :

- Resize the images to ≈ 800x600 pixels
- Random crop
- Random rotation

Preparing Dataset

Create PASCAL VOC Dataset (for banknotes detection pipeline) :

- Hand-label the boxes on the banknotes for every images
- The result is XML file in PASCAL VOC format

Preparing Dataset

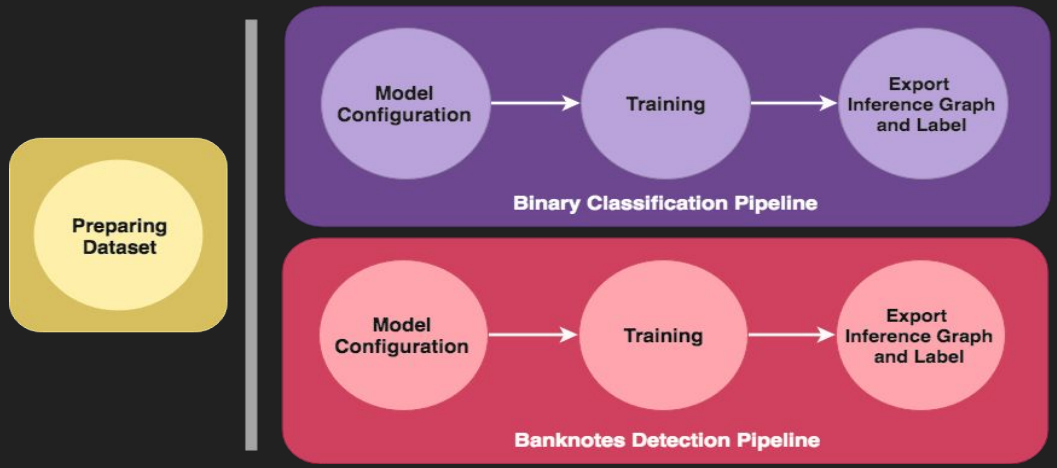
Generate TFRecord (for banknotes detection pipeline) :

- Combine the images and XML
- Saved as TFRecord, the standard file format used by TensorFlow

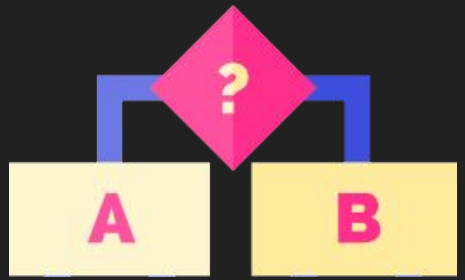
Framework

The design framework can be divided into two parts :-

1. Preparing the dataset & Model training pipeline



Binary Classification Pipeline



Binary Classification Pipeline

The size of the dataset:

	Banknotes	None	Total
Train	3,519	2,416	5,935
Validate	622	621	1,243

Binary Classification Pipeline

Model Configuration:

- Input size of 300x300 pixels
- 7 convolution layers (feature extractor)
  - Batch normalization & Max pooling on 1 - 5 layers
- 3 fully connected layers (classification)
  - Drop out between each layers
- Output 2 classes (binary)

Layer (type)	Output Shape
conv2d_1 (Conv2D)	(None, 300, 300, 32)
batch_normalization_1 (Batch	(None, 300, 300, 32)
max_pooling2d_1 (MaxPooling2	(None, 150, 150, 32)
conv2d_2 (Conv2D)	(None, 150, 150, 32)
batch_normalization_2 (Batch	(None, 150, 150, 32)
max_pooling2d_2 (MaxPooling2	(None, 75, 75, 32)
conv2d_3 (Conv2D)	(None, 75, 75, 64)
batch_normalization_3 (Batch	(None, 75, 75, 64)
max_pooling2d_3 (MaxPooling2	(None, 38, 38, 64)
conv2d_4 (Conv2D)	(None, 38, 38, 128)
batch_normalization_4 (Batch	(None, 38, 38, 128)
max_pooling2d_4 (MaxPooling2	(None, 19, 19, 128)
conv2d_5 (Conv2D)	(None, 19, 19, 256)
batch_normalization_5 (Batch	(None, 19, 19, 256)
max_pooling2d_5 (MaxPooling2	(None, 10, 10, 256)
conv2d_6 (Conv2D)	(None, 10, 10, 512)
conv2d_7 (Conv2D)	(None, 10, 10, 512)
max_pooling2d_6 (MaxPooling2	(None, 5, 5, 512)
flatten_1 (Flatten)	(None, 12800)
dense_1 (Dense)	(None, 1024)
dropout_1 (Dropout)	(None, 1024)
dense_2 (Dense)	(None, 1024)
dense_3 (Dense)	(None, 2)

Binary Classification Pipeline

Model Configuration:

- Each layer will be using ReLu as the activation function
- SGD will be used for gradient descent

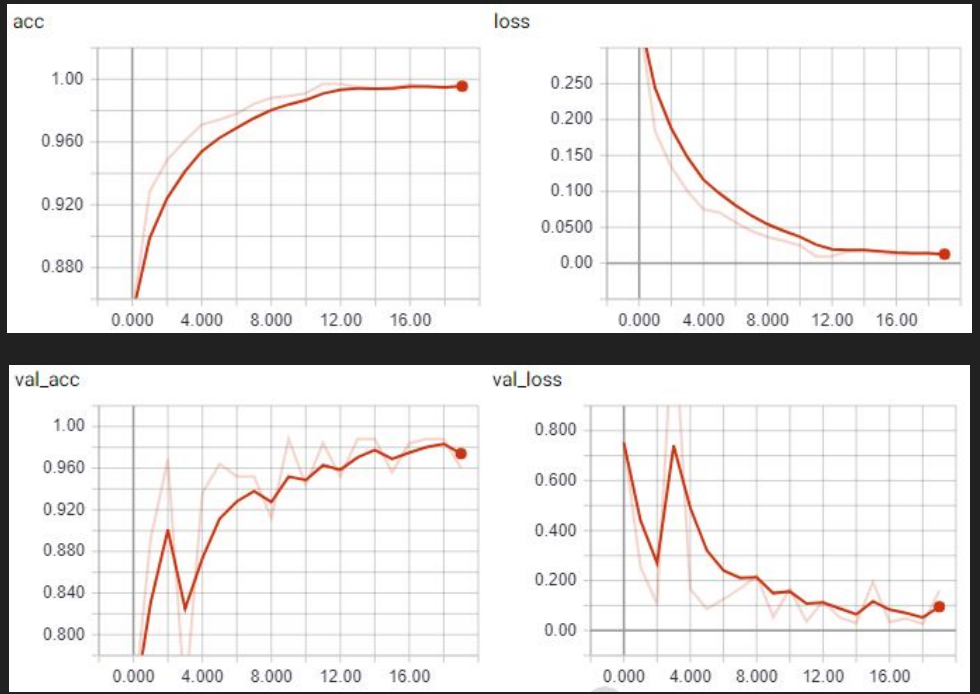
Layer (type)	Output Shape
conv2d_1 (Conv2D)	(None, 300, 300, 32)
batch_normalization_1 (Batch	(None, 300, 300, 32)
max_pooling2d_1 (MaxPooling2	(None, 150, 150, 32)
conv2d_2 (Conv2D)	(None, 150, 150, 32)
batch_normalization_2 (Batch	(None, 150, 150, 32)
max_pooling2d_2 (MaxPooling2	(None, 75, 75, 32)
conv2d_3 (Conv2D)	(None, 75, 75, 64)
batch_normalization_3 (Batch	(None, 75, 75, 64)
max_pooling2d_3 (MaxPooling2	(None, 38, 38, 64)
conv2d_4 (Conv2D)	(None, 38, 38, 128)
batch_normalization_4 (Batch	(None, 38, 38, 128)
max_pooling2d_4 (MaxPooling2	(None, 19, 19, 128)
conv2d_5 (Conv2D)	(None, 19, 19, 256)
batch_normalization_5 (Batch	(None, 19, 19, 256)
max_pooling2d_5 (MaxPooling2	(None, 10, 10, 256)
conv2d_6 (Conv2D)	(None, 10, 10, 512)
conv2d_7 (Conv2D)	(None, 10, 10, 512)
max_pooling2d_6 (MaxPooling2	(None, 5, 5, 512)
flatten_1 (Flatten)	(None, 12800)
dense_1 (Dense)	(None, 1024)
dropout_1 (Dropout)	(None, 1024)
dense_2 (Dense)	(None, 1024)
dense_3 (Dense)	(None, 2)

Binary Classification Pipeline

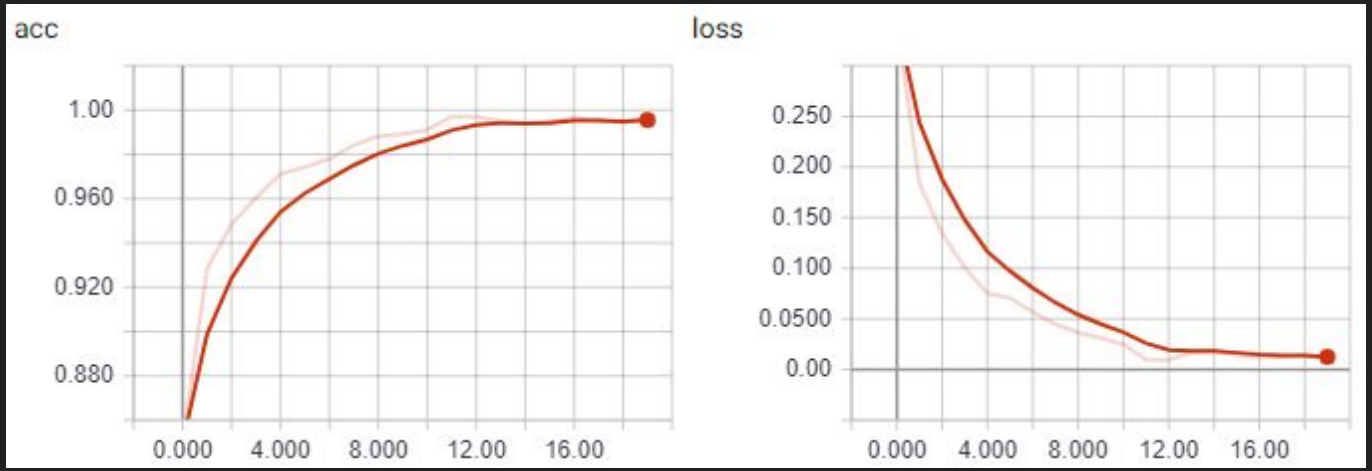
Training:

- 20 epochs
- Batch size of 10 images
- 594 iteration step

1 epoch = 594 iteration steps

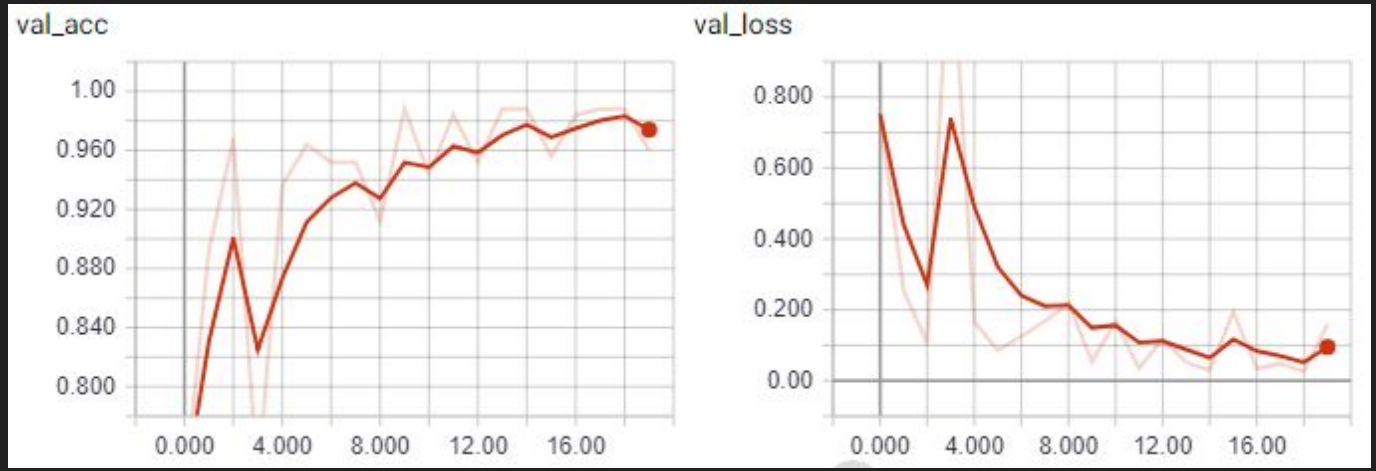


Binary Classification Pipeline



Accuracy and Loss on the training dataset

Binary Classification Pipeline



Accuracy and Loss on the validation dataset

Banknotes Detection Pipeline

Banknotes Detection Pipeline

The size of the dataset:

	20 baht	50 baht	100 baht	500 baht	1000 baht
Train	368	327	291	292	312
Validate	30	30	30	30	30

Banknotes Detection Pipeline

Model Configuration:

- We used the pre-trained model
- The model configuration is based on ssd_inception_v2_coco
- Fast detection rate over a reasonable accuracy
- Trade off between speed and accuracy

Banknotes Detection Pipeline

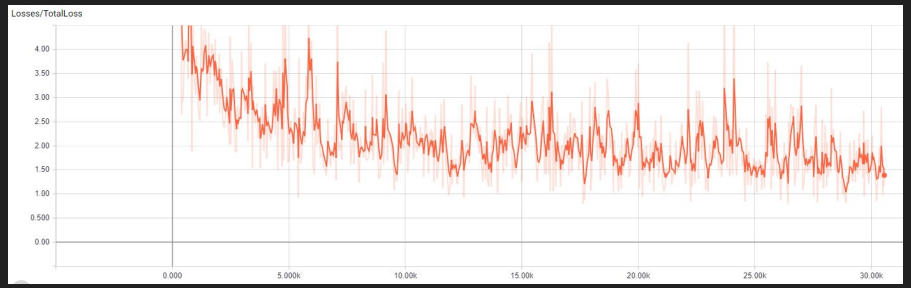
Model Configuration:

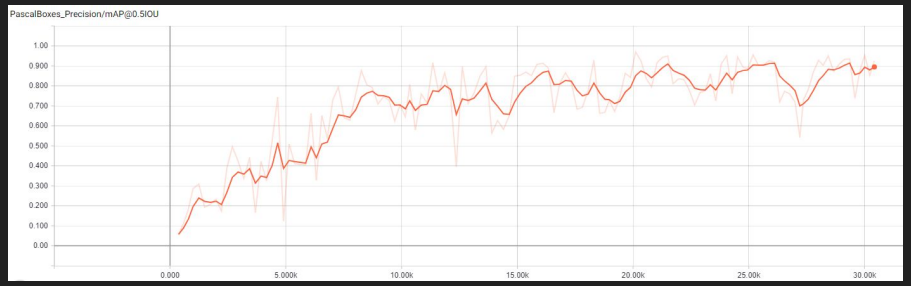
- Input size of 300x300 pixels
- Using inception_v2 (convnet) as feature extractor
- Using the hard mining method to collect the negative images
- RMSprop will be used for gradient descent

Banknotes Detection Pipeline

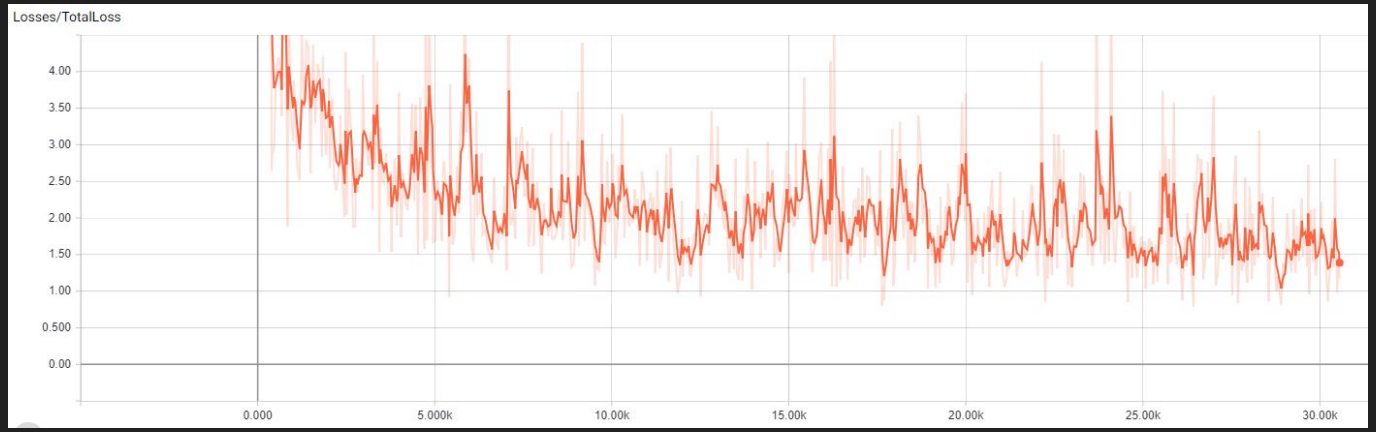
Training:

- 30,000 iterations
- Batch size of 12 images
- The trained-weights will be saved as the checkpoint in every 240 iterations



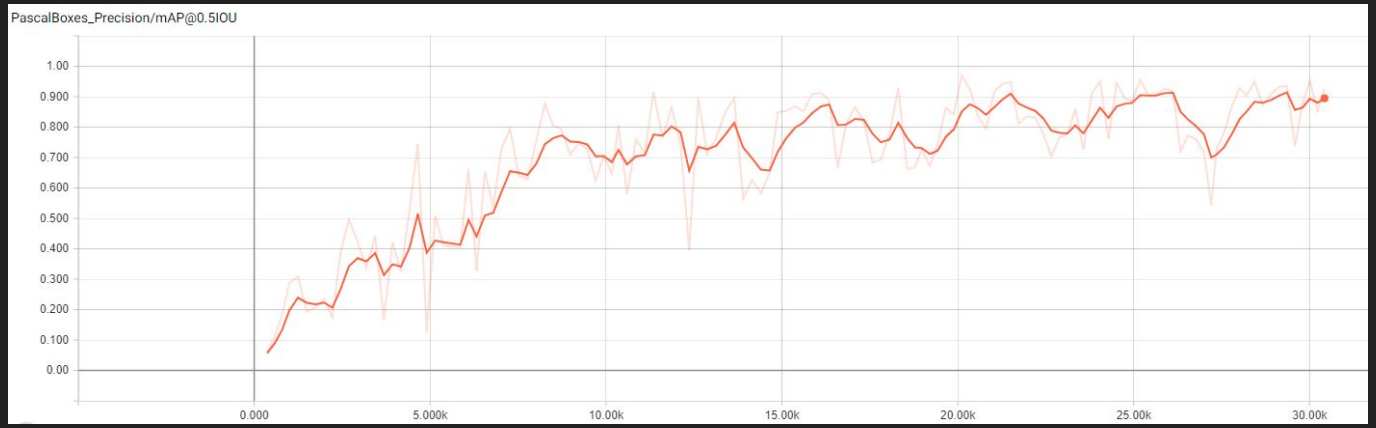


Banknotes Detection Pipeline



Total loss on the training dataset

Banknotes Detection Pipeline



The mean average precision on the validation dataset

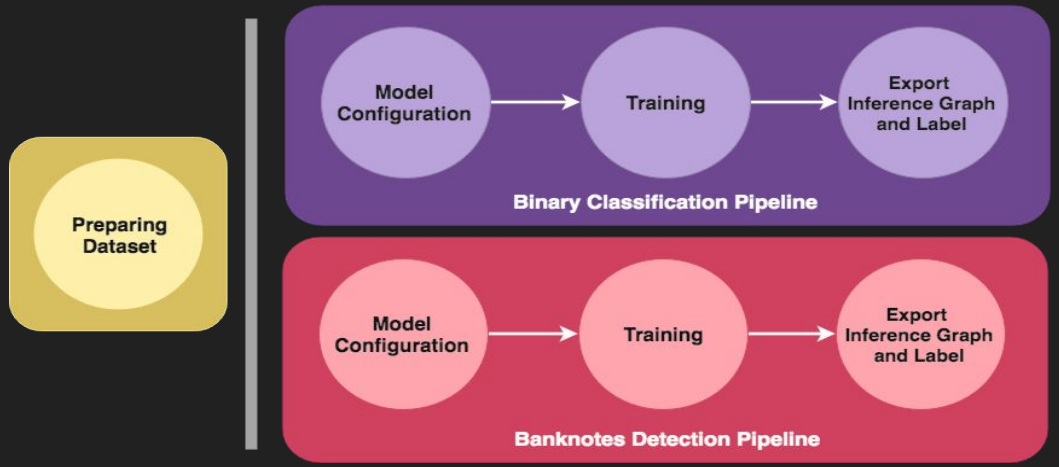
Exporting Inference Graph & Label

- Save the model (trained-weights) as a protocol buffer file (.PB)
- Save the label as a text file (.TXT)

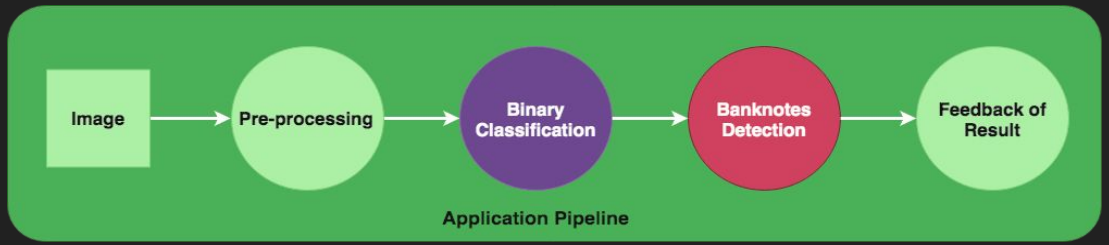
Framework

The design framework can be divided into two parts :-

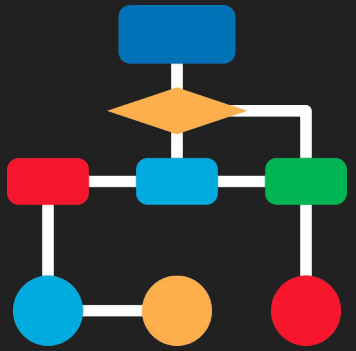
1. Preparing the dataset & Model training pipeline

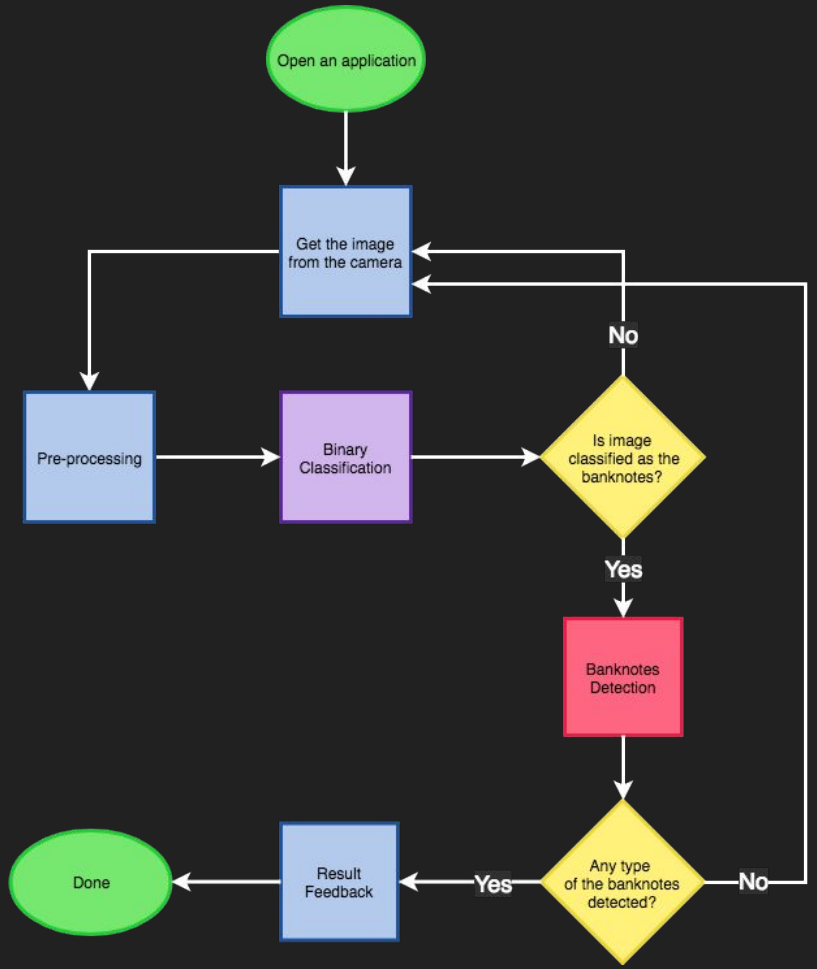


2. Application Pipeline



Application Pipeline

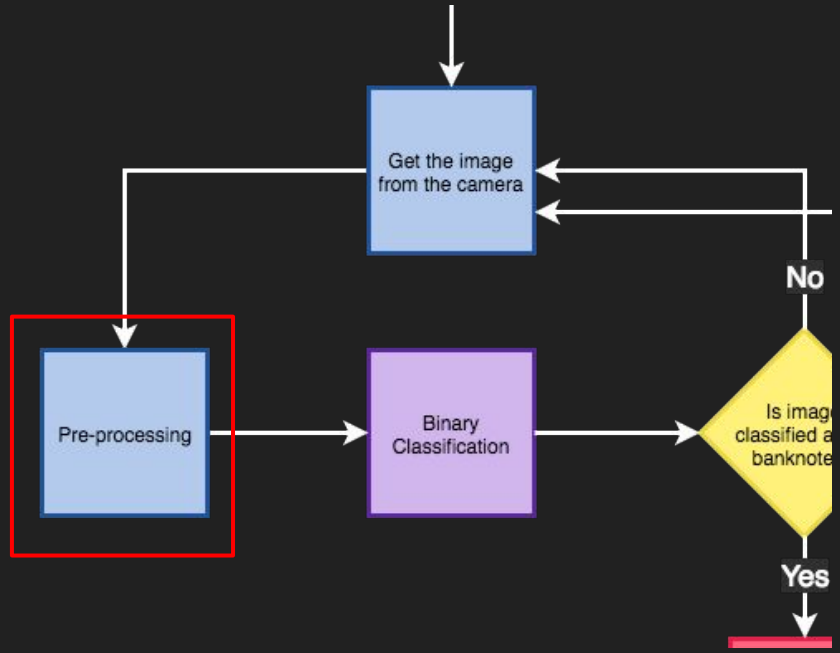



Open an application
Get the image from the camera
Pre-processing
Binary Classification
Is image classified as the banknotes?
No
Yes
Banknotes Detection
Any type of the banknotes detected?
No
Yes
Result Feedback
Done

Application Pipeline

i. Pre-processing:

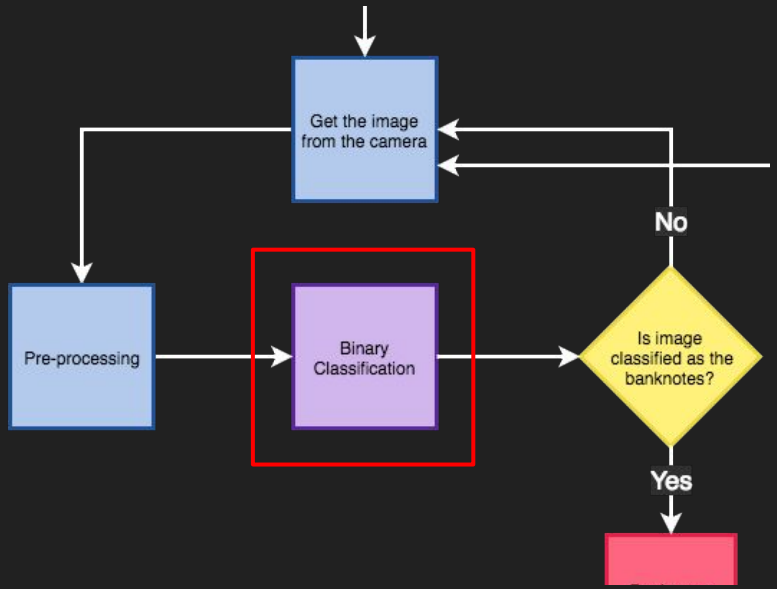
- The default size of the input image is 640x480 pixels.
- Convert YUV to RGB format
- Resize the image to 300x300 pixels
- Data standardization convert the integer to float base.



Application Pipeline

ii. Binary Classification:

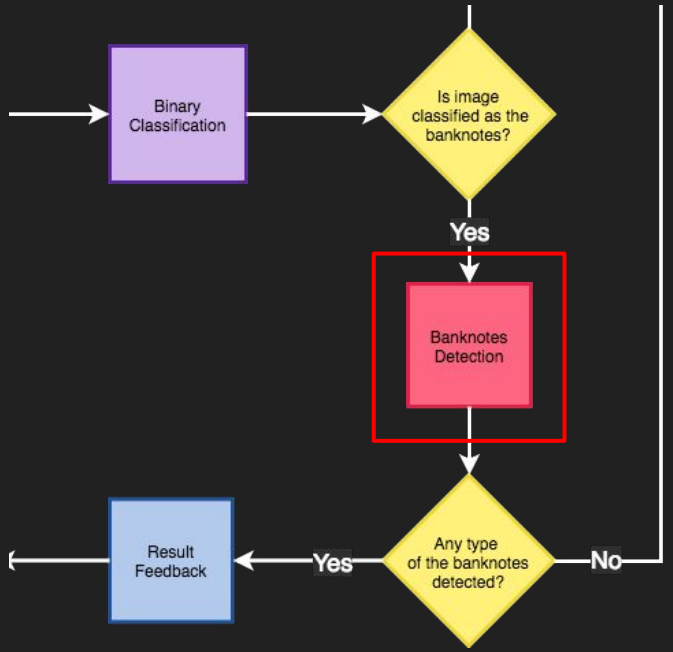
- The model will classify whether the input image contain any banknote or not.
- If it contain any banknotes, it will move to the next stage.
- If it not contain any banknotes, it will skip to new image.



Application Pipeline

iii. Banknotes Detection:

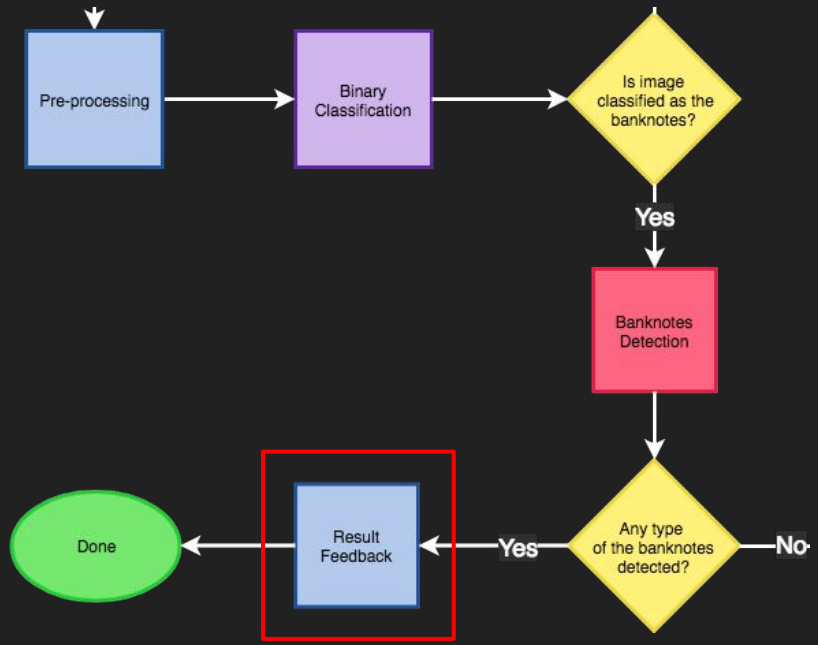
- The model will detect and classify the types of banknotes.
- If the result is higher than the minimum threshold, it will move to next stage.
- If the result is lesser than the minimum threshold, it will skip to the new image.



Application Pipeline

iv. Result Feedback:

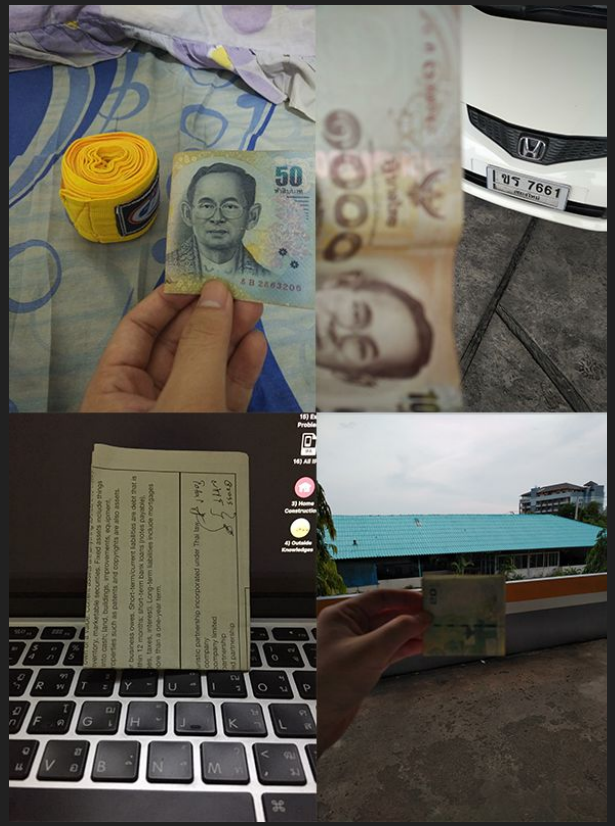
- The output will be in the form of sound.
- Using Google's Text-To-Speech API
- Currently, support two languages
  Thai & English



Evaluation

Evaluation

- The banknote detection and binary classification are separately evaluated
- Use same dataset for both models
- Dataset covers problem like occlusion, viewpoint variation, and scale



Evaluation

Dataset Preparation:

The datasets used for both model, the dataset will consist of the following images:

	20 baht	50 baht	100 baht	500 baht	1000 baht	None	Total
Detection Dataset	30	30	30	30	30	100	250

	Banknotes	None	Total
Binary Dataset	150	100	250

Evaluation

Result - Binary classification model:

		Classified	
		Banknote	None
True	Banknote	142	8
	None	10	90

	Precision	Recall	F-measure
Banknote	0.93	0.95	0.94
None	0.92	0.90	0.91
AVG.	0.93	0.93	0.92
Accuracy	0.93		

Evaluation

Result - Banknotes detection model:

		Detected					
		20 baht	50 baht	100 baht	500 baht	1000 baht	None
True	20 baht	27	1	0	0	0	2
	50 baht	0	25	1	0	0	4
	100 baht	0	1	26	1	0	2
	500 baht	0	3	0	24	0	3
	1000 baht	0	0	0	0	29	1
	None	3	3	2	3	1	88

	Precision	Recall	F-measure
20 baht	0.90	0.90	0.9
50 baht	0.76	0.83	0.79
100 baht	0.90	0.87	0.88
500 baht	0.86	0.80	0.83
1000 baht	0.97	0.97	0.97
AVG.	0.88	0.87	0.88
Accuracy	0.87		

Evaluation

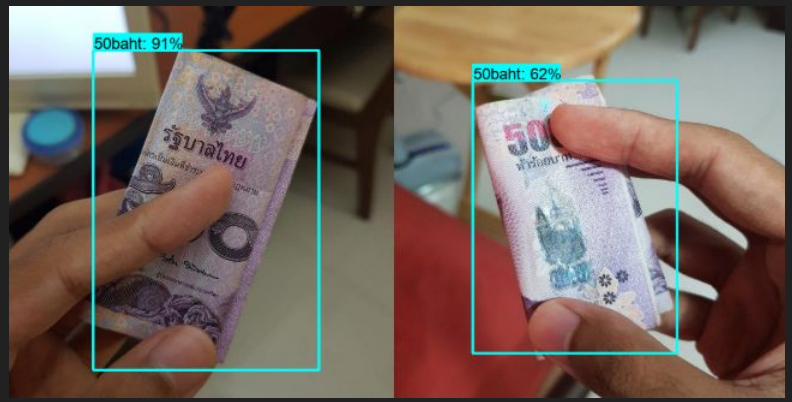
Result - Banknotes detection model:

- Detection time = around 0.2 second
- Detect object that similar to banknote
- Does not perform well on problem like occlusion

Evaluation

Result - Banknotes detection model:

Leftmost and Rightmost at the Back of Banknote



Partially Blocked Banknote

Demo

Q&A