STEEL STOCK HEAT NUMBER DIGITIZATION SYSTEM FOR LOGISTICS SUPPLY CHAIN MANAGEMENT

PHASE 2 PRESENTATION

Trainee Name : Ling Zhao Jia, Danny

Company Name : Keppel FELS Ltd

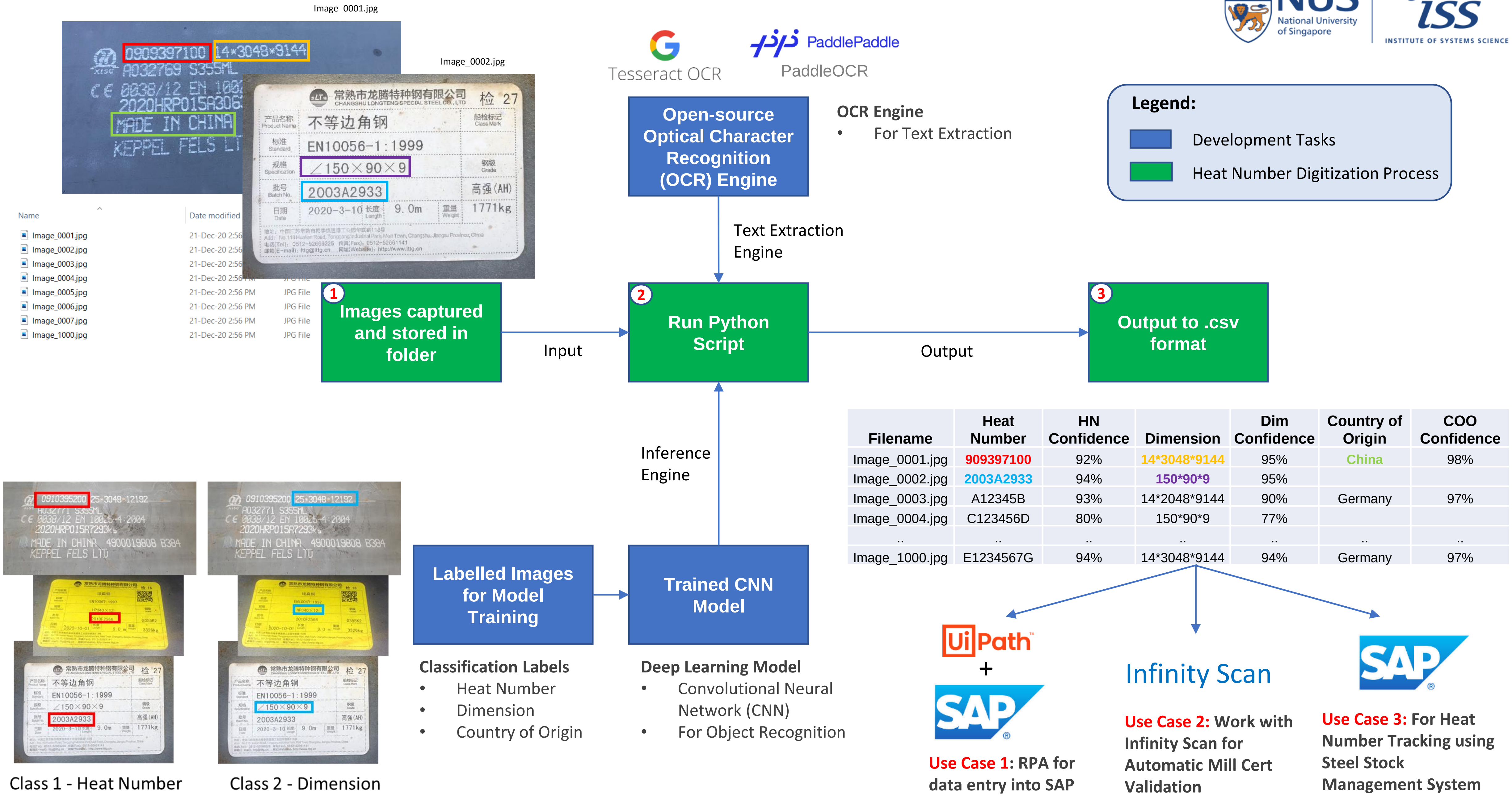
Company Supervisor Name & Email : Mr. Liew Wing Leong (wingleong.liew@keppelfels.com)

NUS-ISS Supervisor Name & Email : Dr. Tian Jing (tianjing@nus.edu.sg)

AGENDA (PHASE 2)

- System Overview Recap
- Original System Design
- Technical progress of Original System Design
- Change in System Requirements
- Technical progress of New System Design
- Issues & Challenges

System Overview



Name	Date modified	
Image_0001.jpg	21-Dec-20 2:56	
Image_0002.jpg	21-Dec-20 2:56	
Image_0003.jpg	21-Dec-20 2:56	
Image_0004.jpg	21-Dec-20 2:56 PM	JPG File
Image_0005.jpg	21-Dec-20 2:56 PM	JPG File
Image_0006.jpg	21-Dec-20 2:56 PM	JPG File
Image_0007.jpg	21-Dec-20 2:56 PM	JPG File
Image_1000.jpg	21-Dec-20 2:56 PM	JPG File

Open-source Optical Character Recognition (OCR) Engine

OCR Engine
- For Text Extraction

Legend:

🟦	Development Tasks
🟩	Heat Number Digitization Process

Text Extraction Engine

1 Images captured and stored in folder

Input

2 Run Python Script

Output

3 Output to .csv format

Inference Engine

Filename	Heat Number	HN Confidence	Dimension	Dim Confidence	Country of Origin	COO Confidence
Image_0001.jpg	909397100	92%	14*3048*9144	95%	China	98%
Image_0002.jpg	2003A2933	94%	150*90*9	95%		
Image_0003.jpg	A12345B	93%	14*2048*9144	90%	Germany	97%
Image_0004.jpg	C123456D	80%	150*90*9	77%		
..	..	..	..	..	..	..
Image_1000.jpg	E1234567G	94%	14*3048*9144	94%	Germany	97%

Labelled Images for Model Training

Trained CNN Model

Classification Labels
- Heat Number
- Dimension
- Country of Origin

Deep Learning Model
- Convolutional Neural Network (CNN)
- For Object Recognition

Class 1 - Heat Number

Class 2 - Dimension

Use Case 1: RPA for data entry into SAP

Use Case 2: Work with Infinity Scan for Automatic Mill Cert Validation

Use Case 3: For Heat Number Tracking using Steel Stock Management System

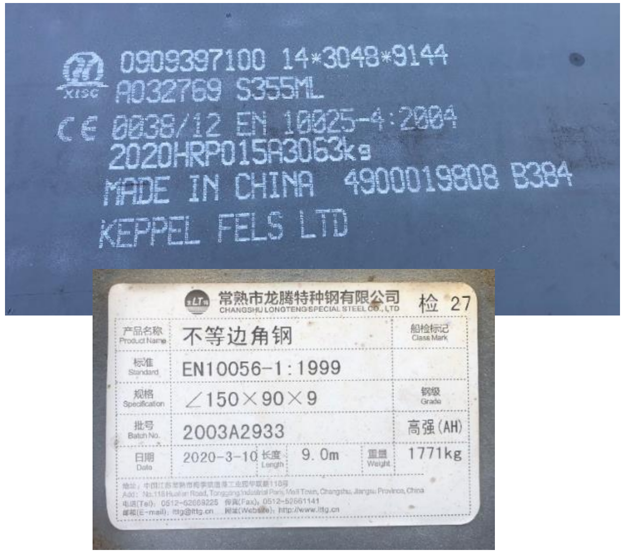
ORIGINAL SYSTEM DESIGN

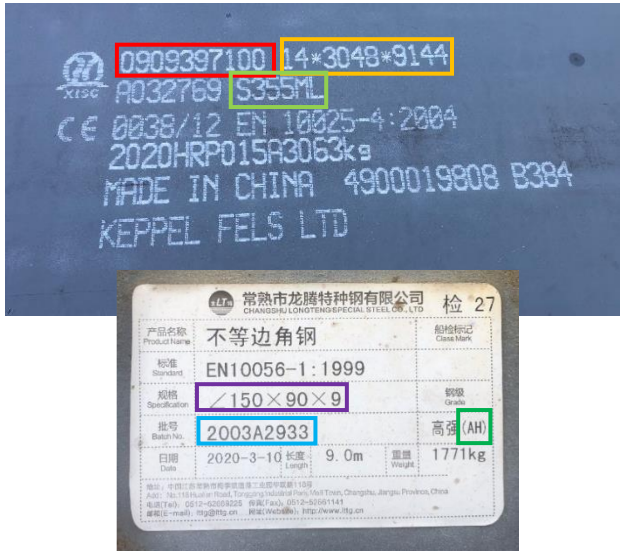
- **Mobile application based on user interaction for classification and recognition**

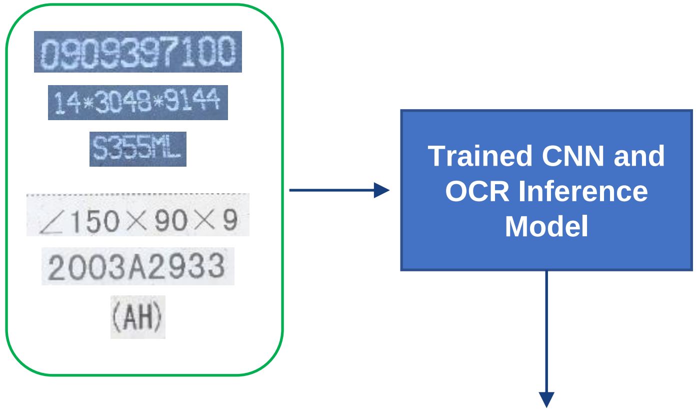
Step 1: User takes photo →

Step 2: User interaction to manually select the fields of interest (Using bounding boxes) →

Step 3: Bounding boxes are automatically cropped and images sent into trained CNN model for inference and output to .csv





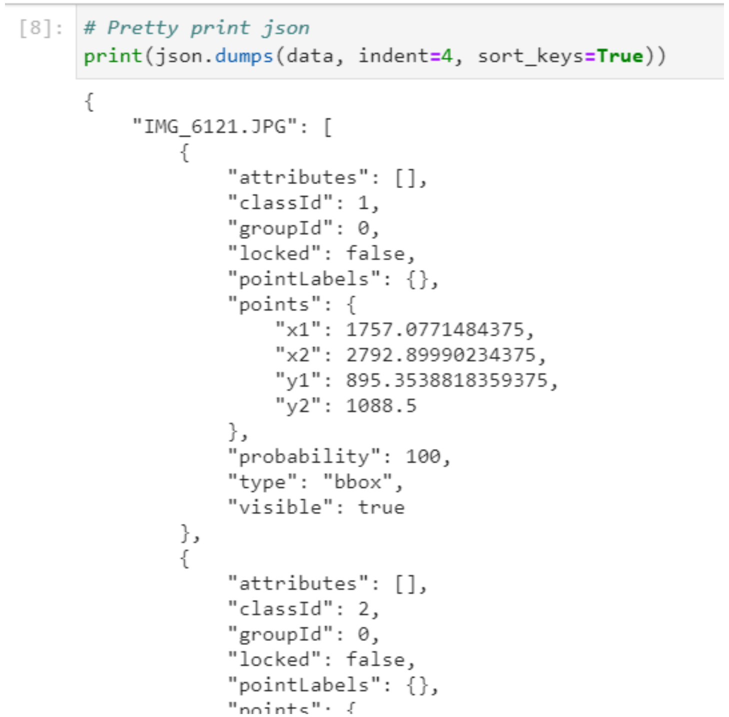


Trained CNN and OCR Inference Model

Filename	Heat Number	HN Confidence	Dimension	Dim Confidence	Country of Origin	COO Confidence
Image_0001.jpg	909397100	92%	14*3048*9144	95%	China	98%
Image_0002.jpg	2003A2933	94%	150*90*9	95%		
Image_0003.jpg	A12345B	93%	14*2048*9144	90%	Germany	97%
Image_0004.jpg	C123456D	80%	150*90*9	77%		
..	..	..	..	..	..	..
Image_1000.jpg	E1234567G	94%	14*3048*9144	94%	Germany	97%

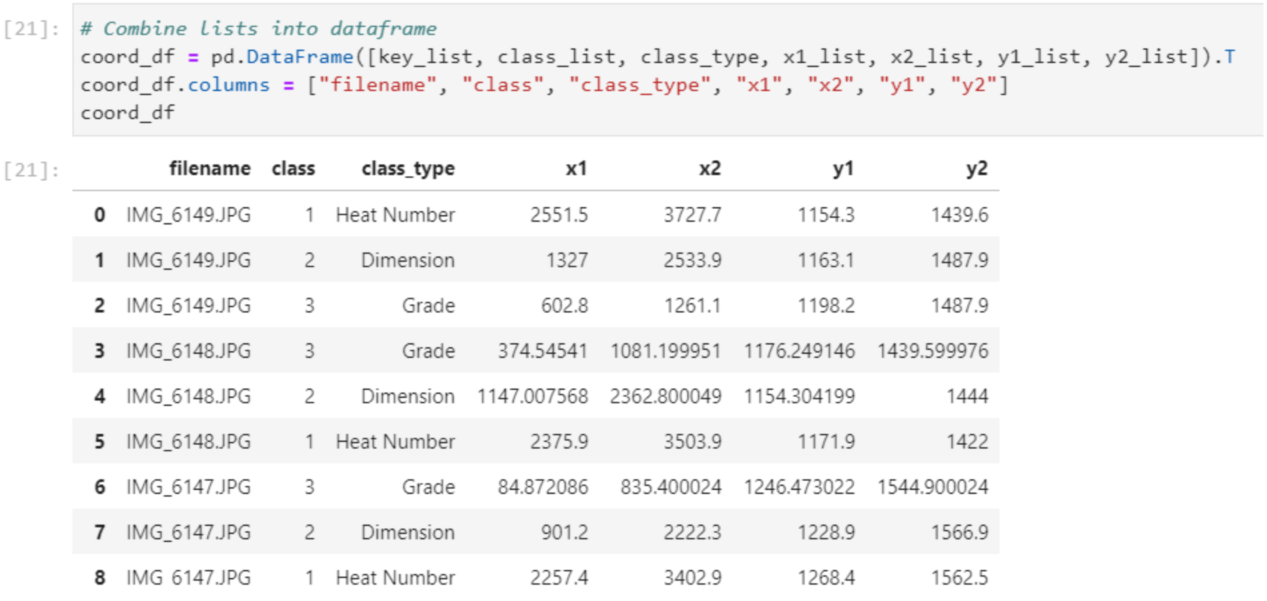
TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

- Image labelling done using SuperAnnotate – output is a .json file
- Python script to read .json file and parse data into dataframe



```
[8]: # Pretty print json
     print(json.dumps(data, indent=4, sort_keys=True))

     {
         "IMG_6121.JPG": [
             {
                 "attributes": [],
                 "classId": 1,
                 "groupId": 0,
                 "locked": false,
                 "pointLabels": {},
                 "points": {
                     "x1": 1757.0771484375,
                     "x2": 2792.89990234375,
                     "y1": 895.3538818359375,
                     "y2": 1088.5
                 },
                 "probability": 100,
                 "type": "bbox",
                 "visible": true
             },
             {
                 "attributes": [],
                 "classId": 2,
                 "groupId": 0,
                 "locked": false,
                 "pointLabels": {},
                 "points": {
```



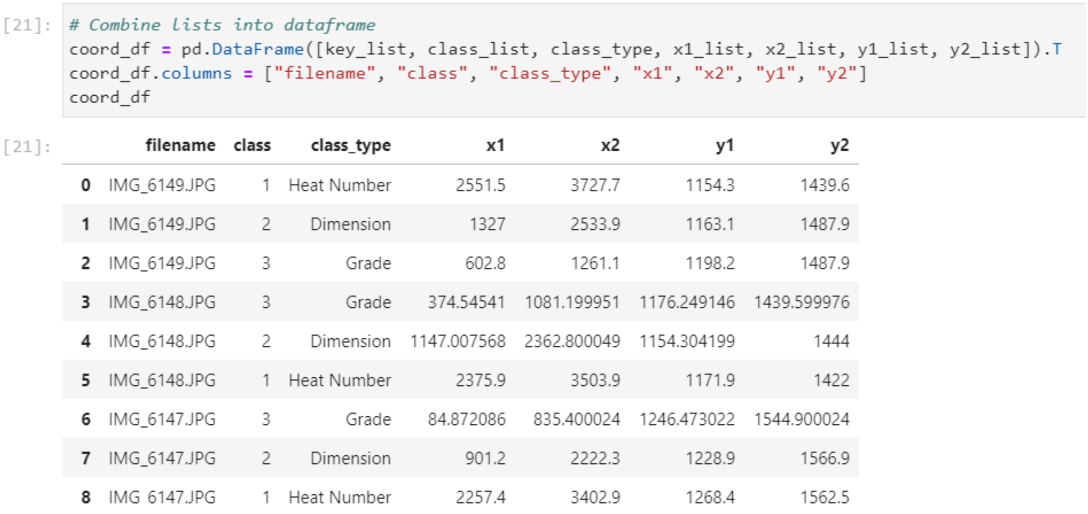
```
[21]: # Combine lists into dataframe
      coord_df = pd.DataFrame([key_list, class_list, class_type, x1_list, x2_list, y1_list, y2_list]).T
      coord_df.columns = ["filename", "class", "class_type", "x1", "x2", "y1", "y2"]
      coord_df
```

[21]:

	filename	class	class_type	x1	x2	y1	y2
0	IMG_6149.JPG	1	Heat Number	2551.5	3727.7	1154.3	1439.6
1	IMG_6149.JPG	2	Dimension	1327	2533.9	1163.1	1487.9
2	IMG_6149.JPG	3	Grade	602.8	1261.1	1198.2	1487.9
3	IMG_6148.JPG	3	Grade	374.54541	1081.199951	1176.249146	1439.599976
4	IMG_6148.JPG	2	Dimension	1147.007568	2362.800049	1154.304199	1444
5	IMG_6148.JPG	1	Heat Number	2375.9	3503.9	1171.9	1422
6	IMG_6147.JPG	3	Grade	84.872086	835.400024	1246.473022	1544.900024
7	IMG_6147.JPG	2	Dimension	901.2	2222.3	1228.9	1566.9
8	IMG 6147.JPG	1	Heat Number	2257.4	3402.9	1268.4	1562.5

TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

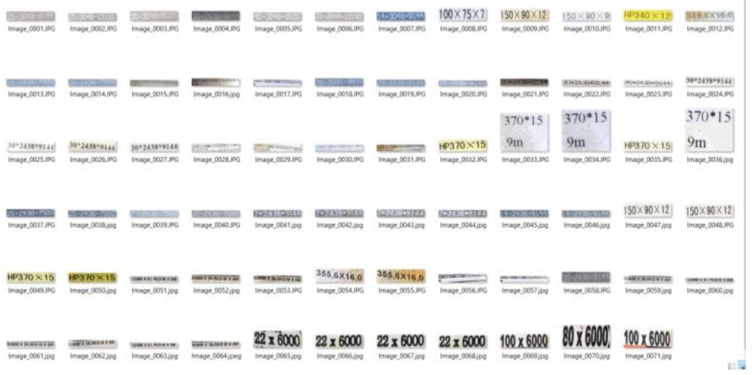
- Python script to use dataframe and automatically crop images based on coordinates

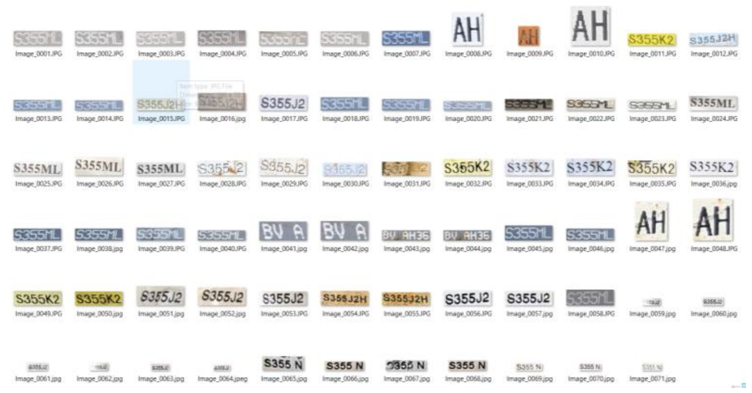


```
[21]: # Combine lists into dataframe
      coord_df = pd.DataFrame([key_list, class_list, class_type, x1_list, x2_list, y1_list, y2_list]).T
      coord_df.columns = ["filename", "class", "class_type", "x1", "x2", "y1", "y2"]
      coord_df
```

[21]:

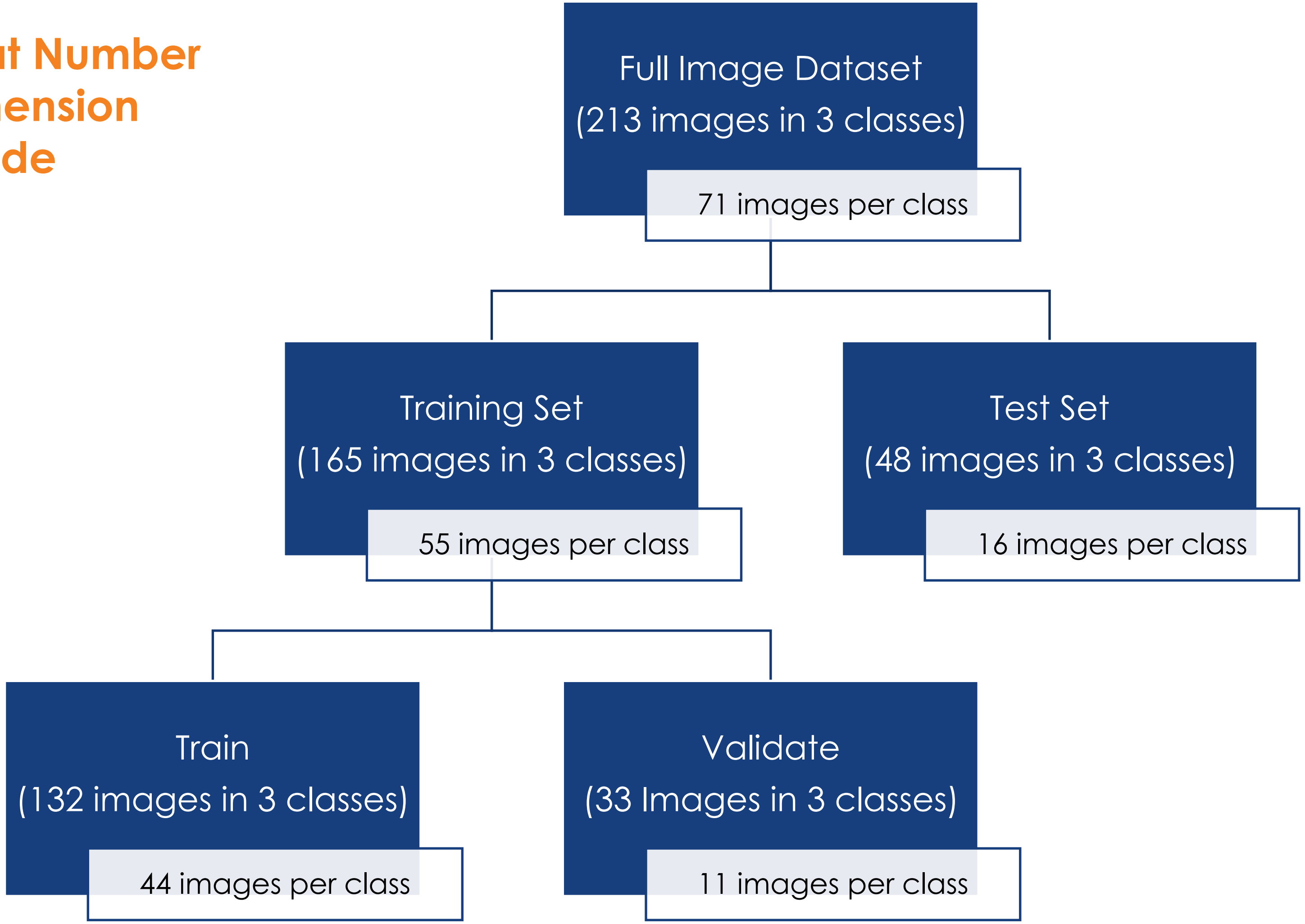
	filename	class	class_type	x1	x2	y1	y2
0	IMG_6149.JPG	1	Heat Number	2551.5	3727.7	1154.3	1439.6
1	IMG_6149.JPG	2	Dimension	1327	2533.9	1163.1	1487.9
2	IMG_6149.JPG	3	Grade	602.8	1261.1	1198.2	1487.9
3	IMG_6148.JPG	3	Grade	374.54541	1081.199951	1176.249146	1439.599976
4	IMG_6148.JPG	2	Dimension	1147.007568	2362.800049	1154.304199	1444
5	IMG_6148.JPG	1	Heat Number	2375.9	3503.9	1171.9	1422
6	IMG_6147.JPG	3	Grade	84.872086	835.400024	1246.473022	1544.900024
7	IMG_6147.JPG	2	Dimension	901.2	2222.3	1228.9	1566.9
8	IMG_6147.JPG	1	Heat Number	2257.4	3402.9	1268.4	1562.5





TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

Class 1: Heat Number
Class 2: Dimension
Class 3: Grade



Full Image Dataset
(213 images in 3 classes)
71 images per class

Training Set
(165 images in 3 classes)
55 images per class

Test Set
(48 images in 3 classes)
16 images per class

Train
(132 images in 3 classes)
44 images per class

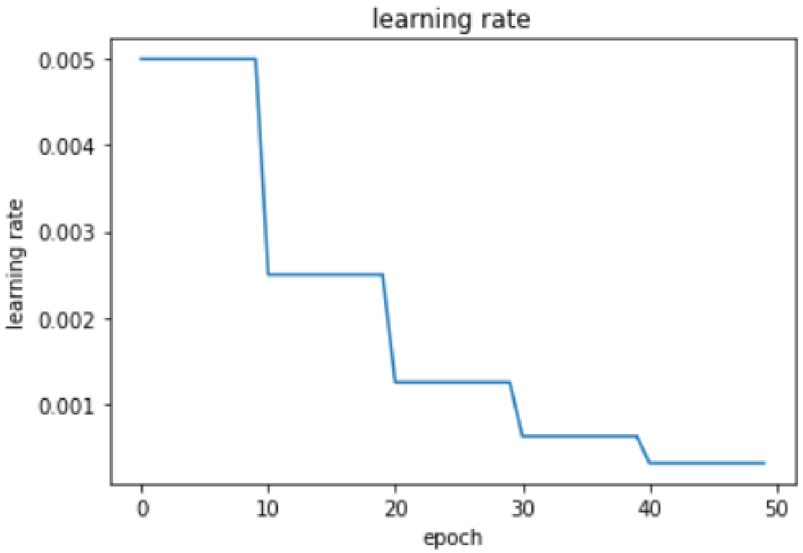
Validate
(33 Images in 3 classes)
11 images per class

TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

- **Convolutional Neural Network (CNN) Model Training**
  - Keras Applications using TensorFlow in Python
  - Trained images on VGG16, ResNet152v2, InceptionResNetv2 and Xception architecture using ‘imagenet’ pre-trained weights
  - Input shape = (224, 224, 3), images are RGB and resized to 224px x 224px
  - Real-time image augmentation using Keras in-built ImageDataGenerator class
  - Used transfer learning by freezing convolutional layers and applying global average pooling then training only the fully connected layers to classify into 3 classes – ‘Heat Number’, ‘Dimension’, ‘Grade’

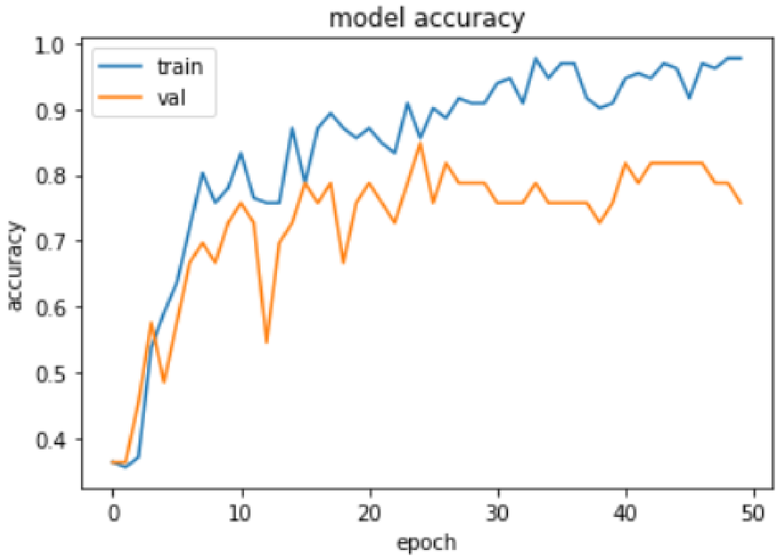
TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

- **Example: Xception model trained for 50 epochs**



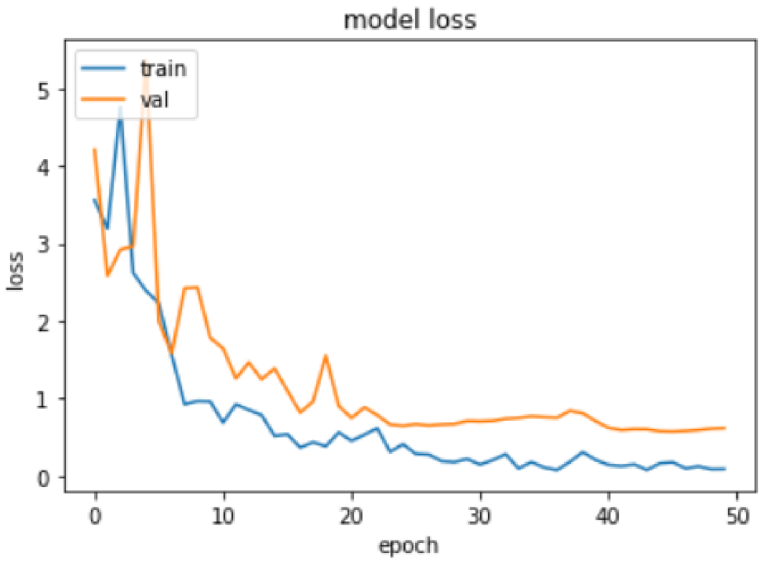
Learning rate scheduler

Initial Learning Rate = 0.005
Schedule = Half the learning rate every 10 epochs



Model Checkpoint

Weights saved based on best validation accuracy score of 84.8%



Test Set Evaluation

Evaluated model on test set of 48 images and achieved a 77% accuracy score

Class 3 – Grade had highest prediction errors

TECHNICAL PROGRESS – ORIGINAL SYSTEM DESIGN

- Original plan was to move on to ensemble voting using the 4 trained models for classification
- To incorporate OCR into the pipeline to recognize text and finalize the python script
- However, after discussion with Sponsor there are some changes to the system requirements

CHANGE IN SYSTEM REQUIREMENTS

- **Sponsor feels user interaction to manually select the text for Classification/OCR still requires the user to understand what text to select**
  - If user is new/untrained employee, he would not know what to select
- **Sponsor would prefer a fully automated solution that does not require the user to know what to select**
  - Send in an image and the script/algorithm would be able to detect text, classify and OCR

CHANGE IN SYSTEM REQUIREMENTS

- **Requires a detection and recognition model to satisfy new requirements**
- **Two open-source end-to-end deep learning models to consider**

YOLO	PaddleOCR
Developed for the purpose of object detection and recognition	Developed for the purpose of text detection and recognition
Widely popular	Not as popular
Able to make use of image prepared for the original system design	Not able to make use of image prepared for the original system design

- **To focus on PaddleOCR as it is built for text detection & recognition**

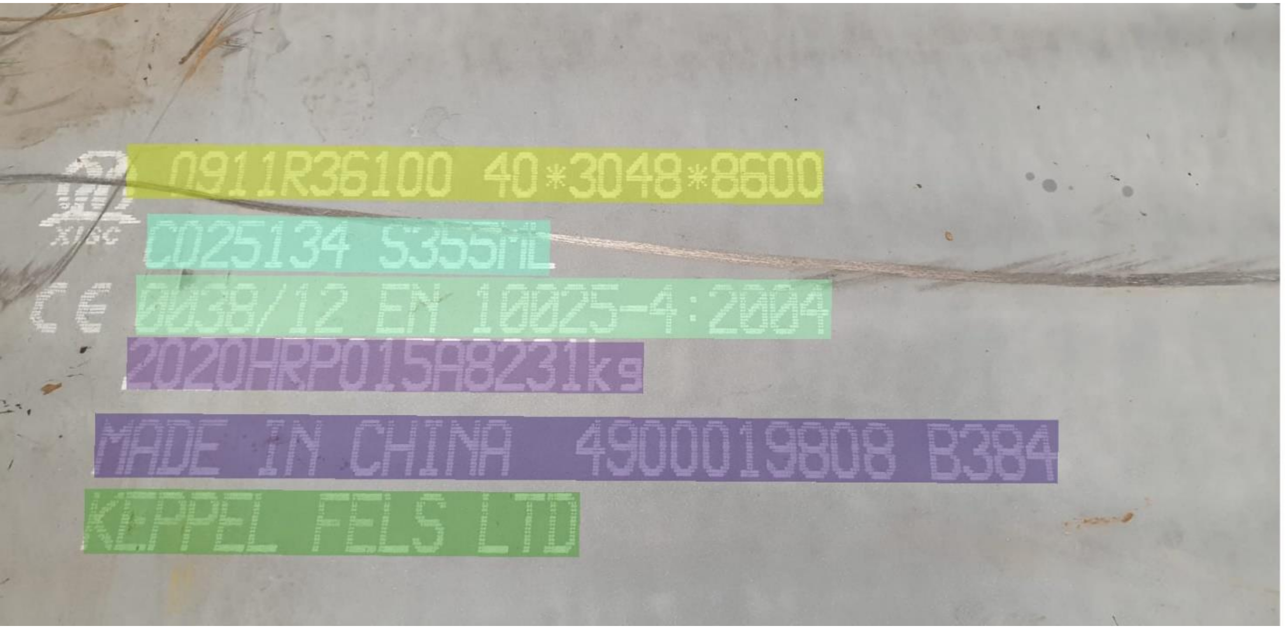
TECHNICAL PROGRESS (NEW SYSTEM REQUIREMENTS)

- **PaddleOCR Pipeline consists of 3 stages**
  - Text detection
  - Text angle classification (Detection box rectification for OCR)
  - Text recognition
- **Text Detection Algorithm Used**
  - Real-time Scene Text Detection with **D**ifferentiable **B**inarization (DB)
- **Text Recognition Algorithm Used**
  - Convolutional Recurrent Neural Network (CRNN)

TECHNICAL PROGRESS (NEW SYSTEM REQUIREMENTS)

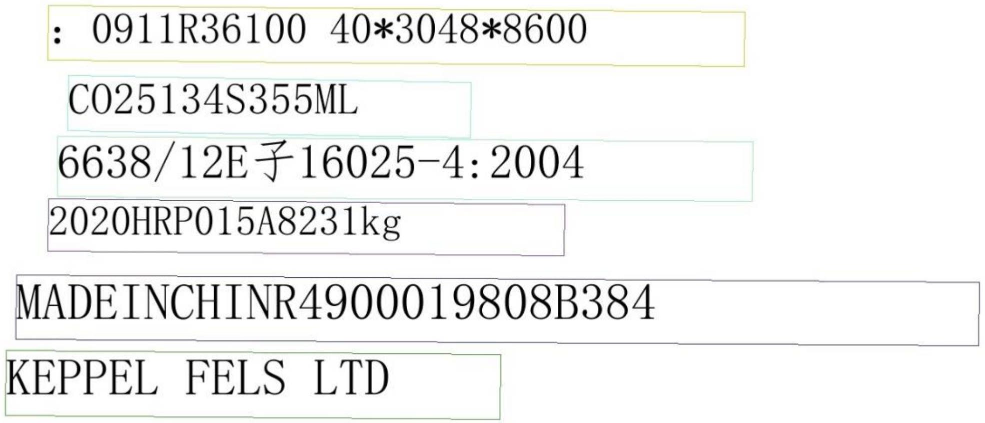
- ## Visualization
  - Run using PaddleOCR's "predict_system.py" via command-line

Actual Image



Highlighted boxes are generated by PaddleOCR's script to show the text detection bounding boxes. Original image does not have boxes.

Text Detection & Recognition Visualization



: 0911R36100 40*3048*8600

C025134S355ML

6638/12E子16025-4:2004

2020HRP015A8231kg

MADEINCHINR4900019808B384

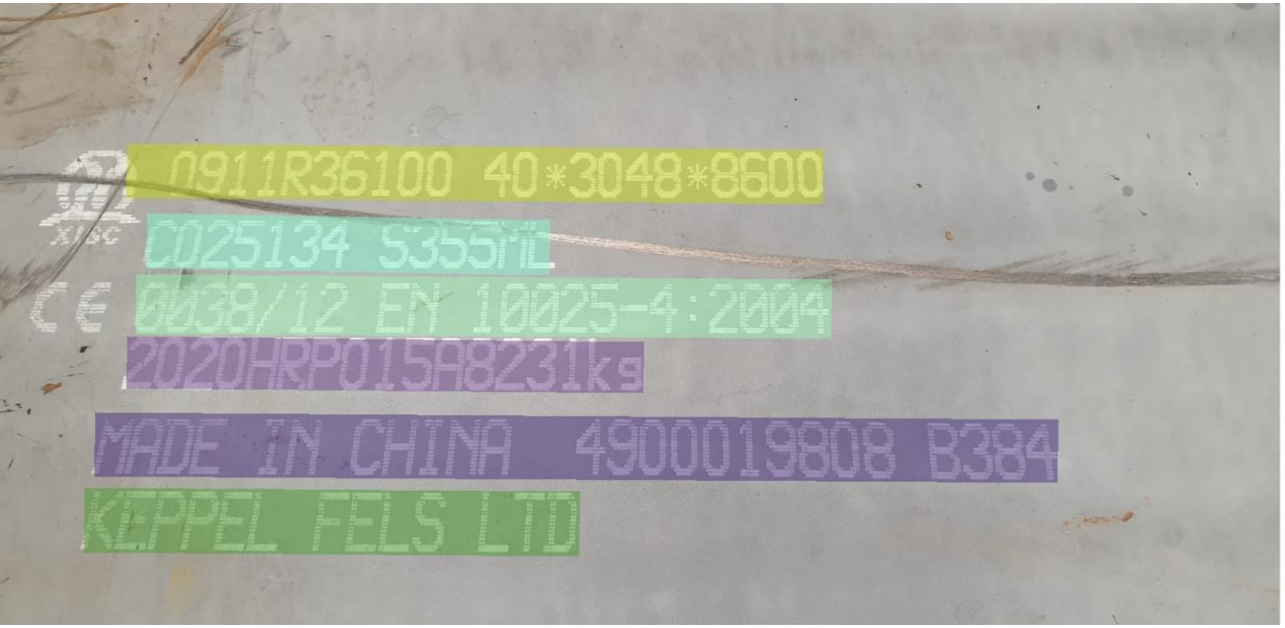
KEPPEL FELS LTD

ISSUES & CHALLENGES

- **Text recognized includes Chinese characters which are not relevant to our application**
  - To read through documentation and force only English character recognition or find a workaround
- **Model occasionally predicts the following characters wrongly:**
  - ‘0’ as ‘O’, ‘B’ as ‘8’, ‘*’ as ‘米’
  - Model may require further training on new images to correctly predict the characters
- **PaddleOCR does not have a tool to output text to csv**
  - To write a Python script to parse the output text onto csv

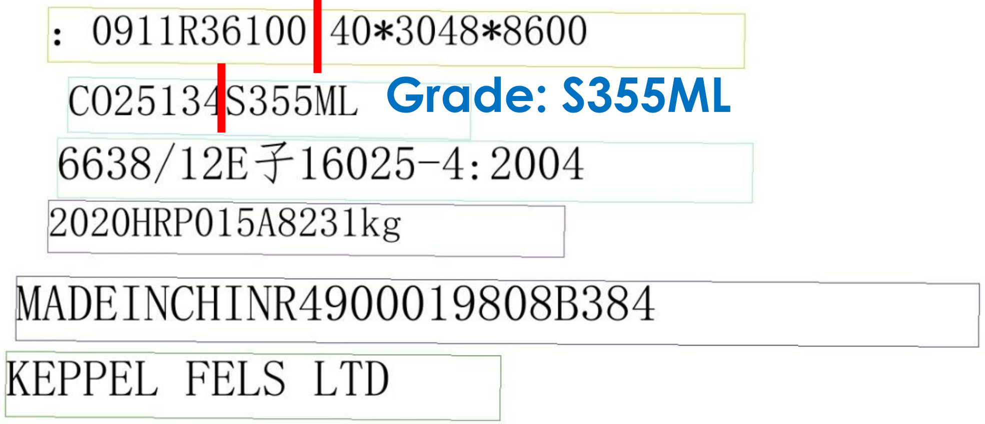
ISSUES & CHALLENGES

- **Text detection is whole line and not separated by spacing**
  - To use a rule-based approach to split the lines up
  - “Heat Number” is always first 10 characters of first line
  - “Dimension” is always after “Heat Number” & contains “*”



Heat Number: 0911R36100

Dimension: 40*3048*8600



: 0911R36100 40∗3048∗8600

C025134S355ML

6638/12E子16025-4:2004

2020HRP015A8231kg

MADEINCHINR4900019808B384

KEPPEL FELS LTD

Grade: S355ML

Q&A

Any Questions?

END