[CVS2021] Computer Vision

PCB Defect Classification

Final Presentation

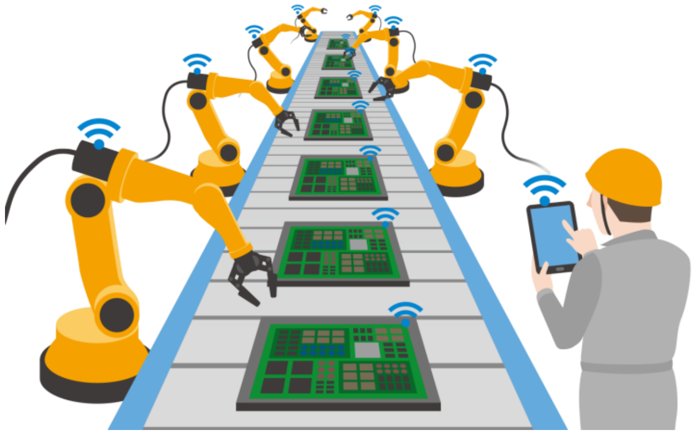
2020011135 Kyoosung So
2020011132 Euisuk Chung
2020021319 Yunseung Lee

Contents

Backgrounds

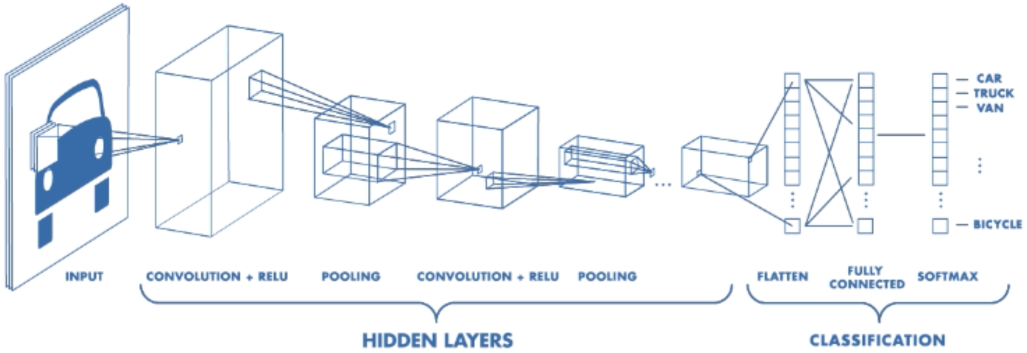
[Motivation 1]

- ✓ Lots of image data are being generated at industrial sites.
- ✓ By analyzing these images, more efficient work and higher profit are expected.



[Motivation 2]

- ✓ Convolutional Neural Network (CNN) has been used for image processing model due to its high performance on many tasks such as classification, detection.



[Research Question]

Q1: Can we <u>classify PCB(Printed Circuit Board) defects</u> using CNN models?

Q2: Is there <u>performance difference</u> depending on the type of CNN models?

Methods

[Dataset: DeepPCB]

- ✓ 1,500 image pairs(total 3000) of PCB (Printed Circuit Board)
- ✓ 6 Types of Defect Class: [1]open, [2]short, [3]mouse-bite, [4]spur, [5]pin hole, [6]spurious copper

[Data Preprocessing]

- ✓ Created multi-label based on given label

[Experiment Settings]

- ✓ Train : Val : Test = 6.8 : 1.2 : 2
- ✓ Epoch set 50 (for comparison)

[Tested Algorithms]

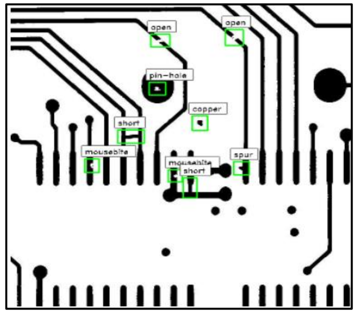
- ✓ VGGNet (11,16,19)
- ✓ DenseNet (121, 169, 201)
- ✓ ResNet (18, 50, 101)
- ✓ EfficientNet (b4, b5, b7)

[Tested Parameter]

- ✓ Use multiple CNN layer depths (Depends on the model)
- ✓ Use Pretrained Model (O/X) (ImageNet - 1000 class)
- ✓ Use Oversampling Method (O/X) (Using ImbalancedDatasetSampler)

PCB Fault Detection → Multi-Label Classification

Train set



Label: [1, 0, 1, 0, 1, 1]

→

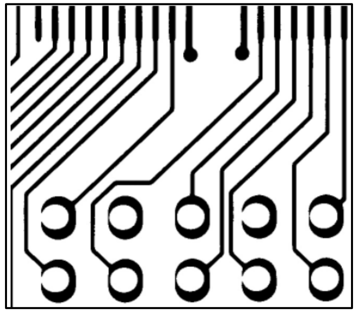
Train our model

CNN Models

(ResNet-50, etc.)

→

Test set



→

Predict & Get Accuracy

[0.95, 0.13, 0.31, 0.01, 0.56, 0.99]

↓

[1, 0, 0, 0, 1, 1]

ACC: 87.51%

Experiments

Model	Version	Batch Size	Pretrained (T/F)	Oversampled (T/F)	Train ACC	Val ACC	Test ACC
VGGNet	11	8	F	F	58.06%	**59.86%**	**59.25%**
VGGNet	16	8	F	F	**58.17%**	**59.86%**	**59.25%**
VGGNet	19	8	F	F	**58.17%**	**59.86%**	**59.25%**
VGGNet	19	8	T	F	**58.17%**	**59.86%**	**59.25%**
VGGNet	19	8	F	T	56.76%	40.14%	40.75%

[VGGNet - 11, 16, 19]

Model	Version	Batch Size	Pretrained (T/F)	Oversampled (T/F)	Train ACC	Val ACC	Test ACC
DenseNet	121	8	T	F	**91.39%**	91.11%	**91.86%**
DenseNet	121	8	F	F	91.26%	**91.16%**	91.67%
DenseNet	121	8	F	T	90.68%	77.37%	80.14%
DenseNet	169	8	F	F	91.23%	90.93%	62.78%
DenseNet	201	8	F	F	89.98%	91.11%	40.75%

[DenseNet - 11, 16, 19]

Model	Version	Batch Size	Pretrained (T/F)	Oversampled (T/F)	Train ACC	Val ACC	Test ACC
ResNet	18	16	T	T	**98.46%**	**92.18%**	**93.42%**
ResNet	101	16	T	F	91.41%	90.97%	91.78%
ResNet	18	16	T	F	92.57%	91.62%	91.64%
ResNet	50	16	T	F	91.60%	91.06%	89.42%
ResNet	18	16	F	T	96.60%	84.21%	88.25%
ResNet	50	16	F	F	58.05%	60.19%	59.44%
ResNet	18	16	F	F	88.10%	85.46%	46.81%
ResNet	101	16	F	F	89.39%	90.65%	45.69%

[ResNet – 18, 50, 101]

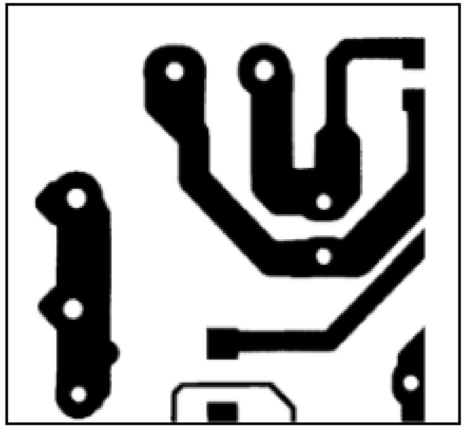
Model	Version	Batch Size	Pretrained (T/F)	Oversampled (T/F)	Train ACC	Val ACC	Test ACC
EfficientNet	B4	8	T	T	**99.67%**	93.94%	**98.47%**
EfficientNet	B4	8	T	F	95.59%	**95.37%**	95.17%
EfficientNet	B5	2	T	F	92.03%	91.16%	92.25%
EfficientNet	B7	2	T	F	91.51%	90.60%	92.14%
EfficientNet	B4	8	F	F	57.84%	59.86%	59.25%
EfficientNet	B5	2	T	T	94.31%	56.16%	56.42%
EfficientNet	B4	8	F	T	88.68%	53.38%	52.89%
EfficientNet	B7	2	T	T	88.86%	48.61%	48.86%

[EfficientNet – B4, B5, B7]

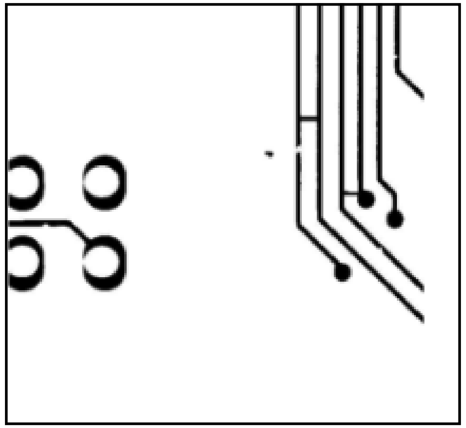
Best Model Result

Pretrained EfficientNet B4 Oversampled Result (ACC : 98.4722%)

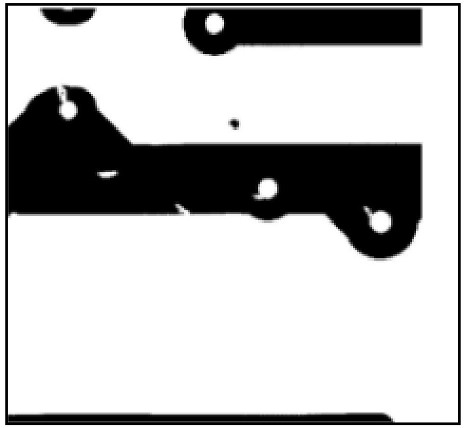
```
Ground Truth     : [0 0 0 0 0 0]
Model Prediction: [0 0 0 0 0 0]
```



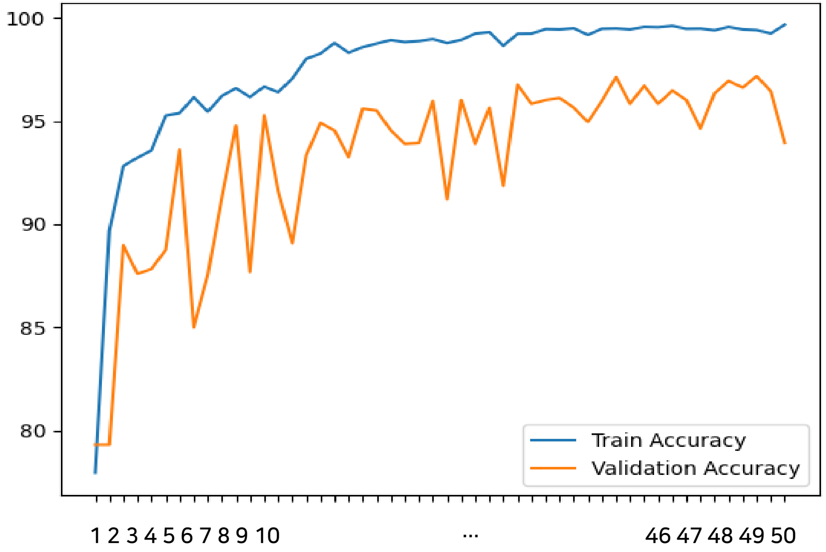
```
Ground Truth     : [1 1 1 0 1 0]
Model Prediction: [1 1 1 0 1 0]
```



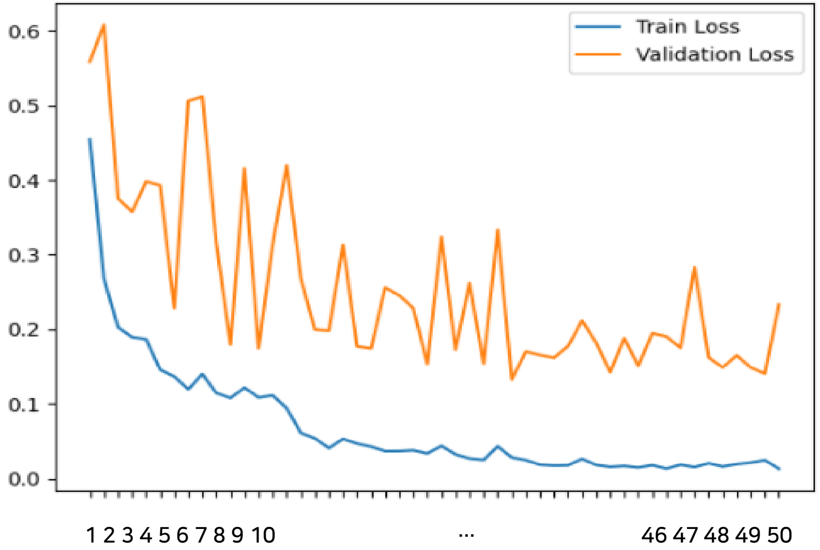
```
Ground Truth     : [1 0 1 0 1 1]
Model Prediction: [1 0 1 0 1 1]
```



Model Train/Val Accuracy Plot (per epoch)



Model Train/Val Loss Plot (per epoch)



Conclusion

Lessons Learned

- ✓ Model generally improves with the following conditions :
  1) When class imbalance of the training dataset is resolved (using oversampling)
  2) When weights are initialized (using pretrained model)
- ✓ Depth of the model does not guarantee model performance. It can even cause overfitting.

Limitations

- ✓ Due to limited amount of time, we were only conducted experiments on limited conditions :
  1) Few CNN based Models : VGGNet, DenseNet, ResNet, EfficientNet
  2) Fixed Fully Connect Layer : (1000, 128) → (128, 6)
  3) Fixed Epoch : 50
  4) Fixed Oversampling Method : ImbalancedDatasetSampler
  5) Different Batch Size per model due to lack of computing power

Future Works

- ✓ Try changing the layers and filter size of the model manually to prevent overfitting and improve performance
- ✓ Try finding just batch size and epoch.
- ✓ Implement early stopping methods to prevent overfitting

Thank You!