Deep-Learning Based Classification Models for Wafer Defective Pattern Recognition

Jeong Cheol Seo
jeongseo@Stanford.edu
Recorded presentation: https://youtu.be/otHMwuNgaTI

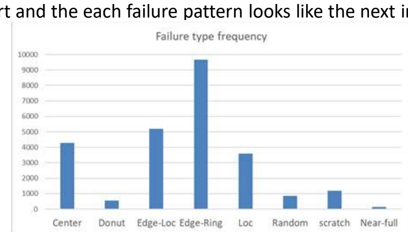
ABSTRACT AND OBJECTIVE

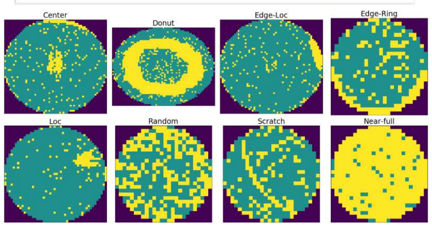
Deep-learning based multi-class classification models are trained for wafer map pattern of 9 defectives using four different architectures and a published wafer map dataset in Kaggle[1].

- Improving the ability to recognize the defect patterns of the wafer maps is required [2]
- VGG-16, ResNet-50 and two Simplified VGG-16 are trained on NVIDIA RTX 2080(8G) with Keras.
- Dataset is unbalanced so two data augmentation ways are used: Convolutional Autoencoder and Rotating.
- Rotating is generating better data to make better prediction
- 'Scratch' class F1 score is the lowest and Class Activation Map shows the reason.

DATA AND FEATURES

- This wafer map dataset consist of 172950 images with manual label(9 labels)
- The last label('none' : no defect) occupies 85.2%.
- Each failure type data(14.8%) is distributed like the following chart and the each failure pattern looks like the next images.





[Failure wafer map]

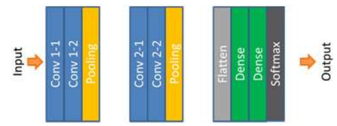
- **Training and Test data** split is done by 80% : 20%

	Center	Donut	Edge-Loc	Edge-Ring	Loc	Near-full	Random	Scratch	none	Total
Traing	1889	6	593	43	462	35	110	86	11839	15063
Test	475	1	154	13	111	13	27	16	2956	3766

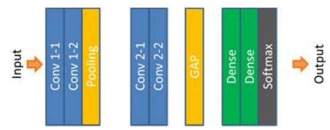
- **Data Augmentation**
  : For making a balanced training data (**10K for each class**)
  1) **Convolutional Autoencoder** : reconstructed images[3]
  2) **Rotating** : degrees randomly chosen
- Due to noisy ground truth labeled data, minimum 10K is required[4]
- Test is done using the originally distributed 20% dataset.

METHODS AND MODELS

- Deep Learning Models used for wafer map classification
  - **VGG-16**
  - **ResNet-50**
  - Simplified VGG-16(**SV**) : initial 2 Conv. and FC module



  - **GAP-SV** : Global Average Pooling layer used



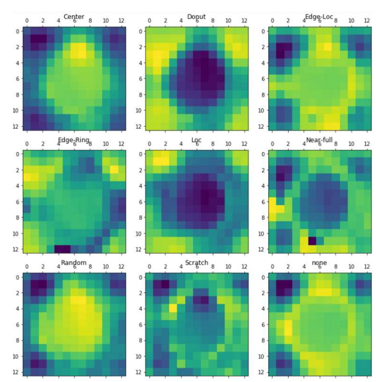
- **Hyperparameter tuning for customized models(SV / GAP-SV)**
  - **Initializer** : Xavier(uniform/normal) or He(uniform/normal)
    ➔ **Xavier(uniform)** show the best in experiments(Adam/SV)

Initializer	Xavier_normal	**Xavier_uniform**	he_normal	he_uniform
F1 macro avg.	0.82	**0.89**	0.83	0.84

  - **Optimizer** : Adam is better than RMSprop
  - **Learning rate(Lr) , Regularizer-L2(λ) and Dropout** (experiments done) :
    ➔ Dropout(0.4), L2(0.001), Lr(0.001) picked as shown in the table

(Adam/SV)

Dropout	L2(λ)	F1 score(macro avg.)	
		Lr=0.001	Lr=0.002
0.2	0.001	0.86	0.86
	0.01	0.84	0.77
	0.02	0.83	0.74
	0.03	0.81	0.79
0.3	0.001	0.86	0.84
	0.01	0.83	0.84
	0.02	0.67	0.82
	0.03	0.82	0.84
0.4	**0.001**	**0.89**	0.86
	0.01	0.81	0.85
	0.02	0.84	0.88
	0.03	0.64	0.79



[Class Activation Map(CAM)]

RESULTS

- Precision, Recall and F1 Score are reviewed for all test case
- **F1 Score is used for final metrics (Accuracy is same for all)**

[F1 Score(Macro avg) by data augment method and model]

Class	Data Augment: Conv. Autoencoder				Data Augment: Rotating			
	SV	GAP-SV	**VGG16**	ResNet50	SV	**GAP-SV**	**VGG16**	ResNet50
Center	0.98	0.97	**0.99**	0.96	0.99	**0.98**	**0.98**	0.99
Donut	1	0.67	**1**	0.67	1	**1**	**1**	1
Edge-Loc	0.82	0.84	**0.85**	0.78	0.85	**0.87**	**0.86**	0.85
Edge-Ring	0.83	0.74	**0.92**	0.91	0.85	**0.92**	**1**	0.83
Local	0.72	0.76	**0.74**	0.61	0.78	**0.8**	**0.78**	0.78
Near-full	0.79	0.87	**0.86**	0.87	0.92	**0.96**	**1**	1
Random	0.78	0.91	**0.86**	0.85	0.93	**0.92**	**0.83**	0.93
Scratch	0.46	0.76	**0.51**	0.29	0.68	**0.78**	**0.71**	0.64
none	0.99	0.99	**0.99**	0.98	0.99	**0.99**	**0.99**	0.99
Macro avg	0.82	0.83	**0.86**	0.77	0.89	**0.91**	**0.91**	0.89
Train Acc.	0.99	0.99	**0.99**	0.99	0.99	**0.99**	**0.99**	0.99
Test Acc.	0.97	0.97	**0.97**	0.97	0.97	**0.98**	**0.97**	0.97

	precision	recall	**f1-score**	# Data
Center	0.99	0.98	**0.98**	442
Donut	1	1	**1**	1
Edge-Loc	0.86	0.85	**0.86**	158
Edge-Ring	1	1	**1**	6
Loc	0.77	0.78	**0.78**	109
Near-full	1	1	**1**	11
Random	1	0.71	**0.83**	35
Scratch	0.89	0.59	**0.71**	27
none	0.98	0.99	**0.99**	2977
macro avg	0.94	0.88	**0.91**	
weighted avg	0.97	0.97	**0.97**	

[Precision, recall and f1-score for the VGG16 with rotating]

- Scratch and Loc is hard to predict
- **Scratch's F1 score is lower than other class's for all cases**
- The CAM image shows clues for this
  * **Scratch** looks similar with 'Near-full' and 'none'
  * **Loc** looks similar with 'Edge-Ring'
- CAM images looks similar with each classes map pattern.

CONCLUSIONS

- ❑ **Training accuracy: 0.99, Test Accuracy: 0.98 is the best (GAP-SV)**
- ❑ **VGG-16** is the best for both augmented dataset (f1 score:0.91)
- ❑ **GAP-SV** is best with rotating data (training time is half of VGG-16)
- ❑ **Rotating** is better augment method for wafer map data
- ❑ **Future works** : Improve 'Scratch' and 'Loc' map's prediction and How to apply this to the production?

[Reference]

[1] https://www.kaggle.com/qingyi/wm811k-wafer-map
[2] Kiryong Kyeong and Heeyoung Kim, IEEE TRANSACTIONS ON SEMICONDUCTOR MANUFACTURING, VOL. 31, NO. 3, AUGUST 2018
[3] https://towardsdatascience.com/aligning-hand-written-digits-with-convolutional-autoencoders-99128b83af8b
[4] Rajpurkar & Irvin et al., PLOS Medicine, 2018
[5] https://github.com/JeongSeo74/DeepLearning-CS230 **(Source Code)**