Surface Defect Image Classification with Convolutional Neural Network

Po-Hsiang Wang

pohwa065@stanford.edu

1. Motivation and prior approaches

Surface defects on semiconductor wafers have various morphologies. Among these defects some of them are "killer defects" that have huge impact on the final yield of the device. Convolutional neural network (CNN) and CNN-based transfer learning have been used for defect classification in materials manufacturing [1,2,3]. More complex architectures such as hybrid CNN-LSTM [4,5] model is also tested. Here, we proposed to identify these killer defects during the inspection at early stage using CNN with the help of both generative adversarial network (GAN) for data augmentation and autoencoder (AE) for image noise reduction (Fig. 1b).

2. Model description

2.1 CNN model for classification

CNN model: CONV2D →Relu →MAXPOOL →CONV2D →Relu →MAXPOOL → Flatten → fully connected layer with softmax cross entropy loss and L2 regularization

2.2 GAN model for data augmentation

GAN model: Auxiliary Classifier GAN (AC-GAN) [6] was used. It is similar to conditional GAN except the discriminator perform classification in addition to discriminating real and synthetic image (Fig.1c). Here, we are not enabling the classification capability of this model, i.e. only use this model for generating one class of image at a time. The loss function is binary cross entropy for both generator and discriminator ($J^{(D)}$ = -$J^{(G)}$). Architecture of the model is shown in Fig. 2a-b. The generator transforms random noise input into 128x128x1 images, while discriminator generates feature vectors from input images for binary classification.

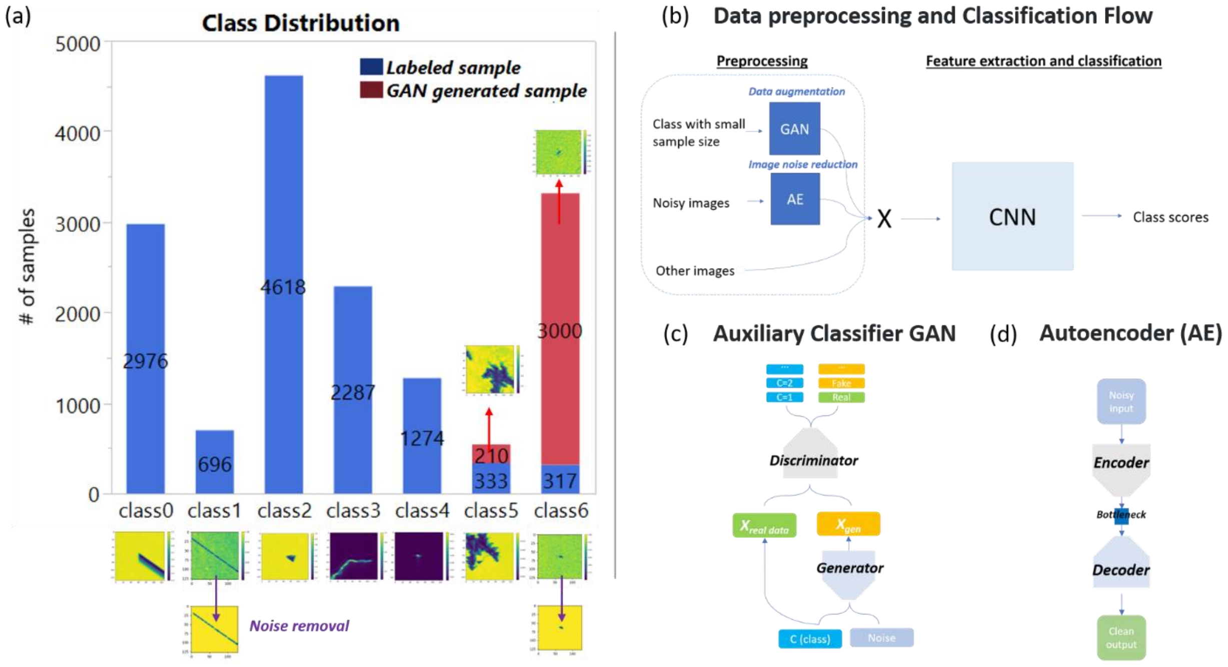


Fig. 1 (a) Sample class distribution. C5 and C6 have GAN generated synthetic images added; noise are removed for C1 and C6 (b) Proposed data preprocessing flow for training a CNN model (c) AC-GAN model (d) Autoencoder (AE) model

2.3 AE model for image denoise

Stacked autoencoder model [7] consists of an encoder and a decoder (Fig. 1d). Number of filters per layer first decreases with each subsequent layer in the encoder, and increases back in the decoder (Fig. 2c). The decoder is symmetric to the encoder in terms of layer structure. Finally, sigmoid activation function converts output to 0-1.

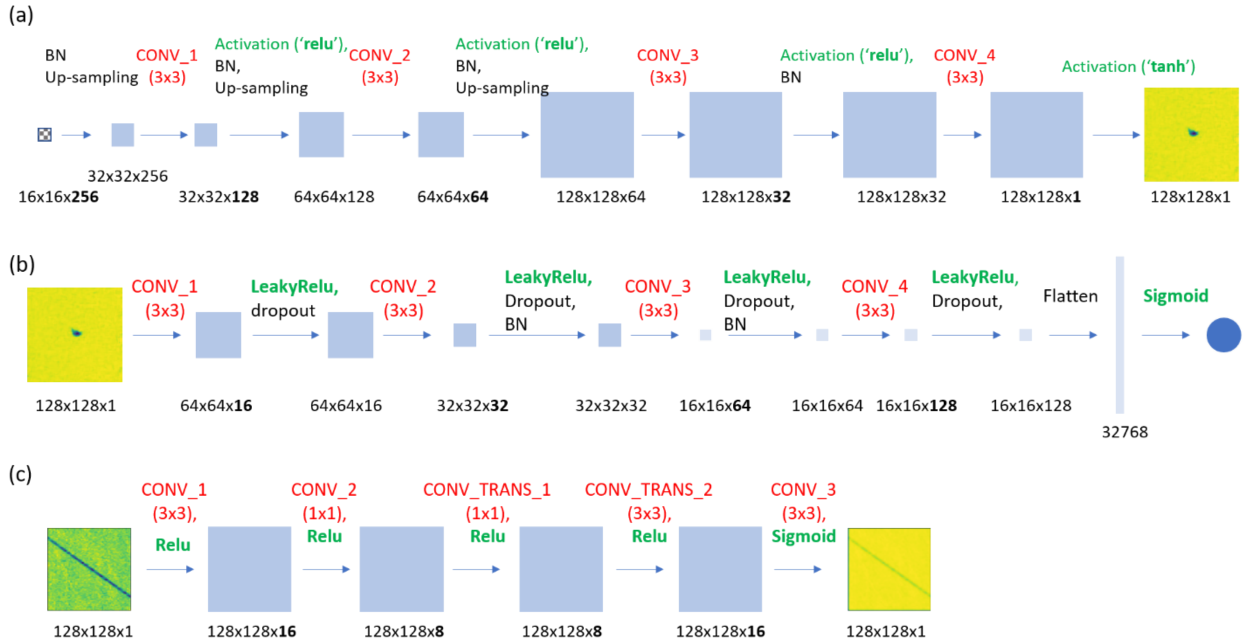


Fig.2 (a) Generator consists of several convolution layers (CONV) follow by activation, batch normalization (BN) and up-sampling. (b) Discriminator also contains series of CONV layers follow by activation and dropout. At the end, it has a fully-connected layer with sigmoid function for binary classification. (c) Stacked autoencoder with convolution layers (encoder) and transposed convolution layers (decoder).

3. Dataset

About 12k of images (resize to 128x128x1) are labeled. Most of them have well-defined defect shapes (Fig.1a). Apart from labeled data, 210 and 3000 synthetic images of C5 and C6 were added to the dataset respectively. Also, in some of the experiments, C1 and C6 are replaced with denoised version of the images.

4. Result

4.1 GAN generated synthetic images

The images on the left in Fig. 3 are few examples of original C6 samples in the dataset, these images are taken as an input to the discriminator together with generated images from the generator. After about 10000 iterations, the final synthetic images looks similar to the real images (Fig. 3). Here we trained the generator with different image sizes, namely 28x28x1, 64x64x1 and 128x128x1. The motivation is that the smaller the image size, the fewer parameters in the model to be trained and thus, faster training. After training, all the images are resized to 128x128x1 (same as original images in the dataset). Note that the 28x28 images has much lower resolution and even resized it to 128x128, it still looks blur compared with the 64x64 and 128x128 images. The 128x128x1 images are quite successful in the sense that it reproduces not only the defect shape (small, comet-like with short tail) but also the noisy background.

Odena et al. [6] claimed that generating higher resolution images (by GAN) improves discriminability. To test this hypothesis, these generative images are fed to a pre-trained CNN network (our baseline network), refer to e3,e4 and e5 in Table1-3. The 128x128 images resized from 28x28 synthetic images has 0% accuracy (all of them are predicted as another class) while the 64x64 and 128x128 synthetic images achieved ~80% accuracy suggesting high degree of similarity to the original images.

The effect of kernel size to the GAN generated images is also evaluated (Fig. 4). The 3x3 filter reproduces the zigzag edges in the original images perfectly while a larger kernel size, 9x9 and 6x6 in the early stage of the model tend to make the boundary smoother. When doubling the number of filters in each convolution layer, the model failed to reduce loss to generate valid images.

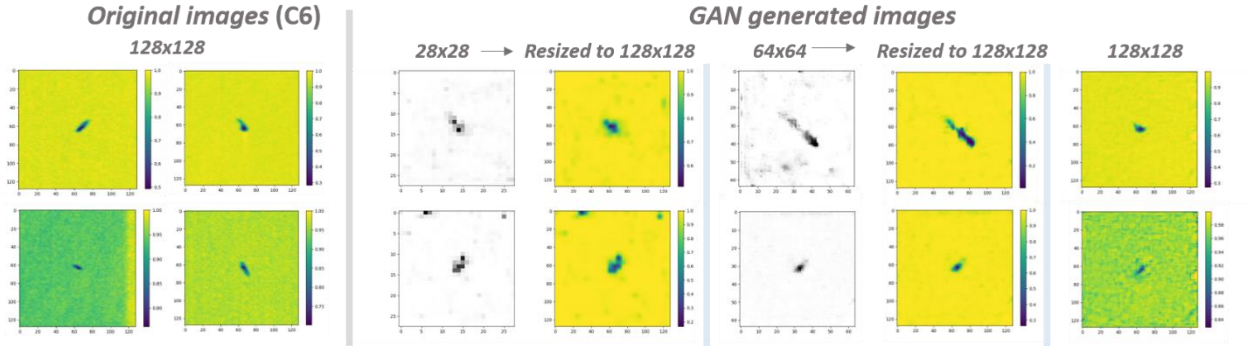


Fig.3 Original and synthetic images of C6. 28x28, 64x64 generated images are resized to 128x128.

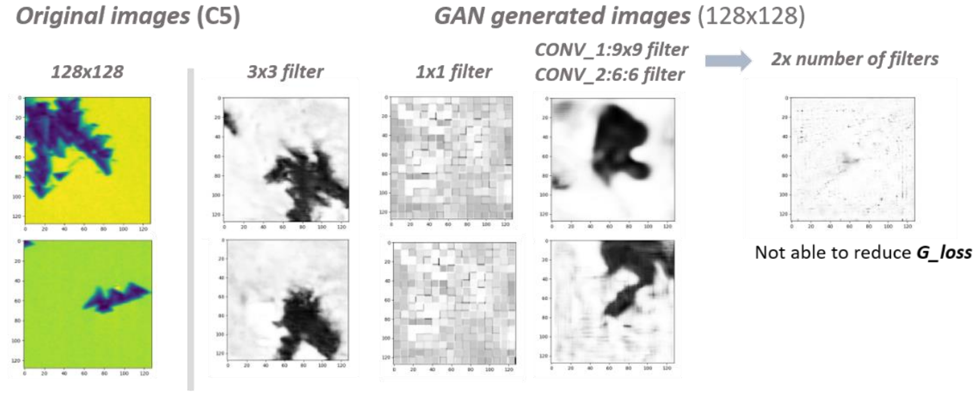


Fig.4 Original and synthetic images of C5. The effect of different kernel size in the convolution layer is shown.

4.2 AE denoised images

In this dataset, Some of the C1 in the training set have relatively weak signal and becomes discontinues. The classifier may not perform well when C6 is also having low signal-to-noise. The hypothesis is that if the model can learn the line feature from C1 with strong signal samples, it will be unlikely to classify C6 (a short comet-like shape) as C1. Here, we trained AE by fitting X (original image) to Y (adding random noise to original image), then applied this model on raw C1/C6 images. The outcome are LV0 images in Fig. 5. Since the noise seems to be removed leaving some low intensity features in LV0, the features are boosted to LV1 and LV2 by manipulating pixel intensities.

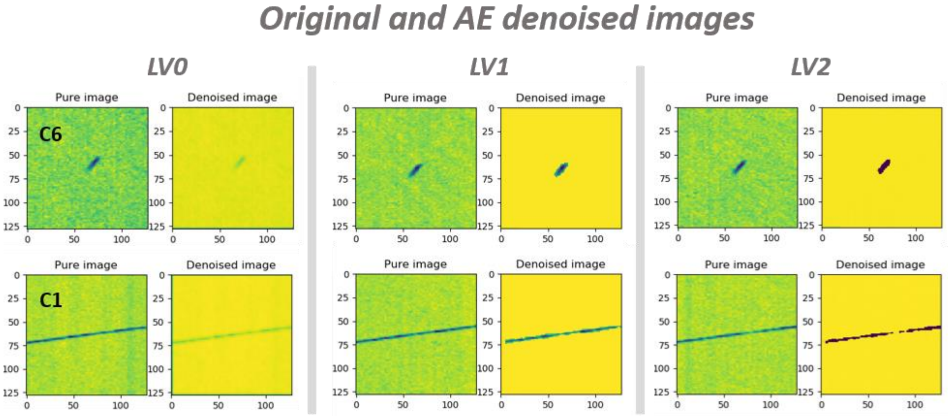


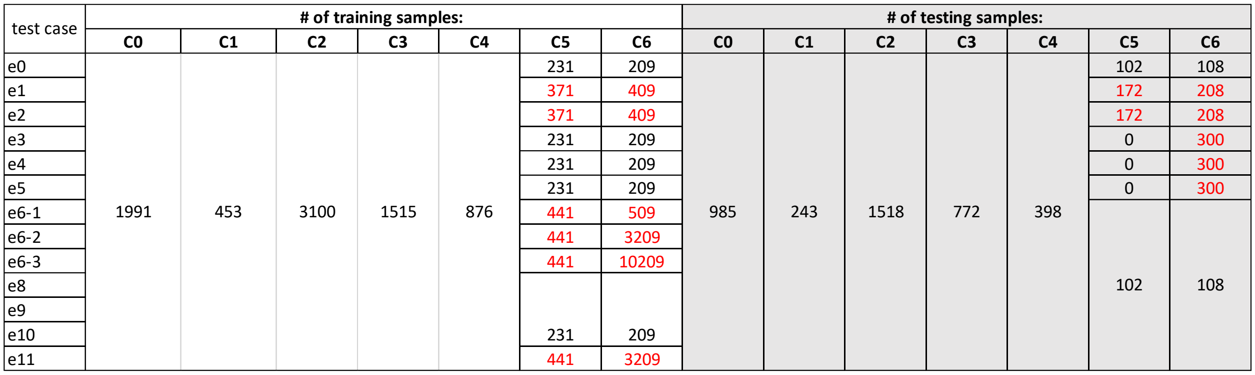
Fig.5 Original and AE denoised images. The intensity of the feature are boosted to different levels.

4.3 CNN Classifier performance with data augmentation and denoised images

The test conditions are summarized in Table 1. e0 is the baseline model (without any data augmentation). e1-2 have synthetic data in both training and test set. e3-5 use pre-trained baseline model to test on the GAN generated images. e6 uses both real data and synthetic data to train the model and applies this model to only the real data in test set.

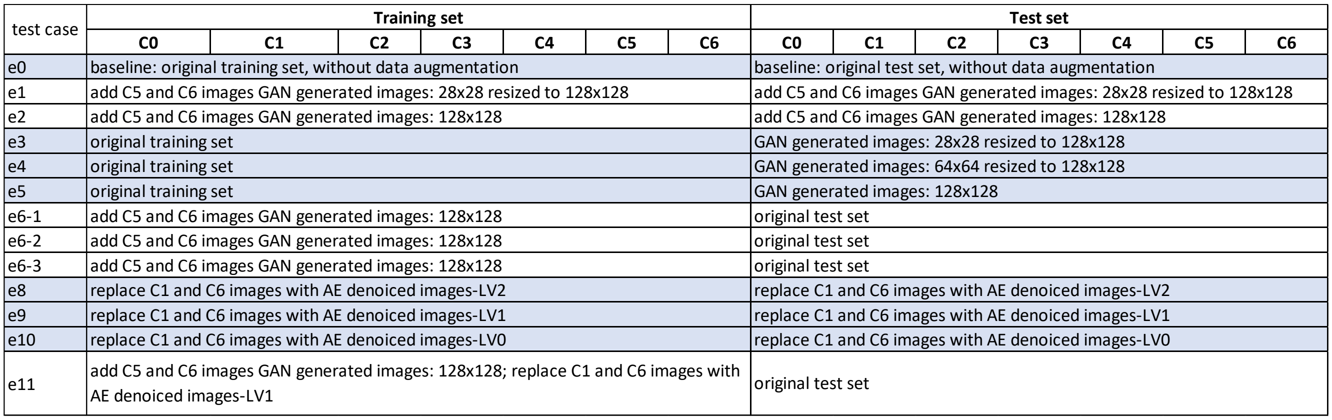
The baseline model performed quite well except for C5 and C6. Adding GAN synthetic samples in both training and testing set improves the accuracy of C5 and C6 (Table 3). The average training and test accuracy are close, indicating no overfitting. In the real use-case, e6-2, with data augmentation in training set, the model performance of C5 increases by 3% (90.2% to 93.1%) and C6 increases by 4.5% (76% to 80.5%). It is worth noting that the size of augmentation also effects the classifier performance: C6 accuracy is only 27% and 15% when adding too little (300) or too much (10000) synthetic C6 images.

Table1: Test conditions



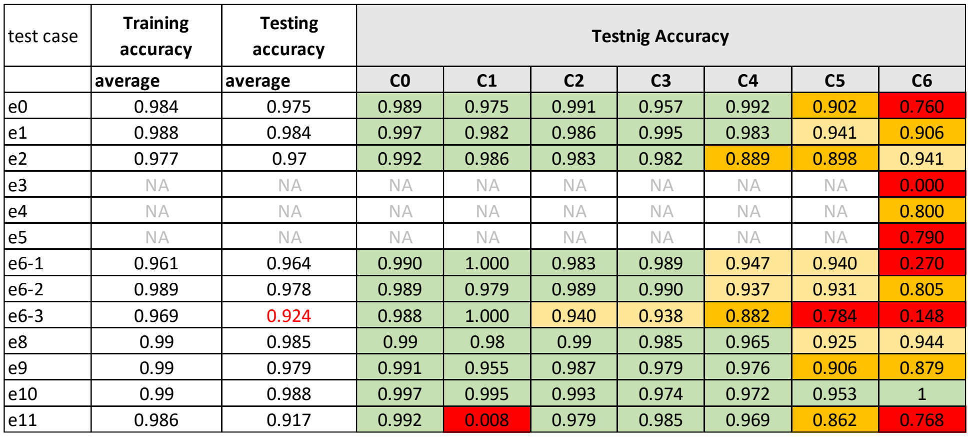
test case	# of training samples:							# of testing samples:						
	C0	C1	C2	C3	C4	C5	C6	C0	C1	C2	C3	C4	C5	C6
e0	1991	453	3100	1515	876	231	209	985	243	1518	772	398	102	108
e1						371	409						172	208
e2						371	409						172	208
e3						231	209						0	300
e4						231	209						0	300
e5						231	209						0	300
e6-1						441	509						102	108
e6-2						441	3209							
e6-3						441	10209							
e8						231	209							
e9														
e10														
e11						441	3209							

Table2: Sample distribution in training and testing set.



test case	Training set							Test set						
	C0	C1	C2	C3	C4	C5	C6	C0	C1	C2	C3	C4	C5	C6
e0	baseline: original training set, without data augmentation							baseline: original test set, without data augmentation						
e1	add C5 and C6 images GAN generated images: 28x28 resized to 128x128							add C5 and C6 images GAN generated images: 28x28 resized to 128x128						
e2	add C5 and C6 images GAN generated images: 128x128							add C5 and C6 images GAN generated images: 128x128						
e3	original training set							GAN generated images: 28x28 resized to 128x128						
e4	original training set							GAN generated images: 64x64 resized to 128x128						
e5	original training set							GAN generated images: 128x128						
e6-1	add C5 and C6 images GAN generated images: 128x128							original test set						
e6-2	add C5 and C6 images GAN generated images: 128x128							original test set						
e6-3	add C5 and C6 images GAN generated images: 128x128							original test set						
e8	replace C1 and C6 images with AE denoiced images-LV2							replace C1 and C6 images with AE denoiced images-LV2						
e9	replace C1 and C6 images with AE denoiced images-LV1							replace C1 and C6 images with AE denoiced images-LV1						
e10	replace C1 and C6 images with AE denoiced images-LV0							replace C1 and C6 images with AE denoiced images-LV0						
e11	add C5 and C6 images GAN generated images: 128x128; replace C1 and C6 images with AE denoiced images-LV1							original test set						

Table3: Accuracy of each Class for each testcase



test case	Training accuracy	Testing accuracy	Testnig Accuracy						
	average	average	C0	C1	C2	C3	C4	C5	C6
e0	0.984	0.975	0.989	0.975	0.991	0.957	0.992	0.902	0.760
e1	0.988	0.984	0.997	0.982	0.986	0.995	0.983	0.941	0.906
e2	0.977	0.97	0.992	0.986	0.983	0.982	0.889	0.898	0.941
e3	NA	NA	NA	NA	NA	NA	NA	NA	0.000
e4	NA	NA	NA	NA	NA	NA	NA	NA	0.800
e5	NA	NA	NA	NA	NA	NA	NA	NA	0.790
e6-1	0.961	0.964	0.990	1.000	0.983	0.989	0.947	0.940	0.270
e6-2	0.989	0.978	0.989	0.979	0.989	0.990	0.937	0.931	0.805
e6-3	0.969	0.924	0.988	1.000	0.940	0.938	0.882	0.784	0.148
e8	0.99	0.985	0.99	0.98	0.99	0.985	0.965	0.925	0.944
e9	0.99	0.979	0.991	0.955	0.987	0.979	0.976	0.906	0.879
e10	0.99	0.988	0.997	0.995	0.993	0.974	0.972	0.953	1
e11	0.986	0.917	0.992	0.008	0.979	0.985	0.969	0.862	0.768

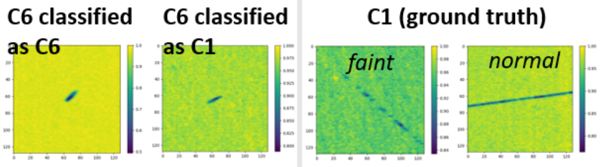
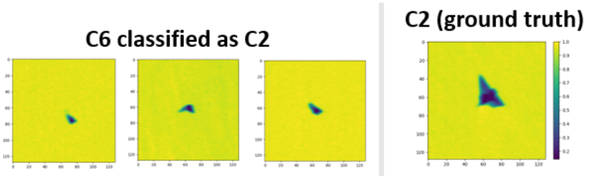
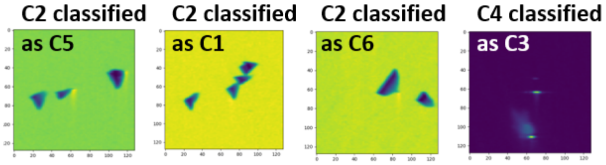
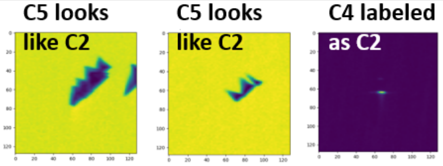
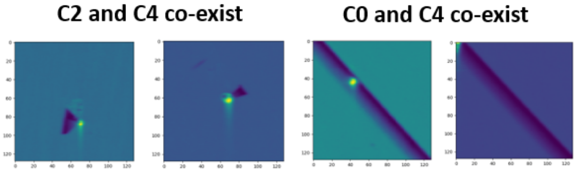
e8-e10 used both denoised C1/C6 images for both training and testing. The accuracy is higher than baseline model (e0) confirming our hypothesis that if the images in our samples all have high signal to noise ratio, i.e. can be clearly differentiated from the background, then the classification will not suffer from the noises in the images. However, if only the training set have denoised images, the classifier has poor performance on C1. It would be interesting to see what kind of features have been learned during each activation in this model to understand why it can not be applied to original dataset.

5. Error analysis and future work

The focus will be improving the C6 performance while maintaining the others. The source of error in the e6-2 classifier is summarized as below (Table 4). AE does not further improve the accuracy as we wished.

Also, it is worth trying a deeper CNN network (currently only use 2 convolution layers) and transfer learning if a deeper model trained on similar dataset is available. We have tried adding one more convolutional layer with more filters, but the result are quite similar to the baseline model.

Table4: Accuracy of each Class for each testcase

Misclassified images	Source of error	Future work
Class 6 classified as class 1  	Some of the **C1** in the training set have relatively weak signal and becomes discontinues. The classifier may not performed well when **C6** is also having low signal-to-noise	Tried Autoencoder (AE) for image noise reduction in the training set. However, in the test set the classifier only recognize strong **C1** which yield **low accuracy in C1 and no improvement in C6**
Class 6 classified as class 2  	These **C6** examples have triangular shapes which is the signature of a typical C2 example	This could be a limitation of this classifier if C2 and C6 looks exact the same except for the size. C6 is relatively small so another filter can be applied to recover C6 from the C2 bucket after classification
 	Multiple defects appear in one image	Define this as a object detection problem, label with bounding box information and try YOLO (not within the scope of this project)
 	Mislabeled and ambiguous images	Consider these as noise to the model, collect/label more samples to reduce the effect of such noise
 	Images seems to contain multiple classes. For example, a triangle (C2) with a bright dot (C4) or a bar (C0) and a bright dot (C4) can appear in a same image	Define it as multiple binary classification problem: Use multilabel with sigmoid cross entropy loss function

6. Conclusion

The synthetic images from GAN are similar to the original images, confirmed not only visually but also through a baseline classifier. With the help of these GAN generated images in the training set, the classification accuracy of the class having small labeled data improves by 4.5%. AE successfully removed the background noise in the images from the training set. However, the model failed to recognize key features of the raw images in the test set.

References

[1] Imoto, Kazunori, et al. "A CNN-based transfer learning method for defect classification in semiconductor manufacturing." *2018 International Symposium on Semiconductor Manufacturing* (ISSM). IEEE, 2018.

[2] Masci, Jonathan, et al. "Steel defect classification with max-pooling convolutional neural networks." *The 2012 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2012.

[3] Tao, Xian, et al. "Automatic metallic surface defect detection and recognition with convolutional neural networks." *Applied Sciences* 8.9 (2018): 1575.

[4] Liu, Tianyuan, et al. "A hybrid CNN–LSTM algorithm for online defect recognition of CO2 welding." *Sensors* 18.12 (2018): 4369.

[5] Zhao, Yudi, et al. "A visual long-short-term memory based integrated CNN model for fabric defect image classification." *Neurocomputing* 380 (2020): 259-270.

[6] Odena, Augustus, Christopher Olah, and Jonathon Shlens. "Conditional image synthesis with auxiliary classifier gans." *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017.

[7] Mayur Thakur, Sofia K. Pillai , “A Hybrid System Using CNN and AE for Noisy Image Classification,” *International Journal of Computer Sciences and Engineering*, Vol.7, Issue.4, pp.870-875, 2019.