Void defect will exist during the wafer packaging process.

Usually AA(Adhesive Attach) void is caused by moisture absorption to adhesive layer between substrate and die. if void area exceeds the limit, it will spread out in reflow oven and then cause package fail.

The current process is that we request operators to identify void region by naked eyes which will raise some problems:

1. Sonix and Sonoscan do not have auto image classification function
2. So workforce wasted, currently we need 3 operators to check around 100 images per shift
3. Naked eyes can easily identify void issue only when the affected area is large enough and happens continuously.

The objective of this project is to detect the void area by machine automatically, as a support method to help operators to detect void defects.

Here is the project roadmap, we have done the image collection, pro-process and image classification prototyping, and we are currently trying to use image segmentation technique to detect the void area. Then we will cooperate with PTI to setup server for real time image segmentation test, and finally we hope this system could implement in PTI production line successfully.

Our first approach for this project is image classification, we tried to use Convolutional neural networks IncrptionV3. The Inception network is a computational efficiency network, it uses filter concatenation to connect all filter outputs together depth-wise from a set of convolution layers and pooling layer, each of them will be followed with a 1x1 convolutional layer which are used as bottlenecks to reduce parameters. In this case, we cropped the original SAT image to get single die images, then we labelled them and put them into the Inception network for training. Finally we could use the trained model to predict whether the component has void.

For training and testing, we use 237 good samples and 233 bad samples for training, 62 good samples and 60 bad samples for testing. The final sensitivity and specificity is 95% and 82.3% respectively, and the processing efficiency is 6 sec per component.

The second approach of this project is segmentation. For image segmentation, we just annotate the void area of the image, and put them into Mask R-CNN for training, then the trained model could identify where is the void area and mask them out. The advantages of this approach is that it has better accuracy and we could calculate the void area which is more intuitive rather than just classify good and bad.

The Mask R-CNN architecture has 2 stages:

In stage1, firstly, the image will input to FPN (Feature Pyramid Network) which is a feature detector from low level to high level and then output the combined features to RPN (Region Proposal Network), RPN is a object detector, it will scan all FPN top-bottoms and proposes which may contain objects, then ROI Align will speed up the feature extraction of proposed region, and then output to stage 2.

In stage 2, there is another neural network takes proposed regions from the first stage and assign them to several specific areas in a feature map level, scans these areas, and then generates object classes, bounding box and mask.

So here is another view of Mask RCNN, you could see that the feature of the image will be extracted by CNN which is FPN, then the RPN will locate the proposed features in the feature map, and output them into stage 2 through ROI Align. Finally, in stage 2, the FC layers will classify the class and bounding box of the object, and Convolution layers for mask region prediction.

There are some test examples, the middle images are the original components CSAM image, the left images are annotated by engineers and the right images are annotated by MaskRCNN model. Although there are some mismatch between engineer and machine, the result could still think as great, since for the CSAM image, different engineers could annotate different areas, so the mask areas could not be totally same.

In this approach, we used 248 annotated images for training and validation, then we used 53 bad images and 67 good images for testing. So for the result, the sensitivity and specificity are 98% and 95% respectively, the processing efficiency is 0.42 sec / component on a RTX GRAPHICS CARD. So it is a big improvement compared with the image classification.