Using Deep Learning ADC for Defect Classification for Automatic Defect Inspection

Bryce Chi, Andy Chen, Jay Chen, Terry Voots, Maruko Wu, Cheolkyu Kim, Zhuan Liu
Onto Innovation
16 Jonspin Road
Wilmington, Massachusetts, 01887, U.S.A.
Ph: 978-253-6200
Email: bryce.chi@ontoinnovation.com andy.chen@ontoinnovation.com jay.chen@ontoinnovation.com
terry.voots@ontoinnovation.com maruko.wu@ontoinnovation.com cheolkyu.kim@ontoinnovation.com
zhuan.liu@ontoinnovation.com

Abstract

In traditional semiconductor packaging, manual defect review after automated optical inspection (AOI) is an arduous task for operators and engineers, involving review of both good and bad die. It is hard to avoid human errors when reviewing millions of defect images every day, and as a result, underkill or overkill of die can occur. Automatic defect classification (ADC) can reduce the number of defect images that need to be reviewed by operators. The ADC process can also be integrated with AOI engines to reduce nuisance defect images to reduce AOI image capturing time.

This paper will focus on how to utilize Onto Innovation's TrueADC® software product to build ADC classifiers using a multi-engine (ME) solution. The software supports CNN, DNN and KNN algorithms. The use of CNN and DNN are currently mainstream in the development of deep learning (DL) for ADC classification in the semiconductor industry. We will address how to improve classification by using multiple models in the classification process with unique algorithms. As a result, the user can achieve industry requirements with very demanding specifications, like high accuracy, high purity, and high classification rate with very low escape rates.

Key words

ADC, TADC, AOI, DMS, Underkill, Overkill, Accept, Reject, CNN, Deep Learning, DNN, KNN, Manual Review, Optical Inspection, Transfer Learning, Nuisance defect, 3rd party tool, AI, Artificial Intelligence

Introduction

Automated optical inspection (AOI) is an established process for yield management and process control in semiconductor assembly & testing (SAT) facilities. One reality of the inspection process is that millions of defect images are likely generated where 100% of them will be reviewed offline by an operator. Manual review is very time consuming and human eye fatigue is a risk that can directly impact review quality because of judgement error. In Industry 4.0, an ADC solution becomes an important part of automation because it can reduce operator loading and it is expected to reduce the cost of HVM manufacturing. Deep learning is a branch of machine learning which is completely based on artificial neural networks [1]. As neural networks mimic the human brain, so deep learning is also a kind of mimic of the human brain. In deep learning, we don't need to explicitly program everything. [2]

The concept of deep learning is not brand new. The idea has been around for many years, but the main limitations have been processing power and limited data. In the last 20 years, because available processing power has increased exponentially, deep learning algorithms and machine learning algorithms are now more practical. To put things in perspective, deep learning is a subdomain of machine learning. With accelerated computational power of large data sets, deep learning algorithms can self-learn hidden patterns within data to make predictions. [3]

In essence, you can say that deep learning is a type of machine learning. They both are a type of artificial intelligence (AI) that are trained in with large amounts of data and deal with many computational units working in tandem to perform predictions. While CNN applications use relatively fewer pre-processing actions compared to other image classification algorithms, Onto Innovation has combined multiple techniques. The network learns to optimize the Or kernels through automated learning, whereas in traditional algorithms these filters are hand-engineered. [4]

Independence from prior knowledge and human intervention in feature extraction is a major advantage. Convolutional layers convolve the input and pass its result to the next layer. This is like the response of a neuron in the visual cortex to a specific stimulus. Each convolutional neuron processes data only for its receptive field.[5]

This paper will introduce how we use TrueADC software integrated with Onto Innovation's AOI tools to build up a robust ADC library. Through the combination of multiple classification engines (ME) like KNN+CNN models, the classifier will have increased accuracy and will reduce the escape rate. For integrated in-line ADC, we call it "Tool Centric" ADC.

We also offer a solution that extends the ADC software to support third party AOI machines, making available Onto Innovation's optimized CNN models. This enables the industry to implement one ADC platform for multiple AOI in HVM manufacturing. Therefore, Onto Innovation's TrueADC can support both in-line and off-line applications (Fig 1).

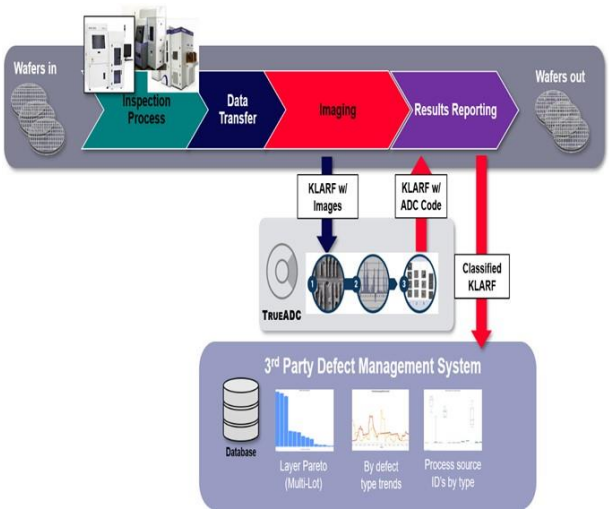


Figure 1. In-line & Off-line data flow diagram

The main purpose of ADC is to classify defect codes automatically. Some defect images like scratches and pad defects are shown below to provide more understanding about what we are trying to solve in this paper (Fig 2).

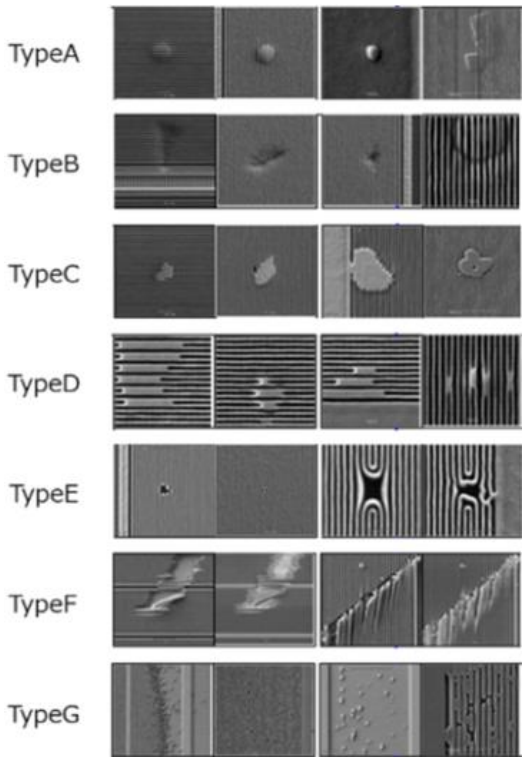
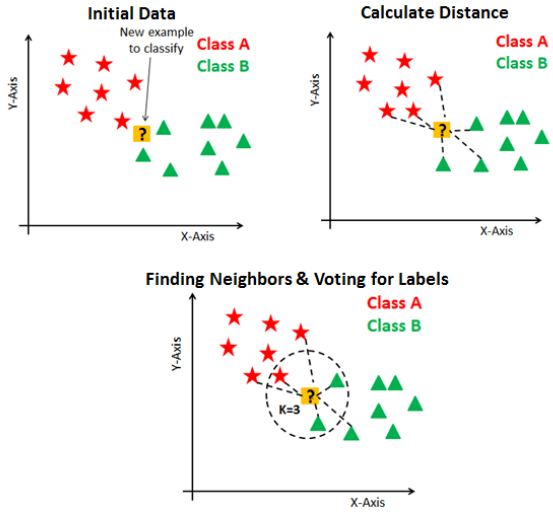


Figure 2. Typical defect types

ADC Methods

KNN (also known as k-NN, the **k**-**n**earest **n**eighbors algorithm) is an established and known machine learning algorithm based on a supervised learning technique. It is used in many different areas, including handwriting detection, image recognition, and video recognition. It is based on the local minimum of the target function which is used to learn an unknown function of desired precision and accuracy. The algorithm also finds the neighborhood of an unknown input, its range or distance from it, and other parameters (Fig 3).



Nearest-neighbor algorithm

a) A pseudo code for the nearest neighbor algorithm is

ALGORITHM *Nearest-neighbor*$(D[1...n,1...n],s)$
//Input: A $n\times n$ distance matrix $D[1...n,1...n]$ and an index s of the starting city.
//Output: A list Path of the vertices containing the tour is obtained.
for $i \leftarrow 1$ to n do Visited $[i] \leftarrow$ false
Initialize the list Path with s
Visited $[s] \leftarrow$ true
Current $\leftarrow s$
for $i \leftarrow 2$ to n do
    Find the lowest element in row current and unmarked column j containing the element.
    Current $\leftarrow j$
    Visited$[j] \leftarrow$ true
    Add j to the end of list Path
Add s to the end of list Path
return Path

Figure 3. KNN algorithm

CNN (**C**onvolutional **N**eural **N**etwork) is a neural network that has one or more convolutional layers based on deep learning techniques and it can be used for image classification, segmentation, object detection and for other auto correlated data (Fig 4).

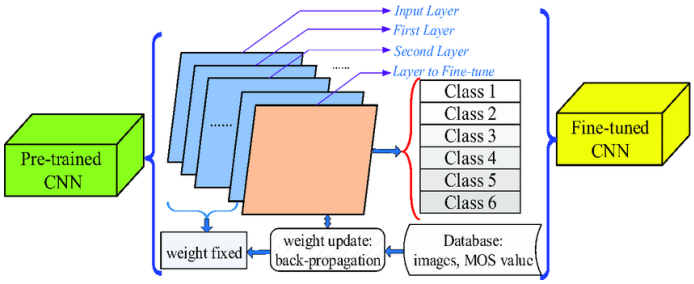


Figure 4. Six label classification of CNN model

An image classification by the CNN model involves the extraction of features from each image to observe detailed patterns in the dataset (Fig 5).

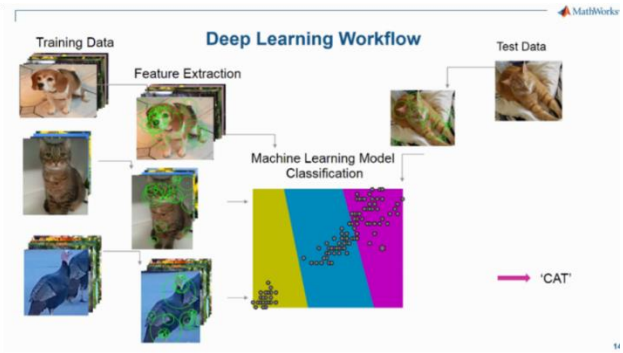


Figure 5. Deep learning feature extraction diagram

Experimental results

The result by traditional KNN model

In this section, the only classifier we will use is a KNN model.

Performance of the first engine's classification is shown in the matrix of Table 1. The green defect, class code 2051, is identified as non-killer (or acceptable, nuisance) defect, the black class code 254 is defined as unknown defect in our classification engine, and the red class codes are all killer defects (reject).

ManualVADC	198	201	202	203	209	210	213	2051	254	Total	Classified	Accuracy
198	33	5		41				6	4	89	95.5%	38.8%
201		83						2	1	86	98.8%	97.6%
202	8		18	1					2	29	93.1%	66.7%
203	3	2	1	149				3	18	176	89.8%	94.3%
209					14					14	100%	100%
210		1		2		12		1	4	20	80%	75%
213				1		3				4	100%	0%
2051								43		43	100%	100%
Total	44	91	19	194	14	15	0	55	29	461	93.7%	81.5%
Purity	75%	91.2%	94.7%	76.8%	100%	80%	N/A	78.2%	0%			

Table 1. The performance matrix result of KNN model

The KNN classification results show that the purity of defect class code 190, 203 and 2051 are all around 75%. It tells that there might be opportunity to gain more confidence for this model on classifying those three defect class codes. KNN is driven by image matching and performance is tightly coupled to images that were used in the supervised sorting of the library. This can be observed in testing the library's performance against itself and against an independent image set. While KNN uses a smaller set of defect features found through defect isolation, the limited number can often demonstrate overlap between classes.

We investigated those misclassified defect images and found the result of many misclassified defect images were inferior quality or related to defect isolation. So, we explored additional classification engines, the CNN\DNN models on the same image set.

The result by CNN model only

In this section, we used only a CNN model as the classifier. The images used by the test KNN library are also used for training the CNN model. TrueADC software supports self-defined models, transfer learning models, and imported models. While our test covered multi-model recipes, we selected one for this discussion that demonstrated great results for the target image set. Our chosen example of the CNN model using transfer learning had the following parameters:

Pre-trained model: Resnet50; Batch size:32: Learning rate: 0.001, Optimizer: Adamax

Figure 6 shows the result of the CNN model training and evaluation. Although it looked great, the verification loss was not ideal nor going down smoothly. A reasonable conclusion would be we had under populated defect classes, so defect definitions were overlapping and possibly the images selected. The resulting performance matrix is shown in Table 2.

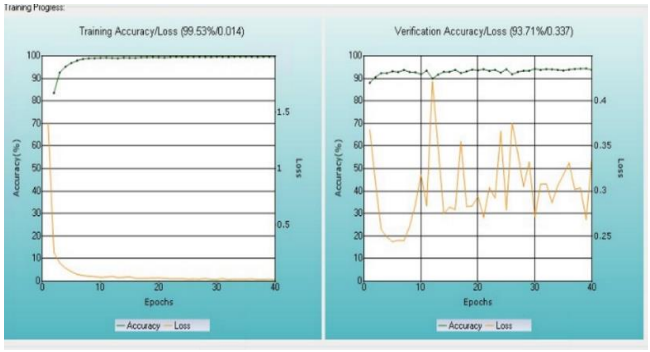


Figure 6. CNN model training Accuracy/Loss and Verification Accuracy/Loss.

ManualVADC	1	198	201	202	203	204	205	209	210	213	2051	2061	254	Total	Classified	Accuracy
198		45												45	100%	100%
201			250		1								7	258	97.3%	99.6%
202				50										50	100%	100%
203	1	4	5	2	1256				5	1	10	1	47	1332	96.5%	97.7%
204					1	44								45	100%	97.8%
205							519						1	520	99.8%	100%
209								17						17	100%	100%
210					4				152			1	5	162	96.9%	96.8%
213										383				383	100%	100%
2051	3				17						1523	1	20	1564	98.7%	98.6%
2061					2							1129	14	1144	98.8%	99.8%
Total	54	49	255	52	1281	44	519	17	157	384	1533	1131	94	5570	98.3%	98.9%
Purity	92.6%	91.8%	98%	96.2%	98%	100%	100%	100%	96.8%	99.7%	99.3%	99.7%	0%			

Table 2. The performance matrix result of CNN only

The result of combining KNN and CNN models

After the CNN model had classified most of the defects, we observed several defect escapes happening at defect code 203. Referring to Table 2, after checking a few of the escaped cases, we realized that most of the cases have a noise pattern. Images containing noise patterns can confuse a trained model and can induce misclassified defects (Fig 7).

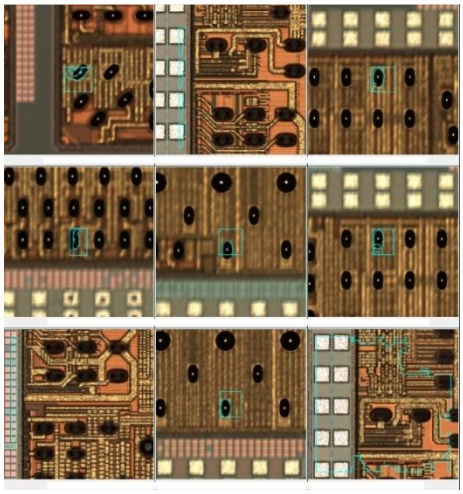


Figure 7. Cases of images with noise pattern

So, combining two models, which are KNN and CNN, becomes an approach. The second model will be set as double check rule in case one of the classes was classified incorrectly (Table 3).

Performance Matrix ▾ Report@9/20/2022 10:39:49 AM(User: Final Code, Group By: Individual)

ManualVADC	198	201	202	203	207	209	210	213	2051	2061	2063	254	Total	Classified	Accuracy
198	154												154	100%	100%
201		27											27	100%	100%
202			149									1	150	99.3%	100%
203	1			349				9				5	364	98.6%	97.2%
207					99		1						100	100%	99%
209						57							57	100%	100%
210							177						177	100%	100%
213			24		1			255				6	286	97.9%	91.1%
2051			1	2				2	1198			8	1211	99.3%	99.6%
2061										1301		1	1302	99.9%	100%
2063						1					885	1	887	99.9%	99.9%
Total	155	27	174	351	100	58	178	257	1207	1301	885	22	4715	99.5%	99.1%
Purity	99.4%	100%	85.6%	99.4%	99%	98.3%	99.4%	99.2%	99.3%	100%	100%	0%			

Table 3. The performance matrix result of KNN+CNN

Either to use traditional KNN model or CNN model to classify one result with one defect image, we can build a logic condition in TrueADC software to combine the two models. During defect classification, it could use different models between defect codes.

Offline CNN for third party of AOI

There are direct benefits of an inline ADC solution. One example is the tool receives real time defect class codes. At that point the AOI tool can provide accurate defect classification and completed die disposition. Now the AOI tool can forward completed die binning results along with inspection results to DMS (Defect Management) or YMS (Yield Management) for each wafer. Another in-line ADC process advantage is the AOI tool can be integrated to acquire defect information directly from the AOI tools to get more information on binning strategy development. In addition to in-line ADC, TrueADC software provides support for off-line ADC which is a post inspection process. Today, most commercial and in-house ADC solutions deploy a deep learning method for ADC. The support for runtime third party inspection tools is shown in the flow chart in Figure 8.

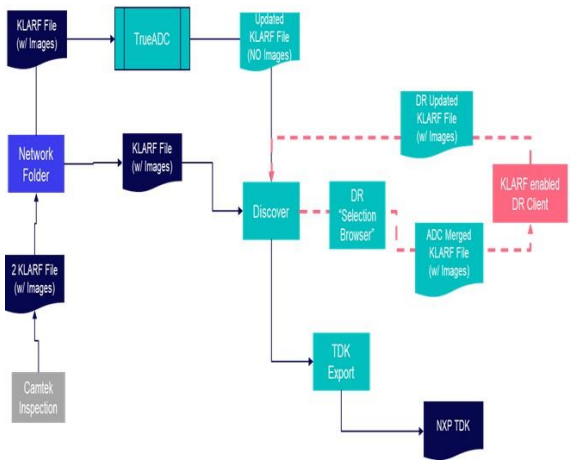


Figure 8. Offline ADC data flow diagram.

TrueADC software can also support off-line ADC on many types of optical imaging tools. The software has three ways to acquire images from third party AOI tools, including KALRF based, file based, and from a DMS database. After the software finishes defect classification, all binning results will be sent to a defect review system and then an operator could review the non-classified defects.

Currently, most AOI vendors could only provide off-line ADC because access to the AOI system is invalid or forbidden. So their ADC modeling will be based on CNN modeling only.

In the following experiment, our final selection was to choose Resnet50 as transfer learning (TF) model and other hyperparameters were set to software default values. The result is shown in Figure 9. It shows that the trend of Training Acc/Loss looks great, but it is not smooth enough on Training Loss. It seemed that the learning curve needs to be improved so we changed learning rate (LR) from 0.01 to 0.001 for better learning precision. The training results are displayed in Figure 10.

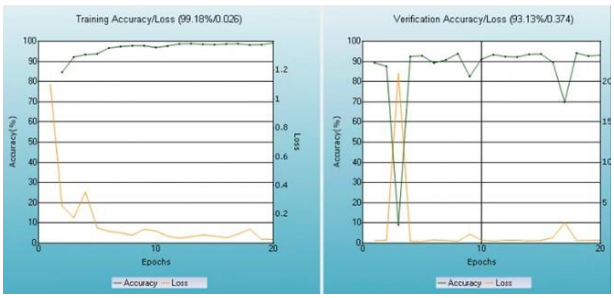


Figure 9. Training result of Resnet50 as TF

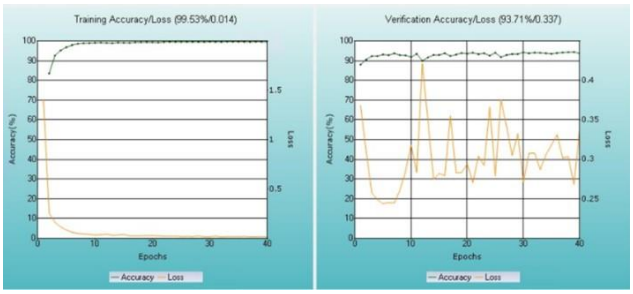


Figure 10. Training result of LR=0.001

Although the training trend looked great on Figure 10, we still noticed that verification loss did not have good convergence during training. This is indicative of what is called overfitting and is common with small data sets. A low loss function is a great indicator of confidence in final accuracy. After we changed the optimizer from Adam to Adamax, it turned out that the overall verification loss was lower, which we can conclude with less overfitting (Fig 11).

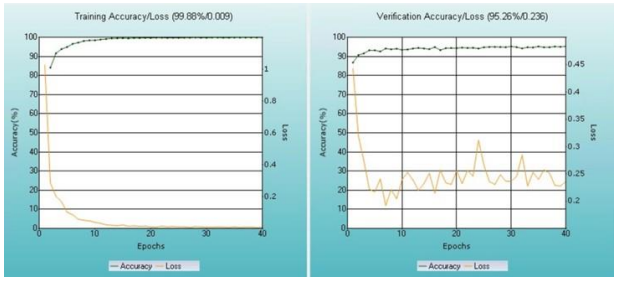


Figure 11. Training result of Adamax as optimizer

Conclusion

An ME ADC process is possible to meet demanding classification performance. It provides choices for classification engines like CNN, DNN, KNN, and software tools to optimize each model allowing a runtime solution to meet customers' specs such as A/P/C rate which was demonstrated to be greater than 90% of A/P/C.

By combining multiple AI engines in the image classification process, we demonstrated superior performance in accuracy and purity, thus there is a reduced risk of die escapes caused by defect image miss-classification. On major binning codes, we have achieved: 1. To have A/P/C rate about 99%, 2. To reduce escape rate about 0.2%, and 3. To reduce overkill rates to about 0.2%.

There are two types of this ADC software to consider. One is inline ADC, which is planned to cover two major models, KNN and CNN, because it can integrate with Onto Innovation's own AOI tools to retrieve reference images for the functional dataset requirement of the KNN model.

Another is called offline ADC which provides full deep learning solution for non-Onto AOI tools. Although there will be lack of reference images, models can learn classification methods by defect images only.

Comparing Table 2 and Table 3, it is easy to conclude that CNN model only, as Table-2, already has quite confidence on A/P/C result and its classification capability is already better than most of non-AI models. Though on minor defect parts, there would be possibility of escaped defects. Since those escaped defects usually have big chances to be misclassified as other class codes, it will be hard to trace and improve for further works in the same condition. In that case, Inline ADC can provide more reliable and convincible solution for auto classification. In the result of Table 3, ADC could reduce the possibility of misclassification on single defect code classification because it introduces KNN model that can prevent and correct misclassification when the CNN model got something wrong.

Acknowledgment

We thank Amy Shay and Joe Fillion for suggesting this paper and help.

References

[1] *LeCun, Yann; Bengio, Yoshua; Hinton, Geoffrey (2015). "Deep Learning". Nature.* ***521*** *(7553): 436–444*

[2] *"Convolutional Neural Networks (LeNet) – DeepLearning 0.1 documentation"*. DeepLearning 0.1. LISA Lab. *Retrieved 31 August 2013.*

[3] *"An introduction to deep learning". IBM Articles. Piyush Madan, Samaya Madhavan. March 2, 2020*

[4] *"Adam: A Method for Stochastic Optimization" Diederik P. Kingma, Jimmy Ba (Dec, 2014)*

[5] *"A CNN-Based Transfer Learning Method for Defect Classification in Semiconductor Manufacturing" Kazunori Imoto Nov,2019*