DEEP PCB TO COCO CONVERTOR

Genti Sreeja, Research Scholar, GITAM University, Hyderabad, India, sgenti@gitam.in

Dr K S Sudeep, Associate Professor, Department of CSE, GITAM University Hyderabad, India. skadavil@gitam.edu

Allena Venkata Sai Abhishek, Research Scholar, GITAM University, Visakhapatnam, India abhishek.avs123@gmail.com

Abstract- Millions of datasets and many models use the input datasets in COCO format. In this paper, we are converting the Deep PCB dataset to COCO format. The Deep PCB is a manufacturing defect data set. It has 1500 image pairs. Each has a template image & a test image. The template image has no defects & corresponding test image that has some defects with the annotations in a text file. The MS COCO (Microsoft Common Objects in Context) dataset is large-scale object detection, segmentation, key-point detection, and captioning. It is widely used for various models. We are trying to convert the deep PCB Manufacturing defect into COCO Format.

Keywords — Deep PCB, MS COCO, JSON, Object detection, key-point detection, Captioning dataset, Image segmentation, Image Annotation, XML

I. INTRODUCTION

Millions of corporate machine learning pipelines have neural network models that use the COCO format [1] [7]. MS COCO dataset consists of 328,000 images. Many corporates are using the dataset format for their machine learning pipelines. The structure can be circulated with its metadata for flexible transfer of knowledge about the dataset among various branches of the companies. Many datasets are converted into COCO format for easy and systematic data transfer. The directories and annotation [19] [20] [21] [22] text files are contained in the Deep PCB [4] dataset. Annotation files are in XML [13] [14] [15] [16] [17], excel or CSV too.

II. BACKGROUND WORK

The Deep PCB dataset contains fifteen hundred pairs of images. Each pair consists of a template image free of defects and a corresponding image that is tested with annotations, including positions of the six most widely used types of defects in PCB - mouse bite, open, short, pinhole, spur, and spurious copper.

We use a bounding box aligned along the axis with an ID for each class [8] of defect in the tested images. The dataset's images have been accumulated using a CCD linear scan with a resolution per millimeter is approximately 48 pixels. The free of defect template images are checked and cleaned manually from the sampled images.

The template and tested image's original size is around 16000 x 16000 pixels. These images are cropped into sub-images in 640 x 640 size. They are aligned with the template using matching methods. Illumination disturbance is avoided by employing binarization, for which a threshold is carefully selected. Various pre-processing algorithms [3] [6] [10] [12] [18] can be used for specific defect detection algorithms [2] for PCB.

For high-accuracy localization and classification of defects for PCB, the image thresholding and registration techniques are widely used. The deep PCB dataset's example pair is shown in Fig.1 and Fig. 2.

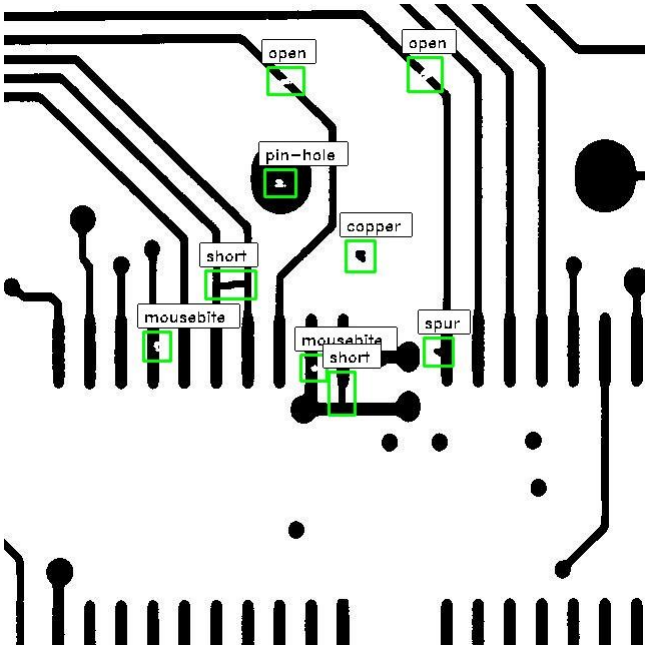

Figure 1. Tested Image Sample

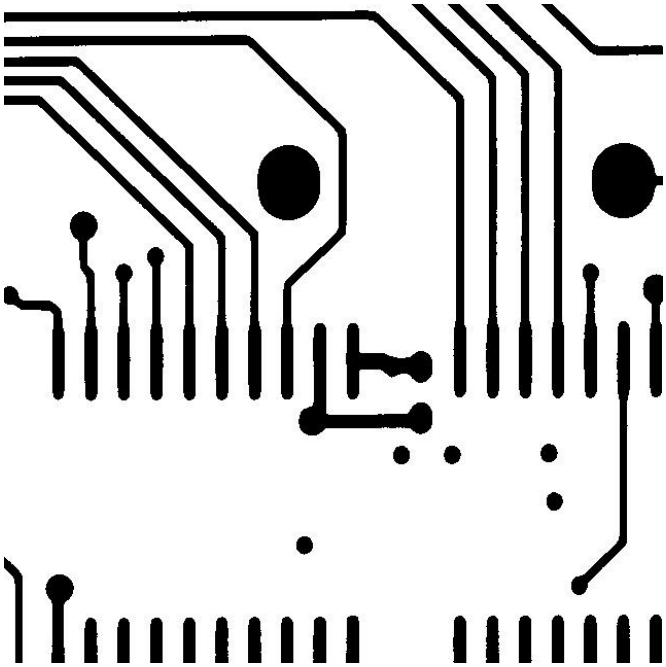

Figure 2. Corresponding Image for the tested Sample

As shown in Fig.1, six frequent types of defected in PCB are annotated, namely –

1. Open
2. Short
3. Mouse Bite
4. Spur
5. Pinhole
6. Spurious Copper.

As the tested image have only a less number of defects, that leads to nearly three to twelve defects in each image of size 640 x 640. It can be arguable that there can be some more artificial defects on each image that is tested according to the defect patterns of PCB6

The number of PCB defects is shown in Fig. 1 & Fig. 2. We separate a thousand images into the training set, and the remaining images are test sets. Each annotated image has an annotation text file with its filename, the template image, the tested image, and the corresponding annotation file.

The MS COCO (Microsoft Common Objects in Context) dataset is an enormous scope dataset that has several annotation types: object recognition, keypoint recognition, stuff segmentation, panoptic segmentation, densepose, and image subtitling. The dataset consists of 328,000 images.

The annotations are stored using JSON [5] [9] [11]. The key-value pair is contained in the JSON object. Strings are included in keys, and JSON types are the values. A colon separates the Keys and values. Each entry of the JSON is separated with a comma.

```
{
   "info"              : info,
   "images"            : [image],
   "annotations"       : [annotation],
   "licenses"          : [license],
}

info{
   "year"              : int,
   "version"           : str,
   "description"       : str,
   "contributor"       : str,
   "url"               : str,
   "date_created"      : datetime,
}

image{
   "id"                : int,
   "width"             : int,
   "height"            : int,
   "file_name"         : str,
   "license"           : int,
   "flickr_url"        : str,
   "coco_url"          : str,
   "date_captured"     : datetime,
}

license{
   "id"                : int,
   "name"              : str,
   "url"               : str,
}
```

Figure 3. COCO Format

```
annotation{
   "id"                : int,
   "image_id"          : int,
   "category_id"       : int,
   "segmentation"      : RLE or [polygon],
   "area"              : float,
   "bbox"              : [x,y,width,height],
   "iscrowd"           : 0 or 1,
}

categories[{
   "id"                : int,
   "name"              : str,
   "supercategory"     : str,
}]
```

Figure 4. COCO Format of Object detection

Each object annotation contains a progression of fields, including the object's class id and segmentation mask. The segmentation format is measured as the objects of interest representing a singular object or a collection of things.

III. RESEARCH METHODOLOGY

MS COCO dataset consists of 328,000 images. Many corporates are using the dataset format for their machine learning pipelines.

The format can be circulated with its metadata for flexible transfer of knowledge about the dataset among various branches of the companies.

Many datasets are converted into COCO format for easy transfer of data. The directories and annotation text files are contained in the Deep PCB dataset. Annotation files can be in XML, excel, notepad, or CSV.

But, in the Deep PCB dataset, the annotations are stored in a text file. Deep PCB dataset into coco format JSON format respectively, taking these image directories and annotation files.

The output is in the form of MS COCO JSON, converted by reading the image directory and the corresponding annotation file, which is in a text file.

The annotation of the defect on the tested image is of the format - x1, y1, x2, y2, type. Here, the (x1, y1) and (x2, y2) are the defect bounding box coordinates of the top left and right bottom corner.

The annotation text file has x1, y1, x2, and y2 coordinates, and the last number in Fig. 5 is the category ID. Each object annotation contains various fields, including the object's class id and segmentation mask. The segmentation format is measured as the objects of interest representing a singular object or a collection of things.

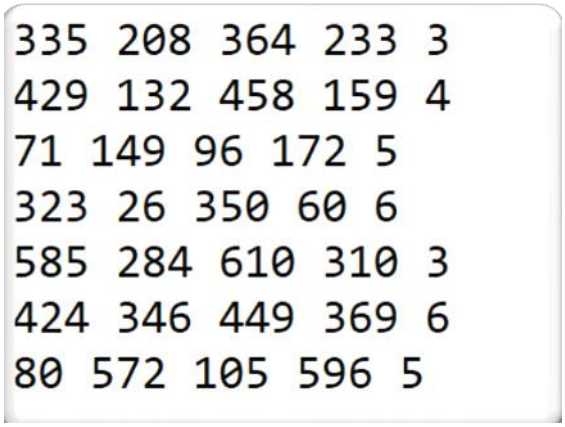

Figure 5. Annotation text file

The class ID with its respective category names are as follows as shown in Fig.6 below –

- 0-background (that is not used)
- 1-open
- 2-short
- 3-mouse bite
- 4-spur
- 5-copper
- 6-pin-hole.

The JSON format has the following keys in COCO format with description –

Info

- "year": year in which the datasaet was made
- "version": version of the dataset
- "description": Description about the dataset
- "contributor":Contributor name to the dataset
- "url": URL of the dataset
- "date_created": Date and time the dataset got created

Images

- "id": Image ID
- "width": width of the image
- "height":height of the image
- "file_name": Image file name
- "license": license of the dataset
- "flickr_url": Flick URL of the dataset
- "coco_url":COCO URL of the dataset
- "date_captured": date and time the dataset was created

Figure 6. COCO Keys Description - 1

Annotation

- "id": Annotation ID
- "image_id": Image ID
- "category_id": Category to which the object belongs to "segmentation": RLE or [polygon]
- "area": area of the object of interest
- "bbox": [x-coordinate of starting point of the bounding box, y-coordinate of starting point of the bounding box, width -coordinate of starting point of the bounding box ,height -coordinate of starting point of the bounding box]
- "iscrowd": 0 or 1

License

- "id": License ID
- "name" :License name
- "url":URL of the license

Categories

- "id": Class ID
- "name":category name
- "supercategory":super category name

Figure 7. COCO Keys Description – 2

The train and validation text file has the name of the image and corresponding annotation text filename

```
[['group20085/20085/20085000_test.jpg', 'group20085/20085_not/20085000.txt'],
 ['group20085/20085/20085001_test.jpg', 'group20085/20085_not/20085001.txt'],
 ['group20085/20085/20085002_test.jpg', 'group20085/20085_not/20085002.txt'],
 ['group20085/20085/20085003_test.jpg', 'group20085/20085_not/20085003.txt'],
 ['group20085/20085/20085004_test.jpg', 'group20085/20085_not/20085004.txt'],
 ['group20085/20085/20085005_test.jpg', 'group20085/20085_not/20085005.txt'],
 ['group20085/20085/20085006_test.jpg', 'group20085/20085_not/20085006.txt'],
 ['group20085/20085/20085007_test.jpg', 'group20085/20085_not/20085007.txt'],
 ['group20085/20085/20085008_test.jpg', 'group20085/20085_not/20085008.txt'],
 ['group20085/20085/20085009_test.jpg', 'group20085/20085_not/20085009.txt'],
 ['group20085/20085/20085010_test.jpg', 'group20085/20085_not/20085010.txt'],
 ['group20085/20085/20085011_test.jpg', 'group20085/20085_not/20085011.txt'],
 ['group20085/20085/20085012_test.jpg', 'group20085/20085_not/20085012.txt'],
 ['group20085/20085/20085013_test.jpg', 'group20085/20085_not/20085013.txt'],
 ['group20085/20085/20085014_test.jpg', 'group20085/20085_not/20085014.txt'],
```

Figure 8. List of Train and validation images filenames and corresponding annotation filenames

The custom functions are made to store the data required in COCO format, a dictionary format. The width and height are needed to be calculated by following.

Width = $x_2 - x_1$
Height = $y_2 - y_1$

The image ID in images and annotation key of the COCO dictionary are the same. There can be multiple annotations for a single image.

The categories contain the name of the category, its ID, and super-category.

License and info can be filled based on their respective keys description shown in Fig 6 and Fig 7.

IV. Result And Discussions

The Deep PCB set annotations and image directories have been read and converted into the COCO format, a JSON file. COCO format can be converted from various annotation formats, namely XML, Excel, Text files, etc.

The respective COCO format JSON has five keys – info, license, images, annotation, and categories. Each has the individual values based on the description mentioned in Fig 6 and Fig 7.

The output JSON file containing the converted COCO format with the data of Deep PCB image directories and respective annotations are shown in figure 9, figure 10, and figure 11.


```
{
    "info": {
        "description": "DeepPCB-2-COCO-Format-2022",
        "url": "",
        "version": "1.0",
        "year": 2022,
        "contributor": "",
        "date_created": "2022/1/10"
    },
    "licenses": [
        {
            "url": "",
            "id": 0,
            "name": ""
        }
    ],
    "images": [
        {
            "id": 1000,
            "license": 0,
            "coco_url": "",
            "flickr_url": "",
            "height": 640,
            "width": 640,
            "file_name": "20085000_test.jpg",
            "date_captured": "2022"
        },
        {
            "id": 1001,
            "license": 0,
            "coco_url": "",
            "flickr_url": "",
            "height": 640,
            "width": 640,
            "file_name": "20085001_test.jpg",
            "date_captured": "2022"
        },
```


Figure 9. COCO JSON Output - 1


```
    ],
    "annotations": [
        {
            "id": 0,
            "category_id": 3,
            "iscrowd": 0,
            "segmentation": [],
            "image_id": 1000,
            "area": 0.0,
            "bbox": [
                409.0,
                394.0,
                26.0,
                28.0
            ]
        },
        {
            "id": 1,
            "category_id": 3,
            "iscrowd": 0,
            "segmentation": [],
            "image_id": 1000,
            "area": 0.0,
            "bbox": [
                275.0,
                383.0,
                44.0,
                34.0
            ]
        },
        {
            "id": 2,
            "category_id": 4,
            "iscrowd": 0,
            "segmentation": [],
            "image_id": 1000,
            "area": 0.0,
            "bbox": [
                8.0,
                163.0,
                28.0,
                28.0
```


Figure 10. COCO JSON Output - 2

```
    "categories": [
        {
            "supercategory": "open",
            "id": 1,
            "name": "open"
        },
        {
            "supercategory": "short",
            "id": 2,
            "name": "short"
        },
        {
            "supercategory": "mousebite",
            "id": 3,
            "name": "mousebite"
        },
        {
            "supercategory": "spur",
            "id": 4,
            "name": "spur"
        },
        {
            "supercategory": "copper",
            "id": 5,
            "name": "copper"
        },
        {
            "supercategory": "pin-hole",
            "id": 6,
            "name": "pin-hole"
        }
    ]
}
```


Figure 11. COCO JSON Output - 3

V. CONCLUSION

There are millions of corporates machine learning pipelines with neural network models that use the COCO format. The JSON file containing the data about the dataset in COCO format is a python dictionary dumped into JSON. COCO format is used as it has many images and is used flexibly by many models that can smoothly transfer data about the dataset through various departments of the company. The values for the respective keys are effectively stored by reading the images in the image directories and the corresponding annotation text files. The individual COCO format JSON has five keys – info, license, images, annotation, and categories. Each has the respective values stored against the key.

REFERENCES

[1] "Microsoft COCO: Common Objects in Context Data format" https://cocodataset.org/#format-data

[2] Allena Venkata Sai Abhishek , Sonali Kotni, 2021, Detectron2 Object Detection & Manipulating Images using Cartoonization, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 08 (August 2021),

[3] Sujith Jayaprakash Balamurugan E. 2015. A Comprehensive Survey on Data Preprocessing Methods in Web Usage Mining.

[4] Sanli Tang, Fan He, Xiaolin Huang, Jie Yang (2019) "Online PCB Defect Detector On A New PCB Defect Dataset" arXiv:1902.06197 [cs.CV]

[5] Lv, Teng & Yan, Ping & He, Weimin. (2018). Survey on JSON Data Modelling. Journal of Physics: Conference Series. 1069. 012101. 10.1088/1742-6596/1069/1/012101.

[6] Allena Venkata Sai Abhishek, Venkateswara Rao Gurrala "AugStatic - A Light-Weight Image Augmentation Library", International Journal of Emerging Technologies and Innovative Research (www.jetir.org), ISSN: 2349-5162, Vol.9, Issue 5, page no.b735-b742, May-2022, Available :http://www.jetir.org/papers/JETIR2205199.pdf

[7] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár (2015) "Microsoft COCO: Common Objects in Context" arXiv:1405.0312 [cs.CV]

[8] Allena Venkata Sai Abhishek, Dr. Venkateswara Rao Gurrala, Dr. Laxman Sahoo, "RESNET18 MODEL WITH SEQUENTIAL LAYER FOR COMPUTING ACCURACY ON IMAGE CLASSIFICATION DATASET", International Journal of Creative Research Thoughts (IJCRT), ISSN:2320-2882, Volume.10, Issue 5, pp.c176-c181, May 2022, Available at :http://www.ijcrt.org/papers/IJCRT2205235.pdf

[9] Peng, Dunlu & Cao, Lidong & Xu, Wenjie. (2011). Using JSON for Data Exchanging in Web Service Applications. 7.

[10] Allena Venkata Sai Abhishek and Venkateswara Rao Gurrala, Improving Model Performance and Removing the Class Imbalance Problem Using Augmentation, International Journal of Advanced Research in Engineering and Technology (IJARET). 13(5), 2022, pp. 14-22 doi: https://doi.org/10.17605/OSF.IO/CQMY5

[11] Garima Dutt, Anup Singh Kushwaha (2016) "Review of JSON Data Interchange Format" https://www.ijirt.org/Article?manuscript=143722

[12] Anusha Kanukolanu, Dr. S Phani Kumar, Allena Venkata Sai Abhishek (2022) "Augmentation Techniques Analysis with removal of Class Imbalance using PyTorch for Intel Scene Dataset" https://ijirt.org/Article?manuscript=155039

[13] Anca-Raluca Breje, Robert Győrödi, Cornelia Győrödi, Doina Zmaranda, George Pecherle (2018) Comparative Study of Data Sending Methods for XML and JSON Models (IJACSA) International Journal of Advanced Computer Science and Applications

[14] Papaleo, Laura. (2013). Introduction to XML and its applications. 10.1142/9789812836304_0006.

[15] Saba, Amir & Shahab, Elham & Abdolrahimpour, Hadi & Hakimi, Mahsa & Moazzam, Akbar. (2017). A Comparative Analysis of XML Documents, XML Enabled Databases and Native XML Databases.

[16] Jennifer Widom (1999) "Data Management for XML: Research Directions" http://ilpubs.stanford.edu:8090/404/1/1999-48.pdf

[17] Mustafa Atay (2017) "DEVELOPMENT AND MAINTENANCE OF XML-BASED VERSUS HTML-BASED WEBSITES: A CASE STUDY" https://arxiv.org/ftp/arxiv/papers/1707/1707.01818.pdf

[18] Allena Venkata Sai Abhishek, Dr. Venkateswara Rao Gurrala, "AUGMENTED BALANCED IMAGE DATASET GENERATOR USING AUGSTATIC LIBRARY", IJRAR - International Journal of Research and Analytical Reviews (IJRAR), E-ISSN 2348-1269, P- ISSN 2349-5138, Volume.9, Issue 2, Page No pp.536-543, May 2022, Available at : http://www.ijrar.org/IJRAR22B1901.pdf

[19] Mahmud, S M Hasan & Hossin, Md & Jahan, Hosney & Noori, Sheak & Bhuiyan, Touhid. (2018). CSV-ANNOTATE: Generate annotated tables from CSV file. 10.1109/ICAIBD.2018.8396169.

[20] Marcelo Arenas, Francisco Maturana, Cristian Riveros, Domagoj Vrgoc (2020) A framework for annotating CSV-like data

[21] Bondarenko, Solomiia. (2021). Annotation. 10.13140/RG.2.2.21113.19044.

[22] Kvasnytsia, Roksoliana. (2021). ANNOTATION.