IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)
An International Open Access, Peer-reviewed, Refereed Journal

PRINTED CIRCUIT BOARD FAULT DETECTION

M. Venkatesh, Asst.Professor,ECE Dept, BVRIT HYDERABAD Collage of Engineering for Women, Hyderabad

CHAKKA JAHNAVI

Student, Department of Electronics and Communication Engineering (ECE)
BVRIT HYDERABAD Collage of Engineering for Women, Hyderabad, India

Abstract: Printed circuit board (PCB) is a platform which mechanically supports and electrically connects components in a well-defined manner. PCBs are prone to a variety of defects that impede proper manufacturing, costing companies money. A proper examination of PCB is needed in order to get accurate results. With all of the parts, connections, and other critical aspects of a circuit board that must be examined, some complicated PCBs can have up to nearly thousand opportunities for errors. Locating a single problem out of so many possibilities in a large PCB is difficult with manual inspection methods. In this project "Printed Circuit Board Fault Detection" we will build a full PCB Fault Classifier using image processing and deep learning that automates the task of detecting a PCB with faults rather than giving models over the fence to developers and other engineers. Here we would like to implement a model such that all challenges in detection of faults are met. This helps automate tasks that are manual, time consuming and prone to human error.

***Index Terms* – PCB, Fault Classifier, Image processing, Deep learning.**

I. Introduction

Printed circuit boards (PCBs) are the foundational building block of most modern electronic devices. Semiconductors, connectors, resistors, diodes, capacitors and radio devices are mounted to, and "talk" to one another through the PCB.PCB's have mechanical and electrical attributes that make them ideal for various applications. Founded in 1977, Printed Circuits LLC has since become a ground-breaking printed circuit board manufacturer. Originally manufacturing all types of PCB's, they drove towards specialization in rigid flex and flexible circuit manufacturing in the mid 1990's.PCB's were developed in the early 20th century but have had a continued escalated development in technology since then. The advancement and widespread adoption of technology in PCBs has paralleled the rapid advancement in semiconductor packaging technology and has enabled industry professionals to invest in smaller and more efficient electronics. Conventional PCB's can be as simple as a single layer of circuitry or can go to fifty layers or more. They consist of electrical components and connectors linked via conductive circuits – usually copper, with the purpose of routing electrical signals and power within and between devices.

Usage of Printed Circuit Boards

Compared to traditional wired circuits, PCBs offer a number of advantages. Their small and lightweight design is appropriate for use in many modern devices, while their reliability and ease of maintenance suit them for integration in complex systems. Additionally, their low cost of production makes them a highly cost-effective option. These qualities are some of the reasons PCBs find application across industries, including in medical, aerospace, military, industrial and commercial.

1.1 Materials used for Printed Circuit Board:

The primary materials used in the manufacturing of PCBs are fiberglass or plastic substrates, copper, solder mask, and nomenclature ink. Common plastic substrates for flexible circuits include polyimide (PI), liquid crystal polymer (LCP), polyester (PET), and polyethylene naphthalate (PEN). The purpose of the substrate is to provide a non-conductive base upon which the conductive circuits can be constructed and insulated from one another.

Due to high electrical conductivity, copper is the most used conducting material for circuitry in PCBs. The laminates come with thin sheets of copper foil laminated to one or both sides of the plastic. The thickness and number of layers required are largely dependent upon the application for which the PCB will be used. multi-layered PCBs are constructed by alternating layers of copper circuitry and insulating materials to complete the PCB.1 oz copper i.e 35µm thick or 1.4 mils is Standard internal layer copper thickness for "standard construction product for 1 oz and 2 oz finished copper weight selections. This is also the standard starting copper weight on the external layers for PCBs with the 2 oz finished copper weight selection.

Soldermask is a liquid, usually an epoxy material, which is applied onto the outerlayers of rigid PCBs. It is also called as Solder stop Mask or Solder Resist. It is also commonly used on the rigid sections of rigid flex PCB's. Soldermask is primarily designed to insulate the copper circuits on outerlayers from oxidation from the environment. Soldermask is also designed to control and retain the flow of solder when the components are assembled to the PCB. Solder Mask is traditionally green,but is also available in many colours.

1.2 Fabrication of Printed Circuit Board:

The construction and fabrication of PCBs include the following steps:

- Chemically imaging and etching the copper layers with pathways to connect electronic components
- Laminating the layers together, using a bonding material, that also acts as electrical insulation, to create the PCB
- Drilling and plating the holes in the PCB to connect all of the layers together electrically
- Imaging and plating the circuits on the outside layers of the board
- Coating both sides of the board with soldermask and printing the nomenclature markings on the PCB
- The boards are then machined to the dimensions that are in the designer's perimeter gerber file.

After these steps, the PCB board is ready for components to be assembled to it. Most commonly the components are attached to the PCB by soldering the components directly onto exposed traces called pads and holes in the PCB. Soldering can be done by hand, but more typically is accomplished in very high-speed automated assembly machines.

Two of the most common PCB assembly methods are surface-mount device (SMD) or thru-hole technology (THT). The use of either depends on the size of the components and the configuration of the PCB. SMD is useful for directly mounting small components on the exterior of the PCB, while THT is ideal for mounting large components through large pre drilled holes in the board.

1.3 Reasons for Faults in PCB:

Printed circuit boards, or PCBs, are essentially the lifeblood of the devices we use every day. Cell phones, TVs, cars, street lights and so many other devices rely on PCBs, which is why it can be extremely disruptive when one fails. PCBs fail for a litany of reasons, sometimes even before they've been shipped for use. By understanding the causes of PCB failure, it's easie r to prevent future failures.

1.Defects:

Defects are the most prevalent cause of PCB failure. PCBs are extremely sensitive to electrostatic discharge (ESD), and though we can only feel ESD in the thousands of volts, the smallest discharge can cause a component defect. Other potential defects include misaligned layers, short circuits, crossed signals etc.

2. Burnt Components

Component burning is a type of PCB defect, but it's worth noting as a major cause of PCB failure. And a burnt component could do more than require a replacement part -- it might require a whole board overhaul. The three main causes of burnt components include extreme heat, improper component placement, component failure or technician error.

3. Environmental Factors

Circuit boards are sensitive to outside factors as well. Heat and humidity can cause expansion in PCBs, resulting in warping and potentially damaged soldered joints. This is why PCB manufacturing is usually done in a climate-controlled environment where humidity is kept at a safe level.

4. Soldering Issues

Solder is a key ingredient in the PCB process. It's what maintains the contact between a component and circuit, but it can occasionally become contaminated and result in a board failure. If there is too much moisture in the solder, it can become conductive and cause short circuiting.

5. Human Error

Design engineers, assemblers, quality engineers, and seemingly countless others are involved in the production cycle. So it's not a surprise that human error plays a part in more than a few PCB failures. There is shortage of honest (or lazy) mistakes that can harm a PCB, including Misreading a schematic, Incorrectly installing components, Placing traces too close together, resulting in a short, Poor soldering.

6. Age

PCBs tend to age more like humans than fine wine. As they get older, they begin to break down. Components begin to fail after reaching the end of their expected life cycle, and the PCB must be fixed or replaced with a new one.

1.4 Most Common Faults in PCB

Faults in PCB can have numerous reasons for their Occurrence. Here are the most Common Faults of Printed Circuit Board.

- Spur
- Spurious Copper
- Mouse Bites
- Missing hole
- Open Circuit
- Short Circuit

These faults will affect the proper functioning of a Printed Circuit Board and may result in false and inaccurate results. Thus, PCB fault detection is very much important.

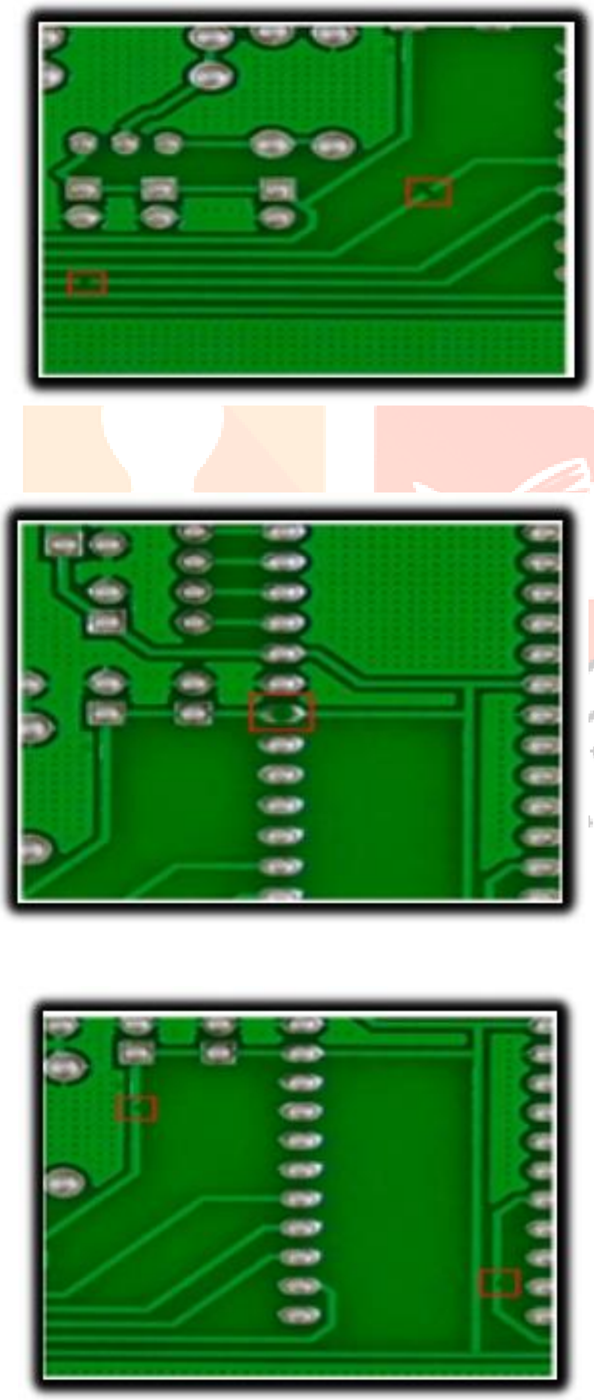


Fig 3. Open Circuit

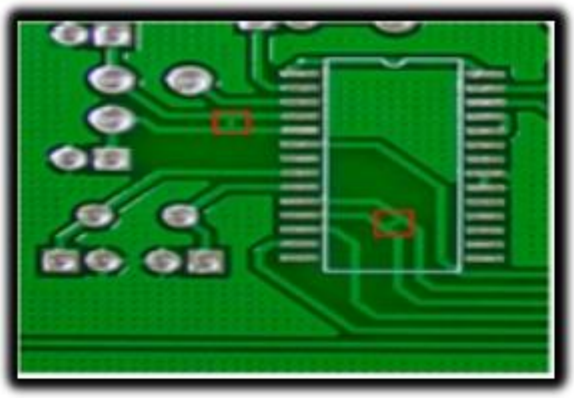


Fig 4. Spur

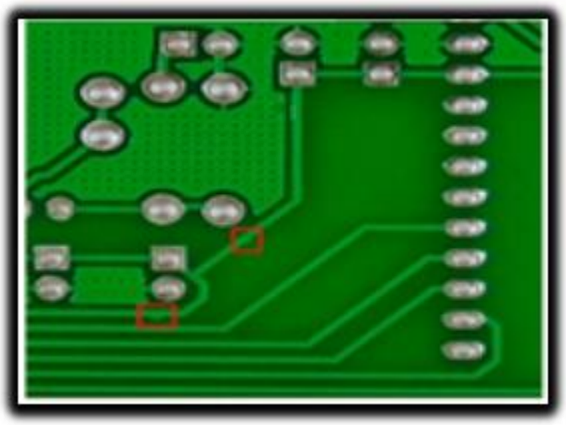


Fig 5. Spurious Copper

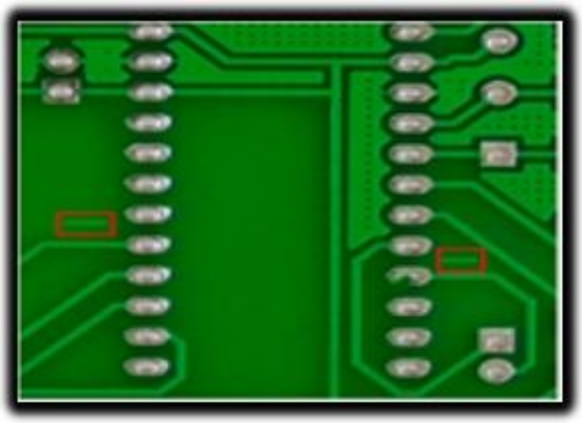


Fig 6. Short Circuit

II. LITERATURE SURVEY

Statistics shows that the global sales value of Printed Circuit Board
products are 82 million U.S Dollars. The fabrication of a complete PCB assembly requires an array of machines and materials, which include Screen printers, Conveyors, Pick-and-place systems etc. In case of a Complex PCB there is a greater possibility for the rise of faults. Faults in the PCB during manufacturing stage may arise because of incorrect placement of components, lack of space in the PCB, Bad Component Placement etc., in few cases some of these issues may get overlooked at initial stages. As a result, huge faults are observed in the PCB in long run. Also, though everything is perfect during the manufacturing stage, we still have faults in the PCB after the production. This happens because of Exposure to heat, dust and moisture, accidental impact i.e. drops and falls, thermal stress, poor connectivity between the components, chemical fluid leakage and power overloads/surges. The most common PCB faults are Bad Soldering, Mouse Bites, Pin holes, Spur, Short, Spurious Copper, Aging of components, Slivers. There are many techniques for PCB Fault detection and each technique has its advantages and disadvantages.

Micro Sectioning Analysis, also called as Cross-Sectioning is a PCB fault detection Method which identifies faults like Component Defects, Thermo Mechanical Failures, Processing failures due to solder reflow and raw material evaluations. Here in this technique a failure analyst will remove a two-dimensional slice out of a sample, which uncovers features within the board. Later the analyst will put the slice in a medium such that the defect gets repaired. One advantage of this technique is the ability to position the sample on a flat surface and invest each of its parts together. Disadvantage with this technique is that it needs manual power and is a destructive method which becomes more complex in case of large Printed Circuit Boards.

PCB Fault Detection using Image Processing is another technique but does not need the manual intervention at every stage of fault detection. Image Subtraction is the algorithm used here. The PCB to be inspected is compared with the reference image and the subtracted image shows defects. After the image subtraction the defects are indicated with white spots, if there is no white spot then there is no defect in the PCB image. Advantage with this technique is that it allows for verification of overall defects in the geometry of the board. But the disadvantage is that it suffers from the practical problems such as reflectivity variation, lightning sensitivity, image registration, color variation and additional noise because of misalignment of components while comparing the boards.

PCB Defect Detection Using Denoising Convolution Auto encoder is another technique for PCB Fault detection. The proposed auto encoder is trained with a dataset containing image pairs of defective and intact PCBs. Salt and pepper noise is added to the defective PCB such that auto encoder learns better features. The output of the Convolution network is the repaired PCB. Difference between the repaired PCB and input PCB will result in the fault occurred in that respective PCB. Here the subtraction algorithm should be more improved in order to get the accurate result.

Optical Microscopy is one of the popular techniques for the PCB fault detection. This process uses a high-Power microscope with visible light. This technique can detect faults such as improper construction, soldering and assembly issues. The order of magnification is 50,000X to 100,000X. Because of this magnification, result of this technique is much accurate when compared to other techniques. But this technique is limited to detect only few faults and the set up for this technique is also not economical.

III. BASIC REQUIREMENTS

3.1 Artificial Intelligence

Artificial intelligence has been playing a key role in computer science and the field of computers. This term is gaining much more importance in the recent years due to the advances taking place in the field of artificial intelligence and machine learning. It is to develop software applications which are self - learning, and which imitates the human mind in reasoning, problem - solving, decision - making etc. AI is used and has the capability to outperform human actions. Using these technologies, computers can be trained to accomplish specific tasks by processing large amounts of data and recognizing patterns in the data. Artificial intelligence (AI) makes it possible for machines to learn from experience, adjust to new inputs and perform human-like tasks.

Types of Artificial Intelligence: Here are the 4 types of Artificial Intelligence

- ➢ Limited Memory
- ➢ Theory Mind
- ➢ Reactive Machines
- ➢ Self-Awareness

Examples of Artificial Intelligence:

- ➢ Siri, Alexa and other smart assistants
- ➢ Self-driving cars
- ➢ Conversational Bots
- ➢ Email Spam filters
- ➢ Netflix Recommendations

3.2 Machine Learning

It is the subfield of Artificial Intelligence that develops computer programs, and accesses the data by providing the ability for the system to learn and improve automatically by finding the patterns present in the database without any human intervention. The model training method in machine learning is been classified as supervised learning and unsupervised learning based on the 27 types of the data. The simple definition of the Machine Learning is finding patterns in the dataset and predicting the future by using those patterns. Machine Learning became popular currently due to the plenty of data, increased computer power and effective algorithms. Learning Process involves two steps i.e., identifying the patterns in the provided data and recognizing those patterns if they are seen again.

There are undeniable practical advantages to machine learning, namely:

- ➢ Intelligent big data management – The sheer volume and variety of data being generated as humans and other environmental forces interact with technology would be impossible to process and draw insights from without the speed and sophistication of machine learning.
- ➢ Smart devices – From wearable devices that track health and fitness goals to self-driving cars to "smart cities" with infrastructure that can automatically reduce wasted time and energy, the Internet of Things (IOT) holds great promise, and machine learning can help make sense of this significant increase in data.
- ➢ Rich consumer experiences – Machine learning enables search engines, web apps, and other technology to customize results and recommendations to match user preferences, creating delightfully personalized experiences for consumers

3.3 Deep Learning

It is a subset of Machine Learning in Artificial Intelligence, which contains neural network with three or more layers. These neural networks simulate the behaviour of human brains and allow it to learn from large data sets. Approximate predictions are made by a single layered neural network. Adding hidden layers can help to optimize and improve accuracy. In Deep learning, recognition accuracy is higher than ever before. This helps the users to meet their expectations. The hierarchy of concepts allows the computer to learn complicated concepts by building them out of simpler ones. If we draw a graph showing how these concepts are built on top of each other, the graph is deep, with many layers. For this reason, we call this approach to AI deep learning.

Deep-learning methods are representation-learning methods with multiple levels of representation, obtained by composing simple but non-linear modules that each transform the representation at one level (starting with the raw input) into a representation at a higher, slightly more abstract level. The key aspect of deep learning is that these layers of features are not designed by human engineers: they are learned from data using a general-purpose learning procedure.

IV. SOFTWARE REQUIREMENT

4.1 Visual Studio Code:

Visual Studio Code is commonly called as VS Code. It is a source code editor made by Microsoft for Windows, Linux and MacOs that can be used with variety of programming languages like Java, Java Script, Python, Go, Nodejs, Fortan and C++. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring and embedded git. Users can change the theme, keyboard shortcuts, preferences and can install extensions that add additional functionality.

Source control is a built in Feature in Visual Studio Code. It has a dedicated tab inside of the menu bar where you can access version control settings and view changes made to the current project. To use the feature you must link Visual Studio Code to any supported version control system. This allows you to create repositories as well as to make push and pull requests directly from the Visual Studio Code program.

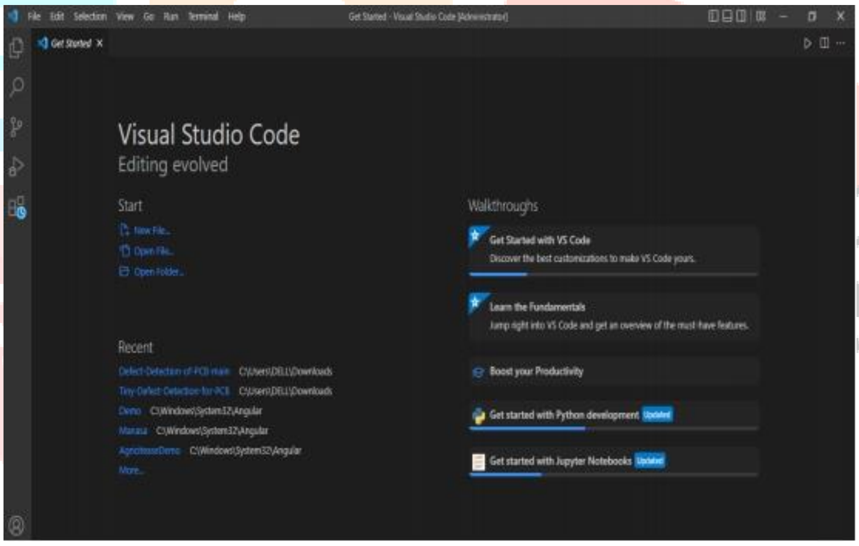

File Edit Selection View Go Run Terminal Help
Get Started - Visual Studio Code [Administrator]
Get Started
Visual Studio Code
Editing evolved
Start
New File...
Open File...
Open Folder...
Recent
Walkthroughs
Get Started with VS Code
Discover the best customizations to make VS Code yours.
Learn the Fundamentals
Jump right into VS Code and get an overview of the must-have features.
Boost your Productivity
Get started with Python development
Updated
Get started with Jupyter Notebooks
Updated
More...

Fig 7. Visual Studio Code

Advantages of Visual Studio Code:

- Support for multiple programming languages
- Extensions and Support
- Git Support
- Multiple Projects and Terminal Support.
- Hierarchy Structure and Improved Code
- Intellisense and Cross Platform Support.

4.1.1. Python:

Python is an interpreted high-level general-purpose programming language. Its design philosophy emphasizes code readability with its use of significant indentation. Its language constructs as well as its object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

Advantages of Python:

- Easy to Read, Learn and Write
- Improved Productivity
- Interpreted Language
- Dynamically Typed
- Free and Open-Source
- Vast Libraries Support
- Portability

4.2 Libraries:

A library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs.

Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization etc. The Python Standard Library contains the exact syntax, semantics, and tokens of Python. It contains built-in modules that provide access to basic system functionality like I/O and some other core modules. Most of the Python Libraries are written in the C programming language. The Python standard library consists of more than 200 core modules. All these work together to make Python a high-level programming language. Python Standard Library plays a very important role. Without it, the programmer s can't have access to the functionalities of Python. But other than this, there are several other libraries in Python that make a programmer's life easier.

Here are some of the python libraries used:

- Tensorflow
- Pandas
- Numpy
- Keras

Tensorflow:

This library was developed by Google in collaboration with the Brain Team. It is an open-source library used for high-level computations. It is also used in machine learning and deep learning algorithms. It contains a large number of tensor operations. Tensor flow allows to Build and train ML models easily using intuitive high-level APIs like Keras with eager execution, which makes for immediate model iteration and easy debugging, Easily train and deploy models in the cloud in the browser, or on-device no matter what language you use, A simple and flexible architecture to take new ideas from concept to code, to state-of-the-art models, and to publication faster.

Pandas:

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term " pan el data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc. Pandas is mainly used for data analysis and associated manipulation of tabular data in Dataframes. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features. The development of pandas introduced into Python many comparable features of working with Dataframes that were established in the R programming language. The pandas library is built upon another library NumPy, which is oriented to efficiently working with arrays instead of the features of working on Dataframes.

Numpy:

NumPy (Numerical Python) is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a NumFOCUS fiscally sponsored project. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.

Keras:

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3, Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. As of version 2.4, only TensorFlow is supported. Designed to enable fast experimentation with deep neural networks, it focuses on being user-friendly, modular, and extensible. It was developed as part of the research effort of project ONEIROS (Open-ended Neuro-Electronic Intelligent Robot Operating System), and its primary author and maintainer is François Chollet, a Google engineer. Chollet is also the author of the Xception deep neural network model.

Use of libraries in program

For the easy maintenance of the code, we split the code into different parts and we can use that code later ever we need it. In Python, modules play that part. Instead of using the same code in different programs and making the code complex, we define mostly used functions in modules and we can just simply import them in a program wherever there is a requirement. We don't need to write that code but still, we can use its functionality by importing its module. Multiple interrelated modules are stored in a library. And whenever we need to use a module, we import it from its library. In Python, it's a very simple jo b to do due to its easy syntax. We just need to use import.

4.3 Flask

Flask is a web framework, it's a Python module that lets you develop web applications easily. It's has a small and easy-to- extend core. It's a micro framework that doesn't include an ORM (Object Relational Manager) or such features. A Web Framework represents a collection of libraries and modules that enable web application developers to write applications without worrying about low-level details such as protocol, thread management, and so on. Flask is a WSGI web app framework. It is based on the Werkzeg WSGI toolkit and the Jinja2 template engine. Werkzeug is a WSGI toolkit that implements requests, response objects, and utility functions. This enables a web frame to be built on it. Jinja2 is a popular template engine for Python.A web template system combines a template with a specific data source to render a dynamic web page.This allows you to pass Python variables into HTML templates.

Major advantages of Flask are Ease of setup and use and Freedom to build the structure of the web application. With freedom comes responsibility, similarly, Flask needs the developers to carefully structure it, since Flask doesn't have "flask rules" to follow as compared to frameworks like Django. As the web app increases in complexity, this structuring is what is going to be the foundation.

4.4 HTML

The HyperText Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as img and input directly introduce content into the page. Other tags such as p surround and provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags but use them to interpret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages. Inclusion of CSS defines the look and layout of content. The World Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997. A form of HTML, known as HTML5, is used to display video and audio, primarily using the canvas element, in collaboration with javascript.

4.5 CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript. CSS is designed to enable the separation of presentation and content, including layout, colors, fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page in different styles for different rendering methods, such as on-screen, in print, by voice (via speech-based browser or screen reader), and on Braille- based tactile devices. CSS also has rules for alternate formatting if the content is accessed on a mobile device. The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable. The CSS specifications are maintained by the World Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents. In addition to HTML, other markup languages support the use of CSS including XHTML, plain XML, SVG, and XUL.

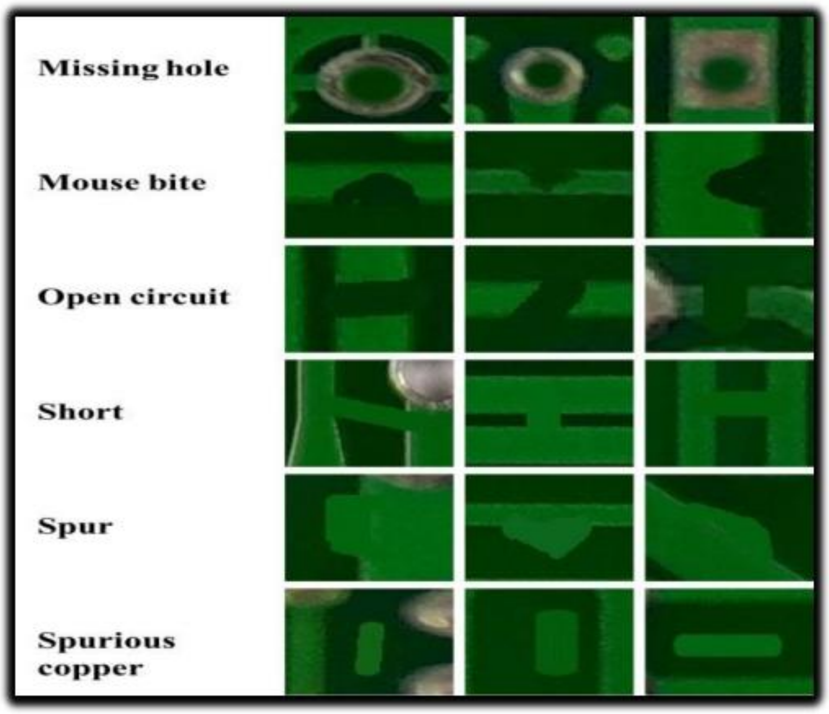
4.6 Bootstrap and Java Script

Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains HTML, CSS and (optionally)Java Script based design for typography, forms, buttons, navigation, and other interface components. As of April 2022, Bootstrap is the eleventh most starred project on GitHub, with over 156,000 stars. The most prominent components of Bootstrap are its layout components, as they affect an entire web page. The basic layout component is called "Container", as every other element in the page is placed in it. Developers can choose between a fixed-width container and a fluid-width container. While the latter always fills the width of the web page, the former uses one of the five predefined fixed widths, depending on the size of the screen showing the page.

JavaScript often abbreviated JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for web page behavior, often incorporating third-party libraries. It has dynamic typing, prototype-based object- orientation, and first-class functions. It is multi-paradigm, supporting event- driven, functional, and imperative programming styles. It has application programming interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

V. DATASET

The Open Lab on Human Robot Interaction of Peking University has released the PCB defect dataset. 6 types of defects are made by photoshop, a graphics editor published by Adobe Systems. The defects defined in the dataset are: missing hole, mouse bite, open circuit, short, spur, and spurious copper. This PCB dataset containing 1500 images with 6 kinds of defetcs (missing hole, mouse bite, open circuit, short, spur, spurious copper) for the use of detection, classification and registration tasks.



VI. PROPOSED METHODOLOGY

In this project we proposed the method of detecting the faults in Printed Circuit Boards using Deep Learning. Through Transfer Learning we use Pre trained Models in the deep learning which helps in getting accurate results. Transfer Learning: A neural network is trained on a data. This network gains knowledge from this data, which is compiled as “weights” of the network. These weights can be extracted and then transferred to any other neural network. Instead of training the other neural network from scratch, we “transfer” the learned features. Pre-Trained Models: A pre-trained model is a model created by someone else to solve a similar problem. Instead of building a model from scratch to solve a similar problem, you use the model trained on other problem as a starting point. By using pre-trained models which have been previously trained on large datasets, we can directly use the weights and architecture obtained and apply the learning on our problem statement. This is known as transfer learning. We “transfer the learning” of the pre-trained model to our specific problem statement. We make modifications in the pre-existing model by fine-tuning the model. Since we assume that the pre-trained network has been trained quite well, we would not want to modify the weights too soon and too much. While modifying we generally use a learning rate smaller than the one used for initially training the model.

One of the pre trained Model being used in this project is **ResNet.**

ResNet:

Residual Network (ResNet) is one of the famous deep learning models that was introduced by Shaoqing Ren, Kaiming He, Jian Sun, and Xiangyu Zhang. In this network, we use a technique called skip connections. The skip connection connects activations of a layer to further layers by skipping some layers in between. This forms a residual block. In order to solve the problem of the vanishing/exploding gradient, this architecture introduced the concept called Residual Blocks. Resnets are made by stacking these residual blocks together. The approach behind this network is instead of layers learning the underlying mapping, we allow the network to fit the residual mapping.

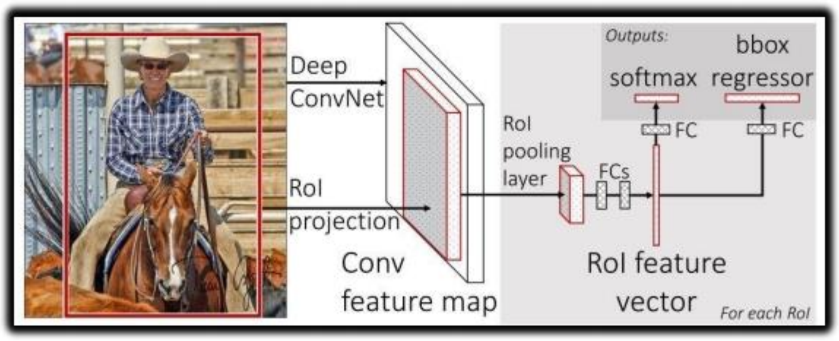


Fig 9: ResNet Model

In the above figure, the very first thing we can notice is that there is a direct connection that skips some layers of the model. This connection is called 'skip connection' and is the heart of residual blocks. The output is not the same due to this skip connection. Without the skip connection, input 'X gets multiplied by the weights of the layer followed by adding a bias term. Then comes the activation function, f() and we get the output as H(x).

H(x)=f(wx + b) or H(x)=f(x).
Now with the introduction of a new skip connection technique, the output is H(x) is changed to H(x)=f(x)+x.

The skip connections technique in ResNet solves the problem of vanishing gradient in deep CNNs by allowing alternate shortcut path for the gradient to flow through. Also, the skip connection helps if any layer hurts the performance of architecture, then it will be skipped by regularization. Keras is an open-source deep- learning library capable of running on top of TensorFlow. Keras Applications provides the following ResNet versions.

- ResNet50
- ResNet50V2
- ResNet101
- ResNet101V2
- ResNet152
- ResNet152V2
-

Fast R-CNN

Fast R-CNN is an object detector that was developed solely by Ross Girshick, a Facebook AI researcher and a former Microsoft Researcher. Fast R-CNN overcomes several issues in R-CNN. As its name suggests, one advantage of the Fast R-CNN over R-CNN is its speed. The main contributions Fast R-CNN are:

- Proposed a new layer called ROI Pooling that extracts equal-length feature vectors from all proposals (i.e. ROIs) in the same image. Compared to R-CNN, which has multiple stages (region proposal generation, feature extraction, and classification using SVM), Faster R- CNN builds a network that has only a single stage.
- Faster R-CNN shares computations (i.e., convolutional layer calculations) across all proposals (i.e. ROIs) rather than doing the calculations for each proposal independently. This is done by using the new ROI Pooling layer, which makes Fast R-CNN faster than R-CNN. Fast R-CNN does not cache the extracted features and thus does not need so much disk storage compared to R-CNN, which needs hundreds of gigabytes.
- Fast R-CNN is more accurate than R-CNN.

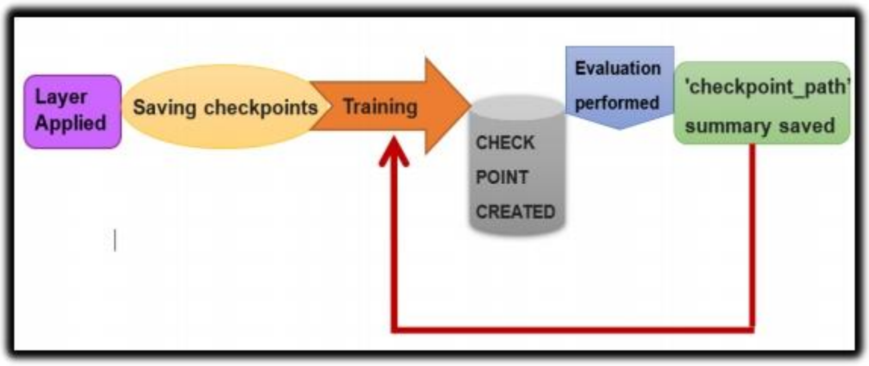


Fig 10: Fast R-CNN Architecture

The general architecture of Fast R-CNN is shown in the figure. The model consists of a single-stage, compared to the 3 stages in R-CNN. It just accepts an image as an input and returns the class probabilities and bounding boxes of the detected objects. The feature map from the last convolutional layer is fed to an ROI Pooling layer. The reason is to extract a fixed-length feature vector from each region proposal. Simply put, the ROI Pooling layer works by splitting each region proposal into a grid of cells. The max pooling operation is applied to each cell in the grid to return a single value. All values from all cells represent the feature vector. If the grid size is 2 ×2, then the feature vector length is 4. In R-CNN, each region proposal is fed to the model independently from the other region proposals. This means that if a single region takes S seconds to be processed, then N regions take S*N seconds. The Fast R-CNN is faster than the R-CNN as it shares computations across multiple proposals.

Check Point:

It is an approach where a snapshot of the state of the system is taken in case of system failure. If there is a problem, not all is lost. The checkpoint may be used directly, or used as the starting point for a new run, picking up where it left off. When training deep learning models, the checkpoint is the weights of the model. These weights can be used to make predictions as is, or used as the basis for ongoing training. The Keras library provides a check pointing capability by a callback API.

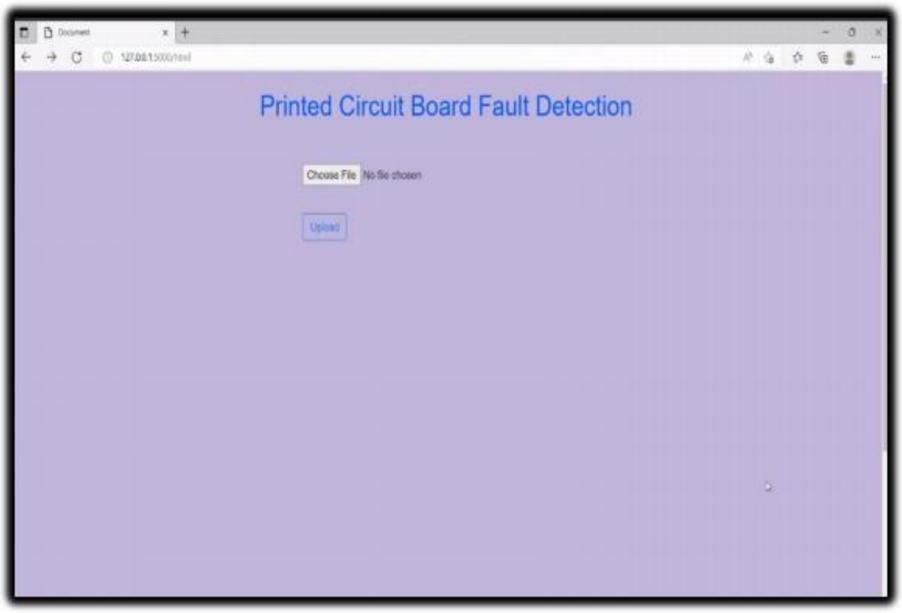

Printed Circuit Board Fault Detection
Choose File No file chosen
Upload

Fig 11. CKPT Block Diagram

Given above is a block diagram explaining the generation of checkpoints during training. Firstly, inside the neural network model the layers as required by the model are applied and then the initial check point file is saved in the model directory as model.ckpt file. After that the main training begins. The training occurs on no of steps as provided by coder and it roughly calculates loss after every 100 steps. When the model loss is minimum as the model feels as if over fitting will occur beyond this point, the training stops and evaluation is performed on the validation data set.

VII. OUTPUT

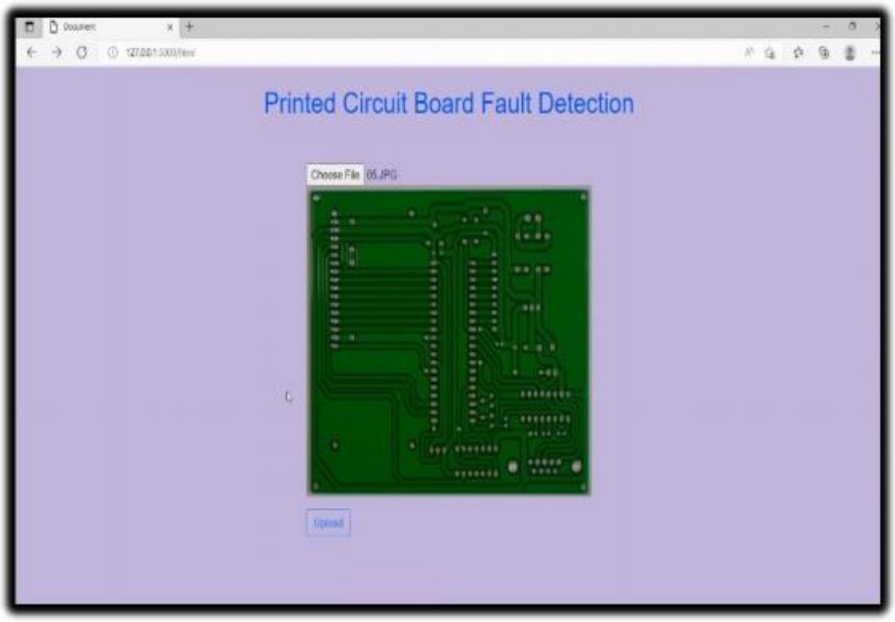


Fig 12. Home Screen

The above figure corresponds to the starting web page of the Application. Here the user can upload image of the printed circuit boards for which the defect detection has to be evaluated.

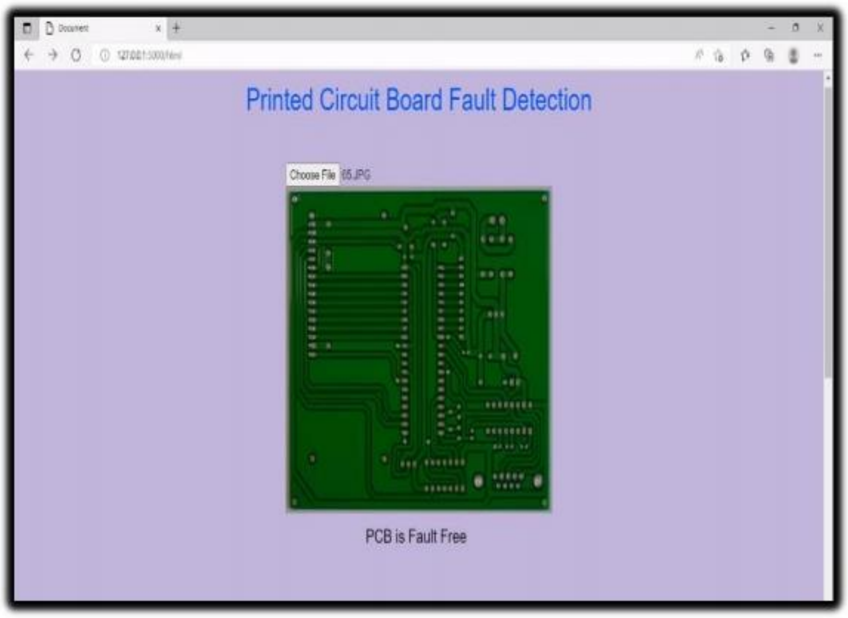


Fig 13. Preview of the image

The above figure represents the preview display of the image that is being uploaded. This helps user to once again view the image uploaded by them. After clicking on Upload the server redirects to another page which confirms regarding the fault detection in PCB.

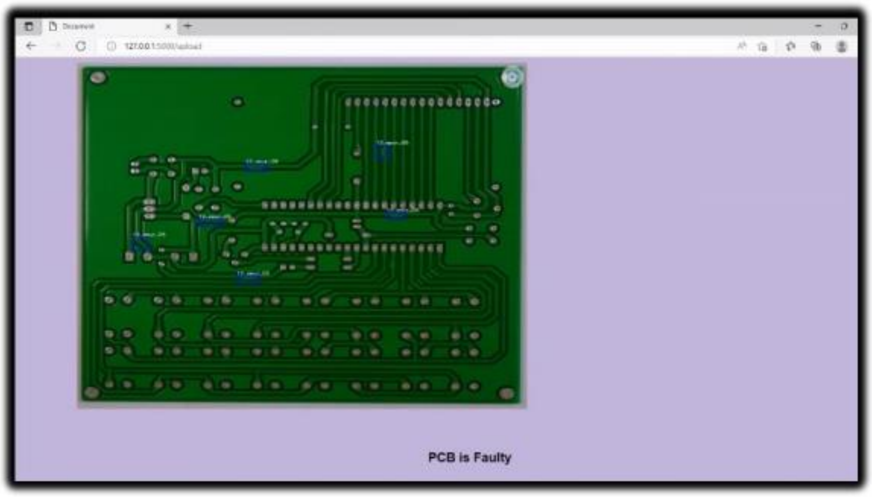


Fig 14. Fault free PCB

The above figure represents the template that is rendered when the uploaded image of PCB does not have any fault in it. Thus, the fault free image is being displayed and shows the user a statement "PCB is Fault Free".

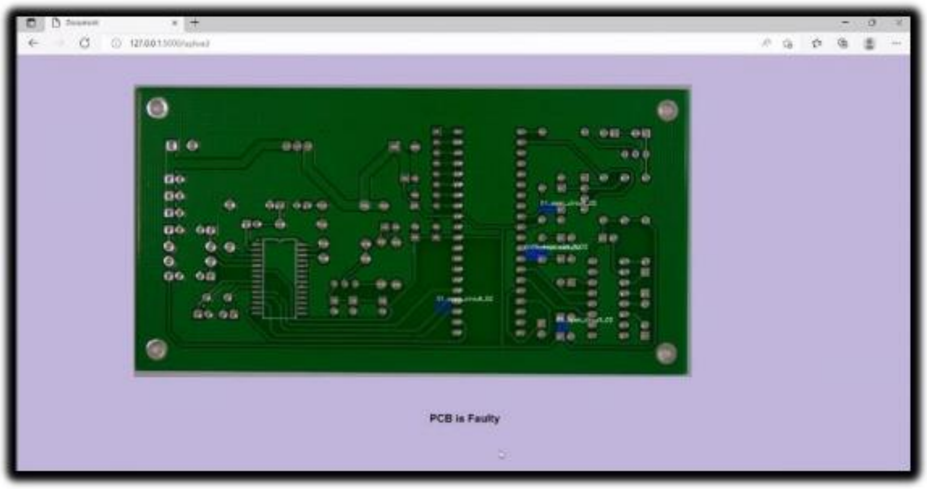


Fig 18. Open Circuit Detection.

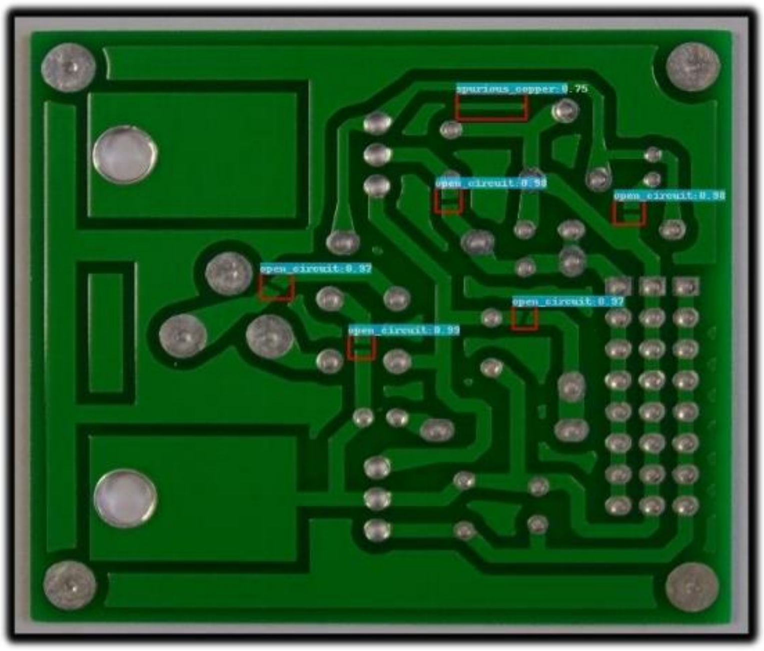


Fig 19. Spur Detection

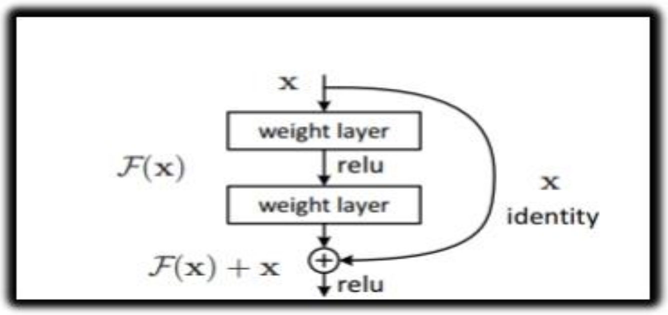

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Fig 15. Fault detected

VIII. CONCLUSION

Inspecting and diagnosing defects manually is challenging. An automated solution to detect and classify faulty PCBs is proposed. A literature review of existing fault detection methods is done and it is understood that a model which is more efficient is needed. A model to detect if a Printed Circuit Board is faulty or not and if it is faulty the respective faults are identified and labelled is been implemented.

IX. REFERENCES

1. Harshitha R, Apoorva G C , Ashwini M C, Kusuma T S, 2018, Components Free Electronic Board Defect Detection and Classification Using Image Processing Technique, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) NCESC – 2018 (Volume 6 – Issue 13)

2. ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2.

3. Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28, pp.91-99.

4. Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. In Advances in Neural Information Processing Systems 25. Curran Associates, Inc.. 2012.

5. J. P. R. Nayak, B. D. Parameshachari, K. M. S. Soyjaudah, Rajashekarappa, R. Banu and A. C. Nuthan, "Identification of PCB faults using image processing ,2017 International Conference on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT), 2017, pp. 1-4, doi: 10.1109/ICEECCOT.2017.8284602.

6. W. Wang, Y. Yang, X. Wang, W. Wang, and J. Li, "The development of convolution neural network and its application in image classification: a survey," Optical Engineering, vol. 58, no. 4, Article ID 040901, 2019.

6. E. L. Denton, W. Zaremba, J. Bruna, Y. LeCun, and R. Fergus, "Exploiting linear structure within convolutional networks for efficient evaluation," in Advances in Neural Information Processing Systems, pp. 1269 –1277, MIT Press, Cambridge, MA, USA, 2014.

7. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.

8. J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng. Quantized convolutional neural networks for mobile devices, 1512.06473, 2015.

X. APPENDIX

10.1 app.py

```
from flask import Flask, request, render_template,redirect,url_for,session,flash import os,shutil
from werkzeug.utils import secure_filename
import pathlib
import tools.eval as parse
import numpy as np
import libs.configs.cfgs as configuration
from PIL import Image

ALLOWED_EXTENSIONS = {'jpg'}
def allowed_file(filename):
return '.' in filename and \
filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

app = Flask(__name__)
app.secret_key = b'_5#y2L"F4Q8z\n\xec]/'
app.config['UPLOADED_FILES'] = 'static'
@app.route('/html')
def index():

return render_template("index.html")

@app.route('/detect', methods =["GET", "POST"])
def detect():
if request.method == "POST":
# getting input with name = fname in HTML form
fname = request.form.get("fname")
print(fname)
return "Welcome to Uneritx " + fname
return render_template("detect.html")

@app.route("/upload", methods=["GET", "POST"])
def upload():
if request.method == 'POST':
try:
if os.path.isfile(app.config['UPLOADED_FILES']) or
os.path.islink(app.config['UPLOADED_FILES']):
os.unlink(app.config['UPLOADED_FILES'])
elif os.path.isdir(app.config['UPLOADED_FILES']):
shutil.rmtree(app.config['UPLOADED_FILES'])
except Exception as e:
print('Failed to delete %s. Reason: %s' %
(app.config['UPLOADED_FILES'], e))
os.mkdir(app.config['UPLOADED_FILES'], 0o666)
f = request.files['File']
print(f.filename)
img1 = Image.open("D:/Tiny-Defect-Detection-for-
PCB/newimage/01_missing_hole_01.jpg")
img1.save(os.path.join(app.config['UPLOADED_FILES'],secure_filename("0
1_missing_hole_01.jpg")))

img = Image.open(f.stream)
img.save(os.path.join(app.config['UPLOADED_FILES'],secure_filename(f.fil
ename)))
```

```
parse.eval(np.inf,eval_dir=configuration.ROOT_PATH+"static",annotation_di
r=configuration.ROOT_PATH+"tools/test_annotation",showbox=False)

path = configuration.ROOT_PATH+"output/detectedimgs/"

# Getting the list of directories
dir = os.listdir(path)

# Checking if the list is empty or not
if len(dir) == 0:
print("Empty directory")
return render_template("noerror.html", uploaded_image=f.filename)
else:
print("Not empty directory")
return render_template("detect.html", uploaded_image=f.filename)
return render_template("detect.html")

@app.route('/static/<filename>')
def send_uploaded_file(filename=''):
from flask import send_from_directory
return send_from_directory(configuration.ROOT_PATH+"output/detectedimgs/",
filename)

if __name__=='__main__':
app.run(debug = True)
```

10.2 index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=<<for>>, initial-scale=1.0">
<title>Document</title>

<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css">
<!-- Optional JavaScript -->
<!-- jQuery first, then Popper.js, then Bootstrap JS -->
<script src="https://code.jquery.com/jquery-3.3.1.min.js"></script>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js"></scr
ipt>
<script
src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" ></script
>
<script
src="https://cpwebassets.codepen.io/assets/common/stopExecutionOnTimeout-
1b93190375e9ccc259df3a57c1abc0e64599724ae30d7ea4c6877eb615f89387.js"></sc
ript>
<script
src="https://cdnjs.cloudflare.com/ajax/libs/gsap/1.19.0/TweenMax.min.js"></script>
<style>
body {
padding: 0;
margin: 0;
```

```
background-color: #c3b3db;
font-family: 'Montserrat', sans-serif;
}

body,
html {
width: 100%;
height: 100%;
}

.img-container {
/* background-image: url('https://s3-us-west-
2.amazonaws.com/s.cdpn.io/409269/valley.jpg'); */
background-position: top center;
background-size: cover;
width: 100%;
height: 100%;
z-index: 5;
position: absolute;

}

h2 {
font-size: 70px;
margin-bottom: 0;
}

h3 {
font-size: 40px;
margin-bottom: 0;
}

.preloader {
width: 100%;
height: 100%;
background-color: #c3b3db;
z-index: 10;
position: absolute;
}

.circle {
border-radius: 190px;
box-shadow: 0 4px 8px rgba(0, 0, 0, 0.4);
position: absolute;
top: 50%;
left: 50%;
opacity: 0;
}

.circle1 {
background-color: #7752d5;
width: 240px;
height: 240px;
margin-top: -120px;
margin-left: -120px;
}

.circle2 {
```

```
background-color: #8362d9;
width: 170px;
height: 170px;
margin-top: -85px;
margin-left: -85px;
}

.circle3 {
background-color: #9f88d6;
width: 100px;
height: 100px;
margin-top: -50px;
margin-left: -50px;
}
</style>
</head>
<body>
<h1 style="color:rgb(0, 85, 255);text-align:center;margin-top:25px;">Printed Circuit Board Fault
Detection</h1>
<script>
function send() {
const formData = new FormData();
const files = document.getElementById("fname");

formData.append("file", files.files[0]);
const requestOptions = {
headers: {
"Content-Type": "multipart/form-data",
},
mode: "no-cors",
method: "POST",
files: files.files[0],
body: formData,
};
console.log(requestOptions);

fetch("http://localhost:5000/upload", requestOptions).then(
(response) => {
console.log(response.data);
}
);
}
</script>
{% block content %}
<div style="width: 100%;display: inline-flex;padding: 50px;">
<form style="margin: auto;" action = "http://127.0.0.1:5000/upload" method = "POST" enctype =
"multipart/form-data">

<input type = "file" name = "File" id="file" title="" onchange="loadFile(event)" />
<p><img id="output" width="500px" height="400px" /></p>
<script>

var loadFile = function(event) {
this.title= ""
var image = document.getElementById('output');
image.src = URL.createObjectURL(event.target.files[0]);
};
</script>
```

```
<!-- <h4 style="color:rgb(0, 0, 0);margin-left:150px;">PCB is Fault Free</h2>--> <input
onclick="runanimate()" type = "submit" class="btn btn-outline-primary"
value = "Upload" />
</form>

</div>
<div>

<div class="img-container">
</div>
<div class="preloader">
<div class="circle circle1"></div>
<div class="circle circle2"></div>
<div class="circle circle3"></div>
</div>
</div>

<script>

function runanimate() {
var $circles = $('.circle'),
tl = new TimelineMax(),
random1 = getRandomNumber(),
imgUrl1 = 'https://s3-us-west-2.amazonaws.com/s.cdpn.io/409269/valley.jpg?' +
random1,
image1 = $('<img>');

function loaderOut() {
console.log('Image is done loading.');
}

function getRandomNumber() {
return Math.floor(Math.random() * 10000);
}

image1.on('load', loaderOut);
image1.attr('src', imgUrl1);

TweenMax.set($circles, {scale: 0});

tl.insert(
TweenMax.staggerTo($circles.toArray(), 1, {
opacity: 1,
scale: 1,
ease: Power1.easeIn
}, 0.2)
);

tl.insert(
TweenMax.staggerTo($circles.toArray(), 0.5, {
scale: 1.2,
boxShadow: '0 25px 25px rgba(0, 0, 0, 0.4)',
repeat: -1,
yoyo: true,
ease: Power1.easeOut
}, 0.2), '-=0.4'
);
}
```

```
</script>
{% endblock %}
</body></html>
```

10.3 detect.html

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
<style>
body {
padding: 0;
margin: 0;
background-color: #c3b3db;
font-family: 'Montserrat', sans-serif;
}</style>
</head>
<body>

<div style="width: 800px;height:500px;display: inline-flex;padding: 50px;">
<img style="width:12000px;margin-left:150px ;"
src="{{ url_for('send_uploaded_file', filename=uploaded_image) }}" />

</div>
<h3 style="color:rgb(0, 0, 0);margin-top:10px;text-align:center;">PCB is Faulty</h3>
</body>
</html>
```

10.4 eval.py

```
from __future__ import absolute_import
from __future__ import print_function
from __future__ import division

import os, sys
import tensorflow.compat.v1 as tf
tf.disable_v2_behavior()
import time
import cv2
import pickle
import numpy as np
sys.path.append("../")

from data.io.image_preprocess import short_side_resize_for_inference_data
from libs.configs import cfgs
from libs.networks import build_whole_network
from libs.val_libs import voc_eval
import argparse
from help_utils import tools

def eval_with_plac(det_net, real_test_imgname_list, img_root, draw_imgs=False): # 1. preprocess img
img_plac = tf.placeholder(dtype=tf.uint8, shape=[None, None, 3]) # is RGB. not BGR
img_batch = tf.cast(img_plac, tf.float32)
img_batch = short_side_resize_for_inference_data(img_tensor=img_batch,
target_shortside_len=cfgs.IMG_SHORT_SIDE_LE
N,
length_limitation=cfgs.IMG_MAX_LENGTH)
img_batch = img_batch - tf.constant(cfgs.PIXEL_MEAN)
img_batch = tf.expand_dims(img_batch, axis=0)

detection_boxes, detection_scores, detection_category =
det_net.build_whole_detection_network(
input_img_batch=img_batch,
gtboxes_batch=None)

init_op = tf.group(
tf.global_variables_initializer(),
tf.local_variables_initializer()
)

restorer, restore_ckpt = det_net.get_restorer()

config = tf.ConfigProto()
config.gpu_options.allow_growth = True

compute_time = 0

compute_imgnum = 0

with tf.Session(config=config) as sess:
sess.run(init_op)
if not restorer is None:
restorer.restore(sess, restore_ckpt)
print('restore model')

all_boxes = []
for i, a_img_name in enumerate(real_test_imgname_list): print((img_root+"/"+a_img_name))
raw_img = cv2.imread(img_root+"/"+a_img_name)
raw_h, raw_w = raw_img.shape[0], raw_img.shape[1]

start = time.time()
resized_img, detected_boxes, detected_scores, detected_categories = \ sess.run(
[img_batch, detection_boxes, detection_scores, detection_category], feed_dict={img_plac: raw_img[:, :, ::-
1]} # cv is BGR. But need RGB
)
end = time.time()
compute_time = compute_time + (end - start)
compute_imgnum = compute_imgnum + 1
print("{} cost time : {} ".format(a_img_name, (end - start))) print(draw_imgs)

if draw_imgs:
show_indices = detected_scores >= cfgs.SHOW_SCORE_THRSHOLD show_scores =
detected_scores[show_indices]
show_boxes = detected_boxes[show_indices]
show_categories = detected_categories[show_indices]
final_detections =
draw_box_in_img.draw_boxes_with_label_and_scores(np.squeeze(resized_img, 0),
boxes=show_boxes,
labels=show_categories,
scores=show_scores)
```

```
if not os.path.exists(cfgs.TEST_SAVE_PATH):
os.makedirs(cfgs.TEST_SAVE_PATH)

print(cfgs.TEST_SAVE_PATH + '/' + a_img_name + '.jpg') cv2.imwrite(cfgs.TEST_SAVE_PATH + '/' +
a_img_name +
'.jpg',final_detections[:, :, ::-1])

xmin, ymin, xmax, ymax = detected_boxes[:, 0], detected_boxes[:, 1], \
detected_boxes[:, 2], detected_boxes[:, 3]

resized_h, resized_w = resized_img.shape[1], resized_img.shape[2]

xmin = xmin * raw_w / resized_w
xmax = xmax * raw_w / resized_w
ymin = ymin * raw_h / resized_h
ymax = ymax * raw_h / resized_h

boxes = np.transpose(np.stack([xmin, ymin, xmax, ymax]))
dets = np.hstack((detected_categories.reshape(-1, 1),
detected_scores.reshape(-1, 1),
boxes))
all_boxes.append(dets)

tools.view_bar('{} image cost {}s'.format(a_img_name, (end - start)), i + 1,
len(real_test_imgname_list))

print('\n average_training_time_per_image is' + str(compute_time /
compute_imgnum))
return all_boxes

def eval(num_imgs, eval_dir, annotation_dir, showbox):
print(num_imgs, eval_dir, annotation_dir, showbox)

test_imgname_list = [item for item in os.listdir(eval_dir)
if item.endswith(('.jpg', 'jpeg', '.png', '.tif', '.tiff'))]
if num_imgs == np.inf:
real_test_imgname_list = test_imgname_list
else:
real_test_imgname_list = test_imgname_list[: num_imgs]

faster_rcnn =
build_whole_network.DetectionNetwork(base_network_name=cfgs.NET_NAME,is_t
raining=False)
all_boxes = eval_with_plac(det_net=faster_rcnn,
real_test_imgname_list=real_test_imgname_list,
img_root=eval_dir,
draw_imgs=showbox)

voc_eval.voc_evaluate_detections(all_boxes=all_boxes,
test_annotation_path=annotation_dir,
test_imgid_list=real_test_imgname_list)

def parse_args():

parser = argparse.ArgumentParser('evaluate the result with Pascal2007 stdand') parser.add_argument('--
eval_imgs', dest='eval_imgs',
help='evaluate imgs dir ',
default='../data/pcb_test/JPEGImages', type=str)
```

```
parser.add_argument('--annotation_dir', dest='test_annotation_dir',
help='the dir save annotations',
default='../data/pcb_test/Annotations', type=str)
parser.add_argument('--showbox', dest='showbox',
help='whether show detecion results when evaluation',
default=False, type=bool)
parser.add_argument('--GPU', dest='GPU',
help='gpu id',
default='2', type=str)
#parser.add_argument('--eval_num', dest='eval_num',
# help='the num of eval imgs',
# default=np.inf, type=int)
parser.add_argument('--eval_num', dest='eval_num',
help='the num of eval imgs',
default=100, type=int)
args = parser.parse_args()
return args

if __name__ == '__main__':

args = parse_args()
print(20*"--")
print(args)
print(20*"--")
eval(np.inf,eval_dir=args.eval_imgs,annotation_dir=args.test_annotation_dir,show
box=args.showbox)
```

10.5 voc_eval.py

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function
from asyncio.windows_events import NULL
from tkinter import image_names

import xml.etree.ElementTree as ET
import os,shutil
import pickle
from matplotlib import image
import numpy as np

import matplotlib.pyplot as plt
import pylab as pl
from sklearn.metrics import precision_recall_curve
from itertools import cycle

from libs.label_name_dict.label_dict import NAME_LABEL_MAP
from libs.configs import cfgs
from help_utils.tools import *
import cv2
import time
def write_voc_results_file(all_boxes, test_imgid_list, det_save_dir):

imagearray = []
imagenames = []
'''
```

```
:param all_boxes: is a list. each item reprensent the detections of a img.
the detections is a array. shape is [-1, 6]. [category, score, xmin, ymin, xmax, ymax] Note that: if none
detections in this img. that the detetions is : []

:param test_imgid_list:
:param det_save_path:
:return:
'''
for cls, cls_id in NAME_LABEL_MAP.items():
if cls == 'back_ground':
continue
print("Writing {} VOC resutls file".format(cls))

mkdir(det_save_dir)
det_save_path = os.path.join(det_save_dir, "det_"+cls+".txt")
with open(det_save_path, 'wt') as f:
print(test_imgid_list)

for index, img_name in enumerate(test_imgid_list):
path = 'D:/Tiny-Defect-Detection-for-PCB/static/'+img_name+".jpg"
savedir = "D:/Tiny-Defect-Detection-for-PCB/output/detectedimgs"
try:
if os.path.isfile(savedir) or os.path.islink(savedir):
os.unlink(savedir)
elif os.path.isdir(savedir):
shutil.rmtree(savedir)
except Exception as e:
print('Failed to delete %s. Reason: %s' % (savedir, e))
os.mkdir(savedir, 0o666)
print(path)
# Reading an image in default mode
global image
image = cv2.imread(path)

this_img_detections = all_boxes[index]

# font
font = cv2.FONT_HERSHEY_SIMPLEX

# fontScale
fontScale = 1

# Blue color in BGR
color = (254, 254, 254)

# Line thickness of 2 px
thickness = 2
this_cls_detections = this_img_detections[this_img_detections[:, 0]==cls_id]
if this_cls_detections.shape[0] == 0:
print("pcb is clear")
break # this cls has none detections in this img
for a_det in this_cls_detections:
print(float('{:.3f}'.format(a_det[1])))
if(float('{:.3f}'.format(a_det[1]))>0.3):
imagearray.append(a_det)
imagenames.append(cls)

f.write('{:s} {:.3f} {:.1f} {:.1f} {:.1f} {:.1f}\n'.
format(img_name, a_det[1],
a_det[2], a_det[3],
a_det[4], a_det[5])) # that is [img_name, score, xmin, ymin, xmax,
ymax]

for i, a_det in enumerate(imagearray):
cv2.putText(image, imagenames[i]+"="+str(float('{:.3f}'.format(a_det[1]))), (int(a_det[2])-5,
int(a_det[3])-5), font,fontScale, color, thickness, cv2.LINE_AA)
cv2.rectangle(image, (int(a_det[2]), int(a_det[3])), (int(a_det[4]), int(a_det[5])), (248, 2, 6), 7)
cv2.imwrite(savedir+'/'+img_name+".jpg", image)

# cv2.imshow(img_name, image)
```

10.6 cfgs.py

```
from __future__ import division, print_function, absolute_import
import os
import tensorflow as tf
# ------------------------------------------------
VERSION = 'FPN_Res101_0117_OHEM'
NET_NAME = 'resnet_v1_101'
ADD_BOX_IN_TENSORBOARD = True

# ---------------------------------------- System_config
ROOT_PATH = "D:/Tiny-Defect-Detection-for-PCB/"
print (20*"++--")
print (ROOT_PATH)
GPU_GROUP = "2"
SHOW_TRAIN_INFO_INTE = 10
SMRY_ITER = 100
SAVE_WEIGHTS_INTE = 10000

SUMMARY_PATH = ROOT_PATH + 'output/summary'
TEST_SAVE_PATH = ROOT_PATH + 'tools/test_result' INFERENCE_IMAGE_PATH = ROOT_PATH
+ 'tools/demos_backup' INFERENCE_SAVE_PATH = ROOT_PATH + 'tools/inference_results'

if NET_NAME.startswith("resnet"):
weights_name = NET_NAME
else:
raise NotImplementedError

PRETRAINED_CKPT = ROOT_PATH + 'data/pretrained_weights/' + weights_name + '.ckpt'
TRAINED_CKPT = os.path.join(ROOT_PATH, 'output/trained_weights')

EVALUATE_DIR = ROOT_PATH + 'output/evaluate_result_pickle/' #test_annotate_path =
'/home/yjr/DataSet/VOC/VOC_test/VOC2007/Annotations'
test_annotate_path = 'D:/Tiny-Defect-Detection-for-
PCB/PCB_DATASET/Annotations/'

# ------------------------------------------ Train config
RESTORE_FROM_RPN = False
IS_FILTER_OUTSIDE_BOXES = False
FIXED_BLOCKS = 0 # allow 0~3
USE_07_METRIC = False

RPN_LOCATION_LOSS_WEIGHT = 1.
RPN_CLASSIFICATION_LOSS_WEIGHT = 1.0
```

```
FAST_RCNN_LOCATION_LOSS_WEIGHT = 1.0
FAST_RCNN_CLASSIFICATION_LOSS_WEIGHT = 1.0
RPN_SIGMA = 3.0
FASTRCNN_SIGMA = 1.0

MUTILPY_BIAS_GRADIENT = None # 2.0 # if None, will not multipy
GRADIENT_CLIPPING_BY_NORM = None # 10.0 if None, will not clip

EPSILON = 1e-5
MOMENTUM = 0.9
LR = 0.001 # 0.001 # 0.0003
#DECAY_STEP = [60000, 80000] # 50000, 70000
DECAY_STEP = [10000, 20000] # 50000, 70000
#MAX_ITERATION = 150000
MAX_ITERATION = 30000

# ------------------------------------------- Data_preprocess_config
DATASET_NAME = 'pcb' # 'ship', 'spacenet', 'pascal', 'coco'
# PIXEL_MEAN = [123.68, 116.779, 103.939] # R, G, B. In tf, channel is RGB. In openCV, channel is BGR
PIXEL_MEAN = [21.25, 85.936, 28.729]
IMG_SHORT_SIDE_LEN = 600 # 600
IMG_MAX_LENGTH = 3000 # 1000
CLASS_NUM = 6
# -------------------------------------------Fast-RCNN config
ROI_SIZE = 14
ROI_POOL_KERNEL_SIZE = 2
#USE_DROPOUT = False
USE_DROPOUT = True
KEEP_PROB = 1.0
SHOW_SCORE_THRSHOLD = 0.6 # only show in tensorboard

#FAST_RCNN_NMS_IOU_THRESHOLD = 0.3 # 0.6 FAST_RCNN_NMS_IOU_THRESHOLD = 0.3
FAST_RCNN_NMS_MAX_BOXES_PER_CLASS = 100
FAST_RCNN_IOU_POSITIVE_THRESHOLD = 0.5 FAST_RCNN_IOU_NEGATIVE_THRESHOLD = 0.0 # 0.1 < IOU < 0.5 is negative
FAST_RCNN_MINIBATCH_SIZE = 256 # if is -1, that is train with OHEM
# FAST_RCNN_MINIBATCH_SIZE = -1
FAST_RCNN_POSITIVE_RATE = 0.25

ADD_GTBOXES_TO_TRAIN = True
```