Guide AnoPCB

Contents

1 Purpose and Motivation

Designing PCBs requires experience and knowledge about bad designs, e.g. sensitive signals being influenced aggressively switching ones in close proximity. Those interactions cannot be detected by formal checks.

AnoPCB is a plugin for the PCB design tool KiCad that uses machine learning to detect anomalies in PCB layouts. The plugin is written in python and uses the Tensorflow framework for its machine learning aspects. The architecture is split into two parts: A plugin that is integrated into the GUI of KiCad Pcbnew editor and a server. The plugin is used to extract information from the PCB layout and to instruct the server. The server houses the machine learning models and can be communicated with via the http protocol.

AnoPCB utilizes an Autoencoder-like machine learning architecture which has to be trained on "good" PCB layouts before it can be used to detect anomalies in other PCBs. The Autoencoder can reconstruct configurations it has learned during training. Unusual or rare configurations will therefore be recognized as anomalies.

AnoPCB is meant to be used as a scientific tool. To use it effectively the user must have an understanding of PCB design to choose "good" PCB layouts for training, to categorize their signals and to interpret the plugins results. AnoPCB can by no means replace an experienced designer.

2 Installation

2.1 Plugin

AnoPCB only works on the Linux version of KiCad. To install the plugin, either install the plugin manually by moving the plugin project folder (source code and resources) into the the directory ~/kicad/scripting/plugins/ or install anopcb_x.x.x.deb via apt.

2.2 Server

After downloading the server, it can be started from any directory, no installation is required. The wheel file can also be installed with pip.

Another way to install the server is to use the integrated docker support. In this case, please ensure that docker is installed and running on your system (please mind how to install docker fully on your distribution). The plugin can then start the server locally automatically via docker. For further information look into Server Management.

3 General Workflow

After plugin and server have been installed successfully KiCad can be started. For the plugin to work a project containing a kicad_pcb layout file has to be opened. The plugin can be accessed after opening said file with the Pcbnew editor. If the installation has been successful the plugin's icon can be found in the menu-bar, the plugin is started by clicking on it.

First the correct settings and preferences should be chosen (check out Settings for reference).

Now the PCB layout can be annotated with signal information. Each net should be classified (check out Annotating PCB Layouts for reference). Annotation is required for evaluation, as well as training data generation.

To progress further a Tensorflow machine learning model is needed that can find anomalies in PCB layouts. When analyzing the layout it is cut up into vertical 2D slices which are fed into the network. The slice dimensions can be changed in "preferences" and must match the models input (check out creating a Tensorflow Model for reference). For the next steps a running server is required (check out Server Management for reference).

You can either use a pretrained model matching your needs or use a template model and train it yourself.

After the model has been sent to the server it can be set as the active model using the “model configuration” dialog by selecting the model and pressing the “serve model” button.

Now the model can be trained using the “train model” dialog. First a dataset for training and validation must be created from an annotated PCB layout which only requires two button presses (check out creating datasets for reference). To train the model on multiple datasets multiple PCB layouts have to be annotated. After the model has been trained it can be evaluated on one or multiple datasets using the “Test model” button. For further reference on training check out Training a Tensorflow Model.

Now the model can be used to search annotated PCB layouts for anomalous configurations. Pressing the “analyze board” button will analyze the PCB currently opened in Pcbnew. Note that interpreting the results is not at all trivial and hugely depends on how the model was trained (check out Analyzing a PCB Layout for reference).

4 Settings

Settings can be changed in the preferences menu. There are several settings:

Setting	Explanation
Server address/port	The IP and port the plugin should try to connect to
Slice X/Y	The dimensions of the slices to be fed into the machine learning algorithm.
Server type	Whether the server uses a remote address or is started locally by the plugin.
Save filter data locally	Whether analysis results should be saved.
Signals 1-8	The mapping between signal numbers and names.

5 Annotating PCB Layouts

5.1 Annotation conventions

Each net in a given PCB layout has to be categorized (annotated). By default there are 8 categories. Each category responds to a number from 1 to including 8, by default. Which signal type corresponds to which number can be chosen freely, except for the “no signal” category which always corresponds to 0. The numbers can be mapped to names in the settings. Note that the machine learning model only learns the numbers, not their meaning. A model trained with a different mapping between numbers and categories will not produce useful results.

For our models we use the following mapping:

Category	Number
Digital stable	1
Digital switching	2
Analog	3
Analog Sensitive	4
Supply	5
Unknown	8

5.2 Direct Annotation

Components can be selected in Pcbnew and manually annotated with one of the categories. Multiple Nets can be selected and annotated at once.

You can also lookup and change annotations in the netlist dialog inside the “regex” dialog.

5.3 Regular Expressions

Nets can be annotated by using pattern matching via the “regex” dialog. The dialog allows the user to create a list of [name, signal] tuples, tuples are added via the “add” button. Upon pressing the “annotate” button the names are matched against all available net-names and the nets are annotated with the corresponding signals. The “netlist” button shows a list of all nets and their annotations.

NOTE that this does NOT support regular expressions. Instead, the algorithm just simply checks, whether the name is a prefix, suffix or infix.

6 Creating a Tensorflow Model

6.1 Architecture

The Server expects a Tensorflow model with the following input shape: (slice width, slice height, number of signal types). The slice width can be chosen at will (our default value is 28). The slice height should match the number of layers of the PCBs to be analyzed. The number of signal types is 8 by default, this number can be changed by altering the server-code (check out Server parameters for reference). The input to the model is a tensor of slices taken through the PCBs layers, the 3rd dimension of the input encodes the signal type of any sliced component or track as a one-hot vector. Note that “no signal” or empty space does not have its own class, it is represented by an empty vector. The server expects an Autoencoder-like architecture with 3 outputs of following shapes:

Reconstructed: (slice width, slice height, number of signal types)

Latent: (latent size)

Error: scalar

The outputs have to follow exactly this order. "Reconstructed" is the output of the Autoencoder, it is the models attempt at reconstructing the input after routing it through a bottleneck. "Latent" is the output of said bottleneck layer at the heart of the Autoencoder. It is used to cluster the outputs. "Error" is an error computed between the input and "Reconstructed" for each individual slice. The metric used to compute this error can be chosen freely, we use MSE and slight variations of it but we suspect there are much better metrics for the job. The Error is used to determine how anomalous a given slice is. If the model has not seen similar slices before the error will be high, which indicates an anomalous configuration.

6.2 Loss and Metrics

If the model is compiled and saved as an h5-file before sending it to the server it requires the following compile parameters: loss = [loss-function, None, None] and loss_weights = [1.0, 0.0, 0.0], because we only want to train on the "Reconstructed" output. We recommend a sigmoid activation for the last Autoencoder output and binary cross-entropy as the loss-function. The models outputs resemble one-hot encoded vectors but since the “no-signal” category corresponds to a zero-vector a softmax activation cannot be utilized.

If the model was send to the server via the "send model" function in the "model configuration" GUI as an uncompiled json-config those parameters will be applied automatically. An example for a working ML-model is given in the Autoencoder.py file in the Server directory.

6.3 Sending models to the server

A model can be sent to the server via the AnoPCB Interface. The “send model” function can be accessed from the “model configuration” dialog. If the model has been saved as an uncompiled json string (via Tensorflow’s to_json function) an optimizer and loss function can be chosen and will be applied correctly. If the model has been saved as an h5-file those entries will be ignored and the model will not be modified.

7 Server Management

There are two ways to start the server: Manually or via Docker. To start the server manually all the required python modules must be installed. The list of modules can be found in the pip_requirements.txt file. The Server can be started from a terminal in the server’s directory with the following commands:

- “python AnomalyServer.py PORT”
- “python AnomalyServer.py IP-ADDRESS PORT”

Substituting PORT for the port the server is supposed to use for communication (we use 8420). Substituting IP-ADDRESS for the ip-address the server is supposed to use for communication (standard being 0.0.0.0).

If installed via pip, you can type “anopcb-server” instead of “python AnomalyServer.py.

You can append “--local”, to save the server files in the current working directory instead of ~/.anopcb/ on linux or %ROAMING% on windows.

To connect to the server its port and the IP-address of the device it is running on have to be entered in the plugin settings. Now the plugin can communicate with the server.

To start the server via docker, choose either ‘CPU’ or ‘GPU’ in the preferences of AnoPCB. This will ensure, that a docker server instance is started on demand. ‘CPU’ and ‘GPU’ stand for the variant of tensorflow to be use, meaning that tensorflow will either use the CPU or the GPU for its processing.

To update the server via docker, head to the preferences and select ‘update docker server’ in the menu bar.

Please note that the server does not use a secure connection and should only be exposed to safe local networks.

8 Training a Tensorflow Model

8.1 Creating datasets

A dataset corresponding to the current PCB layout can be created from the “train model” dialog by pressing the “Send slices” button. The PCB’s layout will be turned into slices that can be used by the machine learning model and will be sent to the server. To remove duplicate slices and mirror the remaining ones the “Augment” button can be pressed (recommended). The dataset’s names have the following pattern: PCB-name_count_width_height_a. The _a only shows up if the dataset has been augmented. Datasets can also be exported and saved locally via the preferences “export” menu. Those can be imported into the server via the training “import” menu.

8.2 Automatic Train function

The model can be trained from the “train model” dialog on datasets that have been sent to the server. The datasets used for training and validation during training can be chosen in the corresponding lists. The two sets have to be the same or disjoint. The model is trained on a random batch from the chosen datasets of the chosen batch size. If datasets for training and validation are the same they will be split. The following only applies in that case: If the combined batch sizes for training and validation are larger than the size of the datasets the dataset for validation will be made smaller. If the batch size for training alone is larger than the size of the datasets no validation will be performed. If datasets for training and validation are disjoint and a batch size is larger than the datasets the entire dataset will be used, which is recommended. The model will train for the chosen number of epochs. One epoch means training on 5000 samples, then validating. If the validation loss rises for 5 epochs in succession, the training is stopped (overfitting). Otherwise the training will be stopped after the chosen amount of minutes or epochs (whichever is reached first).

8.3 Server parameters

Some server parameters can be changed easily by editing the AnomalyServer.py file.

The number of different signal types expected by the server can be changed from the default of 8 by changing the variable NR_CHANNELS = 8 (line 19). Note that the architecture of the employed Tensorflow model has to match that number.

The number of samples used for training per epoch can be changed from the default of 5000 by changing the variable SAMPLES_PER_BATCH = 5000 (line 20).

The number of epochs until the training process is stopped due to overfitting can be changed from the default value of 5 by changing the variable PATIENCE_MAX = 5 (line 21).

9 Analyzing a PCB Layout

An annotated PCB layout can be analyzed by a properly trained model. The model has to be set as active model using the “configure model” dialog and can be analyzed by pressing the “analyze board” button. After the analysis is done (which may take some time) two results are shown: A histogram showing how the reconstruction errors are distributed, and a dialog showing a top-down view of the layout. The results can be clustered and then displayed. Since every slice creates some error it is recommended to only display those with a comparatively high one by setting the threshold according to the histogram. Note that everything the machine learning model has not seen before or only seen rarely will produce a high reconstruction error. This means that not every anomaly is a design flaw, some will be neutral or even good, novel designs the model has not seen before. If the corresponding checkbox in the plugin’s settings is ticked on, the latest analysis results will be saved locally. Additionally screenshots of the layout and the anomalies can be taken.

10 Data Management

The server stores its available datasets and machine learning models in either fixed paths, in the current working directory or a docker volume.

For windows, look into “%APPDATA%\.anopcb\”.
For linux, look into “~/.anopcb/”.

The machine learning models are stored as h5-files in the “models” directory and the datasets are stored as json-files in the “datasets” directory. Files can be removed manually from both directories

without causing problems. Adding files manually is only recommended for machine learning models that follow the correct format (check out creating a Tensorflow model for reference).

The plugins data is unique for each KiCad project and the corresponding files are stored in the project folder. The plugin saves the net annotations, preferences and regexes whenever the project is saved manually. Those files can be edited manually but it is not recommended. If the corresponding checkbox in the plugins settings is ticked, the plugin will also save the PCB analysis results.

NOTE that KiCad should be restarted before starting a second pcb project and using the anopcb plugin.

11 Disclaimer

The server employs no security features and should only be exposed to trusted networks and users. The predictions made by the machine learning model are in no way guaranteed to be correct.

12 Authors

Students at TU-Ilmenau: Henning Franke, Julian Kuners, Paul Kucera, Murad Babayev and Nicholas Heyer.

Researchers at IMMS and TU-Ilmenau: Georg Gläser, Marco Seeland, Martin Grabmann, Tom Reinhold and Patrick Mäder.