Ultra-Fast Power Module Inductance Estimation using Convolutional Neural Networks

Kubulus, Pawel Piotr; Beczkowski, Szymon Michal; Munk-Nielsen, Stig; Jørgensen, Asger Bjørn

Published in:
2023 IEEE Energy Conversion Congress and Exposition, ECCE 2023

DOI (link to publication from Publisher):
10.1109/ECCE53617.2023.10361953

Publication date:
2023

Document Version
Accepted author manuscript, peer reviewed version

Ultra-Fast Power Module Inductance Estimation using Convolutional Neural Networks

1st Pawel Piotr Kubulus
AAU Energy
Aalborg University
Aalborg, Denmark
ppk@energy.aau.dk

2nd Szymon Michal Beczkowski
AAU Energy
Aalborg University
Aalborg, Denmark
sbe@energy.aau.dk

3rd Stig Munk-Nielsen
AAU Energy
Aalborg University
Aalborg, Denmark
smn@energy.aau.dk

4th Asger Bjorn Jorgensen
AAU Energy
Aalborg University
Aalborg, Denmark
abj@energy.aau.dk

***Abstract*—The widespread usage of wide bandgap (WBG) semiconductors forces extra emphasis on the early estimation of the layout parasitic elements. Be it a printed circuit board or a power module, layout optimization is necessary to minimize the negative effects of present inductances. Unfortunately, multiple invocations of inductance extraction software can be time-consuming. In this work, state-of-the-art convolutional neural networks (CNN) are applied in order to lower the time consumption of inductance estimation without compromising the accuracy.**

***Index Terms*—inductance extraction, power module, printed circuit board, neural networks, layout optimization**

I. Introduction

With the recent growth in wide bandgap (WBG) device applications, a need for fast and accurate assessment of layout parasitic elements is becoming evident. The parasitic inductance is one of the most of important factors limiting the switching speed, significantly contributing to the final EMI and potentially leading to instability or failure of paralleled switches. For this reason, accurate knowledge of the parasitic inductances during the design process is necessary. The accurate parasitic inductance extraction is normally performed numerically, using the method of moments (MoM) based software such as Ansys Q3D, or finite element method (FEM) like Ansys Maxwell or COMSOL. Aside from specific very high-frequency cases, those methods give a fairly accurate estimation leading to the widespread usage in manual design. However, the above-mentioned methods tend to be computationally costly when multiple consecutive inductance extractions are to be performed, what limits their usability for layout optimization. For optimization where the computational time of the parasitic inductance extraction is a crucial factor, more commonly used methods are to utilize lumped element circuit methodologies, such as with analytical microstrip equations available in PowerSynth [1] with some accuracy loss, or crude mesh analysis present in FastHenry [2]. Recent developments brought FFT-accelerated partial element equivalent circuit (PEEC) [3] and volume integral equation (VIE) [4] methods, achieving a significant reduction in computation time while maintaining good accuracy. An even higher improvement has been achieved using the Current Bunch concept and loop-based method at the cost of accuracy [5]. As power module layouts are normally constrained by the fixed packaging dimensions, such as two-layer ceramic with copper traces on one side only, and can be approximately treated as 2.5D structures, makes them a good candidate for a fundamentally different approach to parasitic inductance extraction with the help of machine learning. Machine learning has been applied to some well-defined inductance estimation tasks, such as microstrips [6], coils [7] or parametrized bondwires [8]. However, it has not been yet applied to inductance estimation problems where geometry can significantly vary and is not easy to describe with a few simple parameters.

In this paper, the viability of using deep neural networks (DNN) for fast and accurate parasitic inductance estimation is investigated for power module 2D layouts, to reduce the computational time, thus unlocking the potential of improved optimization of such layouts.

II. Data Generation

In order to use neural networks for inductance estimation, a labeled dataset for DNN training has to be prepared. Based on those data, weights between neurons are to be optimized until the desired behavior is reached, in the approach also known as supervised learning. Data generation is an important step in procuring highly accurate neural networks. Both quantity and quality are crucial factors, as a rule of thumb the more data are available the more accurate estimation can be achieved. For this reason, simulation data are strongly preferred, as experimental data preparation would require thousands of unique trace pairs to be procured. The full data generation scheme has been depicted in the figure below.

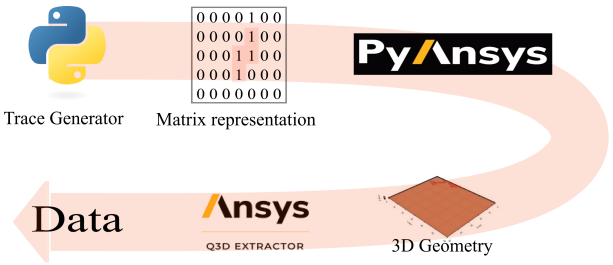

Fig. 1. Data generation scheme

As most of the literature compares the accuracy and speed of parasitic inductance extraction methods to Ansys Q3D, it has been chosen for training data generation. The generation has been automated using the PyAEDT package, offering a link between Ansys Q3D and trace shape generation in Python. The traces are represented as 2D matrices, where ones and zeros represent copper square and clearance respectably. Sinks and sources are placed at the ends of the traces, as the final network will assume a source at the end of the trace closer to the substrate edge and a sink on the other side. Due to this approach, mutual inductance sign is also implied. The randomized trace pairs are generated using a python script, based on a maze solver and parameters described in the table below.

TABLE I
PARAMETERS FOR DATA GENERATION

Parameter	Value	Unit
Substrate dimensions	50x60	mm
Trace width	2	mm
Trace height	0.3	mm
Maximum trace length	40	mm
Minimum trace length	4	mm
Trace material	Copper	-
Substrate material	Al2O3	-

The generated self-inductance data have been depicted in Fig. 2.

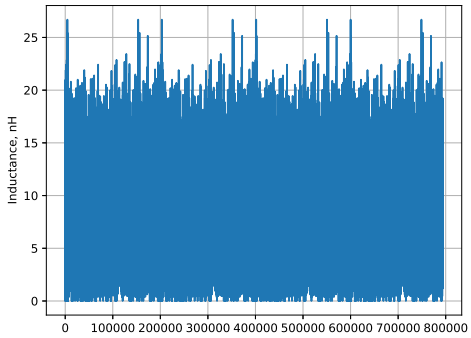

Fig. 2. Generated self-inductance data

There are no significant outliers in the raw data, and the range between 0 and 20nH is well populated. This is a desirable case for machine learning as the network trained on a set well-covering the intended application range, is expected to achieve higher accuracy. The generated mutual inductance data have been depicted in Fig. 3.

In this case, significant outliers are present and the coverage is much worse. For this reason, a worse performance is expected, compared to the self-inductance. The outliers have not been removed, due to them representing cases of high coupling which are essential to capture.

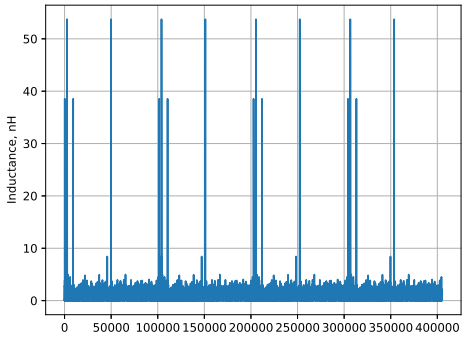

Fig. 3. Generated mutual inductance data

III. TOPOLOGY SELECTION

Convolutional neural networks (CNN) are DNN known for good performance when applied to pattern recognition-related problems [9]. In order to do so, the 2D matrices representing the DBC structure are scanned using multiple filters to capture present features in a process called convolution. Next, max pooling downsamples the feature maps. Depending on the architecture, after one or more of those cycles the dimension-reducing flatten operation is performed to make it a viable input for dense layers. The fully-connected dense layers further process the acquired features into the inductance estimations. The described process has been depicted in the Fig 4.

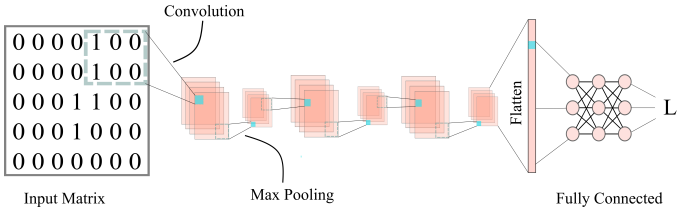

Fig. 4. CNN structure and principle of working

One of the main struggles of basic CNN was that training difficulty increased with the number of layers. As the training process is essentially the optimization of weights in the network structure by error backpropagation, bigger models suffered from decreased accuracy due to the optimization problem size. The available optimizers increasingly struggle in minimizing the error in basic CNN structure with added number of layers [10]. In order to solve this problem, the residual network (ResNet) has been developed [10]. The reason for the optimization algorithms performance decrease has been identified as a struggle with providing identity mapping [10]. The solution was to precondition the problem by changing the CNN structure, introducing shortcut connections passing the residual to the deeper layers [10]. The authors proposed using so-called residual blocks for

building neural networks based on this approach. A simplified ResNet structure with two residual blocks has been depicted in the figure below.

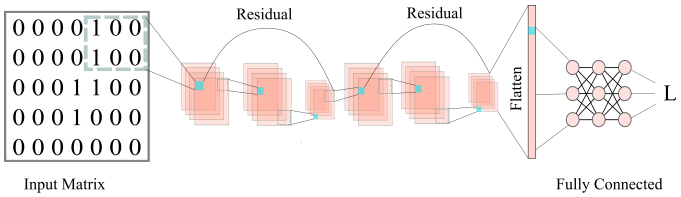

Fig. 5. ResNet structure and principle of working

Further improvements have been brought by the introduction of DenseNet architecture, depicted in Fig. 6. The architecture introduces so-called dense blocks, providing each layer with a connection to all the preceding ones by concatenating the features from preceding layers with input to the current layer [11]. This solution provides several benefits. Firstly, contrary to ResNets, the information passed is not potentially constrained by summation with the preceding features. Secondly, by having access to information form all of the preceding layers, feature levels are more diversified [11]. In basic CNN or ResNet, increased depth forces final levels to rely on higher-level features created by passing through multiple convolutions as it is the only available information. In DenseNet, information from all of the feature levels is available allowing for mixing of high- and low-level features accordingly to the needs [11]. Finally, in basic CNN and ResNet architectures, different layers tend to contribute minimally or learn redundant features. The availability of all preceding features in DenseNets reduces the tendency to redundancy, reducing the overall number of network parameters necessary [11].

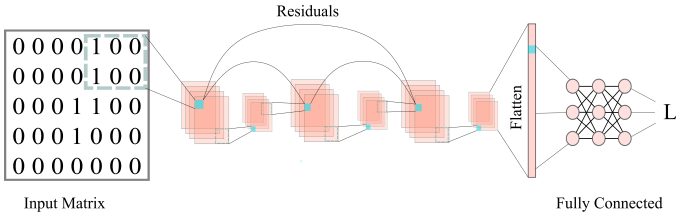

Fig. 6. DenseNet structure and principle of working

The three above-mentioned types of CNN architectures have been implemented in Keras and PyTorch Lightning and explored in order to find the best-performing networks. In this work, self and mutual inductance estimation are executed using two separate networks, to allow for customizing their hyperparameters separately. The network hyperparameters can include number of layers, filter dimensions, the number of neurons in fully connected layers etc. Architecture parameters and the considered range have been described in the table below.

TABLE II
PARAMETER RANGES FOR CONSIDERED ARCHITECTURES

Basic CNN	**Min**	**Max**
Number of convolutions	1	5
Kernel size	2x2	5x5
Number of fully connected layers	3	10
Size of fully connected layers	50	2000
ResNet	**Min**	**Max**
Number of Conv2 layers	2	3
Number of Conv3 layers	2	4
Number of Conv4 layers	2	23
Number of Conv5 layers	2	3
DesneNet	**Min**	**Max**
Number of blocks	3	6
Number of layers in a block	6	6
Growth rate	6	12

IV. TRAINING AND RESULTS

Training hyperparameters on the other hand control the process of weight optimization by selecting the optimizer type, learning rate, weight decay, batch size and the number of epochs signifying the number of repetitions of the same data during the training. Training parameters and their investigated ranges for each architecture type are presented in the table below.

TABLE III
TRAINING PARAMETER RANGES

Basic CNN	**Min**	**Max**
Optimizer type	Adam	SGD
Learning rate	0.00001	0.5
Weight decay	0.001	0.01
Batch size	50	5000
ResNet	**Max**	**Min**
Optimizer types	SGD	-
Learning rate	2	4
Weight decay	2	23
Batch size	2	3
DenseNet	**Max**	**Min**
Optimizer types	Adam	SGD
Learning rate	2	4
Weight decay	2	23
Batch size	2	3

The models were trained using the Huber loss function and the absolute mean error has been used as a metric. The training results has been depicted in the figure below.

V. FINAL VALIDATION AND ERROR ANALYSIS

For additional validation, a hundred new random trace pairs are generated and evaluated using ANSYS Q3D and trained networks prepared. The evaluation has been performed on

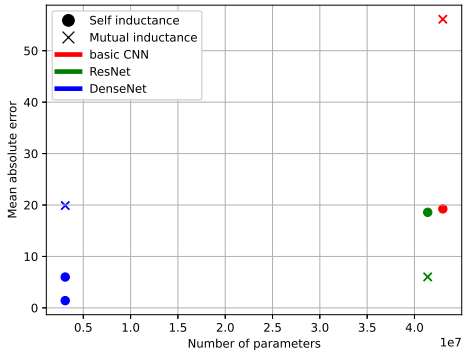

Fig. 7. Training results for different networks considered

a Lenovo Thinkpad equipped with Intel(R) Core(TM) i5-10210 and 64GB RAM. The execution times for models were measured using Python time.time() function on a batch input model and divided by the batch size, as a significant acceleration was noticed when evaluating multiple layouts in once batch. The summary of validation results is presented in Table IV. Full result are available in the GitHub repository [13].

TABLE IV
RESULT OVERVIEW

Method	Evaluation time	Acceleration	Reported error
Q3D	120s	-	-
Loop-based [12]	-	7x	5.5%
Basic CNN Keras	0.004s	3000x	8%
Basic CNN PyTorch	0.00032s	375000x	20%
Best accuracy (Resnet + DenseNet)	0.00064s	187500x	1.4/6%

As can be seen in Table IV, an acceleration of inductance estimation has been achieved compared to the methods available in the literature.

VI. CONCLUSION

In this work, the viability of DNN for trace inductance estimation has been verified, achieving at least 3000 times speedup compared to the commercial software, and 400 times compared to the state-of-the-art works. This paves the way towards ultra-fast layout optimization of power modules, at the cost of a slight accuracy decrease. This decrease can potentially be solved by architecture refinement and an increase in the training dataset size. More refined architectures will be presented in the full version of this paper, along with detailed error analysis.

REFERENCES

[1] Q. Le, T. Evans, S. Mukherjee, Y. Peng, T. Vrotsos and H. A. Mantooth, "Response surface modeling for parasitic extraction for multi-objective optimization of multi-chip power modules (MCPMs)," 2017 IEEE 5th Workshop on Wide Bandgap Power Devices and Applications (WiPDA), Albuquerque, NM, USA, 2017, pp. 327-334, doi: 10.1109/WiPDA.2017.8170568.

[2] Q. Le, I. Al Razi, Y. Peng and H. A. Mantooth, "Fast and Accurate Inductance Extraction for Power Module Layout Optimization Using Loop-Based Method," 2021 IEEE Energy Conversion Congress and Exposition (ECCE), Vancouver, BC, Canada, 2021, pp. 1358-1365, doi: 10.1109/ECCE47101.2021.9595620.

[3] N. Marconato and F. Lucchini, "Application of FFT-PEEC Method for Nonlinear Inductance Extraction," 2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME), Mauritius, Mauritius, 2021, pp. 1-6, doi: 10.1109/ICECCME52200.2021.9590864.

[4] A. C. Yucel, I. P. Georgakis, A. G. Polimeridis, H. Bağcı and J. K. White, "VoxHenry: FFT-Accelerated Inductance Extraction for Voxelized Geometries," in IEEE Transactions on Microwave Theory and Techniques, vol. 66, no. 4, pp. 1723-1735, April 2018, doi: 10.1109/TMTT.2017.2785842.

[5] L. Wang, Z. Zeng, Y. Yu, K. Ou and J. Wang, "Current-Bunch: A Fast and Accurate Tool to Extract and Optimize Parasitics of Power Packaging," 2020 IEEE Energy Conversion Congress and Exposition (ECCE), Detroit, MI, USA, 2020, pp. 3171-3177, doi: 10.1109/ECCE44975.2020.9236262.

[6] S. Newberry and A. Zadehgol, "A Deep Neural Network Modeling Methodology for Extraction of RLGC Parameters in μ-wave and mm-wave Transmission Lines," 2022 IEEE International Symposium on Electromagnetic Compatibility and Signal/Power Integrity (EMCSI), Spokane, WA, USA, 2022, pp. 74-79, doi: 10.1109/EMCSI39492.2022.9889553.

[7] J. Stillig, N. Parspour, D. Ewert and T. J. Jung, "Feasibility Study on Machine Learning-based Method for Determining Self-and Mutual Inductance," 2022 International Seminar on Intelligent Technology and Its Applications (ISITIA), Surabaya, Indonesia, 2022, pp. 193-198, doi: 10.1109/ISITIA56226.2022.9855321.

[8] Q. Liu, Z. Shao, Y. Zhang and J. Mao, "A Fast and Accurate Method for Bond Wires Inductances Extraction Based on Machine Learning Strategy," 2020 IEEE MTT-S International Microwave Workshop Series on Advanced Materials and Processes for RF and THz Applications (IMWS-AMP), Suzhou, China, 2020, pp. 1-3, doi: 10.1109/IMWS-AMP49156.2020.9199770.

[9] D. Dai, "An Introduction of CNN: Models and Training on Neural Network Models," 2021 International Conference on Big Data, Artificial Intelligence and Risk Management (ICBAR), Shanghai, China, 2021, pp. 135-138, doi: 10.1109/ICBAR55169.2021.00037.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

[11] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708

[12] Q. Le, I. A. Razi, T. M. Evans, S. Mukherjee, Y. Peng and H. Alan Mantooth, "Fast and Accurate Parasitic Extraction in Multichip Power Module Design Automation Considering Eddy-Current Losses," in IEEE Journal of Emerging and Selected Topics in Power Electronics, doi: 10.1109/JESTPE.2022.3175150.

[13] https://github.com/AAU-CoDE/ML-Inductance-extractor