Adv. Radio Sci., 21, 77–87, 2023
https://doi.org/10.5194/ars-21-77-2023

Advances in Radio Science
Open Access Proceedings

AI Models for Supporting SI Analysis on PCB Net Structures: Comparing Linear and Non-Linear Data Sources

Julian Withöft[1], Werner John[2], Emre Ecik[1], Ralf Brüning[3], and Jürgen Götze[1]

[1]TU Dortmund/Information Processing Lab, Otto-Hahn-Straße 4, 44227 Dortmund, Germany
[2]Pyramide 2525, Doerener Weg 4B, 33100 Paderborn, Germany
[3]EMC Technology Center/Zuken GmbH, Am Hoppenhof 30, 33104 Paderborn, Germany

Correspondence: Julian Withöft (julian.withoeft@tu-dortmund.de)

Received: 15 February 2023 – Accepted: 4 September 2023 – Published: 1 December 2023

Abstract. Signal integrity (SI) is an essential part in assuring the functionality of microelectronic components on a printed circuit board (PCB). Depending on the complexity of the designed interconnect structure, even the experienced PCB developer might be reliant on multiple design cycles to optimally configure the PCB parameters, which eventually results in a very complex, time-consuming and costly process. Under these aggravating conditions, artificial intelligence (AI) models may have the potential to support and simplify the SI-aware PCB design process by building predictive models and proposing design solutions to streamline the existing workflows and unburden the PCB designer. In this paper, the AI approach is divided into two separate stages consisting of neural network (NN) regression in the first step and parameterization of the PCB net structure in the second step. First, the NN models are applied to learn the relationship between the electrical parameters and the resulting signal quality captured by domain-oriented signal features in the time domain. Second, based on the trained NN models, on the one hand, the k-nearest neighbor (kNN) method is utilized to select solution candidates within the feature space, while on the other hand, genetic algorithms (GA) are applied to directly optimize the parameters of the interconnect structure. Moreover, the influence of the simulation abstraction level is investigated by comparing simulation data originating from linear and I/O buffer information specification (IBIS)-based non-linear modeling of the integrated circuit (IC) characteristics concerning the prediction accuracy and direct transferability. Finally, transfer learning concepts are evaluated to exchange learned knowledge representations between the different modeling of the IC characteristics to improve data efficiency and reduce computational complexity.

1 Introduction

The various stages of the design of printed circuit boards (PCB) as well as system integration and handling of physical couplings are addressed by the different engineers involved. In this context, ensuring signal integrity (SI) is an integral part to comply with functional and electromagnetic compatibility (EMC) design constraints at PCB level. For designers who have little to no experience with SI trying to fulfill these design constraints is challenging and error-prone.

Artificial intelligence (AI) models may serve as a possible reference during the PCB design process for designers working on the selection of the routing structure in the pre-layout phase. These models are intended to assist them in specifying signal topologies and routing structures that will affect the SI of the system. For the realization of AI models, an extension has to be made with respect to AI-specific data sources and objects based on a generalized design process for electronic systems. In the future, AI-supported pre-layout measures will be used in the specification and design phases. For this purpose, e.g. for the development of an SI concept for upcoming design tasks, the generalized PCB design process presented in John et al. (2022) and the extended combined process and phase model presented in Ecik et al. (2023) are essential. The generalized approach for the integration of AI models into the PCB design process is illustrated in Fig. 1 demonstrating how different AI models might be utilized and combined to support SI-compliant design decisions. Therefore, the integration of domain knowledge from experienced PCB developers is of utmost importance to detect adequate design solutions, support the unexperienced developer and effectively reduce the number of design cycles.

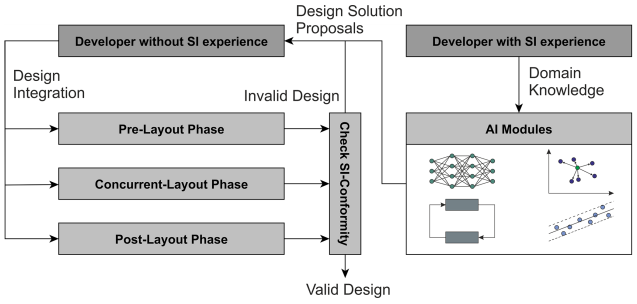

Figure 1. Illustration of the generalized procedure to integrate AI-based SI design support into the PCB design process with consideration of domain knowledge from the experienced SI developer.

Recently, AI or in particular methods from the machine learning (ML) subdomain have received growing attention in the realm of electronic design automation (EDA) and PCB design under SI constraints. Most prominently the prediction and optimization of eye diagram parameters such as eye height and eye width from parameters of the transmitter, interconnect and receiver has been a main research focus. In Lu et al. (2018) support vector regression (SVR) and neural network (NN) ML methods were utilized to predict the eye diagram parameters from the specified PCB parameters. Originating from this initial publication several improvements with respect to the ML implementations have been achieved. With respect to the SVR, an active-subspace method has been proposed in Ma et al. (2020a, b) to improve the algorithmic performance. Regarding the NN, methods to improve the data efficiency such as transfer learning in Zhang et al. (2019) and semi-supervised learning in Chen et al. (2020) have been deployed.

The inverse problem setting, which initializes with the target eye performance to find the required electrical parameters, was firstly discussed in the prospect on future work section of Lu et al. (2018). In Trinchero et al. (2019) and Roy et al. (2019) this inverse design was addressed similarly to the forward model by directly applying SVR and NN regression approaches, respectively. The inverse problem formulation however, is intrinsically ill-posed as ambiguities and one-to-many mappings result from the physical relationships leading to non-unique solutions and inaccuracies in the predictions. Therefore, procedures to include the forward model into the inverse regression problem formulation have been recently developed by Ma et al. (2022), where tandem NNs are used to integrate the forward model into the process to improve the inverse predictions.

Alternatively, Kim et al. (2018) and Zhang et al. (2022) suggested to remain solely with the forward NN model and then applying an optimization method such as a genetic algorithm (GA) based on this model to find an optimal solution. However, the simplifying usage of the forward model has to be compensated by the additional computational complexity induced by the utilization of an optimization algorithm.

2 Methodology

In this paper, we present a different approach compared to the eye diagram-based applications that are widely used in the literature. Extraction of specific signal descriptive time domain features enables SI analysis and design support during the PCB layout phases. Various common PCB net structures can be investigated using this generic approach, see John et al. (2022). Furthermore, a two-stage AI framework is established. During the first step, NN regression models are implemented to learn the relationship between the electrical PCB parameters and the defined signal time domain features in both, forward and inverse direction. Then, during the second step, the learned representations stored in the NN models are utilized to parameterize the interconnect structure and provide design suggestions. In this context, the k-nearest neighbor (kNN) algorithm is applied as a feature-based preselection method to find solution candidates within a feature space population as input for the inverse NN model, while the GA optimizes the PCB parameters directly based on the forward NN model.

Figure 2 provides an overview of the underlying toolchain and data flow of the proposed methodology. This includes the simulation environment for data generation, the data processing and feature extraction to provide learning datasets and finally the two-stage AI framework. The utilized simulation environments are LTspice (see Analog Devices, 2021) and CADSTAR/eCADSTAR (see Zuken, 2021a, b), with only the latter two capable of reasonably translating I/O buffer information specification (IBIS) models for non-linear integrated circuit (IC) characteristics, while LTspice is limited to linear IC modeling. The simulation data is then further processed and the time-domain features are extracted within Python to generate learning datasets. These features are defined by:

$$E_\mathrm{c} = \int_{-\infty}^{\infty} |V(t)^2| \,\mathrm{d}t < \infty \quad E_\mathrm{d} = \sum_{i=1}^{J} |V_i(t)^2|(t_i - t_{i-1}) \quad (1)$$

$$H = -\sum_{k \in K} p_k \log_2 p_k \quad H_\mathrm{scaled} = \frac{H}{H_\mathrm{max}} = \frac{H}{\log_2 K} \quad (2)$$

$$\mathrm{SR} = \frac{V_{90\,\%} - V_{10\,\%}}{t_\mathrm{r}} = \frac{V_{90\,\%} - V_{10\,\%}}{t(V_{90\,\%}) - t(V_{10\,\%})} \quad (3)$$

$$V_\mathrm{max} = \max(V) \quad (4)$$

where E_d, H_scaled, SR and V_max correspond to discrete energy, scaled entropy, slew rate and maximum voltage, respectively. Also, the variable J is associated with the number of voltage samples and K with the number of unique voltage samples, while the variable p_k is equivalent to the voltage amplitude probability density. The created learning

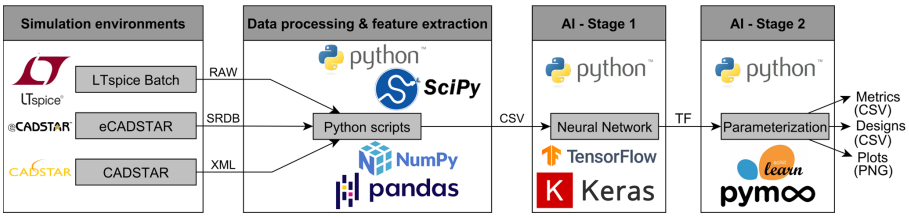

Figure 2. Illustration of the underlying toolchain and data flow of the proposed approach. The two elements on the left represent the provision of learning data, while the two elements on the right show the two-stage AI framework.

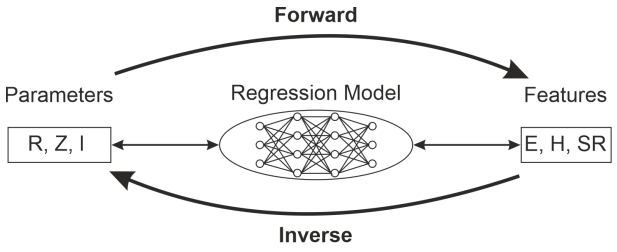

Figure 3. Illustration of the forward and inverse mode regression. Forward mode regression corresponds to the prediction of features from given electrical parameters, while inverse mode regression represents the direct opposite i.e. the prediction of electrical parameters from given features.

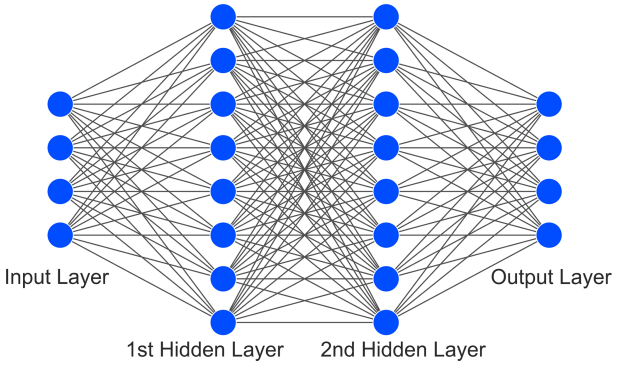

Figure 4. Illustration of the structure of the implemented NN architecture with input, output and two hidden layers.

datasets are then applied to the two-stage AI framework in the next step. In this context, the NN architectures are built with the Keras library (Chollet, 2015), which is embedded into the Tensorflow framework (Abadi et al., 2015), while the kNN algorithms are implemented using the scikit-learn library (Pedregosa et al., 2011) and the GA is derived from the pymoo library (Blank and Deb, 2020).

3 Machine Learning Methods

The most important element of the AI framework are the NN architectures. NNs have been proven to be a suitable tool for regression tasks to support SI analysis, see Lu et al. (2018) and Roy et al. (2019). The implemented NN regression models realize the prediction of numeric values either in forward or inverse mode as shown in Fig. 3. The structure of the utilized NN is depicted in Fig. 4 consisting of input, output and two hidden layers, each with a specified number of neurons, which hold an adjustable weight and an activation function. During the training of the NN a certain amount of data examples corresponding to the batch size are inserted in the input layer and propagated through the network with its current weights to finally compute the loss at the output layer. For the implemented regression this loss is equivalent to the mean squared error (MSE) loss:

$$\mathrm{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2 \tag{5}$$

where N, y_i and $\hat{y}_i$ are associated with the number of data examples, true value and predicted value, respectively. The loss is then iteratively minimized by adjusting the neuron weights using backpropagation and automatic differentiation principles in the individual training steps also known as epochs. For further information on the fundamentals of NNs, see Goodfellow et al. (2016). Finally, this results in a number of NN hyperparameters, which are summarized in Table 1 for the implemented NN.

Important metrics for the evaluation of the regression performance are the root mean squared error (RMSE), which is determined by extracting the root from the MSE of Eq. (5), to measure the actual occurring prediction deviations in the same order of magnitude as the observed variable and the R^2-Score as defined by:

$$R^2 = \frac{\sum_{i=1}^{N}(\hat{y}_i - \overline{y_i})^2}{\sum_{i=1}^{N}(y_i - \overline{y_i})^2} = 1 - \frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{N}(y_i - \overline{y_i})^2} \tag{6}$$

denoting the explainability of the regression, where $\overline{y_i}$ is associated with the mean true value.

To ensure the performance and convergence of the NN, prior to the training, the input data x and output data y is normalized according to:

$$x_{\mathrm{norm.}} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} \quad y_{\mathrm{norm.}} = \frac{y - y_{\min}}{y_{\max} - y_{\min}} \tag{7}$$

Table 1. Hyperparameter setting of the NN.

Hyperparameter	Value
Number of Epochs	1000
Optimizer	Adam
Batch Size	32
Learning rate η	1×10^{-3}
Neurons (Hidden Layer)	32
Activation Function (Hidden Layer)	ReLU

Table 2. List of BO-tunable hyperparameters and their lower and upper tuning limits. Note that each of the hidden layer hyperparameters are individually tuned. The additional hyperparameter dropout factor realizes a regularization by randomly ignoring a defined percentage of neurons during each training epoch.

Hyperparameter	Lower	Upper
Learning rate η	1×10^{-4}	1×10^{0}
Neurons (Hidden Layer)	32	512
Dropout Factor (Hidden Layer)	0	0.8
Activation Function (Hidden Layer)	ReLU, Tanh, Sigmoid	

Also, the dataset is split into 70 % training, 20 % validation and 10 % testing data. The NN model is exclusively exposed to the training data for the learning process, while the validation data is simultaneously utilized to evaluate the performance on unseen data. The testing data is not employed before the training of the model is completed and is then utilized to evaluate the performance on data that has not been considered during the whole training process.

In addition to the default hyperparameter configuration denoted in Table 1, another configuration based on Bayesian optimization (BO) for hyperparameter tuning is realized. The tuning with BO is based on an underlying Gaussian process regression, which models the MSE loss as the objective function with the NN hyperparameters as input variables, which is iteratively minimized by taking previous evaluations into consideration. For further insights on BO, see Frazier (2018). The mentioned properties render BO to be superior in terms of efficiency especially for high-dimensional hyperparameter spaces compared to random or grid search, which roam the hyperparameter space more blindly and are not equipped with any form of memory of past evaluations. For the implemented BO, 50 iterations are carried out and the tunable hyperparameters with their respective tuning limits are represented in Table 2. The number of epochs and batch size hyperparameters are not tuned to allow a fair comparison to the default configuration.

The implemented kNN algorithm is a basic unsupervised ML method, which can be used for feature-based preselection. In collaboration with the inverse NN model this has proven to be a promising approach for SI design support as described in John et al. (2022). kNN is a proximity-based approach that searches k feature sets, which are distance-wise the closest to a given feature example within a feature population sampled from a multivariate normal distribution encompassing the entire feature space. The features are normalized according to Eq. (7) before application. The kNN algorithm is realized with k set to 5 using the Euclidean distance metric and the brute force computation method.

Finally, the GA is a search and optimization procedure belonging to the class of evolutionary algorithms. In combination with the forward NN model, GA methods have proven to be effective in finding optimal SI design solutions as demonstrated in Kim et al. (2018) and Zhang et al. (2022). By definition, the GA optimization is initialized with a randomly sampled population of size M, which iteratively goes through the evolutionary phases of evaluation, selection, crossover and mutation to eventually minimize a given optimization objective. The implemented GA is realized with a population size M of 100 and with Simulated Binary Crossover (SBX), see Deb et al. (2007). For further information on the underlying GA principles, see Goldberg (1989) as well as Bäck and Schwefel (1993). For the objective functions of the implemented single-target GA the Mean Absolute Error (MAE) as defined by:

$$\mathrm{MAE} = \frac{1}{N} \sum_{i=1}^{N} |\hat{y}_i - y_i| \tag{8}$$

is chosen as the minimization metric. Fundamentally, there are two different objectives for the GA, the most simple and obvious one being the minimization of the MAE distance to a desired pre-defined feature set. This leads to the optimization objective O_1 defined by:

$$\min O_1 = \mathrm{MAE}(\boldsymbol{F}; \boldsymbol{F}_{\mathrm{target}}), \tag{9}$$

where $\boldsymbol{F}$ is the feature vector computed by the NN forward model and $\boldsymbol{F}_{\mathrm{target}}$ is the desired feature set. Another more implicit approach is the simultaneous minimization or maximization of certain features e.g. in this specific case the maximization of energy and slew rate, while concurrently minimizing entropy and maximum voltage. This problem definition is appropriate from a SI perspective as the desired signal waveforms should have a rather high slew rate and energy to prevent slowly rising signal slopes, while the maximum voltage should be rather low to prevent overshoots. Additionally, the entropy should also be rather low corresponding to signals with less information content, which are more similar to an ideal square wave signal, while high entropy correlates with overshoot or low slew rate signals, both of which are suboptimal. This results in the optimization objective O_2 defined by:

$$\min O_2 = \mathrm{MAE}\left([1 - E_{\mathrm{d}}; H_{\mathrm{scaled}}; 1 - \mathrm{SR}; U_{\mathrm{max}}]; \mathbf{0}\right), \tag{10}$$

Note that in both instances the features are normalized according to Eq. (7).

Table 3. Applied parameter variation. Note that each parameter is varied individually and not simultaneously.

Parameter	Values
R_1	[1×10^{-9},10,15,22,33,47,68] Ω
$Z_{0,1}$	[Min. 35 ; Max. 70 ; Step 5] Ω
R_2	[1×10^{-9},10,15,22,33,47,68,100,150] Ω
$Z_{0,2}$	[Min. 35 ; Max. 70 ; Step 5] Ω
l_1	100 mm
l_2	40 mm

4032 data samples

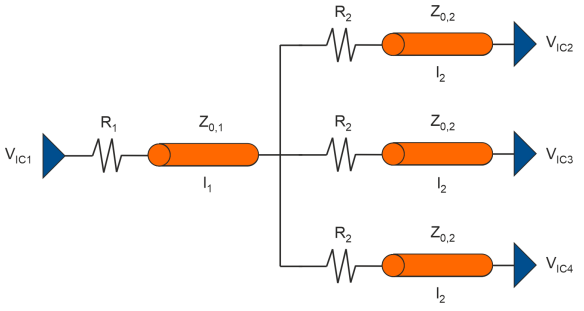

Figure 5. Schematic of the far end cluster topology.

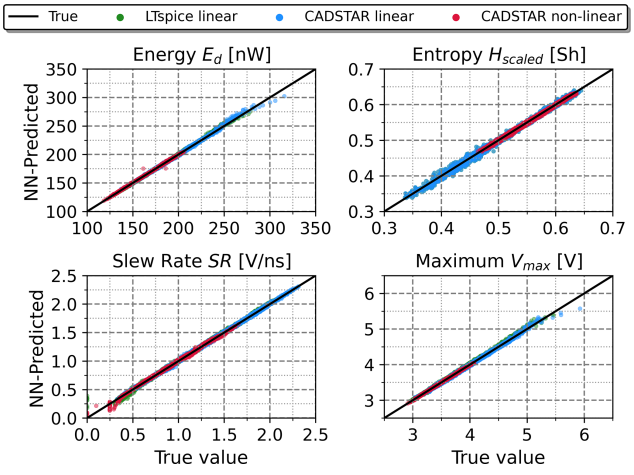

Figure 6. Regression performance of the forward NN model with the BO hyperparameter configuration in terms of true vs. predicted feature values on the entire dataset.

4 Generation of Simulation Data

For the training of the AI models data has been generated using the LTspice and the CADSTAR simulation environments based on the far end cluster shown in Fig. 5. The far end cluster operates with a clock frequency of 50 MHz at 3.3 V using the Texas Instruments SN74AC86 IBIS model for driver and receiver. Concerning the transmission lines, routing on inner layers (symmetric stripline) was assumed. A parameter variation as shown in Table 3 was utilized to generate 4032 simulation data samples. Besides the IBIS-based non-linear simulation in CADSTAR two simulation datasets with linear IC characteristics have been created in LTspice and CADSTAR, respectively, where the voltage as well as rise and fall times were adjusted according to the IBIS model.

Following the results presented in John et al. (2022), the parameters for drivers and receivers in AC technology (AC86) used there for linear and IBIS-based non-linear models were also used for this paper. On the one hand, this allows a detailed comparison with the AI procedures and methods discussed in John et al. (2022), and on the other hand, the further use of the database already available there. Furthermore, we expect that in the future complete AC86 SPICE decks will be available for additional investigations in order to be able to carry out additional process tests.

5 Application of the Artificial Intelligence Models

Based on the far end cluster topology defined in Fig. 5, different simulation data sources with linear and non-linear modeling of the IC characteristics were compared concerning the accuracy, applicability and transferability of the trained AI models.

5.1 Neural Networks

NN architectures have been trained in both, the forward and inverse mode, according to Fig. 3. First, the forward NN was investigated, whose regression accuracy in predicting the features from the electrical parameters is shown in Fig. 6. A consistently high regression accuracy was noticeable for each of the observed simulation datasets, which is also reflected in the regression metrics in Table 4 as testing R^2-Scores above 99 % and invariably small normalized testing RMSE values below 2 % have been observed. As the default hyperparameter configuration already provides very accurate results, the hyperparameter tuning with BO at most only results in marginal improvements in regression performance.

Second, the inverse NN regression has been examined, whose regression performance on the entire dataset is exemplary demonstrated for the prediction of R_1 and $Z_{0,1}$ in Fig. 7. As expected, the regression performance was evidently impaired by the ambiguity of the non-injective learning function, whose mapping consists of multiple sets of electrical parameters corresponding to matching feature settings. However, the color-coded probability density indicates that a larger proportion of predictions is more accurate, while the deviating cases are statistically more unlikely. Furthermore, the regression performance depends on the relative influence of each electrical parameter with respect to the features. This observation is evident from the testing metric results shown in Table 5 with testing R^2-Scores ranging from 24 % to 90 % and testing RMSEs in the range between 6 to

Table 4. R^2-Score and RMSE performance metrics of the forward NN with default and BO hyperparameters based on the entire data. Note that additionally the testing metrics are denoted in brackets.

Metric	Data*	Default NN	BO-NN
Energy E_d [nW]			
R^2-Score	1	0.9985 (0.9984)	0.9992 (0.9991)
R^2-Score	2	0.9981 (0.9986)	0.9981 (0.9981)
R^2-Score	3	0.9991 (0.9971)	0.9995 (0.9978)
RMSE	1	0.8840 (0.8769)	0.6496 (0.6609)
RMSE	2	1.0400 (0.8422)	1.0300 (0.9886)
RMSE	3	0.4841 (0.8139)	0.3552 (0.7138)
Entropy H_{scaled} [Sh]			
R^2-Score	1	0.9916 (0.9899)	0.9933 (0.9897)
R^2-Score	2	0.9960 (0.9952)	0.9977 (0.9958)
R^2-Score	3	0.9970 (0.9959)	0.9983 (0.9973)
RMSE	1	0.0050 (0.0054)	0.0044 (0.0054)
RMSE	2	0.0041 (0.0047)	0.0032 (0.0044)
RMSE	3	0.0022 (0.0025)	0.0017 (0.0020)
Slew Rate SR [V ns^{-1}]			
R^2-Score	1	0.9980 (0.9988)	0.9981 (0.9972)
R^2-Score	2	0.9994 (0.9993)	0.9995 (0.9994)
R^2-Score	3	0.9977 (0.9975)	0.9989 (0.9987)
RMSE	1	0.0209 (0.0158)	0.0207 (0.0241)
RMSE	2	0.0125 (0.0127)	0.0110 (0.0120)
RMSE	3	0.0132 (0.0133)	0.0092 (0.0097)
Maximum V_{max} [V]			
R^2-Score	1	0.9975 (0.9973)	0.9992 (0.9993)
R^2-Score	2	0.9982 (0.9983)	0.9989 (0.9986)
R^2-Score	3	0.9984 (0.9982)	0.9993 (0.9992)
RMSE	1	0.0211 (0.0209)	0.0122 (0.0110)
RMSE	2	0.0197 (0.0185)	0.0155 (0.0167)
RMSE	3	0.0065 (0.0066)	0.0043 (0.0043)

* 1: LTspice linear, 2: CADSTAR linear, 3: CADSTAR non-linear.

42 Ω with R_1 being predicted with the highest and R_2 with the lowest accuracy. The regression performance of the NN based on the non-linear data has been substantially higher than for the NN models trained with the linear data. This result can be explained by the more realistic modeling of the IC characteristics in the non-linear case. The BO-based hyperparameter tuning affects the NN trained with non-linear data and reduces the RMSE effectively, while at the same time this effect cannot be observed for the NNs trained with linear data. However, the stated higher regression performance of the NN based on the non-linear data is visible for both, the BO and default hyperparameter configurations.

A second verification of the inverse NN regression has been performed using a forward remapping as shown in

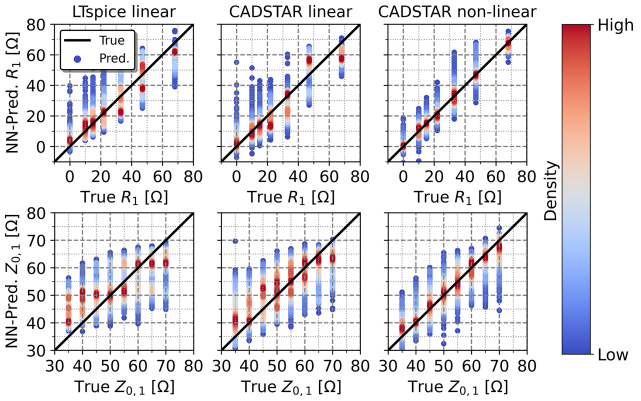

Figure 7. Regression performance of the inverse NN model with the BO hyperparameter configuration in terms of true vs. predicted electrical parameter values exemplary for the prediction of R_1 and $Z_{0,1}$. Note that the color-coded density corresponds to the normalized probability density function for each sweeped parameter value.

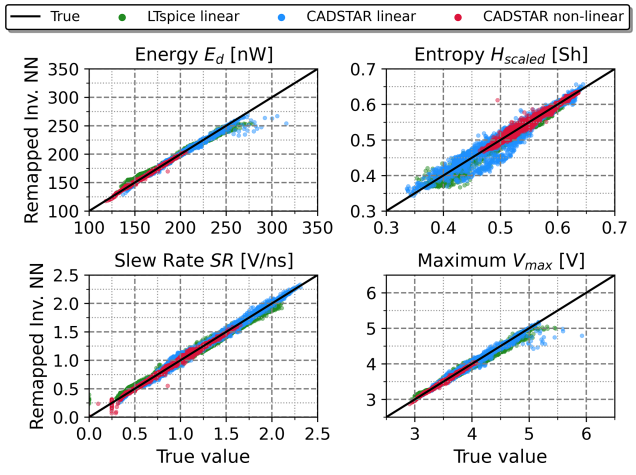

Figure 8. Comparison between the features obtained using the remapping approach based on the inverse NN model and the true feature values.

Fig. 8. The inverse predicted electrical parameters, which had been determined based on features resulting from simulation data, have been re-mapped into the feature space once again utilizing the previously trained forward NN model, see Fig. 6. The utilization of the forward model instead of resimulations has been chosen to prevent long simulation runs required for the large number of examples. Moreover, the forward NN model has proven to be very accurate in predicting the features, while also offering direct comparability of the re-mapped results with the stand-alone forward model. When comparing the remapping of Fig. 8 with the forward model of Fig. 6, a degradation in performance is evident, which can be traced back to the error induced by the inverse NN model. This is also supported by the quality metrics as the R^2-Scores drop by a few percent and the RMSE values at least double throughout the observations. However, the induced error

Table 5. R^2-Score and RMSE performance metrics of the inverse NN with default and BO hyperparameters based on the entire data. Note that additionally the testing metrics are denoted in brackets.

Metric	Data*	Default NN	BO-NN
R_1 [Ω]			
R^2-Score	1	0.8213 (0.7599)	0.8187 (0.7561)
R^2-Score	2	0.8375 (0.7883)	0.8504 (0.8002)
R^2-Score	3	0.9197 (0.8913)	0.9422 (0.9084)
RMSE	1	9.18 (10.04)	9.24 (10.12)
RMSE	2	8.75 (9.43)	8.40 (9.16)
RMSE	3	6.15 (6.76)	5.22 (6.2)
R_2 [Ω]			
R^2-Score	1	0.2764 (0.2804)	0.2892 (0.2789)
R^2-Score	2	0.4247 (0.4060)	0.5269 (0.5204)
R^2-Score	3	0.2562 (0.2390)	0.4284 (0.2863)
RMSE	1	39.32 (41.05)	38.97 (41.10)
RMSE	2	35.06 (37.30)	31.79 (33.52)
RMSE	3	39.86 (42.22)	34.94 (40.89)
$Z_{0,1}$ [Ω]			
R^2-Score	1	0.4556 (0.4361)	0.4463 (0.4069)
R^2-Score	2	0.4575 (0.4326)	0.5623 (0.4749)
R^2-Score	3	0.6217 (0.5510)	0.7860 (0.7212)
RMSE	1	8.45 (8.73)	8.52 (8.95)
RMSE	2	8.44 (8.75)	7.58 (8.42)
RMSE	3	7.05 (7.79)	5.30 (6.14)
$Z_{0,2}$ [Ω]			
R^2-Score	1	0.4217 (0.3339)	0.3788 (0.2806)
R^2-Score	2	0.4943 (0.4678)	0.5079 (0.4595)
R^2-Score	3	0.6296 (0.6340)	0.7530 (0.6705)
RMSE	1	8.71 (9.57)	9.03 (9.95)
RMSE	2	8.15 (8.55)	8.04 (8.62)
RMSE	3	6.97 (7.09)	5.69 (6.73)

*: LTspice linear, 2: CADSTAR linear, 3: CADSTAR non-linear.

seems to be acceptable especially compared with the direct inverse NN results obtained in Fig. 7 and under the aspect that the forward model achieved a very high accuracy. The second verification of the inverse NN regression therefore provides a first assessment on how much the prediction error of the electrical parameters will affect the features and therefore also the predicted signal behavior.

Based on the feature settings of Table 6, the inverse NN model has been tested using real application cases by analyzing the voltage waveforms resulting from the predicted electrical parameters shown in Table 7. These voltage waveforms are visualized for each of the three cases and for each of the three considered simulation datasets in Fig. 9. These show a distinctive behavior for each case, while also matching the expected signal shape. This confirms the suitability of the inverse NN model within a typical use case. Minor overshoots can be observed for the best case of the models trained with the linear datasets, which is most certainly caused by the inferior regression performance in comparison to the model trained with the non-linear dataset.

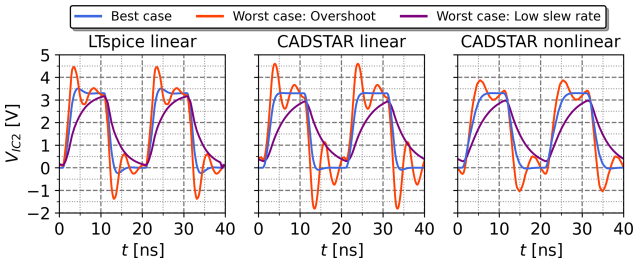

Figure 9. Voltage waveforms resulting from the electrical parameters predicted by the inverse NN based on the feature settings of Table 6.

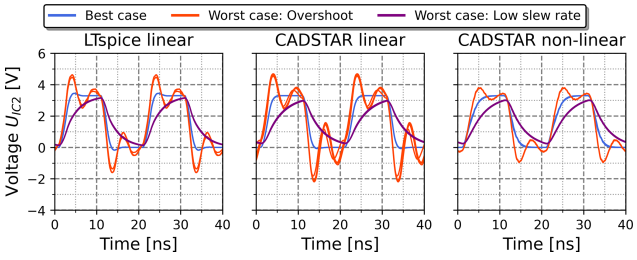

Figure 10. $k = 5$ voltage waveforms resulting from the application of the kNN-NN combination based on the feature settings of Table 6.

5.2 *k*-Nearest Neighbor

The kNN algorithm has been applied for a feature-based preselection based on the cases of Table 6 and was then coupled with the prediction of the inverse NN. The resulting voltage waveforms for each of the cases and simulation datasets are demonstrated in Fig. 10. The voltage waveforms show a distinctive behavior matching the expectations for each case and a very similar behavior for each of the k signals per case. Only small deviations for the best case of linear datasets are observable, which is in accordance with the stand-alone application of the inverse NN in Fig. 9. In general, the voltage waveforms confirm the quality of the proposed combined kNN-NN approach.

To further evaluate the portability between the linear and non-linear data using the proposed kNN-NN method, the direct application has been taken into consideration. Therefore, the electrical parameters resulting from the kNN-NN approach, which were entirely determined based on the linear data, have been extracted and resimulated with non-linear IC characteristics. The result is shown in Fig. 11 and demonstrates how the kNN-NN approach deployed with linear data

Table 6. Feature settings for three defined cases and considered simulation data sources.

Feature	LTspice linear			CADSTAR linear			CADSTAR non-linear		
	OS*	LSR*	BC*	OS*	LSR*	BC*	OS*	LSR*	BC*
Energy E_d [nW]	237	136	211	247	135	191	200	126	171
Entropy H_{scaled} [Sh]	0.60	0.60	0.37	0.62	0.64	0.36	0.60	0.64	0.52
Slew Rate SR [V ns^{-1}]	2.40	0.30	1.46	2.45	0.31	1.32	1.70	0.26	0.79
Maximum V_{max} [V]	4.98	3.02	3.30	5.20	3.02	3.30	3.77	3.02	3.30

* OS: Worst Case – Overshoot; LSR: Worst Case – Low Slew Rate; BC: Best Case.

Table 7. Electrical parameters predicted by the inverse NN based on the feature settings of Table 6.

Data[a]	Parameter	NN Prediction		
		OS[b]	LSR[b]	BC[b]
1	R_1 [Ω]	1	64	17
1	R_2 [Ω]	4	64	21
1	$Z_{0,1}$ [Ω]	41	47	39
1	$Z_{0,2}$ [Ω]	70	38	66
2	R_1 [Ω]	0	68	13
2	R_2 [Ω]	0	150	3
2	$Z_{0,1}$ [Ω]	41	35	35
2	$Z_{0,2}$ [Ω]	70	35	70
3	R_1 [Ω]	6	68	36
3	R_2 [Ω]	0	80	0
3	$Z_{0,1}$ [Ω]	70	36	59
3	$Z_{0,2}$ [Ω]	70	37	70

[a]: LTspice linear, 2: CADSTAR linear, 3: CADSTAR non-linear. [b] OS: Worst Case - Overshoot, LSR: Worst Case – Low Slew Rate, BC: Best Case.

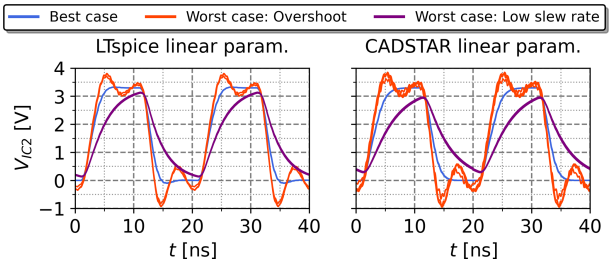

Figure 11. $k = 5$ voltage waveforms simulated with non-linear IBIS models, but parameterized with the kNN-NN combination based on the linear datasets and the feature settings of Table 6.

can be directly applied to perform accurate SI analysis with respect to non-linear IC characteristics for the investigated topology.

In alternative investigations, the kNN algorithm has been initialized similarly to the GA optimization objective defined in Eq. (10) by setting the energy and slew rate features to the maximum and entropy and maximum voltage features to the minimum normalized value i.e. one and zero, respectively. The resulting voltage waveforms of this kNN initialization with subsequent inverse NN coupling are shown for each of the simulation datasets in Fig. 12. The practical applicability of this approach is proven as the parameterized designs generally exhibit SI-aware behavior. Only for the linear data sources some overshoot behavior is visible, which is consistent with previous results.

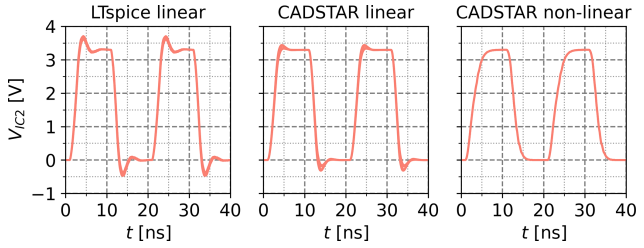

Figure 12. $k = 5$ voltage waveforms resulting from the application of the kNN-NN combination with a feature setting analogical to O_2.

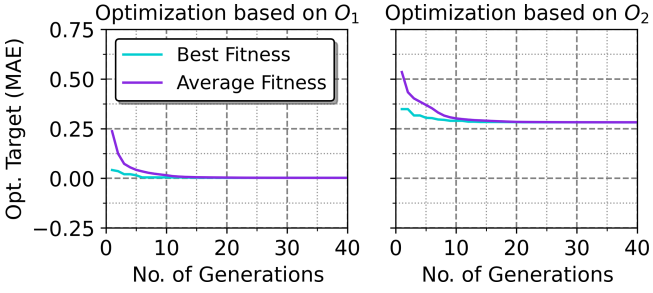

Figure 13. Convergence of the GA over its generations with average and best fitness MAE objective values shown for the CADSTAR linear dataset. Similar convergence behavior has been observed for the other two datasets also.

5.3 Genetic Algorithms

The GA has been applied to select the optimal electrical parameters within the feature space based on the forward NN model using the two optimization objectives defined in the Eqs. (9) and (10) with $\boldsymbol{F}_{target}$ equivalent to the best case feature settings of Table 6. For both objectives, the duration

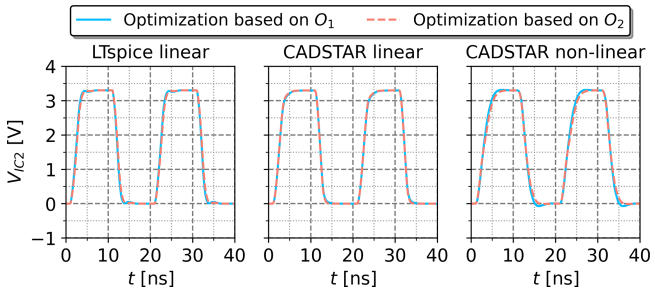

Figure 14. Voltage waveforms resulting from the electrical parameters optimized by the GA based on the objectives O_1 and O_2.

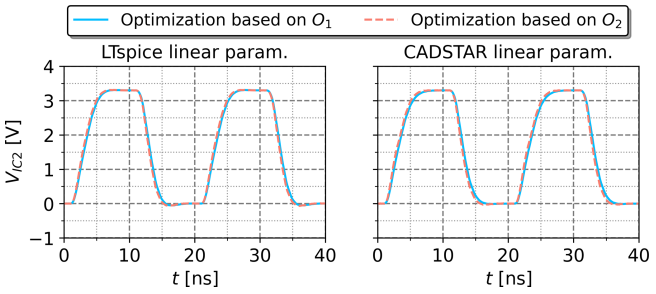

Figure 15. Voltage waveforms simulated with non-linear IBIS models, but parameterized by the GA optimization based on the objectives O_1 and O_2.

Table 8. Electrical parameters optimized by the GA based on objective O_1 and objective O_2.

Data*	Obj.	R_1 [Ω]	R_2 [Ω]	$Z_{0,1}$ [Ω]	$Z_{0,2}$ [Ω]
1	O_1	20	67	41	70
1	O_2	24	19	42	70
2	O_1	26	46	39	70
2	O_2	23	0	35	70
3	O_1	13	71	41	45
3	O_2	36	0	57	70

*: LTspice linear, 2: CADSTAR linear, 3: CADSTAR non-linear.

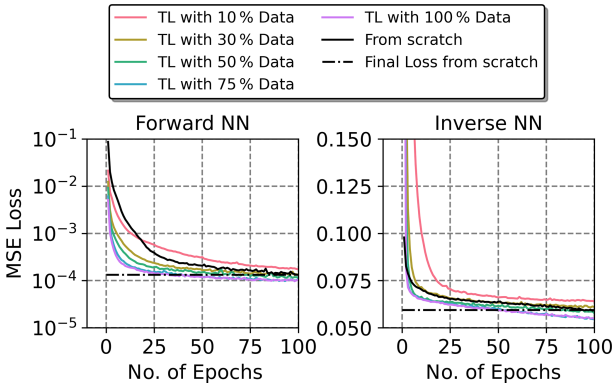

Figure 16. MSE training loss over epochs development during the transfer learning process with different data fractions compared to the training with non-linear data from scratch. Note the logarithmic scaling of the y-axis in case of the forward NN.

of the GA was below 30 s on a common hardware setup. The convergence history of the algorithms is shown for the CADSTAR linear dataset in Fig. 13. It is visible that convergence to a minimum value sets in for both objectives. As expected, the final MAE converges to zero for the first objective, which implies that the distance to the defined best case feature setting embedded into the design space is effectively minimized. For the second objective, a larger residual error remains as the objective can only partly be fulfilled. Due to the physical background of the optimization problem, the possible solution space is limited to valid parameters within the design space. The concurrent optimization of objective O_2 forbids that each feature can be completely maximized or minimized, as this would lead to non-physical solutions, which are intrinsically not allowed. Therefore, the residual MAE indicating the average absolute deviation from the desired target remains above 25 %. The voltage waveforms resulting from the optimized electrical parameters demonstrated in Table 8 are shown in Fig. 14. SI-aware designs have been observed throughout the objectives as well as the different data sources, which validates the usability of the proposed GA method for design optimization tasks.

The direct portability between linear and non-linear data has been evaluated for the utilization of the GA method also. Therefore, the electrical parameters resulting from the GA optimization based on linear data, see Table 8, were extracted and resimulated with non-linear IC characteristics. The resulting voltage waveforms are shown in Fig. 11 and validate that the GA based on linear data can be directly applied to provide accurate SI analysis with respect to non-linear IC characteristics for the investigated topology.

5.4 Transfer Learning

The transfer learning concept, which is also known as fine-tuning, offers another option to ensure portability between the NN architectures trained with linear and non-linear data. The implemented transfer learning has been applied based on a NN pre-trained with the CADSTAR linear dataset, which was then fine-tuned with 100 epochs of the non-linear data. Moreover, the transfer learning applications were applied with various fractions of data such as 10 %, 30 %, 50 %, 75 % and 100 %, which have been randomly sampled from the original dataset. Figure 16 shows the MSE loss over epochs development of the transfer learning process for the forward and inverse mode regression. It is apparent that with data fractions of 50 % and above, the convergence was accelerated resulting in a lower loss after 100 epochs for both, the forward and inverse mode regression, compared to the training from scratch. This is due to the learned knowledge rep-

resentations contained in the pre-trained NNs. Compared to the training from scratch with a data fraction of 30 %, the training MSE loss of the forward transfer learning was at least on a par and the inverse transfer learning was slightly impaired. Finally, for the transfer learning with a fraction of 10 % the amount of data apparently falls below the threshold of required data quantity as the MSE training loss was significantly higher compared to the from scratch trained model for both, the forward and inverse transfer learning. Overall, the application of transfer learning between linear and non-linear IC characteristics entails significant simplifications of the data generation and training processes. The fine-tuning on an existing model that has been trained based on linear data enables savings of at least 50 % during the data generation with non-linear IC characteristics, which then also effectively reduces training complexity and computation times.

6 Conclusions and Future Work

The two-stage AI framework presented in this paper has been validated to support the application in the PCB design process with respect to linear and non-linear data. SI-compliant designs were obtained using the forward NN in combination with a GA for optimization as well as the inverse NN in combination with the kNN algorithm. The different data sources only affected the performance with respect to the inverse NN as the regression accuracy was significantly improved when utilizing the non-linear data. In this context, the NN hyperparameter tuning with BO also had a significant impact in further improving the regression performance, which is in contrast to the other results, where the impact of hyperparameter tuning with BO was marginal at best. Furthermore, it has also been shown that for the investigated application the designs parameterized using the AI framework based on linear datasets can be successfully applied to provide SI design support valid for the non-linear IC characteristics also. Alternatively, the transfer learning capabilities of the NNs have been evaluated for fine-tuning of the models trained with linear data using different fractions of non-linear data. These investigations showed that at least 50 % of the training data can be saved to achieve the same or even improved regression performance.

Future research may incorporate the application of the presented AI framework to technologies from the DDR-SDRAM class such as DDR3/4/5 with much shorter rise- and fall times operating at higher clock frequencies and lower voltage levels to investigate the portability and adaptability of the developed AI models. In particular, the transfer of the developed AI models for different component technologies into practice-oriented AI modules (deployment) will be an important subject. In this context, the further integration of SI constraints will be of great importance, especially considering the timing and resulting skew characteristics of the signals, which also requires a more differentiated examination of the feature selection. Furthermore, the constraint integration during the parameterization stage may include feature weighting and multi-objective optimization methods to provide customization to the developer's needs. In the end, combining the forward and inverse NN models in the first AI stage as well as different parameterization methods such as kNN and GA in the second AI stage within an iterative process may yield synergy effects resulting in significant improvements of the framework.

Code and data availability. The Python code and simulation data is available from the corresponding author upon request.

Author contributions. JW developed the AI concepts and framework for this part of the research topic by designing the AI models and training methods. JW wrote the initial and final manuscript. WJ contributed to understanding the SI knowledge domain and the evaluation of the prediction results (e.g. determining and understanding overshoot behavior with sufficient accuracy). He also supported the improvement of AI models in terms of SI design aspects and processes. In addition, he supervised the overall research process in the progressivKI project (UseCase #2: SI PCB Design) and contributed in part to the improvement of the manuscript. EE contributed to the comparison of predicted signals on PCB net structures with his results from the application of decision trees for anomaly detection and SI prediction verification. RB supported the development of AI models by providing appropriate data sets and understanding the perspective of PCB designers with respect to the application of SI design rules. RB also supported the efficient use of the ZUKEN eCADSTAR SI/PI design environment in the context of the progressivKI project (UseCase #2: SI PCB Design). In addition, RB brought his knowledge of using IBIS models to the overall concept for this paper. EE, WJ, RB, and JG discussed the results, contributed to the final manuscript, and provided editorial input. JG guided the entire research process at TU Dortmund University (Information Processing Laboratory) and conducted the final review.

Competing interests. The contact author has declared that none of the authors has any competing interests.

Disclaimer. Publisher's note: Copernicus Publications remains neutral with regard to jurisdictional claims made in the text, published maps, institutional affiliations, or any other geographical representation in this paper. While Copernicus Publications makes every effort to include appropriate place names, the final responsibility lies with the authors.

Special issue statement. This article is part of the special issue "Kleinheubacher Berichte 2022". It is a result of the Kleinheubacher Tagung 2022, Miltenberg, Germany, 27–29 September 2022.

Financial support. This work is funded as part of the research project progressivKI https://www.edacentrum.de/projekte/progressivKI (last access: 18 October 2023) in the funding programme NFST by the Bundesministerium für Wirtschaft und Klimaschutz (BMWK) of the Federal Republic of Germany (grant no. 19A21006B/O).

Review statement. This paper was edited by Jens Anders and reviewed by two anonymous referees.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X.: TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, https://www.tensorflow.org/ (last access: 18 October 2023), 2015.

Analog Devices: LTspice XVII, https://www.analog.com/en/design-center/design-tools-and-calculators/ltspice-simulator.html (last access: 18 October 2023),

Bäck, T. and Schwefel, H.-P.: An Overview of Evolutionary Algorithms for Parameter Optimization, Evol. Comput., 1, 1–23, https://doi.org/10.1162/evco.1993.1.1.1, 1993.

Blank, J. and Deb, K.: pymoo: Multi-Objective Optimization in Python, IEEE Access, 8, 89497–89509, https://doi.org/10.1109/ACCESS.2020.2990567, 2020.

Chen, S., Chen, J., Zhang, T., and Wei, S.: Semi-Supervised Learning Based on Hybrid Neural Network for the Signal Integrity Analysis, IEEE T. Circuits. Syst. II: Express Briefs, 67, 1934–1938, https://doi.org/10.1109/TCSII.2019.2948527, 2020.

Chollet, F.: Keras [code], https://keras.io (last access: 18 October 2023), 2015.

Deb, K., Sindhya, K., and Okabe, T.: Self-Adaptive Simulated Binary Crossover for Real-Parameter Optimization, GECCO '07, 1187–1194, Association for Computing Machinery, New York, NY, USA, https://doi.org/10.1145/1276958.1277190, 2007.

Ecik, E., John, W., Withöft, J., and Götze, J.: Anomaly Detection with Decision Trees for AI Assisted Evaluation of Signal Integrity on PCB Transmission Lines, 20, submitted to Advances in Radio Science, 2023.

Frazier, P. I.: A Tutorial on Bayesian Optimization, arXiv [preprint], https://doi.org/10.48550/arxiv.1807.02811, 8 July 2018.

Goldberg, D. E.: Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, New York, NY, USA, ISBN: 978-0-201-15767-3, 1989.

Goodfellow, I., Bengio, Y., and Courville, A.: Deep Learning, MIT Press, Cambridge, MA, USA, ISBN: 978-0-262-03561-3, 2016.

John, W., Withöft, J., Ecik, E., Brüning, R., and Götze, J.: A Practical Approach Based on Machine Learning to Support Signal Integrity Design, in: 2022 International Symposium on Electromagnetic Compatibility – EMC Europe, 623–628, https://doi.org/10.1109/EMCEurope51680.2022.9901213, 2022.

Kim, H., Sui, C., Cai, K., Sen, B., and Fan, J.: Fast and Precise High-Speed Channel Modeling and Optimization Technique Based on Machine Learning, IEEE T. Electromagn. C., 60, 2049–2052, https://doi.org/10.1109/TEMC.2017.2782704, 2018.

Lu, T., Sun, J., Wu, K., and Yang, Z.: High-Speed Channel Modeling With Machine Learning Methods for Signal Integrity Analysis, IEEE T. Electromagn. C., 60, 1957–1964, https://doi.org/10.1109/TEMC.2017.2784833, 2018.

Ma, H., Li, E.-P., Cangellaris, A. C., and Chen, X.: High-Speed Link Design Optimization Using Machine Learning SVR-AS Method, in: 2020 IEEE 29th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS), 1–3, https://doi.org/10.1109/EPEPS48591.2020.9231368, 2020a.

Ma, H., Li, E.-P., Cangellaris, A. C., and Chen, X.: Support Vector Regression-Based Active Subspace (SVR-AS) Modeling of High-Speed Links for Fast and Accurate Sensitivity Analysis, IEEE Access, 8, 74339–74348, https://doi.org/10.1109/ACCESS.2020.2988088, 2020b.

Ma, H., Li, E.-P., Wang, Y., Shi, B., Schutt-Aine, J., Cangellaris, A., and Chen, X.: Channel Inverse Design Using Tandem Neural Network, in: 2022 IEEE 26th Workshop on Signal and Power Integrity (SPI), 1–3, https://doi.org/10.1109/SPI54345.2022.9874935, 2022.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E.: Scikit-learn: Machine Learning in Python, J. Mach. Learn. Res., 12, 2825–2830, 2011.

Roy, K., Dolatsara, M. A., Torun, H. M., Trinchero, R., and Swaminathan, M.: Inverse Design of Transmission Lines with Deep Learning, in: 2019 IEEE 28th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS), 1–3, https://doi.org/10.1109/EPEPS47316.2019.193220, 2019.

Trinchero, R., Dolatsara, M. A., Roy, K., Swaminathan, M., and Canavero, F. G.: Design of High-Speed Links via a Machine Learning Surrogate Model for the Inverse Problem, in: 2019 Electrical Design of Advanced Packaging and Systems (EDAPS), 1–3, https://doi.org/10.1109/EDAPS47854.2019.9011627, 2019.

Zhang, H. H., Xue, Z. S., Liu, X. Y., Li, P., Jiang, L., and Shi, G. M.: Optimization of High-Speed Channel for Signal Integrity With Deep Genetic Algorithm, IEEE T. Electromagn. C., 64, 1270–1274, https://doi.org/10.1109/TEMC.2022.3161298, 2022.

Zhang, T., Chen, S., Wei, S., and Chen, J.: A Data-Efficient Training Model for Signal Integrity Analysis based on Transfer Learning, in: 2019 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS), 186–189, https://doi.org/10.1109/APCCAS47518.2019.8953103, 2019.

Zuken: CADSTAR Version 2021.0, Zuken [software], https://www.ecadstar.com/en/product/cadstar (last access: 18 October 2023), 2021a.

Zuken: eCADSTAR Version 2021.1, Zuken [software], https://www.ecadstar.com/en/ (last access: 18 October 2023), 2021b.