Artificial Neural Network (ANN) Based Fast and Accurate Inductor Modeling and Design

Thomas Guillod*, *Member, IEEE,*
Panteleimon Papamanolis*, *Student Member, IEEE,*
Johann W. Kolar*, *Fellow, IEEE*

*Power Electronic Systems Laboratory, ETH Zurich, Switzerland

***Abstract*—This paper analyzes the potential of Artificial Neural Networks (ANNs) for the modeling and optimization of magnetic components and, specifically, inductors. After reviewing the basic properties of ANNs, several potential modeling and design workflows are presented. A hybrid method, which combines the accuracy of 3D Finite Element Method (FEM) and the low computational cost of ANNs, is selected and implemented. All relevant effects are considered (3D magnetic and thermal field patterns, detailed core loss data, winding proximity losses, coupled loss-thermal model, etc.) and the implemented model is extremely versatile (30 input and 40 output variables). The proposed ANN-based model can compute 50′000 designs per second with less than 3% deviation with respect to 3D FEM simulations. Finally, the inductor of a 2kW DC-DC buck converter is optimized with the ANN-based workflow. From the Pareto fronts, a design is selected, measured, and successfully compared with the results obtained with the ANNs. The implementation (source code and data) of the proposed workflow is available under an open-source license.**

***Index Terms*—Power Converters, Inductors, Magnetic Devices, Artificial Neural Networks, Machine Learning, Finite Element Analysis, Pareto Optimization, Open Source Software.**

I. Introduction

THE past few years have seen a rapid development of artificial intelligence applications in research, engineering, and industry [1]. For power electronic systems, the different machine learning methods, i.e., algorithms that learn from data and improve automatically through experience, are particularly interesting for a large variety of applications: fault diagnosis, preventive maintenance, reliability prediction, quality control, control strategies, reverse engineering, advanced modeling, and system or component optimization [2]–[7].

Machine learning has the ability to work with both structured data (organized dataset) or unstructured data (e.g., image, video, text). Machine learning algorithms can be classified into three main groups [1], [8], [9]:

- *Supervised learning* - Learn how to predict values (regression) or categories (classification) from labeled training data (input-output pairs).
- *Unsupervised learning* - Learn how to find patterns (clustering) from a dataset without pre-existing labels. Such methods can also be used for reducing a dataset (data reduction or dimensionality reduction) without losing important information or for detecting anomalies (outlier detection) in a dataset.
- *Reinforcement learning* - Learn how to perform a task by interacting with its environment. The algorithm receives rewards by performing correctly and penalties for performing incorrectly (feedback model).

The most popular implementation of such methods is based on Artificial Neural Networks (ANNs), which share some principles with biological brains. A collection of simple signal processing units (artificial neurons) receive, process, and transmit information from and to the surrounding neurons [1], [8], [9]. The strength (weighting) of the interconnections between the neurons are chosen during the training process, where the ANN is learning from a provided dataset. ANNs represent a broad class of algorithms. For supervised learning, the main classes are [1], [8], [9]:

- *Multilayer Perceptron (MLP)* - Network where the artificial neurons are organized in layers, which are typically fully-connected. The information is passed between the layers with a feed-forward direction. The inputs of the ANNs are vectors. Such ANNs are typically used for problems with structured data or with unstructured data with limited complexity.
- *Convolutional Neural Network (CNN)* - Network where the artificial neurons are organized in feed-forward layers, which are typically sparsely-connected. The usage of convolution layers allows the usage of multi-dimensional data (matrix of tensor) and the detection of complex patterns. Therefore, such ANNs are particularly very well suited to large problems with unstructured data.
- *Recurrent Neural Network (RNN)* - Network where the neurons feature an internal memory. This implies that the outputs do not only depend on the current inputs but also on the previous inputs, making such ANNs interesting for sequential data.

For power electronic systems, ANNs, so far, have been mainly used for fault diagnosis [2], [3] and control strategies [4], [5]. However, another important field of power electronics, requiring complex models and heavy computations, is the modeling and multi-objective optimization of converters and components (e.g., with respect to volume, mass, cost, efficiency) [10]–[13]. Fig. 1 shows a vision of

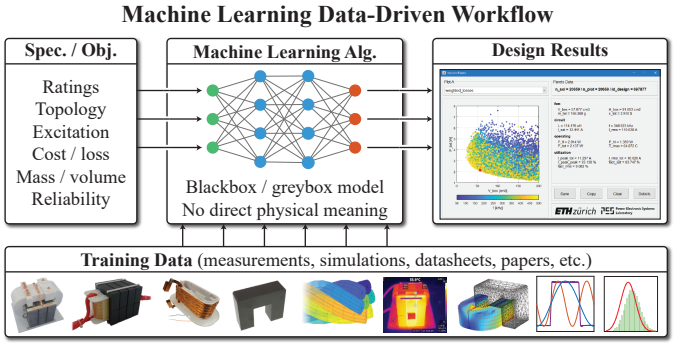

Fig. 1. Data-driven approach for multi-objective optimization of inductors with machine learning and, more specifically, with Artificial Neural Networks (ANNs). The ANNs can be trained with simulations, measurements, datasheets, papers, etc.

what ANNs could provide for multi-objective optimization. From given specifications and goals, the ANNs generate the Pareto fronts and select the optimal designs [6], [7], [14]–[17]. The ANNs learn from a dataset, which can be gained from simulations, measurements, datasheets, papers, etc. With such a workflow, the ANNs do not include a physics-based model, only the input and output variables of the ANNs feature a clear physical meaning.

Nevertheless, in order to be competitive against classical multi-objective optimization algorithms and models, machine learning approaches should avoid several pitfalls and overcome some challenges, i.e. should feature the following properties:

- *Accurate and robust* - The method should be robust and accurate in the complete range, without producing any outlier data.
- *Versatile and flexible* - The method should work for a wide range of parameters and objectives.
- *Extensible and adaptable* - The workflow should be easy to extend (e.g., additional models, parameters, constraints) and to integrate in the design process. Furthermore, it should be possible to accommodate special, project-specific, requirements.
- *Access to internal data* - Not only the optimized variables (e.g., volume, mass, cost, efficiency) should be accessible but also the internal variables and physical parameters (e.g., magnetic field, current density, switching energy, temperatures).
- *Dataset availability* - The dataset, used to train the ANN, should be available or easy to generate.

This paper analyzes the usage of MLP ANNs for modeling and optimization of inductors and proposes a hybrid method, combining the accuracy of 3D Finite Element Method (FEM) and the flexibility of ANNs. Inductors are selected because magnetic components typically represent the bottleneck of multi-objective optimization (e.g., model complexity, computational cost, size and diversity of the design and performance spaces) [11]–[13], [18]. However, all the presented methods are also applicable to other power electronic components (e.g., transformers, semiconductors).

The paper is organized as follows. Section II reviews the fundamentals of ANNs. Section III presents different ANN-

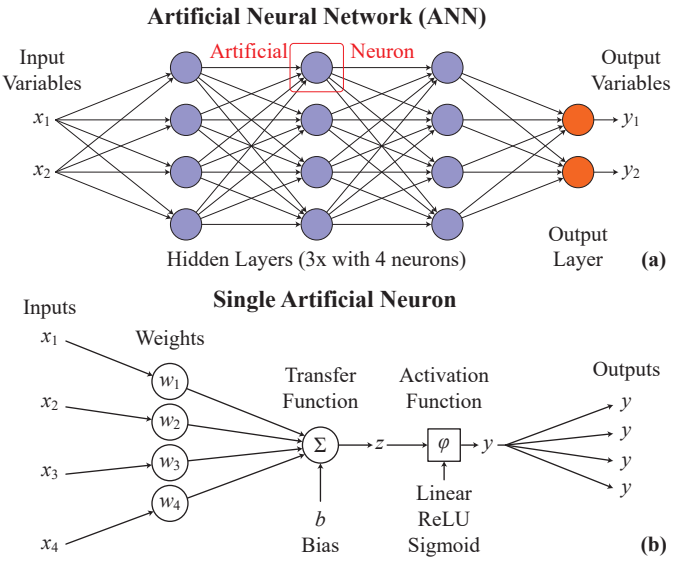

Fig. 2. (a) Structure of an ANN with two inputs, two outputs, three hidden layers, and an output layer. (b) Single artificial neuron where the weights (w_i) and the bias (b) represent the ANN parameters determined during the training process.

based workflows for inductor modeling and/or optimization. In Section IV, the most promising solution is presented in detail and the performances are evaluated in Section V. In Section VI, the method is applied to the optimization, design, and measurement of the inductor of a DC-DC buck converter. In the Appendix details are given about the open-source software implementation of the proposed workflow, "AI-mag" [19].

II. Fundamentals of ANNs

This section introduces the fundamental working principle of MLP ANNs for supervised learning [1], [8], [9]. More specifically, the ANN structure, the training process, the overfitting risk, and data normalization are analyzed for regression problems. The readers who are familiar with these concepts can skip this section.

A. ANN Structure

Fig. 2(a) depicts the computational structure of a MLP ANN, which features several inputs and outputs. The artificial neurons are organized in layers and connected together, like synapses in a brain. The internal layers are called hidden layers (arbitrary number of neurons) and the last layer is the output layer (number of neurons is the number of outputs). ANNs with several hidden layers are usually qualified as deep learning ANNs, while structures with a single hidden layer are shallow ANNs. Adding more neurons and layers extends the learning capabilities of ANNs, allowing the processing of more complex data.

Fig. 2(b) shows how a single artificial neuron is working. First, the different input values are scaled with the weights (w_i), the transfer function is summing the inputs with a bias (b). The resulting value is processed by the activation function and the result is propagated to the connected

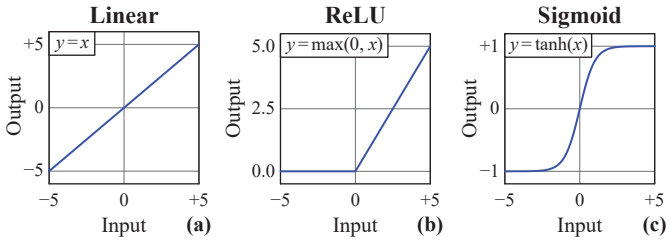

Fig. 3. Typical activation function for ANNs: (a) linear function, (b) rectified linear unit, and (c) sigmoid function.

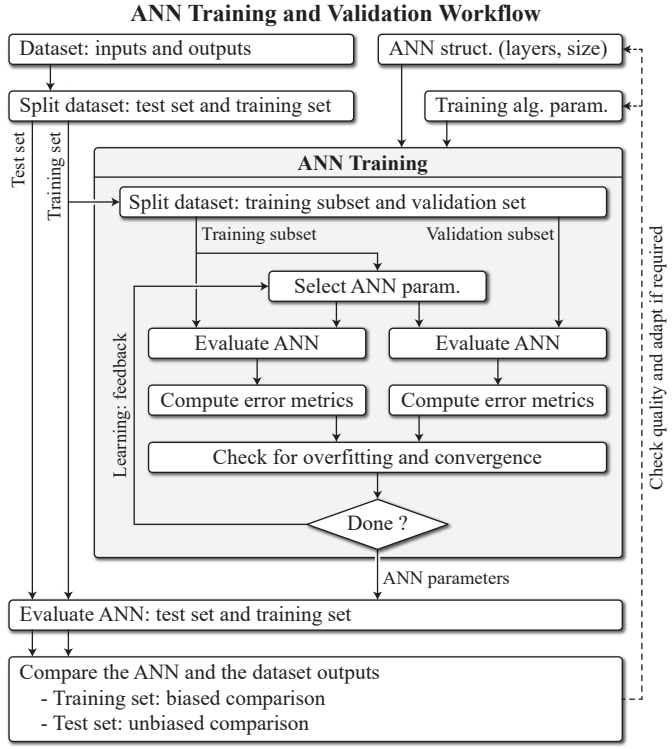

Fig. 4. Workflow for ANN training (supervised learning). The training, overfitting detection, and validation steps are depicted.

neurons. The weights (w_i) and the bias (b) represent the ANN parameters, which are determined by the training process.

Different activation functions can be used for the different layers and the common choices are shown in Fig. 3. The hidden layers, typically, use sigmoids and/or rectified linear units. However, if many hidden layers are used (deep ANNs), rectified linear units should be preferred due to the numerical instability of sigmoids (vanishing gradient) [1]. Regression ANNs (predicting values) feature a linear output layer and classification ANNs (predicting categories) have a sigmoid output layer [9]. This implies that the output values of classification ANNs are real numbers and are, afterwards, transformed (with a given threshold) to binary data [1].

B. ANN Training Process

Fig. 4 describes the training process of an ANN (i.e., the choice of the weights and biases) with respect to a dataset. The different steps are described in the following [1], [9]:

1) The provided samples (pairs of inputs and outputs) are split into a test set and a training set. The split is done randomly and, typically, 80 % of the samples are used for training and 20 % for testing.
2) The training set is subdivided in a training subset and a validation subset. Again, the split is done randomly and, typically, 80 % of the samples are used for the training subset and 20 % for the validation subset.
3) The parameters of the ANN (weights and biases) are selected. For the first iteration, the values are typically selected randomly. For the next iterations, the values are selected with respect to the error metrics obtained in the previous iteration with the training subset.
4) The ANN is evaluated for the training subset and the validation subset and the error metrics are computed between the ANN outputs and the dataset outputs. Widely used metrics are the mean square error (for regression ANNs) and the binary cross-entropy (for classification ANNs).
5) The error metrics of the training subset and the validation subset are compared to stop the training if overfitting is occurring. The error metrics of the training subset are monitored in order to detect the completion of the training when the metrics are converging and stop improving. The overfitting is explained, in more details, in Section II-C.
6) If the convergence is not reached, the error metrics are used to improve the weights and biases for the next iteration. The selection of the weights and biases is explained, in more details, in Section II-D.
7) After the completion of the training, the training set and test set are evaluated and the obtained outputs are compared with the dataset outputs. The comparison is biased for the training set since the same data have been used for the training. For this reason, the test set exists and offers an unbiased validity check.
8) If the performances of the ANN are not sufficient, the ANN structure (e.g., number of hidden layers, number of neurons) or the training algorithms should be updated.

It has to be noted that, due to the random splitting of the dataset and the random initialization of the weights and biases, the ANN training is not a deterministic procedure. For the training iterations, the training subset is divided into several batches (with several samples). For each iteration, a single batch is used for improving the ANN parameters. A training epoch is achieved when all the batches have been processed, i.e., when all the samples of the training subset have been used. The training process of an ANN consists of many epochs. The batch size (number of samples per batch) is a parameter that can affect the quality, stability, and computational cost of the training process.

C. ANN Overfitting

Overfitting means that an ANN is overspecialized with respect to the samples of the training subset, to the detriment of other samples [1], [9]. This is prone to happen with ANNs featuring many artificial neurons compared to

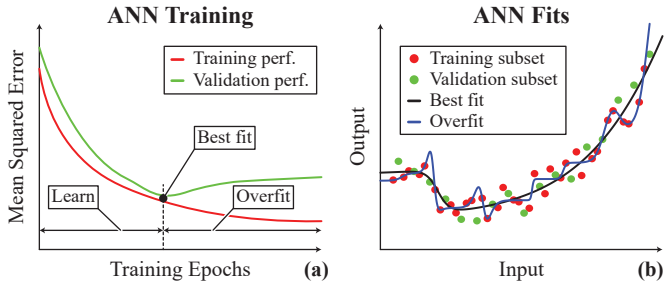

Fig. 5. Overfitting example for a regression ANN with a single input and output. (a) Evolution of the mean square error over the training epochs for the training subset and the validation subset. (b) Noisy training and validation samples, best fit, and overfit.

the size and/or complexity of the dataset. Fig. 5 illustrates this effect for a regression ANN used to fit noisy data. After a certain number of training epochs, the error of the training subset keeps improving while the error of the validation subset starts to deteriorate. It means that the ANN is trying to fit the noise of the training subset, which can be seen from the predicted fits. Therefore, a validation subset is required in order to detect overfitting and terminate the training.

D. ANN Training Algorithm

Selecting the optimal weights and biases of an ANN with respect to a dataset is, fundamentally, an optimization problem. However, an ANN contains hundreds or thousands of weights and biases, resulting in a relatively high level of complexity. For this reason, many different training methods exist. Nevertheless, most of these algorithms share a similar concept: propagation and backpropagation [1], [8], [9].

Fig. 6 presents the fundamental principle of propagation and backpropagation. A simple regression ANN is considered (two inputs, two outputs, a hidden layer, and an output layer) and the selection of the weights is investigated (the selection of the biases is similar). The different steps are described as follows [1], [9]:

1) The weights are initialized with random values.
2) With the selected weights, the ANN is evaluated for the provided inputs. This is done layer by layer from the inputs to the outputs, i.e., propagation.
3) The error between the computed values and the provided outputs is computed. In this example, the sum of the squares is used.
4) The sensitivity of the error with respect to the weights is computed. This is achieved by applying, systematically, the chain rule for the derivative. This is done layer by layer from the outputs to the inputs, i.e., backpropagation.
5) With the obtained sensitivity, the weights are updated in order to diminish the error. This update is controlled by a specified learning rate, which specifies how aggressive the learning algorithm acts.

E. ANN Example: Inductor Scaling Laws

The aforementioned ANN structure and training procedure are applied to a simple example. Scaling laws of non-

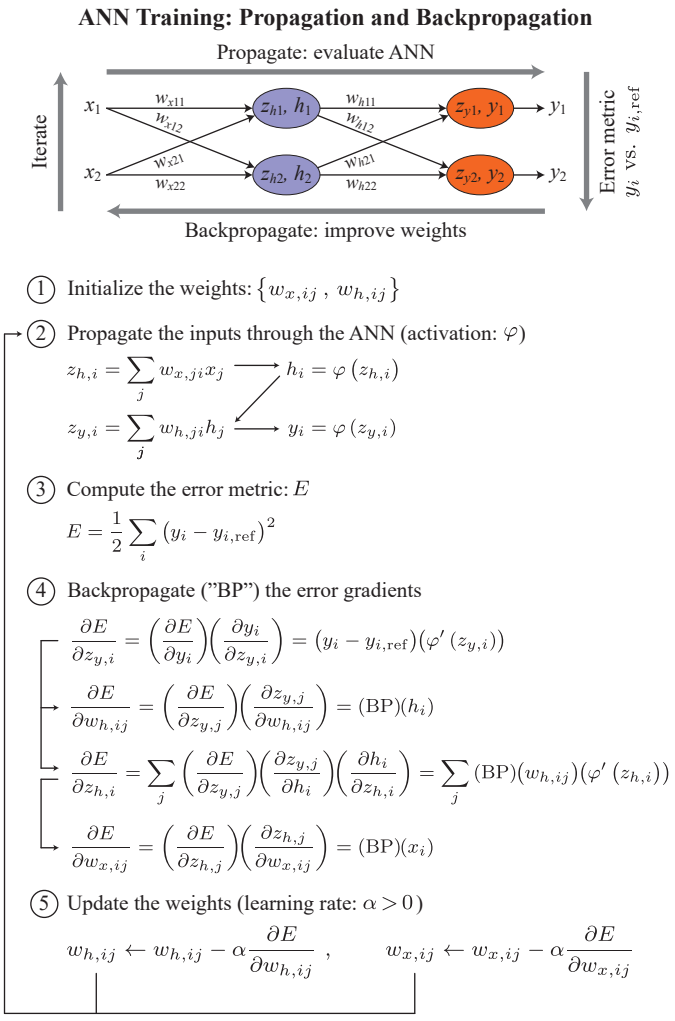

Fig. 6. Basic ANN training algorithm based on the propagation of the inputs and the backprogation of the errors (supervised learning).

isolated DC/DC converter inductors, which predict the losses (P) and the temperature (T) from the boxed volume (V_{box}), the switching frequency (f), the DC current (I_{DC}) and the applied voltage (V_{PWM}) are considered.

$$P = k_{\mathrm{P}} V_{\mathrm{box}}^{\frac{4(2-\beta_{\mathrm{c}})}{3(2+\beta_{\mathrm{c}})}-\frac{1}{3}} f^{\frac{2\alpha_{\mathrm{c}}-2\beta_{\mathrm{c}}}{2+\beta_{\mathrm{c}}}} I_{\mathrm{DC}}^{\frac{2\beta_{\mathrm{c}}}{2+\beta_{\mathrm{c}}}} V_{\mathrm{PWM}}^{\frac{2\beta_{\mathrm{c}}}{2+\beta_{\mathrm{c}}}}, \quad (1)$$

$$T = T_{\mathrm{amb}} + k_{\mathrm{T}} P V_{\mathrm{box}}^{-\frac{2}{3}}, \quad (2)$$

where k_{P} and k_{T} are empirical parameters. The coefficients α_{c} and β_{c} represent the Steinmetz parameters of the core material and T_{amb} the ambient temperature. More details about this empirical model can be found in [20].

The following specifications are considered: $I_{\mathrm{DC}} = 10\,\mathrm{A}$, $V_{\mathrm{PWM}} = 200\,\mathrm{V}$, $T_{\mathrm{amb}} = 55\,^{\circ}\mathrm{C}$, $V_{\mathrm{box}} < 200\,\mathrm{cm}^3$, $T < 130\,^{\circ}\mathrm{C}$, and $f < 750\,\mathrm{kHz}$. With these specifications, the empirical parameters ($k_{\mathrm{P}} = 0.004$, $k_{\mathrm{T}} = 0.02$, $\alpha_{\mathrm{c}} = 1.4$, and $\beta_{\mathrm{c}} = 2.4$) are fitted from the results presented in [21]. Fig. 7 shows the resulting Pareto fronts obtained with (1) and (2).

In order to highlight the working principle of ANNs, different ANN structures (number of layers and activation functions) and variable processing methods (variable transformation and normalization) are compared, using the

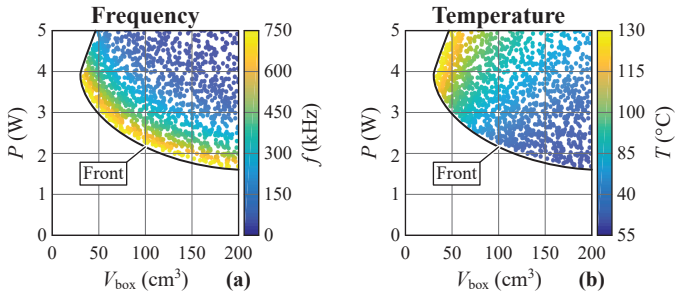

Fig. 7. Pareto fronts obtained with the analytical scaling laws (cf. (1) and (2)). (a) Operating frequency. (b) Operating temperature.

TABLE I
INDUCTOR SCALING LAWS: ANN PERFORMANCE.

Variable	Mode 1	Mode 2	Mode 3	Mode 4
Variable Trf.	no	no	no	log.
Normalization	no	no	yes	yes
Hidden Layer	none	sigm.	sigm.	sigm.
Output Layer	lin.	lin.	lin.	lin.
No. of Neurons	2	5+2	5+2	5+2
No. of Epochs	2	824	191	52
Max. Err. / P	62.7%	13.1%	2.9%	0.5%
RMS Err. / P	13.5%	3.2%	0.4%	0.1%
Max. Err. / T	32.0%	1.1%	2.0%	0.2%
RMS Err. / T	11.4%	0.2%	0.3%	0.0%

aforementioned dataset (2′000 samples). The inputs of the ANN are f and V_{box} and the outputs are P and T. The splitting between the training set and the test set is 80/20%. The ratio between the training subset and the validation subset is also 80/20%. The ANNs are trained with the mean square error as a metric. In order to overcome the non-deterministic nature of the ANN training process, each ANN is trained 100 times. For each training, the performances (number of epochs until convergence is achieved, RMS error, and maximum error) are evaluated and the median values over the 100 training cycles are computed. Tab. I shows the obtained results, which can be interpreted as follows:

- *Mode 1* - No variable transformation and no normalization is done. An ANN with a single linear layer is used (no hidden layer). Due to the simple nature of the network, convergence is achieved quickly. However, the performances of the ANN are low. This is expected since (1) and (2) are clearly not linear equations.
- *Mode 2* - In order to overcome this issue, a non-linear (sigmoid) hidden layer with 5 neurons is added. The performances of the ANN are improving but the number of training epochs drastically increases.
- *Mode 3* - The different variables feature very different orders of magnitude, which is always problematic for numerical methods. Therefore, the variables (inputs and outputs) are normalized for the ANN. A min-max normalization, which linearly maps a variable between zero and one is used. It can be seen that the normalization improves both the training speed and the fitting accuracy.
- *Mode 4* - From (1) and (2), it can be seen that applying a logarithmic variable transformation (inputs and outputs) could help the numerical conditioning of the problem. Therefore, the logarithm of the different variables is taken and the transformed variables are, again, normalized. It can be observed that the variable transformation further improves the convergence and the fitting performances.

It can be concluded that not only the structure of the neural network (number of layers, number of neurons, and activation functions) but also the training method (dataset splitting, algorithm, and error metric) and the variable handling (normalization and eventual variable transformation) are critical to obtain robust, accurate, and fast ANNs.

III. COMPARISON OF ANN-BASED WORKFLOWS

This section first reviews the different models and optimization methods used for multi-objective optimization. Then, different possibilities to integrate ANNs in the workflow are discussed.

A. Modeling and Multi-Objective Optimization

Inductor modeling includes many aspects: current and voltage waveforms, magnetic circuit, thermal behavior, winding losses, and core losses. The different models can be classified into three main categories [13], [14]:

- *Full-analytical models* - The models are based on analytical equations and feature closed-form analytical solutions [20], [22]–[26]. Such models are extremely simple but are too inaccurate for virtual prototyping.
- *Semi-numerical models* - The models are mostly based on analytical equations but do not feature explicit solutions [10], [11], [13], [27]. Such models represent an interesting trade-off between the accuracy and the computational cost.
- *Numerical models* - The parameters are extracted from numerical field simulations (e.g., FEM simulations) [25], [28], [29]. Despite their accuracy, such models are rarely used for optimization due to their heavy computational cost.

The role of the models is to map the design space into the performance space. For magnetic components, it is known that wide regions of the design space are mapped to a narrow region in the performance space, i.e., designs with very different parameters feature similar performances (design space diversity) [21], [29], [30]. This implies that, together with the model non-linearity, implicit constraints, and discrete variables, the optimization of magnetic components is a challenging task. Several methods are used [13], [14]:

- *Brute force grid search* - The design space is systematically sampled and all the combinations are tested [11], [13], [31]. This method is extremely simple and robust but the number of combinations scales exponentially with the number of variables (without additional filters and/or heuristics).
- *Deterministic optimization* - Algorithms such as gradient optimization, simplex method, or geometric

programming are used [10], [14]. These algorithms converge quickly but are problematic with respect to the design space diversity. Moreover, additional constraints exist about the objective functions (e.g., smoothness, posynomial function, no discrete variables).
- *Stochastic optimization* - Stochastic algorithms (e.g., genetic optimization, particle swarm, simulated annealing) represents a good trade-off between the robustness and the computational cost [12], [18], [31]–[33].

In order to obtain a fast, robust, and accurate optimization workflow, the following combinations of the aforementioned models and optimization methods are typically considered. Brute force grid search requires the evaluation of several million designs, which takes several hours or days of computations with a semi-numerical model [12], [29]. Therefore, the number of degrees of freedom is limited and the usage of a numerical model (e.g., FEM) is not a viable solution. Stochastic optimization still requires the evaluation of tens of thousands design possibilities, which still takes several minutes with a semi-numerical model and some hours or days with a numerical model [12], [31]. Therefore, a clear need can be identified to obtain a workflow that is simultaneously fast, accurate, and robust.

B. ANN-Based Modeling and Optimization

The flexibility and reduced computational cost of ANNs offer several opportunities for improving the aforementioned models and methods. Fig. 8 identifies several potential ANN-based workflows:

- *Workflow A* - The complete inductor model is replaced by ANNs (cf. Fig. 8(a)) [6], [15]. The model features a classification ANN, for handling discrete variables (e.g., core shape, core material, litz wire size) and several regression ANNs for the continuous variables. The advantages of this workflow are the reduced computational cost and the total independence from physics-based models. A challenge is the large number of data (simulations, measurements, and/or datasheets) required to train the ANNs with tens of input and output parameters. Moreover, it is difficult to guarantee that the ANNs are accurate for all the possible combinations. The last problem is the prediction of designs where few training samples exist (e.g., new geometry, new material).
- *Workflow B* - In order to overcome these difficulties, a solution is to split the model into different sub-models (cf. Fig. 8(b)) [34]–[37]. The usage of ANNs for sub-component models (e.g., magnetic, losses, thermal) reduces the complexity and facilitates the training of the ANNs.
- *Workflow C* - Another possibility is to use ANNs to improve the accuracy of an existing model (cf. Fig. 8(c)). For example, an analytical model can be improved with an ANN trained with FEM simulation results or measurements. This solution is robust and easy to integrate into a pre-existing model.
- *Workflow D* - Apart from offering advanced models, ANNs can also be used for the multi-objective optimization (cf. Fig. 8(d)) [14], [32], [38], [39]. This can be achieved by training the ANNs only with optimal designs or with advanced techniques such as ANN inversion, reinforcement learning ANN, or neuro-genetic optimization. However, due to the complex nature of inductor multi-objective optimization (design space diversity) and the wide variety of possible objectives (e.g., cost, volume, mass, efficiency, available components), a robust implementation of an ANN-based inductor optimization is a challenging task.

IV. Selected ANN-based Model

From the aforementioned methods and the goals defined in the introduction, an ANN-based inductor model, which features the same accuracy as 3D FEM simulations with a massively reduced computational cost, is presented in detail in the following.

A. Working Principle

The model uses regression MLP ANNs to replace the 3D FEM simulations at the sub-component level (cf. Fig. 8(b)). A first ANN is used to predict the magnetic parameters (e.g., inductance, magnetic flux, magnetic field) and a second ANN is modeling the thermal behavior (e.g., average and hotspot temperatures). Both ANNs are trained with a dataset generated using 3D FEM, whereas a simple analytical solution is used as a base-value for increasing the robustness of the regression (cf. Fig. 8(c)). Fig. 9 shows, schematically, the workflow where two distinct parts are identified: the ANN training and the inductor design evaluation.

The selected model represents a pragmatic approach with regression ANNs, which is robust and easy to extend [15], [36], [40], [41]. Since the model uses ANNs at the sub-component level, internal variables are accessible for inspection. Finally, the dataset required for the training can be easily generated and extended. Besides the low computational cost, the ANN-based workflow also offers the following advantage: the FEM solver is only required for generating the training dataset (cf. Fig. 9(a)) and not for evaluating inductor designs (cf. Fig. 9(b)). This implies that the users of the design tool do not need any FEM solver and/or powerful computer.

B. ANN Input Variables

Before analyzing the workflow in detail, the input variables of the ANNs (magnetic and thermal) should be defined. The following assumptions are made for the inductor: E-shaped core with an air gap, litz wire winding with homogeneously distributed strands, and forced convection cooling [42], [43]. However, the presented workflow can be easily extended to other types of inductors.

All the input variables are summarized in Tab. II. First, the geometry of the inductor is described, cf. Fig. 10. For training the ANN, it is numerically better to define scaled dimensions:

$$r_{\mathrm{w}} = h_{\mathrm{w}} / d_{\mathrm{w}}, \tag{3}$$

$$r_{\mathrm{c}} = z_{\mathrm{c}} / t_{\mathrm{c}}, \tag{4}$$

$$r_{\mathrm{cw}} = A_{\mathrm{c}} / A_{\mathrm{w}}, \tag{5}$$

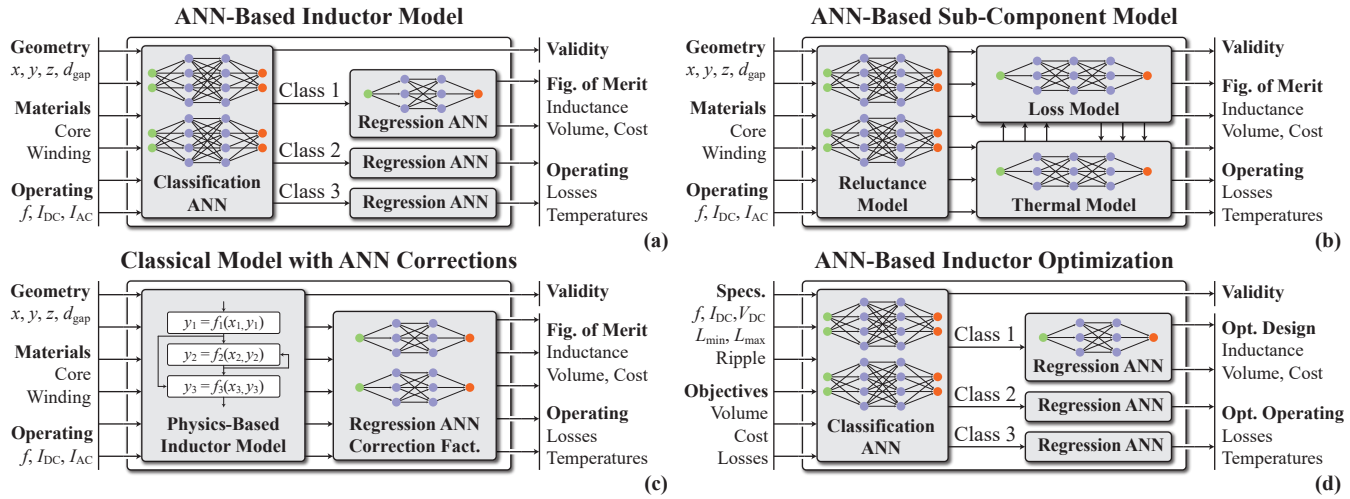

Fig. 8. (a) ANN-based inductor model using a classification ANN and several regression ANNs. (b) ANN-based inductor sub-component model. (c) Physics-based inductor model using an ANN to improve the results (correction factor). (d) ANN-based inductor optimization, generating optimal designs with respect to a given objective.

TABLE II
FEM/ANN INPUT VARIABLES.

Var.	Definition	Unit	Model
V_{box}	Boxed volume	cm^3	both
r_w	Winding window aspect ratio	p.u.	both
r_c	Core cross section aspect ratio	p.u.	both
r_{cw}	Winding and core areas ratio	p.u.	both
r_{gap}	Relative air gap length	p.u.	both
T_a	Ambient temperature	°C	thermal
h_c	Convection coefficient	$W/(m^2K)$	thermal
p_{tot}	Loss per surface	mW/cm^2	thermal
r_p	Winding and core losses ratio	pu	thermal
r_{sat}	Saturation current ratio	p.u.	magnetic
μ_c	Relative core permeability	p.u.	magnetic
β_c	Core Steinmetz coeff.	none	magnetic

TABLE III
FEM/ANN OUTPUT VARIABLES.

Var.	Definition	Unit	Model
$\Delta T_{c,avg}$	Average core temperature	K	thermal
$\Delta T_{c,max}$	Max. core temperature	K	thermal
$\Delta T_{w,avg}$	Average winding temperature	K	thermal
$\Delta T_{w,max}$	Max. winding temperature	K	thermal
$\Delta T_{i,max}$	Max. insulation temperature	K	thermal
L_{int}	Integral of BH, normalized	H	magnetic
B_{int}	Integral of B^β, normalized	T	magnetic
J_{int}	Integral of J^2, normalized	A/m^2	magnetic
H_{int}	Integral of H^2, normalized	A/m	magnetic

$$r_{gap} = d_{gap} / \sqrt{A_c}, \tag{6}$$

where, these four scaled variables, together with the boxed volume (V_{box}), define uniquely the inductor geometry.

For the thermal simulations, the following input variables are added: the ambient temperature (T_a), the convection coefficient (h_c), and the generated losses. The losses are also normalized:

$$p_{tot} = (P_w + P_c) / A_{box}, \tag{7}$$

$$r_p = P_w / P_c, \tag{8}$$

where A_{box} is the boxed area of the component, P_w the winding losses, and P_c the core losses.

For the magnetic simulations, the additional input parameters are the core permeability (μ_c), the core Steinmetz parameter for the flux density (β_c), and the current excitation. All the magnetic simulations are done with a single turn and the current is normalized with the saturation current:

$$r_{sat} = \hat{I} / I_{sat} = \left(\mu_0 \hat{I}\right) / \left(2 d_{gap} B_{sat}\right), \tag{9}$$

where $\hat{I}$ is the peak winding current, I_{sat} the saturation current, and B_{sat} the saturation flux density. For the scaling with respect to the saturation current, the fringing field and the core reluctance are neglected.

C. ANN Output Variables

For the output variables, ANNs would have the ability to learn directly from the field patterns generated by the 3D FEM simulations (magnetic and temperature fields) [1], [44]. However such algorithms are complex and involve very large datasets. Furthermore, for the evaluation of inductor designs, the complete field patterns are not required, some key figures of merit, which are summarized in Tab. III, are sufficient.

For the thermal simulations, the different maximum and average temperature elevations are extracted. The maximum (hotspot) temperatures are used for ensuring the thermal limits and the average temperatures for calculating the material properties (core and winding) used for the loss computations. It has to be noted that, for linear thermal models, the ambient temperature (T_a) does not have any impact on the temperature elevations.

For the magnetic simulations, the definition of the output variables is not straightforward. The inductance (L), the core

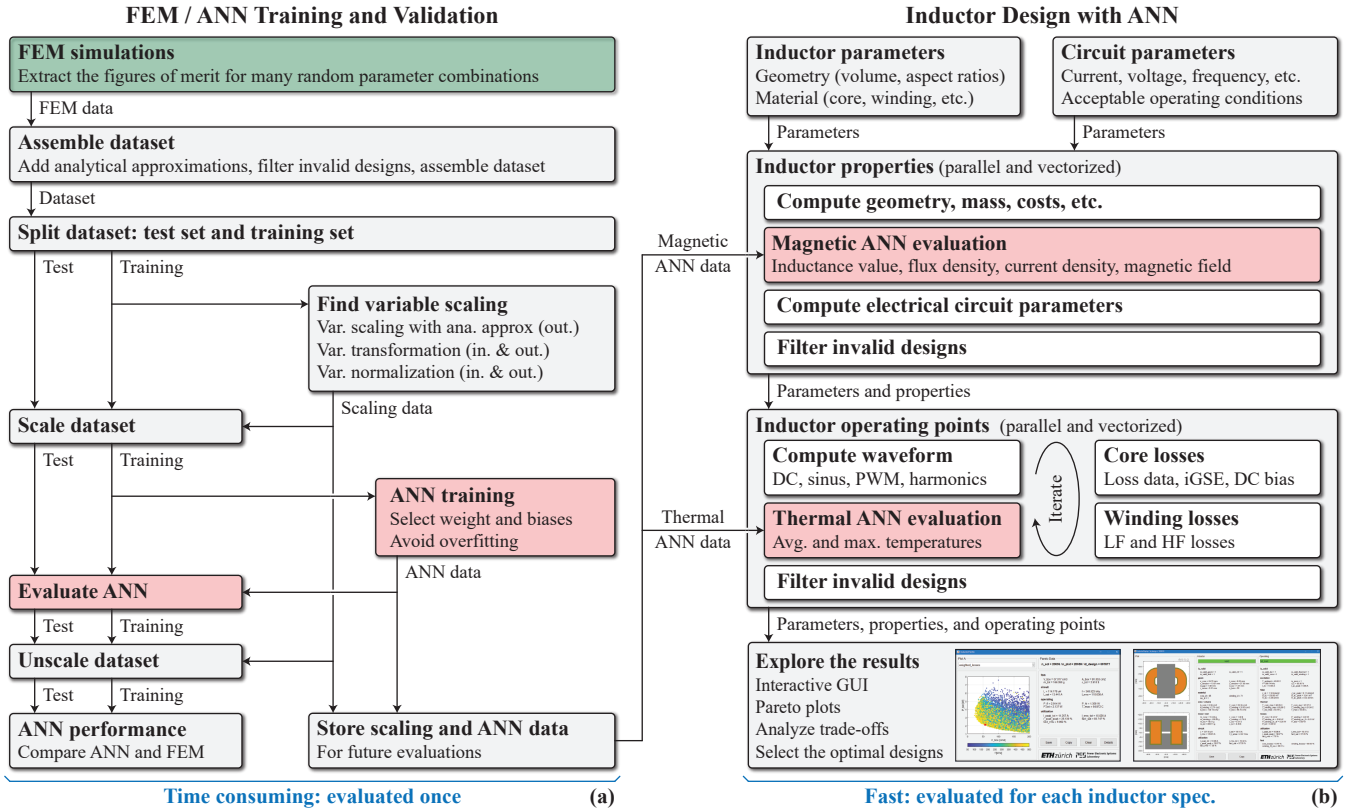

Fig. 9. (a) Workflow for generating the data and training the ANNs for the thermal and magnetic models. (b) Inductor design workflow using the two ANNs. The step requiring a FEM solver is highlighted in green and the ANN-based steps in red.

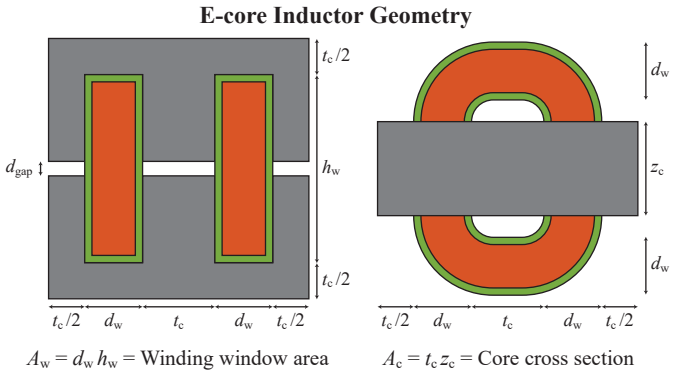

Fig. 10. Considered inductor geometry with an E-shaped core with an air gap and a litz wire winding.

losses (P_c), the low-frequency winding losses ($P_{w,LF}$) and the high-frequency winding losses ($P_{w,HF}$) can be defined as

$$L = \frac{1}{\hat{I}^2} \iiint_{\text{all}} \hat{B}\hat{H}\mathrm{d}V, \tag{10}$$

$$P_c = \iiint_{\text{core}} k_c f^{\alpha_c} \hat{B}^{\beta_c}\mathrm{d}V, \tag{11}$$

$$P_{w,LF} = \iiint_{\text{wdg.}} \frac{1}{2k_w\sigma_w}\hat{J}^2\mathrm{d}V, \tag{12}$$

$$P_{w,HF} = \iiint_{\text{wdg.}} \frac{\pi^2\mu_0^2 d_s^2 k_w \sigma_w f^2}{8}\hat{H}^2\mathrm{d}V, \tag{13}$$

where $\hat{I}$ is the peak winding current, $\hat{B}$ the magnetic flux density, $\hat{H}$ the magnetic field, $\hat{J}$ the current density, and f the operating frequency. The parameters k_c, α_c, and β_c are the Steinmetz parameter of the core material [22]. The winding is defined by: k_w the filling factor, σ_w the conductivity, and d_s the litz wire strand diameter [22]. Please note that, for the sake of simplicity, these equations are defined for pure AC sinusoidal currents. However, more complex waveforms (e.g., DC biases, triangular currents) or loss models (e.g. iGSE for core losses, Bessel functions for winding losses) can be used without changing the definition of the output variables [22], [29], [45], [46]. The current ($\hat{I}$) and the number of turns (N) can be factorized, which leads to

$$L = N^2 L_{\text{int}}, \tag{14}$$

$$P_c = V_c k_c f^{\alpha_c} \left(N\hat{I}B_{\text{int}}\right)^{\beta_c}, \tag{15}$$

$$P_{w,LF} = V_w \frac{1}{2k_w\sigma_w}\left(N\hat{I}J_{\text{int}}\right)^2, \tag{16}$$

$$P_{w,HF} = V_w \frac{\pi^2\mu_0^2 d_s^2 k_w\sigma_w f^2}{8}\left(N\hat{I}H_{\text{int}}\right)^2. \tag{17}$$

where V_c and V_w are the core and winding volumes, respectively. With these equations, the normalized energy and fields can be defined as

$$L_{\text{int}} = \frac{1}{\left(N\hat{I}\right)^2}\iiint_{\text{all}} \hat{B}\hat{H}\mathrm{d}V, \tag{18}$$

$$B_{\text{int}} = \frac{1}{N\hat{I}} \left(\frac{1}{V_{\text{c}}} \iiint_{\text{core}} \hat{B}^{\beta_{\text{c}}} \mathrm{d}V \right)^{\frac{1}{\beta_{\text{c}}}}, \tag{19}$$

$$J_{\text{int}} = \frac{1}{N\hat{I}} \left(\frac{1}{V_{\text{w}}} \iiint_{\text{wdg.}} \hat{J}^2 \mathrm{d}V \right)^{\frac{1}{2}}, \tag{20}$$

$$H_{\text{int}} = \frac{1}{N\hat{I}} \left(\frac{1}{V_{\text{w}}} \iiint_{\text{wdg.}} \hat{H}^2 \mathrm{d}V \right)^{\frac{1}{2}}. \tag{21}$$

With these definitions, the spatial effects (e.g., flux crowding at the corner of the core, air gap fringing field) are taken into account in the inductance and loss computations. Therefore, the normalized variables are used as the output variables for the magnetic simulations and the corresponding ANN. It has to be noted that, for linear magnetic models, the current excitation (r_{sat}) does not impact the normalized output variables.

D. ANN Training

Fig. 9(a) shows the process used to generate the dataset and train the different ANNs. The operation comes with a high computational cost but is executed only once (and not for every inductor design). Moreover, the dataset can be generated in parallel and/or with a distributed computing platform.

At first, many 3D magnetic and thermal simulations are done with different random combinations of the input variables (cf. Tab. II). The output variables are extracted (cf. Tab. III) [47]. In a second step, simple analytical approximations of the output variables are computed. For the thermal model, a lumped equivalent circuit is used [13], [27]. The analytical magnetic model is based on a reluctance circuit (i.e., inductance and magnetic flux density) and analytical approximations with Ampère's circuital law (i.e., magnetic field and current density) [22], [48]. Afterwards, the invalid samples (e.g., non-manufacturable geometries, design with very poor figures of merit) are filtered out and the dataset is obtained.

The resulting data set is split into a test set and a training set. The different variables are then prepared for the ANNs. This process is described in the following:

1) *Variable scaling* - The output variables are scaled with respect to the analytical approximations such that the ANNs are just predicting correction factors between the 3D FEM simulations and the analytical approximations. The scaling is optional and can be described as

$$v \leftarrow \frac{v_{\text{FEM}}}{v_{\text{Ana. Approx}}}. \tag{22}$$

2) *Variable transformation* - In a second step, variable transformations can be applied to the different input and output variables to improve the numerical conditioning. More concretely, a logarithmic transformation is applied to the variables spanning over several orders of magnitude:

$$v \leftarrow \log(v). \tag{23}$$

3) *Variable normalization* - Finally, all the input and output variables are normalized. Typically, a min-max normalization is used which is linearly mapping the variable between zero and one:

$$v \leftarrow \frac{v - \min(v)}{\max(v) - \min(v)}. \tag{24}$$

Afterwards, the scaled, transformed, and normalized dataset is used for the training of the thermal and magnetic ANNs [49]–[51]. Finally, the performances of the ANNs are evaluated using the training set and the test set. Additionally, the ANN parameters (e.g., structure, weights) are saved.

E. Inductor Design

Fig. 9(b) shows the process used to compute inductor designs. All the steps feature a reduced computational cost. Moreover, the complete workflow is parallelized and vectorized. This model offers 30 input and 40 output variables and is, therefore, flexible and versatile.

First, the basic inductor properties (e.g., geometry, mass, cost) of the designs are computed. Afterwards, the magnetic ANN is evaluated and the magnetic properties are extracted (e.g., inductance, saturation current, magnetic and thermal field patterns). The designs with properties which are incompatible with the given specifications are filtered out.

In a second step, the different operating points (e.g. full load, partial load) are computed. The waveforms are generated and analyzed (e.g, DC bias, sinusoidal, triangular, Fourier harmonics). The core losses (different materials) are computed with the iGSE and detailed loss data that take into consideration the impact of the frequency, AC flux density, DC flux density, and temperature [13], [42], [46]. The winding losses (different litz wire strandings) are computed, including the proximity losses and the impact of the harmonics [22], [45]. For both the core and winding losses, the field patterns are coming from the magnetic ANN. The temperatures are evaluated with the thermal ANN. Iterations are made between the loss and thermal models in order to reach the steady-state (coupled loss-thermal model) [13]. Again, the invalid designs (e.g., saturation, thermal limit) and the designs with poor performances (e.g., losses) are filtered out.

Finally, the results are displayed in an interactive tool that allows the exploration of the Pareto fronts and the different trade-offs (e.g., cost, efficiency, volume). Interesting designs can be selected and inspected (geometry, properties, and operating points).

V. Performances of the ANN-Based Model

In this section, the performances of the presented workflow are analyzed. More precisely, the accuracy of the ANNs, the training parameters, and the computational cost are investigated.

A. Considered Datasets

Tab. IV depicts the considered ranges for the input variables, which cover most of inductor designs. For generating the dataset, random combinations of the inputs (in the

TABLE IV
FEM/ANN INPUT VARIABLE RANGES.

$V_{box} \in [10,1000]\,cm^3$	$r_w \in [2,4]$	$r_c \in [1,3]$
$r_{cw} \in [0.3,3]$	$r_{gap} \in [0.005,0.3]$	$T_a \in [25,65]\,°C$
$h_c \in [15,30]\,W/(m^2K)$	$p_{tot} \in [1,1000]\,mW/cm^2$	$r_p \in [0.02,50]$
$r_{sat} \in [0.001,1.0]$	$\mu_c \in [1500,3000]$	$\beta_c \in [2,2.8]$

specified ranges) are generated. For the variables spanning over several orders of magnitude, the random samples are generated on a logarithmic scale. For the generation of the samples, the following procedure is used:

- First, all the points at the edges (upper and lower limit of the ranges) of the dataset, which are unlikely to appear with random sampling, are considered. This corresponds to $2^9 = 512$ and $2^8 = 256$ samples for the thermal and magnetic ANNs, respectively. These samples at the edges are, strictly speaking, not required but improve the validation of the ANNs by ensuring that the extreme cases are included.
- All the remaining samples are chosen randomly. This is one of the great strengths of ANNs: they do not require a regular sampling, which is advantageous with problems with many input variables.

At the end, each dataset (thermal and magnetic) consists of 20′000 samples. The number of valid samples (used for the ANN training) are 18′522 and 18′444 for the thermal dataset and magnetic dataset, respectively.

B. ANN Parameters

For the ANN training, the 3D FEM outputs are scaled with the analytical approximations, the variables spanning over several orders of magnitude are transformed into a logarithmic scale, and all the variables are normalized (min-max normalization). The splitting between the training set and the test set is 80/20 %. The ratio between the training subset and the validation subset is also 80/20 %. The training algorithm uses Levenberg-Marquardt backpropagation with the mean square error as a metric [49]. The ANNs feature two hidden layers (with 10 neurons each, sigmoid activation function) and an output layer (linear activation function).

C. Achieved Accuracy

Fig. 11 and Fig. 12 show resulting performances for the thermal model and magnetic model, respectively. It shows that even if the deviation between the analytical approximations and the 3D FEM simulations is, for some samples, above 100 %, the error between the ANNs and the 3D FEM simulations is always below 3 %. As expected, the analytical approximations are particularly inaccurate for the magnetic field in the winding window. This is due to the air gap fringing field, which is difficult to compute analytically. It should be noted, that since overfitting is explicitly prevented during the training, the error of the training set is very similar to the error of the test set. For this reason, the error metrics are computed for all the samples together (test set and training set). It can be concluded that the ANNs are accurate in the complete range and do not produce any outlier data.

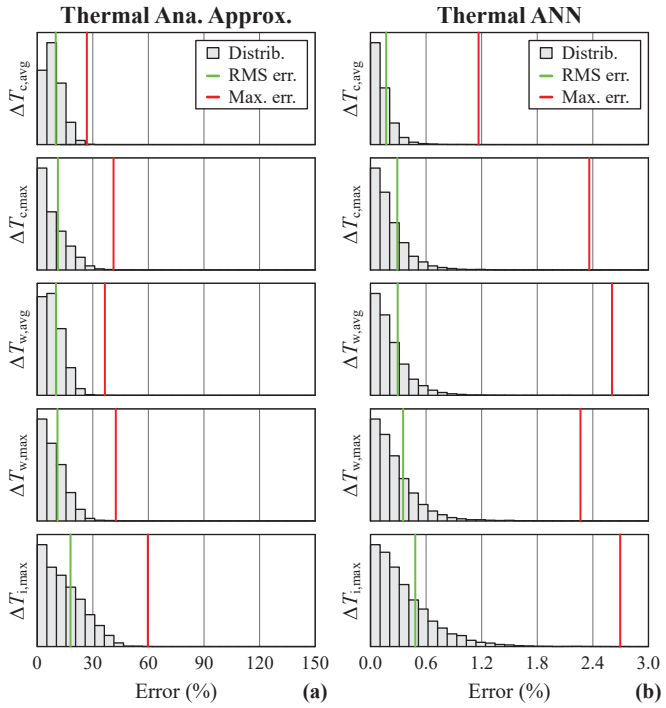

Fig. 11. Error distribution (18′522 samples) for the thermal model. (a) Deviation between the analytical approximations and the 3D FEM simulations. (b) Deviation between the ANN outputs and the 3D FEM simulations. The RMS error and the maximum error (over all the samples) are indicated.

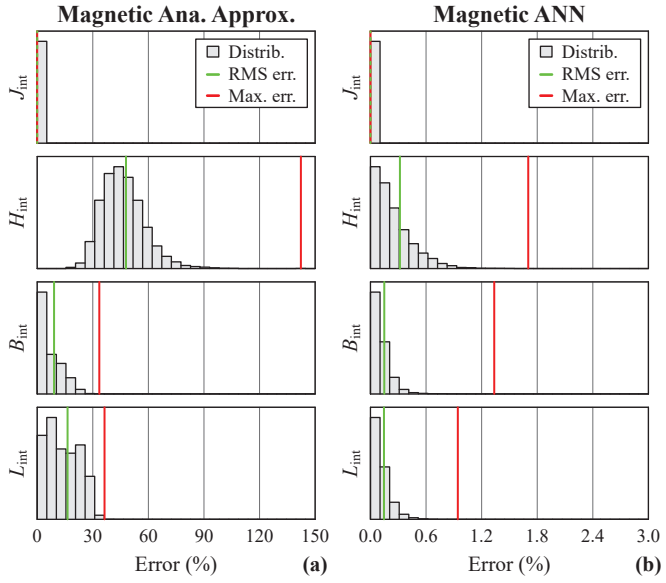

Fig. 12. Error distribution (18′444 samples) for the magnetic model. (a) Deviation between the analytical approximations and the 3D FEM simulations. (b) Deviation between the ANN outputs and the 3D FEM simulations. The RMS error and the maximum error (over all the samples) are indicated.

D. Training Reproducibility

The ANN training process is not deterministic (dataset splitting and initial values). For this reason, the complete

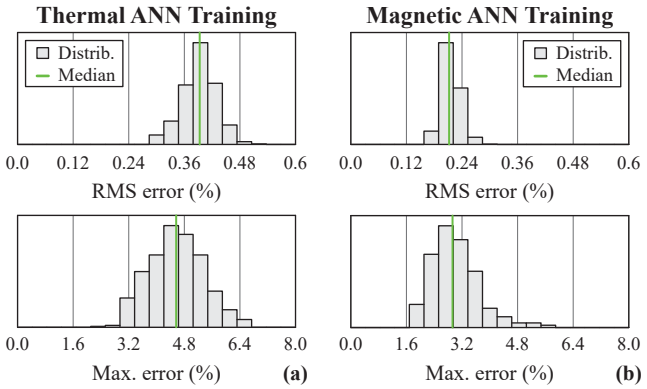

Fig. 13. Performances of the ANNs over 200 trainings. (a) Thermal model ANN. (b) Magnetic model ANN. The selected metrics are the RMS error and the maximum error (over all the samples and all the variables).

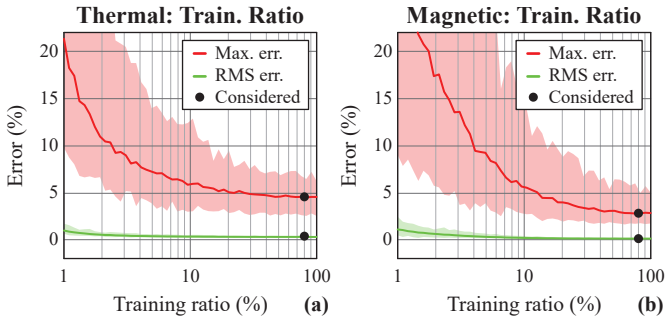

Fig. 14. Performances of the ANNs for different splitting ratio between the training set and the test set. (a) Thermal model ANN. (b) Magnetic model ANN. The selected metrics are the RMS error and the maximum error (over all the samples and all the variables). The shaded areas represent the variations obtained over 200 trainings and the solid lines the median values.

training process of the ANNs is repeated 200 times. Afterwards, the distribution and the median value of the error metrics (RMS error and maximum error, over all the samples) are computed, cf. Fig. 13. It can be observed that the maximum error is, for all the trainings, below 7%. Nevertheless, it is advised to train the ANNs several times to check the reproducibility and to pick the ANNs with the best performances.

E. Size of the Dataset

Another important parameter is the size of the dataset used for the training. Fig. 14 shows the ANN performances (median over 200 trainings) for different training ratios (splitting ratio between the training and the test). The splitting ratio between the training subset and the validation subset remains 80/20%. It can be observed that good performances can be achieved with a training ratio of 10/90% (compared to the nominal value of 80/20%). This implies that the ANNs can be correctly trained with smaller datasets: 5′000 3D FEM simulations (instead of 20′000) would be sufficient.

In comparison, if a multivariate interpolation with a regular grid would be used in place of ANNs, the required dataset would be much larger. A regular grid with 6

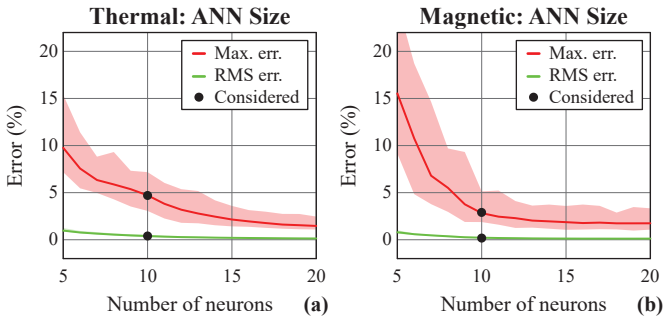

Fig. 15. Performances of the ANNs for different numbers of neurons (per hidden layer). (a) Thermal model ANN. (b) Magnetic model ANN. The selected metrics are the RMS error and the maximum error (over all the samples and all the variables). The shaded areas represent the variations obtained over 200 trainings and the solid lines the median values.

TABLE V
ANN PERFORMANCES: SCALING WITH ANA. APPROX.

Variable	**Median error metrics over 200 trainings**			
Type	thermal	thermal	magnetic	magnetic
Scaling	yes	no	yes	no
RMS Err.	0.4%	0.5%	0.2%	0.5%
Max. Err.	4.6%	5.3%	2.9%	4.6%

values per variable would lead to 10.1 millions samples for the thermal model (9 input variables) and 1.7 millions samples for the magnetic model (8 input variables). This further highlights the advantages and potentials of ANNs for improving inductor models against classical methods such as interpolation.

F. Number of Neurons

A last critical degree of freedom is the structure of the ANNs (number of hidden layers, number of neurons, and activation functions). Two hidden layers with sigmoid activation functions have been selected since such ANNs are well adapted for non-linear fitting [8], [9]. It has to be noted that the choice of the ANN structure is not unique: many different structures may give good performances. Fig. 15 highlights the ANN performances (median over 200 trainings) for different numbers of neurons (per hidden layer). It can be seen that the selected structure (10 neurons per hidden layer) represents a good trade-off between the achieved performances and the computational cost.

G. Scaling with Analytical Approximations

For the ANN trainings, the output variables are scaled with the analytical approximations such that the ANNs are only predicting relative correction factors. Tab. V compares the achieved performances with and without the scaling (median over 200 trainings). It appears that good performances are also achievable without scaling the 3D FEM results with the analytical approximations. Nevertheless, the performances are still improved by the scaling. Moreover, having analytical approximations is very useful for inspecting and debugging the model. For these reasons, in this work, the scaling of the output variables has been considered.

H. Computational Cost

For the benchmarking, a mid-range laptop (Intel Core i7-8650U with 16GB RAM) is used. However, all the workflow (cf. Fig. 9) is parallelized and, therefore, can be massively accelerated with more powerful hardware and/or with a distributed computing platform.

It has been shown (cf. Fig. 14) that 5′000 samples are sufficient for training the ANNs. The 3D FEM models have been optimized in order to obtain a good trade-off between the computation time and the achieved accuracy. More specifically, the models are exploiting the symmetry planes, use a customized mesh size, and a carefully parametrized solver [47]. With this number of samples, the generation of the dataset and the training of the ANNs (cf. Fig. 9(b)) takes 160 hours. Between the 3D FEM simulations and the ANN trainings, the 3D FEM simulations represent, clearly, the bottleneck of the workflow. It should be noted that the generation of the dataset is completely automated and does not require any human supervision. Moreover, this operation is only required once.

Nevertheless, if required, the computational cost could be greatly reduced by using 2D FEM simulations, which are less accurate but much faster (typically 100 times faster). However, depending on the geometrical aspect ratio of the components, the error between 2D and 3D simulation can be significant (up to 30 %). The error is mainly due to the field distribution and heat flow close to the winding head of the inductor, which cannot be easily modeled in 2D. Therefore, if 2D models are used for the optimization, the selected design should be checked with a 3D model to ensure the validity of the optimization.

For the computation of inductor designs (cf. Fig. 9(a)), all the steps feature a reduced computational cost. The computation of the inductor properties takes 3.3 µs (or 300′000 designs per second) and the computation of an operating point takes 20 µs (or 50′000 designs per second). The computing speed and the parallel operation make this model particularly useful for brute force grid search of genetic optimization algorithms, which require the evaluation of many designs (cf. Section III-A).

I. Drawbacks

The first drawback of the proposed model is the increased complexity caused by the addition of the ANNs (cf. Fig. 9(a)). However, this increased complexity is, partially, compensated by the simplification of the inductor design evaluation process (cf. Fig. 9(b)).

The second shortcoming of the method is the requirement to generate a dataset for training the ANNs. It also implies that an extension of the model (e.g., additional parameters, materials, geometries) will require an adaptation of the underlying dataset and retraining of the ANNs.

VI. Case Study: DC-DC Buck Inductor

With the presented workflow and ANNs, different types of inductors can be optimized (e.g. PFC inductors, DC-DC inductors, resonant inductors). In this paper, the inductor of a 2 kW DC-DC buck converter is optimized and measured.

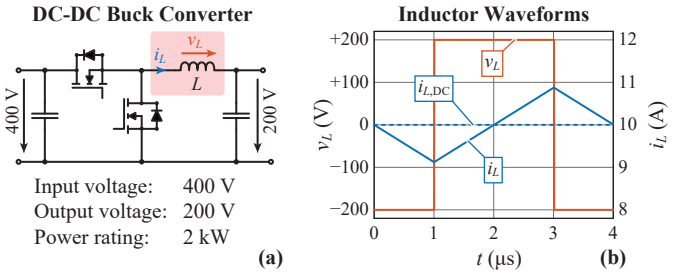

Fig. 16. (a) Considered DC-DC converter. (b) Current and voltage applied to the inductor (shown for 375 kHz and 150 µH).

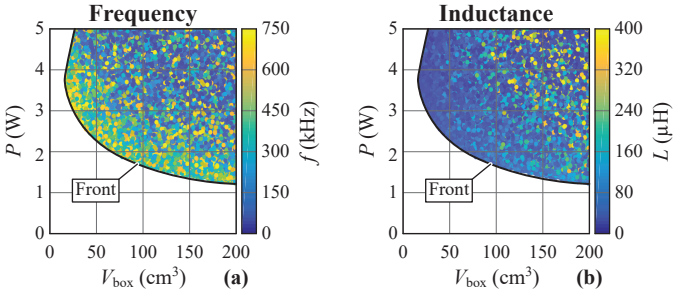

Fig. 17. Obtained Pareto plots and fronts (boxed volume vs. losses) with the ANN-based workflow. (a) Operating frequency. (b) Inductance value.

A. Specifications

Fig. 16 shows the considered 2 kW DC-DC converter stepping down from 400 V to 200 V. The following specifications are chosen: 55 °C ambient temperature, 100 µm litz wire stranding, and TDK N87 core material [42]. This corresponds to the specifications used in [21].

The following variables are optimized the boxed volume ($V_{box} \in [10,200]\,\mathrm{cm}^3$), the operating frequency ($f \in [50,750]\,\mathrm{kHz}$), the geometrical aspect ratios, the air gap length, and the number of turns. For the core and litz wire geometries, custom designs are also considered, in addition to the standard shapes. Due to the extreme computational speed of the ANN-based model, a brute force grid search approach is used.

B. Pareto Fronts

Fig. 17 depicts the Pareto plots and fronts obtained with the ANN-based workflow. For the optimization 5 million designs are considered and 0.7 million designs are valid (e.g., saturation, thermal limit). All the designs are computed in 40 s. In the Pareto fronts, the design space diversity (very different designs located at the same region of the Pareto fronts) can be observed for both the operating frequency and the inductance value [21], [29], [30].

Fig. 18 shows the deviation between the values obtained with the analytical approximations with respect to the ANN-based workflow. It can be seen that significant deviations exist (up to 50 %), highlighting the superiority and usefulness of the ANN-based model compared to analytical approximations. The fact that the analytical approximations are more accurate close to the Pareto fronts (with some exceptions) can be explained by the fact that these designs have

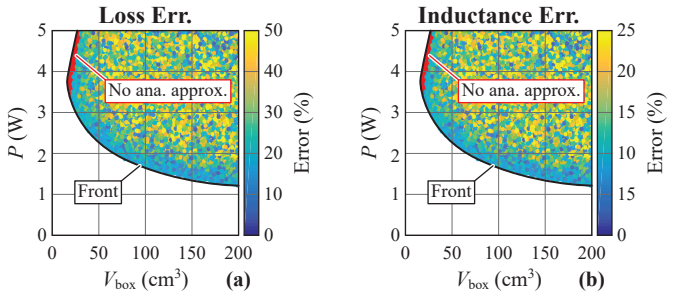

Fig. 18. Deviation between the analytical approximations and the ANN-based workflow. (a) Loss deviation. (b) Inductance deviation. The red area represents the regions where no valid design exists with the analytical approximations.

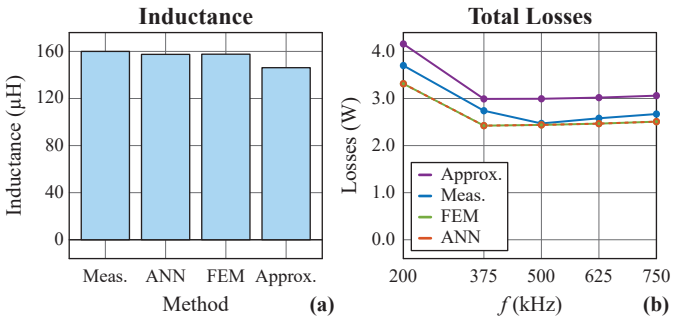

Fig. 19. Measured values compared with the ANN-based workflow, 3D FEM simulations, and analytical approximations. (a) Inductance values. (b) Inductor losses.

TABLE VI
MEASURED INDUCTOR PROTOTYPE.

Variable	Value
Core	E 55/28/21, TDK N87
Air gap	800µm / 2 × 400µm
Conductor	Litz wire / 900 × 100µm
Winding	16 turns / 2 layers
Volume	130cm^3 / boxed volume
Inductance	150µH (datasheet value)
Waveform	DC current + AC sin. voltage
Stress	10A DC + 222V AC RMS
Frequency	200kHz to 750kHz
Ambient	55°C / small air flow
Prototype	

reduced fringing field, proximity losses, and temperature elevations [21]. Therefore, such designs are, typically, less sensitive to the inaccuracies of the analytical approximations.

C. Measured Prototype

Tab. VI shows the properties of the selected design, which is the same as in [21]. It has to be noted that, due to practical restrictions (available core shapes and litz wire geometries), the selected design is not on the Pareto front but close to the front. The prototype is measured, at different frequencies, with DC currents and sinusoidal voltages (with the same volt-second product as applied by the DC-DC buck converter).

The inductance is measured with an Agilent 4924A precision impedance analyzer (measurement uncertainty below ±0.25µH) [52]. The losses are measured with a custom calorimetric setup (measurement uncertainty below ±0.1W) [21]. Fig. 19 compares the measured values with the ANN-based workflow, 3D FEM simulations, and analytical approximations. It can be seen, that the ANN-based workflow, as expected, matches perfectly with the 3D FEM simulations (less than 0.6% deviation for the inductance and the losses). The deviations between the analytical approximations and the measurements are 8.6% for the inductance and 21.2% for the losses. With the ANN-based workflow, the errors are reduced to 1.4% and 11.5%, respectively.

VII. CONCLUSION

This paper examines the potential of ANNs for modeling and optimization of power electronic components, and more specifically, inductors. A promising workflow, which consists of using MLP ANNs for obtaining a fast and accurate inductor model, is selected. More specifically, regression ANNs (trained with 3D FEM simulations) are used for the magnetic and thermal models. This workflow represents a pragmatic solution. The generation of the training dataset is straightforward and the model is accurate, robust, and easy to extend or combine with other models and/or optimization algorithms. The complete implementation is available as an open-source software.

The ANN-based workflow is able to simulate inductors with different geometries, winding strandings, core materials, waveforms, etc. The model offers 30 input and 40 output variables and considers advanced effects such as the 3D magnetic and thermal field patterns, detailed core loss data, proximity winding losses, coupled loss-thermal model, etc. Furthermore, the proposed model features the same accuracy as 3D FEM simulations (less than 3% deviation) with a computational cost reduced by several orders of magnitude (computation of 50′000 per second). Finally, the described computation techniques are successfully applied for designing the inductor of a 2kW DC-DC converter.

Overall, the paper demonstrates how machine learning can be combined with classical power electronic models in order to improve their accuracy and reduce the computational cost, which is particularly interesting for magnetic components.

ACKNOWLEDGMENT

The authors would like to thank G. Mauro, M. Kasper, K. Leong, and G. Deboy from Infineon Technologies Austria AG for the fruitful discussions about the role of artificial intelligence in power electronic optimization.

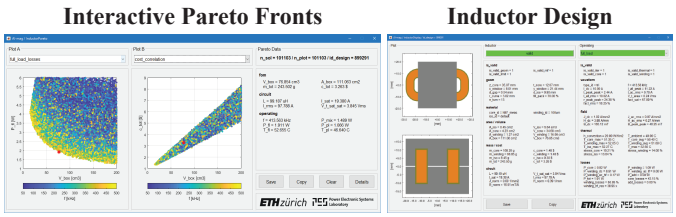

Fig. 20. (a) Graphical user interface enabling an interactive exploration of the Pareto fronts. (b) Geometry and properties of a specific design.

APPENDIX

This appendix describes "AI–mag", the software implementation of the proposed workflow (cf. Fig. 9), which is available under an open source license [19]. All the source code is available together with the required data (e.g., FEM models, core loss data). The dataset used in this paper (3D FEM simulations) and the trained ANNs are also provided.

A. Typical Design Steps

The tool enables many different workflows for modeling and optimizing inductors. The most straightforward method (brute force grid search) can be summarized as follow:

- The different variables to be optimized and the corresponding ranges are chosen (e.g., geometry, material, frequency). The fixed parameters and the applied stress (e.g, waveforms) are defined.
- Filters are defined in order to prevent the computation and/or storage of undesirable designs.
- With the provided data, the tool is automatically generating and computing the different designs.
- The different Pareto plots and/or objective functions are defined (e.g., volume, mass, cost, efficiency) and the results (trade-offs) can be displayed in a graphical user interface, cf. Fig. 20.

B. Used Technologies

The tool is mainly written in MATLAB with some restricted dependencies to COMSOL and Python (13'000 lines of code) [47], [49]–[51]. MATLAB is communicating with both COMSOL and Python over TCP/IP (client/server model). All the code can take advantage of multi-core machines. Furthermore, the dataset generation (3D FEM simulations) can be done with a distributed computing platform (cloud computing or high-performance computing cluster).

COMSOL is used for generating the dataset (3D FEM simulations). It should be noted that COMSOL is only required to generate the dataset, not for running the inductor model (cf. Fig. 9). Moreover, the code is written such that COMSOL can be easily replaced with another FEM solver.

For the regression ANNs, a versatile interface is proposed and is taking care of the dataset handling (e.g., splitting, scaling, normalization), ANN training, ANN validation, performance visualization, and ANN evaluation. For the ANN engine, two different libraries are available:

- *MATLAB* - Using the built-in MATLAB Deep Learning Toolbox as an ANN engine.
- *Python* - Using the Keras ANN API with TensorFlow as an ANN engine backend.

C. Extension of the Tool

Finally, the tool is made such that it is easy to extend its capability (object-oriented programming). The addition of new other inductor types (e.g., core shapes, winding type), magnetic components (e.g., transformers, chokes), or optimization methods (e.g., genetic algorithm) should not represent a problem. ANNs could also be used for other parts of the workflow (e.g., core losses, winding losses).

REFERENCES

[1] S. Skansi, *Introduction to Deep Learning: from Logical Calculus to Artificial Intelligence*. Springer, 2018.
[2] Y. L. Murphey, M. A. Masrur, Z. Chen, and B. Zhang, "Model-Based Fault Diagnosis in Electric Drives Using Machine Learning," *IEEE/ASME Trans. Mechatronics*, vol. 11, no. 3, pp. 290–303, 2006.
[3] B. Li, M. Y. Chow, Y. Tipsuwan, and J. C. Hung, "Neural-Network-Based Motor Rolling Bearing Fault Diagnosis," *IEEE Trans. Ind. Electron.*, vol. 47, no. 5, pp. 1060–1069, 2000.
[4] B. R. Lin and R. G. Hoft, "Power Electronics Inverter Control with Neural Networks," in *Proc. of the IEEE Applied Power Electronics Conf. and Expo. (APEC)*, Mar. 1993.
[5] B. K. Bose, "Neural Network Applications in Power Electronics and Motor Drives — An Introduction and Perspective," *IEEE Trans. Ind. Electron.*, vol. 54, no. 1, pp. 14–33, 2007.
[6] G. Tsekouras, S. Kiartzis, A. G. Kladas, and J. Tegopoulos, "Neural Network Approach Compared to Sensitivity Analysis Based on Finite Element Technique for Optimization of Permanent Magnet Generators," *IEEE Trans. Magn.*, vol. 37, no. 5, pp. 3618–3621, 2001.
[7] A. Sadeghian and J. D. Lavers, "Implementation of Knowledge-Based System for Iron Core Inductor Design," *IEEE Trans. Magn.*, vol. 40, no. 6, pp. 3495–3504, 2004.
[8] R. Atienza, *Advanced Deep Learning with TensorFlow 2 and Keras*. Packt Publishing, 2020.
[9] F. Chollet, "Deep Learning with Python," 2017.
[10] A. Stupar, J. A. Taylor, and A. Prodic, "Posynomial Models of Inductors for Optimization of Power Electronic Systems by Geometric Programming," in *Proc. of the IEEE Control and Modeling for Power Electronics Conf. (COMPEL)*, Jun. 2016.
[11] R. M. Burkart and J. W. Kolar, "Comparative Life Cycle Cost Analysis of Si and SiC PV Converter Systems Based on Advanced η-ρ-σ Multiobjective Optimization Techniques," *IEEE Trans. Power Electron.*, vol. 32, no. 6, pp. 4344–4358, 2016.
[12] A. Garcia-Bediaga, I. Villar, A. Rujas, L. Mir, and A. Rufer, "Multiobjective Optimization of Medium-Frequency Transformers for Isolated Soft-Switching Converters Using a Genetic Algorithm," *IEEE Trans. Power Electron.*, vol. 32, no. 4, pp. 2995–3006, 2017.
[13] R. Burkart, "Advanced Modeling and Multi-Objective Optimization of Power Electronic Converter Systems," Ph.D. dissertation, ETH Zürich, 2016.
[14] T. Guillod and J. W. Kolar, "From Brute Force Grid Search to Artificial Intelligence: Which Algorithms for Magnetics Optimization?" in *Industry Session at the IEEE Energy Conversion Congr. and Expo. (ECCE USA)*, Mar. 2020.
[15] S. Shimokawa, H. Oshima, K. Shimizu, Y. Uehara *et al.*, "Fast 3-D Optimization of Magnetic Cores for Loss and Volume Reduction," *IEEE Trans. Magn.*, vol. 54, no. 11, pp. 1–4, 2018.
[16] Frenetic, "Better Magnetics," Jan. 2020, [Presentation].
[17] J. M. Molina, "Transformer Design Consideration for Full Bridge Phase," in *Industry Session at the IEEE Energy Conversion Congr. and Expo. (ECCE USA)*, Mar. 2020.
[18] K. Watanabe, F. Campelo, Y. Iijima, K. Kawano *et al.*, "Optimization of Inductors Using Evolutionary Algorithms and its Experimental Validation," *IEEE Trans. Magn.*, vol. 46, no. 8, pp. 3393–3396, 2010.
[19] T. Guillod, *AI-mag: Inductor Optimization with FEM/ANN*, https://ai-mag.github.io, ETH Zurich, Power Electronic Systems Laboratory, 2020 (initial version).
[20] J. W. Kolar, J. Biela, and J. Miniböck, "Exploring the Pareto Front of Multi-Objective Single-Phase PFC Rectifier Design Optimization - 99.2% Efficiency vs. 7kW/dm^3 Power Density," in *Proc. of the IEEE Int. Power Electronics and Motion Control Conf. (PEMC)*, May 2009.

[21] P. Papamanolis, F. Krismer, and J. W. Kolar, "Minimum Loss Operation of High-Frequency Inductors," in *Proc. of the IEEE Applied Power Electronics Conf. and Expo. (APEC)*, Mar. 2018.
[22] M. Leibl, "Three-Phase PFC Rectifier and High-Voltage Generator," Ph.D. dissertation, ETH Zürich, 2017.
[23] J. A. Ferreira, *Electromagnetic Modelling of Power Electronic Converters*. Springer, Science & Business Media, 2013.
[24] V. C. Valchev and A. Van den Bossche, *Inductors and Transformers for Power Electronics*. Taylor & Francis, 2005.
[25] M. Leibl, G. Ortiz, and J. W. Kolar, "Design and Experimental Analysis of a Medium Frequency Transformer for Solid-State Transformer Applications," *IEEE Trans. Emerg. Sel. Topics Power Electron.*, vol. 5, no. 1, pp. 110–123, 2017.
[26] T. M. Undeland, J. Lode, R. Nilssen, W. P. Robbins, and N. Mohan, "A Single-Pass Design Method for High-Frequency Inductors," *IEEE Ind. Appl. Mag.*, vol. 2, no. 5, pp. 44–51, 1996.
[27] J. Mühlethaler and J. W. Kolar, "Optimal Design of Inductive Components Based on Accurate Loss and Thermal Models," in *Tutorial at the IEEE Applied Power Electronics Conf. and Expo. (APEC)*, Feb. 2012.
[28] M. Mogorovic and D. Dujic, "100 kW, 10 kHz Medium Frequency Transformer Design Optimization and Experimental Verification," *IEEE Trans. Power Electron.*, vol. 34, no. 2, pp. 1696–1708, 2019.
[29] T. Guillod, "Modeling and Design of Medium-Frequency Transformers for Future Medium-Voltage Power Electronics Interfaces," Ph.D. dissertation, ETH Zürich, 2018.
[30] J. W. Kolar, D. Bortis, and D. Neumayr, "The Ideal Switch is Not Enough," in *Proc. of the IEEE Int. Symp. on Power Semiconductor Devices and ICs (ISPSD)*, Jun. 2016.
[31] D. V. Malyna, J. L. Duarte, M. A. M. Hendrix, and F. B. M. van Horck, "Optimization of Combined Thermal and Electrical Behavior of Power Converters Using Multi-Objective Genetic Algorithms," in *Proc. of the European Conf. Power Electronics and Applications (EPE-ECCE Europe)*, Sep. 2007.
[32] R. J. Pratap, S. Sarkar, S. Pinel, J. Laskar, and G. S. May, "Modeling and Optimization of Multilayer LTCC Inductors for RF/Wireless Applications Using Neural Network and Genetic Algorithms," in *Proc. of. the IEEE Electronic Components and Technology Conf. (ECTC)*, Aug. 2004.
[33] T. Sato, K. Watanabe, H. Igarashi, T. Matsuo *et al.*, "3-D Optimization of Ferrite Inductor Considering Hysteresis Loss," *IEEE Trans. Magn.*, vol. 49, no. 5, pp. 2129–2132, 2013.
[34] P. Burrascano, G. Di Capua, S. Laureti, and M. Ricci, "Neural Models of Ferrite Inductors Non-Linear Behavior," in *Proc IEEE Int. Symposium on Circuits and Systems (ISCAS)*, May 2019.
[35] N. D. Doulamis, A. D. Doulamis, P. S. Georgilakis, S. D. Kollias, and N. D. Hatziargyriou, "A Synergetic Neural Network-Genetic Scheme for Optimal Transformer Construction," *Integrated Computer-Aided Engineering*, vol. 9, no. 1, pp. 37–56, 2002.
[36] J. Li, W. Water, B. Zhu, and J. Lu, "Integrated High-Frequency Coaxial Transformer Design Platform Using Artificial Neural Network Optimization and FEM Simulation," *IEEE Trans. Magn.*, vol. 51, no. 3, pp. 1–4, 2015.
[37] X. Yang, Y. Wang, F. Liu, Q. Yang, and W. Yan, "The Use of Neural Networks Combined with FEM to Optimize the Coil Geometry and Structure of Transverse Flux Induction Equipments," *IEEE Trans. Appl. Supercond.*, vol. 14, no. 2, pp. 1854–1857, 2004.
[38] S. Carcangiu, A. Fanni, and A. Montisci, "Multi Objective Optimization Algorithm Based on Neural Networks Inversion," in *Proc. of the Int. Work-Conference on Artificial Neural Networks (IWANN)*, Jun. 2009.
[39] K. Li, T. Zhang, and R. Wang, "Deep Reinforcement Learning for Multiobjective Optimization," *IEEE Trans. Cybern.*, 2020.
[40] D. Umbrello, G. Ambrogio, L. Filice, and R. Shivpuri, "A Hybrid Finite Element Method–Artificial Neural Network Approach for Predicting Residual Stresses and the Optimal Cutting Conditions During Hard Turning of AISI 52100 Bearing Steel," *Elsevier Materials and Design*, vol. 29, no. 4, pp. 873–883, 2008.
[41] O. Arndt, T. Barth, B. Freisleben, and M. Grauer, "Approximating a Finite Element Model by Neural Network Prediction for Facility Optimization in Froundwater Engineering," *Elsevier Advances in Complex Systems Modeling*, vol. 166, no. 3, pp. 769–781, 2005.
[42] TDK, "Ferrites and Accessories," May 2017, [Application Notes].
[43] T. Appel and H. Rossmanith, "The Benefit of Formed or Compacted Litz-Wire Coils," in *Proc. of the Int. Exhib. and Conf. for Power Electronics, Intelligent Motion, Renewable Energy and Energy Management (PCIM Europe)*, May 2016.
[44] A. Khan, V. Ghorbanian, and D. Lowther, "Deep Learning for Magnetic Field Estimation," *IEEE Trans. Magn.*, vol. 55, no. 6, pp. 1–4, 2019.
[45] T. Guillod, J. Huber, F. Krismer, and J. W. Kolar, "Litz Wire Losses: Effects of Twisting Imperfections," in *Proc. of the Workshop on Control and Modeling for Power Electronics (COMPEL)*, Jul. 2017.
[46] K. Venkatachalam, C. R. Sullivan, T. Abdallah, and H. Tacca, "Accurate Prediction of Ferrite Core Loss with Nonsinusoidal Waveforms Using only Steinmetz Parameters," in *Proc. of the IEEE Workshop on Computers in Power Electronics (CIPE)*, Jun. 2002.
[47] COMSOL, "COMSOL Multiphysics 5.4," Oct. 2018, [Reference Manual].
[48] W. T. McLyman, *Transformer and Inductor Design Handbook*. CRC Press, 2004.
[49] MathWorks, "MATLAB R2018b Deep Learning Toolbox," Dec. 2018, [Reference Manual].
[50] F. Chollet and others, *Keras: The Python Deep Learning API*, https://keras.io, 2015 (initial version).
[51] Google Brain Team and others, *Tensorflow: A System for Large-Scale Machine Learning*, https://www.tensorflow.org, 2015 (initial version).
[52] Agilent Technologies, "Agilent 4294A Precision Impedance Analyzer, Operation Manual," Feb. 2003, [Reference Manual].

Thomas Guillod (M'18) received the M.Sc. degree in electrical engineering and information technology in 2013 from ETH Zurich, Switzerland with a focus on power electronics, numerical analysis, and field theory. In 2013, he joined the Power Electronic Systems Laboratory at ETH Zurich as a Ph.D. student and, in 2018, as a postdoctoral researcher. His current research interests include MV converters, high-frequency magnetic components, design and optimization methods, and numerical methods.

Panteleimon Papamanolis (S'17) studied electrical engineering at the National Technical University of Athens (NTUA), with majors in energy conversion and electric power systems. In 2014 he continued his MSc studies at ETH Zürich in Robotics, Systems and Control. Since November 2016 he is with the Power Electronic Systems Laboratory at ETH Zurich as a Ph.D. student, focusing on the modeling, optimization and measurement of magnetic components and on 3-phase AC/DC rectifier converters for EV charging applications.

Johann W. Kolar (F'10) received his M.Sc. degree in Industrial Electronics and Control Engineering and his Ph.D. degree in Electrical Engineering (summa cum laude/promotio sub auspiciis praesidentis rei publicae) from the Vienna University of Technology, Austria, in 1997 and 1999, respectively. Since 1984, he has been working as independent researcher and international consultant in close collaboration with the Vienna University of Technology, in the fields of power electronics, industrial electronics and high performance drive systems. He is currently a Full Professor and the Head of the Power Electronic Systems Laboratory at the Swiss Federal Institute of Technology (ETH) Zurich. He has proposed numerous novel PWM converter topologies, modulation and control concepts, multi-objective power electronics design procedures, etc. and has supervised 70+ Ph.D. students. He has published 900+ scientific papers in international journals and conference proceedings, 4 book chapters, and has filed 190+ patents. He has presented 30+ educational seminars at leading international conferences, has served as IEEE PELS Distinguished Lecturer from 2012 through 2016, and has received 36 IEEE Transactions and Conference Prize Paper Awards, the 2014 IEEE Power Electronics Society R. David Middlebrook Achievement Award, the 2016 IEEE William E. Newell Power Electronics Award, the 2016 IEEE PEMC Council Award, and two ETH Zurich Golden Owl Awards for excellence in teaching. He has initiated and/or is the founder of 4 ETH Spin-off companies. The focus of his current research is on ultra-compact and ultra-efficient SiC and GaN converter systems, ANN-based power electronics components and systems design, Solid-State Transformers, Power Supplies on Chip, as well as ultra-high speed and ultra-light weight drives, bearingless motors, and energy harvesting.