Received 19 July 2023, accepted 19 August 2023, date of publication 22 August 2023, date of current version 5 September 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3307499

RESEARCH ARTICLE

Deep Learning-Based Meta-Modeling for Multi-Objective Technology Optimization of Electrical Machines

VIVEK PAREKH[1,2], **DOMINIK FLORE**[2], **AND SEBASTIAN SCHÖPS**[1]
[1]Computational Electromagnetics Group, Technische Universität Darmstadt, 64289 Darmstadt, Germany
[2]Powertrain Solutions, Mechanical Engineering and Reliability, Robert Bosch GmbH, 70442 Stuttgart, Germany

Corresponding author: Vivek Parekh (Parekhvivek49@gmail.com)

This work was supported by Robert Bosch GmbH and the SFB TRR 361 CREATOR under Grant 492661287 funded by the German Research Foundation (DFG). Furthermore, the third author thanks Aylar Partovizadeh and Dimitrios Loukrezis for the fruitful discussions regarding machine learning for electric machines.

ABSTRACT Optimization of rotating electrical machines is both time- and computationally expensive. Because of the different parametrization, design optimization is commonly executed separately for each machine technology. In this paper, we present the application of a variational auto-encoder (VAE) to optimize two different machine technologies simultaneously, namely an asynchronous machine and a permanent magnet synchronous machine. After training, we employ a deep neural network and a decoder as meta-models to predict global key performance indicators (KPIs) and generate associated new designs, respectively, through unified latent space in the optimization loop. Numerical results demonstrate concurrent parametric multi-objective technology optimization in the high-dimensional design space. The VAE-based approach is quantitatively compared to a classical deep learning-based direct approach for KPIs prediction.

INDEX TERMS Asynchronous machine, deep neural network, key performance indicators, multi-objective optimization, permanent magnet synchronous machine, variational auto-encoder.

I. INTRODUCTION

Electrical machines play a pivotal role in the modern era, powering everything from home appliances to electric vehicles and industrial equipment. To reduce manufacturing costs, electrical machines are numerically optimized via virtual prototyping, which involves finite element (FE) simulations or analytical calculations before the actual machine is constructed. Design evaluations using these classical techniques are both costly and time-consuming. Therefore, it is essential to find faster optimization methods to ensure a more sustainable and energy-efficient design workflow.

In recent years, there has been a significant increase in the use of machine learning-based meta-modeling for the accelerated numerical optimization of electrical machines for various purposes. For example, the study in [1] shows how trained data-driven deep learning (DL) models estimate magnetic field distribution for different low-frequency

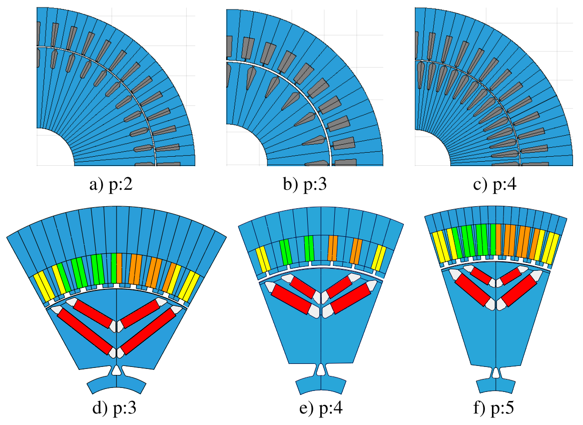


FIGURE 1. Representative geometries of ASM (a-c) and PMSM (d-f) for different pole pairs (p).

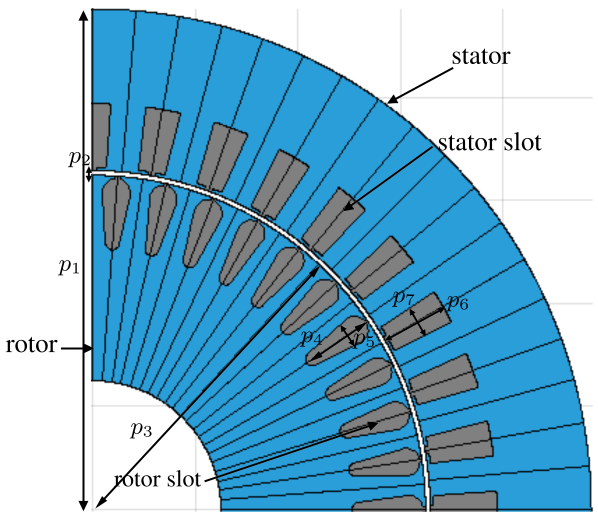
electromagnetic devices such as a transformer, a coil in the air, and an interior permanent magnet machine. In another work [2], a convolutional neural network (CNN), a type of deep neural network (DNN), is used as a meta-model for diagnosing stator winding faults of a permanent magnet

The associate editor coordinating the review of this manuscript and approving it for publication was Bijoy Chand Chatterjee.

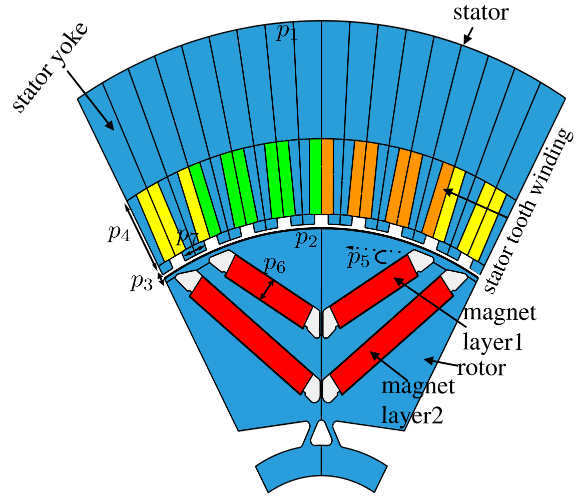
synchronous machine (PMSM). The paper [3] presents how various machine learning methods assist in fault detection for induction machines. The concurrent application of variational autoencoder (VAE) and DNN for technology optimization of electromagnetic devices is investigated in [4]. An unsupervised learning-based anomaly detection model using a VAE for fault diagnosis in electric drive systems is presented in [5]. Several successful works of machine learning-based meta-modeling for the optimization of electromagnetic devices have been discussed in [6]. Many articles [7], [8], [9], [10], [11], [12], [13] demonstrate the successful application of different machine-learning approaches at different stages of the design and optimization of electrical machines. The reduction of computational time required to generate sufficient data for training large-scale machine learning-based meta-models needs to be addressed. In order to tackle this issue, a method is proposed in [14] for generating a large amount of data from a small number of FE simulation results using a deep generative model and a CNN. In a recent study [15], an approach for topology optimization of PM motors using the VAE and the neural network was demonstrated to generate various shapes and predict their corresponding motor characteristics within the optimization loop. However, the VAE often fails while reconstructing images in this approach. In [16], it has been shown how cross-domain key performance indicators (KPIs) can be estimated with high accuracy for different input representations of PMSM, i.e., image-based and parametric input using various DNNs. The parameter-based representation was observed to be more suitable concerning prediction accuracy with less computational effort than the image-based model. However, it can not deal with multiple topologies of PMSM concurrently. In [17], it is demonstrated how we can circumvent this problem for differently parameterized topologies of PMSM using VAE. The encoder maps the complex, high-dimensional combined input design space into a lower-dimensional unified latent distribution. In the latent space, multi-topology objective optimization was performed by predicting KPIs using the DNN and generating associated new designs with the decoder.

So far, to the best of our knowledge, machine learning methods have been applied for a scenario dealing with the numerical optimization of a single machine type at a time, for example, PMSM, asynchronous machine (ASM), or DC machine. In this paper, we aim to perform concurrent multi-technology objective optimization (MTOO) for two different machine types, namely PMSM and ASM. Both machine types are distinctly parameterized and operate on different working principles. This difference can affect the performance and efficiency of the machines in various applications. For example, PMSMs typically have higher efficiency and power density than ASMs but may also be more expensive to manufacture due to the cost of the permanent magnets.

This paper proposes two significant differences compared to our previous works. We consider, in addition to geometric parameters, more challenging varying topological parameters such as pole pairs, number of slots per pole per phase,



(a)



(b)

FIGURE 2. Different machine technologies (a) ASM (b) PMSM.

winding connection (star or delta), etc. Secondly, in order to handle these challenging parameters and the additional zeros in the combined design space, we propose a new optimization procedure that enhances synchronization between the decoder and the KPI predictor in the latent space. This proposed procedure improves the prediction accuracy of KPIs and associated design parameters. Additionally, we provide numerical analysis of the direct DL approach using a DNN for KPIs prediction [16].

The paper is organized as follows: in the next section, we explain the dataset details and training workflow. Section III discusses the network architecture and training details. Quantitative results are demonstrated in Section IV, and finally, the work is concluded in Section V.

II. DATASET, TRAINING PROCEDURE AND MOO

This section is divided into three subsections: the details of the datasets, the explanation of the VAE-based training

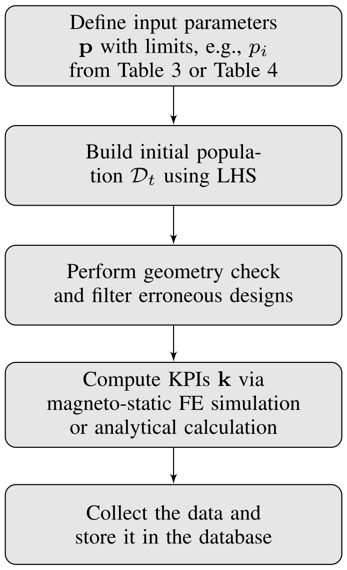

Define input parameters
$\mathbf{p}$ with limits, e.g., p_i
from Table 3 or Table 4
Build initial population $\mathcal{D}_t$ using LHS
Perform geometry check and filter erroneous designs
Compute KPIs $\mathbf{k}$ via magneto-static FE simulation or analytical calculation
Collect the data and store it in the database

FIGURE 3. General workflow for electrical machine data-generation.

TABLE 1. KPIs information.

	KPIs	Value
k_1	Material cost	Euro
k_2	Maximum power	kW
k_3	Maximum torque	Nm

TABLE 2. System parameters.

	System parameter	Value	Unit
c_1	Inverter input DC voltage	650	V
c_2	Inverter input DC current	400	A
c_3	Rotational speed	[1, 16000] (step size: 1000)	rpm

workflow, and the formulation of the multi-objective optimization (MOO) problem.

A. DATASET

For the study, we build two datasets: one for an ASM and one for a PMSM. The usual industrial workflow of data generation is explained in Figure 3 for any electrical machine class. There may be some modifications in the workflow at the phase when the design evaluation is conducted. In this study, we evaluate a PMSM design with a time-intensive magneto-static FE simulation (see [18]), while the ASM design is evaluated with analytical calculations (see [19], [20]). The initial steps are identical in both instances of data generation. For example, specifying design parameters with constraints, creating a population with a Latin hypercube sampling technique [21] to cover the entire design space, and doing geometry checks for filtering erroneous designs using computer-aided design software (e.g., [22]).

TABLE 3. ASM design parameters.

	Input design parameter	Min	Max	Unit
p_1	Stator outer diameter	159	232	mm
p_2	Air gap	0.65	1.7	mm
p_3	Rotor outer diameter	85	190	mm
p_4	Rotor slot height	10	21	mm
p_5	Rotor slot width	0.6	1.5	mm
	Discrete parameters	**Value**		**Unit**
p_{15}	Slots per pole per phase	[2-4]		-
p_{16}	Pole pairs (p)	[2-4]		-
p_{17}	Stator winding connection	star/delta		-
p_{18}	Winding scheme	short pitch/full pitch		-

TABLE 4. PMSM design parameters.

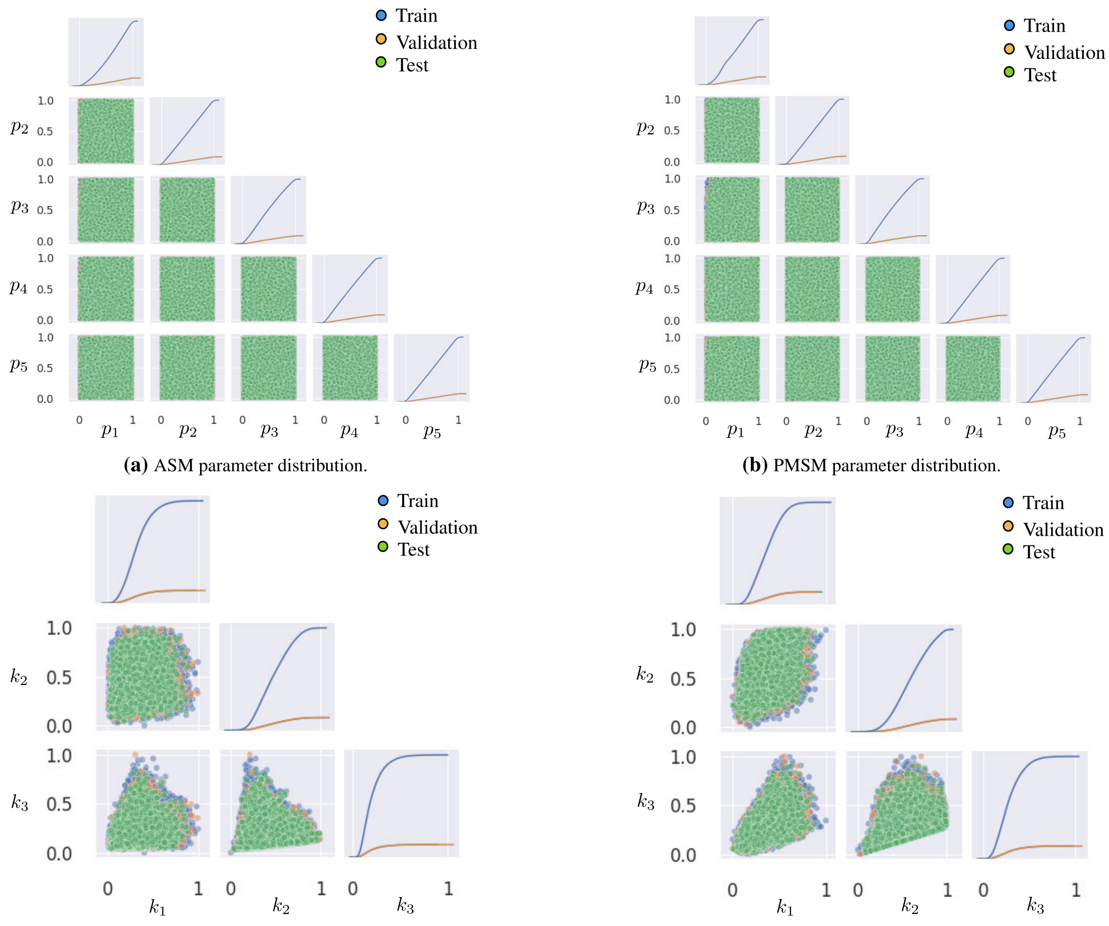
	Input design parameter	Min	Max	Unit
p_1	Stator outer diameter	159	232	mm
p_2	Rotor outer diameter	100	197	mm
p_3	Air gap	0.8	2.2	mm
p_4	Stator tooth height	10	20	mm
p_5	Angle of magnet layer1	17	32	mm
	Discrete parameters	**Value**		**Unit**
p_{30}	Slots per pole per phase	[2-4]		-
p_{31}	Pole pairs (p)	[3-5]		-
p_{32}	Stator winding connection	star/delta		-
p_{33}	Winding scheme	short pitch/full pitch		-

1) DATASET: ASM

We selected $T_{\text{ASM}} := 50387$ valid ASM designs from the initial population. There is no fixed number of samples for data generation, but a large number of samples is usually preferred for the data-driven DL approach. There are total $d_1 := 18$ varying design parameters chosen for this data generation. From all the varying parameters, a few essential design parameters are detailed in Table3. Some parameters from Table 3 are shown in Figure 2a. Representative samples with varying pole pairs can be seen in Figure 1. Figure 4a and Figure 4c visualize distribution of the listed parameters and KPIs. The dataset contains topology changing parameters such as the number of slots per pole per phase p_{15}, and varying pole pairs p_{16}. Electrical parameters are included as design parameters, i.e., stator winding connection (star/delta) and winding scheme (short pitch/full pitch winding). The evaluation time for one design is about $5 - 7$ minutes on a single-core CPU.

2) DATASET:PMSM

We selected $T_{\text{PMSM}} := 51532$ valid designs from the initial population. The number of samples for this dataset is close to the number of the previous ASM dataset ($\leq 3\%$); otherwise, the network can be biased towards one machine technology during training. The number of varying design parameters for PMSM is $d_2 := 33$. The PMSM dataset also incorporates variability for topological and electrical parameters.



(a) ASM parameter distribution.

(b) PMSM parameter distribution.

(c) ASM KPIs distribution.

(d) PMSM KPIs distribution.

FIGURE 4. Visualization parameter and KPIs distribution.

A few crucial parameters listed in Table 4 can be observed in Figure 2b, and representative samples with the varying pole pairs are illustrated in Figure 1. The distribution of the listed parameters and KPIs is illustrated in Figure 4b and Figure 4d, respectively.

Our goal is to perform concurrent MOO for both technologies. Hence, both datasets were generated under the assumption of identical KPIs and constant system parameters. The KPIs and constant system parameters information are given in Table 1 and Table 2, respectively. Furthermore, during data generation, it is assumed that the cost of standard materials (such as aluminum and copper) is the same for the ASM and the PMSM. The significant difference in the cost comes when considering magnets for the PMSM.

B. TRAINING PROCEDURE

We employ a VAE-based workflow; it maps the input data to a low-dimensional latent space through a probabilistic encoder and then reconstructs it through a decoder [23]. As explained in [17], we first create a combined design space by concatenating the design vectors of all the given machines. It creates d-dimensional design vector with $d = 1 + d_1 + \ldots + d_M$, where $d_1, \ldots, d_M$ represent input dimension of each different machine technology $t = 1, \ldots, M$.

After concatenation, we define any ith sample in the combined dataset as a d-dimensional vector if it is from the technology t as

$$\mathbf{p}^{(i)} = [t, \mathbf{0}, \ldots, \mathbf{0}, \mathbf{p}_t^{(i)}, \mathbf{0}, \ldots, \mathbf{0}]$$

and KPIs vector for the design as $\mathbf{k}^{(i)} = \mathbf{k}_t(p_t^{(i)})$.

The total combined input dataset with all the machine types can be mathematically described by

$$\mathcal{D} := \left\{ \mathbf{p}^{(i)} \;\middle|\; \text{for } i = 1, \ldots, T_{\text{tot}} \right\}. \quad (1)$$

The assumption is that l-dimensional unseen variables $\mathbf{z}$ from a latent distribution can describe all the d-dimensional input samples from the dataset $\mathcal{D}$. It is also assumed that the input parameters of each machine type are independent of each other. Therefore, as described in [17], to reconstruct parameters with high accuracy, the latent dimension l should be at least higher than the maximum input dimension of all the machine types, i.e., $l \geq \max_M(d_M)$ and also $l \leq d$.

The encoder network computes the conditional distribution $\mathbf{P}(\mathbf{z}|\mathbf{p})$ with the presumption that $\mathbf{z}$ follows the standard normal distribution. It is written as

$$(\boldsymbol{\upsilon}, \boldsymbol{\sigma}) := \mathbf{E}_{\phi}(\mathbf{p}). \tag{2}$$

where outputs mean ($\boldsymbol{\upsilon}$) and diagonal components of the covariance matrix as a vector (σ) represent latent distribution parameters with the dimension l. ϕ are trainable encoder network ($\mathbf{E}_{\phi}$) parameters.

To compute and backpropagate gradients during training, as described in [24], the latent vector $\mathbf{z}$ is sampled using a reparametrization trick by

$$\mathbf{z} = \boldsymbol{\upsilon} + \boldsymbol{\sigma} \odot \boldsymbol{\varepsilon}, \tag{3}$$

where $\boldsymbol{\varepsilon} \sim \mathcal{N}(0, \mathbf{I})$ is a noise vector, and $\odot$ is the component-wise dot product. The decoder network $\mathbf{D}_{\theta}$ takes latent vector $\mathbf{z}$ as input. It approximates the conditional distribution $\mathbf{P}(\mathbf{p}|\mathbf{z})$ i.e.,

$$\hat{\mathbf{p}} := \mathbf{D}_{\theta}(\mathbf{z}). \tag{4}$$

where θ are the trainable parameters of the decoder. Simultaneously, we train DNN with latent input ($\mathbf{z}$) to predict the KPIs. It is written as

$$\hat{\mathbf{k}} := \mathbf{K}_{\beta}(\mathbf{z}), \tag{5}$$

where β are trainable parameters of the DNN and $\hat{\mathbf{k}}$ is vector of KPIs prediction. The primary goal of training the VAE is to minimize the errors in prediction, parameter reconstruction, and encoding process by optimizing the network parameters θ, ϕ, β simultaneously. One important step before training is defining the loss function. The choice of the training loss function depends on the specific task at hand. The MSE is commonly used as a loss function for regression tasks [25, Chapter 5], although other options, such as mean absolute error (MAE), exist. Through experiments, we determined that the MSE provides better prediction accuracy for our datasets. The MSE is a practical selection for parameter reconstruction since the input data is scalar. The total training loss comprises three components: the first two terms are squared error for parameter reconstruction and KPIs prediction, and the third term is Kullback-Leibler (KL) divergence for regularization in the latent space. Total VAE training loss is specified in terms of network parameters, input vector $\mathbf{p}^{(i)}$, and actual KPIs $\mathbf{k}^{(i)}$ by

$$\mathbf{L}(\theta, \phi, \beta; (\mathbf{p}^{(i)}, \mathbf{k}^{(i)})) = \left\| \mathbf{p}^{(i)} - \hat{\mathbf{p}}^{(i)} \right\|^2 + \left\| \mathbf{k}^{(i)} - \hat{\mathbf{k}}^{(i)} \right\|^2 + \mathbf{D}_{\mathrm{KL}}\Big(\mathbf{P}(\mathbf{z}^{(i)}|\mathbf{p}^{(i)}, \theta) \;||\; \mathbf{z} \sim \mathcal{N}(0, \mathbf{I})\Big).$$

The KL divergence $\mathbf{D}_{\mathrm{KL}}$ minimizes the difference between encoder distribution and prior distribution over latent variables. It works as a regularizer term in the loss function to provide continuity and completeness in the latent space. It means the samples nearby in the latent space remain similar when decoded while preserving meaningful representation [26]. Here, for independent training of the DNN in a supervised manner [16] for each machine technology, we input $\mathbf{p}_t$ as the input vector instead of latent vector $\mathbf{z}$. It is written as

$$\hat{\mathbf{k}}_t := \mathbf{K}_{\gamma}(\mathbf{p}_t). \tag{6}$$

Here, γ are network parameters, $\hat{\mathbf{k}}_t$ is the predicted KPIs vector, and $\mathbf{p}_t$ is an input vector of the individual machine technology with dimension d_t. The loss function during the network training is kept the same (MSE) as of the VAE for KPIs prediction. The only obvious change in the network structure is the input layer, compared to the used DNN during the VAE training. All the networks are trained using a standard back-propagation algorithm [27]. Figure 5 describes training workflow.

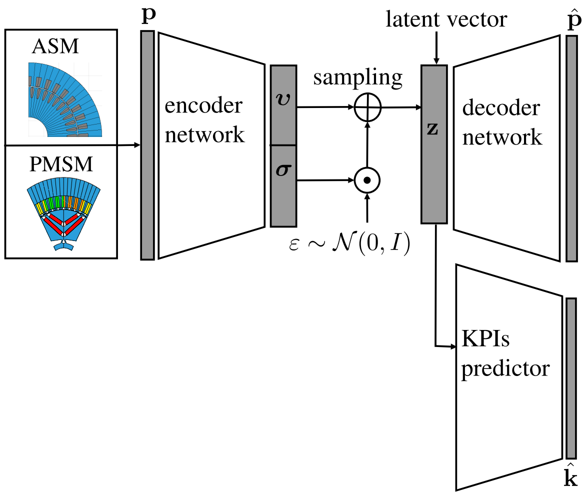


FIGURE 5. VAE-based training workflow [17].

C. MOO

Every electrical machine design optimization comprises many design variables, constraints, and competing objectives (see Table 1). It leads to the generalized MOO problem formulation

$$\min_{\mathbf{p}} k_a(\mathbf{p}), \quad a = 1, \ldots, n_{\mathrm{obj}} \tag{7}$$

$$\text{s.t. } c_j(\mathbf{p}) \leq 0, \quad j = 1, \ldots, n_{\mathrm{cons}} \tag{8}$$

$$p_i^{\mathrm{L}} \leq p_i \leq p_i^{\mathrm{U}}, \quad i = 1, \ldots, n_{\mathrm{param}} \tag{9}$$

where $\mathbf{p}$ represent input vector and p_i^{L} and p_i^{U} are parameter bounds, $k_a(\mathbf{p})$ denote KPIs, $c_j(\mathbf{p})$ are constraints for design evaluation. Any commonly practiced multi-objective optimizer [28] can solve (7-9).

In this study, we propose two MOO workflows: one for MTOO via continuous latent space using the VAE (see Figure 6) and another (Figure 7) for the DNN-based classical workflow for individual machine technology.

In the VAE-based MTOO, randomly generated latent vector $\mathbf{z}$ is input to the optimization process. As shown in Figure 6, first, we give the latent vector as input $\mathbf{z}$ to the decoder($\mathbf{D}_{\theta}$), and the decoder predicts design parameters for the related technology. From the data pre-processing,

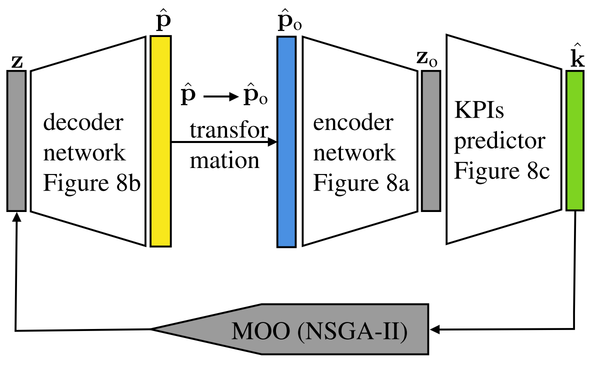


FIGURE 6. **Proposed VAE-based optimization workflow.**

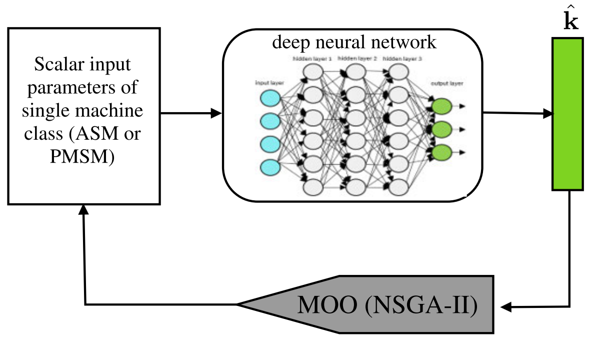

Scalar input parameters of single machine class (ASM or PMSM)
deep neural network
k̂
MOO (NSGA-II)

FIGURE 7. **Individual DNN based optimization workflow.**

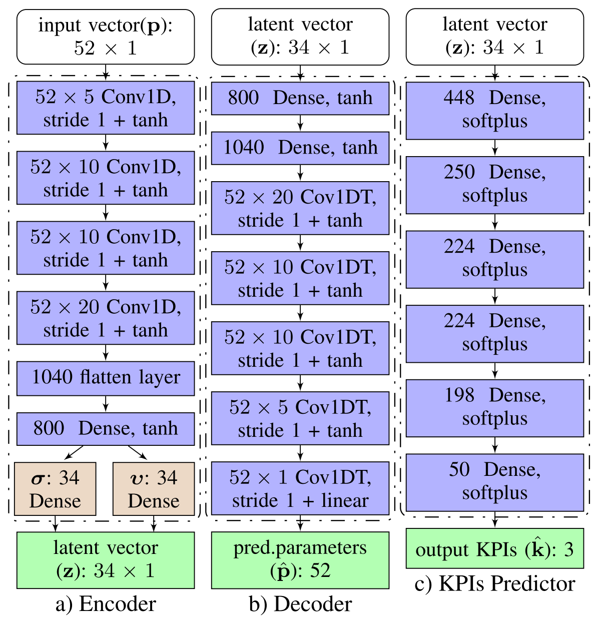

input vector(p): 52 × 1
52 × 5 Conv1D, stride 1 + tanh
52 × 10 Conv1D, stride 1 + tanh
52 × 10 Conv1D, stride 1 + tanh
52 × 20 Conv1D, stride 1 + tanh
1040 flatten layer
800 Dense, tanh
σ: 34 Dense
υ: 34 Dense
latent vector (z): 34 × 1
a) Encoder
latent vector (z): 34 × 1
800 Dense, tanh
1040 Dense, tanh
52 × 20 Cov1DT, stride 1 + tanh
52 × 10 Cov1DT, stride 1 + tanh
52 × 10 Cov1DT, stride 1 + tanh
52 × 5 Cov1DT, stride 1 + tanh
52 × 1 Cov1DT, stride 1 + linear
pred.parameters (p̂): 52
b) Decoder
latent vector (z): 34 × 1
448 Dense, softplus
250 Dense, softplus
224 Dense, softplus
224 Dense, softplus
198 Dense, softplus
50 Dense, softplus
output KPIs (k̂): 3
c) KPIs Predictor

FIGURE 8. **Network structure.**

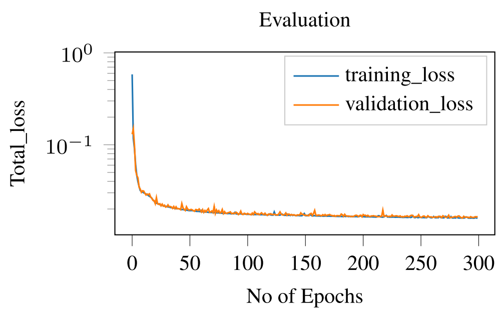


FIGURE 9. **Training and validation loss curves.**

we have prior knowledge about the positions of the actual design parameters in the concatenated form of the input vector. Hence, we keep those values unchanged and fill the remaining values with zeros except for the first entry. The first entry indicates technology type, so we replace the predicted continuous value with a known integer value. We also replace predicted discrete parameters, such as pole pairs, slots per pole per phase, winding scheme, etc., with integer values based on prior knowledge. This makes the input vector ($\hat{\mathbf{p}}$) in the form ($\hat{\mathbf{p}}_\text{o}$) that the encoder ($\mathbf{E}_\phi$) expects. Then the encoder network creates a new latent vector $\mathbf{z}_\text{o}$, which is input to the KPIs predictor. The standard DNN-based MOO workflow (Figure 7) operates on one machine technology at a time, so we input a design vector with parameterization $\mathbf{p}_t \in \mathbb{P}_t \subset \mathbb{R}^{d_t}$.

III. NETWORK ARCHITECTURE AND TRAINING SPECIFICATIONS

A. NETWORK ARCHITECTURE

As illustrated in Figure 8, there are three networks: the encoder ($\mathbf{E}_\phi$), the decoder ($\mathbf{D}_\theta$) and the DNN ($\mathbf{K}_\beta$). The network structure and training hyperparameters are obtained randomly through trial and error by evaluating approximately twenty configurations starting with the base network configuration from [17]. The details are as follows,

- *Encoder network*: The encoder($\mathbf{E}_\phi$) or inference network consists of four 1D convolutional layers. These layers are significant for learning essential features from the combined design vector and determining whether the design is ASM or PMSM from the technology indicator (the first input parameter of the design vector). A flattened layer and a dense layer follow these convolutional layers. Three output layers follow the dense layer. Two output layers are the distribution parameters mean ($\boldsymbol{\upsilon}$) and variance ($\boldsymbol{\sigma}$) that sample the final output latent vector ($\mathbf{z}$).
- *Decoder network*: The decoder network ($\mathbf{D}_\theta$), also called the generative network, consists of two dense layers, four 1D transposed convolutional layers, and one output layer corresponding to the dimension of the integrated design space. It follows the reverse structure of the encoder except for the output layer. The output layer has a linear activation function.
- *DNN*: The DNN is also interchangeably known as KPIs predictor in this study. The DNN has one input, five dense layers, and one output layer. For individual DNNs, we use an identical network structure (except the input layer) and hyperparameters, where we only train the DNN for one machine type at a time.

B. TRAINING SPECIFICATIONS

The total number of samples in a combined dataset is $T_\text{tot} := T_\text{ASM} + T_\text{PMSM} := 101919$. We split a total number of

TABLE 5. Training hyperparameters detail.

Parameters	Value
Learning rate	10^{-4}-10^{-5}
Total number of epochs	300
Validation patience	20
Optimizer	Adam[29]
Latent space dimension	34 (as $d_{\text{PMSM}} := 33$)
Loss functions	KL-divergence and MSE
Batch size	50

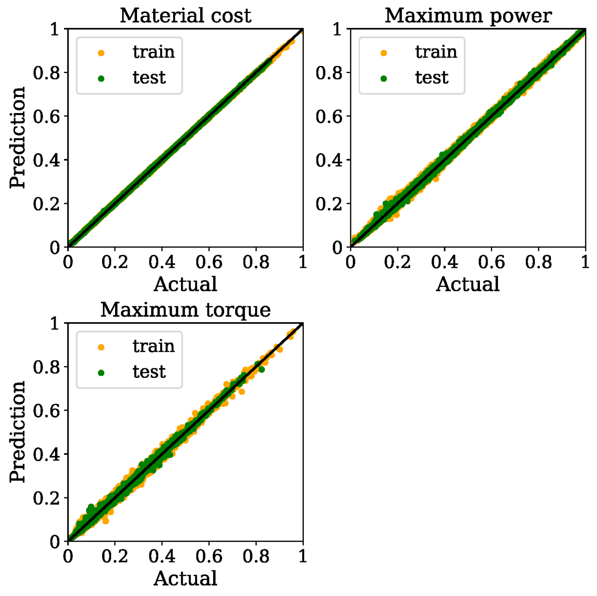


FIGURE 10. Predictions of the KPIs over test samples.

TABLE 6. KPIs evaluation over test samples.

	Prediction accuracy			
	MAE	RMSE	PCC	MRE(%)
k_1	0.43	0.53	1	0.71
k_2	1.90	2.54	1	1.31
k_3	3.96	6.52	0.99	1.76

samples (T_{tot}) into three sets: training ($\sim$ 80% of T_{tot}), validation ($\sim$ 10%), and testing ($\sim$ 10%). Figure 9 displays training and validation curves. Table 5 gives the details of the training hyperparameters. The network training was carried out on NVIDIA Quadro M2000M GPU. It took $\sim$ 1.5 hours to complete the VAE training for the multi-technology scenario, whereas separate DNNs take around $\sim$ 15 minutes with the single machine technology. The magneto-static FE simulation takes $2-4$ h/sample for PMSM, and analytical calculation takes around $5-7$ minutes/sample for ASM on a single-core CPU.

IV. NUMERICAL RESULTS

In this study, our primary focus is on the concurrent multi-technology scenario; therefore, we will explain the evaluation of the trained VAE in more detail. After training, we test trained models on the test dataset. Table 6 gives evaluation details of test samples for all three global KPIs. We use unitless mean relative error (MRE), the root mean squared error (RMSE), MAE, and Pearson correlation coefficient (PCC) to measure the correlation between input and target variables. Figure 10 illustrates prediction plots of all these three KPIs over test samples. We can see that the maximum torque KPI

TABLE 7. ASM parameters evaluation over test samples.

Parameters	Reconstruction accuracy			
	MAE	RMSE	PCC	MRE(%)
Stator outer diameter	0.39	0.483	0.99	0.19
Air gap	0.002	0.005	1	0.37
Rotor outer diameter	0.55	0.636	0.99	0.41
Rotor slot height	0.05	0.069	0.99	0.37
Rotor slot width	0.004	0.005	1	0.4
Stator slot height	0.05	0.004	1	0.32

TABLE 8. PMSM parameters evaluation over test samples.

Parameters	Reconstruction accuracy			
	MAE	RMSE	PCC	MRE(%)
Stator outer diameter	0.51	0.63	0.99	0.26
Rotor outer diameter	0.39	0.5	0.99	0.27
Air gap	0.006	0.007	1	0.41
Stator tooth height	0.054	0.072	0.99	0.37
Angle magnet layer 1	0.11	0.14	0.99	0.48
Height of magnet layer 1	0.012	0.015	1	0.27

TABLE 9. MOO settings information.

Settings	Value
Sampling approach	random initialization
Population per generation	1000
Stopping criteria	100 generations
Number of objectives	2
Crossover, mutation probability	0.9

TABLE 10. Design evaluation from VAE Pareto front.

KPIs	Design A (PMSM)			Design B (ASM)		
	FE simulation	Prediction	RE(%)	FE simulation	Prediction	RE(%)
k_1	153.53	153.49	0.026	45.47	46.27	1.75
k_2	402.98	406.70	0.92	241.7	237.82	1.60
k_3	294.61	286.63	2.70	195.46	187.52	4.06

TABLE 11. Design evaluation from DNNs Pareto front.

KPIs	Design C (PMSM)			Design D (ASM)		
	FE simulation	Prediction	RE(%)	FE simulation	Prediction	RE(%)
k_1	130.81	129.51	0.99	46.68	46.73	0.53
k_2	401.8	404	0.54	227.48	235	3.3
k_3	313.05	316.27	1.02	158	200.36	26.81

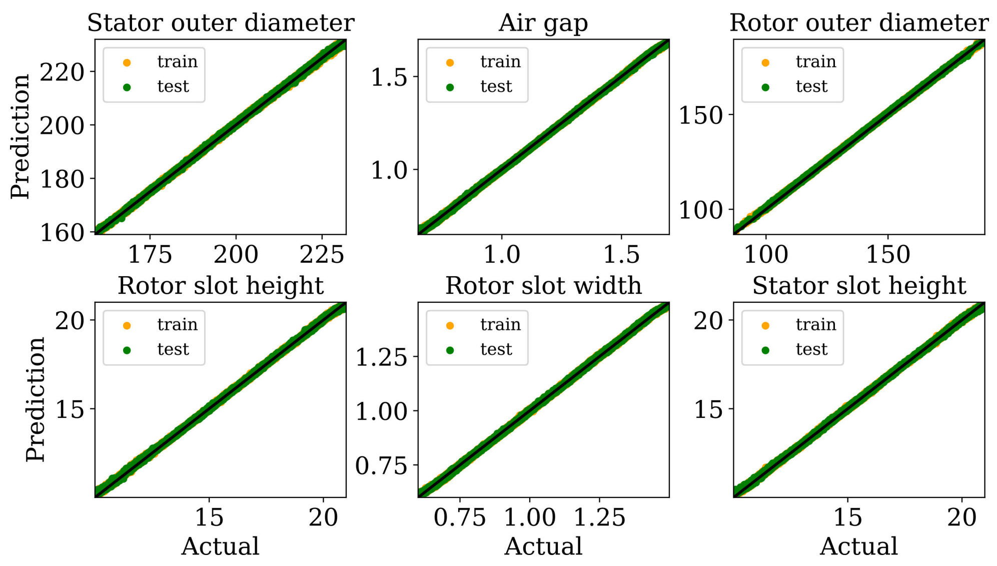


FIGURE 11. ASM parameters prediction plot over test samples.

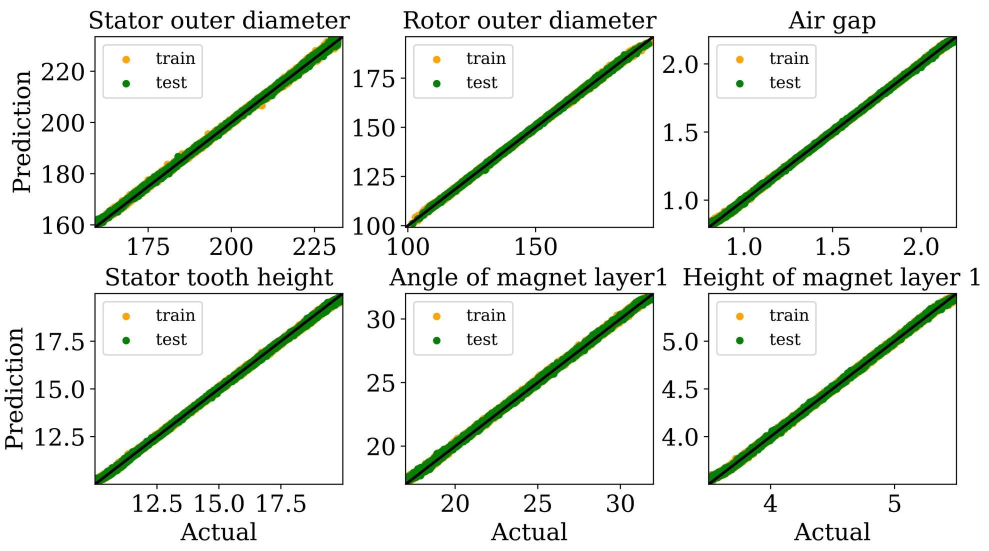


FIGURE 12. PMSM parameters prediction plot over test samples.

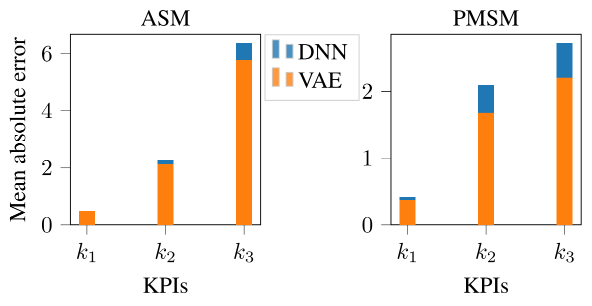


FIGURE 13. Comparison VAE vs DNN.

(k_3) has a higher MAE (3.96 Nm). We present a numerical analysis of parameter reconstruction of six parameters of each machine type in Table 7 and Table 8. It is observed that the parameter reconstruction is obtained with high precision. Figure 11 and Figure 12 display prediction plots. The other parameters, which are not illustrated here, also have higher reconstruction accuracy.

Figure 13 displays a basic numerical comparison of the KPIs prediction performance between the VAE and the DNN using the MAE on the same test samples. For the numerical analysis, the DNN is trained with a twin network configuration and hyperparameters as used for the DNN for the latent input. The DNNs are directly trained on the input parameters of each machine type using a supervised learning approach. The VAE has a slightly better prediction accuracy than the trained DNN for the single machine type. This is likely due

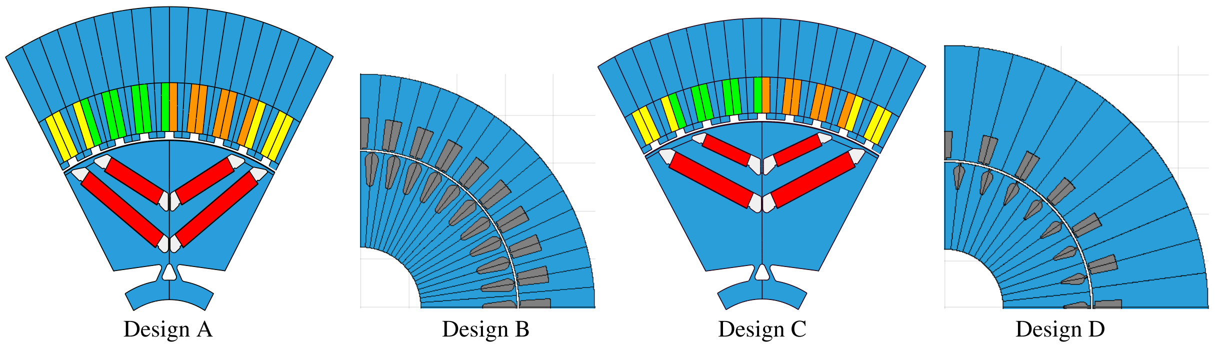


FIGURE 14. Pareto designs.

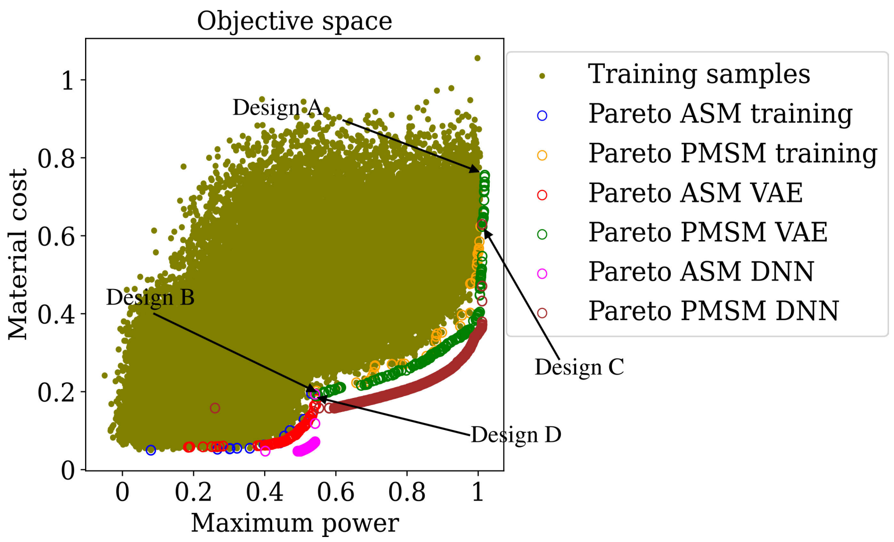


FIGURE 15. Pareto-fronts for Maximum power and Material cost, where model training samples are in olive, the Pareto-front of ASM training samples is in blue, and the Pareto-front of PMSM training samples is in orange. Pareto-fronts for the VAE-based approach are displayed in red (PMSM) and green (ASM), and Pareto-fronts for the DNN-based direct approach are shown in brown (PMSM) and magenta (ASM).

to more training samples in the combined dataset and more accurate functional mapping between latent input and output global KPIs.

The trained models (encoder, decoder, and DNN) are used for MTOO for $T = 2$ technologies. We propose an improved optimization workflow Figure 6. The proposed workflow improves synchronization between the decoder and the KPIs predictor in the optimization loop. It also handles many discrete input parameters effectively.

We perform MOO for two competing global KPIs: material cost and maximum power. We use the genetic algorithm NSGA-II, which can handle many design variables [28]. We set the MOO hyperparameters by experience; see Table 9 for values. The optimization process includes input parameter bounds as a constraint to reduce invalid design generation. The optimization requires roughly $\sim$ 2.5 hours. We also run individual MOO for each machine type with the separately trained DNN model. Identical hyperparameter settings as of the VAE-based optimization are used. Each machine optimization takes around $\sim$ 40 $-$ 50 minutes. Figure 15 depicts different Pareto-fronts for the training samples, the VAE approach, and the separate DNN approach. All Pareto-fronts from DNNs and VAE show power or cost-effective designs. It can be seen that Pareto-fronts based on VAE and DNNs predictions contain designs that are not present in the training data. We illustrate two designs from each Pareto front. Design A (PMSM) and Design B (ASM) from the VAE Pareto front. Similarly, Design C (PMSM) and Design D (ASM) are from the individually trained DNN models. We recalculated all these four designs with their conventional approach. Table 10 shows the evaluation for all three KPIs with percentage relative error (RE). We can see that Design B (ASM) has a high (4.06%) RE for the maximum torque KPI. Likewise, Table 11 evaluates Design C (PMSM) and D (ASM). For Design D, RE is very high, 26.81%, possibly due to a poor approximation of the meta-model for the maximum torque for that design. It is seen from Figure 15 that the DNN-based separate approach shows a more efficiently obtained

Pareto front. We checked twenty designs from each Pareto front of both approaches. For the direct DNN-based approach, most designs from the more efficiently obtained Pareto region were found geometrically invalid. We observe that, even if we apply input parameter bounds as constraints to lower invalid design generation, we get much higher invalid designs ($\sim 60\%$) compared to VAE-based concurrent optimization. If the design is valid, then the prediction has a high deviation after recalculating with the conventional approach. Design D from Figure 15 is the example where we observe that recalculation produces a higher deviation in the prediction of torque KPI (RE is 26.81%; see Table 11). This can be improved by adding geometry checks during optimization. However, that investigation is beyond the scope of this work.

V. CONCLUSION AND OUTLOOK

We present the application of the VAE-based approach for optimizing two different machine types (ASM and PMSM) simultaneously over a common set of KPIs, i.e., material cost and maximum power. The numerical results demonstrate high prediction accuracy for parameter reconstruction and KPIs in a complex design space. This enables the optimization of several electrical machine technologies with a single meta-model training. The quantitative analysis for the DNN-based direct approach for optimizing each machine type is also demonstrated. The MOO results show that the direct DNN-based approach has a more efficiently obtained Pareto region, but the VAE outputs more valid meaningful designs than independent optimization with DNN. However, the DNN-based optimization takes less computational effort than the VAE-based approach for fewer machines (in this study, two). We expect a linear increase in the computational time during optimization for multiple machine types when the DNN-based models are trained separately. On the contrary, only a little increase in the computational time is expected for the VAE-based approach. Future work may include the application of trained meta-models in other query scenarios, such as sensitivity analysis and uncertainty quantification. The more challenging situation, e.g., optimization for more than two machine technologies, can also be considered.

REFERENCES

[1] A. Khan, V. Ghorbanian, and D. Lowther, "Deep learning for magnetic field estimation," *IEEE Trans. Magn.*, vol. 55, no. 6, pp. 1–4, Jun. 2019, doi: 10.1109/TMAG.2019.2899304.

[2] P. Pietrzak, M. Wolkiewicz, and T. Orlowska-Kowalska, "PMSM stator winding fault detection and classification based on bispectrum analysis and convolutional neural network," *IEEE Trans. Ind. Electron.*, vol. 70, no. 5, pp. 5192–5202, May 2023, doi: 10.1109/TIE.2022.3189076.

[3] S. Quabeck, W. Shangguan, D. Scharfenstein, and R. W. D. Doncker, "Detection of broken rotor bars in induction machines using machine learning methods," in *Proc. 23rd Int. Conf. Electr. Mach. Syst. (ICEMS)*, Nov. 2020, pp. 620–625, doi: 10.23919/ICEMS50442.2020.9291033.

[4] M. Tucci, S. Barmada, A. Formisano, and D. Thomopulos, "A regularized procedure to generate a deep learning model for topology optimization of electromagnetic devices," *Electronics*, vol. 10, no. 18, p. 2185, Sep. 2021, doi: 10.3390/electronics10182185.

[5] J. Shim, G. C. Lim, and J.-I. Ha, "Unsupervised anomaly detection for electric drives based on variational auto-encoder," in *Proc. IEEE Appl. Power Electron. Conf. Expo. (APEC)*, Mar. 2022, pp. 1703–1708, doi: 10.1109/APEC43599.2022.9773565.

[6] P. Di Barba, "Future trends in optimal design in electromagnetics," *IEEE Trans. Magn.*, vol. 58, no. 9, pp. 1–4, Sep. 2022, doi: 10.1109/TMAG.2022.3164204.

[7] L. Jin, F. Wang, and Q. Yang, "Performance analysis and optimization of permanent magnet synchronous motor based on deep learning," in *Proc. 20th Int. Conf. Electr. Mach. Syst. (ICEMS)*, Aug. 2017, pp. 1–5, doi: 10.1109/ICEMS.2017.8056321.

[8] H. Kurtovic and I. Hahn, "Neural network meta-modeling and optimization of flux switching machines," in *Proc. IEEE Int. Electr. Mach. Drives Conf. (IEMDC)*, May 2019, pp. 629–636, doi: 10.1109/IEMDC.2019.8785344.

[9] S. Cesay, P. Teng, R. Wang, H. Yue, A. Khan, and D. Lowther, "Generalizable DNN based multi-material hysteresis modelling," in *Proc. IEEE 20th Biennial Conf. Electromagn. Field Comput. (CEFC)*, Oct. 2022, pp. 1–2, doi: 10.1109/CEFC55061.2022.9940692.

[10] A. Fatemimoghadam, Y. Yan, L. V. Iyer, and N. C. Kar, "Permanent magnet synchronous motor drive using deep-neural-network-based vector control for electric vehicle applications," in *Proc. Int. Conf. Electr. Mach. (ICEM)*, Sep. 2022, pp. 2358–2364, doi: 10.1109/ICEM51905.2022.9910710.

[11] Y.-B. Yan, J.-N. Liang, T.-F. Sun, J.-P. Geng, Gang-Xie, and D.-J. Pan, "Torque estimation and control of PMSM based on deep learning," in *Proc. 22nd Int. Conf. Electr. Mach. Syst. (ICEMS)*, Aug. 2019, pp. 1–6, doi: 10.1109/ICEMS.2019.8921886.

[12] A. K. A. Talukder, B. Wang, and Y. Sakamoto, "Electric machine two-dimensional flux map prediction with ensemble learning," in *Proc. 25th Int. Conf. Electr. Mach. Syst. (ICEMS)*, Nov. 2022, pp. 1–4, doi: 10.1109/ICEMS56177.2022.9982881.

[13] M. R. Raia, S. Ciceo, F. Chauvicourt, and C. Martis, "Multi-attribute machine learning model for electrical motors performance prediction," *Appl. Sci.*, vol. 13, no. 3, p. 1395, Jan. 2023, doi: 10.3390/app13031395.

[14] Y. Shimizu, S. Morimoto, M. Sanada, and Y. Inoue, "Automatic design system with generative adversarial network and convolutional neural network for optimization design of interior permanent magnet synchronous motor," *IEEE Trans. Energy Convers.*, vol. 38, no. 1, pp. 724–734, Mar. 2023, doi: 10.1109/TEC.2022.3208129.

[15] H. Sato and H. Igarashi, "Fast topology optimization for PM motors using variational autoencoder and neural networks with dropout," *IEEE Trans. Magn.*, vol. 59, no. 5, pp. 1–4, May 2023, doi: 10.1109/TMAG.2023.3242288.

[16] V. Parekh, D. Flore, and S. Schöps, "Deep learning-based prediction of key performance indicators for electrical machines," *IEEE Access*, vol. 9, pp. 21786–21797, 2021, doi: 10.1109/ACCESS.2021.3053856.

[17] V. Parekh, D. Flore, and S. Schöps, "Variational autoencoder-based metamodeling for multi-objective topology optimization of electrical machines," *IEEE Trans. Magn.*, vol. 58, no. 9, pp. 1–4, Sep. 2022, doi: 10.1109/TMAG.2022.3163972.

[18] S. J. Salon, *Finite Element Analysis of Electrical Machines*. Norwell, MA, USA: Kluwer, 1995.

[19] D. Gerling, *Induction Machines*. Berlin, Germany: Springer, 2015, pp. 135–188, doi: 10.1007/978-3-642-17584-8_4.

[20] I. Boldea and S. A. Nasar, *The Induction Machines Design Handbook*, 2nd ed. Boca Raton, FL, USA: CRC Press, 2018, doi: 10.1201/9781315222592.

[21] M. McKay, R. Beckkman, and W. Conover, "Comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 21, pp. 266–294, Jan. 2000, doi: 10.1080/00401706.2000.10485979.

[22] C. Geuzaine and J.-F. Remacle, "Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities," *Int. J. Numer. Methods Eng.*, vol. 79, no. 11, pp. 1309–1331, Sep. 2009.

[23] D. P. Kingma and M. Welling, "An introduction to variational autoencoders," *Found. Trends Mach. Learn.*, vol. 12, no. 4, pp. 307–392, 2019, doi: 10.1561/2200000056.

[24] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–14.

[25] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org

[26] A. Amini and A. Amini, *MIT 6.s191: Introduction to Deep Learning*. Cambridge, MA, USA: Massachusetts Institute of Technology, Lecture Note 2021. [Online]. Available: https://introtodeeplearning.com/
[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.
[28] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002, doi: 10.1109/4235.996017.
[29] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015, pp. 1–15.

VIVEK PAREKH received the bachelor's degree in electronics and communication from Gujarat Technological University, in 2012, and the M.Sc. degree in information technology from the University of Stuttgart, in 2019. He is currently pursuing the Ph.D. degree with the Institut für Teilchenbeschleunigung und Elektromagnetische Felder, Technische Universität Darmstadt (TU Darmstadt), under the fellowship of Robert Bosch GmbH. From 2013 to 2015, he was the Assistant Manager with Reliance Jio Infocomm Ltd., in operation and maintenance. His research interests include machine learning, deep learning, reinforcement learning, optimization of electrical machine design, and industrial simulation process.

DOMINIK FLORE received the bachelor's degree in maschinenbau with specialization in mechatronik and the master's degree in maschinenbau with specialization in product development from the University of Paderborn, in 2012 and 2013, respectively, and the Ph.D. degree from ETH Zürich, in 2016, with the topic of "Experimentelle Untersuchung und Modellierung des Schädigungsverhaltensfaserverstärkter Kunststoffe." He is currently a Development Engineer for the reliability of the electric machine with Robert Bosch GmbH, Stuttgart. His research interests include product development in the field of electrical machines with the application of machine learning and optimization of the industrial simulation process.

SEBASTIAN SCHÖPS received the M.Sc. degree in business mathematics and the joint Ph.D. degree in mathematics and physics from Bergische Universität Wuppertal and Katholieke Universiteit Leuven. He was a Professor of computational electromagnetics with the Interdisciplinary Centre of Computational Engineering, Technische Universität Darmstadt, in 2012. His research interests include coupled multiphysical problems, bridging computer-aided design and simulation, parallel algorithms for high-performance computing, digital twins, uncertainty quantification, and machine learning.